UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Aplicaciones inmersivas dentro de plataforma Web: ejercicios de interpretación de flauta

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Navarro Huerta, Álvaro

Tutor/a: Agustí Melchor, Manuel

CURSO ACADÉMICO: 2023/2024

Resum

Per als alumnes de primària, aprendre a tocar un instrument musical pot resultar una tasca complicada, amb hores de pràctica limitades a l'escola i massa alumnes per aula, el professor pot no tenir temps per corregir-los a tots. Per això és necessari la pràctica a la llar, però per a un nen resulta difícil, aprendre i memoritzar les posicions dels dits correctes. Les aplicacions estàtiques per a l'ensenyament poden ser avorrides per als infants, per això es planteja el desenvolupament d'una aplicació interactiva que emprant el WebAudio API i un model 3D resulti més atractiva als ulls del estudiant.

Paraules clau: Aplicacions immersives; Web; realitat virtual; realitat augmentada; interpretació de flauta, exercicis daprenentatge; autoaprenentatge.

Resumen

Para los alumnos de primaria, aprender a tocar un instrumento musical puede resultar una tarea complicada, con horas de practica limitadas en la escuela y demasiados alumnos por aula, el profesor puede no tener tiempo para corregirlos a todos. Por eso es necesario la practica en el hogar, sin embargo para un niño resulta difícil, aprender y memorizar las posiciones de los dedos correctas. Las aplicaciones estáticas para la enseñanza pueden resultar aburridas para los infantes, por eso se plantea el desarrollo de una aplicación interactiva que empleando el WebAudio API y un modelo 3D resulte más atractiva a los ojos del estudiante.

Palabras clave: Aplicaciones inmersivas; Web; realidad virtual; realidad aumentada; interpretación de flauta, ejercicios de aprendizaje; autoaprendizaje.

Abstract

For elementary school students, learning to play a musical instrument can be a complicated task, with limited hours of practice at school and too many students per classroom, the teacher may not have time to correct them to everyone. That is why practice at home is necessary, however for a child finds it difficult to learn and memorize the correct finger positions. Static teaching applications can be boring for students. infants, that is why the development of an interactive application that using the WebAudio API and a 3D model is more attractive to the eyes of the student.

Key words: Immersive applications; Web; virtual reality; augmented reality; flute performance, learning exercises; self-study.

Índice general

Índice de figuras

Índice de tablas

CAPÍTULO 1
Introducción

Aprender a tocar un instrumento musical ha demostrado tener grandes beneficios en el desarrollo de los niños ya sea a nivel físico, como la mejora de la coordinación y habilidades motoras; a nivel emocional, ya que puede promover el bienestar emocional y el desarrollo de habilidades sociales saludables; como a nivel mental ya que la práctica musical también mejora la memoria, la atención y la capacidad de concentración.

La flauta dulce es el instrumento por excelencia en el aprendizaje musical en las escuelas primarias debido a varios factores que la hacen idónea como instrumento de iniciación para los niños. El primero de ellos es su accesibilidad, ya que es muy asequible económicamente y es fácil encontrar modelos baratos de calidad adecuada. Otras ventajas que presenta es que es ligera y transportable, manejable y que presenta un sonido suave y dulce que la hace agradable para los niños. A las anteriores se le suman que es sencilla para comenzar a tocar, promueve el desarrollo de la respiración de forma correcta y fomenta la coordinación ojo-mano.

Sin embargo, para los alumnos de primaria, aprender a tocar este instrumento musical puede resultar una tarea complicada, ya que el aprendizaje musical requiere una práctica regular y dedicación, lo cual puede ser difícil de encajar dentro del horario escolar. Establecer una rutina de práctica es importante para avanzar y mejorar, pero las horas destinadas a la práctica de un instrumento musical están muy limitadas en la escuela primaria. Existe además el problema añadido de la sobresaturación, con demasiados alumnos por aula, el profesor se ve incapaz de dedicar a los estudiantes el tiempo necesario a cada uno de ellos, con dificultades añadidas a los alumnos con necesidades especiales.

Por estos motivos se hace necesario una rutina de práctica en el hogar, sin embargo para un niño resulta difícil, aprender y memorizar las posiciones de los dedos de forma correcta sin una guía apropiada.

El aprendizaje de la práctica de flauta dulce mediante libros en casa puede presentar varios problemas añadidos para los infantes. Los libros de texto pueden ser utilizados de una manera pasiva, sin la interacción y la guía necesarias de un profesor, y esta circunstancia puede conducir a una mala comprensión de la materia y a la larga a una falta de motivación. Cabe destacar que los libros carecen de formas de estimulación sensorial necesarias para el aprendizaje de algunos niños, tales como los juegos con sonidos o el desarrollo de habilidades por

imitación, lo que limita la capacidad de los niños para mantener su atención en el proceso. Además los libros de texto no son personalizables, por lo que su contenido podría ser excesivamente difícil o fácil, y no pueden tener en cuenta las necesidades especificas de cada niño por lo que puede causar frustración en ellos y conducir al abandono de la práctica.

Por los motivos anteriores en este trabajo se plantea el desarrollo de una aplicación interactiva que empleando el Web Audio API y un modelo 3D pueda resultar más atractiva a los ojos del estudiante, facilitando su aprendizaje y su adhesión a la práctica.

Las tecnologías inmersivas se han popularizado recientemente a partir de la creación de Occulus en el año 2012. Estos dispositivos están diseñados para crear una experiencia de inmersión en la realidad virtual y presentan una alta resolución y un amplio campo de visión.

Desde entonces grandes empresas han apostado por esta tecnología. Este tipo de tecnología intenta replicar experiencias reales mediante el uso del software.

Dependiendo de la aplicación se agrupan en sistemas de realidad virtual y sistemas de realidad aumentada.

La realidad virtual que desarrolla entornos completamente digitales mediante el uso de software especializado. El objetivo es crear una experiencia completamente inmersiva. En este entorno virtual se interactúa mediante el uso de unas gafas o cascos de VR, que cubren los ojos , y son dispositivos más grandes y complejos; y controladores.

Respecto a la realidad aumentada, esta toma capturas del mundo real y les agrega una representación 3D de un objeto sobre la imagen. El objetivo es mejorar la experiencia en el mundo real, agregando elementos virtuales que agreguen información. En este entorno los objetos pueden ser agregados con o sin marcadores, con marcadores detecta el marcador y sitúa sobre él un objeto. Sin marcador busca una superficie plana para colocar el objeto 3D.

Con esta tecnología se pretende crear un entorno de realidad virtual donde el alumnado pueda aprender mediante una serie de ejercicios de flauta, fácilmente adaptable a su nivel y necesidades específicas de aprendizaje.

1.1 Motivación

Durante los años mas recientes, muchas empresas están optando cada vez más por modelos de negocio orientados a la programación en la nube. Esta tendencia provoca que los programadores tengan que incorporar conocimientos sobre desarrollo web más avanzados. Así pueden dar más visibilidad a sus habilidades. Para mejorar estas habilidades se ha planteado este trabajo, con el que se pretende además ayudar a los estudiantes que quieran mejorar su habilidad musical.

Las tecnologías de realidad virtual resultan muy interesantes de aprender y desarrollar, pero requiere de componentes que todavía no se encuentran disponibles en la mayoría de hogares. Por este motivo se pretende emplear la inmersión mediante el uso de una cámara en primera persona, similar al de muchos video-

juegos. No obstante, el entorno a desarrollar para el uso de la tecnología de realidad virtual resulta similar pues emplea un entorno tridimensional que puede ser adaptado a partir de escenas sin caracteristicas de realidad virtual.

1.2 Objetivos

El objetivo de este trabajo es desarrollar una aplicación multimedia para la web, esta aplicación consistirá en un entorno 3D en el cual se reproducirá una partitura musical para flauta, dicha partitura será creada por el usuario o leída desde un archivo. A la vez se dibujará una flauta y unas manos que acompañaran el sonido reproducido

Para ello se proponen diferentes objetivos para este proyecto

1. Desarrollar un entorno inmersivo 3D en el que el usuario se pueda ver representado y tomarlo como referencia, para familiarizarse y aprender a usar un instrumento.
2. Desarrollar un sistema de sonido en un entorno 3D
3. Desarrollar una interfaz en el entorno para la carga e interpretación de ese audio, sincronizando como es animada junto con la reproducción.

1.3 Estructura de la memoria

Para ordenar los objetivos expuestos en el documento se va a estructurar en capítulos.

- El primer capitulo contiene la introducción al tema y los objetivos a lograr durante el trabajo.
- El segundo capitulo llamado estado del arte, hablará sobre distintas tecnologías y otros proyectos relacionados que se han consultado para la creación de este proyecto.
- El tercer capitulo habla sobre distintas herramientas para el desarrollo del trabajo.
- El cuarto capitulo abordará el problema propuesto para el trabajo y realizará un desglose del mismo en tareas.
- El quinto capitulo expone el despliegue del proyecto sobre un servidor y las pruebas sobre diversos navegadores.
- El sexto capitulo realiza una conclusión al trabajo y expone futuras ampliaciones.

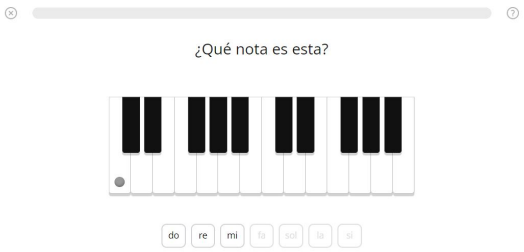
CAPÍTULO 2
Estado del arte

La enseñanza musical puede diferenciarse mediante distintos modelos musicales. El más común en las escuelas es el modelo académico, este propone como forma de adiestramiento la lectoescritura musical a través de la explicación de obras musicales canonizadas y consagradas en el tiempo. En este modelo no se tienen en cuenta las ideas ni intereses de los estudiantes pues el conocimiento se considera la referencia central. El aprendizaje se realiza escuchando, asimilando conceptos, dominando la lectoescritura musical y la teoría musical. Incluye un corpus de repertorio especifico y favorece la memorización de información de carácter anecdótico o episódico o de conocimiento enciclopédico. El conocimiento musical académico es establecido por instancias superiores, en función del conocimiento por la tradición, en la que el profesor tiene un rol técnico de ejecutar las propuestas elaboradas por otras personas. En el modelo práctico sugiere la enseñanza un objetivo práctico desligado del significado comunicativo y cultural. Esta visión subjetivista considera como relevante la satisfacción personal ligada a la música, especialmente al disfrute. En este modelo el profesor puede elegir otros materiales más atractivos para el aprendizaje fomentando las actividad creativa[1].

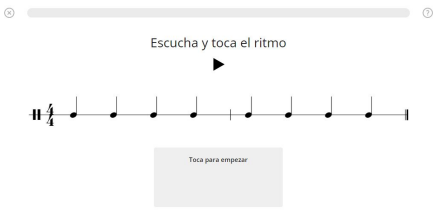
La aplicación propuesta encaja en el modelo más practico de la enseñanza al permitir el desarrollo de partituras, usando las tecnologías disponibles para la web con la misión de dar mayor accesibilidad. Las tecnologías móviles, estandarizadas entre la población más joven, abren más oportunidades de aprendizaje autodidacta. Para un estudiante puede resultarle más sencillo acudir a herramientas externas más interactivas.

Ejemplos de ejercicios de teoría musical, el sitio web "Musica"[6] propone ejercicios de lectura, escritura e interpretación interactivos realizados sobre HTML5 con uso de CSS y JavaScript para componer la página web con un diseño Responsive"() que permite ser utilizada también desde un dispositivo móvil. Este sitio permite acceder a diferentes niveles de ejercicios, véase la figura 2.1a o ser capaz de seguir el ritmo 2.1b, etc. El resultado de los ejercicios se pueden guardar, se se registra una cuenta para poder llevar el registro del progreso del alumno.

Otra web incluyen una ayuda visual a la hora de utilizar el instrumento[7]. Estas web proponen ejercicios para el alumno, pero carecen del elemento inmersivo e interpretativo que se propone para este proyecto. La figura 2.2 muestra un ejemplo de juego en el que adivinar la nota, mientras que la figura 2.3 muestra



(a) Escoger la tecla que ejecuta una nota.



(b) seguir el ritmo de una interpretación

Figura 2.1: Ejemplo ejercicios web

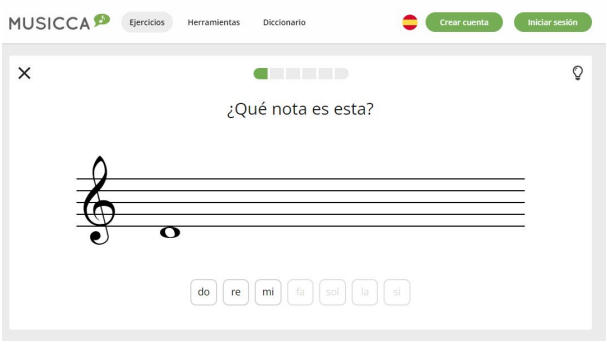


Figura 2.2: Ejercicio de la web https://www.musicca.com/es.

una partitura donde el alumno tiene que pulsar el botón correspondiente a la nota destacada. Este tipo de ejercicios son similares a los que proponemos desarrollar desde este documento y pueden servir como fuente de inspiración. Al utilizar una partitura con notas ayuda al alumnado a identificar la posición de la misma para la memorización.

En aplicaciones de escritorio, Musescore, de software libre permite crear partituras para una gran variedad de instrumentos y reproducirlas. Es ampliamente usada por profesionales. En 2017 paso a formar parte del grupo Muse Group. El problema de esta aplicación para con nuestro proyecto resulta de la abrumadora cantidad de opciones para usuarios sin experiencia en el campo de la creación musical. Sin un conocimiento previo, muchos de los iconos de la interfaz no son comprensibles. No obstante esta aplicación puede resultar útil para identificar elementos de diseño. Esta aplicación suple la necesidad de diseñar una partitura y se centra en la creación de melodías que pueden ser reproducidas. Para los alumnos es una herramienta que fomenta su creatividad, sin embargo no enseña el nombre de las notas ni contiene el aspecto inmersivo que se espera conseguir en este proyecto. Un ejemplo de esta aplicación se puede ver en la figura 2.4.

Para la realización de este trabajo se ha tomado como ejemplo de partida la búsqueda de trabajos de temática similar, para empezar este trabajo podría considerarse una ampliación del trabajo desarrollado por Víctor Hernández [12], en ese trabajo se desarrollaba una aplicación web y una aplicación del WebAudio API, nuestro trabajo es similar añadiendo una dimensión tridimensional al desarrollo, permitiendo que el usuario pueda tener una herramienta mas intuitiva para el aprendizaje musical. Este proyecto permite la reproducción de una melodía breve que puede ser compuesta al pulsar sobre la partitura o cargada a través

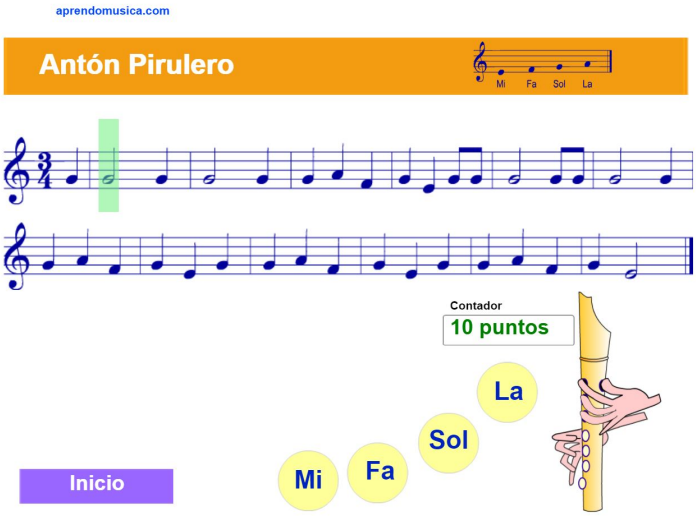


Figura 2.3: Ejercicio de la web https://aprendomusica.com/.

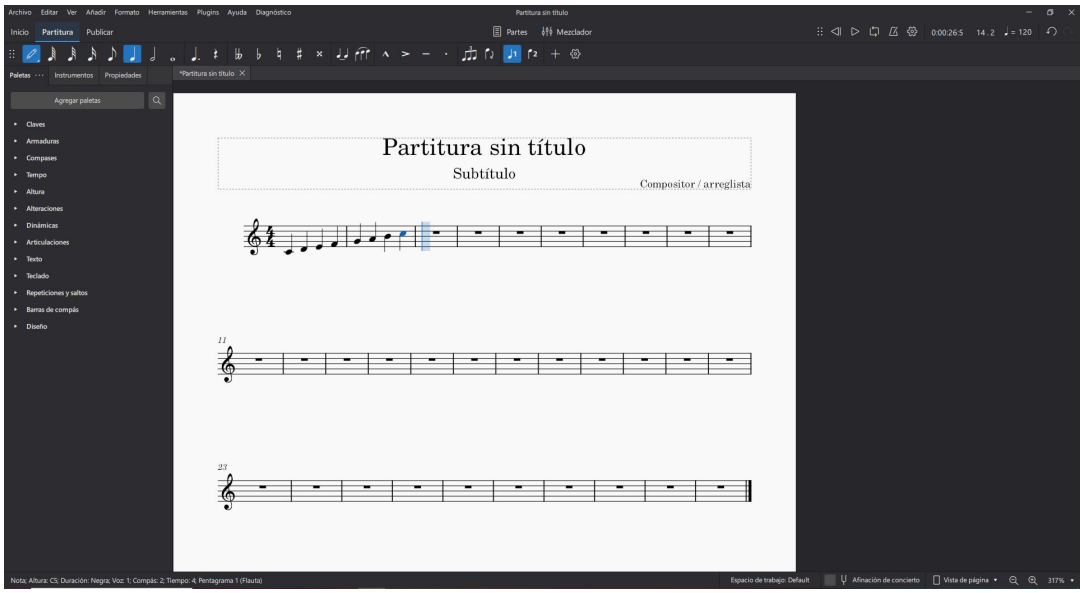


Figura 2.4: Ejemplo aplicación Musescore 4.

de un archivo. De este proyecto podemos extraer ideas para el desarrollo de la ejecución del programa y para organizar las notas sobre un fichero. un ejemplo de la aplicación es mostrado en la figura 2.5.

Otro ejemplo que se ha examinado ha sido el propuesto por Jorge Alcañiz [13], en este proyecto se propone una aplicación en la que se realiza un reconocimiento OCR para partituras musicales, este proyecto podría añadir valor al propuesto en este trabajo, al permitir mayor rapidez a la hora de implementar partituras en el programa. En su trabajo además propone diversos ejercicios para que los alumnos puedan adquirir conocimiento sobre la identificación de notas, pues permite que el alumnado relacione cada símbolo sobre el pentagrama con la nota correspondiente. Un ejemplo de los ejercicios propuestos es el expuesto en la figura 2.6 donde se asocia cada nota con el nombre correspondiente.

Un último ejemplo es el trabajo propuesto por Carlos Durán[11], esta aplicación propone la creación de una aplicación web que permita la visualización

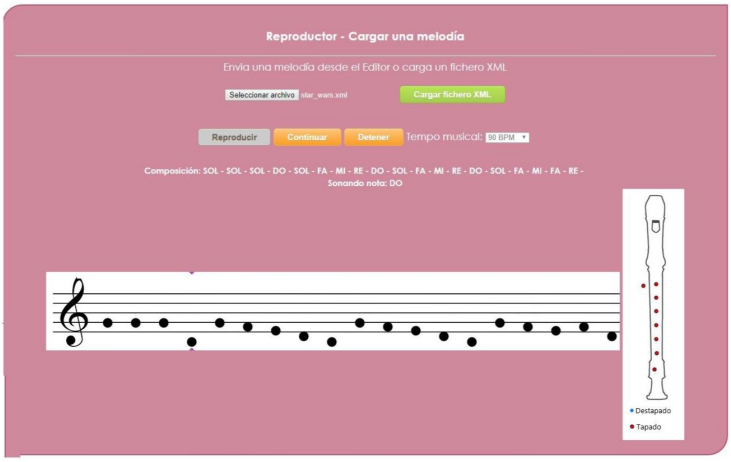


Figura 2.5: Ejemplo aplicación web Victor Hernandez.

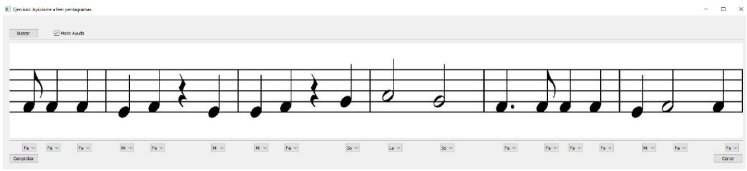


Figura 2.6: Ejemplo ejercicio aplicación Jorge Alcañiz.

de objetos 3D, en su caso piezas de obras de arte. Usando un sistema de base de datos para manejar la carga de las piezas de la aplicación como la gestión de usuarios. En su caso emplea el framework Ruby on Rails por haberlo empleado antes para otros proyectos. Un ejemplo sobre una figura de la aplicación propuesta es mostrado en la figura 2.7.

