Comenzado el	jueves, 14 de mayo de 2020, 10:33
Estado	Finalizado
Finalizado en	jueves, 14 de mayo de 2020, 13:31
Tiempo empleado	2 horas 58 minutos

Información

Síntesis digital del sonido

En esta práctica vamos a diseñar y programar sistemas de síntesis en Csound.

Esta primera sesión está dedicada a conceptos y técnicas auxiliares, como la programación y uso de widgets para el control en tiempo real de al síntesis, el uso de filtros y la generación de envolventes. La segunda sesión estará más enfocada a practicar diferentes técnicas de síntesis.

Información

Control mediante interfaces gráficos: uso de widgets

Vamos a ver algunos ejercicios sobre el uso y programación de los *widgets* de CsoundQt. Esta utilidad tiene 2 aspectos diferenciados:

PROGRAMACIÓN

El diseño y comportamiento del interfaz de usuario (el *widget*) se hace gráficamente en la ventana de widgets. Ahí se define:

- Tipo de widget a usar.
- Canal por el que transmite y recibe datos.
- Configuraciones que dependen de su tipo.

USO

Hace referencia a qué hacer con ellos, cómo se comunican con nuestros programas. Esto se hace en el código de Csound, incluyendo:

- Cómo se reciben los datos desde los widgets.
- Cómo se les envía información.
- Qué hacer con ella.

Información previa: modo edición vs. modo ejecución

El panel de widgets tiene estos dos modos de trabajo:

- El modo edición, que se usa para la programación gráfica de los interfaces y
- El modo uso, que se utiliza para usar el sistema diseñado.

Para cambiar de uno a otro, dependiendo del sistema operativo y la versión de CsoundQt, podemos usar Ctrl–E, Cmd-E o (esto funciona en cualquier caso) ir al menú **Edit → Widget edit mode**.

Información

Verificaciones de CsoundQt

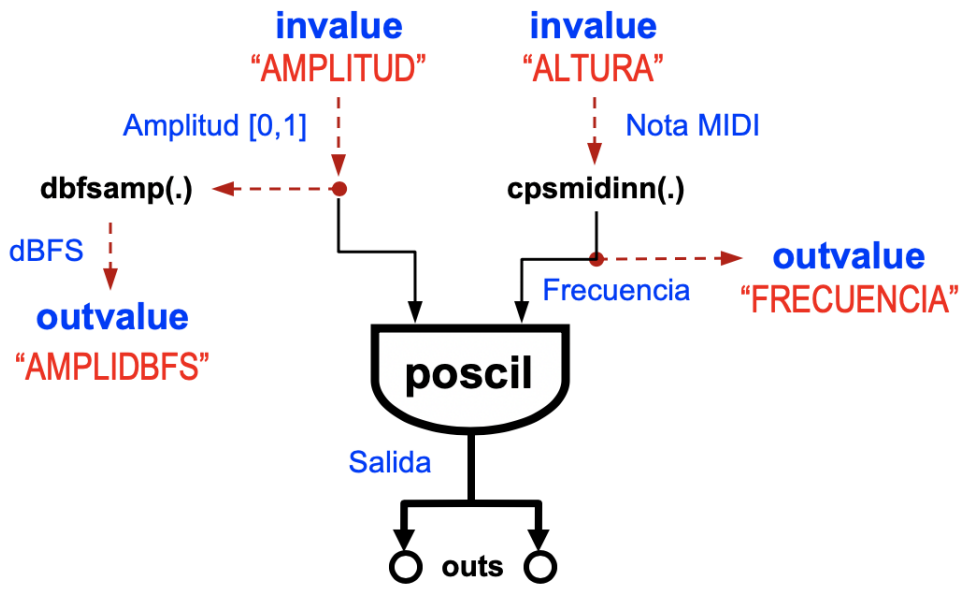
Antes de empezar dedícale 1 minuto a hacer unas comprobaciones que te ahorrarán tiempo y problemas después:

1. Comprueba que Csound no tiene desabilitados los eventos en tiempo real. Para ello,
   - Selecciona el botón **Configure → General** y comprueba que no hay seleccionadas ninguna de las 4 opciones de la ventana "**Performance tweaks**"
2. Activa los atajos de teclado para que funcione el Control+E para usar el modo edición, entre otros. Para ello,
   - Selecciona el menú **Edit → Set Keyboard Shortcuts** y pulsa el botón [Restore defaults].

