ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Proyecto fin de Carrera Grado

Grado en Ingeniería Informática de Computadores

Fundamentos del Lenguaje Csound

Realizado por
Rafael Escudero Lirio

Dirigido por
Víctor Jesús Díaz Madrigal

Departamento
Lenguajes y Sistemas Informáticos

Sevilla, septiembre de 2020 (v.0.0.1)

Resumen

El lenguaje de programación Csound está principalmente destinado a la síntesis y producción de sonido en un ámbito musical. Es un lenguaje de código abierto, accesible en la plataforma GitHub y su compilador está programado en C, de ahí su nombre. Recibe periódicamente contribuciones de desarrolladores de diferentes partes del mundo y se encuentra en su versión 6.14.0 a la fecha de realización de este trabajo. Es relativamente poco conocido al ser de uso muy específico y al estar toda su documentación dedicada a personas angloparlantes.

Al ser éstas las circunstancias del lenguaje se ha formado a su alrededor una comunidad dedicada que se agranda con el paso del tiempo, con el fin de seguir desarrollando su tecnología y de ahí la justificación del principal motivo del presente trabajo de fin de grado:

Dar a conocer los fundamentos del lenguaje Csound.

Para tal fin, se estructurará el presente trabajo a modo de guía introductoria de uso del lenguaje. Destacando las principales características de éste y priorizando la escalada progresiva de complejidad al decidir el orden de exposición de los conceptos, con intención de favorecer el aprendizaje a medida que se vaya usando el documento. Se priorizará también que el contenido se exponga de manera unitaria para favorecer el uso del presente documento como guía de consulta rápida de conceptos básicos de Csound.

Se exponen a continuación los principales objetivos del presente trabajo:

- Dar a conocer en mayor medida el lenguaje de programación Csound.
- Proporcionar una guía de aprendizaje introductorio al lenguaje Csound.
- Proporcionar un documento de consulta rápida de conceptos del lenguaje Csound.
- Proporcionar ejemplos prácticos de programación usando el lenguaje Csound a modo de demostración capacitiva del lenguaje.

Por último destacar que tecnologías como Csound invitan al trabajo colaborativo e interdisciplinar en distintos ámbitos como son en este caso la informática y la música. Es por ello fundamental dar a conocer sus diferentes usos con el fin último de ampliar el desarrollo de los conocimientos tecnológicos.

Índice general

Índice de cuadros

Índice de figuras

Índice de código

CAPÍTULO 1

Introducción al Lenguaje y Fundamentos del Sonido

1.1– ¿Qué es Csound?

Csound es un lenguaje de programación de alto nivel orientado a objetos dedicado a la síntesis, edición y producción de sonido. Su sintaxis es concreta y su compilador está codificado en lenguaje C. Entre los usos prácticos del lenguaje podemos encontrar ejemplos en la página oficial (https://csound.com/projects.html) de proyectos dedicados a la síntesis de música en directo, edición de sonido mediante efectos generados programáticamente y creación de interfaces VST o instrumentos virtuales entre otros.

Podemos referirnos a Csound como un "Compilador de Sonido".

1.2– ¿Por qué usar Csound?

Hay muchas buenas razones para usar Csound. Csound es software libre de código abierto con licencia LGPL, está en constante desarrollo y tiene una comunidad de desarrolladores que crece día a día,proporciona además un punto de conexión entre la disciplina informática y el ámbito de la música y el sonido. Es además una gran herramienta científica al facilitar la exposición y experimentación de conceptos relacionados a las ondas del sonido, con una amplia librería de funciones y objetos útiles para ello. En lo que se refiere a la composición musical, Csound se ha usado principalmente para generar música electrónica a lo largo de su historia aunque podemos encontrar a compositores de cualquier género músical o tipo de instrumento sacándole provecho al lenguaje. No es de extrañar vistas las tendencias actuales en el mundo de la música, donde para triunfar a nivel multitudinario es prácticamente imprescindible contar con un buen productor que sepa embellecer el sonido. Es frecuente ver a usuarios del lenguaje interpretando su música en directo con ayuda directa de éste. Y por último debo destacar que Csound es compatible con los principales sistemas operativos del mercado, desde Windows a iOS pasándo por distrbuciones Linux. Y como veremos más adelante en esta guía, sus funciones pueden llamarse desde el código de otros lenguajes como Python, java o C.

