



Using Sound in Macromedia Flash

by Jeff Essex

Sponsored by Apple Computer, Inc.
Apple Developer Connection

media

Using Sound in Macromedia Flash

by Jeff Essex



If you're creating multimedia content for the Web, Flash and Shockwave are likely to be part of your daily life. This article explores the use of sound in Flash presentations. It won't serve as a basic tutorial: you can get that from the documentation. Instead, I'll cover some tricks and workarounds that may come in handy on your projects.

This article assumes you have a basic working knowledge of audio file formats (AIFF, WAV, etc.) and understand a bit about sample rates and sample resolutions (8-bit/22 kHz vs. 16-bit /44.1 kHz, etc.) If you need help with multimedia audio basics, you may want to skip to the Resources section at the end of this article for more information.

Flash Audio - A Quick Review

Flash is a vector-based drawing and animation program that's popular among web designers because it packs lots of graphic content into a small file. Flash also has basic interactivity and the ability to play sounds (AIFF on Macintosh, WAV on Windows). The greatest strength of Flash's audio support is its ability to reuse the same sound data in different ways. This is tremendously helpful in reducing file size, and with a bit of creativity, can produce great soundtracks. There are three areas to focus on when optimizing audio in Flash:

- [Tweaking waveforms in the Frame Properties Dialog](#)
- [Setting Export options for each sound in a Library](#)
- [Designing a palette of sounds and layering them in the Timeline](#)

Before looking at each of these in more detail, here's a quick list of the steps for adding sound to a Flash document:

1. Import a sound into the Library (File > Import)
2. Create a new layer in the timeline (Insert > Layer - layers are similar to sprite channels in Director)
3. Add a keyframe within that layer (Insert > Keyframe)
4. Open the Frame Properties dialog (double-click the keyframe)
5. Assign a sound to the frame (click the Sound tab and choose a sound from the pop-up menu)

You can also drag and drop sounds from the Library onto the Stage, but only if the active layer already contains a keyframe. Flash will place the sound at the nearest previous keyframe in that layer.



Tweaking in the Frame Properties Dialog

The Frame Properties Dialog lets you change the properties of sounds in a surprising number of ways:

- Envelope control handles give you precise control of fade-in and fade-out, as well as volume level. Even better, the envelopes can vary the position of monaural sounds between the Left and Right channels of a stereo field. You get the impact of stereo, but save half the file size by using mono.
- The Time In and Time Out controls let you change where playback of the sound will start or stop. Combine this with the volume envelopes, and it becomes easy to squeeze a variety of sound effects from one file.

You can set a sound to loop a specified number of times. Even better, you can tweak the envelopes over the course of the entire sequence of loops (see Figure 1 below).

