

Rapport de Travail, d'Études et de Recherche

Développer des instruments virtuels et des effets qui pourront être intégrés dans un lecteur audio multipiste

Encadré par: Michel Buffa

Sommaire

1.Introduction

- **Présentation du groupe**
- **Présentation du sujet**

2.État de l'art

- **Concepts**
- **Présentation de Web audio**

3.Travail effectué

- **Demos implémentées**
 - **Boîte à rythmes**
 - **Synthétiseur**
 - **Microphone**
 - **Lecteur audio**
- **Nos applications**
 - **Boîte à Rythmes**
 - **Synthétiseur**
- **Application joignant 2 instruments**

4.Gestion de projet

- **Wikipédia**
- **GitHub**
- **Gestionnaire des tickets**
- **Réunions régulières**
- **Points faibles / Points forts**

5.Conclusion

6.Perspectives et réflexions personnelles

1.Introduction

Pour la petite histoire, le monde du web s'est considérablement amélioré depuis son origine, au début, le web ne ressemblait pas à ce que l'on connaît, l'affichage était rudimentaire mais avec l'avènement de NCSA Mosaic en 1993 qui est le premier navigateur à avoir affiché les images (GIF et XBM) dans les pages web, puis à supporter les formulaires interactifs dans les pages. Il a causé une augmentation exponentielle de la popularité du World Wide Web.

La gestion de l'Audio sur le Web était assez primitive jusqu'à très récemment (grâce à des plugins tels que Flash et QuickTime). L'introduction de la balise <audio> en HTML5 est très importante, elle permet la lecture audio de base en continu . Mais, elle n'est pas assez puissante pour gérer les applications audio plus complexes.

Ce qui nous amène à l'une des toutes dernières innovations, l'API Web Audio qui introduit de nouvelles fonctionnalités audio à la plate-forme web. L'API est capable de positionner de façon dynamique ou spatiale et de mélanger plusieurs sources sonores dans l'espace tridimensionnel. Elle dispose d'un puissant système modulaire de routage, effets à l'appui, un moteur de convolution (**info**) pour simulation de la pièce, plusieurs départs, mixages, etc... la lecture du son horaire fixe est assuré pour les applications musicales nécessitant un haut degré de précision rythmique. Étayer l'analyse en temps réel de visualisation et le traitement direct de JavaScript est également pris en charge.

Nous allons étudier cette API dans le but de développer nos propres instruments de musique dans le cadre de ce TER.

Groupe:

Les personnes composant le groupe sont:

- Brunel Emmanuel
- Oliver Donovan
- Porta Benjamin
- Wenzinger Quentin

Nous sommes étudiants en M1 IFI.

Sujet:

La librairie Web Audio permet de créer des applications audio de grande qualité, qui vont tourner dans le browser : logiciels multipiste, lecteurs audio avancés (avec boucles, visualisations qui dansent en musique, etc), instruments virtuels (synthétiseurs, échantillonneurs), support midi, effets temps réel. Pour ceux qui connaissent un peu la Musique Assistée par Ordinateur (MAO), on a l'équivalent des VSTs et des VSTis directement accessibles en JavaScript. Cette API est très récente et tourne depuis peu dans tous les navigateurs sauf IE (à venir pour la version 12). Le but de ce TER consiste à développer des instruments virtuels et des effets qui pourront être intégrés dans un lecteur audio multipiste écrit par Michel Buffa et ses étudiants, destiné aux musiciens amateurs désirant travailler leur instruments en jouant par dessus des "backtracks", des pistes d'accompagnement. Par exemple: je prends une chanson des Guns and Roses ou d'ACDC, la vraie chanson, j'enlève la guitare solo et c'est moi qui joue par dessus. Dans ce cas, si je veux enregistrer le mix de ma guitare + la backtrack, je dois brancher ma guitare dans une entrée audio de l'ordinateur, Web Audio le supporte. Mais j'aimerais aussi avoir le son d'Angus Young ou de Slash. Là il me faut de la distorsion, un peu de compression, égaliser les fréquences, etc Ça tombe bien, on a ça aussi tout fait dans Web Audio. Il est aussi possible de faire ses propres effets. Si je suis pianiste nomade, j'ai peut être envie d'utiliser un instrument virtuel, un orgue, un piano virtuel, un synthétiseur... Dans ce cas sur mon ipad, je dessine un clavier, et je veux jouer dessus, ou bien je branche un vrai clavier midi sur mon ipad ou sur mon PC et je veux que les événements midi entrant se transforment en sons générés par l'ordinateur. Et je veux toujours pouvoir enregistrer le résultat. Etc... Bien sûr, tout cela sans RIEN installer sur son ordinateur, juste avec une application web écrite en HTML5! Bien sûr, nous ne proposerions pas ce sujet si tout cela n'était pas possible. Le sujet du TER consiste donc à explorer cette nouvelle API, développer quelques effets et instruments virtuels, intégrer le tout dans le lecteur audio multipiste (pour s'accompagner et enregistrer nos performances).