Información

Ejercicio 9.1:

Vamos a hacer un sencillo oscilador pero controlado en tiempo real mediante los *widgets* de CsoundQt. El esquema del instrumento es el siguiente.



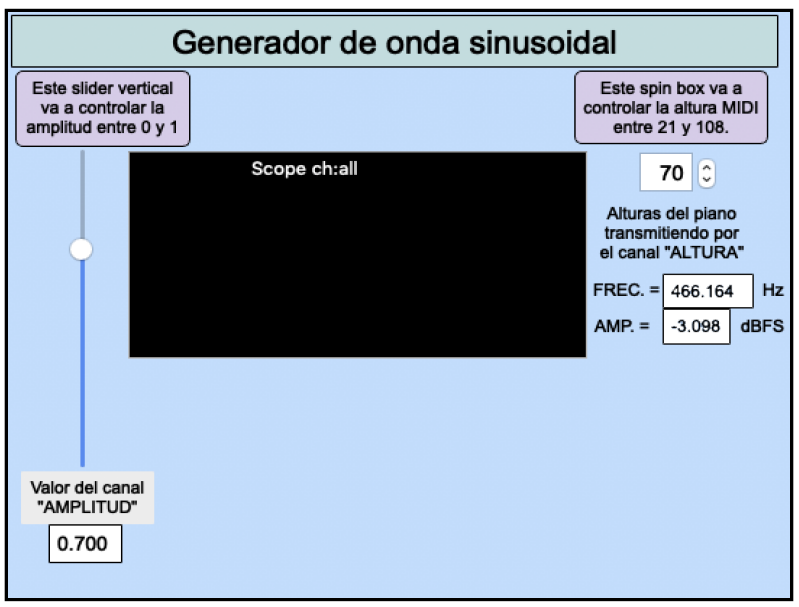
Los operadores **invalue** y **outvalue** son los que se usan para realizar la comunicación (líneas discontinuas) desde y hacia los widgets, respectivamente, usando los canales indicados entre comillas.

Primero se trata de hacer unos *widgets* para controlar y visualizar la amplitud y frecuencia del sonido sintetizado. Al pulsar con el botón derecho del ratón sobre el panel, se desplegará el menú que te permitirá crear nuevos *widgets* (Create New ›). Al crearlos se despliega la ventana de propiedades. Esto también puede hacerse más tarde con el botón derecho del ratón.

- Descarga esta PLANTILLA (botón derecho → guardar como)
- En el instrumento y en la partitura hay algunos comentarios que deberás eliminar cuando se indique para la entrega del ejercicio.

Programación del interfaz gráfico

- Asegúrate de que se visualice el panel de *widgets* (menú **Ver** → **widgets**).
- Aparecerá un panel con textos en los que se describe lo que tiene que programarse y un rectángulo central (es un **scope**) para ver la onda que vas a generar.
- Widgets para la amplitud:
  - Un **slider** vertical (a la izquierda). Propiedades: Min = 0, Max = 1, Nombre del canal = AMPLITUD.
  - Un **display** (bajo el cartel de la amplitud) para ver el valor del control de amplitud. Nombre del canal = AMPLITUD, Font Size = 14, botón Background = activado.
  - 2 **Labels** con los textos "AMP. =" y "dBFS", respectivamente. En ambos casos Font Size = 12.
  - Otro **display** (entre los labels anteriores) para ver la amplitud en dBFS. Canal AMPLIDBFS, Font Size = 12 y Background activado.
- Widgets para la frecuencia:
  - un **spin box** (arriba a la derecha) para la altura MIDI del sonido. Min = 21, Max = 108, Resolution = 1, Texto = 0, Nombre del canal = ALTURA, Font Size = 14, Alignment = Right.
  - 2 **Labels** con los textos "FREC. =" y "Hz", respectivamente. Font Size = 12 para ambos.
  - Un **Display** para mostrar la frecuencia en Hz. Canal FRECUENCIA, Font Size = 12 y el botón Background activado.
- El aspecto final que debe tener el panel debe ser como el de esta figura:



Programación del instrumento

Entradas:

- Lo primero que hay que programar en el instrumento **widgets** es la entrada de los valores desde los *widgets* con las 2 órdenes **invalue** necesarias para los canales "AMPLITUD" y "ALTURA". Este operador sólo admite como salidas variables de tipo k-.

