_ TELECOM ESCUELA TÉCNICA VLC SUPERIOR DE UPV INGENIEROS DE TELECOMUNICACIÓN

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

# ENTORNO DE AUDIO USANDO LA NUEVA API DE HTML5

**Javier Latorre Playán**

**Tutor: Joaquín Cerdá Boluda**

**Cotutor: Germán Ramos Peinado**

Trabajo Fin de Grado presentado en la Escuela Técnica Superior de Ingenieros de Telecomunicación de la Universitat Politècnica de València, para la obtención del Título de Graduado en Ingeniería de Tecnologías y Servicios de Telecomunicación

Curso 2013-14

Valencia, 1 de julio de 2014

## Resumen

Este trabajo tiene como objetivo el diseño y programación de una aplicación de audio sobre la nueva API de audio de HTML 5. Para ello, utilizamos el programa SoundCool, que es propiedad de la Universidad Politécnica de Valencia y, a partir de los módulos que implementa, los adaptaremos al lenguaje antes mencionado, con el propósito de hacerlo más accesible y atractivo visualmente.

Para poder llevar a cabo lo mencionado anteriormente, se ha realizado, en primer lugar, un trabajo de investigación sobre los módulos propios de SoundCool y se ha visto si era posible su adaptación a HTML 5 y en segundo lugar se ha procedido a implementarlo.

El reto a abordar era aprender un nuevo lenguaje de programación (dos si tenemos en cuenta Javascript) y conseguir realizar con éxito una primera aproximación a la adaptación de SoundCool, cuya versión primitiva ya está recibiendo premios y consideraciones, y a la que se desea seguir contribuyendo.

## Resum

Aquest treball té el propòsit el disseny i programació d'una aplicació d'àudio en la nova API d'àudio de HTML5. Per això, utilitzarem el programa SoundCool, propietat de la Universitat Politècnica de València i, a partir dels mòduls que hi implementa, els adaptarem al llenguatge abans dit, per tal de fer-lo més accessible i atractiu visualment.

Per fer això, s'ha realitzat, en primer lloc, un treball d'investigació sobre els mòduls propis de SoundCool, i s'ha vist si era possible la seua adaptació a HTML 5. En segon lloc, s'ha començat la seua implementació.

El repte era aprendre un nou llenguatge (dos, si tenim en compte Javascript) i aconseguir realitzar amb èxit una primera aproximació a l'adaptació de SoundCool, ja que la versió inicial de la qual està aconseguint premis i consideracions.

## Abstract

The objective of this document was the design and development of an audio application in the new audio API of HTML 5. To achieve that objective, we used SoundCool, a program property of the UPV, and using the modules that this application implements, we tried to adapt them to the HTML language, so he could be more accessible.

In order to have success in this task, a research of the modules has been done by the developer, trying to know if the adaptation was even possible. In second place, the program has been implemented.

In this process, the author has achieved to learn a new development language (two, if we consider Javascript) and to adapt SoundCool to it.

# Índice

# Índice de figuras

# 1. Introducción

## 1.1 Objetivo del TFG

Decía el ilustre matemático Gottfried Leibniz que "*la música es el placer que experimenta la mente humana al contar sin darse cuenta de que está contando*", y es que la relación entre la ciencia y la música ha sido una constante en la historia de la humanidad.

Siendo consciente de ello, tanto el autor de este documento como su tutor y cotutor, decidieron realizar la adaptación de SoundCool (un programa propiedad de la UPV que tenía como objetivo el aprendizaje de música por parte de los niños de manera sencilla y divertida en clase) al lenguaje HTML 5.

Humildemente, hemos intentado mejorar SoundCool, haciéndola más accesible, más libre, más rápida y más atractiva visualmente. Obviamente, la API de audio web de HTML 5 es muy limitada, pero hay que tener en cuenta que se está comenzando, y que, al cabo de no demasiado tiempo, el desarrollo multimedia dentro de páginas Web será enorme.

Básicamente, lo que se ha intentado es desarrollar módulo por módulo SoundCool para HTML 5, usando todo lo necesario según la Web Audio API, accesible en Web.

## 1.2 Metodología y desarrollo del TFG

### *1.2.1 Gestión del TFG*

Para conseguir el objetivo planteado anteriormente, era necesaria una organización en tareas de lo que se iba a realizar, puesto que, al fin y al cabo, se pretendía realizar una adaptación por parte de un solo individuo de un programa diseñado por un equipo entero de investigadores de UPV.

Lo primero que se planteó fue qué requisitos había para realizar la mencionada adaptación. Por supuesto, era necesario aprender lenguajes no conocidos por el autor, tanto el propio HTML 5 como Javascript, puesto que la Web Audio API está expresada en su "estándar" en este último lenguaje.

Por otra parte, se hacía necesario el conocimiento interno de la aplicación a adaptar, y los problemas que podían plantear por su imposibilidad de programar los diferentes módulos que la formaban.

Por tanto, este TFG ha requerido un esfuerzo de investigación por parte del autor, lo cual le consumió bastante tiempo a la hora de la propia programación e implementación de la aplicación, que sería el último paso de nuestro proyecto.

Para ello, se contó con la colaboración de los dos tutores en todo momento, tanto en las dudas que generaba HTML 5 o Javascript como a la hora de comprender y entender las entrañas de SoundCool. La consulta a los expertos ha sido una constante en el desarrollo de este proyecto.

### *1.2.2 Distribución en tareas*

Dividir todo lo necesario en pequeñas tareas era un elemento clave en la realización del proyecto ("Divide y vencerás").

Así, en primer lugar, se intentó clasificar los deberes en 3 grandes grupos: aprendizaje, investigación e implementación.

### *1.2.3 Aprendizaje*

En la primera parte de nuestro proyecto, procedemos a aprender las bases de lenguajes de programación completamente nuevos. Para ello, se cuenta con la colaboración de los tutores, que facilitan al autor los recursos web necesarios para el proceso. Durante un trimestre, se procede a realizar cursos de Coursera sobre desarrollo web en HTML 5 y sobre JavaScript en Web, además de memorizar el estándar sobre desarrollo de la Web Audio API de HTML 5.

Además, para completar la formación y el aprendizaje, el autor consigue bibliografía, detallada al final del documento, con la que completará los conocimientos que, según el criterio de él mismo y de sus tutores, era necesario para el desarrollo del proyecto.

Las prisas y la rapidez son una constante en este apartado del proyecto, pudiendo considerar incluso que se han adquirido conocimientos en un tiempo récord. Obviamente, el autor no es un experto en ambos lenguajes de programación, pero podríamos decir que ya puede ponerlo en su currículo.

### *1.2.4 Investigación*

Después del aprendizaje de las bases de lo que se iba a utilizar para la programación de nuestra adaptación, se procede a investigar qué se debe programar.

Para ello, subdividiremos el proceso en pequeñas tareas: código de SoundCool, búsqueda de elementos similares, diseño de la aplicación y perfeccionamiento.

#### *1.2.4.1 Código de SoundCool*

Posiblemente, el hecho que más retrasa el desarrollo del proyecto, al depender de terceros.

Una vez se consigue que los programadores de SoundCool nos "presten" su código, se debe proceder a la investigación del mismo. Para esto, observamos cada uno de los módulos que se nos han presentado y se realiza un pequeño resumen a papel de lo que se considera necesario y de lo que parece difícil de realizar.

En el apartado de desarrollo se ampliará la información sobre cada módulo y su caracterización.

La principal dificultad de esta tarea radica en el desconocimiento del lenguaje en el que está programado SoundCool, algo que se soluciona mediante la adquisición de documentación descriptiva sobre los módulos del programa.

#### *1.2.4.2 Búsqueda de elementos similares*

Una vez se ha descubierto cómo funciona SoundCool, se comienza entonces a hacer un pequeño esquema sobre elementos similares a los que se utilizarán en nuestra adaptación. Básicamente, se realiza una búsqueda sobre programas ya realizados en la web audio API y se revisa la documentación correspondiente a la misma. De esta manera, se consigue un pequeño esbozo de la programación a realizar y que funciones se han de utilizar.

#### *1.2.4.3 Diseño de la aplicación*

Cuando se tiene el esbozo de los módulos encontrados que son similares a los de SoundCool, se procede a realizar una subdivisión más, esta vez sobre el orden que se debe seguir y los módulos a adaptar en los que más dudas se tienen sobre su correcto diseño. Se realizan diversos dibujos sobre el aspecto externo que debe tener la aplicación y cómo va a ser exactamente la función a utilizar y cómo deben funcionar los correspondientes listener de los que constará el código JavaScript.

Otro problema a tener en cuenta es cómo se realiza la comunicación entre JavaScript y HTML 5, pero, por suerte, tras el apartado de aprendizaje esto resulta más fácil de lo que parecía.

Una vez se han realizado los dos primeros apartados, este último parece más sencillo, pues solamente consta de unir lo que ya se tiene por separado, pero no por ello es prescindible.

#### *1.2.4.4 Perfeccionamiento*

Cuando se tiene el esquema definitivo, que luego estará descrito en su apartado correspondiente, en las figuras necesarias, hay que proceder a la parte del aspecto externo. Los CSS son, posiblemente, la parte que más ha costado al autor de este trabajo, ya que en la formación necesaria, paso esto un poco por alto. Fue el tutor el que le hizo caer en la cuenta de la importancia de este apartado.

Así, se emplea una cantidad de tiempo importante en intentar matizar ciertos aspectos de los módulos que son realmente claves a la hora de dar un aspecto externo atractivo y visualmente cómodo.

### *1.2.5 Implementación*

Como parte final, se empezó a hacer lo que realmente era el objetivo del trabajo, la implementación, la programación, la adaptación.

Tomando como base los programas investigados anteriormente, se empezó a programar sin tener en cuenta que el aspecto externo fuera el adecuado, implementando sobre lo ya implementado, para ver si lo que se había pensado sobre el papel funcionaba correctamente.

Esto dio pie a muchos errores después, a la hora de unirlo todo, sobre todo en torno a pequeños matices difíciles de detectar.

Finalmente, se llega al boceto definitivo.