2.État de l'art

Concepts:

Définitions des principes majeurs nécessaires à la bonne compréhension de notre TER:

Abstraction : Le son est représenté par un **signal analogique**. Pour le numériser, on l'**échantillonne** :

Échantillonnage temporel: on prend la mesure du signal à intervalle régulier (fréquence d'échantillonnage).

Échantillonnage numérique: la valeur analogique est convertie en valeur numérique sur un nombre limité de bits.

Le signal est enregistré sous forme numérique selon deux principales méthodes :

Méthode temporelle: chaque échantillon temporel est enregistré (formats wave, au, ...).

Méthode fréquentielle: on décompose le signal par transformée de Fourier (formats MP3, ...).

Échantillon: Un échantillon (**sample** en anglais) est un extrait de musique ou un son réutilisé dans une nouvelle composition musicale, souvent joué en boucle. L'extrait original peut être une note, un motif musical ou sonore quelconque. Il peut être original ou réutilisé en dehors de son contexte d'origine.

Synthèse sonore: C'est un ensemble de techniques pour la **génération de signaux sonores**.

- Au niveau musical, elle permet de créer de nouveaux objets sonores. Dans ce contexte, le but n'est pas de reproduire des sons existants mais plutôt d'en inventer de nouveaux.
- Au niveau des télécommunications, elle permet de réduire la quantité d'informations lors de la transmission d'un message audio : celui-ci est alors décrit par ses paramètres de synthèse qui sont les seules données transmises.
- Au niveau de la réalité virtuelle et des jeux vidéo, la synthèse sonore permet d'augmenter la sensation de présence de l'auditeur-acteur en gérant les interactions entre l'acteur et son environnement sonore. L'acteur agit de façon directe ou indirecte sur les paramètres de synthèse.

MAO: La **musique assistée par ordinateur** (MAO) regroupe l'ensemble des utilisations de l'informatique comme outil associé à la chaîne de création musicale depuis la composition musicale jusqu'à la diffusion des œuvres, en passant par la formation pédagogique au solfège ou aux instruments.

VST: (sigle de **Virtual Studio Technology**) est l'un des protocoles pour les plugins audio existant dans le monde des logiciels.

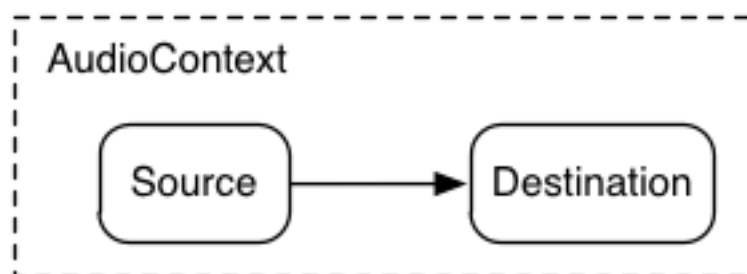
•**Instruments VST (VSTi)**. Ces plugins sont des instruments de musique virtuels pilotés en midi, à l'instar des machines "hardware". Les sons sont calculés et générés par le processeur de l'ordinateur, ou lus à partir de banques sonores. Ces "VST Instruments" permettent de simuler de vrais appareils comme un synthétiseur, un sampler ou une boîte à rythmes.