  - Hazlo al principio del instrumento, donde se indican las variables kAlt y kAmp.
- Luego, el valor de altura se convierte a frecuencia sobre la variable kFrec con **cpsmidinn**(·) (esto se da hecho).

Salidas:

- Para enviar valores desde el instrumento a los widgets hay que usar operadores **outvalue** (**outvalue** "CANAL", variable -- ver material previo). Estas órdenes deben ir después de que se lean los valores con **invalue** y del conversor de MIDI a frecuencia.
  - Por el canal "FRECUENCIA" debes enviar la frecuencia previamente calculada para cada nota MIDI.
  - Por el canal "AMPLIDBFS" debes enviar la amplitud convertida a dBFS. Este valor lo puedes obtener con la función **dbfsamp**(·).

Síntesis:

- Después, donde se indica en la plantilla, falta crear la onda con un oscilador **poscil** que, si sólo se le da la amplitud y la frecuencia (en ese orden), genera una onda sinusoidal con esos parámetros.
- La salida será estéreo pero con la misma señal por ambos canales con **outs** (se da programada).

- Cuando esté listo, pulsa el Play de CsoundQt y se activará el instrumento **widget** durante 1 minuto (partitura: `i "widget" 0 60`) y podrás probar si el sistema funciona correctamente. Por ejemplo, si la altura = 81 (LA_5), la frecuencia debe salir 880 Hz y si la amplitud es = 0.5, la amplitud en dBFS = -6,021.

Para terminar el ejercicio y prepararlo para la entrega

- Ajusta el **slider** de la amplitud para que quede a 0.5 (lo más cercano a este valor y nunca inferior a él).
- Ajusta el **spin box** de la altura para que se quede al valor de altura MIDI 60 (DO_4).
- Elimina los comentarios de las líneas:
  - `kAlt = 60`
  - `kAmp = .5`
- En la partitura:
  - Descomenta la línea: **`t 0 5000`**
- Guarda ahora el proyecto y no hagas nada más con él hasta el siguiente ejercicio.

Información

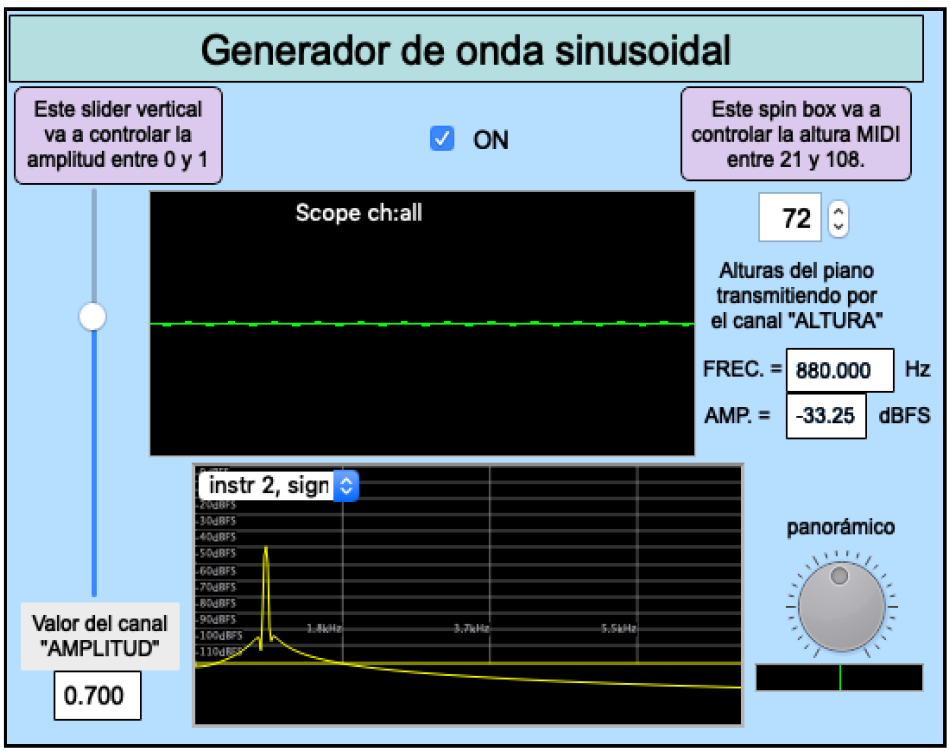
Ejercicio 9.2:
knobs, spin boxes, controllers y graphs