### *1.2.6 Diagrama temporal*

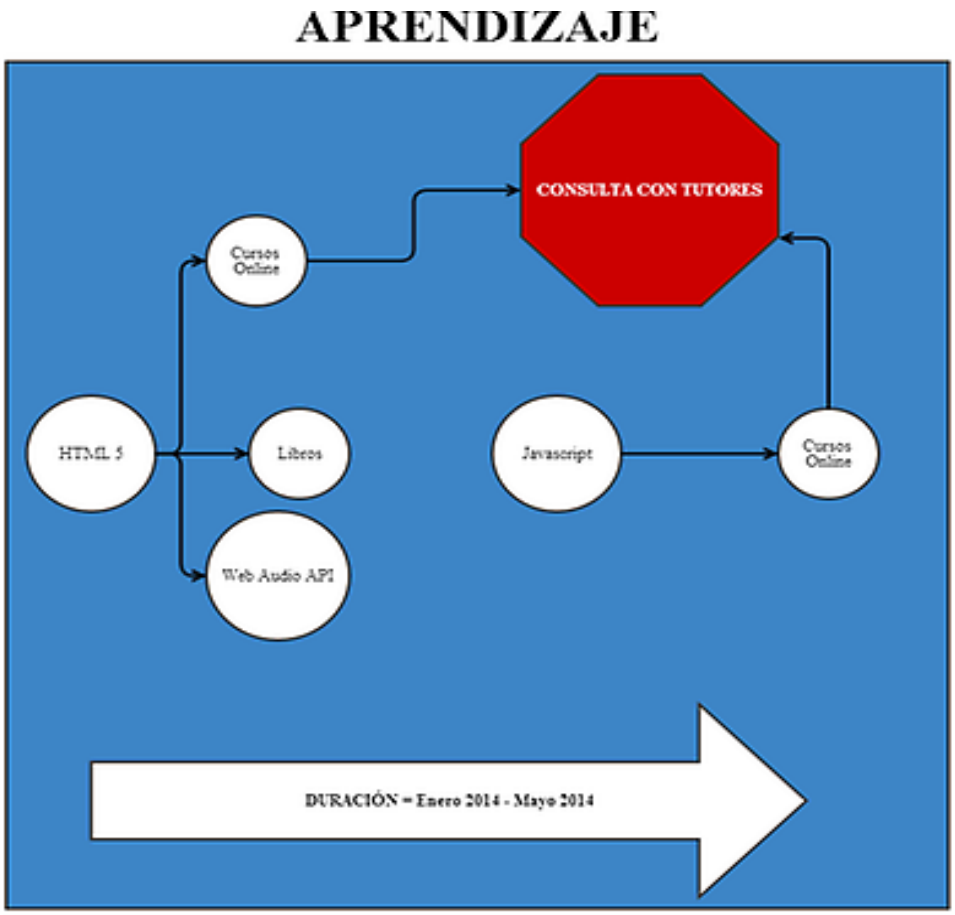



**Figura 1. Diagrama temporal del aprendizaje.**

# INVESTIGACIÓN

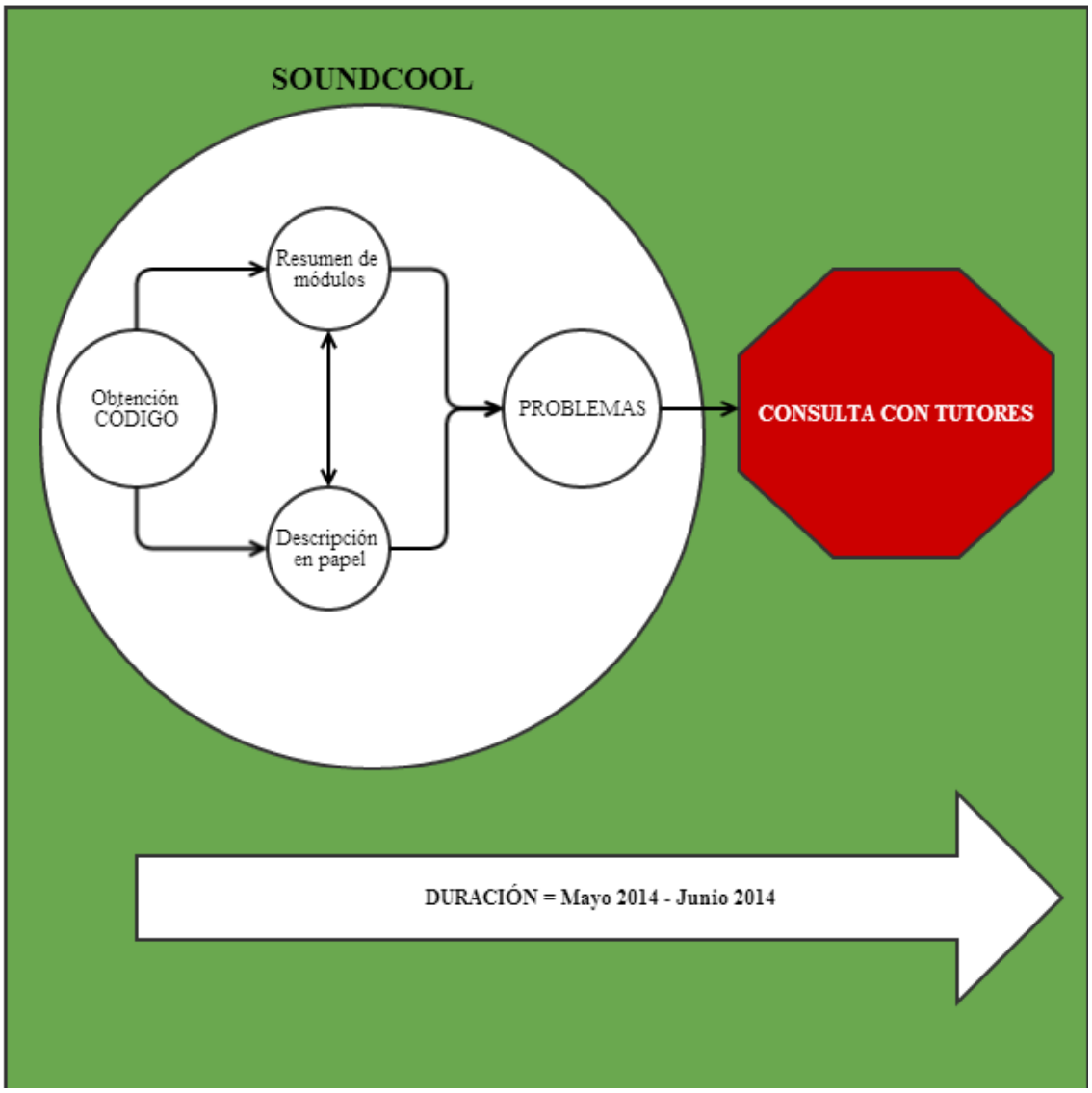



**Figura 2. Diagrama temporal de la investigación de SoundCool.**

# INVESTIGACIÓN

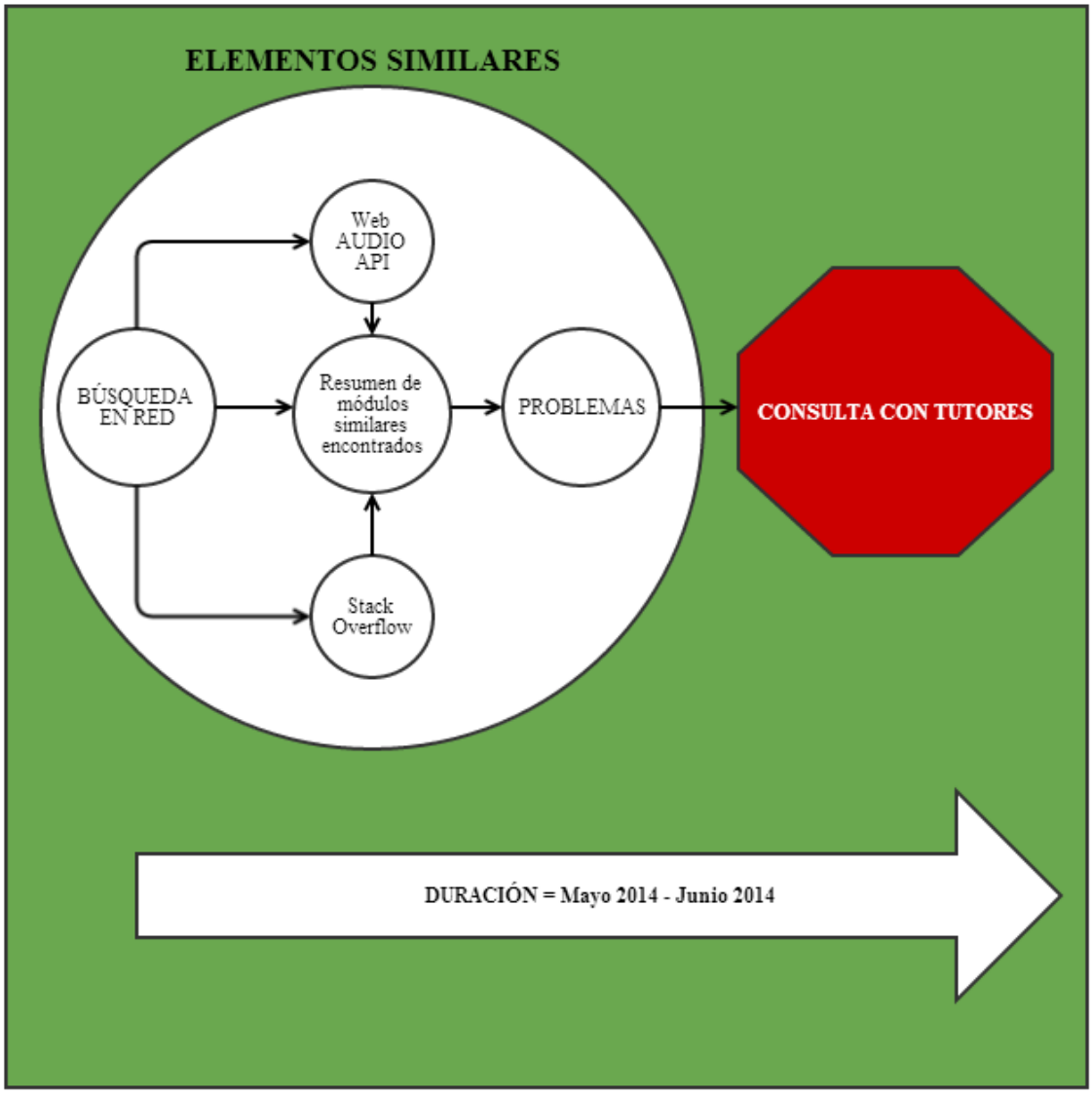



**Figura 3. Diagrama temporal de la investigación de módulos similares a los necesarios.**

# INVESTIGACIÓN

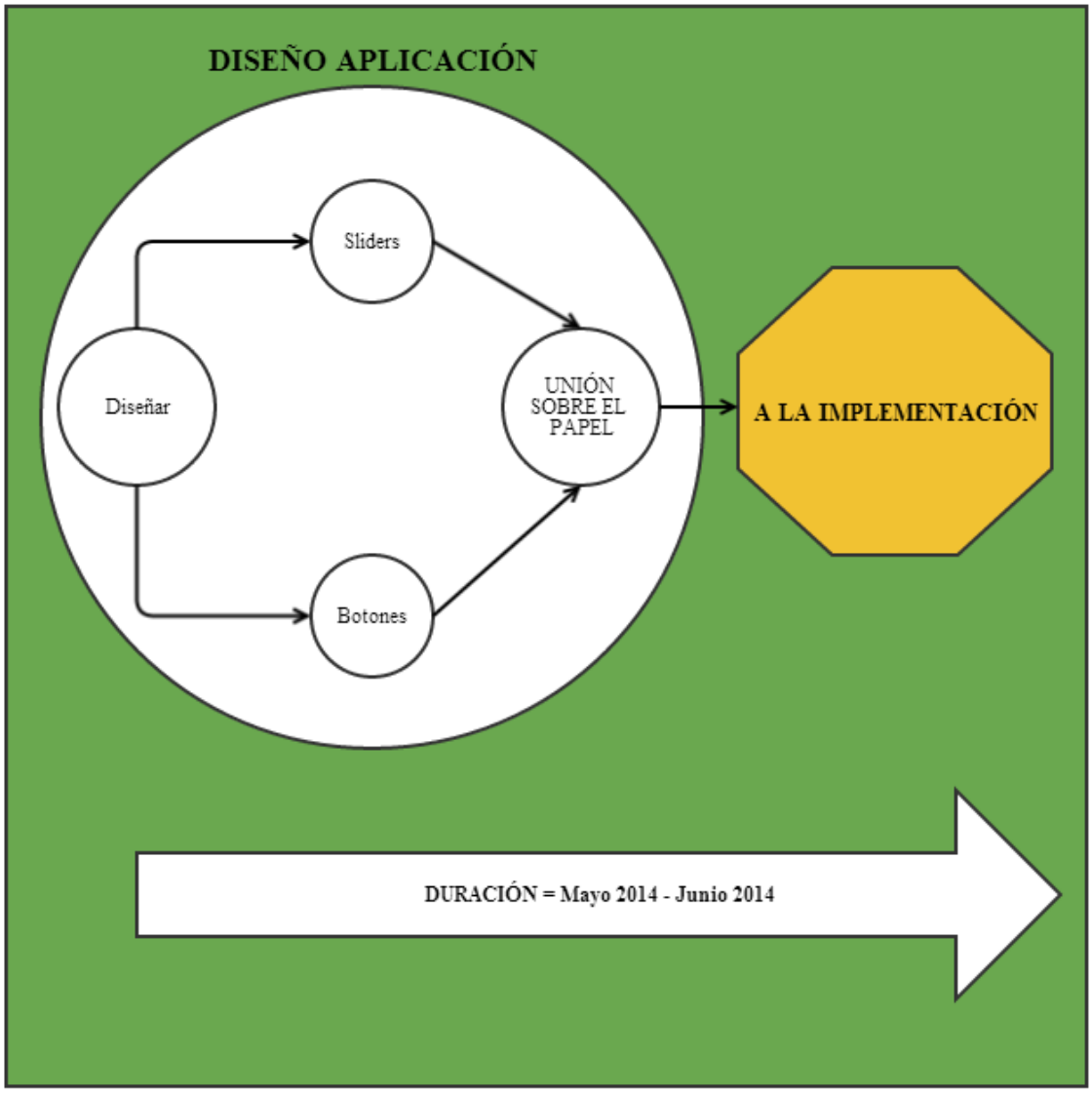



**Figura 4. Diagrama temporal de la investigación del diseño de la aplicación.**

# INVESTIGACIÓN

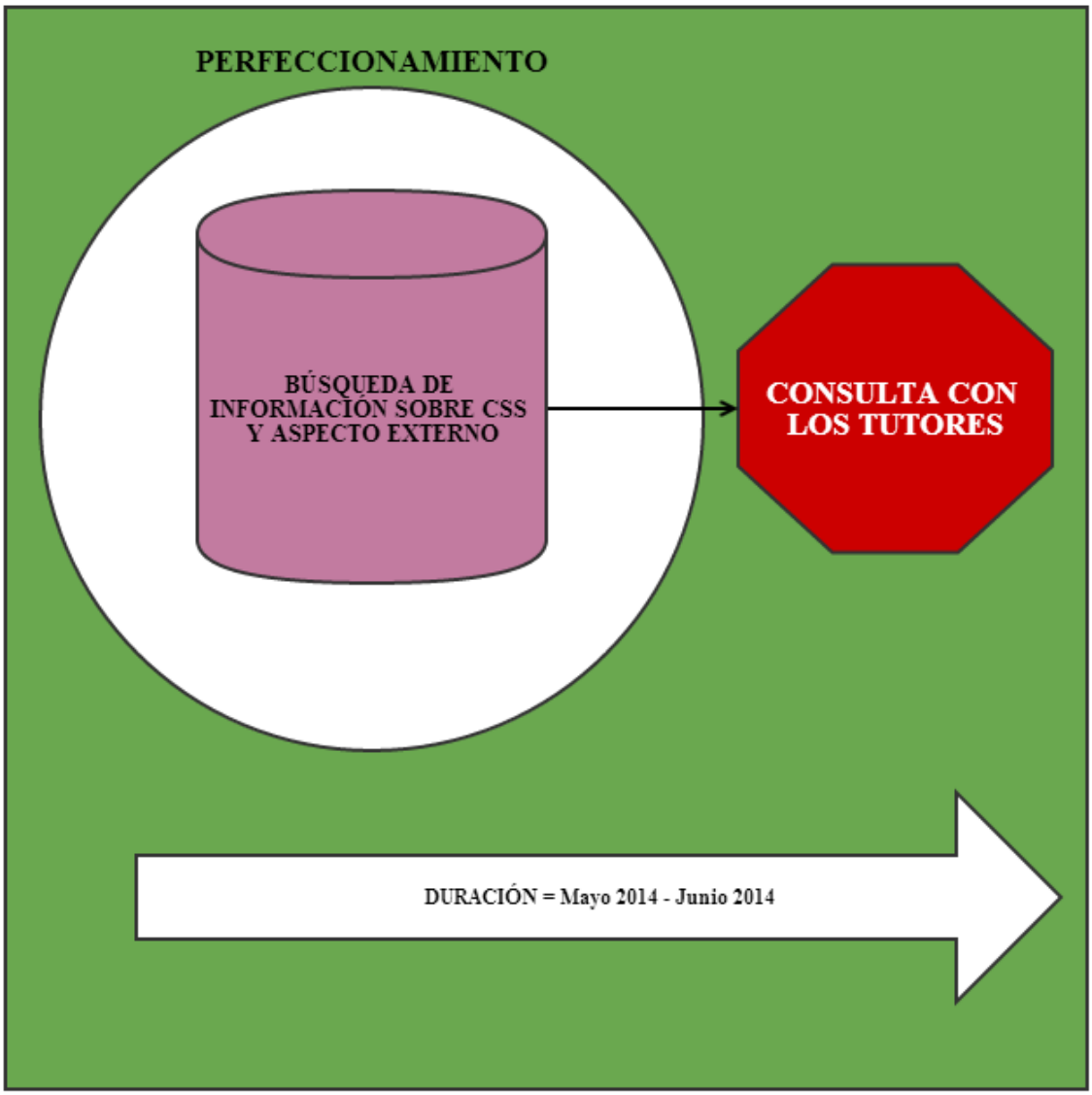



**Figura 5. Diagrama temporal de la investigación del perfeccionamiento.**

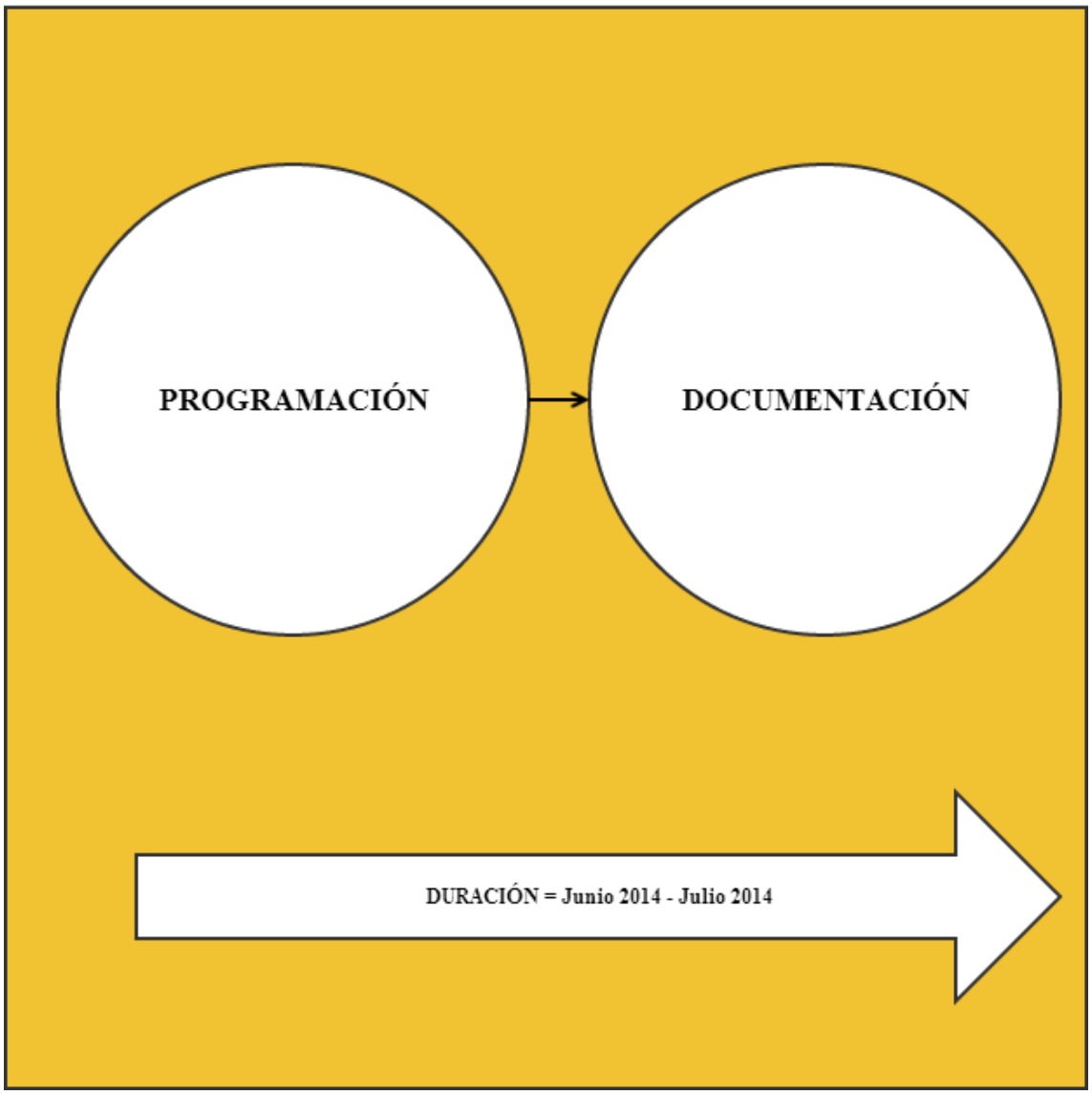



**Figura 6. Diagrama temporal de la implementación de la aplicación.**

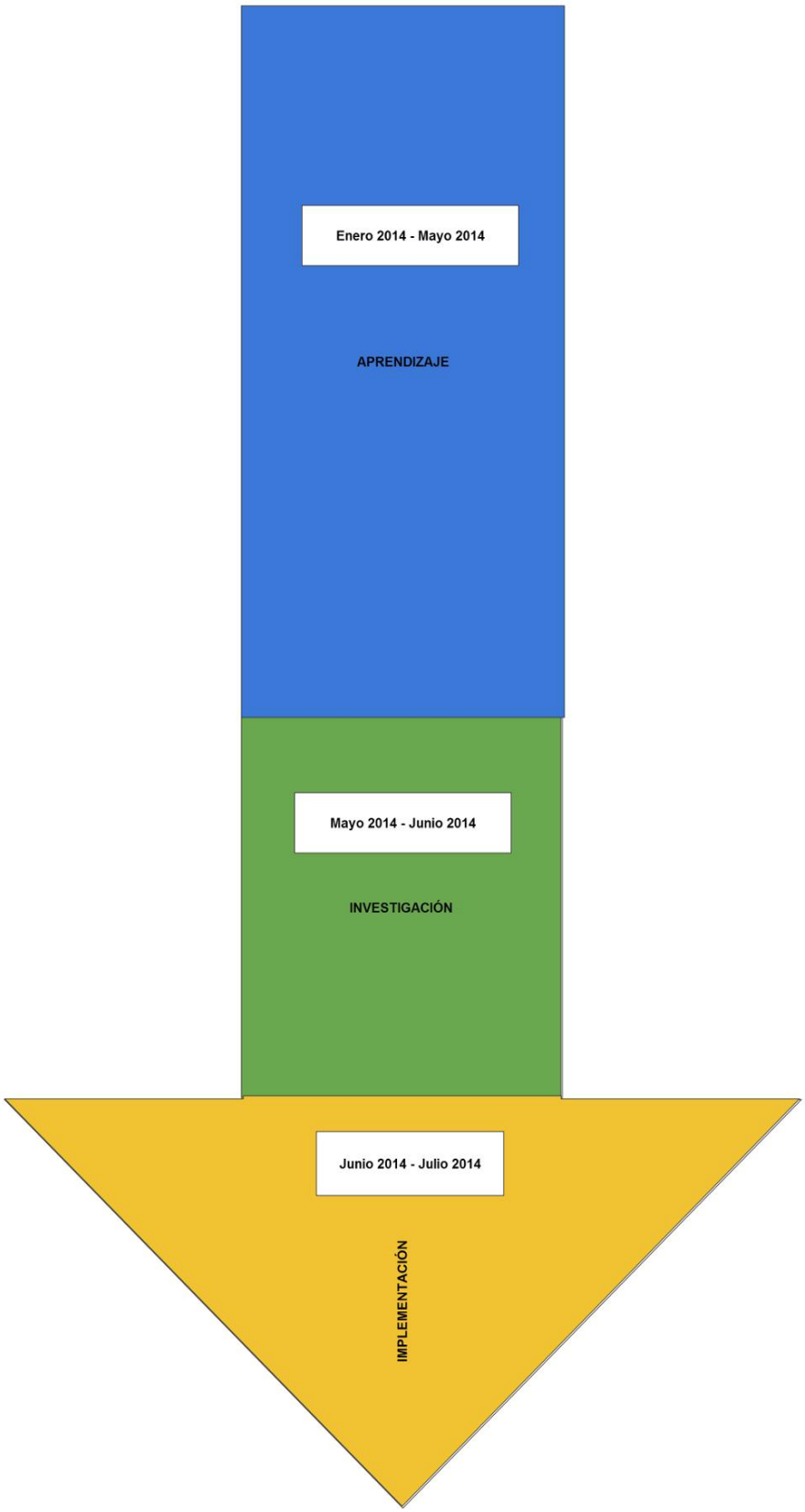



**Figura 7. Diagrama temporal. Resumen.**

## 1.3 Esquema

En el presente apartado, se resumirá de una manera esquemática, con el objetivo de hacer más fácil la lectura del trabajo, el contenido del proyecto a desarrollar.

En el capítulo 2, abordaremos el contexto en el que se ha realizado este TFG, así como la importancia de los nuevos lenguajes de programación web y su capacidad de mejora y evolución debido a su filosofía de código abierto. Para saber por qué hemos realizado el trabajo en toda su integridad debe conocerse sobre qué se ha realizado y con qué acompañantes de baile.

En el capítulo 3, nos adentraremos en las entrañas de la presente adaptación, describiendo en profundidad cada uno de los módulos programados. Describiremos el esqueleto de nuestro organismo y cómo sus células se conectan entre ellas para crear nuestros objetivos, para dar forma nuestras ideas.

En el capítulo 4, se mostrarán los resultados obtenidos, acompañados de una pequeña demostración gráfica.

En el capítulo 5, se reflexionará acerca de la consecución de los objetivos planteados, a modo de conclusión, además de abordar las líneas futuras que debe seguir nuestro trabajo, si es que se desea que evolucione.

# 2. Contexto del TFG

A continuación, describiremos el contexto tecnológico en el que ha tenido que desarrollarse nuestra aplicación, sobre todo a nivel de lenguajes utilizados y de programas inspiradores.

En primer lugar, describiremos HTML 5 y su aplicación a elementos multimedia en páginas web.

En el segundo apartado, abordaremos cómo Javascript ha influido en el desarrollo de las aplicaciones web y en qué sentido lo ha hecho.