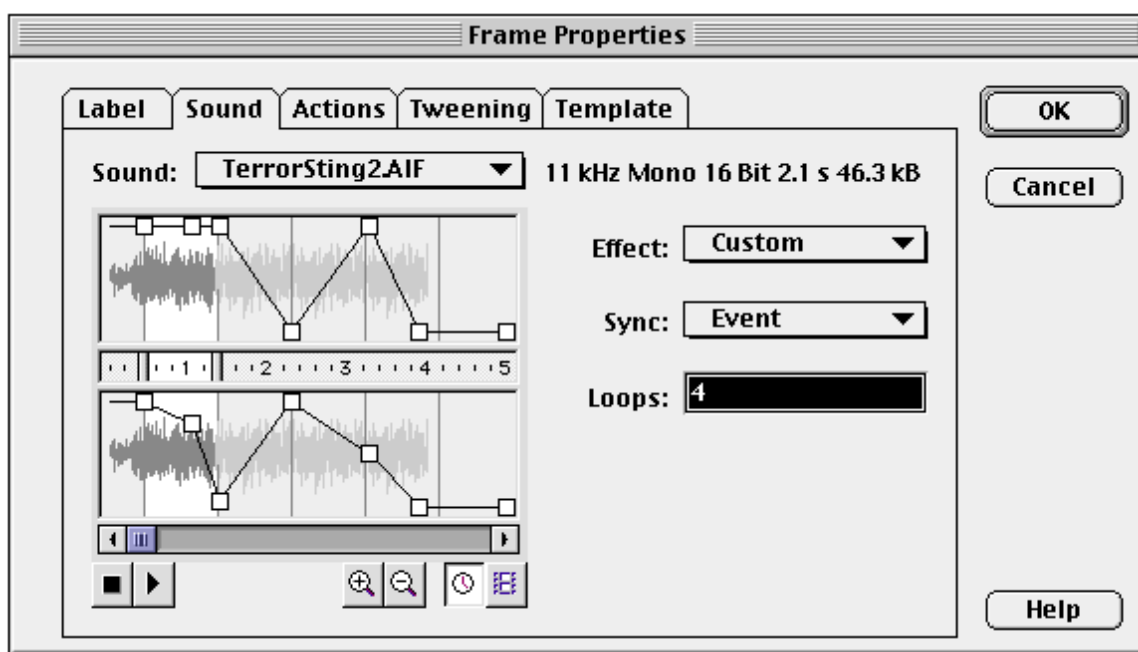


Figure 1. The Frame Properties Dialog

Here's an example of envelopes affecting a looped sound. The sound has been set to loop four times, and the Envelope has been set to pan the sound between channels then fade out. [Click here](#) to hear the example.

When possible, avoid using the Time In and Time Out controls to crop "silence" from the beginning and ending of files. Instead, use an audio editing program like Sound Edit 16, Peak or Sound Forge to remove any leading or trailing silences, and to verify that the audio is cropped as close to the ends as possible to reduce file size. The waveform display in Flash is slightly inaccurate. What looks like a flat line of silence in a Flash waveform may actually contain audio content.

On the plus side, Flash only exports the audio that's between the Time In and Time Out controls. This can be handy for quick and dirty edits on sound files. Still, it's best to edit sounds before importing.

The Frame Properties Dialog also lets you select one of four options for synchronizing sound with animation:



- *Event* sync triggers sounds at a specific keyframe. This is useful for button clicks and sound effects that are related to animation. Event sounds always play through to the end of the sound, unless specifically stopped by the *Stop* option (see below) or a *Stop all Sounds* Frame action. You can play multiple instances of an Event sound by retriggering the sound in subsequent keyframes. For example, imagine the “whoosh” sound of numerous cars driving by. An Event sound lets you use the same sound multiple times, without cutting off the previous use of the sound.
- *Stream* sync locks the animation to the sound, forcing Flash to drop frames if the picture falls behind the sound. This is useful for locking longer sequences of sound and animation. Don’t be fooled by the name, though. Stream sounds don’t actually stream the audio across an Internet connection in the same way that QuickTime, RealAudio and Shockwave Audio provide streaming. Instead, think of Stream sounds as providing a QuickTime-style ability to lock the audio and animation into a synchronized stream of media. A major drawback of using Stream sync is that Flash doesn’t reuse the audio. For example, if you place two instances of a 20-second Stream sound in the Timeline, Flash will create 40 seconds worth of data when saving the final playback version of the file. The flip side of this behavior, however, can be a real bonus. Let’s say you’re creating a complex mix of music, voice and sound effects that lasts for one minute. With Stream sounds, you can do whatever you want for that minute: no need to reuse the same sounds over and over, or have looping music or ambience. Once you’ve paid the price of admission (i.e. larger file size), you can layer all the sounds you want, since the result will be mixed to a single audio stream. In some cases, if you’re using a whole bunch of event sounds, you might actually save space by switching to Stream sounds.
- *Start* works like an Event sound, but if a sound is already playing, it won’t retrigger a new instance of the sound. **Look out!** This is incorrectly described in the Flash 3.0 documentation, but correct in the Flash 2.0 docs. *Start* can be useful if you want playback to jump to another frame, but not interrupt the sound. Triggering two instances of the same voice or music file would result in simultaneous playback of two copies of the file (great for creating crowd scenes or avant garde music, but not much help otherwise!).
- *Stop* cuts off an Event sound at a specific keyframe. This is handy for cutting off long Event sounds when a user action forces playback to jump to a new location, or when a long or looping sound needs to stop at a specific point to match the animation. Used wisely, Stop can greatly improve the synchronization of sound with animation, since Flash’s frame rate can vary due to processor speeds, animation complexity, modem speeds, and the amount of Web traffic at any given time.