1.3– Breve historia de Csound

De manera resumida, Csound fue desarrollado en un principio por Barry Vercoe (véase la figura 1.1) en 1985 en el MIT[1] Media Lab. Y desde la década de los años 1990, una amplia variedad de desarrolladores ha colaborado a su código abierto, aportando además documentación, ejemplos y artículos sobre el lenguaje.

Figura 1.1: Barry Vercoe

Para hablar de los verdaderos orígenes de Csound debemos remontarnos a la década de los años 1970, a los orígenes de la producción informática de sonido. Matt Mathews creó MUSIC, el primer lenguaje informático para la generación de ondas de audio digital. En éste se basarían sus posteriores iteraciones: Music1, Music2, Music3, Music4, Music4B... Hasta llegar a Music11, desarrollado por el mentado Barry Vercoe, y del cual Csound es sucesor directo. Posteriormente el desarrollo del lenguaje ha continuado gracias a su comunidad con John Fitch de la University of Bath a la cabeza y como dueño del repositorio de código abierto en la plataforma GitHub. En la actualidad se realizan periódicamente las ICSC[2], conferencias dedicadas a Csound a nivel internacional de manera periódica, siendo la última fecha de realización el 27 de septiembre de 2019 en la actualidad del presente documento.

1.4– Las características del lenguaje

A continuación se muestra una lista de las características principales y técnicas del lenguaje:

- Usado para la síntesis, edición y análisis de sonido y música.
- Lenguaje de código abierto.
- Programación funcional.
- Licencia de distribución LGPL.
- Orientado a objetos.
- Compilador programado en lenguaje C.
- 30 años de desarrollo.
- Compatibilidad con otros lenguajes y retrocompatibilidad de versiones.

[1] Massachusetts Institute of Technology
[2] International Csound Conference

1.5– Alternativas a Csound

Voy a hablar de tres principales alternativas a Csound, siendo estos lenguajes dedicados principalmente a la síntesis, edición y análisis del sonido.