En el tercer apartado, tratamos el contenido de nuestra aplicación "padre", SoundCool, la motivación que se tuvo para crearla y cómo ha influido en nuestra capacidad de diseño del proyecto.

En el último apartado, se hablará del lienzo en el que pintamos nuestra obra, las herramientas que nos ayudan a construir nuestro proyecto, la Web Audio Api.

## 2.1. HTML 5

Los avances en la programación de páginas web han sido extremadamente grandes en los últimos años, y sería estúpido por parte de desarrolladores e ingenieros no percatarse de ello.

En 2010 HTML 5 obtuvo mucha presencia en los medios al declarar Apple que renunciaba a dar soporte a la tecnología Flash, hasta entonces universalmente usada para multimedia y sobre todo vídeo. El motivo alegado es que se trata de un estándar abierto, que consume menores recursos y ofrece las mismas prestaciones. En cualquier caso HTML 5 es un estándar universalmente aceptado por la industria. [AlvGarAl2012]

A diferencia de las anteriores actualizaciones HTML, las especificaciones de HTML 5 tienen una vista más amplia de lo que se necesita para mejorar el desarrollo y la programación de diseños Web durante los próximos 10 años y más. [DavMath2010]

El principal criterio de diseño de HTML 5 ha sido el de resolver problemas prácticos, lo que hace que se hayan adoprado soluciones orientadas a facilitar el trabajo en situaciones reales.[AlvGarAl2012]

El audio y el vídeo son una gran novedad. Sitios como Youtube, Pandora y Last FM atraen millones de usuarios cada semana con sus masivas bibliotecas multimedia. Cada vez más usuarios se conectan a Internet para ver contenidos multimedia. [DavMath2010]

El nuevo estándar se ha enriquecido con medios para simplificar el trabajo con las nuevas herramientas de gestión de contenidos y facilitar la inclusión de elementos multimedia[AlvGarAl2012]

Hay que aprovecharse de las mejoras, por lo que decidimos realizar nuestro trabajo sobre este estándar, que conseguiría tener una visualización óptima e intuitiva, además de un tratamiento adecuado del audio, que era al fin y al cabo el objetivo de esta adaptación.

Con HTML 5 será posible construir páginas de una cierta complejidad, y se abrirá la puerta para profundizar en formas más sofisticadas de construir contenidos para Internet.