In most situations, Event sync is the most common setting. Event sync makes the most of Flash’s ability to repurpose sounds with a variety of Frame Property settings, so use it whenever possible. Also, the functionality of Stream sync indicates that it could be most useful for forcing synchronization between animation and long segments of dialog or music. In practice, however, Flash’s audio compression schemes compromise the quality of the audio to such an extent that it’s wise to avoid using Flash as a delivery strategy for long segments of dialog or music over standard Web connections (for workarounds in Flash, see the section below on [Shockwave Audio and the Flash Asset Xtra](#)). If you’re planning on delivering Flash content over an Intranet or other local delivery mechanism, you can get away with using the higher quality audio compression options. To get a better grasp on this issue, let’s explore compression options in Flash.



The Sound Properties Dialog - Making or Breaking your Audio

Flash uses two techniques to reduce the size of audio data in presentations as they are exported to the Shockwave Flash format for delivery on the web: *downsampling* and *ADPCM compression*. These settings are applied in the Sound Properties dialog. You can improve the perceived quality of your Flash audio by understanding and working within the limitations of these audio processes. Or, to look at it from the negative angle, applying the wrong settings can leave you with horrible soundtracks.

Downsampling makes files smaller by reducing the number of samples per second. For example, the standard sample rate for audio CDs is 44,100 samples per second (i.e. 44.1 kHz). Flash can downsample to rates of 22, 11 and 5.5 kHz. Higher sample rates allow for more high-frequency content to be represented in a sound file. Here are some general guidelines for comparing the quality of various sample rates:

- 44.1 kHz sounds great, just like CDs
- 22 kHz is fine for most multimedia applications, similar to FM radio
- 11 kHz offers a heavily muffled but intelligible representation of a sound
- 5.5 kHz can convey bonks, thumps, and other low frequency sounds, but not much else

Click on any of the sample rates to hear an example of how the rates affect a file containing voice mixed with music.

You can also choose between stereo or mono for each sample rate. In almost all cases, you'll probably want to choose mono to save space. Stereo could be useful for music, but it will double the size of the audio data. Instead, try to create stereo effects using the Envelope handles, as described in the previous section.

ADPCM compression makes the file smaller by reducing the number of bits (ones or zeroes) used to store the data. ADPCM uses mathematical formulas to reduce the number of bits required in a signal. Most multimedia developers are familiar with 4:1 ADPCM compression schemes like QuickTime's IMA, or Microsoft's MS-ADPCM. When these 4:1 compressors are applied to a 16-bit sound (the standard for audio CDs), the result is a 4-bit sound. Keep in mind that, thanks to the use of ADPCM, this 4-bit compressed audio still sounds much better than the old uncompressed 8-bit standard that many of us were familiar with in years past. Flash provides ADPCM compression from 2-bit to 5-bit (i.e. from 8:1 to roughly 3:1). Higher bit rates sound better. As you move toward lower bit rates, distortion becomes more pronounced. At the bottom end, 2-bit compression sounds like a scratchy old record.

It's hard to overemphasize the importance of using the Sound Properties dialog to apply custom sample rate and ADPCM compression settings to each sound in your presentation. DON'T export all your sounds using the Default rate pop-up menus in the Export dialog. You should ALWAYS audition sounds in the Sound Properties dialog, and experiment with compression settings and sample rates to get the best results for each individual sound. There will be tradeoffs between size versus quality, and you'll have to use your ears to judge which settings are appropriate for which sounds. Using the Default export values practically guarantees that some of your sounds will be horribly compromised, while others may take more space than necessary.



To apply a specific sample rate and ADPCM compression setting:

1. Double-click a sound in the Library window to bring up the Sound Properties dialog.
2. Choose a sample rate and bit depth (11 kHz mono, 3-bit is a good place to start)
3. Click the "Play" button to audition the sound.
4. Does it make you cringe? If so, increase the sample rate or bit depth. If not, try reducing them further.