Lenguajes Gráficos

- Max/MSP
- PureData

Lenguaje Escrito

- SuperCollider

1.5.1. Max/MSP

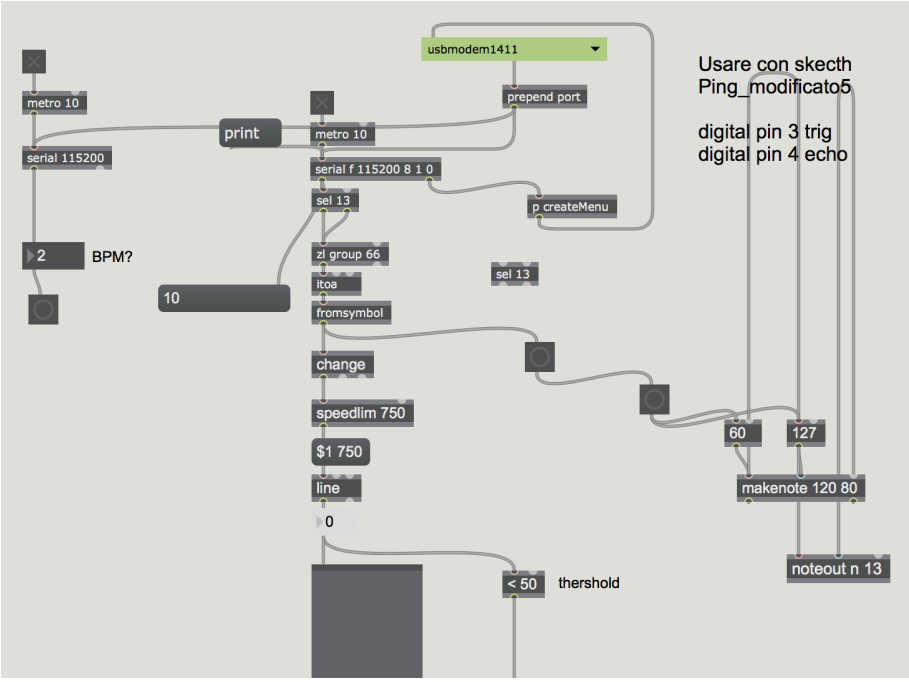

Figura 1.2: Ejemplo Max/Msp

1.5.2. PureData

1.5.3. SuperCollider

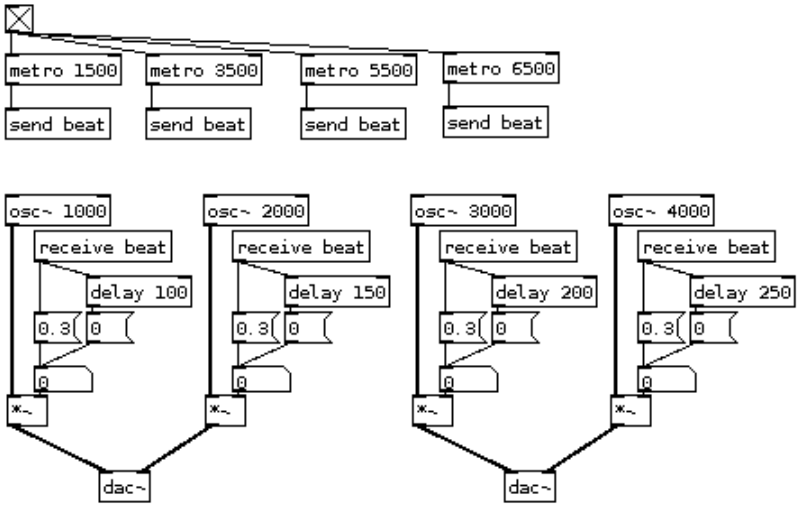

Figura 1.3: Ejemplo PureData

CAPÍTULO 2

Partes y plantillas

2.1– Partes

Todas las memorias de Trabajo fin de Grado deberán constar de las siguientes partes

- Portada (según formato oficial). No debe incluir número de página. Debe incluir:
  - Sello de la universidad de Sevilla a dos tintas
  - ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA
  - Trabajo fin de Grado
  - *Denominación del Grado*
  - Realizado por: *Nombre y apellidos del estudiante*
  - Dirigido por: *Nombre y apellidos del tutor o tutores*
  - Departamento *Nombre del departamento en el que se lee el TfG*
  - Sevilla, *Mes y año de la convocatoria de entrega*
- Preliminares: las páginas no se numeran o se numeran con números romanos.
  - Resumen en castellano (máximo una hoja)
  - Abstract (resumen en inglés, obligatorio para el caso de las memorias escritas en inglés, opcional para las escritas en castellano)
  - Agradecimientos (opcional)
  - Índice general (contenido de la memoria, con mención de las partes en que está dividida)
  - Índice de figuras (opcional)
  - Índice de cuadros (mal llamados en general, tablas) (opcional)
  - Índice de código o algoritmos (opcional).
- Cuerpo de la memoria, dividida en capítulos. El contenido de la memoria ha de incluir los elementos característicos de un proyecto de ingeniería o de un estudio o trabajo en el ámbito de una investigación, los cuales, en el sentido más amplio, son:
  - Definición de objetivos.
  - Análisis de antecedentes y aportación realizada.
  - Análisis temporal y de costes de desarrollo.

- Análisis de requisitos, diseño e implementación.
- Manual de usuario, en su caso.
- Pruebas.
- Comparación con otras alternativas.
- Conclusiones y desarrollos futuros

Estos puntos podrán ser ajustados y modificados en función de la naturaleza del proyecto realizado.

- Bibliografía: se deben documentar las fuentes bibliográficas utilizadas en el formato APA 2009.
- Índice alfabético o glosario (Lista ordenada de los conceptos, los nombres propios, etc.; que aparecen en la memoria, con las indicaciones necesarias para su localización)(opcional)
- Apéndices: Si la memoria contiene algún artículo de investigación o similar, este se incluirá en los Apéndices. Es necesario, en ese caso, incluir en la página anterior una hoja con la citación bibliográfica. En esta citación, el título del artículo se debe enlazar con la página web de la revista en la que aparecen el resumen o abstract y el acceso al texto completo.

