

Grabación y reproducción de sonido.  
Generación de sonido

# Introducción

- La **grabación y reproducción de sonido** consiste en la “inscripción” de ondas sonoras en algún medio para su posterior recreación.
- En la grabación digital de sonido se debe transformar una señal analógica en un formato digital mediante el procedimiento de [muestreo](#).
- El audio grabado digitalmente se puede almacenar en tres tipos básicos de [formatos](#):
  - Audio sin compresión: el más típico es el [WAV](#).
  - Audio con compresión sin pérdida: p.ej. [FLAC](#).
  - Audio con compresión con pérdida: el más típico es el [MP3](#).
- En Processing contamos con dos bibliotecas tradicionalmente usadas para grabación y reproducción de sonido: Sound (no recomendada) y Minim (recomendada).
- Si se usa Minim se emplearán las clases **AudioInput** y **AudioRecorder** para grabar sonido y las clases **AudioPlayer** y **AudioSample** para reproducir sonido digital.
- Puesto que la reproducción de sonido con Minim ya se cubrió en otra sesión en esta **solo veremos ejemplos de captura de sonido**.

# Introducción

- La generación o sintetización de sonido consiste en la producción de música mediante el procesamiento algorítmico de señales digitales.
- En Minim es posible generar sonido mediante el uso de los denominados UGens ([\*Unit Generators\*](#)).
- En Processing puede usarse otra biblioteca más avanzada denominada [Beads](#). De usarse esta biblioteca se recomienda la lectura del tutorial [\*"Sonifying Processing: The Beads Tutorial"\*](#).
- Difícilmente puede generarse audio digital de calidad sin unos mínimos conocimientos de música. Pueden encontrarse buenos ejemplos en [\*"Processing for Musicians"\*](#).



# Introducción

- Cuestiones que **no** vamos a tratar en la asignatura (pero tienen que ver con audio digital):
  - **MIDI (*Musical Instrument Digital Interface*)**: es un estándar para la interconexión de instrumentos musicales y otros dispositivos electrónicos (como ordenadores). Processing dispone de la biblioteca MidiBus. Cuestión aparte es la reproducción de archivos MIDI, un ejemplo del uso de Minim con el paquete MIDI de Java puede verse [aquí](#).
  - ***Text to Speech* (aka generación de habla)**: existen múltiples tecnologías para transformar cadenas de texto con lenguaje natural en habla. En Processing puede usarse la biblioteca [ttslib](#).




# Captura y grabación de sonido con Minim

- Estudiaremos los ejemplos:
  - `minim_teclado_grabadora.`
  - `minim_grabadora_micro.`
  - `minim_analysis.`

# Generación de sonido con Sonic Pi

- Las bibliotecas más habituales para generar sonido con Processing son [Minim](#) y [Beads](#).
- Se recomienda su estudio a aquellas personas que tengan unos conocimientos mínimos de música y, en esos casos, se valorará muy positivamente el desarrollo, no de un ejercicio, sino de un tutorial sencillo que pueda guiar a alguien sin conocimientos musicales.
- El resto de nosotros usaremos [Sonic Pi](#) :)
- Sonic Pi es una aplicación destinada a *live coding* pero, obviamente, puede usarse para la programación “tradicional” de piezas musicales.

# Generación de sonido con Sonic Pi

- A través de tres ejemplos de uso de Sonic Pi se cubrirán algunos de los aspectos más importantes de la herramienta.
- Dos de los proyectos nos acercarán a los conceptos de [audificación](#) y [sonificación](#):
  - Un script que aproxima el valor de  $\pi$  mediante la [fórmula de Leibniz](#). En el oído izquierdo “oiremos” a  $\pi$  y en el derecho cómo nos aproximamos a su valor real. 
  - Otro script nos mostrará la evolución de las concentraciones de CO<sub>2</sub>, óxidos nitrosos y metano en la atmósfera desde 1970 hasta 2012. Nos permite, además, ilustrar que podemos explotar las posibilidades del lenguaje Ruby desde Sonic Pi. 
- El ultimo es un proyecto más “musical” en el que se creará una pieza usando samples, ~~locuciones~~ y progresiones de acordes. 

# Generación de sonido con Sonic Pi

- Los ejemplos ilustran, por un lado, las posibilidades que ofrece Sonic Pi y, por otro, la cruda realidad: Sonic Pi **es** un instrumento musical y, como tal, requiere unos mínimos conocimientos de música por parte de quien lo emplea.
- A continuación se ofrecen algunos recursos online para poder profundizar en el uso de esta herramienta:
  - <https://sonic-pi.net/tutorial.html>
  - <http://www.daveconservatoire.org/lesson/synth-parameters>
  - <https://sonic-pi.mehackit.org/>
- **El tutorial de hoy está disponible [aquí](#).**
- Si el tutorial se os hiciera corto podéis echarle un ojo, además de a los 3 recursos anteriormente mencionados, a [Petal](#).



# Comunicando Sonic Pi con Processing y viceversa

- Tanto Sonic Pi como Processing implementan el protocolo [Open Sound Control](#) (OSC) que permite la interconexión de sintetizadores de sonido, ordenadores y dispositivos multimedia.
- Mediante OSC podéis controlar desde Sonic Pi visualizaciones programadas en Processing así como comunicar acciones que el usuario realice con un *sketch* de Processing con Sonic Pi para producir sonidos.
- Una ventaja de OSC es que podéis comunicar programas ejecutándose en ordenadores diferentes aunque en los ejemplos suelen ejecutarse en la misma máquina.
- A continuación se presentan dos ejemplos ilustrativos y no excesivamente complejos:
  - [A visualisation of a four part round of Frere Jaques.](#)
  - [Sonic Pi 3 visualizer.](#)

# Entregable

- Puede usarse cualquiera de las herramientas vistas o mencionadas en esta sesión: **Sonic Pi**, Minim o Beads.
- Se desaconseja entregar código que solo suponga un cambio incremental sobre tutoriales o ejemplos ya existentes.
- No obstante, hay que evitar complicar innecesariamente el ejercicio así que se valorará muy positivamente la brevedad del código y el uso mesurado de samples.
- Si alguien considera que puede elaborar un tutorial ilustrativo sobre alguna funcionalidad concreta de **Sonic Pi** o cualquiera de las bibliotecas también puede llevarlo a cabo como ejercicio.