Watch out when auditioning sounds! You won't really hear the effect of any sample-rate changes or ADPCM compression when the sound plays in the timeline, or when you play it using the Play button found at the top of the Library window. You also won't hear the effects if you leave the "Default" settings active in the Sound Properties dialog - *even if you previously chose settings when exporting a movie!* The ONLY way to know how your audio will sound when it's played back in a browser is to apply settings in the Sound Properties dialog, and preview the sound in that dialog.

Experiment with nudging the sample rates and bit depths up and down, and test the result. If a sound seems too muted at 5.5 kHz and 3-bit ADPCM, it might sound better at 11 kHz and 2-bit ADPCM, and the result won't be too much larger. Increasing the sample rate can also significantly reduce distortion at a particular bit depth. Keep in mind that setting the sample rate higher than the rate of the original source file won't improve quality. Once a file has been downsampled, the extra audio data is gone for good, unless you go back to the master source files (you *are* archiving your source files at 16-bit, 44.1 kHz, right?). The chart below (Figure 2) can help you determine which combination of settings will result in the smallest file.

Sample Rate	# Bits	kBits/second	kBytes/second
44.1	5	220.5	27.563
44.1	4	176.4	22.050
44.1	3	132.3	16.538
44.1	2	88.2	11.025
22.05	5	110.3	13.781
22.05	4	88.2	11.025
22.05	3	66.2	8.269
22.05	2	44.1	5.513
11.025	5	55.1	6.891
11.025	4	44.1	5.513
11.025	3	33.1	4.134
11.025	2	22.1	2.756
5.5	5	27.5	3.438
5.5	4	22.0	2.750
5.5	3	16.5	2.063
5.5	2	11.0	1.375

Figure 2. File sizes at various sample rates and ADPCM bit depths.

Here are some tips for setting export options for sounds:

- Higher compression ratios (4- or 5-bit) are better for longer sounds. With shorter sounds (button sounds and short sound effects), it's easier to get away with lower ratios (2- or 3-bit), since the listener has less of an opportunity to perceive the distortion. For example, there might be one or two distorted glitches in a button sound, but the sound plays so quickly that the listener is likely to think that the glitches were actually part of the sound. Longer sounds (music, voice or ambient effects like wind or city traffic) give the listener more of an opportunity to perceive how the audio *should* sound, and allows more time for crackling distortion to become apparent.
- Lower compression ratios are more effective on sounds with very abrupt or chaotic waveforms (like crackling fires, crashes, explosions or the clickety-clack of train wheels), and are less effective on sounds with consistent, predictable, "smooth" waveforms (like wind, waves, or musical tones). Sounds that a listener would reasonably expect to be loud or distorted are perfect candidates for 2-bit ADPCM.
- Don't use 8-bit samples instead of 16-bit samples to save space. The ADPCM compression routines work from 16-bit data. If you import 8-bit samples, they'll just be converted to 16 bits before compression is applied. All the quantization noise and distortion found in 8-bit samples will be carried into the Flash file.
- Try to use higher sample rates when it's important to retain high frequency content (bells, etc). Be especially careful when applying low sample rates to these types of sounds, as it can create strange aliasing effects. This was particularly problematic in Flash 2.0, and while it's improved in Flash 3.0, the aliasing artifacts still exist. If you know that a sound is destined to be set to sample rates of 5.5 kHz or 11 kHz, you can often get better results by downsampling the audio in a separate audio editing program (like [SoundEdit 16](#) or [BIAS Peak](#)), then importing the result into Flash.
- When creating lots of audio for larger projects, consider creating a separate Library of sounds that have your preferred sample rates and bit depths applied to them. Give the Library file to the artists and programmers to assure that sounds will be exported with the proper settings. Of course, this assumes that the "Override sound settings" option isn't selected in the Export dialog.