## 2.2. Javascript

Originalmente, HTML servía para construir páginas Web estáticas, es decir, documentos con texto, imágenes y enlaces, pero cuyo contenido no variaba y se almacenaba en ficheros. Sin embargo, hoy en día, en la mayoría de las Web, los contenidos se generan combinando plantillas e información contenida en las bases de datos, no a partir de ficheros estáticos. [AlvGarAl2012]

La tecnología JavaScript no es nueva. Sus comienzos se remontan a 1993, cuando Netscape Communications incluyó en su navegador Web una tecnología de scripting llamada LiveScript. Desde entonces, se tomó la costumbre de incorporar un sencillo lenguaje de programación en las páginas, para dar interactividad a la Web.[DavMath2010]

Además, es posible introducir una respuesta interactiva dentro de una página sin necesidad de enviar una petición contra un servidor remoto. De esta forma, es posible incluir elementos propios de programas como menús, cálculos automáticos, editores de texto, herramientas de diseño, o controles de todo tipo. Esto se consigue insertando elementos especiales y código en un lenguaje de script que interpreta el navegador, Javascript. [AlvGarAl2012]

Javascript es un lenguaje de programación de scripts que se combina fácilmente con documentos HTML, ofreciendo dinamismo a las páginas. Es un lenguaje interpretado y basta un sencillo editor de texto para insertar los programas en HTML y dar vida a nuestras páginas: ventanas de mensajes, elementos del documento que cambian de aspecto, validación de formularios o animaciones son sólo algunos ejemplos de lo que se puede conseguir con la ayuda de este lenguaje. [AlvGarAl2012]

Su sintaxis básica es intencionadamente similar a Java para reducir tiempo de aprendizaje a programadores de estos lenguajes, pero eso es todo. El código fuente Javascript se incluye directamente en las páginas HTML, no como un objeto diferenciado que se carga de forma independiente. El propio navegador es el intérprete y no hay máquinas virtuales externas para ello. La facilidad de uso y la inmediatez de respuesta es una baza a favor de su uso. [AlvGarAl2012]

La nueva versión de Javascript permite construir aplicaciones de tipo escritorio que se ejecutan en el interior de los navegadores Web. [DavMath2010]

## 2.3. SoundCool

SoundCool es una aplicación cuyo objetivo es el procesamiento de audio en tiempo real, accesible tanto desde Android como desde la Kinect, con un propósito docente. Si los alumnos pueden experimentar como cambia la música ante sus ojos (u oídos), su aprendizaje será mucho más sencillo y divertido.