Présentation de Web audio

Avant de se familiariser à l'API Web audio, nous nous devons de montrer ce que l'on peut faire avec le **tag <audio>** de l'API audio de base en html5. Elle gère le streaming, c'est un objet **DOM** avec propriétés, événements et méthodes, mais peu adapté pour des applications telles que séquenceur, instruments virtuels ...

L'API a été conçu pour permettre le **routage modulaire**. Les opérations de base sont réalisées avec des nœuds **audio** qui sont liés ensemble pour former un **graphe** de routage audio. Plusieurs sources, avec différents types de canaux sont pris en charge, même dans un contexte unique. Cette conception modulaire permet la souplesse nécessaire pour créer des fonctions audio complexes avec des effets dynamiques.

Exemple d'un routage simple:



Ce qui nous donne avec un seul son:

```
var context = new AudioContext();

function playSound() {
  var source = context.createBufferSource();
  source.buffer = dogBarkingBuffer;
  source.connect(context.destination);
  source.start(0);
}
```

Les Nœuds sont reliés par leurs entrées et sorties. Chaque entrée ou sortie est constitué de plusieurs canaux, toute structure de canal discret est pris en charge, y compris les mono, stéréo, quad, 5.1, etc... .

Les **sources audio** peuvent provenir de plusieurs endroits:

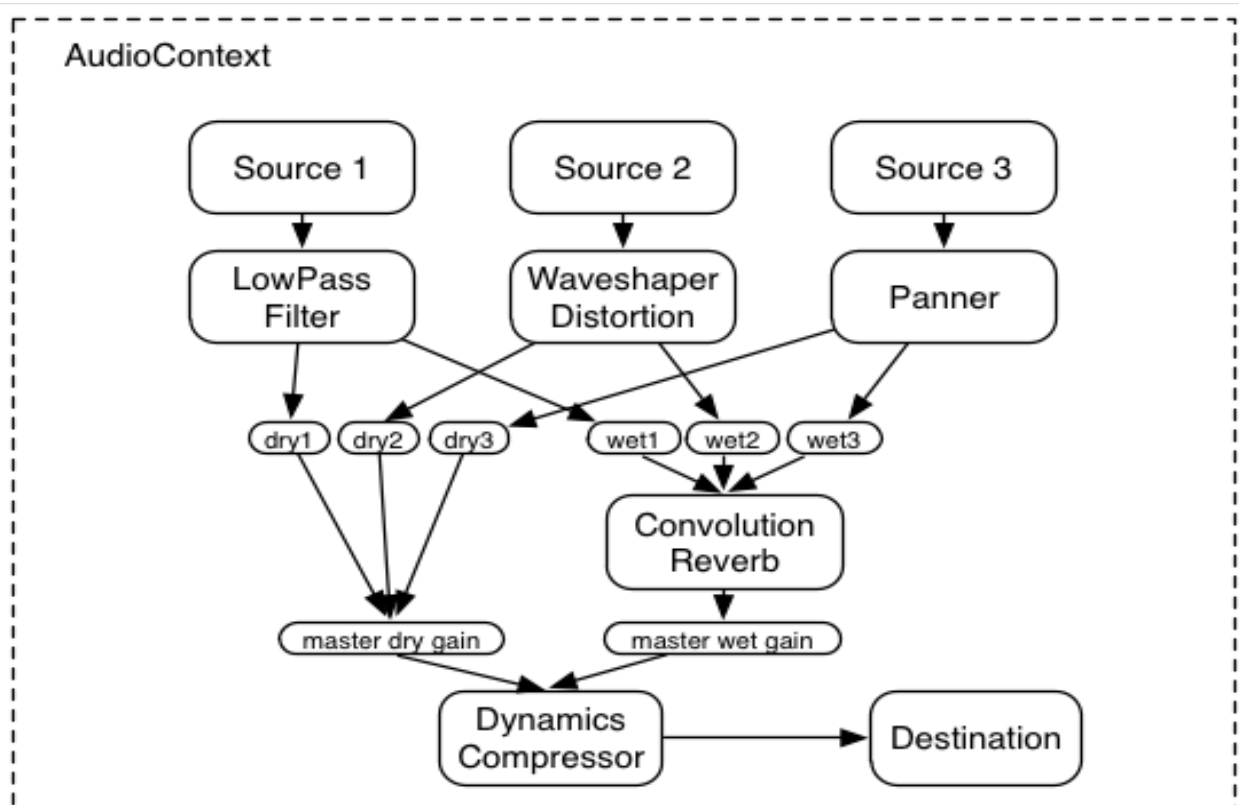
- Générées par JavaScript par un **nœud audio** (comme un oscillateur)
- A partir de données **PCM** brut
- Via des éléments de médias HTML (comme les balises **<video>** ou **<audio>**)
- D'un **MediaStream WebRTC** (comme une webcam).
- A l'aide du plugin **Recorder.js**, pour enregistrer/exporter la sortie des nœuds web audio, grâce à ses méthodes:

⇒ **rec.record()** ⇒ **rec.stop()** ⇒ **rec.clear()** ⇒ etc...

Le temps est contrôlé avec précision, une faible latence, avec la possibilité de cibler des échantillons spécifiques, même à un taux d'échantillonnage élevé.

L'API Web Audio permet également de contrôler la manière dont l'audio est spatialisé. Elle traite de l'atténuation induite par la distance ou le décalage Doppler induit par une source mobile (ou l'auditeur en mouvement).

Exemple d'un routage plus complexe:

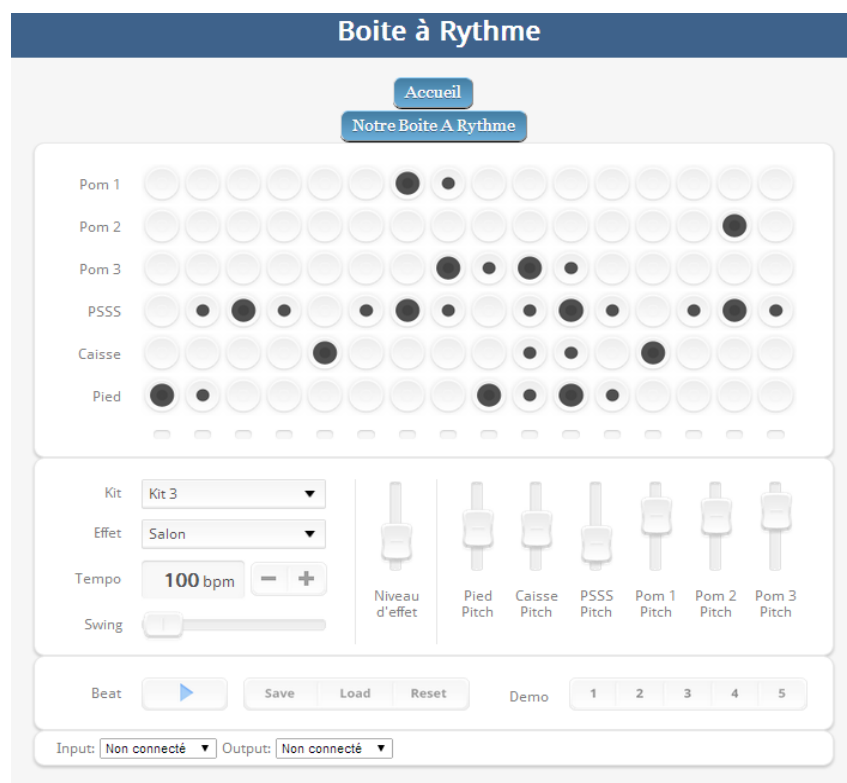


3. Travail effectué

Nous avons commencé par étudier différentes démonstrations disponibles sur internet dans le but de nous familiariser à l'Api Web audio.