- Guarda el proyecto anterior como uno nuevo de nombre `ej9.2.csd`.
- Cambia el fichero de salida a `ej9.2.wav`.
- En el instrumento, vuelve a comentar las líneas
  - `kAlt = 60`
  - `kAmp = .5`
- En la partitura, vuelve a comentar la línea: **`t 0 5000`**

Programación del interfaz gráfico

- Añade arriba en el centro un botón **Check Box**, que hará de interruptor, con nombre de canal ONOFF (el resto se queda como está) y a su derecha un **Label** con el texto ON y Font Size = 14.
- Añade en la parte inferior derecha (ver figura) un **knob** para controlar el panorámico. Sus propiedades deben ser las siguientes:
  - canal = PANORAMA, mínimo = 0, máximo = 1
- Sobre él añade la etiqueta (**label**) "panorámico" (tamaño de fuente = 12, alineación = centro)
- Bajo él añade un **controller** para visualizar la posición del panorámico estéreo (ver su forma en la figura). Sus propiedades serán:
  - Horizontal channel name = PANORAMA
  - Vertical channel name = (*dejar en blanco*)
  - Type = line
  - Min X value = 0
  - Max X value = 1
- Añade a su derecha un **graph** (ver figura) para ver el espectro de la onda generada. La única de sus propiedades a editar será: Index Channel name = outFT.

El aspecto final que debe tener el panel debe ser como el de la figura:

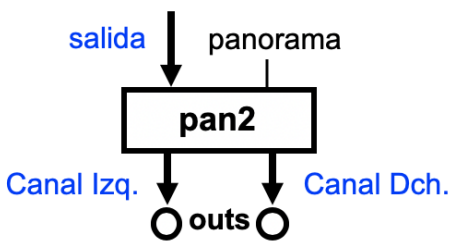


Programación del instrumento

- Al principio del instrumento, junto con los anteriores **invalue**, programa la recepción del botón interruptor con **invalue** leyendo el canal "ONOFF". Almacena el valor leído en una variable de nombre kOn.
- Añade también la lectura del panorama (por el canal "PANORAMA").
- Añade a continuación una orden para que, si se desactiva el botón, se cancele la onda, haciendo su amplitud cero:

```
kAmp = kAmp*kOn    ;  kOn vale 1 cuando está activado y 0 cuando no lo está
```

- Utiliza luego (justo antes del **outs**) la variable leída para el panorama como entrada de un operador **pan2** de control del panorama estéreo, que repartirá la señal generada por **poscil** entre los dos canales estéreo (0 = todo en un canal, 1 = todo en el otro, 0.5 = sonido centrado). Ver figura:



- Las señales de salida que tenía el operador **outs** habrá que cambiarlas por las 2 salidas del **pan2**.

- Para ver el espectro de la onda de salida, al final del código, hay que descomentar las 2 órdenes que envían información mediante **outvalue** `"outFT",1` y la que dibuja el espectro con la orden **`dispfft aSal,`** `0.1, 2048`
- Compila y comprueba que funciona según lo esperado.

Para terminar el ejercicio y prepararlo para la entrega

- Deja activado el **Check Box** ON.
- Ajusta el *slider* de la amplitud para que quede a 0.6.
- Ajusta el *spin box* de frecuencia para que se quede a un valor 64 (MI_4).
- Ajusta el *knob* para que el panorámico quede totalmente a la izquierda.
- Elimina los comentarios de las líneas:
  - `kAlt = 60`
  - `kAmp = .5`
- Añade justo después de la línea anterior, esta: `kOn  = 1`
- Descomenta la línea: **`t 0 5000`**
- Guarda así el proyecto.

Información

Ejercicio 9.3: sintetizador con control externo

Hay 3 formas de controlar un instrumento Csound:

1. Desde la partitura de Csound
2. Por eventos en tiempo real desde el panel de widgets o desde el teclado virtual de Csound
3. Por eventos MIDI que lleguen mientras se esté ejecutando el programa

En prácticas hemos probado ya el control de tipo 1. Ahora el objetivo es adaptar un sintetizador sustractivo sencillo para pueda ser controlado por las vías 2 y 3.