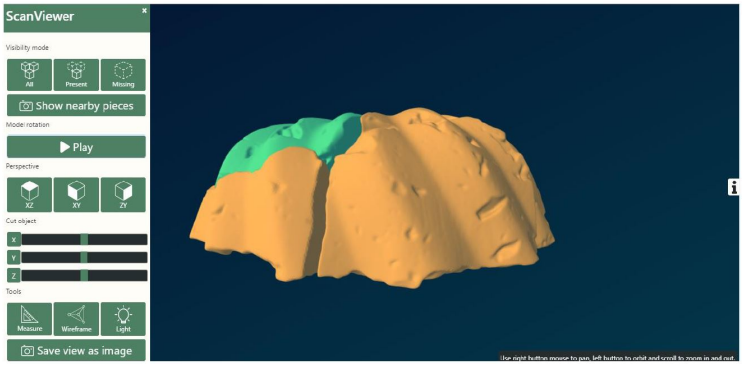


Figura 2.7: Ejemplo aplicación Carlos Durán.

Existen en el mercado de realidad virtual muchas aplicaciones que sirven como reproductores de piano y de ayuda para tocarlo[34][35].Ademas hay para crear diversas mezclas de sonidos[33]. Pero podría resultar más compleja para novicios. No existen reproductores para la flauta dulce que haya encontrado, la aplicación más similar a nuestra visión está desarrollada para piano[36]. Un ejemplo de esta aplicación esta mostrada en la figura 2.8.

Los tipos de ejercicios vistos en [6] son interesantes desde el punto de vista del nivel que los objetivos de este trabajo quieren presentar. Aunque deben adaptarse para nuestro propósito, para lo que hay que redefinir cómo se presentan en base al

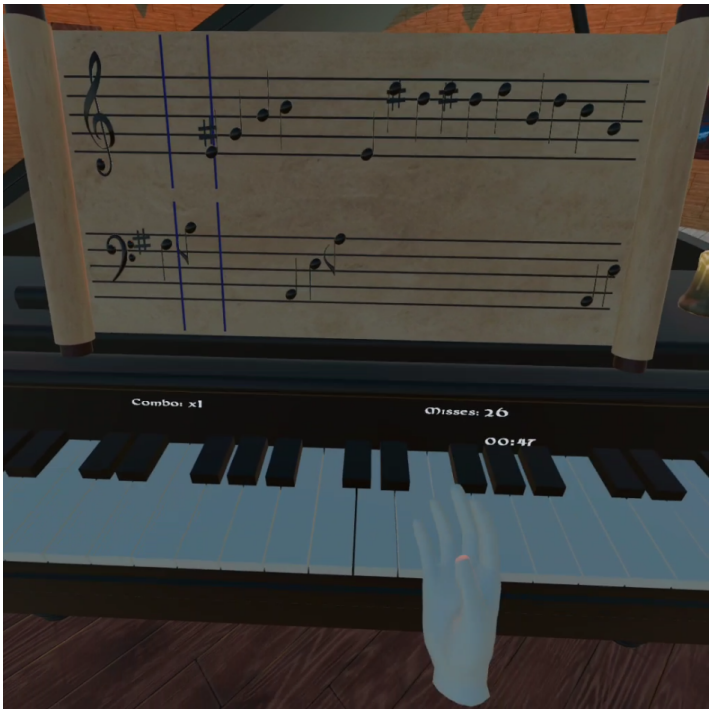


Figura 2.8: Ejemplo aplicación VR.

uso de la flauta como elemento y cómo se emplean técnicas que hagan al alumno sentirse más protagonista haciendo mayor énfasis en la interfaz visual y el uso del audio para crear una atmósfera inmersiva. La idea de guardar el progreso del alumno será evaluada a lo largo de este trabajo, aunque no es un objetivo del mismo.

CAPÍTULO 3
Herramientas de desarrollo

Vistos los objetivos que nos hemos propuesto y a raíz de la revisión del capítulo de .estado del arte.es necesario conocer y estudiar esas tecnologías y herramientas que se han utilizado en esos productos. Ya hemos establecido que las características de nuestro trabajo demandan la realización de una aplicación en entorno web, por lo que será necesario emplear técnicas de desarrollo web, se tomara el rol de un diseñador front-end cuyo trabajo consiste en realizar la parte de cara al usuario.[19]

3.1 Imagen 3D en páginas web

La introducción del estandar HTML5 ha permitido la introducción de elementos en 3d en las páginas web y diversas tecnologías que permiten su visualización.

La primera de ellas es X3D [45] esta herramienta ha sido desarrollada por el Web3D Consortium [18] que sirve como evolución sobre la base de Virtual Reality Modeling Language (VRML) y que ha desarrollado su propio estándar desde 2004. Esta integrado en HTML5 y no requiere de aplicaciones externas para funcionar en navegadores. Trata a los elementos como si fueran los elementos para crear un SVG. Este modelo solo acepta modelos 3d con formato .x3d. Como es mostrado en la tubería de procesos los modelos deben llevar un proceso de conversión para ser mostrados en los navegadores web3.1.

Al realizar una prueba, lo primero que indica es que para crear nuestra escena necesitaremos o un navegador html5 o un motor de renderización con funcionalidad completa para X3D[46]. Elegimos la primera opción, al tener el navegador

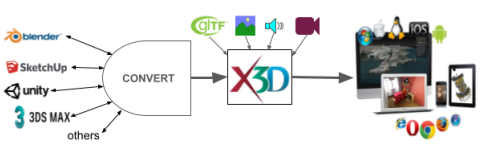


Figura 3.1: Cubo mediante X3DOM

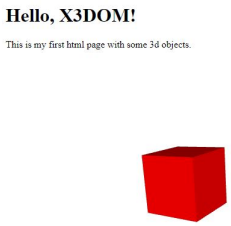


Figura 3.2: Cubo mediante X3DOM

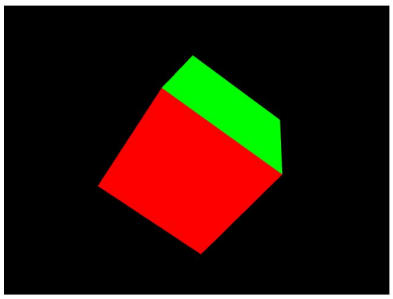


Figura 3.3: Cubo mediante WebGL

ya instalado. Este ejemplo muestra como la librería de x3dom es cargada junto con la página y el código se ejecuta mediante lenguaje de marcas. Figura 3.2

Otra opción para realizar desarrollos en 3d es mediante una API llamada WebGL [22] [3] [5],desarrollada por el Khronos Group, que permite ejecutar códigos de control sobre la GPU. Esta se añade como un elemento canvas sobre la web y mediante javascript será añadido el contexto que permite la visualización.[2]

WebGL permite al contenido web emplear una API basada en OpenGL 2.0 para realizar visionados 3D en HTML sin el uso de plugins. Consiste en un código de control escrito en JavaScript y un código de efectos especiales (shader code) ejecutado en la Unidad de Procesamiento Grafico (GPU). Los elementos de WebGL pueden ser combinados con otros elementos HTML y compuestos junto con otras partes de la página. Se ha seguido el tutorial [30] y se muestra como se crea el contexto de la aplicación sobre código javascript. El resultado de este ejemplo es mostrado en la figura 3.3

3.1.1. WebGL

La habilidad de WebGL para integrarse sin problemas con los estándares web la convierte en una herramienta inmersiva para los desarrolladores web. Con la capacidad de renderizar modelos complejos 3D en tiempo real, con iluminación realista, sombras y propiedades de físicas. Se utilizará esta tecnología para el desarrollo de este proyecto y se tratará por tanto de explicar el funcionamiento de la misma.

Lo fundamental en el proceso de ejecución es definir los shaders, estos son programas que toman la información de los vértices que genera una forma y ge-

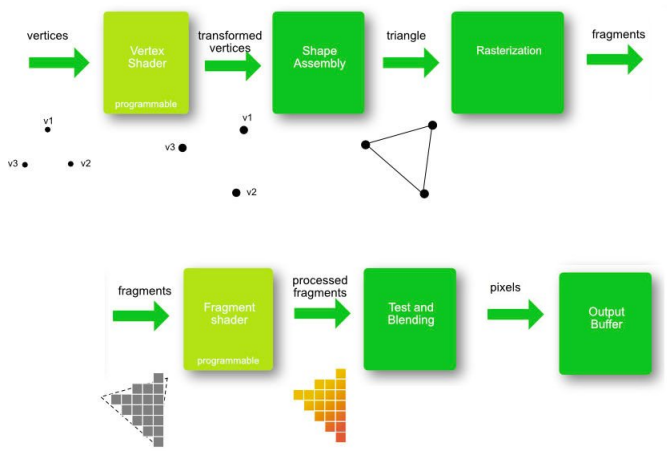


Figura 3.4: WebGL pipeline

nera la información necesaria para pintar los píxeles en la pantalla,están escritos en OpenGL Shading Language.

Hay dos funciones ejecutadas al general el contexto WebGL el vextex shader y el fragment shader.

Cada vez que tiene que renderizarse una figura, se hace una llamada al vertex shader. Este se encarga de transformar el punto de sus coordenadas originales a las coordenadas de la GPU, este sistema es conocido como clip space, es conformado por un cubo que va desde el punto (-1,-1,-1) a (1,1,1). Todo lo que este fuera de este fuera de este cubo no es renderizado y si existe un punto en el interior que conecte con el exterior el triangulo es descompuesto en otros más pequeños.

La transformada de los vértices se guardará en una variable especial llamada `gl_position`. El vertex shader también puede aplicar la textura del téxel que se aplica al vértice y los cálculos para la iluminación.

Tras ese proceso se produce un proceso de rasterización, esto es convertir las coordenadas tridimensionales en coordenadas 2d convirtiendo las formas en sets de fragmentos.

El fragment shader es llamado por cada píxel, de cada forma que debe ser dibujado después de haber sido procesado por el vertex shader. Debe averiguar que téxel aplicar al píxel y luego aplicar la iluminación apropiada al color. El color se devuelve a la capa WebGL y se almacena en la variable especial `gl_FragColor`

Un ejemplo de la pipeline de este proceso se puede encontrar en la figura 3.4 [22]

A pesar del potencial de WebGL manejar esta herramienta resulta en un trabajo muy complejo que requiere de demasiado esfuerzo. Por ello se va a recurrir al uso de librerías que operan sobre WebGL. Se exploran distintas opciones.

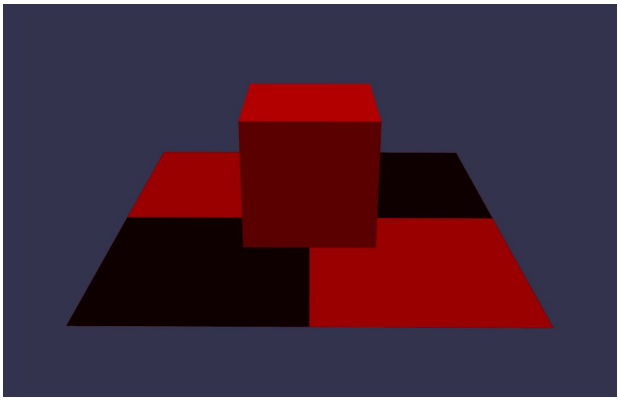


Figura 3.5: Cubo mediante Babylon

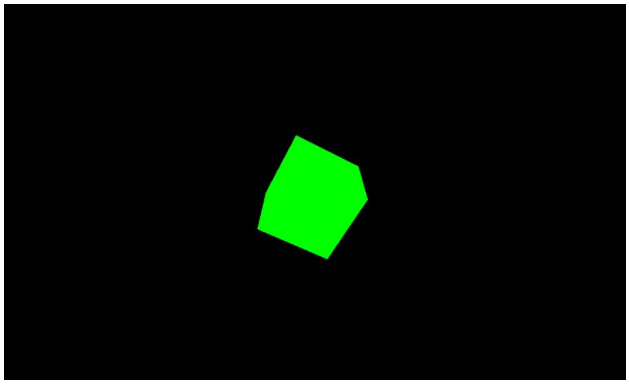


Figura 3.6: Cubo mediante Three.js

3.1.2. Babylon.js

La librería Babylon.js ha sido desarrollada por Microsoft y David Cathue como un motor 3D que puede realizar imágenes fotorrealistas y juegos interactivos. La ultima versión puede funcionar en cualquier motor que soporte WebGL2.0

Al buscar en la documentación de la librería,indica que dispone de un playground online, que se centra en el desarrollo de la escena. Pero la documentación esta organizada de forma más obtusa y por ello se descarta esta librería. Mostramos el primer ejemplo del tutorial para compararlo con el resto de figuras. Figura 3.5

3.1.3. Three.js

La librería Three.js creada por Ricardo Cabello es una librería Javascript que funciona como una API para el desarrollo de aplicaciones sobre WebGL.

Se ha decidido probar esta librería para realizar el ejemplo de crear un cubo. La librería Three[14] ofrece dos opciones, descargar los archivos de la librería o usar una CDN. Se elige esta ultima opción pues las dos anteriores también han hecho uso de ellas. El ejemplo necesita de un servidor local que reparta los archivos al navegador. Se debe instalar node.js y ejecutar serve para que funcione el ejemplo. El ejemplo incluye una animación de rotación básica. Figura 3.6

Comparando estas librerías con el desarrollo sobre WebGL puro facilita mucho la tarea pues muchas de las figuras más simples ya estan declaradas y no es necesario crear los puntos directamente, por otra parte se diferencian mejor los elementos que conforman la escena y encapsula los cálculos matriciales.

3.2 Audio en páginas web

Dentro del contexto de HTML5 tenemos un elemento <audio>[20] que proporciona una base para incorporar ficheros de audio y cuya reproducción depende de la implementación que cada navegador haga de las recomendaciones del W3C, además algunas bibliotecas que proporcionan funcionalidades relacionadas con el uso del audio. Una especificación de uso de audio para desarrollo de aplicaciones web puede ser realizada mediante el uso de Web Audio API [21] esta API permite manejar las operaciones de audio sobre un contexto de audio, y ha sido diseñado para aceptar un grafo de encaminamiento modular, es decir, las operaciones de audio son llevadas a cabo en nodos, que están enlazados juntos para formar un grafo de encaminamiento de audio. Así permite crear el audio puede ser computado matemáticamente o mediante el uso de grabaciones de audio.

Se ha explorado el uso de la creación de sonidos. La librería Tone.js [44] se usa para generar sonidos de sintetizadores y efectos, se han creado modelos para algunos instrumentos, pero no hay creado ninguno para flauta actualmente.

Sin embargo debido a las características propias de la aplicación a desarrollar se emplearan grabaciones para resultar en sonidos más naturales.

3.3 Herramientas para desarrollo de aplicaciones web

La complejidad de una aplicación web con los objetivos propuestos y el uso de los medios que acabamos de revisar nos llevan a un contexto de desarrollo complejo. Introducir todas estas características en una aplicación necesita un entorno de desarrollo más parecido al que se venimos utilizando al desarrollar una aplicación nativa para un computador, con las características propias de un desarrollo a ejecutar sobre un cliente web, con características de uso de bibliotecas de funciones, dependencias, compilación, creación del distribuible y pruebas de ejecución en diferentes navegadores.

En particular hay dos grandes pilares:

a) El uso de módulos en JavaScript b) El entorno de desarrollo donde llevar a cabo la implementación, desarrollo, depuración y generación del distribuible.

3.3.1. Módulos en JS

Los programas JavaScript comenzaron siendo bastante pequeños — la mayor parte de su uso en los primeros días era para realizar tareas de scripting aisladas,

proporcionando un poco de interactividad a tus páginas web donde fuera necesario, por lo que generalmente no se necesitaban grandes scripts. Ahora tenemos aplicaciones completas que se ejecutan en navegadores con mucho JavaScript.[50]

Un módulo es un archivo de JavaScript que agrupa funciones, clases, variables que luego pueden ser exportadas y utilizadas en otras partes de nuestra aplicación. Un módulo permite ocultar funcionalidad del mismo y solo exportar aquello para lo que ha sido implementado.[51]

Esta característica agregada a JavaScript nos permite implementar programas mucho más ordenados y facilita la reutilización del código.

3.3.2. Entorno de desarrollo

Desde las páginas de Three.js se recomienda Vite [40] pero existen alternativas a Vite que son más o menos populares dependiendo del contexto de la aplicación web a desarrollar por las características de estos entornos

Lo que hace que Vite destaque es su enfoque de empaquetado bajo demanda. En lugar de pre-empaquetar todo el código y los activos, Vite aprovecha los módulos ES nativos de los navegadores modernos, sirviendo el código directamente al navegador durante el desarrollo. Esto conduce a una Sustitución de Módulos en Caliente (HMR, Hot Module Replacement) casi instantánea y a tiempos de arranque en frío reducidos.

El servidor de desarrollo de Vite brilla con este enfoque bajo demanda, permitiendo a los desarrolladores ver los cambios rápidamente sin una recompilación completa. También utiliza Rollup, para unas construcciones de producción eficientes. Como resultado, Vite ofrece un desarrollo rapidísimo y un sólido rendimiento en producción.

Webpack es la piedra angular del desarrollo web moderno, en constante evolución desde 2012. Lo mejor de Webpack es cómo organiza los componentes del sitio web. Optimiza los tiempos de carga y la experiencia del usuario organizando el código en módulos.

La adaptabilidad de Webpack es una ventaja notable. Los desarrolladores pueden personalizar los proyectos para tareas simples o complejas. Permite a los desarrolladores adaptar los flujos de trabajo y construir procesos con precisión.[41]

CAPÍTULO 4
Identificación y análisis de soluciones

4.1 Planificación temporal

Para realizar el proyecto se ha diseñado un diagrama de Gantt para establecer los apartados por los que se espera que pase el proyecto y establecido un limite de tiempo para cada tarea. Se muestra en la figura 4.1.

En primer lugar se busca que tecnologías existen para el desarrollo de gráficos 3D sobre un navegador. Se estudiarán que ejemplos pueden ser adaptado para nuestra aplicación. Como inicio se trata de emplear la menor cantidad de librerías para ir añadiendo al desarrollo según se estime apropiado.

Para el desarrollo de la aplicación se emplea 5 semanas, pues se espera que surjan imprevistos que requieran del rediseño de la lógica. Por último se incluye una fase de pruebas sobre navegadores y se aparta un tiempo para la redacción de la memoria.

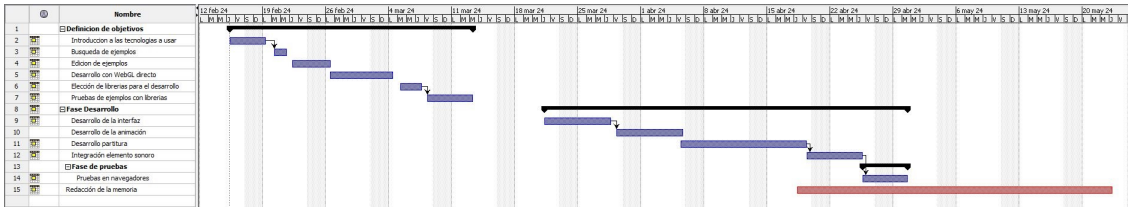


Figura 4.1: Diagrama de Gantt

4.2 Diseño de la interfaz

En primer lugar se plantea que el elemento canvas ocupe toda la pantalla y todos los elementos de interacción se encuentren en su interior. Existe una función que permite identificar un objeto interno del canvas al pulsar sobre el en el interior del canvas. Se propone un diseño en el cual la flauta se encuentre en el centro de la pantalla. En la edición de la partitura se propone que el pentagrama se encuentre en la parte superior, mientras que en la parte inferior se sitúan los

botones que conforman las notas. Inicialmente se propuso que fuera mediante una interacción con el pentagrama. Pero se descarto ya que si se exportaba para otros dispositivos podría resultar incoveniente debido al tamaño. Por último se plantea un menú lateral para seleccionar distintos modos de juego. Figura 4.2

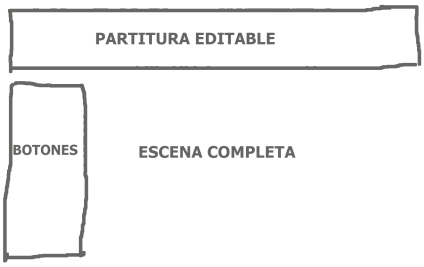


Figura 4.2: Primer boceto

Se tiene planeado diseñar la aplicación para ordenadores portátiles o de sobremesa con al menos una resolución de ancho de pantalla de 1600 píxeles debido a que se busca diseñar una aplicación con al menos tres bloques de notas.

CAPÍTULO 5
Solución propuesta

En esta parte del proyecto se mostraran los ejercicios que se pretenden realizar sobre el proyecto y se describirá cual han sido el proceso llevado a cabo para realizarlos. El código completo de las aplicaciones es adjuntado como anexo.

5.1 Ejercicios

Se ha propuesto varios ejercicios que la aplicación sea capaz de realizar. El primero es un modo libre que permite poner notas sobre la partitura y reproducirlas. El segundo es un modo en el que se genera una partitura de doce notas y se debe replicar por el sonido, iluminándose en rojo las incorrectas y en verde las correctas.

5.2 Desarrollo con WebGL

Se ha partido tomando como ejemplo los tutoriales, usando el ejemplo con las figuras del cubo y el tetraedro para familiarizarse con el entorno 3D. En primer lugar se crea una pagina web sencilla donde se cargan dos scripts,uno con funciones sobre matrices y el otro con el programa a desarrollar, junto con un elemento canvas donde se desarrolla ejecuta la aplicación.

Dentro de este espacio, la posición de los elementos es almacenado en matrices de 4x4 donde se almacena la posición respecto al origen(0,0,0) y las coordenadas de transformación.