Demos implémentées

Boîte à rythmes



Une boîte à rythmes est un instrument de musique électronique imitant une batterie ou des instruments de percussions.

À la différence d'un séquenceur conventionnel, la boîte à rythmes est basée sur la programmation de **patterns**, qui sont des groupes finis de mesures reproduits de façon cyclique. La programmation se fait entrant les notes une à une sur un graphique de patterns, divisé en mesures, elles-mêmes subdivisées selon un choix préalable de l'utilisateur (en noires, croches, etc.). Une autre différence réside dans le fait que dans une boîte à rythme le concept de durée d'une note n'existe pas, **chaque son étant toujours reproduit en totalité**.

Synthétiseur



Un synthétiseur, ou trivialement synthétiseur, est un instrument de musique capable de créer et de manipuler des sons électroniques au moyen de tables d'ondes, d'échantillons ou d'oscillateurs électroniques produisant des formes d'ondes que l'on modifie à l'aide de circuits composés de filtres, de modulateurs d'amplitude, de générateurs d'enveloppe.

Parmi les techniques de base, les plus répandues sont la synthèse additive, la synthèse soustractive, la synthèse FM, la modélisation physique ou la modulation de phase.

Ici la technique employée est celle de la synthèse sonore additive, qui consiste à créer un son en superposant des signaux sinusoïdaux harmoniques. Elle bénéficie des progrès de l'informatique en matière de puissance de calcul et il est aujourd'hui facile d'ajouter en temps réel des dizaines d'oscillateurs dans des logiciels de synthèse sonore modulaires tels que Max/MSP ou Reaktor fonctionnant sur un ordinateur domestique courant. Ici la synthèse est gérée par une interface graphique, qui permet notamment de piloter les oscillateurs.



La section Mode permet de faire des modifications globales sur la sortie du synthétiseur :

- On pourra y modifier la fréquence des notes, également appelé tempo.
- Le trémolo des oscillateurs 1 et 2. (variation de l'intensité de la note autour d'une valeur moyenne en conservant la valeur de départ)
- Altérer la forme de l'onde de sortie. (sinusoidale, carré, scie ou triangle)



Un synthétiseur est composé de plusieurs oscillateurs. L'idée est d'additionner le signal des différents oscillateurs pour créer un nouveau signal qui sera la résultante des différents signaux. On peut donc créer un signal complexe à partir de signaux simples. Ici on peut modifier ces filtres pour chaque oscillateurs :

- La forme de l'onde.
- L'intervalle, qui correspond aux octaves.
- Le Detune pour désaccorder finement un oscillateur : cela donne l'impression d'avoir plusieurs sons, ou un son plus épais.
- Le Mélange, qui représente la variation du mouvement d'une note entre ses hauteurs.