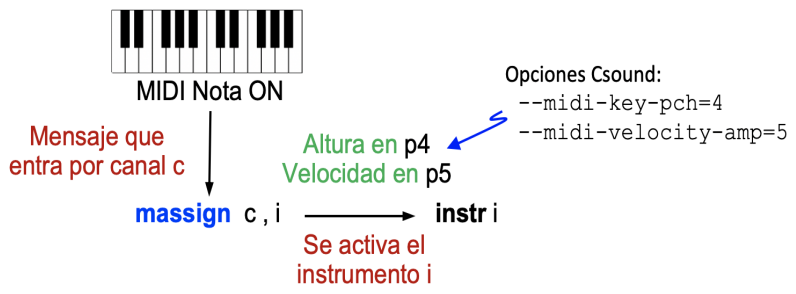
El sintetizador de partida se suminsitra en el siguiente PROYECTO (botón derecho → guardar como). Antes de hacer nada, compílalo y comprueba que funciona (sonará un MI_3 enviado desde la partitura).

Control en tiempo real con el teclado virtual de Csound

El teclado virtual que tiene Csound emula órdenes MIDI. El resultado será el mismo que si lo usamos con un controlador MIDI hardware. Las modificaciones que hay que hacer en el proyecto suministrado son las siguientes:

- **En las opciones de compilación**:
  - Cuando un instrumento se controla desde la partitura se le envían los controles como parámetros p2, p3, p4... Cuando el control es vía MIDI, estos parámetros llegan mediante NOTE ON, OFF y resto de controles. Csound tiene un sistema para compatibilizar ambas formas de control y que los instrumentos reciban los parámetros MIDI como si vinieran de la orquesta. Para ello debes añadir a las opciones de compilación, las siguientes:
    - `--midi-key-pch=4` : Hace que la altura de un NOTA ON MIDI la vea el instrumento como **p4** en alturas de Csound.
    - `--midi-velocity-amp=5` : Hace que la velocidad de un NOTA ON MIDI la vea el instrumento como **p5** entre 0 y lo que valga **0dbfs**.
- **En la cabecera** de la orquesta (después de **0dbfs** = 1):
  - Cada vez que llega una orden MIDI, se envía al instrumento cuyo número sea igual al del canal MIDI de la orden. Esto se puede modificar con la orden **massign**:
    - La orden **`massign`** `canal,instrumento` envía los mensajes que entren por el canal `canal` al instrumento `instrumento`. Si se pone `canal` = 0, se enviará a ese instrumento lo que entre por todos los canales.
    - Por lo tanto, la orden que debes utilizar es **massign** 0, "sinte"

Resumen esquemático del funcionamiento del control MIDI en Csound



- **En el instrumento**:
  - Nada. Se trata de comprobar que este instrumento pude funcionar de igual manera controlado por la partitura o por un teclado MIDI.
- **En la partitura**:
  - Comenta la orden **`i`** y descomenta la orden **`e`** 40, que mantendrá el instrumento activo durante 40 segundos cada vez que se compile.

Para probar el instrumento:

- Compila el instrumento y despliega el teclado virtual de CsoundQt (busca el icono correspondiente al teclado virtual o la opción en el menú **Ver**). Estará activo durante 40 s para que lo pruebes con ese teclado.

Control desde un fichero MIDI

Ya hemos visto cómo controlar un instrumento mediante órdenes MIDI. Ahora se trata de controlarlo mediante órdenes MIDI que lleguen desde un fichero MIDI.

- En las opciones de compilación:
  - debes añadir la opción `-F smb.mid` para que tome ese fichero MIDI como entrada para el control.
- En el instrumento:
  - Nada. El instrumento es compatible con cualquier tipo de control gracias a las anteriores opciones de compilación.
- En la partitura:
  - Cambia el tempo a 105 BPM constante y desde el principio.
  - Con la orden de fin de partitura **`e`** `40` el sistema estará activo durante 40 tiempos = 22,86 segundos.

Cuando esté listo para compilar, usa la siguiente para probar el programa.

- Descarga y usa este FICHERO MIDI.
- Compila y escucha el resultado.