Don't forget the distinction between settings applied to sounds in the Library (i.e. sample rate and bit depth for the exported sound data) vs. settings applied to sounds in the Timeline (instances of a sound from the Library, with different start and stop points and volume envelopes for each instance). The sample rate and bit-depth options selected in the Sound Properties dialog are applied to every instance of that sound in the Timeline. To make an analogy with Flash graphics, sounds behave like Symbols in the Timeline : they're both instances of objects from the Library, and can have multiple attributes applied to them at different points in the Timeline. Export settings, on the other hand, have a similar effect with both sounds and JPEG images in the Library. They can have multiple instances in the Timeline, but compression settings applied to the "parent" in the Library are applied to all the "child" instances in the Timeline.

Design Strategies

It's a big challenge to pack a compelling multimedia experience into a small file. Flash movies can be incredibly compact, thanks to the efficiency of vector graphics. Add a bit of audio, on the other hand, and your file size can quickly get out of control. In Flash this sometimes leads to one of three results: (a) reuse the same piece of audio over and over until the repetition drives people crazy; (b) compress and downsample the audio until it sounds atrocious; or (c) after experiencing #1 and #2 several times, avoid using audio altogether.

Luckily, Flash provides enough flexibility that, with a bit of creative design, you can get a lot of mileage out of just a few well-chosen sounds. Flash lets you reuse a sound as many times as you like, but it only has to download once

(unless you're using Stream sync, as described earlier). Once you've paid the price for that download, it's in your best interests to figure out how many times the sound can be used in an interesting way. By combining Flash's ability to mix sound layers in the Timeline, and the ability to apply different start/stop points and volume envelopes to each instance of a sound, you can create some pretty compelling effects. The combined ability of these two methods is not too far removed from the capabilities of multi-track audio editing and mixing programs like SoundEdit 16 or Deck. In fact, it puts Flash far ahead of Macromedia Director! Here are some ideas:



Layering

Create subtle variations in the tone of a sound effect by layering a short quiet sound "beneath" the main effect. For example, Figure 3 shows a tiny blip created from a section of voiceover.

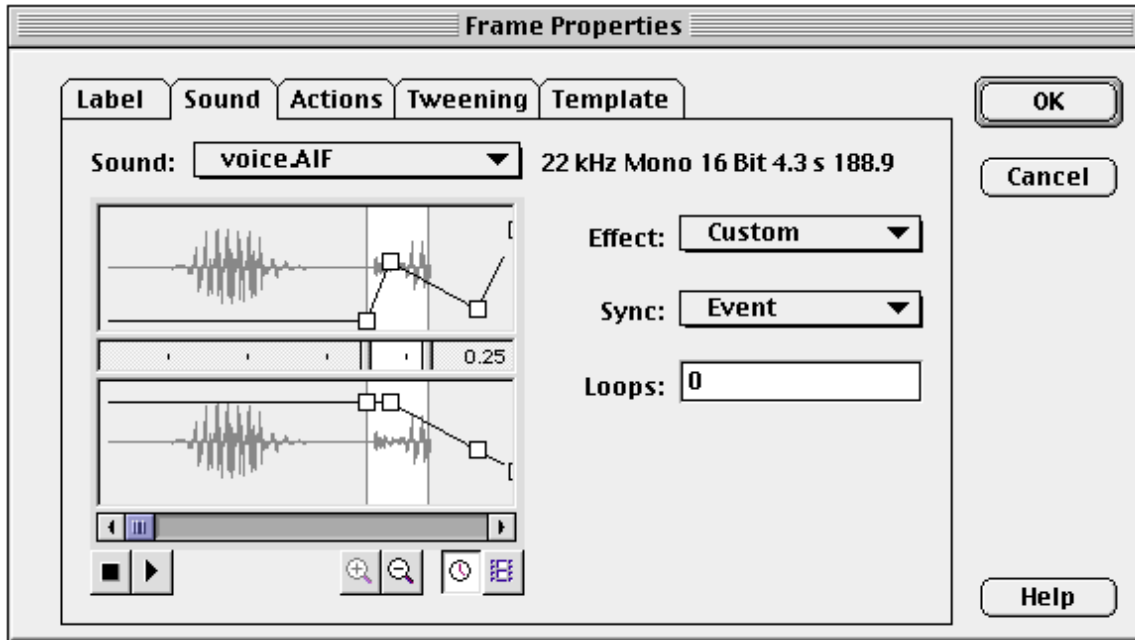


Figure 3. A tiny section of voiceover or other audio content adds depth to a [button click sound](#).

