Música por computadora

Ernesto Romero y Hernani Villaseñor

Centro Multimedia 2012

Sesión 5

Canales - Out
SynthDef

Canales de salida

SuperCollider es un programa que nos permite manejar la salida del sonido por múltiples canales. Las tarjetas de sonido de las computadoras están configuradas en estéreo, es decir cuentan con dos canales: izquierdo y derecho, que en SC están asignados al número 0 para izquierdo y 1 para derecho.

En el caso de contar con una interfaz de audio tenemos la posibilidad de usar más de 2 canales. Algunas interfaces de audio cuentan hasta con 8 salidas simultáneas. Los 8 canales de una interfaz de este tipo enumera sus canales a partir del número 1, mientras que SC lo hace a partir del 0. Entonces, si mandamos una señal por el canal 0 de SC saldrá por el canal 1 de la interfaz.

Canales de la Tarjeta de sonido	1 2 3 4 5 6 7 8
Canales de salida de SuperCollider	0 1 2 3 4 5 6 7

En SC existen varias clases que nos ayudan a trabajar con los canales de audio asignando por donde saldrá el sonido, dos de las más comunes son: Out.ar y Pan2.ar.

Out.ar (canal, señal)

Asigna la salida del sonido a partir de un canal específico. Ese canal define un punto de partida u offset a partir del cual se va a distribuir el sonido.

canal: 0 izquierdo, 1 derecho, 3, 4, 5, ...multicanal
señal: cualquier UGen.

Ejemplos:

```
{Out.ar(0,Saw.ar(40)*EnvGen.kr(Env.perc(0.01,1),doneAction:2))}.
scope  // izquierda
```

```
{Out.ar(1,Saw.ar(40)*EnvGen.kr(Env.perc(0.01,1),doneAction:2))}.
scope  // derecha

{Out.ar(2,Saw.ar(40)*EnvGen.kr(Env.perc(0.01,1),doneAction:2))}.
scope  //  un tercer canal que solo sonara si tenemos una
interface de audio
```

Pan2.ar (señal, posición)

Distribuye el sonido entre dos canales consecutivos conservando su potencia. Es decir, que no suena más fuerte cuando está en los dos canales al mismo tiempo ni más quedito cuando está solo en uno o en otro. Si el Pan2 esta dentro de un Out los canales consecutivos en los que se distribuyen se cuentan a partir del offset del Out.

señal: cualquier oscilador o generador de sonido.
posición: -1 izquierda, 0 centro, 1 derecha y con todo el intervalo continuo entre -1 y 1 extrapolando el sonido entre los dos canales o bocinas.

```
{Pan2.ar(Pulse.ar(100,0.01, 0.2),-1)}.scope // izquierdo

{Pan2.ar(Pulse.ar(100,0.01, 0.2),0)}.scope // centro

{Pan2.ar(Pulse.ar(100,0.01, 0.2),1)}.scope // derecho
```

Dentro de un Out con canal offset en 1

```
{Out.ar(1,Pan2.ar(Pulse.ar(100,0.01, 0.2),-1))}.scope // derecho

{Out.ar(1,Pan2.ar(Pulse.ar(100,0.01, 0.2),0))}.scope // derecho y
tercer canal

{Out.ar(1,Pan2.ar(Pulse.ar(100,0.01, 0.2),1))}.scope // tercer
canal
```

SynthDef

Es una clase para crear sonidos complejos a partir de una plantilla de código. Su variedad y funcionalidad es bastante flexible y permite construir sonidos con alto grado de manipulación. Podríamos pensarlo como el diseño de los instrumentos de una orquesta que posteriormente sonarán cuando se ejecuten notas con ellos.