Con esta idea en la cabeza, SoundCool puede servir también para otros propósitos, como la grabación en línea por parte de grupos de música, o la composición de canciones entre personas que se encuentran muy separadas geográficamente.

Si se desea obtener más información sobre el funcionamiento de SoundCool, se puede consultar en el Anexo I, al final del presente documento.

¿Qué mejor manera de mejorarlo que intentar adaptarlo a un lenguaje que lo permita introducirse en la Web? Si se consigue que llegue a todas partes, su éxito está garantizado.

## 2.4. Web Audio API

La introducción del elemento <AUDIO> en HTML 5 es muy importante, ya que permite reproducción básica de audio en streaming. Sin embargo, no es lo suficientemente potente para manejar aplicaciones de audio más complejas. Para videojuegos basados en web muy sofisticados o para aplicaciones interactivas se requiere de otra solución.

La especificación de la Web Audio API describe una API Javascript de alto nivel, que sirve para procesar y sintetizar audio en aplicaciones web. El paradigma de lo que se desea conseguir es un gráfico de enrutamiento de audio, donde una serie de AudioNodes se conectan para definir y procesar en conjunto el audio.

La API está diseñada para ser usada en conjunción con otras APIs y elementos en la plataforma web, especialmente XMLHttpRequest. Además, si desea programar con ella juegos o aplicaciones interactivas, en la API se presupone el uso de un canvas 2D o de gráficos basados en la API de gráficos 3D de WebGL.

Este documento se publicó por el Audio Working Group como un borrador de trabajo, con el objetivo de ser una recomendación WC3. Es por tanto un esbozo y puede ser actualizado, sustituido o quedarse obsoleto a causa de la aparición de otros documentos.

# 3. Desarrollo del trabajo.

A continuación, procederemos a explicar con mayor detalle el desarrollo de la programación de la adaptación a HTML 5 del programa SoundCool, siguiendo el esquema temporal indicado anteriormente.
En primer lugar, procederemos a describir cada uno de los módulos implementados en la adaptación, que se pueden comparar en el documento anexo con los módulos que implementa el SoundCool primitivo, el original.
En el segundo apartado, explicaremos de manera general cómo está implementada la función principal de nuestra adaptación.
Por último, se podrá observar un diagrama de bloques del programa.

## 3.1. Descripción de los módulos

### *3.1.1. Output*

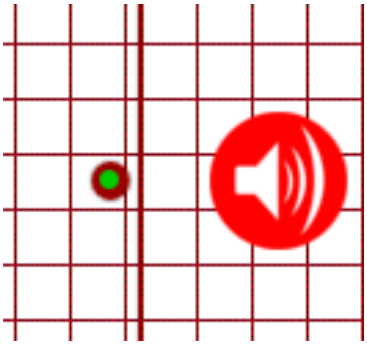


**Figura 8. Aspecto del Output.**

Es la salida de audio del programa. Conecta directamente el contenido de sonido con la tarjeta de audio de nuestro PC. Es la adaptación del módulo MAudio.
No se ha conseguido implementar el acceso a la tarjeta de audio, puesto que el contacto con periféricos desde JavaScript/HTML 5 es un reto que escapa al objetivo o a los límites de este TFG.

```
<div id="output" class="destination">
    <div class="node node-input"><span class="node-button"> </span></div>
</div> <!-- /#output -->
```

**Figura 9. Código HTML del Output.**

Realizamos una división lógica <DIV> de la pantalla, y asociaremos esta división a la salida, asignándole solamente un nodo de entrada.

### *3.1.2. Delay*

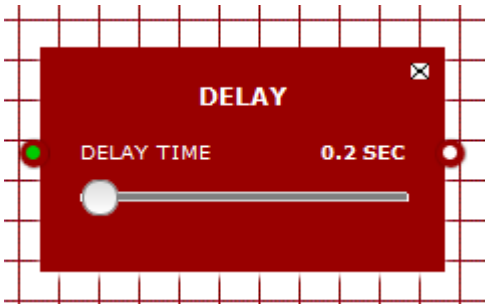



**Figura 10. Aspecto del módulo Delay en SoundCool.**

Implementa un retardo del sonido, lo que se puede utilizar para dar un efecto de eco, o la sensación de estar en una habitación, es muy común utilizarlo en la edición de audio y es uno de los módulos más sencillos de implementar, puesto que aparece como tal descrito en la Web Audio API.
El único matiz que podríamos destacar es la presencia de un feedback en el módulo de SoundCool, que no es posible implementar dentro de la API de HTML 5, puesto que tendríamos que adentrarnos en las entrañas de un código protegido por licencia, y tampoco parece un elemento excesivamente clave en el desarrollo de esta adaptación.
Implementa un slider que permite ampliar o reducir los segundos de Delay que queremos implementar al módulo.

```
function createDelay() {
    var module = createNewModule( "delay", true, true );
    addModuleSlider( module, "delay time", 0.2, 0.0, 10.0, 0.01, "sec", onUpdateDelay );

    // after adding sliders, walk up to the module to store the audioNode.
    module = module.parentNode;

    var delayNode = audioContext.createDelay();
    delayNode.delayTime.value = 0.2;
    module.audioNode = delayNode;

    if (this.event)
        this.event.preventDefault();
}
```

**Figura 11. Código Javascript del módulo Delay.**

```
function onUpdateDelay(event, ui) {
    updateSlider(event).audioNode.delayTime.value = event.target.value;
}
```

**Figura 12. Código Javascript del módulo Delay, cuando se actualiza el slider.**

Cuando actualizamos el valor del slider, se asigna un valor distinto de tiempo de Delay al módulo.

### *3.1.3. Live Input*

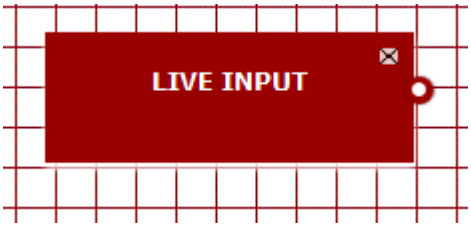



**Figura 13. Aspecto del Live Input.**

Manda a la salida lo capturado por el micrófono. Es la adaptación de Mrecord, que permitían a SoundCool hacer esto. Ha sido imposible cambiar la entrada de grabación, puesto que es algo intrincado en JavaScript y en ningún lugar aparece directamente.

```
function createLiveInput() {
    var module = createNewModule( "live input", false, true );

    // after adding sliders, walk up to the module to store the audioNode.
    module = module.parentNode;
    var err = function(e) {
        alert('Error getting audio');
        console.log(e);
    };

    if (navigator.getUserMedia )
       navigator.getUserMedia({audio:true}, gotStream.bind(module), err );
      else if (navigator.webkitGetUserMedia )
       navigator.webkitGetUserMedia({audio:true}, gotStream.bind(module), err );
      else if (navigator.mozGetUserMedia )
       navigator.mozGetUserMedia({audio:true}, gotStream.bind(module), err );
      else
       return(alert("Error: getUserMedia not supported!"));

    if (this.event)
        this.event.preventDefault();
}
```

**Figura 14. Código Javascript de LiveInput.**

Se utiliza el recurso getUserMedia, que es lo establecido en la Web Audio API para capturar la entrada de micrófono.

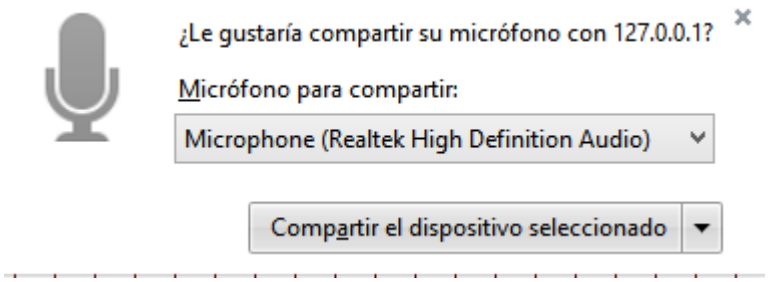



**Figura 15. Código Javascript de LiveInput.**

Cuando se activa este módulo, el propio explorador nos requiere permisos para utilizar nuestra entrada de audio.

### *3.1.4. Panning*

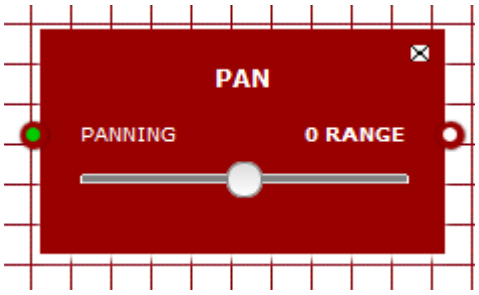



**Figura 16. Aspecto del módulo Panning.**

Implementa un cambio en el panning del audio de salida (es decir, si sale más por el altavoz correspondiente a la izquierda o a la derecha). El slider horizontal se encarga de modificar el parámetro requerido.
Adapta el módulo Mpan de SoundCool, y ha requerido un poco más de profundidad en la investigación dentro de Web Audio API de HTML 5, a pesar de que su aplicación es relativamente sencilla.

```
function createPanner() {
    var module = createNewModule( "pan", true, true );
    addModuleSlider( module, "panning", 0, -45, 45, 1, "range", onUpdatePanning );

    // after adding sliders, walk up to the module to store the audioNode.
    module = module.parentNode;

    var pannerNode = audioContext.createPanner();
    pannerNode.panningModel = "equalpower";
    pannerNode.setPosition = (0,0,0);
    module.audioNode = pannerNode;

    if (this.event)
        this.event.preventDefault();
}
```

**Figura 17. Código Javascript del módulo Panning.**

```
function onUpdatePanning(event, ui) {
    var xpos = Math.sin(event.target.value * (Math.PI / 180));
    updateSlider(event).audioNode.setPosition = (xpos,0,0);
}
```

**Figura 18. Código Javascript del módulo Panning.**

Se cambia la posición al respecto de un observador imaginario situado en (0,0,0), hemos implementado un desplazamiento en horizontal.