Información

Filtros en síntesis

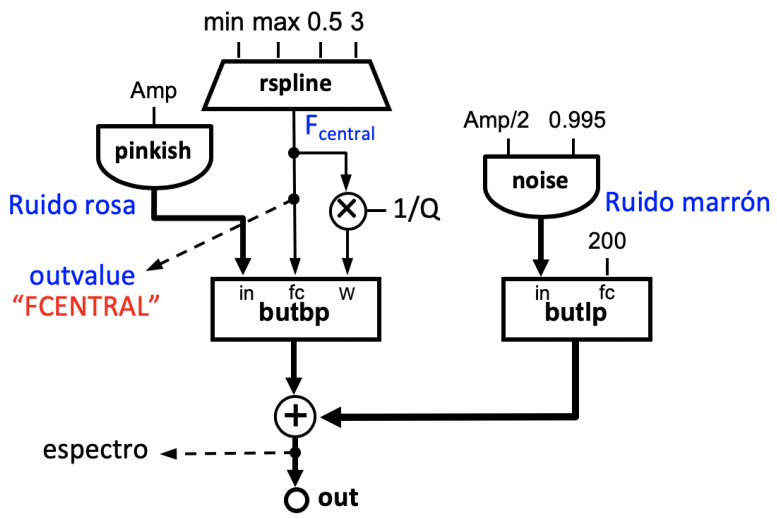
Ejercicio 9.4: Filtros pasa-baja y pasa-banda

Vamos a hacer un instrumento que aplique filtros para hacer una versión sintética del sonido del viento. El sonido tendrá 2 componentes:

- Una parte de ruido rosa filtrado por un pasa-banda estrecho. Esto producirá una parte afinada para simular el silbido del viento con su frecuencia central que estará controlada por una curva aleatoria.
- Una parte de ruido marrón filtrado por una pasa-baja para aportar más potencia al sonido.

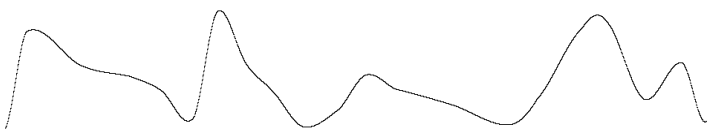
Se proporciona esta PLANTILLA (botón derecho → guardar como) que contiene la cabecera de la orquesta, un operador definido por el usuario para visualizar el espectro y el "esqueleto" básico del instrumento **viento**. También se incluye los widgets para un espectrógrafo y para visualizar la evolución de la frecuencia del pasa-banda.

El esquema del instrumento se presenta en la siguiente figura:



Aclaraciones:

- En la parte izquierda se genera el ruido rosa, con **pinkish**. La amplitud será máxima; es decir iAmp = **0dbfs**.
- El ruido rosa se filtra con el pasa-banda **butbp**, que necesita su frecuencia central f_c y ancho de banda W:
  - La frecuencia central la proporciona **rspline** (*random spline*) que genera una curva aleatoria entre los valores mínimo (min = 30) y máximo (max = 3000), con coeficientes de cambio entre 0.5 y 3. Esto genera curvas suaves de este tipo:



  - El ancho de banda se obtiene dividiendo la f_c por el valor de Q = 10 (filtro estrecho), causando una percepción de altura a la frecuencia del filtro ($Q = f_c / W$).
- En la parte derecha se genera el ruido marrón con **noise**, que luego es filtrado por el pasa-baja **butlp**, con frecuencia de corte = 200 Hz, para evitar un exceso del altas frecuencias.
- Ambos ruidos se suman al final y se sacan con **out**.

Visualización:

- Los *widgets* se dan programados.
- Para comunicarte con el controller que muestra la f_c debes usar el canal "FCENTRAL".
- Para ver el espectro, ya se proporciona el código.

- Compila el instrumento y escucha el resultado. La partitura está preparada para que suene durante 11 segundos.

Información

Envolventes en síntesis

Una de las dificultades de aplicar envolventes en síntesis es que puede que no conozcamos a priori la duración de los sonidos (si depende de cuando llegue un nota MIDI OFF) y, por tanto, el diseño de las curvas debe ser diferente.