Click one of the following to hear a sample of vocal button clicks: [vox1](#), [vox2](#), [vox3](#), [vox4](#)

Playing two or three of these very quiet effects, layered with a prominent button click sound, creates a small suite of related sounds that would be much more interesting to listen to than a simple "click." Practically any sound will do for an application like this. If you're drawing from voice as a source, try to find sounds created by pronouncing the letters B, K, T, P for percussive effects, and use S or Z to add a brighter tone. Pan the sound to one side of the stereo field to enhance the perception of depth that the layer adds to the original sound

Here's another example of the layering technique. Everyone knows that in the real world, we rarely hear the exact same sound twice. Think of footsteps crossing a room, someone knocking on a door, or objects falling to the ground with a "plop". There's always a bit of variation in the real world, and when we hear the exact same sound repeatedly, it reinforces the realization that we're hearing something canned and artificial. Working with the example of footsteps, imagine starting off with four distinct footstep samples, all similar, but with slightly different tonal qualities. If we just loop the four footsteps over and over, it would sound pretty dull: 1-2-3-4-1-2-3-4, etc., ad nauseum. But with layering, we can extend the depth of the sound without making the file any bigger, for example:

Foreground sound layer: 1 -2 -3 -4 -1 -2 -3 -4 -1 -2 -3 - 4 ([click here](#) to hear a static, repeated loop)
 Background sound layer: 3- 4 -1 -2- 4 -3 -1 - 2 ([click here](#) to hear a layered loop)



Taking this a “step” further, altering the volume level of each instance of either layered sound will create a slightly different tone to the resulting mix of sounds. To borrow a graphic metaphor, think of it as varying the transparency levels or brightness of two superimposed colors. Each variation creates a slightly different shade.

Retrigger and Layer Wisely

If you need to create a looping ambience for a scene (i.e. waves, a rushing river, wind or city traffic), don’t just loop the same sample over and over. Create several “instances” of the ambience on separate layers. Give each one a different start and stop time. Let some of them fade in or out quickly, and have others fade in or out slowly. Copy them out several times across the two layers, and adjust their start times to create a continuous ambient soundtrack. Take advantage of Flash’s ability to retrigger a sound multiple times in the same layer to create variety. The *Stop All Sounds* Frame action can come in handy here, when you need to stop all the sounds at once.

Create a Sound Palette

Since Flash is so accommodating in reusing sounds, it pays to spend some extra time considering which sounds are most useful as raw material for slicing into tiny segments and layering against other sounds. Look for sounds that have a lot of sonic variations in a short period of time. Try to find a balance of sounds across the entire spectrum: some with high frequencies or lots of sizzle, and others with plenty of low frequency and “kick.” This gives you more options for altering tones. For example, I once worked on an animation that included lots of impacts, chewing, crunching bones, and rustling plants. I was able to tweak one sound (the crackling of a falling tree) into all of these effects through careful use of layering and volume control.

Work With Distortion

If you’re likely to end up with distortion, why not use it to your advantage? For example, the effect of a car crash or falling tree often seems lifeless when brought down to a sample rate of 5.5 kHz. The downsampling removes high frequencies from the signal, so we don’t hear the crack of branches or the crunch of breaking glass. The distortion created by 2-bit ADPCM actually helps these kinds of effects to sound more “natural” because it adds the snap and crackle back into the signal. If you’re going to use music samples, maybe you should go out of your way to find one with lots of distorted guitar!

Spend Your Budget Wisely

It can pay off to use higher quality settings for sounds that are used all the time. For example, the perceived quality of your audio may go up quite a lot if you use a setting of 22 kHz and 2-bit ADPCM for a very quick mouseclick. This will preserve the crisp high frequencies of the signal, while the distortion created by the 2-bit compression is likely to get lost in the sound of the click. If users are going to be generating lots of mouseclick sounds, it could be worth using a few extra bytes.

