Valentin Ramirez, mateo martinez, IGNACIO FERNANDEZ y PABLO RUIZ

universidad camilo jose cela

30 de septiembre de 2024

Live coding

Sonic pi

**ÍNDICE**

1. Ejercicio propuesto
2. Solución planteada
3. Problemas encontrados
4. Bibliografía

**Ejercicio propuesto**

El desafío consiste en realizar un concierto en vivo utilizando Sonic Pi, una herramienta de live coding que permite programar música en tiempo real. Para crear una canción en Sonic Pi, será necesario controlar varios elementos de forma simultánea, como la batería, el bajo, la melodía principal y otros sonidos, todo mientras se modifica el código durante la ejecución.

Para que la canción sea interesante y dinámica, se deberá manejar la sincronización entre los diferentes elementos, asegurándose de que los ritmos y las melodías entren y salgan en el momento adecuado. Todo esto será hecho en directo, mostrando el código en una pantalla semi-transparente, con la posibilidad de añadir visuales o videos de fondo para complementar la experiencia.

Es importante dominar la programación en Sonic Pi, el uso de variables para modificar parámetros como el tono o la duración de las notas, así como saber estructurar el código para que la canción evolucione de manera fluida a lo largo de la performance en vivo.

Sonic Pi utiliza como lenguaje de programación Ruby, un lenguaje de alto nivel que permite a programadores poco experimentados aprender de manera bastante intuitiva, a la vez que permite a programadores avanzados aplicar sus conocimientos a la música de manera bastante simple.

Nosotros nunca habíamos programado en Ruby pero se nos ha hecho muy fácil de aprender, es un lenguaje que se parece mucho al python en algunos aspectos, como los bucles o las funciones. Al ser un lenguaje interpretado permite actualizar en código en directo, mientras que corría el antiguo, rápidamente cambia sin detenerse aplicando las novedades del nuevo código ejecutado, esto nos permite ir aplicando el desarrollo de nuestra canción en directo, añadiendo y eliminando aquellos sonidos que elijamos mientras se mantienen los que no se han tocado

Para este proyecto elegimos programar una canción de *Tech House*, un subgénero de música de la rama de la música electrónica que busca un equilibrio entre la emoción y energía que te da el techno con la pero con un ritmo más bajo como es el *House* que te permite bailarlo de una manera más desestresada y fluir con un Groove más marcado. Para una canción de *Tech House* lo mas común es un tempo que suele oscilar dependiendo de la canción o el artista entre 120 y 130 BPM, por lo que para nuestro proyecto elegimos que el tempo fuese de 125.

**Solución planteada**

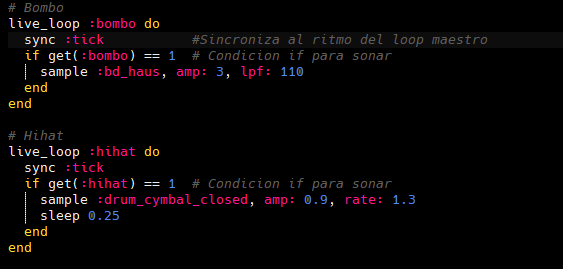
Para resolver el desafío de hacer un concierto en vivo con Sonic Pi, hemos desarrollado una estrategia que se basa en controlar varios elementos sonoros a la vez, sincronizarlos correctamente y modificar el código en tiempo real. Usamos live\_loops para manejar distintos patrones musicales como la batería, el bajo y la melodía, lo que nos permite ejecutarlos de forma simultánea

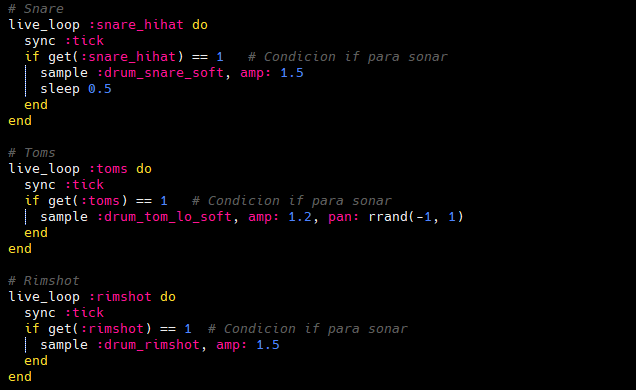
También, utilizamos variables dinámicas para ajustar cosas como el volumen, el tono y añadir efectos mientras estamos en directo, lo que nos da flexibilidad para cambiar el sonido sin detener la música. Además, el código se proyecta en una pantalla semi-transparente, acompañado de visuales o videos de fondo, lo que hace que la presentación sea más inmersiva.

Con esta estructura flexible, podemos ir añadiendo o quitando elementos según el desarrollo de la canción, logrando que la música evolucione de manera natural a lo largo de la presentación. Esta solución nos ha permitido mantener un control preciso de los sonidos y crear una experiencia en vivo dinámica y creativa.

De esta manera aplicamos una serie de loops que presentaremos a continuación que controlamos a través de un controlador que empieza con todos los valores de los loops a 0, y si los cambiamos a 1, los loops propuestos cumplirán una condición if que hará que ese instrumento en concreto comience a sonar.

