

## 4th May 2010 Reproducir Sonidos en ActionScript 3.0

Cuando reproducimos sonido en la web, la idea que debemos tener clara, es que **normalmente el archivo de sonido se cargará externamente** ( nunca lo incorporaremos a la línea de tiempo, salvo que sea un efecto de sonido muy corto, y que pese poco ).

Al cargarse externamente, el archivo de audio **se reproducirá por descarga progresiva**, es decir que no hará falta que se haya descargado totalmente el archivo para que podamos empezar a escucharlo. **El formato será mp3.**

En la última versión de ActionScript, el tema del sonido se ha modificado bastante. Los que recuerden AS 2.0 estarán familiarizados con **la clase Sound**, que era la que nos permitía hacer todas las operaciones básicas con sonidos.

En la última versión de ActionScript, **se han creado varias clases que complementan a la clase Sound**. Por un lado esto hace que sea más complicado reproducir un archivo de audio, pero por el otro nos da muchas más posibilidades, como por ejemplo hacer un espectro de sonido, cosa impensable en la versión anterior de ActionScript

leer      más...      [http://desarrolloparaweb.blogspot.com/2010/05/reproducir-sonidos-en-actionscript-30.html]

### Las clases relacionadas con el sonido

Los sonidos en AS 3.0 se manipulan mediante varias clases que trabajará juntas, esto nos dará mayor control y habilidad para manejar sonidos. Las clases más relevantes para trabajar con sonido son:

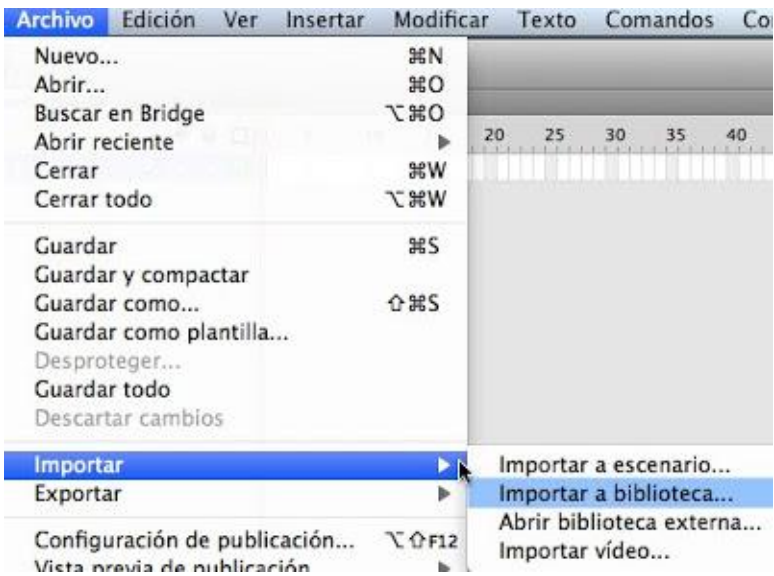
- **Sound:** es la clase principal y el punto de partida para reproducir un archivo de audio.
- **SoundChannel:** un archivo de audio se podrá reproducir a través de esta clase, lo que nos permitirá proporcionar controles adicionales, como por ej. detener y continuar la reproducción
- **SoundTransform:** esta clase se utiliza para controlar el sonido y el balance ( propiedad pan )
- **SoundMixer:** esta clase tiene un control global sobre todos los sonidos que se reproducen en flash. Su función más básica es la de detener todos los sonidos independientemente de la fuente de donde vengan.
- **SoundLoaderContext:** básicamente tiene la función de establecer el "buffer time" o número de milisegundos que se tarda en precargar un flujo de sonido en un buffer

A continuación veremos ejemplos para cargar y reproducir un sonido.

### Reproducir un sonido importado en Flash

Aunque no es lo recomendable, puede haber casos en los que nos interese importar un sonido aunque estemos trabajando para web ( por ejemplo un efecto de sonido corto, un sonido de disparo para un juego, una explosión, etc ), y puede haber otros casos en los que estemos desarrollando un cd y nos interese incorporarlo en la línea de tiempo para que no se pueda tener acceso a él desde fuera.

Lo primero que haremos es importar el sonido a la bibliotecca



[http://4.bp.blogspot.com/\_G62iQFG\_F58/S-BAaIeV95I/AAAAAAAAABa4/fs0Orpqblp4/s1600/importar.jpg]

Una vez importado, nos aparecerá en la biblioteca, y podremos reproducirlo y ver el espectro de sonido



[[http://3.bp.blogspot.com/\\_G62iQFG\\_F58/S-BBHITSqUI/AAAAAAAAABbA/\\_Qp01mTRwas/s1600/biblioteca.jpg](http://3.bp.blogspot.com/_G62iQFG_F58/S-BBHITSqUI/AAAAAAAAABbA/_Qp01mTRwas/s1600/biblioteca.jpg)]

Si hacemos click con el botón derecho, accederemos a un desplegable; hacemos click en la última opción que es propiedades, y nos aparecerá un cuadro de menú como el siguiente:



[[http://4.bp.blogspot.com/\\_G62iQFG\\_F58/S-BB66cq-MI/AAAAAAAAABbI/ukntAMntgcg/s1600/props.jpg](http://4.bp.blogspot.com/_G62iQFG_F58/S-BB66cq-MI/AAAAAAAAABbI/ukntAMntgcg/s1600/props.jpg)]

Igual que para importar otros objetos desde la biblioteca, activamos la opción **exportar para ActionScript**, y ponemos un **nombre de clase**. Este nombre nos servirá para crear instancias de ese sonido. Si nos fijamos la clase base seguirá siendo **flash.media.Sound**. Al darle a aceptar nos saldrá un mensaje de advertencia que simplemente nos informa de que va a crear la clase MiSonido, le damos a ok

```
var so:Sound = new MiSonido();
so.play();
```