La numeración de las páginas de la bibliografía y del glosario debe continuar la del cuerpo de la memoria. Los apéndices pueden llevar su propia numeración independiente o usar la general del cuerpo de la memoria.

2.2– Impresión

La entrega del memoria y en su caso, el depósito en biblioteca, se hacen en formato electrónico. Debido a ello, la memoria se presentará *a una cara*. Si se requiriera por algún motivo la impresión de la misma, se recomienda vivamente preparar la memoria adecuadamente para su impresión. Algunas sugerencias:

- Dejar una página en blanco cuando sea necesario para que los capítulos comiencen siempre en página impar (derecha)
- Ajustar los márgentes para que el exterior sea ligeramente más grande que el interior.
- Ajustar las cabeceras y pies de página (en su caso). Por ejemplo, si el número de página ocurriese en un lateral de la cabecera o pie, este debe ser siempre el exterior (derecho para las páginas impares, izquierdo para las pares)

2.3– Plantillas

Se encuentra disponible una plantilla LaTex en la sección de documentos de la plataforma de la aplicación de TfG de la ETSII `https://tfc.eii.us.es/TfG/`. Esa plantilla ha sido utilizada para preparar este documento. Se espera en breve disponer de plantillas de ejemplo para las aplicaciones OpenOffice Writer y Office Word.

Nota: Las plantillas se proporcionan como ejemplo, las condiciones obligatorias son las que constan en este procedimiento.

CAPÍTULO 3

Ejemplos

3.1– Manejo de la bibliografía

En esta sección mostramos brevemente ejemplos de referencia a la bibliografía citando un libro (de Sousa, 2004), un artículo (Bezos, 2007) y una página web (Autores, 2014).

Se recuerda que son campos obligatorios en todos los ítems de la bibliografía: autor(es), título del libro o artículo y año de publicación. En el caso de páginas web, es obligatoria la fecha de la última consulta. En general, la bibliografía debe ayudar al lector a encontrar fácilmente los ítems citados.

3.2– Código

En general se debe evitar incluir código o seudocódigo en la memoria. Si fuese preciso, se destacará de forma que sea fácilmente identificable y se indexarán los trozos de código incluidos. Un ejemplo puede verse a continuación.

```
%COMANDO PARA INSERTAR UN CUADRO UTILIZANDO EL FORMATO:
%1---> especificar numero de columnas y su alineacion ejm:
   % |r||c|c| r=right, c=center,l=left
%2---> especificar el caption o titulo de la figura
%3---> label para hacer referencia a la tabla insertada
%4---> contenido de tabla separando columnas con & y filas con \\
\newcommand{\cuadro}[4]{
      \begin{table}[htb]
         \centering
            \begin{tabular}{#1}
               \hline
                  #4
               \hline
            \end{tabular}
         \caption{#2}
         \label{#3}
      \end{table}
}
```

Código 3.1: Código de ejemplo en LaTeX

3.3– Imágenes

Este es un ejemplo de inclusión de figura en el texto (véase la figura 3.1).

Figura 3.1: Ada Lovelace

Figuras y cuadros se colocarán preferentemente tras el párrafo en el que son llamados por primera vez. Si no cupieran, se colocarán (en orden de preferencia):

- Al final de la página en que se llaman
- Al principio de la siguiente página
- Al final del capítulo

siempre respetando el orden de aparición en el texto.

3.4– Cuadros (mal llamados Tablas)

Este es un ejemplo de inclusión de cuadro en el texto. Véase el cuadro 3.1

elemento elemento	elemento elemento	elemento elemento

Cuadro 3.1: Cuadro de prueba

Referencias

Autores, V. (2014). *Escuela Técnica Superior de Ingeniería Informática.* ETSII. Descargado de `http://www.informatica.us.es` (fecha de consulta: 24 de Noviembre de 2014)

Bezos, J. (2007). The titlesec and titletoc packages. *TexEmplares*, *8*, 283–298.

de Sousa, J. M. (2004). *Ortografía y ortotipografía del español actual*. Trea.