Ejercicio 9.5: Generación y uso de envolventes

Vamos a usar un generador sencillo para añadirle envolventes de diferentes tipos. Utiliza esta PLANTILLA (botón derecho → guardar como).

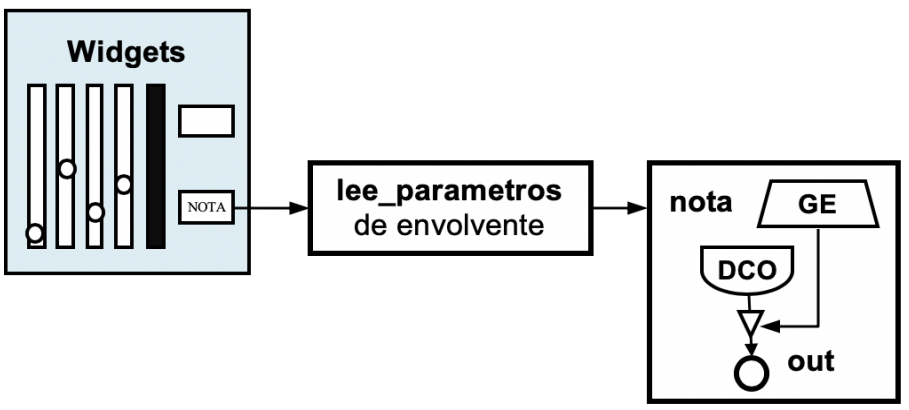
La plantilla incluye los siguientes elementos:

- Opciones de compilación y cabecera.
- Un panel de *widgets* (activa la visualización del panel de *widgets* para verlos), que contiene:
  - 4 deslizadores para el control independiente de los tiempos de ataque, decaimiento y relajación, y la amplitud de sostenimiento.
  - Un menú para elegir el tipo de envolvente.
  - Un botón etiquetado **[NOTA]** para activar el sistema cada vez.
- Un instrumento (**lee_parametros**) ya programado y que no hay que modificar para leer los vlaores de los widgets y activar con ellos el instrumento **nota** que genera el sonido cada vez que se active.
- La estructura básica del instrumento **nota**.
- Una partitura que mantiene activo el sistema durante 2 minutos cada vez que se compila, mediante la instrucción

```
f 0 [2*60]
```

Funcionamiento del programa:

- Cada vez que compiles dispones de 2 minutos en los que el programa espera que actives notas pulsando la tecla del panel de *widgets*.
- Cada vez que pulses esa tecla, el instrumento **lee_parametros** lee los valores de los parámetros de envolvente que hayas establecido en el panel y llama a **nota** (con el *opcode* **event**) para que haga sonar una nota de 2 segundos.
- El instrumento **nota** creará la envolvente y la multiplicará por la onda antes de enviarla a la salida.
- Hay una orden **prints** con la que podrás ver en la consola los valores de los parámetros de la envolvente cada vez que lo actives.



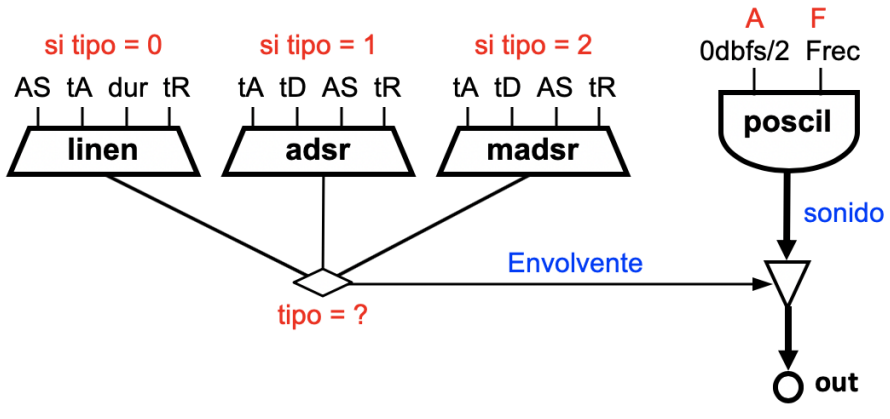
Parámetros de envolvente en el instrumento **nota**:

- Tiempo de ataque (absoluto): llega al instrumento en **p6** con el valor del primer *slider* de los *widgets*. Como debe ser en tiempo absoluto en segundos, será el valor t_A = **p6**.
- Tiempo de decaimiento (relativo): llega en **p7** con el valor del segundo *slider* como una proporción entre 0.01 y 0.30. Para que sea un tiempo proporcional a la duración de la nota debe ir multiplicado por la duración, t_D = **p7***iDur.
- Amplitud de sostenimiento: vendrá en **p8** entre 0.01 y 1. Como la envolvente la vamos a crear normalizada, este es el valor que hay que usar directamente: A_S = **p8**
- Tiempo de relajación (relativo): viene en **p9**, también como una proporción de la duración, por lo que hay que hacer lo mismo que con t_D: t_R = **p9***iDur.

- Protección de duración: para evitar configuraciones extremas, añade la siguiente protección sobre el tiempo de relajación, después de calcularlo:
  - **si** $t_A + t_D + t_R >$ duración **entonces** t_R = duración – ($t_A + t_D$) **si no** t_R se deja con el valor que tenga.

Generación y aplicación de las envolventes

Programa los generadores de envolvente según las especificaciones del siguiente esquema:



El sonido lo generará el oscilador **poscil**. Se usará una envolvente u otra en función de cuál sea el tipo de envolvente seleccionado en el panel de widgets. El rombo del esquema simboliza que se usa una u otra envolvente en función del valor del tipo elegido. Esto se implementará con la siguiente estructura:

```
if iTipo=0 then
  Crear envolvente AR/ASR con linen
elseif iTipo=1 then
  Crear envolvente ADSR con adsr
elseif iTipo=2 then
  Crear envolvente ADSR-X con madsr
endif
```

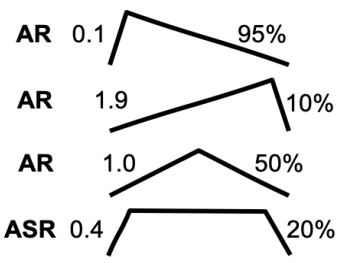
Prueba del programa:

AR / ASR

Prueba los siguientes valores de los parámetros, para comprobar que está bien implementado (son 3 AR y 1 ASR) (comprueba la evolución de la amplitud y los valores de tiempos, mediante el **prints**). Asegúrate de tener visibles los *widgets*. Activa una nota con el botón **[NOTA]** para cada par de parámetros (cada nota que actives durará 2 segundos):

En todos estos casos, t_D = 0.01 y A_S = 1, para que no influyan.

- t_A = 0.10, t_R = 0.95 (ataque = 0.1 s y relajación = 95% de la duración)
- t_A = 1.90, t_R = 0.10 (ataque = 1.9 s y relajación = 10% de la duración)
- t_A = 1.00, t_R = 0.50 (ataque = 1.0 s y relajación = 50% de la duración)
- t_A = 0.40, t_R = 0.20 (ataque = 0.4 s y relajación = 20% de la duración)



ADSR

Para este tipo de envolventes, prueba los siguientes valores de los parámetros:

- Cambia el tipo de envolvente en el *widget* menú a ADSR.
- t_A = 0.20, t_D = 0.20, A_S = 0.50, t_R = 0.20 (para comprobar que va bien)
- t_A = 0.20, t_D = 0.20, A_S = 0.50, t_R = 0.90 (para comprobar que este caso nos hemos quedado sin fase estable)

ADSR-X (ADSR extendida)

Cuando no conocemos la duración de una nota (porque, por ejemplo, la ponemos en marcha mediante un evento MIDI on), no podemos usar el método anterior. Se soluciona usando el operador **madsr** (ADSR-extendido), que comienza la fase de relajación cuando termina la activación del instrumento, bien porque llegue un nota off MIDI o porque se termine la activación.

- Para probarlo, cambia el tipo de envolvente en el *widget* menú a ADSR-X.
- Activa una nota con los mismos parámetros que en la última activación anterior. Deberás percibir una clara extensión del sonido y que ahora sí que hay fase estable.

Fin del ejercicio

Para la entrega, tal y como esté el programa en este momento, comenta en la partitura la orden `f 0 [2*60]` y descomenta la orden `i "lee_parametros" 0 0.1 72`

◄ Uso básico de los widgets en CsoundQt (~8 min.)

Ir a...

Entrega de la primera sesión de la práctica 9 (P) ►