Para crear un objeto tridimensional, como un cubo,es necesario asignar los puntos que la componen, como esta compuesto de triángulos, cada lado se compone de 6 puntos, ya que aunque los puntos se encuentren en la misma ubicación solo pueden ser usados para componer una cara. Por este motivo para crear un cubo son necesarios 36 puntos.

En este punto se plantea crear una aplicación con las figuras de cuadrado y tetraedro para componer una escena y tratar de animarla. Como aun no esta implementado un sistema de iluminación. Se colorea cada cara con grado de un color para distinguir los extremos. El resultado se muestra en la Figura 5.1.

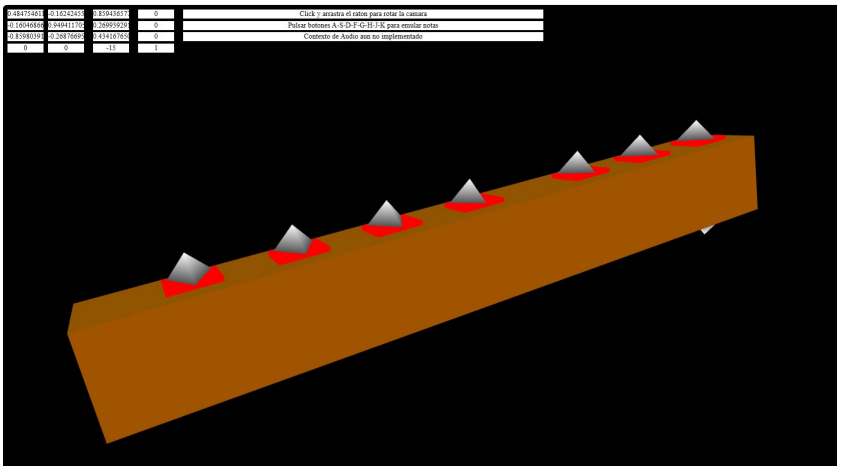


Figura 5.1: Versión usando solo WebGL

WebGL en si mismo no incluye los cálculos para crear un sistema de iluminación, simplemente usa dos funciones, vertex shader y fragment shader, todo lo demás espera ser creado mediante funciones. Esto unido al hecho que los objetos creados tienen los campos de posición, rotación y escala en una matriz y si bien se podrían crear más elementos para editar estos campos, resulta más interesante investigar el funcionamiento de alguna de las librerías de WebGL mencionadas.

5.3 Desarrollo con Three.js

Entre las opciones mencionadas la que disponía de más ejemplos y mayor documentación es Three.js. Esta librería cuenta con métodos para crear la iluminación, para tratar con los objetos de manera más intuitiva y para cargar objetos creados por aplicaciones de modelado externas. Por ello se continuara este proyecto usándola. Se puede generar el proyecto usando archivos provenientes de una red de entrega de contenido o descargando los archivos a un directorio local, se elige la segunda opción.

Durante la fase de desarrollo se usara la herramienta de construcción Vite, esta permite empaquetar y optimizar el código y ejecutarlo en servidores de prueba automáticamente.

```
npm install --save three
npm install --save-dev vite
npx vite
```

Esto permite tener un entorno de desarrollo local en que será probada la aplicación. Con el uso de esta librería muchas de las tareas a realizar son realizadas por los métodos de la librería, para cargar objetos externos se utilizaran los métodos GLTFLoader y STLLoader. Se han escogido dos objetos externos de repositorios web para facilitar el desarrollo[55] [54]. Al cargarlos tenemos que transformar su posición para que se acerquen al punto (0,0,0) del entorno de nuestra aplicación.

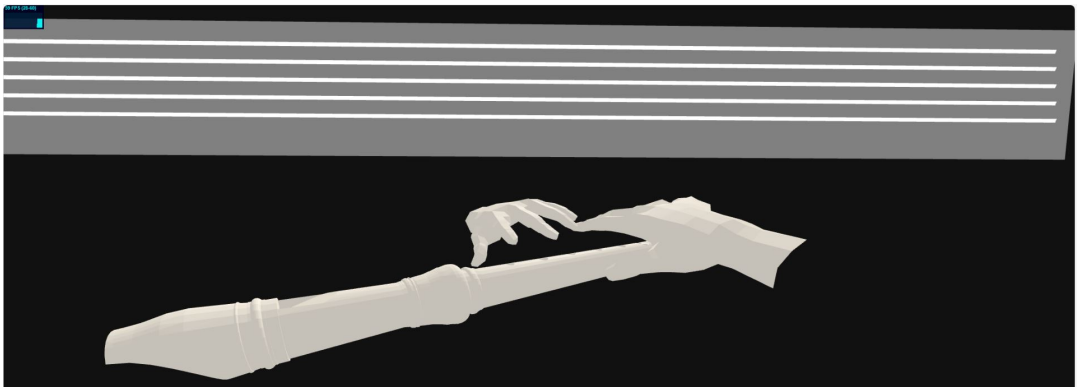


Figura 5.2: Three.js con partitura mediante planos

Al momento de crear las animaciones, los objetos cargados tienen un campo llamado animations, esta compuesto de una lista de elementos AnimationClip. Este elemento contiene el nombre,la duración de la animación y una lista de KeyFrameTrack, que incluye el tiempo y la posición que debe tener cada punto en un momento concreto. Pero para crear animaciones es necesario que el modelo posea un esqueleto. Como no se tiene experiencia creando animaciones, se opta por crear un esqueleto básico que se compone de un nodo base y 5 hijos que parten de él. Para determinar que parte corresponde a cada hueso del esqueleto se ha decidido por usar un rectángulo y cada punto en el interior de ese rectángulo se añade al nodo del esqueleto. Existen dos atributos que añadir a la geometría que son skinIndex y skinWeight. Estos determinan el hueso al que se incluye y cuanta influencia tiene su movimiento sobre el punto. Como no se ha exportado una animación, se crea una básica, que consiste en un giro de 30º sobre el eje Z. Se emplea una lista de las posiciones que tendrán los dedos. Así cuando sea necesario, la lista se llera en un sentido o en otro dependiendo de la nota a pulsar.

Con la animación realizada, el siguiente paso es crear la partitura y las notas. En primer lugar se plantea realizar una partitura insertada dentro del espacio del canvas, mediante el uso de planos para las lineas del pentagrama, y usando texturas para representar las notas.

Al poner esta idea en acción, se comprueba que el uso de una cámara de perspectiva distorsiona los elementos planos como se muestra en la Figura 5.2 y es necesaria otra alternativa. Tras una búsqueda se encuentra la librería vexflow que permite crear partituras musicales directamente en el navegador.

El uso de esta librería permite añadir un elemento SVG o un elemento CANVAS a un elemento de la pagina web. Mediante el uso de un renderer se establece el tamaño y el contexto a ejecutar. Como ya se esta usando un canvas se emplea la opción de usarlo como SVG.

Al renderer se le añaden los distintos sistemas que conforman el pentagrama, y se le incluyen las notas. En la notación musical las notas están ordenadas por octavas,nombradas las escalas desde la a hasta la g, y ocho octavas por escala. En el caso de la flauta dulce las notas que usaremos corresponden todas las de la escala 4 y la c de la escala 5. junto con la nota hay que definir el valor musical temporal de la misma. Este determina si la figura es un tono, semitono, etc. Su valor puede definirse mediante la letra correspondiente o mediante un valor nu-

mérico, comenzando en 1 para un tono y multiplicándolo por dos para obtener los correspondientes semitonos.

En nuestro programa, cada vez que el método es llamado genera una nueva instancia es agregado al elemento , por lo que previamente se vacía antes de llamar a la siguiente instancia.

La librería no permite que el valor de las notas sea mayor al permitido, por ejemplo 5 cuartos de tono, y se le establecen clausulas de control. Para llevar un control del numero de notas por bloque, se plantea el uso de dos listas una que incluya todas las notas y otra que contenga la posición de la primera nota de cada bloque.

Para la reproducción del sonido, se empleara grabaciones de sonidos de flauta. En un entorno 3D el elemento de audio puede ser interpretado como un Audio que tiene un volumen definido o PositionalAudio en el que el volumen además es influido por la distancia del elemento a un AudioListener.

En la aplicación se crea un elemento AudioListener asociado a la cámara y ocho PositionalAudio correspondientes a cada nota a reproducir. Estos elementos son asociados a un cubo situado al final del elemento correspondiente al de la flauta.

Se define una distancia máxima y una distancia en la que el sonido se atenuara dependiendo de la distancia a la cámara y cuan rápido el sonido desciende.

Ahora falta unir la animación con la reproducción del sonido, se utiliza un delay para sincronizar la animación y el sonido. Algunas pistas de audio producen un pop al cortar la reproducción, esto es debido a que se esta usando una única pista para los sonidos y pausandola. Si una pista de audio se corta en un punto en el que la onda no esta en el punto 0 se escucha ese pop, cuanto más cercano a cero menos se escucha, habría que obtener más muestras y asignarlas mejor. Una vez completado este paso se tiene un editor y reproductor de música, mostrado en la Figura 5.3

El siguiente objetivo es crear un archivo que almacene las notas que han sido creadas, se propuso emplear el formato MusicXML al ser un formato de notación usado por otras aplicaciones. Este formato emplea una estructura de partes similar a la utilizada por la librería vexflow, esto ayuda a comprender esta estructura del documento. Sin embargo de momento se emplea un archivo de texto simple en que se serán guardadas las notas.

Debido a la característica de javascript de ser asíncrona, no puede garantizar la carga en un único método, así que se usan dos botones para cargar el archivo primero y otro para añadir al código principal las notas leídas.

Como el objetivo de desarrollar todo el entorno dentro del canvas no es posible, se emplea una hoja de estilos css para aplicarla a la pagina. el resultado mostrado en la Figura 5.4.

Se desarrolla un segundo modo de juego en el cual se crea una lista de doce notas y el objetivo es escribir esas notas usando el oído para conseguirlo. Se añade interfaz adicional

Para probarlo en un servidor de producción local se usa vite para generar el código optimizado dentro de una carpeta /dist

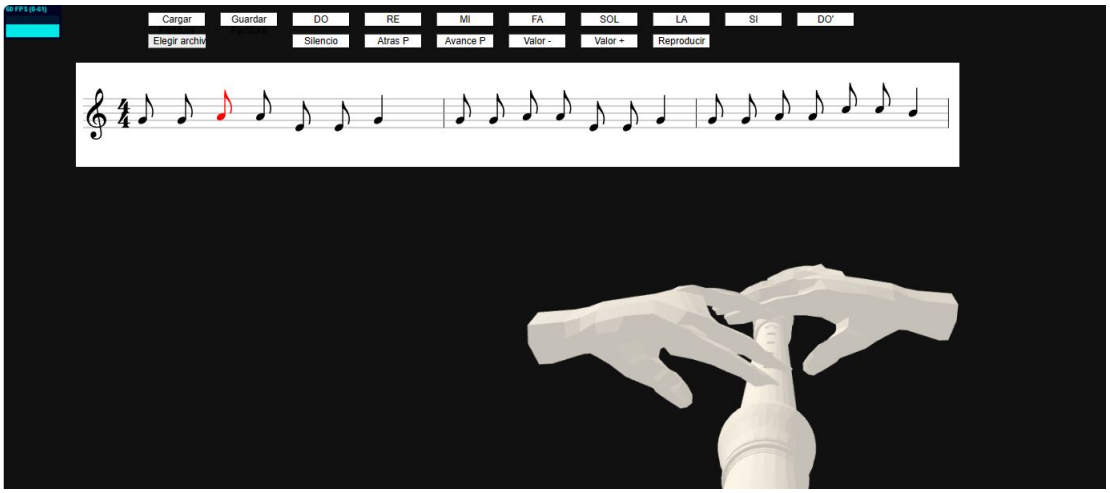


Figura 5.3: Vexflow con reproducción

Figura 5.4: Hoja estilos aplicada

```
npx vite build
```

CAPÍTULO 6
Despliegue

6.1 Prueba en el servidor

Para probar el proyecto en un servidor se ha decidido por emplear una maquina virtual de ubuntu, se le instala el servidor nginx. Al buscar la dirección local en el navegador aparecerá la pagina de prueba de nginx. Se copian los archivos en la dirección /var/www/dist y se crea un archivo de configuración para acceder al sitio en la dirección /etc/nginx/sites-enabled ahora al reiniciar el servidor se ejecuta nuestro programa.

```
sudo apt update
sudo apt install nginx
sudo systemctl start nginx
```

```
server {
listen 80;
listen[::]: 80;
server_name 10.0.2.15;
root /var/www/dist
index index.html; location / {
try_files $uri $uri/ =404;
}
}
```

Una vez comprobamos que la aplicación funciona se decide subirla a un host externo. Se elige Firebase de Google para ello porque es un servidor de hosting gratuito. Para comenzar hay que descargarse el cliente de Firebase que incluye las herramientas para subir los archivos a la web. Nos situamos en el directorio raíz y ejecutamos la secuencia, nos pregunta que servicio usar, sobre que proyecto y el directorio donde se encuentran los archivos a subir, en nuestro caso dist. El programa genera un archivo llamado index.html como nuestra pagina se llama igual indicamos que no sobreescriba. Una vez inicializado hay que desplegarlo.

Al finalizar el despliegue se muestra las direcciones que analizan el sitio y la dirección web del mismo.

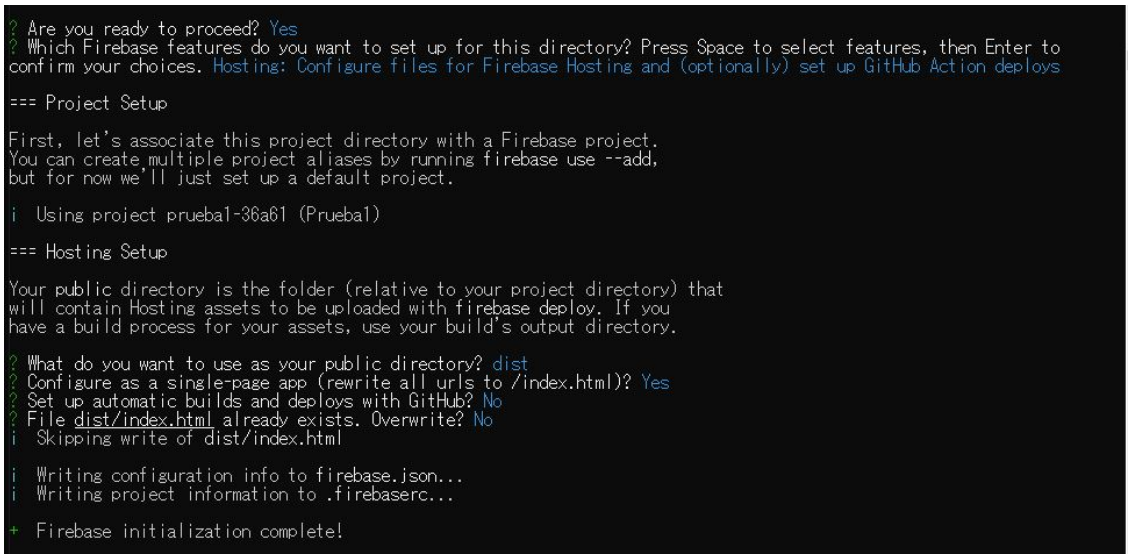


Figura 6.1: Captura de consola ejecutando init

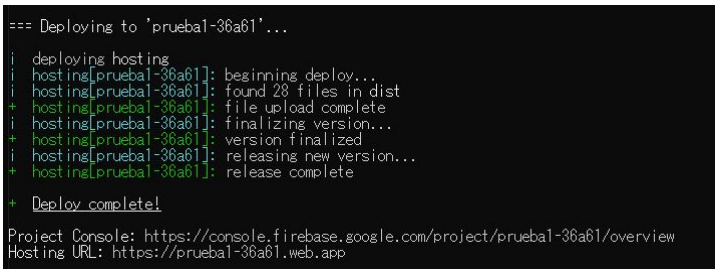


Figura 6.2: Captura de consola ejecutando deploy

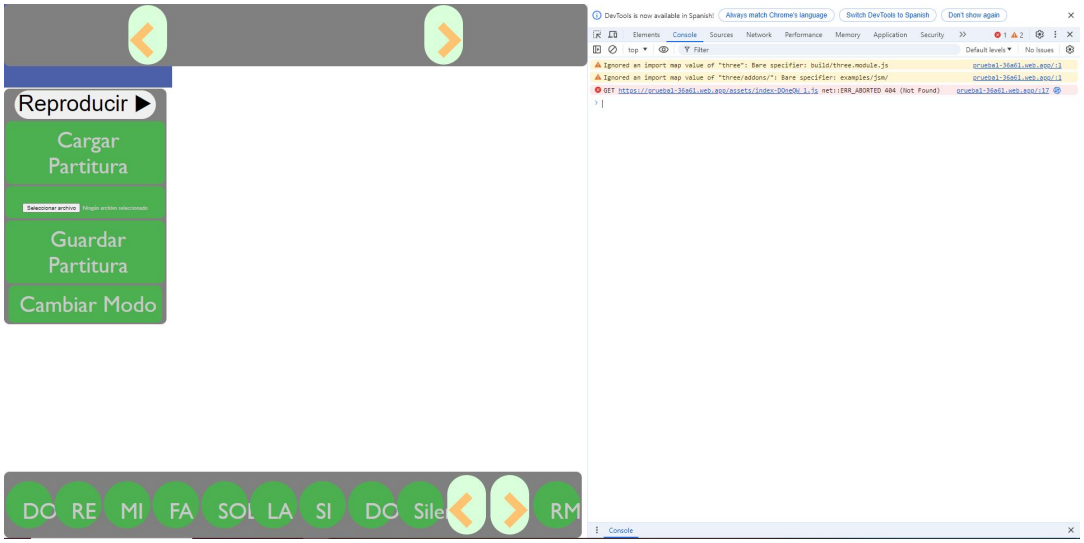


Figura 6.3: Captura la pagina web

Sin embargo al ir a la dirección web no parece encontrar el código del script. https://prueba1-36a61.web.app/

Tras realizar distintas pruebas se determina que hay un problema en el renombrado de las librerias durante el proceso de compilación y es lo que esta produciendo los errores.

CAPÍTULO 7
Conclusiones

El desarrollo de la aplicación ha sido complicado, la falta de experiencia con este tipo de desarrollos ha provocado diversos retrasos con respecto al proceso creativo. Por una parte, elegir al principio la tecnología prácticamente sin librerías adicionales supone una mayor carga de trabajo, que muchas librerías han solucionado, haciendo del trabajo productivo innecesariamente más complejo de lo que supondría haber comenzado desde un nivel más elevado. Por otro lado permite tener un mayor conocimiento de cada una de las herramientas y de como agregan valor a cada parte del proceso de producción. Consideramos los objetivos parcialmente cumplidos, pues se ha desarrollado la aplicación sobre un entorno que es capaz de ser ejecutado en un servidor local y que falla en uno externo.

Debido a cursar una rama distinta a la que corresponde este los conocimientos necesarios HTML y CSS han sido estudiados durante el desarrollo y por ello son de un nivel básico. No obstante, para la implementación de WebGL al estar basada en OpenGL y esta tecnología haber sido estudiada en la asignatura Arquitectura y entornos de desarrollo para videoconsolas ha resultado útil para la comprensión del desarrollo en un entorno tridimensional. En cuanto a la tecnología Javascript a pesar de no haber sido estudiada, la capacidad lógica aplicada durante otras muchas asignaturas de la carrera ha permitido no tener dificultades aplicándola.

Este trabajo me ha servido para afianzar algunos conocimientos y expandir otros que no había explorado, así cómo buscar distintas resoluciones y adaptarse a problemas que surgen durante el desarrollo.

7.1 Trabajos Futuros

Aunque no se ha alcanzado a desarrollar una aplicación inmersiva, la documentación de las librerías explica como con pequeños cambios es posible lograr la integración del programa con estas.

Mediante la librería MediaPipe es posible realizar un seguimiento de la posición de las manos en tiempo real. Puede ser posible que mediante ajustes se pueda usar para sincronizar la aplicación y otorgarle funcionalidad.

Bibliografía

[1] JORQUERA JARAMILLO, María Cecilia. Modelos didácticos en la enseñanza musical: el caso de la escuela española. Rev. music. chil., Santiago , v. 64, n. 214, p. 52-74, dic. 2010 . Disponible en http://www.scielo.cl/scielo.php?script=sci_arttext&pid=S0716-27902010000200006&lng=es&nrm=iso accedido en 21 abr. 2024. http://dx.doi.org/10.4067/S0716-27902010000200006.

[2] Schafer, Steven M. *Web Standards Programmer's Reference [Electronic Resource] : HTML, CSS, JavaScript, Perl, Python, and PHP.*. 1st edition, Wiley Pub., 2005.

[3] Parisi, Tony., et al. *WebGL : Up and Running. Illustrated by Robert Romano.*. First edition., O'Reilly, 2012.

[4] Dirksen, Jos. *Learn Three. Js : Program 3D Animations and Visualizations for the Web with JavaScript and WebGL.*. 4th ed., Packt Publishing, Limited, 2023.

[5] Cozzi, Patrick., and Patrick Cozzi. *WebGL Insights [Electronic Resource].*. [First edition]., CRC Press, 2015.

[6] Lasse Grubbe. (2019). Musicca – Aprende teoría musical de forma gratuita Consultado [7 de junio de 2024] a https://www.musicca.com/es.