Si nos fijamos al crear una instancia, la sintaxis varía un poco respecto a la regla general, en la parte de la izquierda indicamos como tipo de datos **Sound**, pero en la derecha especificamos el nombre de nuestra clase, en éste caso **MiSonido()**;

Reproducir un archivo de audio externo

En muchos casos será la opción más apropiada para la web. Lo que haremos es cargar el sonido mediante el método **load** que incorpora la clase **Sound**.

```
var mySound:Sound = new Sound();
```

```
var req:URLRequest = new URLRequest("elfari.mp3");
mySound.load(req);
mySound.play();
```

Como en todos los casos en los que hacemos un load, la ruta la debemos incluir en una instancia del objeto URLRequest ( la única excepción es la propiedad source de algunos componentes, que nos permite poner directamente la ruta ).

Detener un sonido

Esta sería otra de las tareas básicas a la hora de trabajar con audio, pues bien, la clase Sound no tiene ningún método para detener un sonido.

Deberemos usar la clase SoundChannel para hacer un stop(). Los métodos y propiedades más interesantes de esta clase son:

- **stop( )**
- **position( )**: nos da el tiempo de reproducción en el que se encuentra
- **soundTransform**: la usaremos para poder manipular el volumen y el balance

Por lo tanto, ahora modificaremos un poco el código anterior, para dar cabida al nuevo objeto:

```
var mySound:Sound = new Sound();
var myChannel:SoundChannel = new SoundChannel();
var req:URLRequest = new URLRequest("Thunderstruck.mp3");
mySound.load(req);
myChannel = mySound.play();
```

La instancia de SoundChannel se crea igual que cualquier otra clase. Lo más importante en éste caso es la última línea, en la que en vez de poner mySoundPlay, éste método lo incorporamos dentro de la instancia que hemos creado de SoundChannel.

Ahora para detener el sonido, haremos como si el sonido fuese la instancia de SoundChannel

```
function pararSonido(e:MouseEvent):void{
    myChannel.stop();
}
parar_btn.addEventListener(MouseEvent.CLICK, pararSonido );
```

y para volver a reproducir, simplemente tendremos que volver a ejecutar la línea en la que ejectuábamos el método play, pero dentro de una función para asignársela a un botón.

```
function reproducir(e:MouseEvent):void{
    myChannel = mySound.play();
}
reproducir_btn.addEventListener(MouseEvent.CLICK, reproducir );
```

Pero como podemos ver, el problema es que siempre que detenemos un sonido, vuelve al principio

Pausar un sonido

Para solucionar este tema, utilizaremos **la propiedad position del objeto SoundChannel** y la guardaremos en una variable. Cuando ejecutamos el **método play**, como **primer parámetro** le podemos indicar **el punto donde queremos que empiece la reproducción**.

```
var mySound:Sound = new Sound();
var myChannel:SoundChannel = new SoundChannel();
var req:URLRequest = new URLRequest("Thunderstruck.mp3");
mySound.load(req);
myChannel = mySound.play();
```

```
var posicion:Number = 0;
function pausar(e:MouseEvent):void{
    posicion = myChannel.position;
    myChannel.stop();

}
parar_btn.addEventListener(MouseEvent.CLICK, pausar );
```

```
function reproducir(e:MouseEvent):void{
    myChannel = mySound.play(posicion);
}
reproducir_btn.addEventListener(MouseEvent.CLICK, reproducir );
```

Cambiar el volumen

Para este caso ya deberemos utilizar el objeto SoundTransform, y lo asignaremos a la propiedad soundTransform del objeto SoundChannel.

El objeto SoundTransform tiene la propiedad volume cuyo valor va de 0 a 1 como en la mayoría de los casos. Para modificar el volumen, simplemente **asignaremos un valor a ésta propiedad, y luego asignaremos la instancia de SoundTransform a la propiedad soundTransform del objeto SoundChannel.**

Es algo parecido a la relación que hay entre el objeto TextField y el objeto TextFormat. Siempre que modificamos una propiedad de TextFormat, tenemos que volver a asignársela al campo de texto:  
myTextField.setTextFormat(TextFormat). Con el volumen pasaría algo parecido:

```
myTransform.volume = 0.5;

myChannel.soundTransform = myTransform;
```

```
var mySound:Sound = new Sound();
var myChannel:SoundChannel = new SoundChannel();
var myTransform = new SoundTransform();
var lastPosition:Number = 0;
mySound.load(new URLRequest("myFavSong.mp3"));
myChannel = mySound.play();
myTransform.volume = 0.5;
myChannel.soundTransform = myTransform;
```

```
pause_btn.addEventListener(MouseEvent.CLICK, onClickPause);
```

```
function onClickPause(e:MouseEvent):void{
    lastPosition = myChannel.position;
    myChannel.stop();
}
```

```
play_btn.addEventListener(MouseEvent.CLICK, onClickPlay);
```

```
function onClickPlay(e:MouseEvent):void{
    myChannel = mySound.play(lastPosition);
    myChannel.soundTransform = myTransform;
}
```

Publicado 4th May 2010 por [José Balaguer](#)

Etiquetas: [ActionScript 3.0](#)

1

 Ver comentarios



**cesar** [19 de agosto de 2010, 8:46](#)


exceleeente  
muchas gracias  
apesar de ser novato  
funciono a la primera

el unico detalle q no se como saltear  
es q al darle varios click al boton de play, el sonido se reproduce sobre el anterior y queda la cagada

sigo buscando como arreglar eso  
saludos!

[Responder](#)

Introduce tu comentario...

Comentar como: Christian (Gooq ▾)

Cerrar sesión

Publicar

Vista previa

☐ Avisarme