Build Complex Sequences with Loops and Stop Sync

You can achieve great synchronization between effects and animation using loops and Stop sync. Here’s a tip from Dave Hollinden, Audio Engineer at [Jet City Studios](#) (a fabulous group of Flash developers in Seattle). Dave wanted to create the effect of a fanfare, followed by a drum roll, followed by a rimshot when the user clicked a button. Launching the fanfare was easy - just add the sound to a layer. Launching the drum roll was easy, too - create a short drum roll loop and set it to play 100 times. The nice trick was responding to the user’s mouseclick. Playback jumped to another frame with a “Stop” sync event applied to the drum roll, while the rimshot was triggered in another layer.

Flash "Synthesis"

While we're on the subject of loops, you can actually use Flash as a synthesizer that generates tones from looping. Any looped waveform plays at a pitch which is determined by the length of the waveform: the shorter the loop, the higher the pitch of the tone. Set the Start and Stop points as close together as you can manage, then crank the number of loops up to 15 or more. Remember, you can adjust the volume Envelopes across the entire series of loops, so the tone can fade in or out as well as pan across a stereo field. For an example, see Figure 4.

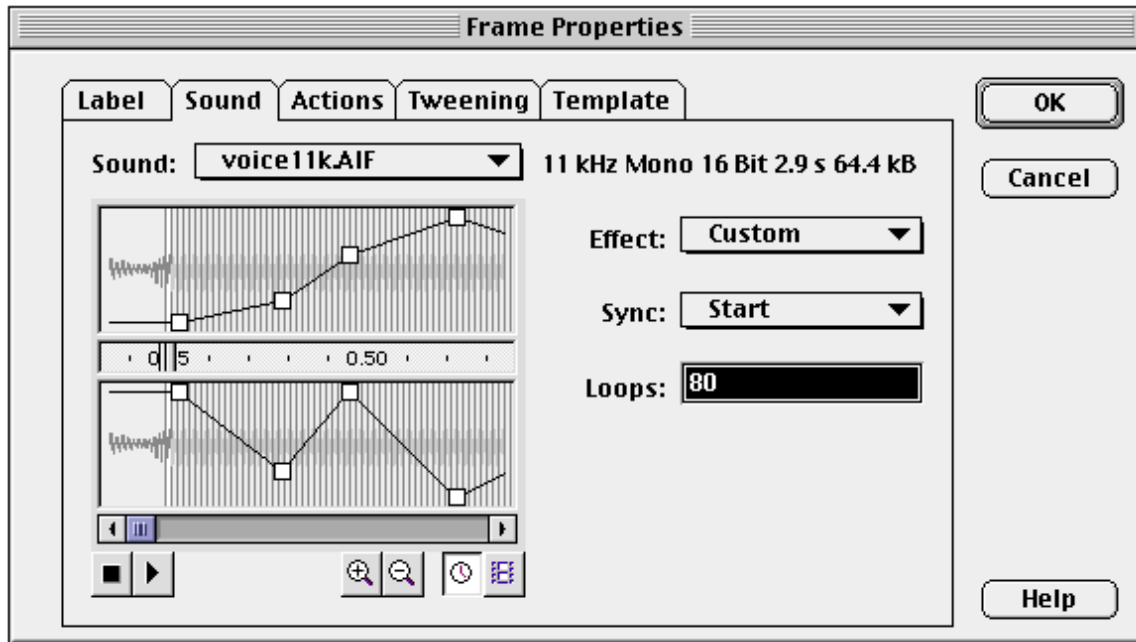


Figure 4. Flash "synthesis" can be created using tiny looped sections of a waveform.

[Click here](#) to hear the sample above, or [click here](#) for a different sample.

Cheesy Elvis Reverb

Ever hear the "slapback" delay on old Elvis records? You can add a similar type of ambience to an effect by retriggering it in multiple frames. Recall that sounds played with Event sync play all the way through to the end, and retriggering that sound in another frame will actually launch a new instance of the sound instead of cutting the existing sound off. Use this technique to create several "echoes" of a sound, and experiment with the placement of the sounds to build echoing effects ranging from small concrete chambers to large canyons.