[7] Aprendo Musica con las TIC Consultado [7 de junio de 2024] a https://aprendomusica.com.

[8] blender.org - Home of the Blender project - Free and Open 3D Creation Software Consultado [7 de junio de 2024] a https://www.blender.org/.

[9] Thingiverse - Digital Designs for Physical Objects Consultado [7 de junio de 2024] a https://www.thingiverse.com/

[10] Newsfeed - Sketchfab Consultado [7 de junio de 2024] a https://sketchfab.com/feed

[11] Durán Roca, C. (2018). Aplicación web para la visualización de objetos 3D fragmentados en piezas. http://hdl.handle.net/10251/107051.

[12] Hernández Medrano, V. (2019). Desarrollo de una aplicación para la Web utilizando el Web Audio API. http://hdl.handle.net/10251/127334

[13] Alcañiz Villanueva, Jorge. Aplicación de ayuda a la iniciación al uso de partituras musicales. Universitat Politècnica de València, 2020. http://hdl.handle.net/10251/151758

[14] Three.js – JavaScript 3D Library Consultado [7 de abril de 2024] a https://threejs.org/.

[15] Babylon.js: Powerful, Beautiful, Simple, Open - Web-Based 3D At Its Best Consultado [7 de junio de 2024] a https://babylonjs.com/.

[16] Difference Between Three.js and Babylon.js: What Actually Should You Choose? | by Shariq Ahmed | Medium Consultado [5 de junio de 2024] a https://medium.com/@shariq.ahmed525/difference-between-three-js-and-babylon-js-what-actually-should-you-choose-996

[17] Frameworks de WebGL: Three.js frente a Babylon.js Consultado [10 de junio de 2024] a https://ichi.pro/es/frameworks-de-webgl-three-js-frente-a-babylon-js-60023737767572.

[18] W3C Consultado [7 de abril de 2024] a https://www.w3.org/.

[19] Singhalpriyansh. (2020). Intro to Modern Web Development. Medium. Consultado [22 de julio de 2024] a https://medium.com/javarevisited/intro-to-modern-web-development-d714563c87e.

[20] Frameworks de WebGL: Three.js frente a Babylon.js Consultado [10 de junio de 2024] a https://ichi.pro/es/frameworks-de-webgl-three-js-frente-a-babylon-js-60023737767572.

[21] Web Audio API - Referencia de la API Web | MDN Consultado [7 de abril de 2024] a https://developer.mozilla.org/es/docs/Web/API/Web_Audio_API.

[22] WebGL Public Wiki Consultado [7 de abril de 2024] a https://www.khronos.org/webgl/wiki/Main_Page.

[23] Dev.Opera — An Introduction to WebGL — Part 1 Consultado [7 de abril de 2024] a https://dev.opera.com/articles/introduction-to-webgl-part-1/.

[24] Graphic Pipeline Consultado [7 de abril de 2024] a https://opentechschool-brussels.github.io/intro-to-webGL-and-shaders/log1_graphic-pipeline.

[25] WebGL Academy Interactive Tutorials Consultado [7 de abril de 2024] a http://www.webglacademy.com/.

[26] WebGL Samples Consultado [7 de abril de 2024] a https://webglsamples.org/.

[27] OpenGL Wiki Consultado [7 de abril de 2024] a https://www.khronos.org/opengl/wiki/Main_Page.

[28] WebGL: 2D and 3D graphics for the web - Web APIs | MDN Consultado [7 de abril de 2024] a https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API.

[29] WebGL: Animating a 3D scene | Packt Hub Consultado [7 de abril de 2024] a https://hub.packtpub.com/webgl-animating-3d-scene/.

[30] WebGL tutorial - Web APIs | MDN Consultado [7 de abril de 2024] a https://developer.mozilla.org/en-US/docs/Web/API/WebGL_API/Tutorial.

[31] Fun with WebGL 2.0 : The Introduction - YouTube Consultado [7 de abril de 2024] a https://www.youtube.com/watch?v=LtFujAtKM5I&list=PLMinhigDWz6emRKVkVIEAaePW7vtIkaIF&index=1.

[32] MusicXML 4.0 Consultado [7 de abril de 2024] a https://www.w3.org/2021/06/musicxml40/.

[33] SoundStage | SideQuest Consultado [19 de junio de 2024] a https://sidequestvr.com/app/360/soundstage.

[34] Just a Piano | SideQuest Consultado [19 de junio de 2024] a https://sidequestvr.com/app/8574/just-a-piano.

[35] VRtuos | SideQuest Consultado [19 de junio de 2024] a https://sidequestvr.com/app/494/vrtuos.

[36] Music Sheet Trainer VR | SideQuest Consultado [19 de junio de 2024] a https://sidequestvr.com/app/7688/music-sheet-trainer-vr.

[37] VexFlow - HTML5 Music Engraving Consultado [7 de abril de 2024] a https://www.vexflow.com/.

[38] Immersive Web Developer Home Consultado [22 de junio de 2024] a https://immersiveweb.dev/.

[39] Getting Started | Vite Consultado [7 de abril de 2024] a https://vitejs.dev/guide/.

[40] ¿Por qué Vite? | Vite Consultado [7 de abril de 2024] a https://es.vitejs.dev/guide/why.html.

[41] Vite vs. Webpack: una comparativa detallada - Kinsta® Consultado [7 de abril de 2024] a https://kinsta.com/es/blog/vite-vs-webpack/.

[42] A Sample MusicXML Encoding - MusicXML Consultado [7 de abril de 2024] a https://www.musicxml.com/publications/makemusic-recordare/notation-and-analysis/a-sample-musicxml-encoding/.

[43] Free MusicXML viewer from Soundslice Consultado [7 de abril de 2024] a https://www.soundslice.com/musicxml-viewer/.

[44] Tone.js Consultado [7 de abril de 2024] a https://tonejs.github.io/.

[45] Web3D Consortium | Open Standards for Real-Time 3D Communication Consultado [21 de junio de 2024] a https://www.web3d.org/.

[46] Getting Started with X3D | Web3D Consortium Consultado [21 de junio de 2024] a https://www.web3d.org/getting-started-x3d.

[47] X3DOM Documentation: Overview Consultado [7 de abril de 2024] a https://doc.x3dom.org/index.html.

[48] Web3D Consortium | Open Standards for Real-Time 3D Communication Consultado [7 de abril de 2024] a https://www.web3d.org/.

[49] Getting Started | X ITE X3D Browser Consultado [7 de abril de 2024] a https://create3000.github.io/x_ite/.

[50] Getting Started | X ITE X3D Browser Consultado [7 de abril de 2024] a https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Modules.

[51] Getting Started | X ITE X3D Browser Consultado [7 de abril de 2024] a https://www.tutorialesprogramacionya.com/javascriptya/temarios/descripcion.php?inicio=100&cod=102.

[52] WebGL Month. Day 17. Exploring OBJ format - DEV Community Consultado [7 de abril de 2024] a https://dev.to/lesnitsky/webgl-month-day-17-exploring-obj-format-6fn.

[53] GLSL Shaders - Game development | MDN Consultado [21 de junio de 2024] a https://developer.mozilla.org/en-US/docs/Games/Techniques/3D_on_the_web/GLSL_Shaders.

[54] A Recorder Flute by ArturoV - Thingiverse Consultado [7 de abril de 2024] a https://www.thingiverse.com/thing:91903.

[55] Hand (low poly) - Download Free 3D model by scribbletoad (@scribbletoad) [d6c802a] Consultado [7 de abril de 2024] a https://sketchfab.com/3d-models/hand-low-poly-d6c802a74a174c8c805deb20186d1877

APÉNDICE A
Codigo

Proyecto Incial usando solo WebGL - index.html

```
1
2  <!DOCTYPE html>
3  <html>
4    <head>
5      <meta charset='utf-8'/>
6      <script src="libs.js"></script>
7      <script src="main.js"></script>
8    </head>
9    <body style='margin: 0'>
10     <canvas id='your_canvas' style='position: absolute; background-
           color: black;'></canvas>
11
12     <div id="m0" style="position: absolute; z-index: 1000; display:
           block; top: 10px; left: 10px; width: 80px; text-align: center;
           height: 20px; background-color: white;" >QQ</div>
13     <div id="m1" style="position: absolute; z-index: 1000; display:
           block; top: 10px; left: 100px; width: 80px; text-align: center;
            height: 20px; background-color: white;" >QQ</div>
14     <div id="m2" style="position: absolute; z-index: 1000; display:
           block; top: 10px; left: 200px; width: 80px; text-align: center;
            height: 20px; background-color: white;" >QQ</div>
15     <div id="m3" style="position: absolute; z-index: 1000; display:
           block; top: 10px; left: 300px; width: 80px; text-align: center;
            height: 20px; background-color: white;" >QQ</div>
16
17     <div id="m4" style="position: absolute; z-index: 1000; display:
           block; top: 35px; left: 10px; width: 80px; text-align: center;
           height: 20px; background-color: white;" >QQ</div>
18 <div id="m5" style="position: absolute; z-index: 1000; display: block;
       top: 35px; left: 100px; width: 80px; text-align: center; height: 20
       px; background-color: white;" >QQ</div>
19 <div id="m6" style="position: absolute; z-index: 1000; display: block;
       top: 35px; left: 200px; width: 80px; text-align: center; height: 20
       px; background-color: white;" >QQ</div>
20 <div id="m7" style="position: absolute; z-index: 1000; display: block;
       top: 35px; left: 300px; width: 80px; text-align: center; height: 20
       px; background-color: white;" >QQ</div>
21
22 <div id="m8" style="position: absolute; z-index: 1000; display: block;
       top: 60px; left: 10px; width: 80px; text-align: center; height: 20
       px; background-color: white;" >QQ</div>
```

```
<div id="m9" style="position: absolute; z-index: 1000; display: block; top: 60px; left: 100px; width: 80px; text-align: center; height: 20px; background-color: white;" >QQ</div>
<div id="m10" style="position: absolute; z-index: 1000; display: block; top: 60px; left: 200px; width: 80px; text-align: center; height: 20px; background-color: white;" >QQ</div>
<div id="m11" style="position: absolute; z-index: 1000; display: block; top: 60px; left: 300px; width: 80px; text-align: center; height: 20px; background-color: white;" >QQ</div>

<div id="m12" style="position: absolute; z-index: 1000; display: block; top: 85px; left: 10px; width: 80px; text-align: center; height: 20px; background-color: white;" >QQ</div>
<div id="m13" style="position: absolute; z-index: 1000; display: block; top: 85px; left: 100px; width: 80px; text-align: center; height: 20px; background-color: white;" >QQ</div>
<div id="m14" style="position: absolute; z-index: 1000; display: block; top: 85px; left: 200px; width: 80px; text-align: center; height: 20px; background-color: white;" >QQ</div>
<div id="m15" style="position: absolute; z-index: 1000; display: block; top: 85px; left: 300px; width: 80px; text-align: center; height: 20px; background-color: white;" >QQ</div>

 <div id="info" style="position: absolute; z-index: 1000; display: block; top: 10px; left: 400px; width: 800px; text-align: center; height: 20px; background-color: white;" >Click y arrastra el raton para rotar la camara</div>
<div id="info2" style="position: absolute; z-index: 1000; display: block; top: 35px; left: 400px; width: 800px; text-align: center; height: 20px; background-color: white;" >Pulsar botones A-S-D-F-G-H-J-K para emular notas</div>
<div id="info3" style="position: absolute; z-index: 1000; display: block; top: 60px; left: 400px; width: 800px; text-align: center; height: 20px; background-color: white;" >Contexto de Audio aun no implementado</div>

</body>
</html>
```