### *3.1.5. Direct Input*

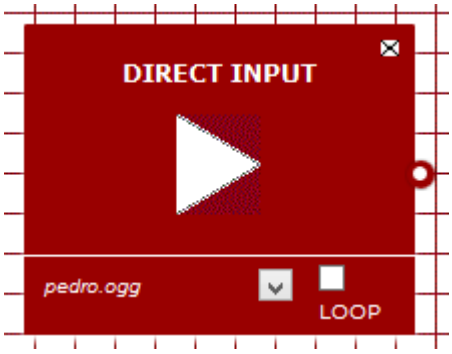



**Figura 19. Aspecto del DirectInput.**

Es la adaptación del módulo MdirectInput, que permitía mandar a la salida un archivo tal cual estaba a la entrada. En este caso, elegiremos de una lista predeterminada ciertos archivos de música que podrán reproducirse. Permite reproducirse en Loop.

```
function createAudioBufferSource( buffer ) {
    var module = createNewModule( "direct input", false, true );

    var play = document.createElement("img");
    play.src = "img/ico-play.gif";
    play.style.marginTop = "10px";
    play.alt = "play";
    play.onclick = onPlayABSource;
    module.appendChild( play );

    module = module.parentNode;
    module.className += " has-footer has-loop";

    // Add footer element
    var footer = document.createElement("footer");

    var looper = document.createElement("div");
    looper.className = "loop";
    var label = document.createElement("label");
    var check = document.createElement("input");
    check.type = "checkbox";
    check.onchange = onToggleLoop;
    label.appendChild(check);
    label.appendChild(document.createTextNode(" Loop"));
    looper.appendChild(label);
    footer.appendChild(looper);

    var sel = document.createElement("select");
    sel.className = "ab-source";
    var opt = document.createElement("option");
```

d

**Figura 20. Código Javascript de Direct Input.**

### *3.1.6. Gain*

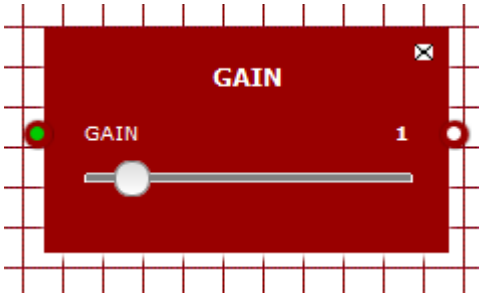



**Figura 21. Aspecto del módulo Gain.**

Implementa básicamente un cambio de volumen. En SoundCool, no aparece este módulo en ningún sitio, sino que aparece implícito dentro de los módulos correspondientes. Sin embargo, debido al carácter modular que queremos darle a nuestra adaptación, nos ha parecido más adecuado y más sencillo a nivel de código según la Web Audio API hacerlo de la siguiente manera.

```
function createGain() {
    var module = createNewModule( "gain", true, true );
    addModuleSlider( module, "gain", 1.0, 0.0, 10.0, 0.01, "", onUpdateGain );

    // after adding sliders, walk up to the module to store the audioNode.
    module = module.parentNode;

    var gainNode = audioContext.createGain();
    gainNode.gain.value = 1.0;
    module.audioNode = gainNode;

    if (this.event)
        this.event.preventDefault();
}
```

**Figura 22. Código Javascript del módulo Gain.**

```
function onUpdateGain(event) {
    updateSlider(event).audioNode.gain.value = event.target.value;
}
```

**Figura 23. Código Javascript del módulo Gain.**

## 3.2. El programa principal

A continuación, se procede a describir cómo se ha realizado el programa principal, es decir, la pantalla de inicio que permite clasificar los módulos y mostrarlos de una manera atractiva.

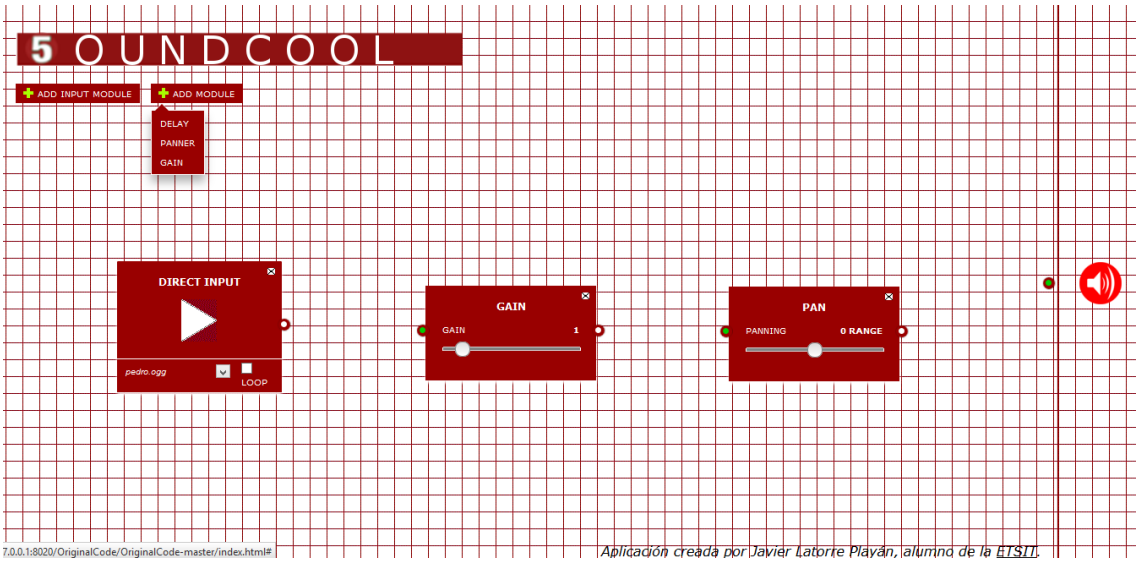



**Figura 24. Imagen de los respectivos módulos del programa y de la lista desplegable de menús.**

Se tienen menús que se despliegan en la parte superior, donde se muestran los módulos. Debajo de ellos, podemos observar el canvas donde se introducirán los módulos. Pulsando en su parte superior podemos arrastrarlos, y pulsando en la x de la parte superior derecha podemos cerrarlos.

El módulo Output aparece por defecto a la derecha de la pantalla, puesto que se sobreentiende que vamos a querer escuchar lo que diablos estemos haciendo. Sin embargo, hasta que no unamos a su entrada cualquier archivo de audio, no podremos implementar nada.

Para conectar los módulos hay que unir los botones de los que consta la entrada y la salida, en el caso de que tuvieran ambas.

Cuando se pide el acceso a un periférico, cada explorador actuará como propiamente decida.

```
<div id="main" class="container">
    <header id="header">
        <img src="img/logo.png" alt="5oundCool: A HTML 5 adaptation (v1.0)" />
    </header>
    <nav id="nav">
        <ul id="controls" class="container">
            <li>
                <a href="#">Add Input Module</a>
                <div class="sub-menu">
                    <ul>
                        <li><a href="#" id="cabs">Direct Input</a></li>
                        <li><a href="#" id="cliv">Live Input / Record</a></li>
                    </ul>
                </div>
            </li>
            <li>
                <a href="#">Add Module</a>
                <div class="sub-menu">
                    <ul>

                        <li><a href="#" id="cdel">Delay</a></li>
                        <li><a href="#" id="cpan">Panner</a></li>
                        <li><a href="#" id="cgai">Gain</a></li>

                    </ul>
                </div>
            </li>
        </ul>
```

**Figura 25. Código HTML de la organización del menú.**

# 4. Resultados obtenidos

En el presente apartado, se mostrarán una serie de capturas de pantalla sobre el aspecto que tienen algunas de las posibilidades que proporciona nuestra adaptación, a modo de demostración.

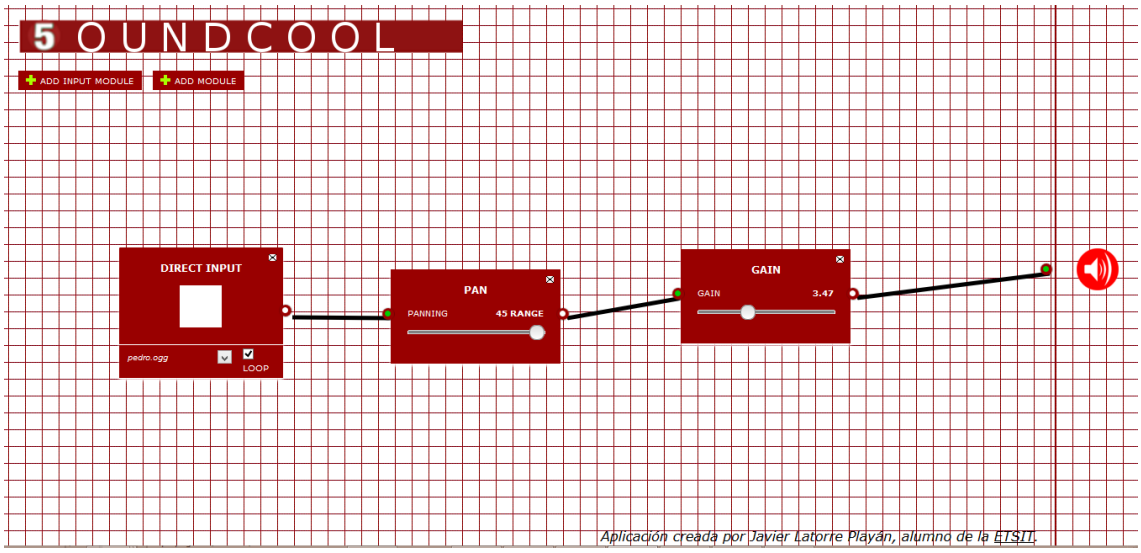



**Figura 26. Ejemplo de interconexión de módulos.**

En este caso, el archivo pedro.ogg, estaría reproduciéndose en bucle, saliendo todo por el altavoz derecho y aumentándose su volumen en, aproximadamente, x3’5.

La combinación de los diferentes módulos descritos anteriormente da pie a todas las combinaciones imaginables.