Use Director and the Flash Asset Xtra to Play Shockwave Audio

While Flash's audio compression isn't state of the art, you can take a roundabout route to enjoy the superior performance of Shockwave Audio (SWA), a variant of MP3 that's built into Shockwave for Director. The Flash Asset Xtra lets you play Flash movies from within Director (the Flash movie can either be linked externally to the Director movie, or embedded right into Director's cast). You can send Lingo commands from Flash to Director by embedding the text of the Lingo command in the GetURL portion of Flash's Instance Properties or Frame Properties dialog. In this case, sending a Lingo puppetsound command at a particular frame or button action could trigger playback of a SWA file. The exact steps are described in more detail in the Director documentation. This approach may not be feasible in some situations. Designers may prefer to stay entirely in the Flash environment, and use the smaller Flash plug-in instead of the larger Director Shockwave plug-in. Complex presentations may suffer a bit from the extra overhead of playing Flash within Director. But when audio quality is at a premium, the ability to play SWA could be a helpful alternative.

You may be wondering, "Why didn't Macromedia use SWA in Flash instead of downsampling and ADPCM?" Apparently, the amount of code required for SWA would double the size of the Flash plug-in. Macromedia is reluctant to make users wait twice as long to download the plug-in just to boost the audio quality. But they are committed to improving audio performance in future versions of Flash. We'll have to wait and see.



Using RealFlash

Another option for playing Flash content is the RealFlash player from RealNetworks. Developed jointly by Macromedia and RealNetworks, RealFlash converts Flash animations and soundtracks into linear, streaming animations with synchronized sound. Here's one place where Flash's Stream sync option is very useful, since it provides the same type of linear, synchronized playback that one can expect when watching a RealFlash movie. Stream sync makes it possible to tweak a soundtrack to fit animation, and be sure of a similar result once the resulting RealFlash movie has been compiled. The one drawback of RealFlash files is that audio quality is poor on slower modem connections like 28.8. While traditional RealAudio files can use all the bandwidth of the modem, RealFlash movies have to split the bandwidth between graphics and audio. For more information on creating RealFlash, check out the Resources section below.

Summary

Flash has the potential to create compelling soundtracks if you take the time to become familiar with its capabilities. There are three key areas for tapping this potential: 1) creating multiple "instances" of sounds, with unique envelopes and start/stop points for each instance; 2) setting Export options for each sound; and 3) layering multiple sounds in the Timeline. You have to be careful with Flash's audio compression scheme, but even this weak spot can be used to your advantage in some situations. I hope the techniques in this article will give you some new ideas when the time comes for you to set out on your next web adventure with Flash.

Resources

Macromedia	http://www.macromedia.com
Flash Central	http://www.flashcentral.com
RealFlash	http://www.real.com/devzone/library/creating/flash
Sound Forge	http://www.sonicfoundry.com
BIAS Peak and Deck II	http://www.bias-inc.com

For more information on multimedia audio basics, check out my book, "[Multimedia Sound and Music Studio](#)." Special thanks to Dave Hollinden at [Jet City Studios](#) for a rollicking round of tip-swapping!

About the Author

Jeff Essex, Creative Director of [audiosyncrasy](#), specializes in digital audio and MIDI to create music, sound effects and voiceover for multimedia. In addition to his creative and audio engineering skills, he is intimately familiar with multimedia tools and technology. A veteran of MacroMind and Macromedia, he served two years as Technical Support Lead for Macromedia Director and SoundEdit. He is credited on over 40 CD-ROM titles, including products from Virgin Sound and Vision, Corbis Productions, Mindscape, 3DO, and Disney Interactive. For the past year he has worked as the primary composer and audio consultant for the leading children's interactive web site. His book, [Multimedia Sound and Music Studio](#), (Random House/Apple New Media Library, 1996) is the definitive guide to multimedia audio production. It won the Computer Press Award for Best Advanced How-To Book of 1996.