Proyecto Incial usando solo WebGL - main.js

```
function main() {
  var CANVAS = document.getElementById("your_canvas");
  CANVAS.width = window.innerWidth;
  CANVAS.height = window.innerHeight;

  /*========================= CAPTURE MOUSE EVENTS
     ========================= */

  //var AMORTIZATION = 0.95;
  var drag = false;


  var x_prev, y_prev;

  var dX = 0, dY = 0;
  var mouseDown = function(e) {
    drag = true;
    x_prev = e.pageX, y_prev = e.pageY;
```

```
18      e.preventDefault();
19      return false;
20    };
21
22    var mouseUp = function(e){
23      drag = false;
24    };
25
26    var mouseMove = function(e) {
27      if (!drag) return false;
28      dX = (e.pageX-x_prev) * 2 * Math.PI / CANVAS.width,
29        dY = (e.pageY-y_prev) * 2 * Math.PI / CANVAS.height;
30      THETA += dX;
31      PHI += dY;
32      x_prev = e.pageX, y_prev = e.pageY;
33      e.preventDefault();
34    };
35
36    CANVAS.addEventListener("mousedown", mouseDown, false);
37    CANVAS.addEventListener("mouseup", mouseUp, false);
38    CANVAS.addEventListener("mouseout", mouseUp, false);
39    CANVAS.addEventListener("mousemove", mouseMove, false);
40
41  /*Cosas*/
42  //consoleLog.innerHTML += `${hola}<br>`;
43
44
45  document.addEventListener('keydown', logKey);
46  function logKey(e) {
47
48    console.log(e.code);
49    switch(e.code){
50      case 'KeyA': posicion(255); break; //do
51      case 'KeyS': posicion(127); break; //re
52      case 'KeyD': posicion(63);  break; //mi
53      case 'KeyF': posicion(31);  break; //fa
54      case 'KeyG': posicion(15);  break; //sol
55      case 'KeyH': posicion(7);   break; //la
56      case 'KeyJ': posicion(3);   break; //si
57      case 'KeyK': posicion(5);   break; //doa
58    }
59  };
60
61  var pos_anterior=new Uint8Array();
62  var pos_anterior=255;
63
64  var posicion = function(e){
65    var pos= Number(e);
66    var cambio =pos_anterior ^ pos;
67
68    for (var i = 0; i < ARRAY_DEDO.length; i++) {
69      if(isBitOn(cambio,i)){
70        m= ARRAY_DEDO[i];
71
72        if(i==0){
73        if(isBitOn(pos_anterior,i))LIBS.translateY(m,-1.1)
74        else LIBS.translateY(m,1.1)
75        }else{
76        if(isBitOn(pos_anterior,i))LIBS.translateY(m,1.1)
```

```
77          else LIBS.translateY(m,-1.1)
78          }
79      }
80    }
81    pos_anterior=pos;
82  };
83
84  function isBitOn(number, index) {
85    return Boolean(number & (1 << index));
86  };
87
88  var matrixPuntos = function(e) {
89
90    m0.innerHTML = e[0];
91      m1.innerHTML = e[1];
92      m2.innerHTML = e[2];
93      m3.innerHTML = e[3];
94      m4.innerHTML = e[4];
95      m5.innerHTML = e[5];
96      m6.innerHTML = e[6];
97      m7.innerHTML = e[7];
98      m8.innerHTML = e[8];
99      m9.innerHTML = e[9];
100     m10.innerHTML = e[10];
101     m11.innerHTML = e[11];
102     m12.innerHTML = e[12];
103     m13.innerHTML = e[13];
104     m14.innerHTML = e[14];
105     m15.innerHTML = e[15];
106   };
107
108   /*=======CREATE AUDIOCONTEXT======*/
109   /*
110 const audioCtx = new (window.AudioContext || window.webkitAudioContext)
        ();
111
112 // create Oscillator node
113 const oscillator = audioCtx.createOscillator();
114
115 oscillator.type = "square";
116 oscillator.frequency.setValueAtTime(440, audioCtx.currentTime); //
        value in hertz
117 oscillator.connect(audioCtx.destination);
118 oscillator.start();
119
120 */
121
122
123   /*========================== GET WEBGL CONTEXT
        ========================== */
124   var GL;
125   try {
126     GL = CANVAS.getContext("webgl", {antialias: true});
127   } catch (e) {
128     alert("WebGL context cannot be initialized");
129     return false;
130   }
131
132   /*========================= SHADERS ========================= */
```

```
133   /*jshint multistr: true */
134
135   var shader_vertex_source = "\n\
136 attribute vec3 position;\n\
137 uniform mat4 Pmatrix, Vmatrix, Mmatrix;\n\
138 attribute vec3 color; // the color of the point\n\
139 varying vec3 vColor;\n\
140 \n\
141 void main(void) {\n\
142 gl_Position = Pmatrix * Vmatrix * Mmatrix * vec4(position, 1.);\n\
143 vColor = color;\n\
144 }";
145
146   var shader_fragment_source = "\n\
147 precision mediump float;\n\
148 uniform float greyscality;\n\
149 varying vec3 vColor;\n\
150 void main(void) {\n\
151 \n\
152 \n\
153 float greyscaleValue = (vColor.r + vColor.g + vColor.b) / 3.;\n\
154 vec3 greyscaleColor = vec3(greyscaleValue, greyscaleValue,
      greyscaleValue);\n\
155 \n\
156 \n\
157 vec3 color = mix(greyscaleColor, vColor, greyscality);\n\
158 gl_FragColor = vec4(color, 1.);\n\
159 }";
160
161   var compile_shader = function(source, type, typeString) {
162     var shader = GL.createShader(type);
163     GL.shaderSource(shader, source);
164     GL.compileShader(shader);
165     if (!GL.getShaderParameter(shader, GL.COMPILE_STATUS)) {
166       alert("ERROR IN " + typeString + " SHADER: " + GL.
            getShaderInfoLog(shader));
167       return false;
168     }
169     return shader;
170   };
171
172   var shader_vertex = compile_shader(shader_vertex_source, GL.
       VERTEX_SHADER, "VERTEX");
173   var shader_fragment = compile_shader(shader_fragment_source, GL.
       FRAGMENT_SHADER, "FRAGMENT");
174
175   var SHADER_PROGRAM = GL.createProgram();
176   GL.attachShader(SHADER_PROGRAM, shader_vertex);
177   GL.attachShader(SHADER_PROGRAM, shader_fragment);
178
179   GL.linkProgram(SHADER_PROGRAM);
180
181   var _Pmatrix = GL.getUniformLocation(SHADER_PROGRAM, "Pmatrix");
182   var _Vmatrix = GL.getUniformLocation(SHADER_PROGRAM, "Vmatrix");
183   var _Mmatrix = GL.getUniformLocation(SHADER_PROGRAM, "Mmatrix");
184
185   var _greyscality = GL.getUniformLocation(SHADER_PROGRAM, "greyscality
       ");
186   var _color = GL.getAttribLocation(SHADER_PROGRAM, "color");
```

```
187   var _position = GL.getAttribLocation(SHADER_PROGRAM, "position");
188
189   GL.enableVertexAttribArray(_color);
190   GL.enableVertexAttribArray(_position);
191
192   GL.useProgram(SHADER_PROGRAM);
193
194   /*======================== THE CUBE ======================== */
195   // POINTS:
196   var cube_vertex = [
197     -1,-1,-8,      0.63,0.33,0, //
198     1,-1,-8,       0.63,0.33,0,
199     1, 1,-8,       0.63,0.33,0,
200     -1, 1,-8,       0.63,0.33,0,
201
202     -1,-1, 8,      0.63,0.3,0,
203     1,-1, 8,     0.63,0.3,0,
204     1, 1, 8,      0.63,0.3,0,
205     -1, 1, 8,      0.63,0.3,0,
206
207     -1,-1,-8,      0.6,0.33,0,
208     -1, 1,-8,      0.6,0.33,0,
209     -1, 1, 8,      0.6,0.33,0,
210     -1,-1, 8,      0.6,0.33,0,
211
212     1,-1,-8,      0.63,0.33,0,
213     1, 1,-8,      0.63,0.33,0,
214     1, 1, 8,      0.63,0.33,0,
215     1,-1, 8,      0.63,0.33,0,
216
217     -1,-1,-8,      0.63,0.28,0,
218     -1,-1, 8,      0.63,0.28,0,
219     1,-1, 8,      0.63,0.28,0,
220     1,-1,-8,     0.63,0.28,0,
221
222     -1, 1,-8,      0.57,0.33,0,
223     -1, 1, 8,      0.57,0.33,0,
224     1, 1, 8,      0.57,0.33,0,
225     1, 1,-8,      0.57,0.33,0
226
227   ];
228     var small_cube_vertex = [
229     -.5, 0,-.5,     1,0,0, //
230      .5, 0,-.5,     1,0,0,
231      .5, 0.1,-.5,     1,0,0,
232     -.5, 0.1,-.5,     1,0,0,
233
234     -.5, 0, .5,     1,0,0,
235      .5, 0, .5,     1,0,0,
236      .5, 0.1, .5,     1,0,0,
237     -.5, 0.1, .5,     1,0,0,
238
239     -.5, 0,-.5,     1,0,0,
240     -.5, 0.1,-.5,     1,0,0,
241     -.5, 0.1, .5,     1,0,0,
242     -.5, 0, .5,     1,0,0,
243
244     .5, 0,-.5,      1,0,0,
245     .5, 0.1,-.5,      1,0,0,
```

```
246     .5, 0.1, .5,      1,0,0,
247     .5, 0, .5,      1,0,0,
248
249     -.5,0,-.5,     1,0,0,
250     -.5,0, .5,     1,0,0,
251      .5,0, .5,     1,0,0,
252      .5,0,-.5,     1,0,0,
253
254     -.5, 0.1,-.5,     1,0,0,
255     -.5, 0.1, .5,     1,0,0,
256      .5, 0.1, .5,     1,0,0,
257      .5, 0.1,-.5,     1,0,0
258
259   ];
260
261   var CUBE_VERTEX = GL.createBuffer ();
262   GL.bindBuffer(GL.ARRAY_BUFFER, CUBE_VERTEX);
263   GL.bufferData(GL.ARRAY_BUFFER,
264                 new Float32Array(cube_vertex),
265     GL.STATIC_DRAW);
266
267       var SMALL_CUBE_VERTEX = GL.createBuffer ();
268   GL.bindBuffer(GL.ARRAY_BUFFER, SMALL_CUBE_VERTEX);
269   GL.bufferData(GL.ARRAY_BUFFER,
270                 new Float32Array(small_cube_vertex),
271     GL.STATIC_DRAW);
272
273
274   // FACES:
275   var cube_faces = [
276     0,1,2,
277     0,2,3,
278
279     4,5,6,
280     4,6,7,
281
282     8,9,10,
283     8,10,11,
284
285     12,13,14,
286     12,14,15,
287
288     16,17,18,
289     16,18,19,
290
291     20,21,22,
292     20,22,23
293
294   ];
295   var CUBE_FACES = GL.createBuffer ();
296   GL.bindBuffer(GL.ELEMENT_ARRAY_BUFFER, CUBE_FACES);
297   GL.bufferData(GL.ELEMENT_ARRAY_BUFFER,
298                 new Uint16Array(cube_faces),
299     GL.STATIC_DRAW);
300
301     /*========================= THE TETRAHEDRON
          ========================= */
302   // POINTS:
303   var tetrahedron_vertex = [
```

```
304     // base face points, included in the plane y = -1
305     -.4, 0, -.4,      1, 0, 0,
306     .6, 0, 0,      0, 1, 0,
307     -.4, 0, .4,       0, 0, 1,
308
309     // corner:, in white
310     0, 0.5, 0,      1, 1, 1
311   ];
312
313   var TETRAHEDRON_VERTEX = GL.createBuffer ();
314   GL.bindBuffer(GL.ARRAY_BUFFER, TETRAHEDRON_VERTEX);
315   GL.bufferData(GL.ARRAY_BUFFER,
316                 new Float32Array(tetrahedron_vertex),
317     GL.STATIC_DRAW);
318
319   // TETRAHEDRON FACES:
320   var tetrahedron_faces = [
321     0, 1, 2, // base
322
323     0, 1, 3, // side 0
324     1, 2, 3, // side 1
325     0, 2, 3  // side 2
326   ];
327   var TETRAHEDRON_FACES = GL.createBuffer ();
328   GL.bindBuffer(GL.ELEMENT_ARRAY_BUFFER, TETRAHEDRON_FACES);
329   GL.bufferData(GL.ELEMENT_ARRAY_BUFFER,
330                 new Uint16Array(tetrahedron_faces),
331     GL.STATIC_DRAW);
332
333   /*========================= MATRIX ========================= */
334
335   var PROJMATRIX = LIBS.get_projection(40, CANVAS.width/CANVAS.height,
        1, 100);
336   var MOVEMATRIX = LIBS.get_I4();
337   var VIEWMATRIX = LIBS.get_I4();
338   var DEDO0 = LIBS.get_I4();
339   var DEDO1 = LIBS.get_I4();
340   var DEDO2 = LIBS.get_I4();
341   var DEDO3 = LIBS.get_I4();
342   var DEDO4 = LIBS.get_I4();
343   var DEDO5 = LIBS.get_I4();
344   var DEDO6 = LIBS.get_I4();
345   var DEDO7 = LIBS.get_I4();
346
347   var ARRAY_DEDO=[DEDO0,DEDO1,DEDO2,DEDO3,DEDO4,DEDO5,DEDO6,DEDO7];
348
349   var AGUJERO0 = LIBS.get_I4();
350   var AGUJERO1 = LIBS.get_I4();
351   var AGUJERO2 = LIBS.get_I4();
352   var AGUJERO3 = LIBS.get_I4();
353   var AGUJERO4 = LIBS.get_I4();
354   var AGUJERO5 = LIBS.get_I4();
355   var AGUJERO6 = LIBS.get_I4();
356   var AGUJERO7 = LIBS.get_I4();
357
358
359   //var AUDIO_MATRIX = LIBS.get_I4();
360
361   LIBS.translateZ(VIEWMATRIX, -15);
```

```
362
363   LIBS.translateY(DEDO0, -1.1);
364   LIBS.translateZ(DEDO0, -7);
365   LIBS.flipX(DEDO0);
366   LIBS.translateY(AGUJERO0, -1.1);
367   LIBS.translateZ(AGUJERO0, -7);
368
369
370   LIBS.translateY(DEDO1, 1.1);
371   LIBS.translateZ(DEDO1, -7);
372   LIBS.translateY(AGUJERO1, 1);
373   LIBS.translateZ(AGUJERO1, -7);
374
375   LIBS.translateY(DEDO2, 1.1);
376   LIBS.translateZ(DEDO2, -5);
377   LIBS.translateY(AGUJERO2, 1);
378   LIBS.translateZ(AGUJERO2, -5);
379
380
381   LIBS.translateY(DEDO3, 1.1);
382   LIBS.translateZ(DEDO3, -3);
383   LIBS.translateY(AGUJERO3, 1);
384   LIBS.translateZ(AGUJERO3, -3);
385
386   LIBS.translateY(DEDO4, 1.1);
387   LIBS.translateZ(DEDO4, 0);
388   LIBS.translateY(AGUJERO4, 1);
389   LIBS.translateZ(AGUJERO4, 0);
390
391   LIBS.translateY(DEDO5, 1.1);
392   LIBS.translateZ(DEDO5, 2);
393   LIBS.translateY(AGUJERO5, 1);
394   LIBS.translateZ(AGUJERO5, 2);
395
396   LIBS.translateY(DEDO6, 1.1);
397   LIBS.translateZ(DEDO6, 4);
398   LIBS.translateY(AGUJERO6, 1);
399   LIBS.translateZ(AGUJERO6, 4);
400
401   LIBS.translateY(DEDO7, 1.1);
402   LIBS.translateZ(DEDO7, 6);
403   LIBS.translateY(AGUJERO7, 1);
404   LIBS.translateZ(AGUJERO7, 6);
405
406
407
408
409   var THETA = 0, PHI = 0;
410
411   /*========================= DRAWING ========================= */
412  GL.enable(GL.DEPTH_TEST);
413  GL.depthFunc(GL.LEQUAL);
414  GL.clearColor(0.0, 0.0, 0.0, 0.0);
415  GL.clearDepth(1.0);
416
417
418
419   var time_prev = 0;
420
```

```
421   var m0 = document.getElementById("m0");
422   var m1 = document.getElementById("m1");
423   var m2 = document.getElementById("m2");
424   var m3 = document.getElementById("m3");
425   var m4 = document.getElementById("m4");
426   var m5 = document.getElementById("m5");
427   var m6 = document.getElementById("m6");
428   var m7 = document.getElementById("m7");
429   var m8 = document.getElementById("m8");
430   var m9 = document.getElementById("m9");
431
432   var m10 = document.getElementById("m10");
433   var m11 = document.getElementById("m11");
434   var m12 = document.getElementById("m12");
435   var m13 = document.getElementById("m13");
436   var m14 = document.getElementById("m14");
437   var m15 = document.getElementById("m15");
438
439   var animate = function(time) {
440     var dt = time - time_prev;
441     /*if (!drag) {
442       dX *= AMORTIZATION, dY *= AMORTIZATION;
443       THETA += dX, PHI += dY;
444
445     }
446         LIBS.set_I4(MOVEMATRIX);
447         LIBS.set_I4(DEDO1);
448         LIBS.set_I4(DEDO2);
449
450
451
452     LIBS.rotateY(dibujar[x], THETA);
453     LIBS.rotateX(dibujar[x], PHI);
454
455   */
456
457     LIBS.rotateY(VIEWMATRIX, THETA);
458     LIBS.rotateX(VIEWMATRIX, PHI);
459
460     THETA=PHI=0;
461
462
463     matrixPuntos(VIEWMATRIX);
464
465     time_prev = time;
466
467
468     GL.viewport(0, 0, CANVAS.width, CANVAS.height);
469     GL.clear(GL.COLOR_BUFFER_BIT | GL.DEPTH_BUFFER_BIT);
470
471     GL.uniformMatrix4fv(_Pmatrix, false, PROJMATRIX);
472     GL.uniformMatrix4fv(_Vmatrix, false, VIEWMATRIX);
473     GL.uniformMatrix4fv(_Mmatrix, false, MOVEMATRIX);
474
475     GL.bindBuffer(GL.ARRAY_BUFFER, CUBE_VERTEX);
476
477     GL.vertexAttribPointer(_position, 3, GL.FLOAT, false, 4*(3+3), 0);
478     GL.vertexAttribPointer(_color, 3, GL.FLOAT, false, 4*(3+3), 3*4);
479     GL.bindBuffer(GL.ELEMENT_ARRAY_BUFFER, CUBE_FACES);
```

```
480
481     GL.uniform1f(_greyscality, 1);
482     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
483
484     GL.bindBuffer(GL.ARRAY_BUFFER, SMALL_CUBE_VERTEX);
485     GL.vertexAttribPointer(_position, 3, GL.FLOAT, false, 4*(3+3), 0);
486     GL.vertexAttribPointer(_color, 3, GL.FLOAT, false, 4*(3+3), 3*4);
487     GL.bindBuffer(GL.ELEMENT_ARRAY_BUFFER, CUBE_FACES);
488
489     if(LIBS.getY(DEDO0)==-1.1)GL.uniform1f(_greyscality, 1);
490     else GL.uniform1f(_greyscality, 0);
491     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO0);
492     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
493
494     if(LIBS.getY(DEDO1)==1.1)GL.uniform1f(_greyscality, 1);
495     else GL.uniform1f(_greyscality, 0);
496     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO1);
497     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
498
499     if(LIBS.getY(DEDO2)==1.1)GL.uniform1f(_greyscality, 1);
500     else GL.uniform1f(_greyscality, 0);
501     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO2);
502     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
503
504     if(LIBS.getY(DEDO3)==1.1)GL.uniform1f(_greyscality, 1);
505     else GL.uniform1f(_greyscality, 0);
506     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO3);
507     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
508
509     if(LIBS.getY(DEDO4)==1.1)GL.uniform1f(_greyscality, 1);
510     else GL.uniform1f(_greyscality, 0);
511     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO4);
512     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
513
514     if(LIBS.getY(DEDO5)==1.1)GL.uniform1f(_greyscality, 1);
515     else GL.uniform1f(_greyscality, 0);
516     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO5);
517     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
518
519     if(LIBS.getY(DEDO6)==1.1)GL.uniform1f(_greyscality, 1);
520     else GL.uniform1f(_greyscality, 0);
521     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO6);
522     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
523
524     if(LIBS.getY(DEDO7)==1.1)GL.uniform1f(_greyscality, 1);
525     else GL.uniform1f(_greyscality, 0);
526     GL.uniformMatrix4fv(_Mmatrix, false, AGUJERO7);
527     GL.drawElements(GL.TRIANGLES, 6*2*3, GL.UNSIGNED_SHORT, 0);
528
529   GL.uniform1f(_greyscality, 0);
530     GL.uniformMatrix4fv(_Mmatrix, false, DEDO0);
531     GL.bindBuffer(GL.ARRAY_BUFFER, TETRAHEDRON_VERTEX);
532     GL.vertexAttribPointer(_position, 3, GL.FLOAT, false, 4*(3+3), 0);
533     GL.vertexAttribPointer(_color, 3, GL.FLOAT, false, 4*(3+3), 3*4);
534     GL.bindBuffer(GL.ELEMENT_ARRAY_BUFFER, TETRAHEDRON_FACES);
535
536     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
537
538   GL.uniform1f(_greyscality, 0);
```

```
539   GL.uniformMatrix4fv(_Mmatrix, false, DEDO1);
540     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
541
542     GL.uniformMatrix4fv(_Mmatrix, false, DEDO2);
543     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
544
545     GL.uniformMatrix4fv(_Mmatrix, false, DEDO3);
546     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
547
548     GL.uniformMatrix4fv(_Mmatrix, false, DEDO4);
549     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
550
551     GL.uniformMatrix4fv(_Mmatrix, false, DEDO5);
552     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
553
554     GL.uniformMatrix4fv(_Mmatrix, false, DEDO6);
555     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
556
557     GL.uniformMatrix4fv(_Mmatrix, false, DEDO7);
558     GL.drawElements(GL.TRIANGLES, 3*4, GL.UNSIGNED_SHORT, 0);
559
560
561     GL.flush();
562     window.requestAnimationFrame(animate);
563   };
564   animate(0);
565 }
566
567 window.addEventListener('load', main);
```

Proyecto Incial usando solo WebGL - libs.js

```
1  var LIBS = {
2    degToRad: function(angle){
3      return(angle*Math.PI/180);
4    },
5
6    get_projection: function(angle, a, zMin, zMax) {
7      var tan=Math.tan(LIBS.degToRad(0.5*angle)),
8          A=-(zMax+zMin)/(zMax-zMin),
9            B=(-2*zMax*zMin)/(zMax-zMin);
10
11     return [
12       0.5/tan, 0 ,    0, 0,
13       0, 0.5*a/tan,   0, 0,
14       0, 0,           A, -1,
15       0, 0,           B, 0
16     ];
17   },
18
19   get_I4: function() {
20     return [1,0,0,0,
21             0,1,0,0,
22             0,0,1,0,
23             0,0,0,1];
24   },
25
26   set_I4: function(m) {
27     m[0]=1, m[1]=0, m[2]=0, m[3]=0,
```

```
28        m[4]=0, m[5]=1, m[6]=0, m[7]=0,
29        m[8]=0, m[9]=0, m[10]=1, m[11]=0,
30        m[12]=0, m[13]=0, m[14]=0, m[15]=1;
31    },
32
33    flipX: function(m) {
34
35      m[0]=-m[0]
36      m[5]=-m[5]
37      m[10]=-m[10]
38
39    },
40
41    rotateX: function(m, angle) {
42      var c=Math.cos(angle);
43      var s=Math.sin(angle);
44      var mv1=m[1], mv5=m[5], mv9=m[9];
45      m[1]=m[1]*c-m[2]*s;
46      m[5]=m[5]*c-m[6]*s;
47      m[9]=m[9]*c-m[10]*s;
48
49      m[2]=m[2]*c+mv1*s;
50      m[6]=m[6]*c+mv5*s;
51      m[10]=m[10]*c+mv9*s;
52    },
53
54    rotateY: function(m, angle) {
55      var c=Math.cos(angle);
56      var s=Math.sin(angle);
57      var mv0=m[0], mv4=m[4], mv8=m[8];
58      m[0]=c*m[0]+s*m[2];
59      m[4]=c*m[4]+s*m[6];
60      m[8]=c*m[8]+s*m[10];
61
62      m[2]=c*m[2]-s*mv0;
63      m[6]=c*m[6]-s*mv4;
64      m[10]=c*m[10]-s*mv8;
65    },
66
67    rotateZ: function(m, angle) {
68      var c=Math.cos(angle);
69      var s=Math.sin(angle);
70      var mv0=m[0], mv4=m[4], mv8=m[8];
71      m[0]=c*m[0]-s*m[1];
72      m[4]=c*m[4]-s*m[5];
73      m[8]=c*m[8]-s*m[9];
74
75      m[1]=c*m[1]+s*mv0;
76      m[5]=c*m[5]+s*mv4;
77      m[9]=c*m[9]+s*mv8;
78    },
79
80    translateX: function(m, t){
81      m[12]+=t;
82    },
83      translateY: function(m, t){
84      m[13]+=t;
85    },
86      translateZ: function(m, t){
```

```
    m[14]+=t;
  },
  getY: function(m){
    return m[13];
  }
};
```

Proyecto Final - index.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset=utf-8>
    <meta name="viewport" content="width=device-width, initial-scale
        =1.0">
    <title>Proyecto1</title>
    <link rel="stylesheet" href="/mystyle.css">
  </head>

  <body>
  <table id="TablaPartitura" class="TablaPartitura">
    <tr>
      <th><button id="RetrocesoPartitura"  class="BotonValor"> <i class
          ="arrow left"></i></button></th>
      <th><div id="output" class="output"></div></th>
      <th><button id="AvancePartitura"   class="BotonValor">  <i class=
          "arrow right"></i></button></th>
    </tr>
  </table>

  <table class="TablaLateral">
      <tr><th><button id="Reproducir" class="BotonReproducir">
          Reproducir &#9654;</button></th></tr>
      <tr><th><button id="BotonCargar" class="BotonLateral"> Cargar
          Partitura </button></th></tr>
      <tr><th><input type="file" name="inputfile" id="inputfile" class=
          "ArchivoExterno"></th></tr>
      <tr><th><button id="BotonGuardar"  class="BotonLateral"> Guardar
          Partitura </button></th></tr>
      <tr><th><button id="CambiarModo"  class="BotonLateral"> Cambiar
          Modo </button></th></tr>
  </table>

  <table id="TM1" class="TablaModo1">
    <tr><th><button id="Muestra" class="BotonReproducir"> Muestra
        &#9654;</button></th></tr>
    <tr><th><button id="Comparar" class="BotonReproducir"> Comparar </
        button></th></tr>
    <tr><th><p id="p1"></th></tr>
    <tr><th><p id="p2"></th></tr>
    <tr><th><button id="MuestraRevelada" class="BotonReproducir" hidden
        > VerMuestra </button></th></tr>
  </table>
  <canvas id='your_canvas' width=window.innerWidth; height=window.
      innerHeight></canvas>
  <p hidden id="textoCarga">

  <table id="TablaNota" class="TablaNotas">
      <tr>
```

```
        <th><button id="BotonDo" class="BotonNota"> DO </button></th>
        <th><button id="BotonRe" class="BotonNota"> RE </button></th>
        <th><button id="BotonMi" class="BotonNota"> MI </button></th>
        <th><button id="BotonFa" class="BotonNota"> FA </button></th>
        <th><button id="BotonSol" class="BotonNota"> SOL </button></th>
        <th><button id="BotonLa" class="BotonNota"> LA </button></th>
        <th><button id="BotonSi" class="BotonNota"> SI </button></th>
        <th><button id="BotonDoA" class="BotonNota"> DO' </button></th>
        <th><button id="Silencio" class="BotonNota">- </button></th>
        <th>
          <table class="TablaValor">
        <tr>
          <th><button id="ValorNotaDown" class="BotonValor">  <i class=
              "arrow left"></i></button></th>
          <th><div id="ValorNotaVisual" style="background-color: #
              daffdb;"> </div></th>
          <th><button id="ValorNotaUp"   class="BotonValor">  <i class=
              "arrow right"></i></button></th>
        </tr>
        </table>
        </th>
        <th><button id="RMLAST" class="BotonNota">RM</button></th></th>
      </tr>
  </table>

  <script type="importmap">
    {
      "imports": {
        "three": "https://cdn.jsdelivr.net/npm/three@0.166.1/build/
            three.module.js",
        "three/addons/": "https://cdn.jsdelivr.net/npm/three@0.166.1/
            examples/jsm/"
      }
    }
  </script>
  <script type="module" src="/js/index.js" ></script>
  </body>
</html>
```

Proyecto Final - main.js

```

import * as THREE from 'three'

import Stats from 'three/addons/libs/stats.module'
import { OrbitControls } from "three/addons/controls/OrbitControls"
import { TransformControls } from "three/addons/controls/
    TransformControls"
import { degToRad } from 'three/src/math/MathUtils'
import WebGL from 'three/addons/capabilities/WebGL.js'

import { STLLoader } from 'three/addons/loaders/STLLoader'
import { GLTFLoader } from 'three/addons/loaders/GLTFLoader.js'


import { Vex, Stave, StaveNote, Formatter,Renderer } from "vexflow";
const { Factory } = Vex.Flow;


```

```
// //EasyScore
//Ejemplo1

var system,system2,system3,voice,voice2,voice3, voice11

function vexFlowInicial(){

const div = document.getElementById('output');
while(div.firstChild)div.removeChild(div.firstChild);
const renderer = new Renderer(div, Renderer.Backends.SVG);


renderer.resize(380, 140); // Configure the rendering context.
const context = renderer.getContext();
const stave = new Stave(10, 10, 350); // Create a stave of width 350 at
    position 10, 40 on the canvas.
stave.addClef('treble').addTimeSignature('4/4'); // Add a clef and time
    signature.
stave.setContext(context).draw(); // Connect it to the rendering
   context and draw!
console.log(context)
console.log(renderer)
}

let control, scene, camara, renderer1,manoIz,mano3,manoD, bones,bones2,
   skeleton,skeleton2,skeletonHelper2,mano,mano2,lista,listaDedos,
   listaDedosPulsado,listaAnimFramePointer,startTime,
   reproducirPartituraFlag
let gameMode=0
let selector="original"
let texto ='';
let mixer2,mixer3;
let clock

listaAnimFramePointer = [0,0,0,0,0,0,0,0,0,0,0,0]
listaDedosPulsado = [2,1,1,1,1,1,2,1,1,1,1,1]
lista=[]
for(var i =0;i<8;i++){
  // lista.push(i/100)
  lista.push(i*(0.3/8))
}

//Escenario
const canvas = document.getElementById('your_canvas')

//Escena
scene = new THREE.Scene();
const vertex = new THREE.Vector3();