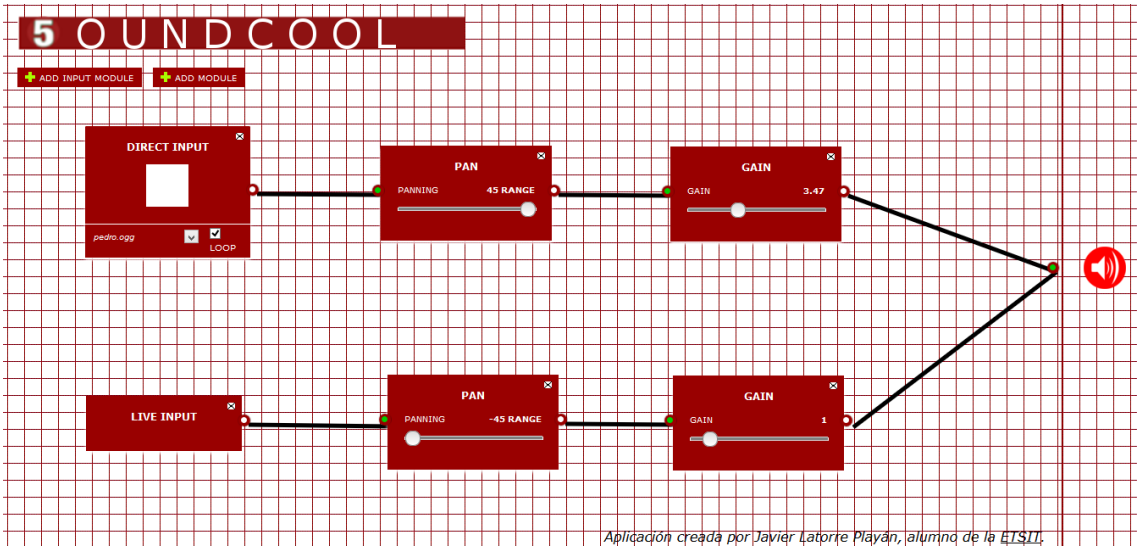



**Figura 27. Ejemplo de interconexión entre varios módulos.**

En la figura 27, podemos observar otro ejemplo de interconexión. En este caso, se escucharía por el altavoz izquierdo lo que fuéramos capturando por nuestro micro y, en la parte derecha, se seguiría escuchando en bucle pedro.ogg.

# 5. Conclusiones y propuesta de trabajo futuro

En el presente apartado se tratarán los aspectos más reflexivos del TFG.

En primer lugar, se hablará de hasta qué punto hemos conseguido el objetivo planteado, a modo de conclusiones.

En segundo lugar, explicaremos cuál es el camino que, creemos, debe seguir SoundCool si desea seguir evolucionando en el mismo sentido que estamos apuntando, las líneas futuras.

## 5.1. Conclusiones

Como el lector habrá podido comprobar, solamente revisando la documentación facilitada de SoundCool, el autor de este TFG no ha conseguido la adaptación perfecta del programa, y ha intentado imitar lo máximo posible a los programadores de la aplicación primitiva, pese a que se ha tomado algunas licencias uniendo módulos y descartando algunas opciones. Sin embargo, podemos concluir que se ha dado un primer paso, se ha puesto la primera piedra en el camino de la adaptación de SoundCool a HTML5 usando la Web Audio API.

El autor, aconsejado por sus tutores, ha decidido ponerse unos límites y realizar solamente una pequeña introducción a lo que se podría conseguir en el aspecto de la evolución de SoundCool, debido fundamentalmente a la falta de tiempo y a la complejidad de algunas de las características del programa original, que, si bien se han investigado en profundidad (como se podrá leer a continuación, en el apartado de propuestas de trabajo futuro), no ha sido posible llevarlas a cabo por pequeños matices en el desarrollo de la aplicación.

Evidentemente, no puede considerarse un éxito rotundo, pero este humilde autor se halla tremendamente orgulloso del esfuerzo realizado, y cree que, para las limitaciones y conocimientos que tenía al comenzar a desarrollarse este proyecto, ha conseguido un gran paso de cara a su evolución como profesional en este campo que, por qué no, también abarca la Ingeniería de Telecomunicaciones, en torno a la cual pretende construir su vida.

El autor también considera que, a pesar de no haber conseguido la funcionalidad en algunos de los módulos que se pretendían programar, si ha reunido el conocimiento teórico necesario para realizarlo en un futuro no demasiado lejano sin demasiados problemas.

Se habrá observado que no se ha implementado el módulo Mixer, esto es porque, dado el carácter modular que tiene la Web Audio API, parecía inútil crear algo que se puede crear con los elementos programados, uniendo módulos de tipo Input y módulos Gain.

Finalmente, se expone en esta conclusión el convencimiento de que la Web Audio API de HTML 5 va a permitir un desarrollo en el mundo del Audio Web desconocido hasta

la fecha. La constante evolución de estos aspectos nos llevará a una Web cada vez más atractiva visualmente y con más complementos, donde el límite estará en la imaginación de los desarrolladores. Seguimos en el túnel y no hay atisbo de final.

## 5.2. Propuesta de trabajo futuro

Por supuesto, el camino a donde debe dirigirse todo aquel que desee completar este trabajo es la completa adaptación de SoundCool a HTML 5.
En primer lugar, faltan por añadir los módulos Mkeyboard y MVST. En cuanto al primero, el autor ha investigado acerca de una manera de conseguirlo, utilizando el código de MidiBridge, una aplicación Java con el código abierto para todo aquel que desee utilizarlo, y que el lector puede consultar en [MidiB00] de la bibliografía.
Utilizando la función de SendMidiEvent, o el ejemplo que la propia aplicación expone para implementar un teclado virtual, ya realizado, no debería suponer demasiado trabajo de cara a una programación futura. La falta de tiempo y ciertos matices en la implementación en HTML 5 es lo que ha impedido al autor la programación de este módulo, lo que supone una de las grandes frustraciones al respecto de este proyecto, en su humilde opinión.

En cuanto al segundo módulo, MVST, la implementación parece bastante más complicada. En la investigación llevada a cabo por el autor del TFG se han encontrado diversos plug-ins que tratan la introducción de instrumentos VST en el entorno de JavaScript y de HTML 5, pero ninguno facilitaba el código necesario. Además, no se contaban con los conocimientos necesarios al respecto de VST como para poder improvisar un módulo sobre la marcha. La decisión tomada fue la de implementar el resto de módulos e investigar después al respecto de los dos que faltaban, pero el tiempo no ha sido un ayudante digno, en este caso, para desgracia de todos.

Continuando, de cara a la completa adaptación, hay ciertos matices que el autor no ha podido tener en cuenta, pero si le hubiera gustado implementar para alcanzar, en su opinión, la perfección o maestría en la adaptación de SoundCool a HTML5.
Entre ellos se encuentra, por ejemplo, el acceso a la tarjeta de sonido que SoundCool implementa en Maudio, la grabación de audio con una entrada distinta al micrófono o el ya mencionado Feedback del módulo de Delay.

También se habrá observado la ausencia del módulo Pitch, esto se debe a que su programación es tremendamente compleja, como así afirman los propios programadores de la Web Audio API. Se podría implementar mediante la programación de una FFT e incluso planteando la unión de JavaScript con MATLAB, pero esto parece excesivo para el motivo que nos ha traído aquí.

Por último, este humilde autor se ha “quedado con las ganas” de intentar la implementación de la Kinect en HTML 5, pero, tal y como concluyeron sus tutores y él, era excesivo para los requisitos de lo que es un TFG. Quizá en un futuro.

# 6. Bibliografía


[DavMath2010] David, M. "*Programación HTML 5*". Ediciones ANAYA Multimedia. 2010.

[WebAud00] "Web Audio API. HTML 5" http://webaudio.github.io/web-audio-api/ [Online]

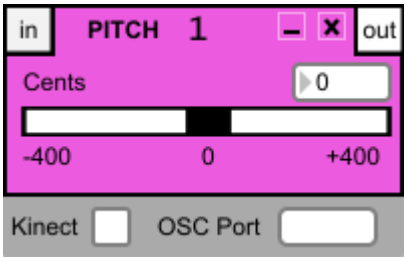
[AlvGarAl] Álvarez García, A. HTML 5, "Manual imprescindible". Ediciones ANAYA Multimedia, 2012.

[MidiB00] "Midi Bridge" http://abumarkub.net/abublog/?page_id=399 [Online]

# ANEXO I. Código SoundCool.

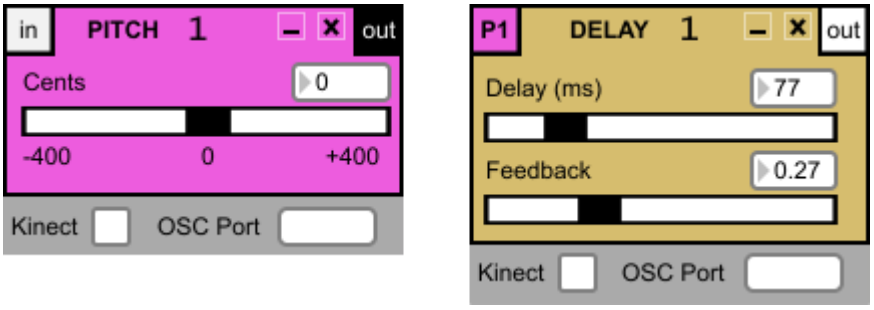
## Funcionamiento básico

El sistema *SoundCool* consta de diferentes módulos que permiten ser interconectados entre sí, produciendo así diferentes efectos y sonidos. Típicamente, un módulo constará de una entrada y una salida, tal y como se muestra en la siguiente figura:

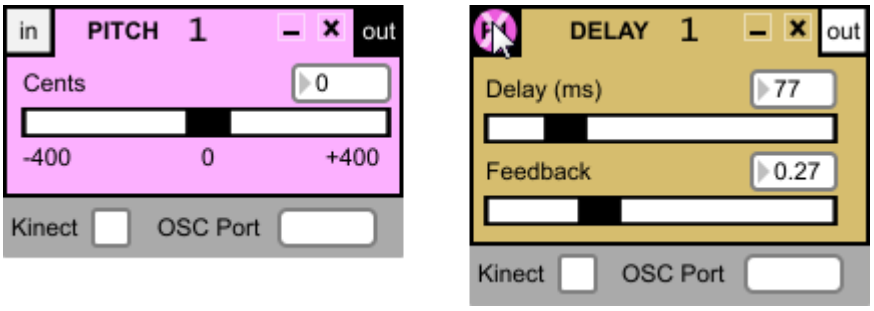



Además, todos los módulos cuentan con un número de identificación, de modo que si abrimos dos o más módulos iguales, cada uno esté identificado con un número distinto. Si cerramos todas las instancias de un mismo módulo, la cuenta se reiniciará.

Lo que llegue a la entrada del módulo, será procesado y llevado a la salida del mismo. Y lo que tengamos en dicha salida (típicamente, audio) lo podremos llevar a la entrada de otro u otros módulos. Para conectar dos módulos pincharemos, en primer lugar, en el botón de salida etiquetado como *out*. Tras esto, dicho botón (originalmente cuadrado) pasará a ser redondo y, al mismo tiempo, el botón de entrada (etiquetado como *in*) de aquellos módulos que podamos conectar pasará a ser también redondo. En este momento, si pinchamos sobre alguno de los botones *in*, se establecerá la conexión entre ambos módulos. Una vez realizada la conexión, el botón de *out* pasará a tener un fondo negro, y el botón de *in* mostrará la inicial y el número de identificación del módulo al que está conectado y tomará como color de fondo el color de fondo de dicho módulo. Para una mejor comprensión, obsérvese la siguiente figura:




La salida de un módulo puede llevarse a uno o más módulos, mientras que a la entrada, únicamente podemos conectar un módulo. Si pasamos el ratón por encima del botón *in* u *out*, se iluminarán el o los módulos conectados a dicha entrada o salida. En caso de pasar el ratón por encima del botón de entrada, además de iluminarse el módulo al que está conectado, aparecerá una "X" sobre el botón, que indica que si hacemos clic se romperá la conexión.