Para entender su funcionamiento analicemos el siguiente SynthDef:

```
(
SynthDef(\sinte1,{|frec=1000, amp=0.5, out=0|
     var sen, env;
     sen = SinOsc.ar(frec,0,amp);
     env = EnvGen.kr(Env.perc(0.1,1),doneAction:2);
     Out.ar(out, sen * env)
     }).add
);

Synth(\sinte1)
```

Lo primero es declarar el SynthDef y ponerle un nombre, para lo que usamos una diagonal invertida y a continuación nombramos el sinte como queramos:

```
(
SynthDef(\sinte1,
```

Luego abrimos una llave y entre pipes | | declaramos cuantos argumentos queramos usar.

```
{|frec=1000, amp=0.5, out=0|
```

Enseguida declaramos nuestras variables, recordemos que tanto argumentos como variables pueden llamarse como queramos, siempre y cuando comiencen con una letra minúscula.

```
     var sen, env;
```

A continuación le damos una funcionalidad a nuestras variables, en este caso le asignamos el UGen de SinOsc a la variable sen y le asignamos a la variable env un envolvente con el objeto EnvGen.kr de tipo percusivo Env.perc.

```
     sen = SinOsc.ar(frec,0,amp);
     env = EnvGen.kr(Env.perc(0.1,1),doneAction:2);
```

Por último asignamos la salida de nuestro SynthDef a 0, con lo cual sonará a partir del canal izquierdo. Multiplicamos sen por env, lo que dará como resultado que el tono de 1000 Hz suene el tiempo que tiene programada la envolvente, en este caso 1.1 segundos lo que ocasiona que suene un pequeño “blink”.

```
     Out.ar(out, sen * env)
```

Y con el mensaje .add le indicamos que se adiera a la memoria de librería del servidor.

```
      }).add
)
```

Para hacer sonar nuestro SynthDef declaramos la siguiente línea de código, la cual indica el nombre del sinte que queremos hacer sonar.

```
Synth(\sinte1)
```

Nota como en el servidor se inserta un nuevo sinte y como desaparace en el momento que termina de sonar, esto es debido al tipo de envolvente usada y el método con el que decimos como ingrese al servidor.

Para modificar los valores de un SynthDef necesitamos igualarlo a una variable y posteriormente mandar los cambios mediante el mensaje .set

```
a=Synth(\sinte1)

// el argumento se antecede de una diagonal invertida y después
de la coma se ingresa el nuevo valor, en este caso 1200

a.set(\frec, 1200)
```

Envolventes

Los envolventes determinan el comportamiento de un sonido respecto al desarrollo de la amplitud en el tiempo, más adelante los veremos con detalle. Por el momento podemos decir que SC maneja envolventes de tipo percusivo, o sea que aparecen y desaparecen por si mismos y otros que requieren de una orden para dejar de sonar, o sea que funcionan mediante un *gate* o disparo.

En este caso el envolvente de tipo percusivo tarda 0.1 de segundo en aparecer y 1 segundo en desaparecer.

```
EnvGen.kr(Env.perc(0.1,1),doneAction:2)
```

Envolvente de tipo asr, tarda 0.1 segs en aparecer, hasta que alcanza la amplitud 1, nosotros le indicamos cuando deje de sonar mediante un *gate*, lo que en este caso le llevará 1 seg en desaparacer.

```
(
var gate=1;
EnvGen.kr(Env.asr(0.1,1,2),gate,doneAction:2);
)
```

Ejemplo de un SynthDef con envolvente percusivo y paneo

```
(
SynthDef(\sinte2,{|frec= 500|
     var sen, pan, env;
     sen=LFTri.ar(frec,0,1);
     pan=Pan2.ar(sen,0,1);
     env=EnvGen.kr(Env.perc(0.1,2),doneAction:2);
Out.ar(0, pan * env);
}).add
)

b=Synth(\sinte2);
```

Ejemplo de un SynthDef con envolvente asr y paneo

```
(
SynthDef(\sinte3,{|frec= 500, gate=1, paneo=0|
     var sen, pan, env;
     sen=LFTri.ar(frec,0,1);
     pan=Pan2.ar(sen,paneo,1);
     env=EnvGen.kr(Env.asr(0.1,1,5),gate,doneAction:2);
Out.ar(0, pan * env);
}).add
)

c=Synth(\sinte3);
c.set(\frec, 600);
c.set(\frec, 700, \paneo, -1);
c.set(\frec, 800, \paneo, 1);
c.set(\frec, 900, \paneo, 0);
c.set(\gate, 0)
```