//Materiales-shaders
const mat1 = new THREE.MeshLambertMaterial({
  color: 0xffffff,
  opacity: 0.8,
  transparent: true,
  dithering: false
})

const mat10 = new THREE.MeshPhongMaterial({
  color: 0xe3dac9,
```

```
  emissive: 0xffffff,
  emissiveIntensity: 0.2,
  shininess:40,
  wireframe: false
})

const texture = new THREE.TextureLoader().load('textures/pdtextures/461223193.jpg' );
const matTEX = new THREE.MeshPhongMaterial( { map:texture } );
// const matTEX = new THREE.MeshLambertMaterial( { map:texture } );
matTEX.needsUpdate = true //have to call this

const planoP = new THREE.PlaneGeometry( 30, 30 );
let imagenP = document.getElementById("output").children
const matPlano = new THREE.MeshBasicMaterial(imagenP)

console.log (matPlano)

// scene.add(planoP)
const texture2 = new THREE.TextureLoader().load('textures/flauta_uv2.jpg' );
const matTEX2 = new THREE.MeshBasicMaterial( { map:texture2 } );
matTEX2.needsUpdate = true //have to call this

// const mat2 = new THREE.MeshBasicMaterial({color: 0x7777ff})
// const matBlanco = new THREE.MeshBasicMaterial({color: 'white'})
// const matGris = new THREE.MeshBasicMaterial({color: 'grey'})

//PARTE ARRIBA
//PARTITURA
let grupoPartitura;
grupoPartitura = new THREE.Group();

function crearGUI(){

  grupoPartitura.add(crearBoton(-4,-1,-15,"BotonDo"))

  camara.add (grupoPartitura)

  console.log(scene)
  console.log(grupoPartitura)
  console.log(camara)
}

//RayCaster
let INTERSECTED;
let theta = 0;
const pointer = new THREE.Vector2();
let raycaster = new THREE.Raycaster();

  // when the mouse moves update the point
document.addEventListener('mousemove', (event) => {
  pointer.x = (event.clientX / window.innerWidth) * 2 - 1
  pointer.y = -(event.clientY / window.innerHeight) * 2 + 1
})

document.addEventListener('click',alHacerClick)
function alHacerClick(){
  raycaster.setFromCamera( pointer, camara );
```

```
128   const intersects = raycaster.intersectObjects( scene.children, true )
          ;
129
130   if ( intersects.length > 0 ) {
131   console.log(intersects)
132
133   const intersect = intersects[ 0 ];
134   // switch(intersect.object.name) {
135   //    case 'Linea2': crearNota('no'); break; //do
136   //    case 'Espacio2': crearNota('doa'); break; //do
137   // }
138   }
139 }
140
141 let listaNotasLogica = []
142 let listaNotasLogicaPointer = [0];
143 let listaNotasLogicaPointerGUI = 0;
144 let valorNota=4;
145
146
147
148 let listaN1 = []
149 let listaN1P= [0];
150 let listaN2 = []
151 let listaN2P= [0];
152
153
154
155 function crearNotaL(lista,contadorNota,nombre,tiempo){
156   var tiempoTotal=0
157   var ii=contadorNota[ contadorNota.length - 1];
158
159   if( !gameMode || gameMode && lista.length <12 ){
160   while(ii<lista.length ){
161     tiempoTotal=tiempoTotal+(1/parseInt(lista[ii][1]));
162     ii++;
163     }
164     tiempoTotal=tiempoTotal+(1/parseInt(tiempo))
165     if(tiempoTotal <= 1) lista.push([nombre,tiempo])
166     if(tiempoTotal == 1) contadorNota.push(lista.length)
167
168     if(!gameMode)
169     if(contadorNota.length < 4)
170     actualizarPartituraL(lista,contadorNota,0);
171     else  actualizarPartituraL(lista,contadorNota,contadorNota.length -
            2);
172     else  actualizarPartituraL(lista,contadorNota,0);
173   }
174   }
175
176   function crearNotaSecreta(lista,contadorNota,nombre,tiempo){
177   var tiempoTotal=0
178   var ii=contadorNota[ contadorNota.length - 1];
179
180   while(ii<lista.length ){
181     tiempoTotal=tiempoTotal+(1/parseInt(lista[ii][1]));
182     ii++;
183     }
184     tiempoTotal=tiempoTotal+(1/parseInt(tiempo))
```

```
185     if(tiempoTotal <= 1) lista.push([nombre,tiempo])
186     if(tiempoTotal == 1) contadorNota.push(lista.length)
187   }
188
189 function crearNota(nombre,tiempo){
190   var tiempoTotal=0
191   var ii=listaNotasLogicaPointer[ listaNotasLogicaPointer.length - 1];
192   while(ii<listaNotasLogica.length ){
193     tiempoTotal=tiempoTotal+(1/parseInt(listaNotasLogica[ii][1]))
194     ii++;
195     }
196     tiempoTotal=tiempoTotal+(1/parseInt(tiempo))
197     if(tiempoTotal <= 1)listaNotasLogica.push([nombre,tiempo])
198     if(tiempoTotal == 1)listaNotasLogicaPointer.push(listaNotasLogica.
          length)
199
200
201     if(listaNotasLogicaPointer.length < 4)
202     actualizarPartitura(0);
203     else  actualizarPartitura(listaNotasLogicaPointer.length - 2);
204   }
205
206 function avancePartitura(){
207   if(!gameMode){
208     listaNotasLogica = listaN1
209     listaNotasLogicaPointer= listaN1P
210   }
211   if(listaNotasLogicaPointerGUI < listaNotasLogicaPointer.length - 1)
        listaNotasLogicaPointerGUI++;
212   actualizarPartitura(listaNotasLogicaPointerGUI)
213 }
214
215 function retrocesoPartitura(){
216   if(!gameMode){
217     listaNotasLogica = listaN1
218     listaNotasLogicaPointer= listaN1P
219   }
220   if(listaNotasLogicaPointerGUI > 0)listaNotasLogicaPointerGUI--;
221   actualizarPartitura(listaNotasLogicaPointerGUI)
222 }
223
224 function valorNotaDown(){
225   if(valorNota != 16)valorNota=valorNota*2;
226   valorNotaGUI(valorNota)
227 }
228 function valorNotaUp(){
229   if(valorNota != 1)valorNota=valorNota/2;
230   valorNotaGUI(valorNota)
231 }
232 function valorNotaGUI(valor){
233   var div = document.getElementById('ValorNotaVisual');
234   console.log(div)
235   if(div)div.removeChild(div.firstChild);
236
237 // Create an SVG renderer and attach it to the DIV element
238 var div = document.getElementById("ValorNotaVisual")
239 const VF = Vex.Flow;
240 var renderer = new VF.Renderer(div, VF.Renderer.Backends.SVG);
241 renderer.resize(70, 100);
```

```
242 var context = renderer.getContext();
243
244 var stave = new VF.Stave(10, 40, 20,{ fill_style: 'white' });
245 var notes = [ new VF.StaveNote({ keys: ["g/5"], duration: valor }),];
246 var voice = new VF.Voice({num_beats: 4,  beat_value: 4}).setStrict(
    false);voice.addTickables(notes);
247 var formatter = new VF.Formatter().joinVoices([voice]).format([voice],
    200);
248 voice.draw(context, stave);
249
250 }
251
252 document.getElementById('BotonDo').addEventListener('click', function()
     { crearNotaL(listaN1,listaN1P,'c/4',valorNota) }, false);
253 document.getElementById('BotonRe').addEventListener('click', function()
     { crearNotaL(listaN1,listaN1P,'d/4',valorNota) }, false);
254 document.getElementById('BotonMi').addEventListener('click', function()
     { crearNotaL(listaN1,listaN1P,'e/4',valorNota) }, false);
255 document.getElementById('BotonFa').addEventListener('click', function()
     { crearNotaL(listaN1,listaN1P,'f/4',valorNota) }, false);
256 document.getElementById('BotonSol').addEventListener('click', function
    (){ crearNotaL(listaN1,listaN1P,'g/4',valorNota) }, false);
257 document.getElementById('BotonLa').addEventListener('click', function()
     { crearNotaL(listaN1,listaN1P,'a/4',valorNota) }, false);
258 document.getElementById('BotonSi').addEventListener('click', function()
     { crearNotaL(listaN1,listaN1P,'b/4',valorNota) }, false);
259 document.getElementById('BotonDoA').addEventListener('click', function
    (){ crearNotaL(listaN1,listaN1P,'c/5',valorNota) }, false);
260 document.getElementById('Silencio').addEventListener('click', function
    (){ crearNotaL(listaN1,listaN1P,'b/4',valorNota+'r') }, false);
261
262 document.getElementById('RMLAST').addEventListener('click', function(){
263
264 if(listaN1P.length !=1 && listaN1P[listaN1P.length-1] == listaN1.length
    )listaN1P.pop(listaN1P.length-1)
265 listaN1.pop(listaN1.length-1)
266
267 if(listaN1P.length < 4) actualizarPartituraL(listaN1,listaN1P,0);
268 else  actualizarPartituraL(listaN1,listaN1P,listaN1.length - 2);
269 }, false);
270
271
272 document.getElementById('CambiarModo').addEventListener('click',
    function() {
273   if(gameMode)cambiarModo(0)
274   else cambiarModo(1)
275 }, false);
276
277 document.getElementById('RetrocesoPartitura').addEventListener('click',
     function() { retrocesoPartitura() }, false);
278 document.getElementById('AvancePartitura').addEventListener('click',
    function() { avancePartitura() }, false);
279 document.getElementById('ValorNotaDown').addEventListener('click',
    function() { valorNotaDown() }, false);
280 document.getElementById('ValorNotaUp').addEventListener('click',
    function() { valorNotaUp() }, false);
281
282 function flagsReproduccion(){
283
```

```
284 if(reproducirPartituraFlag){
285
286   reproducirPartituraFlag=0
287   puntero=0;
288   bloque=1;
289   notaActual=0
290
291 }else if(listaNotasLogica.length){
292
293   reproducirPartituraFlag=1
294   puntero=0;
295   bloque=1;
296   notaActual=0
297 }
298
299 }
300
301 document.getElementById('Reproducir').addEventListener('click',
      function() {
302   listaNotasLogica=listaN1
303   listaNotasLogicaPointer=listaN1P
304   selector="original";
305   flagsReproduccion()
306 }, false);
307
308 document.getElementById('Muestra').addEventListener('click', function()
      {
309   listaNotasLogica=listaN2
310   listaNotasLogicaPointer=listaN2P
311   if(selector=="comparar") selector="original";
312   else selector="generada1";
313
314   flagsReproduccion()
315 }, false);
316
317 document.getElementById('MuestraRevelada').addEventListener('click',
      function() {
318   listaNotasLogica=listaN2
319   listaNotasLogicaPointer=listaN2P
320   selector="original";
321
322   flagsReproduccion()
323 }, false);
324
325 document.getElementById('Comparar').addEventListener('click', function
      () {
326   listaNotasLogica=listaN1
327   listaNotasLogicaPointer=listaN1P
328   var ii = 0
329   var aciertos = 0
330   var t1=""
331   var t2=" de 12 \n"
332   for(var ii=0; ii < listaN1.length;ii++){
333     if(listaN1[ii][0] == listaN2[ii][0]){
334     listaNotasLogica[ii].push("green")
335     aciertos ++;
336     }else listaNotasLogica[ii].push("red")
337   }""
338   if(aciertos<6)t1="La proxima lo haras mejos"
```

```
339   else if(aciertos<8)t1="Bien"
340   else if(aciertos<12)t1="Bravo"
341   else if(aciertos==12)t1="Bravo"
342
343 document.getElementById("p1").innerHTML = aciertos + t2;
344 document.getElementById("p2").innerHTML = t1;
345
346 selector="comparar";
347 flagsReproduccion()
348 }, false);
349
350
351 document.getElementById('BotonGuardar').addEventListener('click',
      function() {
352     var elHtml=''
353     var link = document.createElement('a');
354
355     for(var ii=0; ii<listaNotasLogica.length ;ii++){
356       elHtml+=listaNotasLogica[ii]+ ' '
357     }
358
359     link.setAttribute('download', 'Partitura');
360     link.setAttribute('href', 'data:text/plain;charset=utf-8,' +
          encodeURIComponent(elHtml));
361     link.click();
362 }, false);
363
364 document.getElementById('BotonCargar').addEventListener('click',
      function() {
365   var aux
366   aux = texto.split(" ");
367
368   for (var ii=0;ii<aux.length-1;ii++){
369     aux[ii] =  aux[ii].split(",")
370     aux[ii][1] =Number(aux[ii][1])
371   }
372   // aux.pop(aux.length-1)
373   for (ii=0;ii<aux.length-1;ii++) crearNotaL(listaN1,listaN1P,aux[ii
        ][0],aux[ii][1])
374   listaNotasLogica=listaN1
375   listaNotasLogicaPointer=listaN1P
376   actualizarPartitura(0)
377   console.log(aux)
378   // actualizarPartitura(0)
379 }, false);
380
381 document.getElementById('inputfile').addEventListener('change',
      function () {
382   let fr = new FileReader();
383   fr.onload = function () {
384   texto = document.getElementById('textoCarga').textContent = fr.result
        ;
385 }
386   fr.readAsText(this.files[0]);
387
388 }, false);
389
390 // $(document).ready(function(){
391 //     $('#inputfile').on('change', function(e){
```

```
392 //          readFile(this.files[0], function(e) {
393 //              //manipulate with result...
394 //              $('#output').text(e.target.result);
395 //          });
396
397 //      });
398 // });
399
400 // function readFile(file, callback){
401 //      var reader = new FileReader();
402 //      reader.onload = callback
403 //      reader.readAsText(file);
404 // }
405
406
407 //Objeto
408 const cube= new THREE.BoxGeometry(200,100,2)
409 const material= new THREE.MeshPhongMaterial({
410    color: 'red',
411   // wireframe: true
412 })
413 const CubeMesh = new THREE.Mesh(cube, matPlano)
414 CubeMesh.position.x =20
415
416 const cube2= new THREE.BoxGeometry(2,2,2)
417 const material2= new THREE.MeshPhongMaterial({color: 'blue'})
418 const CubeMesh2 = new THREE.Mesh(cube2, matTEX)
419 CubeMesh2.position.z =20
420
421 //cube.vertices.pus
422 CubeMesh.castShadow = true;
423 CubeMesh.receiveShadow = true;
424 CubeMesh.name='Cubo0'
425
426 //  scene.add(CubeMesh)
427  scene.add(CubeMesh2)
428
429 //Objeto externo - flauta
430 const loader = new STLLoader()
431 loader.load( 'models/flauta_uv2.stl', function (geometry) {
432   const flauta = new THREE.Mesh(geometry, matTEX)
433   // const flauta = new THREE.Mesh(geometry)
434
435   flauta.rotateX(degToRad(270))
436   flauta.position.set(0,0,20)
437   flauta.name='Flauta'
438   console.log(geometry)
439   console.log(flauta)
440   scene.add(flauta)
441 }, (xhr) => { console.log((xhr.loaded / xhr.total) * 100 + '% loaded')
     },
442 (error) => { console.log(error) })
443
444
445 //Objeto externo - mano
446 const loader2 = new GLTFLoader();
447 // const manager = new THREE.LoadingManager();
448 loader2.load( 'models/scene.gltf', async function ( gltf ) {
449   console.log(gltf)
```

```
450  var lay1 = gltf.scene.children
451  var lay2 = lay1[0].children
452  var lay3 = lay2[0].children
453  var lay4 = lay3[0].children
454  var lay5 = lay4[0].children
455  var lay6 = lay5[0].children
456  lay6[0].position.x = 0
457  lay6[0].position.y = 0
458  lay6[0].position.z = 0
459  lay6[0].name='MANO';
460  console.log(lay6[0])
461
462  var esfera1 = lay6[0].geometry.boundingSphere.center
463  var es_X,es_Y,es_Z
464  es_X=-esfera1.x
465  es_Y=-esfera1.y
466  es_Z=-esfera1.z
467  esfera1.x=0
468  esfera1.y=0
469  esfera1.z=0
470  for(var ii=0;ii<lay6[0].geometry.attributes.position.array.length;ii=
         ii+3){
471    lay6[0].geometry.attributes.position.array[ii]=lay6[0].geometry.
         attributes.position.array[ii]+es_X;
472    lay6[0].geometry.attributes.position.array[ii+1]=lay6[0].geometry.
         attributes.position.array[ii+1]+es_Y;
473    lay6[0].geometry.attributes.position.array[ii+2]=lay6[0].geometry.
         attributes.position.array[ii+2]+es_Z;
474  }
475  lay6[0].rotateX(degToRad(270))
476  mano=lay6[0]
477
478  const position = mano.geometry.attributes.position;
479  const vertex = new THREE.Vector3();
480
481  const skinIndices = [];
482  const skinWeights = [];
483
484  var cubeIndice = new THREE.Mesh(new THREE.BoxGeometry(20, 50, 17),
       new THREE.MeshBasicMaterial({ color: "aqua", wireframe: true }));
485  var cubeMedio = new THREE.Mesh(new THREE.BoxGeometry(20, 50, 12), new
       THREE.MeshBasicMaterial({ color: "navy", wireframe: true }));
486  var cubeAn = new THREE.Mesh(new THREE.BoxGeometry(20, 50, 10), new
       THREE.MeshBasicMaterial({ color: "red", wireframe: true }));
487  var cubeMenq = new THREE.Mesh(new THREE.BoxGeometry(20, 50, 12), new
       THREE.MeshBasicMaterial({ color: "green", wireframe: true }));
488  var cubePul = new THREE.Mesh(new THREE.BoxGeometry(12, 50, 24), new
       THREE.MeshBasicMaterial({ color: "blue", wireframe: true }));
489
490  cubeIndice.position.set(-15, -25, 17);
491  cubeMedio.position.set(-20, -25, 3);
492  cubeAn.position.set(-20, -24, -8.5);
493  cubeMenq.position.set( -20, -24, -22);
494  cubePul.position.set( 8, -16, 20);
495
496  // scene.add(cubeIndice);
497  // scene.add(cubeMedio);
498  // scene.add(cubeAn);
499  // scene.add(cubeMenq);
```

```
500   // scene.add(cubePul);
501
502   var d1 = new THREE.Box3(); // for re-use
503   d1.setFromObject(cubeIndice);
504   var d2 = new THREE.Box3(); // for re-use
505   d2.setFromObject(cubeMedio);
506   var d3 = new THREE.Box3(); // for re-use
507   d3.setFromObject(cubeAn);
508   var d4 = new THREE.Box3(); // for re-use
509   d4.setFromObject(cubeMenq);
510   var d5 = new THREE.Box3(); // for re-use
511   d5.setFromObject(cubePul);
512
513   for ( let i = 0; i < position.count; i ++ ) {
514     vertex.fromBufferAttribute( position, i );
515     if(d1.containsPoint(vertex)){
516       skinIndices.push( 0, 0, 0, 1 );
517       skinWeights.push( 0, 0, 0, 1 );
518     }else if(d2.containsPoint(vertex)){
519       skinIndices.push( 0, 0, 0, 2 );
520       skinWeights.push( 0, 0, 0, 1 );
521     }else if(d3.containsPoint(vertex)){
522       skinIndices.push( 0, 0, 0, 3 );
523       skinWeights.push( 0, 0, 0, 1 );
524     }else if(d4.containsPoint(vertex)){
525       skinIndices.push( 0, 0, 0, 4 );
526       skinWeights.push( 0, 0, 0, 1 );
527     }else if(d5.containsPoint(vertex)){
528       skinIndices.push( 0, 0, 0, 5 );
529       skinWeights.push( 0, 0, 0, 1 );
530     }else{
531       skinIndices.push( 0, 0, 0, 0 );
532       skinWeights.push( 0, 0, 0, 1 );
533     }
534   }
535   mano.geometry.setAttribute( 'skinIndex', new THREE.
        Uint16BufferAttribute( skinIndices, 4 ) );
536   mano.geometry.setAttribute( 'skinWeight', new THREE.
        Float32BufferAttribute( skinWeights, 4 ) );
537   mano2=mano
538
539   //crear huesos
540   bones = [];
541
542   let prevBone = new THREE.Bone();
543   bones.push( prevBone );
544   prevBone.position.y = 10;
545
546   const bone1 = new THREE.Bone();
547   bones.push( bone1 );
548   prevBone.add( bone1 );
549
550   const bone2 = new THREE.Bone();
551   bones.push( bone2 );
552   prevBone.add( bone2 );
553
554   const bone3= new THREE.Bone();
555   bones.push( bone3 );
556   prevBone.add( bone3 );
```

```
557
558   const bone4 = new THREE.Bone();
559   bones.push( bone4 );
560   prevBone.add( bone4 );
561
562   const bone5 = new THREE.Bone();
563   bones.push( bone5 );
564   prevBone.add( bone5 );
565
566   bones2 = [];
567
568   let prevBone2 = new THREE.Bone();
569   bones2.push( prevBone2 );
570   prevBone2.position.y = 10;
571
572   const b6 = new THREE.Bone();
573   bones2.push( b6 );
574   prevBone2.add( b6 );
575
576   const b7 = new THREE.Bone();
577   bones2.push( b7 );
578   prevBone2.add( b7 );
579
580   const b8 = new THREE.Bone();
581   bones2.push( b8 );
582   prevBone2.add( b8 );
583
584   const b9 = new THREE.Bone();
585   bones2.push( b9 );
586   prevBone2.add( b9 );
587
588   const b10 = new THREE.Bone();
589   bones2.push( b10 );
590   prevBone2.add( b10 );
591
592
593   manoIz = new THREE.SkinnedMesh( mano.geometry,  mat10 );
594   manoD = new THREE.SkinnedMesh( mano2.geometry,  mat10 );
595
596   skeleton = new THREE.Skeleton( bones );
597   skeleton2 = new THREE.Skeleton( bones2 );
598
599   manoIz.add( bones[0]);
600   manoIz.bind( skeleton );
601
602   manoD.add( bones2[0]);
603   manoD.bind( skeleton2 );
604
605   //manoIz.rotation.z=-1
606
607   manoIz.position.set(-36,16,70)
608
609   // manoIz.rotateZ(degToRad(315))
610   // manoIz.rotateY(degToRad(200))
611   // manoIz.rotateX(degToRad(60))
612
613   //  manoIz.rotateX(degToRad(60))
614   manoIz.rotateY(degToRad(170))
615   manoIz.rotateZ(degToRad(295))
```