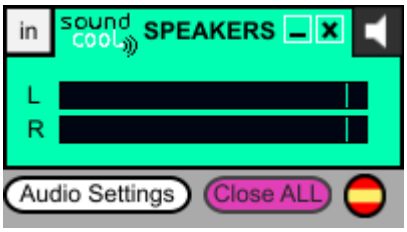


Si cerramos cualquiera de los módulos que forman parte de una conexión, esta se romperá automáticamente.

Todos los módulos cuentan con un botón de minimizado que, independientemente del tamaño maximizado del módulo, compacta todos los módulos a un mismo tamaño de módulo minimizado. Una vez minimizados, podemos volver al tamaño normal mediante el botón de maximizado. En la siguiente figura, podemos ver varios módulos minimizados.


## MAudio

Este módulo se corresponde con la salida de audio de nuestro ordenador, por lo que será siempre el punto final de cualquier configuración realizada con *SoundCool*. Disponemos también de un control de idioma de ayuda, con el que podremos seleccionar el idioma de los mensajes emergentes de ayuda (Inglés/Español). Mediante el botón *Close ALL* se cerrarán todos los módulos que estén abiertos en ese momento. El botón *Audio Settings* abre una nueva ventana en la que podemos configurar diferentes parámetros referentes a la tarjeta de audio de nuestro ordenador.

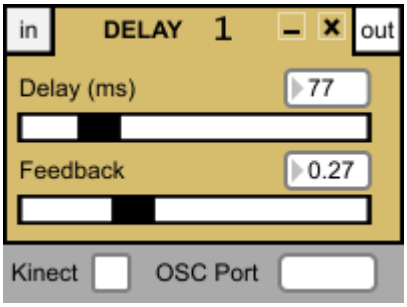



## MPlayer

El módulo reproductor, como podemos intuir por su nombre, nos permite reproducir archivos de audio. Mediante el botón *open*, podremos cargar archivos de audio de diferentes formatos (.wav, .aiff, .mp3, etc.). Una vez cargado el archivo de audio, podremos empezar a reproducirlo, pausar la reproducción o detenerla mediante los botones de control básico de reproducción *play/pause* y *stop*. A la derecha de estos dos botones, podemos ver un tercer botón con el que podremos conmutar entre reproducción hacia adelante o hacia atrás. Un interruptor etiquetado como *LOOP* nos permite activar la reproducción en bucle. El *slider* vertical de la derecha controla el volumen, y el horizontal etiquetado como *Speed* controla la velocidad de reproducción. Disponemos también de una barra de tiempo que nos muestra el avance temporal de la pista reproducida. Si pulsamos en cualquier punto de esta barra, nos desplazaremos a ese punto temporal.

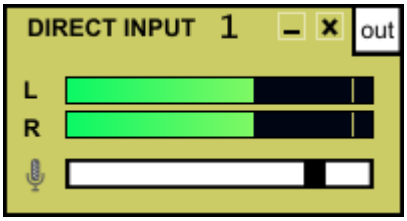

## MDelay

En este módulo disponemos de dos controles: el retraso o *delay* y la realimentación o *feedback*. Con el *delay* controlamos el retraso con el que la señal va ser realimentada según la proporción indicada en el control de *feedback*.



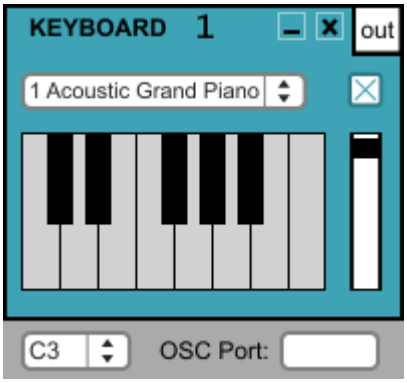

## MDirectinput

Este módulo tiene a su salida la señal captada directamente en el dispositivo entrada de nuestro ordenador (p. ej.: micrófono). Contamos también con un control de volumen.



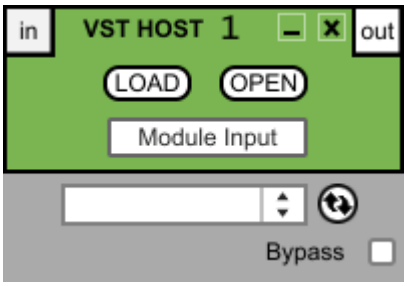

## MKeyboard

El elemento principal de este módulo es el teclado de piano. Mediante el interruptor situado en la parte superior derecha del módulo, podemos seleccionar el modo de funcionamiento. Cuando el interruptor esté activado, al pulsar las teclas del teclado de piano, se enviará a la salida la información de nota y velocidad MIDI. Esto será típicamente llevado a un módulo de VST donde se utilizará un instrumento virtual. Si el interruptor no está activado, se utilizará directamente el sintetizador interno del ordenador, produciendo sonido a través de él sin necesidad de conectarse a otro módulo. Cuando trabajamos en este modo, podemos seleccionar el instrumento MIDI en la caja de selección situada inmediatamente encima del teclado. Para controlar el volumen, disponemos de un *slider* en la parte derecha del módulo. Contamos también con un control de octava en la parte inferior izquierda del módulo.



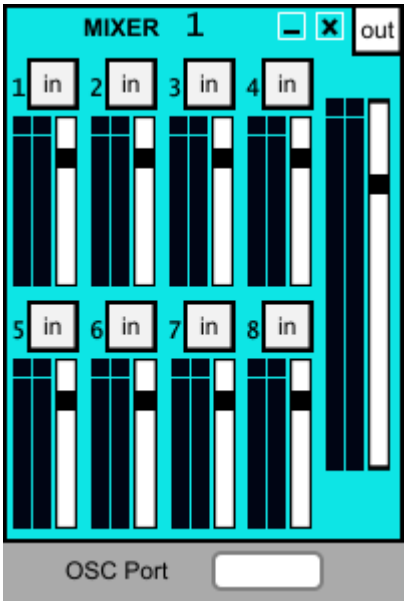

## MVST

Con este módulo podremos utilizar instrumentos o efectos VST. Utilizaremos efectos VST cuando tengamos audio conectado a la entrada. Por otro lado, utilizaremos instrumentos VST cuando tengamos un instrumento MIDI conectado a nuestro ordenador o cuando tengamos conectado a la entrada el módulo *MKeyboard*. La manera de cargar los efectos/instrumentos VST será pulsando el botón *LOAD* y localizando dicho efecto/instrumento en nuestro ordenador. Una vez cargado, podemos acceder a su interfaz gráfica y realizar las modificaciones deseadas mediante el botón *OPEN*. Bajo estos botones, tenemos un botón de selección *Module Input* / *MIDI Input*. Seleccionaremos *Module Input* cuando estemos trabajando con un módulo conectado a la entrada de *MVST* (bien audio, bien MIDI proveniente del módulo *MKeyboard*). Si, por el contrario, estamos trabajando con un instrumento MIDI conectado a nuestro ordenador, seleccionaremos *MIDI Input*. En caso de tener más de un instrumento MIDI conectado a nuestro ordenador, podemos seleccionar cuál queremos utilizar con la caja de selección situada en la parte inferior del módulo. Mediante el botón de refrescado, podremos renovar la lista de instrumentos conectados. Si se observa que el instrumento MIDI no se detecta correctamente, reinicie *SoundCool* después de conectar el instrumento. El interruptor *Bypass* lo utilizaremos cuando, teniendo audio conectado a la entrada, queremos sacarlo tal cual llega, sin aplicarle el efecto VST.



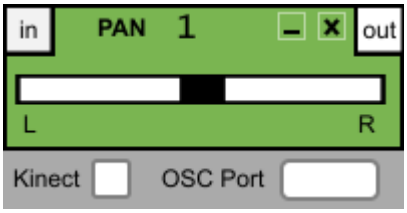

## MMixer

Este módulo cuenta con ocho entradas de audio. Cada una de ellas dispone de un control de volumen individual y la suma de todas ellas se lleva a la salida. Dicha salida cuenta también con un control de volumen. Este es el módulo que emplearemos cuando queramos llevar más de una señal de audio a la salida de audio de nuestro ordenador (*MAudio*).



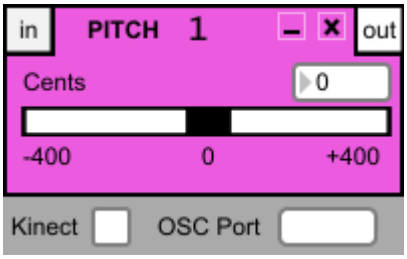

## MPan

Este módulo cuenta con un *slider* horizontal con el que podremos controlar la distribución derecha/izquierda del audio. Si situamos el control del *slider* en el extremo izquierdo, el audio estará en su totalidad en el canal izquierdo y será nulo en el canal derecho. A medida que desplacemos el control hacia la derecha, el audio irá desplazándose hacia el canal derecho hasta llegar a situarse en su totalidad en dicho canal y desaparecer del canal izquierdo.




## MPitch

Este módulo nos permite modificar la afinación del audio que llega a su entrada. Para ello, disponemos de un *slider* horizontal que cambia la afinación desde -400 cents (-2 tonos) hasta +400 cents (+2 tonos). Si a la entrada de este módulo tenemos conectado un *MPlayer*, conseguiremos que no cambie la afinación al cambiar la velocidad en *MPlayer*, con lo que el control de afinación pasa a ser exclusivo de *MPitch*.




## MRecord

Este módulo nos permite grabar lo que llegue a su entrada o lo que llegue directamente desde el dispositivo de entrada de nuestro ordenador. Típicamente, nos interesará grabar la salida de un *MMixer* donde confluyen todos los elementos de nuestro circuito *SoundCool*. Para escoger el modo de grabación (Entrada del Módulo/Dispositivo de Entrada) disponemos de un botón en la parte inferior del módulo (*Module Input* / *Input Device*). Tras pulsar el botón de grabado se

abrirá un cuadro de diálogo donde podremos introducir el nombre del archivo y el formato de grabación (.wav o .aiff). Tras esto, se iniciará una cuenta atrás a partir de "3" después de la cual se iniciará la grabación. El botón de grabación habrá entonces tomado la forma de un botón de *Stop*, que pulsaremos cuando queramos detener la grabación. Tenemos también un control de volumen y un indicador temporal en la parte inferior del módulo.