**Percusión:**





Lo más importante de una buena canción de cualquier género de la música electrónica es una buena percusión que lo haga sentirse como el subgénero que sea, en este caso, para un *Tech House* es importante que la percusión marque un buen ritmo que marque su estilo sin sobrecargarlo de más como podría pasar en otros géneros como el *Hard Tech* o *Techno*, porque mas tarde en la canción cobran mucho más protagonismo otros elementos.

A continuación, explicaremos la función y aporte de cada uno de estos elementos de percusión a una canción de *Tech House.*

***Bombo:*** Es el corazón del track, marca el pulso de la canción proporcionando un golpe grave y consistente que establece el ritmo normalmente de 4x4. Establece la estructura rítmica de la que dependerán el resto de elementos de la canción

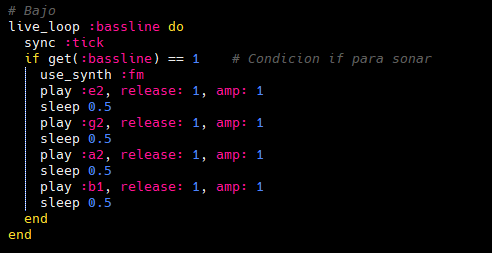
***Hihat:*** Añade un brillo y ritmo entre los beats del bombo de una manera menos violenta lo que le añade una capa de energía y movimiento, que hace que la canción se sienta mas rápida e agil

***Snare/Clap:*** Se suele colocar en el segundo y cuarto beat del compás dando un énfasis rítmico ayudando a marcar los acentos principales de un ritmo

***Toms:*** Los toms añaden una textura más tribal al track que aporta a la profundidad y variación del ritmo

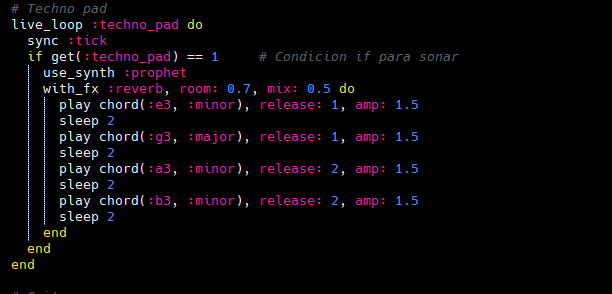
***Rimshot:*** El rimshot es un golpe corto y percusivo que añade detalles finos al ritmo. Es otra capa más de percusión, enriqueciendo el patrón rítmico sin sobrecargar el track.

**Bajo:**



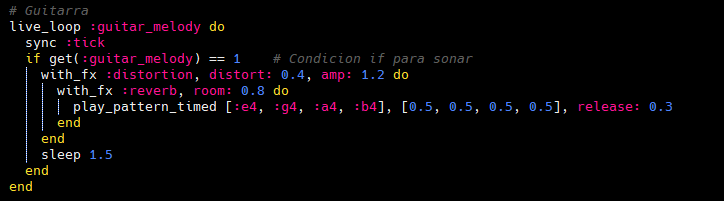
Nuestro primer instrumento aplicado es el bajo que le da un patrón melódico y rítmico que interactúa directamente con el bombo, creando el Groove que tanto define al *Tech House* tal y como lo conocemos. Aporta una energía y profundidad que impulsa el ritmo a otro nivel.

**Techno Pad:**



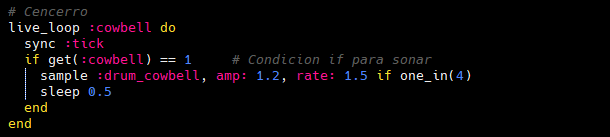
Los pads son sonidos de teclado electrónico muy típicos en el Tech House que añaden una capa emocional y vibratoria a la canción, generando ambiente y profundidad. Pueden generar tensión o relajación según la progresión armónica que se decida tomar

**Melodia de guitarra:**



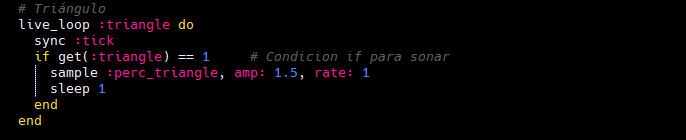
Proporciona una sensación más organica dentro de un track electrónico como lo es el *Tech House*, le da carácter y un toque más humano al asunto.

**Cencerro:**



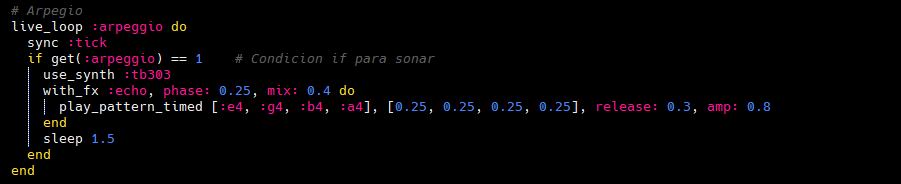
Añade sabor y textura rítmica, generando momentos divertidos o destacando ciertos pasajes del track. En *Tech House*, los cencerros pueden dar un toque tribal o festivo. Solo suena en el cuarto compás.