```
616
617   manoIz.scale.x =1.2
618   manoIz.scale.y =1.2
619   manoIz.scale.z =1.2
620
621   manoIz.skeleton.bones[ 1 ].rotation.x=-0.25
622   manoIz.skeleton.bones[ 2 ].rotation.x=-0.18
623   manoIz.skeleton.bones[ 4 ].rotation.x=0.22
624   scene.add(manoIz)
625
626   manoD.position.set(44,20,0)
627   manoD.scale.x =-1
628   manoD.rotateY(degToRad(-170))
629   manoD.rotateZ(degToRad(-295))
630
631   manoD.skeleton.bones[ 1 ].rotation.x=-0.16
632   // manoD.skeleton.bones[ 2 ].rotation.x=-0.18
633   manoD.skeleton.bones[ 3 ].rotation.x=0.16
634
635   scene.add(manoD)
636
637 },
638
639 (xhr) => { console.log((xhr.loaded / xhr.total) * 100 + '% loaded')},
640 (error) => { console.log(error) }
641 )
642
643 //Suelo
644 const posSuelo = new THREE.PlaneGeometry(6,6)
645 const matSuelo = new THREE.MeshLambertMaterial({color: 'white'})
646 const meshSuelo = new THREE.Mesh(posSuelo,matSuelo)
647 meshSuelo.rotateX(degToRad(270))
648 //scene.add(meshSuelo)
649
650 //Lineas orientacion
651 const tabla = new THREE.GridHelper(100,100)
652 tabla.name='tabla'
653 // scene.add(tabla)
654
655 //Luces
656
657 const ambientLight = new THREE.AmbientLight( 0xffffff, 1.5 );
658 ambientLight.name='ALight'
659 scene.add( ambientLight );
660
661 const pointLight = new THREE.PointLight( 0xffffff, 0.5 );
662 pointLight.position.set(-1,-1,-1)
663 pointLight.name='PLight'
664 scene.add( pointLight );
665
666 const directionalLight = new THREE.DirectionalLight( 0xffffff, 2 );
667 directionalLight.position.set( 1, 1, 1);
668 directionalLight.lookAt(CubeMesh)
669 directionalLight.name='DLight'
670 scene.add( directionalLight );
671
672 //Camara
673
674 let SCREEN_WIDTH = window.innerWidth;
```

```
675 let SCREEN_HEIGHT = window.innerHeight;
676 let aspect = SCREEN_WIDTH / SCREEN_HEIGHT;
677
678  camara = new THREE.PerspectiveCamera(75,aspect,0.1,1000)
679 // camara = new THREE.OrthographicCamera(SCREEN_WIDTH / -2 ,
        SCREEN_WIDTH / 2 ,SCREEN_HEIGHT / 2)
680 const listener = new THREE.AudioListener(); camara.add(listener);
681
682 camara.position.x = -3;
683 camara.position.z = -192;
684 camara.position.y = 65.70;
685
686 camara.name='Camara1'
687 scene.add(camara)
688
689 //Audio
690 //Cubo
691 const mesh1 = new THREE.Mesh(new THREE.BoxGeometry(10, 10, 10), new
        THREE.MeshNormalMaterial({ visible: true }))
692 mesh1.position.set(0, 0, 180)
693 mesh1.name='CuboAudio'
694 scene.add(mesh1)
695 //Do
696 const posSound1 = new THREE.PositionalAudio(listener)
697 const audioLoader1 = new THREE.AudioLoader()
698 audioLoader1.load('/audio/flauta/do.wav', function (buffer) {
699   posSound1.setBuffer(buffer)
700   posSound1.setRefDistance(400)
701   posSound1.setRolloffFactor(50)
702   posSound1.setLoop(false)
703
704   mesh1.add(posSound1)
705
706 })
707 //Re
708 const posSound2 = new THREE.PositionalAudio(listener)
709 const audioLoader2 = new THREE.AudioLoader()
710 audioLoader2.load('/audio/flauta/re.wav', function (buffer) {
711   posSound2.setBuffer(buffer)
712   posSound2.setRefDistance(400)
713   posSound2.setRolloffFactor(50)
714   posSound2.delayNode
715   posSound2.setLoop(false)
716   mesh1.add(posSound2)
717 })
718 //Mi
719 const posSound3 = new THREE.PositionalAudio(listener)
720 const audioLoader3 = new THREE.AudioLoader()
721 audioLoader3.load('/audio/flauta/mi.wav', function (buffer) {
722   posSound3.setBuffer(buffer)
723   posSound3.setRefDistance(400)
724   posSound3.setRolloffFactor(50)
725   posSound3.setLoop(false)
726   mesh1.add(posSound3)
727 })
728 //Fa
729 const posSound4 = new THREE.PositionalAudio(listener)
730 const audioLoader4 = new THREE.AudioLoader()
731 audioLoader4.load('/audio/flauta/fa.wav', function (buffer) {
```

```
732   posSound4.setBuffer(buffer)
733   posSound4.setRefDistance(400)
734   posSound4.setRolloffFactor(50)
735   posSound4.setLoop(false)
736   mesh1.add(posSound4)
737 })
738 //Sol
739 const posSound5 = new THREE.PositionalAudio(listener)
740 const audioLoader5 = new THREE.AudioLoader()
741 audioLoader5.load('/audio/flauta/sol.wav', function (buffer) {
742   posSound5.setBuffer(buffer)
743   posSound5.setRefDistance(400)
744   posSound5.setRolloffFactor(50)
745   posSound5.setLoop(false)
746   mesh1.add(posSound5)
747 })
748 //La
749 const posSound6 = new THREE.PositionalAudio(listener)
750 const audioLoader6 = new THREE.AudioLoader()
751 audioLoader6.load('/audio/flauta/la.wav', function (buffer) {
752   posSound6.setBuffer(buffer)
753   posSound6.setRefDistance(400)
754   posSound6.setRolloffFactor(50)
755   posSound6.setLoop(false)
756   mesh1.add(posSound6)
757 })
758 //Si
759 const posSound7 = new THREE.PositionalAudio(listener)
760 const audioLoader7 = new THREE.AudioLoader()
761 audioLoader7.load('/audio/flauta/si.wav', function (buffer) {
762   posSound7.setBuffer(buffer)
763   posSound7.setRefDistance(400)
764   posSound7.setRolloffFactor(50)
765   posSound7.setLoop(false)
766   mesh1.add(posSound7)
767 })
768 //DoA
769 const posSound8 = new THREE.PositionalAudio(listener)
770 const audioLoader8 = new THREE.AudioLoader()
771 audioLoader8.load('/audio/flauta/do_alt.wav', function (buffer) {
772   posSound8.setBuffer(buffer)
773   posSound8.setRefDistance(400)
774   posSound8.setRolloffFactor(50)
775   posSound8.setLoop(false)
776   mesh1.add(posSound8)
777 })
778
779
780 // add stats
781 const stats = Stats();
782 document.body.appendChild(stats.dom);
783
784 //Visualizacion
785 renderer1 = new THREE.WebGLRenderer({
786     antialias: true ,
787     canvas: canvas
788 })
789 renderer1.outputColorSpace = THREE.SRGBColorSpace;
790 renderer1.shadowMap.enabled = true;
```

```
renderer1.shadowMap.type = THREE.VSMShadowMap;
renderer1.setSize(SCREEN_WIDTH,SCREEN_HEIGHT)
renderer1.setClearColor(0x111111);
//document.body.appendChild(renderer.domElement);

// add orbitcontrols
const controller = new OrbitControls(camara, renderer1.domElement);

controller.enableDamping = true;
controller.dampingFactor = 0.1;
controller.minDistance = 10;
controller.maxDistance = 380;
controller.mouseButtons = {
    LEFT: THREE.MOUSE.ROTATE,
    MIDDLE: THREE.MOUSE.DOLLY,
    RIGHT: ''
}


//controller.minPolarAngle = Math.PI / 4;
//controller.maxPolarAngle = (3 * Math.PI) / 4;

//evento pulsar teclas
document.addEventListener('keydown', logKey);
// document.addEventListener( 'pointerdown', onPointerDown );
// document.addEventListener( 'pointerup', onPointerUp );
// document.addEventListener( 'pointermove', onPointerMove );

function logKey(e) {
  switch(e.code){
    case 'KeyA':
    animarDedo([2,0,0,0,0,2,2,0,0,0,2,2]);
    setTimeout(() => {posSound1.play(); }, 600);
    break; //do
    case 'KeyS':
    animarDedo([2,0,0,0,1,2,2,0,0,0,2,2]);
    setTimeout(() => { posSound2.play(); }, 600);
    break; //re
    case 'KeyD':
    animarDedo([2,0,0,1,1,2,2,0,0,0,2,2]);
    setTimeout(() => { posSound3.play(); }, 600);
    break; //mi
    case 'KeyF':
    animarDedo([2,0,1,1,1,2,2,0,0,0,2,2]);
    setTimeout(() => { posSound4.play(); }, 600);
    break; //fa
    case 'KeyG':
    animarDedo([2,1,1,1,1,2,2,0,0,0,2,2]);
    setTimeout(() => { posSound5.play(); }, 600);
    break; //sol
    case 'KeyH':
    animarDedo([2,1,1,1,1,2,2,0,0,1,2,2]);
    setTimeout(() => { posSound6.play(); }, 600);
    break; //la
    case 'KeyJ':
    animarDedo([2,1,1,1,1,2,2,0,1,1,2,2]);
    setTimeout(() => { posSound7.play(); }, 600);
    break; //si
    case 'KeyK':
```

```
850       animarDedo([2,1,1,1,1,2,2,1,0,1,2,2]);
851       setTimeout(() => { posSound8.play(); }, 600);
852       break; //doa
853
854       case 'KeyY':
855       camara.children[1].children[11].rotateX(degToRad(.1))
856       break; //doa
857       case 'KeyU':
858       camara.children[1].children[11].rotateX(degToRad(-.1))
859       break; //doa
860
861       case 'KeyC': cambioCam();    break;
862       case 'KeyB':
863       listaNotasLogica=[];
864       listaNotasLogicaPointer=[];
865
866       break;
867       case 'KeyR': actualizarPartitura(0);    break;
868       case 'KeyT':
869       //  console.log(lista)
870       //  console.log(listaAnimFramePointer)
871       //  console.log(listaDedosPulsado)
872       console.log(texto)
873       console.log("LA REPRODUCIR")
874       console.log(listaNotasLogica)
875       console.log(listaNotasLogicaPointer)
876
877       console.log(listaNotasLogicaPointerGUI)
878       console.log("L1")
879       console.log(listaN1)
880       console.log(listaN1P)
881
882       console.log("L2")
883       console.log(listaN2)
884       console.log(listaN2P)
885       // console.log(camara.position)
886       // console.log(mesh1.position)
887       //  console.log(clock)
888       //  console.log(posSound1.context)
889       break; //
890   }
891 };
892
893 function actualizarPartituraL(lista,contadorNota,inicio){
894   if(lista){
895   var div = document.getElementById('output');
896   while(div.firstChild)div.removeChild(div.firstChild);
897   const vf = new Factory({ renderer: { elementId: 'output', width: 400,
          height: 140 }, });
898   system = vf.System({width:350 });
899   var notes1=[];
900   var ii = contadorNota[inicio];
901   while(ii < lista.length ){
902     if(ii == contadorNota[inicio + 1])break;
903     notes1.push(vf.StaveNote({keys: [lista[ii][0]],duration: lista[ii
          ][1]}))
904     ii++
905   }
906
```

```
907   voice = vf.Voice().addTickables(notes1);
908   voice.setStrict(false);
909
910
911   system.addStave({voices: [voice] })
912   .addClef('treble')
913   .addTimeSignature('4/4');
914   vf.draw();
915
916   system2 = vf.System({x:10 + system.options.width , width:350});
917   var notes2=[];
918
919   while(ii < lista.length ){
920   if(ii == contadorNota[inicio + 2])break;
921   notes2.push(vf.StaveNote({keys: [lista[ii][0]],duration: lista[ii
          ][1]}))
922   ii++
923   }
924   if(notes2.length != 0){
925   voice2 = vf.Voice().addTickables(notes2);
926   voice2.setStrict(false);
927   system2.addStave({voices: [voice2] })
928   vf.draw();
929   }
930
931   //3reglon
932   system3 = vf.System({x:10 + system.options.width + system2.options.
          width , width:350});
933   var notes3=[];
934   while(ii < lista.length ){
935     if(ii == contadorNota[inicio + 3])break;
936     notes3.push(vf.StaveNote({keys: [lista[ii][0]],duration: lista[ii
            ][1]}))
937     ii++
938   }
939   if(notes3.length != 0){
940     voice3 = vf.Voice().addTickables(notes3);
941     voice3.setStrict(false);
942     system3.addStave({voices: [voice3] })
943     vf.draw();
944   }
945   }
946 }
947
948 function actualizarPartitura(inicio){
949   if(listaNotasLogica){
950   var div = document.getElementById('output');
951   while(div.firstChild)div.removeChild(div.firstChild);
952   const vf = new Factory({ renderer: { elementId: 'output', width: 400,
          height: 140 }, });
953   system = vf.System({width:350 });
954   var notes1=[];
955   var ii = listaNotasLogicaPointer[inicio];
956   while(ii < listaNotasLogica.length ){
957     if(ii == listaNotasLogicaPointer[inicio + 1])break;
958     notes1.push(vf.StaveNote({
959     keys: [listaNotasLogica[ii][0]],
960     duration: listaNotasLogica[ii][1]}))
961     ii++}
```

```
962   voice = vf.Voice().addTickables(notes1);
963   voice.setStrict(false);
964   system.addStave({voices: [voice] })
965   .addClef('treble')
966   .addTimeSignature('4/4');
967   vf.draw();
968
969   system2 = vf.System({x:10 + system.options.width , width:350});
970   var notes2=[];
971
972   while(ii < listaNotasLogica.length ){
973   if(ii == listaNotasLogicaPointer[inicio + 2])break;
974   notes2.push(vf.StaveNote({keys: [listaNotasLogica[ii][0]],duration:
          listaNotasLogica[ii][1]}))
975   ii++}
976   if(notes2.length != 0){
977   voice2 = vf.Voice().addTickables(notes2);
978   voice2.setStrict(false);
979   system2.addStave({voices: [voice2] })
980   vf.draw();}
981
982   //3reglon
983   system3 = vf.System({x:10 + system.options.width + system2.options.
          width , width:350});
984   var notes3=[];
985   while(ii < listaNotasLogica.length ){
986     if(ii == listaNotasLogicaPointer[inicio + 3])break;
987     notes3.push(vf.StaveNote({keys: [listaNotasLogica[ii][0]],duration:
            listaNotasLogica[ii][1]}))
988     ii++}
989   if(notes3.length != 0){
990     voice3 = vf.Voice().addTickables(notes3);
991     voice3.setStrict(false);
992     system3.addStave({voices: [voice3] })
993     vf.draw();}
994   }
995 }
996
997 function actualizarPartituraDestacadoL(lista,contadorNota,inicio,
      destacado){
998   if(lista){
999   var div = document.getElementById('output');
1000  while(div.firstChild)div.removeChild(div.firstChild);
1001  const vf = new Factory({ renderer: { elementId: 'output', width: 400,
          height: 140 }, });
1002  system = vf.System({width:350 });
1003  var notes1=[];
1004  var ii = contadorNota[inicio];
1005  while(ii < lista.length ){
1006    if(ii == contadorNota[inicio + 1])break;
1007    if(ii == destacado) notes1.push(vf.StaveNote({keys: [lista[ii][0]],
          duration: lista[ii][1]}).setStyle({ fillStyle: 'red',
          strokeStyle: 'red' }))
1008    else notes1.push(vf.StaveNote({keys: [lista[ii][0]],duration: lista
          [ii][1]}))
1009    ii++
1010  }
1011  voice = vf.Voice().addTickables(notes1);
1012  voice.setStrict(false);
```

```
1013
1014    system.addStave({voices: [voice] })
1015    .addClef('treble')
1016    .addTimeSignature('4/4');
1017    vf.draw();
1018
1019    system2 = vf.System({x:10 + system.options.width , width:350});
1020    var notes2=[];
1021
1022    while(ii < lista.length ){
1023      if(ii == contadorNota[inicio + 2])break;
1024      if(ii == destacado) notes2.push(vf.StaveNote({keys: [lista[ii][0]],
            duration: lista[ii][1]}).setStyle({ fillStyle: 'red',
            strokeStyle: 'red' }))
1025      else notes2.push(vf.StaveNote({keys: [lista[ii][0]],duration: lista
            [ii][1]}))
1026      ii++
1027    }
1028    if(notes2.length != 0){
1029    voice2 = vf.Voice().addTickables(notes2);
1030    voice2.setStrict(false);
1031    system2.addStave({voices: [voice2] })
1032    vf.draw();
1033    }
1034
1035    //3reglon
1036    system3 = vf.System({x:10 + system.options.width + system2.options.
          width , width:350});
1037    var notes3=[];
1038    while(ii < lista.length ){
1039      if(ii == contadorNota[inicio + 3])break;
1040      if(ii == destacado) notes3.push(vf.StaveNote({keys: [lista[ii][0]],
            duration: lista[ii][1]}).setStyle({ fillStyle: 'red',
            strokeStyle: 'red' }))
1041      else notes3.push(vf.StaveNote({keys: [lista[ii][0]],duration: lista
            [ii][1]}))
1042      ii++
1043    }
1044    if(notes3.length != 0){
1045    voice3 = vf.Voice().addTickables(notes3);
1046    voice3.setStrict(false);
1047    system3.addStave({voices: [voice3] })
1048    vf.draw();
1049    }
1050    }
1051  }
1052
1053  function actualizarPartituraDestacado(inicio,destacado){
1054    if(listaNotasLogica){
1055    var div = document.getElementById('output');
1056    while(div.firstChild)div.removeChild(div.firstChild);
1057
1058    const vf = new Factory({ renderer: { elementId: 'output', width: 400,
          height: 140 }, });
1059    system = vf.System({width:350 });
1060    var notes1=[];
1061    var ii = listaNotasLogicaPointer[inicio];
1062    while(ii < listaNotasLogica.length ){
1063      if(ii == listaNotasLogicaPointer[inicio + 1])break;
```

```
1064      if(ii != destacado)
1065      notes1.push(vf.StaveNote({
1066      keys: [listaNotasLogica[ii][0]],
1067      duration: listaNotasLogica[ii][1]}))
1068      else
1069      notes1.push(vf.StaveNote({
1070      keys: [listaNotasLogica[ii][0]],
1071      duration: listaNotasLogica[ii][1]})
1072      .setStyle({ fillStyle: 'red', strokeStyle: 'red' }))
1073
1074      ii++
1075    }
1076
1077    voice = vf.Voice().addTickables(notes1);
1078    voice.setStrict(false);
1079
1080    system.addStave({voices: [voice] })
1081    .addClef('treble')
1082    .addTimeSignature('4/4');
1083    vf.draw();
1084
1085    system2 = vf.System({x:10 + system.options.width , width:350});
1086    var notes2=[];
1087
1088    while(ii < listaNotasLogica.length ){
1089      if(ii == listaNotasLogicaPointer[inicio + 2])break;
1090      if(ii != destacado)
1091      notes2.push(vf.StaveNote({
1092      keys: [listaNotasLogica[ii][0]],
1093      duration: listaNotasLogica[ii][1]}))
1094      else
1095      notes2.push(vf.StaveNote({
1096      keys: [listaNotasLogica[ii][0]],
1097      duration: listaNotasLogica[ii][1]})
1098      .setStyle({ fillStyle: 'red', strokeStyle: 'red' }))
1099      ii++
1100    }
1101    if(notes2.length != 0){
1102    voice2 = vf.Voice().addTickables(notes2);
1103    voice2.setStrict(false);
1104    system2.addStave({voices: [voice2] })
1105    vf.draw();
1106    }
1107
1108    //3reglon
1109    system3 = vf.System({x:10 + system.options.width + system2.options.
         width , width:350});
1110    var notes3=[];
1111    while(ii < listaNotasLogica.length ){
1112      if(ii == listaNotasLogicaPointer[inicio + 3])break;
1113      if(ii != destacado)
1114      notes3.push(vf.StaveNote({
1115      keys: [listaNotasLogica[ii][0]],
1116      duration: listaNotasLogica[ii][1]}))
1117      else
1118      notes3.push(vf.StaveNote({
1119      keys: [listaNotasLogica[ii][0]],
1120      duration: listaNotasLogica[ii][1]})
1121      .setStyle({ fillStyle: 'red', strokeStyle: 'red' }))
```