**Triangulo:**

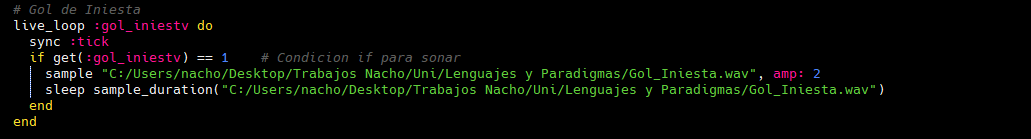


Aporta pequeños toques de brillo y claridad al ritmo, sin interferir con los elementos más pesados como el bombo o el bajo.

**Arpegio:**



El arpegio consiste en una secuencia de notas que se tocan en sucesión rápida. Es muy común en el tech house para añadir un componente melódico que se repite y evoluciona.

**Sample gol de Iniesta:**

Para darle vida y contenido a la canción le hemos añadido un sample de la narración del gol de Iniesta en el mundial para añadir ese toque de euforia antes de que rompa la canción

**Problemas encontrados**

1. Aprender a hacer música

Nuestro principal problema a la hora de empezar a intentar hacer una canción es que ninguno de los 4 integrantes del grupo es DJ, por lo que a pesar de ser grandes aficionados a distintos estilos de música, nunca nos habían puesto en la situación de intentar crear una canción, por lo que ha sido todo un reto para nosotros, para empezar a entender cómo funciona la aplicación nos vimos un tutorial de YouTube, en el cual empezamos a entender que una canción trata de varias “capas” de instrumentos los cuales deben entrar coordinados en el compás. Una vez comprendimos la magia de crear música pudimos empezar a plantear como queríamos que fuese nuestra canción.

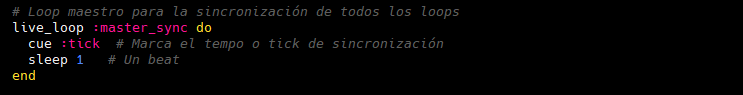
Aunque le pedimos a chat gpt que nos programase una serie de notas para tocar con estos instrumentos la realidad es que ninguno suena del todo bien, y juntos ya deja mucho que desear. Por eso es que buscamos librerías de samples de Tech House y elegimos un par de samples más para utilizar en la canción.

1. Sincronización de nuevos sonidos con los demás sonidos existentes

El segundo problema que tuvimos fue la sincronización de los nuevos sonidos para ello utilizamos la función “sync”. Esta función permite que un *live\_loop* espere a que otro live\_loop envíe una señal, asegurando que ambos empiecen o se mantengan sincronizados correctamente.

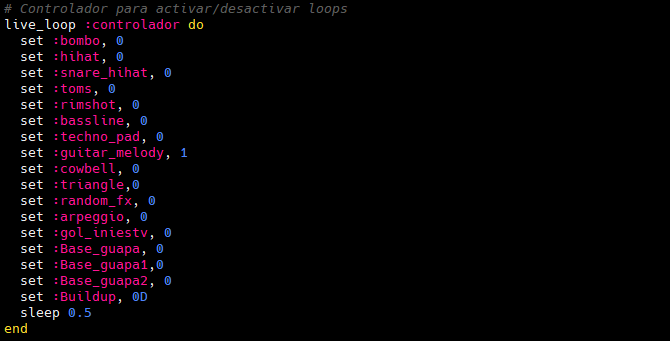
Esta función nos facilitó mucho la creación de la canción y nos ha permitido evitar desajustes entre los sonidos, manteniendo la coherencia rítmica y asegurando que cada nuevo elemento se integre de forma fluida en la canción. También nos ha permitido experimentar con diferentes combinaciones de sonidos y efectos, asegurando que todo siga una estructura sincronizada sin esfuerzo adicional.

Para seguir el compás indicado creamos un loop que generaba un compás universal que se sincronizaba con todos los demás loops.



1. Controlador de instrumentos

Una vez teníamos la sincronización de todos instrumentos, lo único que nos faltaba era un controlador para poder añadir y eliminar instrumentos según nuestro antojo. Al principio dependíamos de un stop en cada bucle el cual comentábamos o quitábamos el corchete haciendo que dejase de sonar ese loop en concreto. Pero viendo que era una manera poco efectiva de aplicar cambios a nuestra canción, asignamos un valor 0 a todos los loops del código y una condición if que haría que la parte de dentro del loop comenzase a ejecutarse, solo si cambiábamos desde el controlador el valor del loop a 1.



**Bibliografia**

**Paginas web utiles:**

<https://sonic-pi.mehackit.org/exercises/es/01-introduction/01-introduction.html>

<https://sonic-pi.net/tutorial.html>

AI:

<https://chatgpt.com/>

Videos de YT:

<https://www.youtube.com/live/yA7epoqrh_o?si=_Tj22D4wiCZpM7bx>

<https://youtu.be/nCpd9rxvgDo?si=vsWfv-ZQlCkkLuPn>

[](https://youtu.be/nCpd9rxvgDo?si=vsWfv-ZQlCkkLuPn)