```
1122      ii++
1123      }
1124    if(notes3.length != 0){
1125    voice3 = vf.Voice().addTickables(notes3);
1126    voice3.setStrict(false);
1127    system3.addStave({voices: [voice3] })
1128    vf.draw();
1129    }}}
1130
1131  function compararPartitura(inicio,destacado){
1132    if(listaNotasLogica){
1133    var div = document.getElementById('output');
1134    while(div.firstChild)div.removeChild(div.firstChild);
1135
1136    const vf = new Factory({ renderer: { elementId: 'output', width: 400,
          height: 140 }, });
1137    system = vf.System({width:350 });
1138    var notes1=[];
1139    var ii = listaNotasLogicaPointer[inicio];
1140    while(ii < listaNotasLogica.length ){
1141      if(ii == listaNotasLogicaPointer[inicio + 1])break;
1142      if(ii <= destacado){
1143      notes1.push(vf.StaveNote({
1144      keys: [listaNotasLogica[ii][0]],
1145      duration: listaNotasLogica[ii][1]})
1146      .setStyle({ fillStyle: listaNotasLogica[ii][2], strokeStyle:
            listaNotasLogica[ii][2] }))
1147      }else{
1148      notes1.push(vf.StaveNote({
1149      keys: [listaNotasLogica[ii][0]],
1150      duration: listaNotasLogica[ii][1]}))
1151      }
1152      ii++
1153    }
1154    console.log(notes1)
1155    voice = vf.Voice().addTickables(notes1);
1156    voice.setStrict(false);
1157
1158    system.addStave({voices: [voice]})
1159    .addClef('treble')
1160    .addTimeSignature('4/4');
1161    vf.draw();
1162
1163    system2 = vf.System({x:10 + system.options.width , width:350});
1164    var notes2=[];
1165
1166    while(ii < listaNotasLogica.length ){
1167      if(ii == listaNotasLogicaPointer[inicio + 2])break;
1168      if(ii <= destacado){
1169      notes2.push(vf.StaveNote({
1170      keys: [listaNotasLogica[ii][0]],
1171      duration: listaNotasLogica[ii][1]})
1172      .setStyle({ fillStyle: listaNotasLogica[ii][2], strokeStyle:
            listaNotasLogica[ii][2] }))
1173      }else{
1174      notes2.push(vf.StaveNote({
1175      keys: [listaNotasLogica[ii][0]],
1176      duration: listaNotasLogica[ii][1]}))}
1177      ii++
```

```
1178    }
1179    if(notes2.length != 0){
1180    voice2 = vf.Voice().addTickables(notes2);
1181    voice2.setStrict(false);
1182    system2.addStave({voices: [voice2] })
1183    vf.draw();
1184    }
1185
1186    //3reglon
1187    system3 = vf.System({x:10 + system.options.width + system2.options.
            width , width:350});
1188    var notes3=[];
1189    while(ii < listaNotasLogica.length ){
1190      if(ii == listaNotasLogicaPointer[inicio + 3])break;
1191      if(ii <= destacado){
1192      notes3.push(vf.StaveNote({
1193      keys: [listaNotasLogica[ii][0]],
1194      duration: listaNotasLogica[ii][1]})
1195      .setStyle({ fillStyle: listaNotasLogica[ii][2], strokeStyle:
              listaNotasLogica[ii][2] }))
1196      }else{
1197      notes3.push(vf.StaveNote({
1198      keys: [listaNotasLogica[ii][0]],
1199      duration: listaNotasLogica[ii][1]}))}
1200      ii++
1201    }
1202    if(notes3.length != 0){
1203    voice3 = vf.Voice().addTickables(notes3);
1204    voice3.setStrict(false);
1205    system3.addStave({voices: [voice3] })
1206    vf.draw();
1207    }}}
1208
1209  function selectorVisualizacionPartitura(origen){
1210  switch(origen){
1211    case "original":
1212    reproducirPartitura();
1213    break;
1214    case "generada1":
1215    reproducirPartituraNoGraphics();
1216    break;
1217    case "comparar":
1218    compararPartituraL();
1219    break;
1220    }
1221  }
1222
1223  function compararPartituraL(){
1224   compararPartitura(0,notaActual)
1225   animarDedoDesdePartitura(listaNotasLogica[notaActual])
1226   animarSonido(listaNotasLogica[notaActual])
1227  }
1228
1229  function reproducirPartituraL(lista,contadorNota){
1230   actualizarPartituraDestacado(lista,contadorNota, puntero,notaActual)
1231   animarDedoDesdePartitura(lista[notaActual])
1232   animarSonido(lista[notaActual])
1233  }
1234
```

```
1235 function reproducirPartituraNoGraphics(){
1236 //  actualizarPartituraDestacado(lista,contadorNota, puntero,notaActual
         )
1237 //  animarDedoDesdePartitura(listaNotasLogica[notaActual])
1238  animarSonido(listaNotasLogica[notaActual])
1239
1240 }
1241
1242 function reproducirPartitura(){
1243  if(!gameMode)actualizarPartituraDestacado(puntero,notaActual)
1244  else actualizarPartituraDestacado(0,notaActual)
1245  animarDedoDesdePartitura(listaNotasLogica[notaActual])
1246  animarSonido(listaNotasLogica[notaActual])
1247 }
1248
1249 function animarSonido(nota){
1250
1251   if(typeof nota[1]=='number') {
1252   switch(nota[0]){
1253     case 'c/4': //do
1254     posSound1.stop();
1255     setTimeout(() => {posSound1.play(); }, 400/nota[1]);
1256     break;
1257     case 'd/4': //re
1258     posSound2.stop();
1259     setTimeout(() => { posSound2.play(); }, 400/nota[1]);
1260     break;
1261     case 'e/4': //mi
1262     posSound3.stop();
1263     setTimeout(() => { posSound3.play(); }, 400/nota[1]);
1264     break;
1265     case 'f/4': //fa
1266     posSound4.stop();
1267     setTimeout(() => { posSound4.play(); }, 400/nota[1]);
1268     break;
1269     case 'g/4': //sol
1270     posSound5.stop();
1271     setTimeout(() => { posSound5.play(); }, 400/nota[1]);
1272     break;
1273     case 'a/4': //la
1274     posSound6.stop();
1275     setTimeout(() => { posSound6.play(); }, 400/nota[1]);
1276     break;
1277     case 'b/4':  //si
1278     posSound7.stop();
1279     setTimeout(() => { posSound7.play(); }, 400/nota[1]);
1280     break;
1281     case 'c/5': //doa
1282     posSound8.stop();
1283     setTimeout(() => { posSound8.play(); }, 400/nota[1]);
1284     break;
1285   }}}
1286
1287 function animarDedoDesdePartitura(nota){
1288
1289   if(typeof nota[1]=='number') {
1290   switch(nota[0]){
1291     case 'c/4': //do
1292     animarDedo([2,0,0,0,0,2,2,0,0,0,2,2]);
```

```
1293      posSound1.stop();
1294      setTimeout(() => {posSound1.play(); }, 400/nota[1]);
1295      break;
1296      case 'd/4': //re
1297      animarDedo([2,0,0,0,1,2,2,0,0,0,2,2]);
1298      posSound2.stop();
1299      setTimeout(() => { posSound2.play(); }, 400/nota[1]);
1300      break;
1301      case 'e/4': //mi
1302      animarDedo([2,0,0,1,1,2,2,0,0,0,2,2]);
1303      posSound3.stop();
1304      setTimeout(() => { posSound3.play(); }, 400/nota[1]);
1305      break;
1306      case 'f/4': //fa
1307      animarDedo([2,0,1,1,1,2,2,0,0,0,2,2]);
1308      posSound4.stop();
1309      setTimeout(() => { posSound4.play(); }, 400/nota[1]);
1310      break;
1311      case 'g/4': //sol
1312      animarDedo([2,1,1,1,1,2,2,0,0,0,2,2]);
1313      posSound5.stop();
1314      setTimeout(() => { posSound5.play(); }, 400/nota[1]);
1315      break;
1316      case 'a/4': //la
1317      animarDedo([2,1,1,1,1,2,2,0,0,1,2,2]);
1318      posSound6.stop();
1319      setTimeout(() => { posSound6.play(); }, 400/nota[1]);
1320      break;
1321      case 'b/4':  //si
1322      animarDedo([2,1,1,1,1,2,2,0,1,1,2,2]);
1323      posSound7.stop();
1324      setTimeout(() => { posSound7.play(); }, 400/nota[1]);
1325      break;
1326      case 'c/5': //doa
1327      animarDedo([2,1,1,1,1,2,2,1,0,1,2,2]);
1328      posSound8.stop();
1329      setTimeout(() => { posSound8.play(); }, 400/nota[1]);
1330      break;
1331    }}}
1332
1333  function animarDedo(e){
1334    for(var ii = 0;ii<listaDedosPulsado.length;ii++){
1335      if(e[ii]==0) listaDedosPulsado[ii]= 0
1336      if(e[ii]==1) listaDedosPulsado[ii]= 1
1337      }}
1338
1339  function animandoDedo(){
1340    for(var ii = 0;ii<6;ii++){
1341    if(listaDedosPulsado[ ii ]==0 && listaAnimFramePointer[ ii ]<7 )
         listaAnimFramePointer[ii]++
1342    if(listaDedosPulsado[ ii ]==1 && listaAnimFramePointer[ ii ]>0)
         listaAnimFramePointer[ii]--
1343    manoIz.skeleton.bones[ ii ].rotation.z = lista[listaAnimFramePointer[
         ii ]];
1344    }for(var ii = 6;ii<12;ii++){
1345    if(listaDedosPulsado[ ii ]==0 && listaAnimFramePointer[ ii ]<7 )
         listaAnimFramePointer[ii]++
1346    if(listaDedosPulsado[ ii ]==1 && listaAnimFramePointer[ ii ]>0)
         listaAnimFramePointer[ii]--
```

```
1347      manoD.skeleton.bones[ ii-6 ].rotation.z = lista[
              listaAnimFramePointer[ ii ]];
1348    }}
1349
1350
1351  function cambiarModo(modoNuevo){
1352    gameMode=modoNuevo
1353    listaN1=[]
1354    listaN1P=[0]
1355
1356    if(gameMode){
1357    document.getElementById("BotonCargar").style.visibility = "hidden";
1358    document.getElementById("BotonGuardar").style.visibility = "hidden";
1359    document.getElementById("inputfile").style.visibility = "hidden";
1360    document.getElementById("TM1").style.visibility = "visible";
1361
1362    document.getElementById("ValorNotaDown").style.visibility = "hidden";
1363    document.getElementById("ValorNotaUp").style.visibility = "hidden";
1364    document.getElementById("RetrocesoPartitura").style.visibility = "
          hidden";
1365    document.getElementById("AvancePartitura").style.visibility = "hidden
          ";
1366
1367    valorNota=4
1368    valorNotaGUI(valorNota)
1369    crearPartituraAleatoria()
1370    } else {
1371      document.getElementById("BotonCargar").style.visibility = "visible"
            ;
1372      document.getElementById("BotonGuardar").style.visibility = "visible
            ";
1373      document.getElementById("inputfile").style.visibility = "visible";
1374      document.getElementById("TM1").style.visibility = "hidden";
1375
1376      document.getElementById("ValorNotaDown").style.visibility = "
            visible";
1377    document.getElementById("ValorNotaUp").style.visibility = "visible";
1378    document.getElementById("RetrocesoPartitura").style.visibility = "
          visible";
1379    document.getElementById("AvancePartitura").style.visibility = "
          visible";
1380    } }
1381
1382  function crearPartituraAleatoria(){
1383    listaN2=[]
1384    listaN2P=[0]
1385    for(var i =0;i<12;i++){
1386    switch(Math.floor(Math.random() * 9)){
1387
1388      case 0:crearNotaSecreta(listaN2,listaN2P,'c/4',4);break;
1389      case 1:crearNotaSecreta(listaN2,listaN2P,'d/4',4);break;
1390      case 2:crearNotaSecreta(listaN2,listaN2P,'e/4',4);break;
1391      case 3:crearNotaSecreta(listaN2,listaN2P,'f/4',4);break;
1392      case 4:crearNotaSecreta(listaN2,listaN2P,'g/4',4);break;
1393      case 5:crearNotaSecreta(listaN2,listaN2P,'a/4',4);break;
1394      case 6:crearNotaSecreta(listaN2,listaN2P,'b/4',4);break;
1395      case 7:crearNotaSecreta(listaN2,listaN2P,'c/5',4);break;
1396      case 8:crearNotaSecreta(listaN2,listaN2P,'b/4',4+'r');break;
1397
```

```
1398    }
1399 }
1400 vexFlowInicial()
1401 }
1402
1403 var cambioCam = function(){
1404    camara.position.x = -3;
1405    camara.position.z = -192;
1406    camara.position.y = 65.70;
1407    camara.lookAt(0,0,0)
1408    controller.reset
1409
1410 };
1411
1412 window.addEventListener( 'resize', onWindowResize );
1413
1414 function onWindowResize() {
1415
1416    SCREEN_WIDTH = window.innerWidth;
1417    SCREEN_HEIGHT = window.innerHeight;
1418    aspect = SCREEN_WIDTH / SCREEN_HEIGHT;
1419
1420    renderer1.setSize( SCREEN_WIDTH, SCREEN_HEIGHT );
1421
1422    // console.log(document)
1423    camara.aspect = aspect;
1424    camara.updateProjectionMatrix();
1425
1426
1427 }
1428
1429 clock = new THREE.Clock();
1430
1431 vexFlowInicial()
1432 valorNotaGUI(valorNota)
1433
1434 startTime = Date.now();
1435
1436 var puntero=0;
1437 var bloque=1;
1438 var notaActual=0
1439 var time1
1440 var time2 = 0;
1441 var ciclo=0;
1442 var siguienteCiclo=0
1443
1444
1445
1446 if ( WebGL.isWebGLAvailable() ) {
1447 function animate() {
1448    requestAnimationFrame(animate);
1449    camara.updateMatrixWorld();
1450    stats.update();
1451    animandoDedo()
1452
1453    time1=(Date.now() - time2)
1454    if(reproducirPartituraFlag) {
1455       if (time1>(120)){
1456          if(ciclo>=siguienteCiclo){
```

```
1457
1458          var numeroCiclo = listaNotasLogica[notaActual][1]
1459
1460          //Es silencio
1461          if(typeof numeroCiclo=='string') { numeroCiclo= parseInt(
                 numeroCiclo.slice(0,-1))
1462          console.log(numeroCiclo)
1463          }
1464          //calcula cuanto dura la nota
1465          siguienteCiclo= siguienteCiclo + 16/numeroCiclo;
1466          //puntero para colorear nota
1467          if(notaActual== listaNotasLogicaPointer[bloque]){
1468          puntero++;
1469          bloque++;
1470          }
1471          selectorVisualizacionPartitura(selector)
1472          notaActual++;
1473        }
1474        //calcula siguiente hora y actualiza ciclo
1475        time2=Date.now()
1476        ciclo++;
1477        //termina la reproduccion y reinicia contadores
1478      if(listaNotasLogica.length == notaActual){
1479        reproducirPartituraFlag = 0
1480        notaActual = 0
1481        ciclo = 0
1482        siguienteCiclo = 0
1483        listaNotasLogicaPointerGUI = 0
1484        bloque = 1
1485        puntero = 1
1486        }
1487      }
1488    }
1489    renderer1.render(scene, camara);
1490    controller.update
1491    }
1492    animate();
1493  } else {
1494    const warning = WebGL.getWebGLErrorMessage();
1495    document.getElementById( 'container' ).appendChild( warning );
1496  }
```

Proyecto Final - mystyle.css

```
1  .BotonNota {
2      padding-top: 30px;
3      padding-right: 30px;
4      padding-bottom: 30px;
5      padding-left: 30px;
6      background-color: #4CAF50;
7      color: rgb(202, 202, 202);
8      border: none;
9      border-radius: 100px;
10     cursor: pointer;
11     width: 5vw;
12     height: 5vw;
13
14     z-index: 1000;
15     /* font-size:2.5em; */
```

```
    font-size:2.5em;
    text-align: center;
    font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS',
        sans-serif;

  }

  .BotonValor {
    padding-top: 30px;
    padding-right: 30px;
    padding-bottom: 30px;
    padding-left: 30px;
    background-color: #daffdb;
    border: none;
    border-radius: 100px;
    cursor: pointer;
    width: 80;
    z-index: 1000;
    font-size:2.5em;
    text-align: center;
    font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS',
        sans-serif;
  }

  .TablaNotas {
    position: absolute;
    bottom: 0;
    left:0;
    width: 80%;
    height: 10%;
    background-color: grey;
    /* transform: translate(-50%, -50%); */

    /* border: 1px none; */
    border-radius: 10px;
    justify-content: space-between;
    z-index: 1000;

    text-align: center;
    align-items: center;
  }

  .TablaPartitura {
    position: absolute;
    top: 0;
    /* left: */
    width: 100%;
    height: 10%;
    background-color: grey;

    /* border: 1px none; */
    border-radius: 10px;
    justify-content: space-between;
    z-index: 1000;

    text-align: center;
    align-items: center;
  }
.output{
```

```
  background-color: white;
}

  .TablaLateral {
    position: absolute;
    top: 16%;
    /* left:  */
    width: 10%;
    height: 10%;
    background-color: grey;

    /* border: 1px none; */
    border-radius: 10px;
    justify-content: space-between;
    z-index: 1000;

    text-align: center;
    align-items: center;
  }

  .TablaModo1 {
    position: absolute;
    top: 16%;
    left:  15%;
    width: 5%;
    height: 5%;
    background-color: grey;

    margin-left: auto;
    margin-right: auto;

    visibility: hidden;
    /* border: 1px none; */
    border-radius: 10px;
    justify-content: space-between;
    z-index: 1000;

    text-align: center;
    align-items: center;
    align-self: center;
  }

  th,td {
    margin: 10px;
    font-size:1em;
  }

  .arrow {
    border: solid rgb(255, 196, 114);
    border-width: 0 3px 3px 0;
    display: inline-block;
    font-size: 0.5em;
    padding: 3px;
  }

  .right {
    scale: 5;
    -webkit-transform: rotate(-45deg);
    transform: rotate(-45deg);
```

```
132   }
133
134   .left {
135     scale: 5;
136     -webkit-transform: rotate(135deg);
137     transform: rotate(135deg);
138   }
139
140   .BotonReproducir {
141     border: none;
142     border-radius: 40px;
143     font-size:2.5em;
144     cursor: pointer;
145   }
146   .BotonLateral {
147     padding-top: 10px;
148     padding-right: 10px;
149     padding-bottom:  10px;
150     padding-left: 20px;
151     background-color: #4CAF50;
152     color: rgb(202, 202, 202);
153     border: none;
154     border-radius: 10px;
155     cursor: pointer;
156     width: 80;
157     z-index: 1000;
158     font-size:2.5em;
159     text-align: center;
160     font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS',
          sans-serif;
161   }
162
163   .ArchivoExterno {
164     padding-top: 30px;
165     padding-right: 30px;
166     padding-bottom: 10px;
167     padding-left: 30px;
168     background-color: #4CAF50;
169     color: rgb(202, 202, 202);
170     border: none;
171     border-radius: 10px;
172     cursor: pointer;
173     width: 80;
174     z-index: 1000;
175     font-size:0.625em;
176     text-align: center;
177     font-family: 'Gill Sans', 'Gill Sans MT', Calibri, 'Trebuchet MS',
          sans-serif;
178   }
```

APÉNDICE B
Objetivos de desarrollo sostenible

Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. Fin de la pobreza.			X	
ODS 2. Hambre cero.				X
ODS 3. Salud y bienestar.				X
ODS 4. Educación de calidad.	X			
ODS 5. Igualdad de género.			X	
ODS 6. Agua limpia y saneamiento.				X
ODS 7. Energía asequible y no contaminante.				X
ODS 8. Trabajo decente y crecimiento económico.		X		
ODS 9. Industria, innovación e infraestructuras.		X		
ODS 10. Reducción de las desigualdades.				X
ODS 11. Ciudades y comunidades sostenibles.				X
ODS 12. Producción y consumo responsables.				X
ODS 13. Acción por el clima.				X
ODS 14. Vida submarina.				X
ODS 15. Vida de ecosistemas terrestres.				X
ODS 16. Paz, justicia e instituciones sólidas.			X	
ODS 17. Alianzas para lograr objetivos.				X

Reflexión sobre la relación del TFG/TFM con los ODS y con el/los ODS más relacionados.

- Fin de la pobreza, la educación es un factor fundamental en el desarrollo de sociedades más prosperas, al estar nuestro trabajo orientado a la formación autodidacta de niños, las sociedades futuras tendrán mayor formación y por tanto más oportunidades.
- Educación de calidad, considero que este proyecto promueve la educación de calidad al tratarse de un aprendizaje autodidacta sobre la materia. Permitiendo así que cada estudiante avance de acuerdo a sus capacidades.

- Igualdad de genero, el aprendizaje es indispensable para cada persona, la aplicación ha sido creada con una gama de colores neutros que no discrimine por genero.
- Trabajo decente y crecimiento económico, la relación que tiene este proyecto es de otorgar oportunidades, al tener un mayor conocimiento musical, es posible desarrollar una pasión que de lugar a más ofertas laborales.
- Industria, innovación e infraestructuras. Al ser un proyecto de aprendizaje mediante la web. El acceso a la información del mismo esta garantizado.
- Paz, justicia e instituciones sólidas. Al tener mayor conocimiento musical resulta en ciudadanos más formados que pueden compartir intereses comunes, fomentando la paz social.