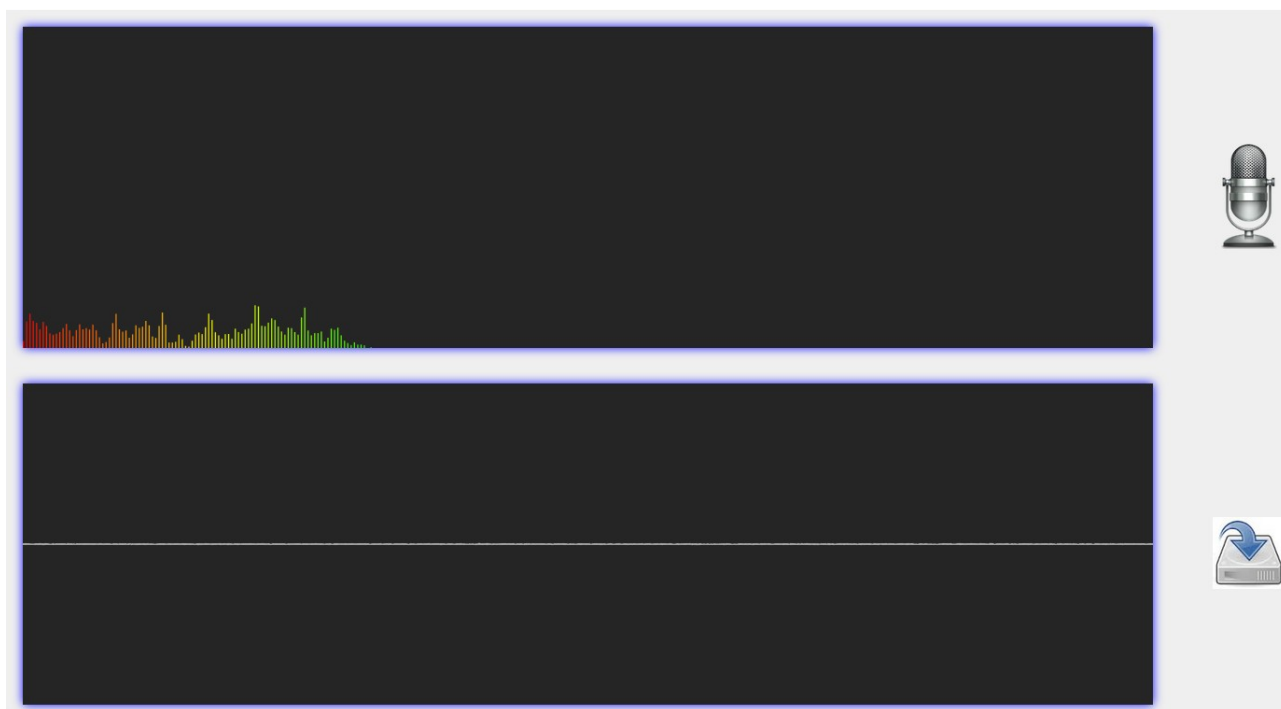


Le son produit par l'oscillateur d'un synthétiseur traverse le module VCF (Voltage Controlled Filter) pour en filtrer les fréquences.

Chaque note aura son propre filtre (contrairement à un equaliseur qui affecte l'ensemble des notes de façon uniforme). Le filtre agira de façon active sur le son en étant modulé au moyen d'enveloppes (enveloppes de hauteur, de vitesse).

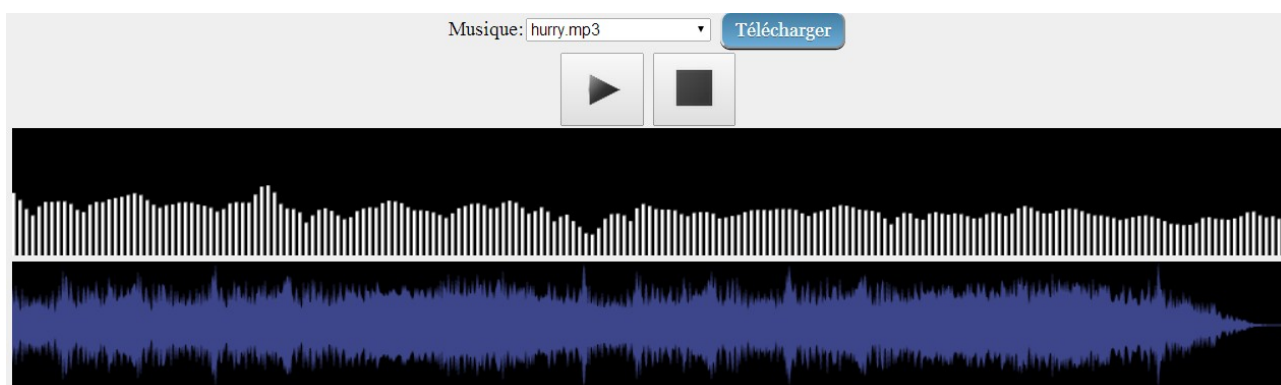
Le rôle du filtre est toujours le même : façonner un timbre de base, au contenu en harmoniques fixe ou variable, en supprimant une partie de ce qu'il reçoit en provenance de l'oscillateur.

Microphone



Permet l'enregistrement de l'entrée audio en direct et de télécharger celle-ci dans des fichiers .WAV

Lecteur audio

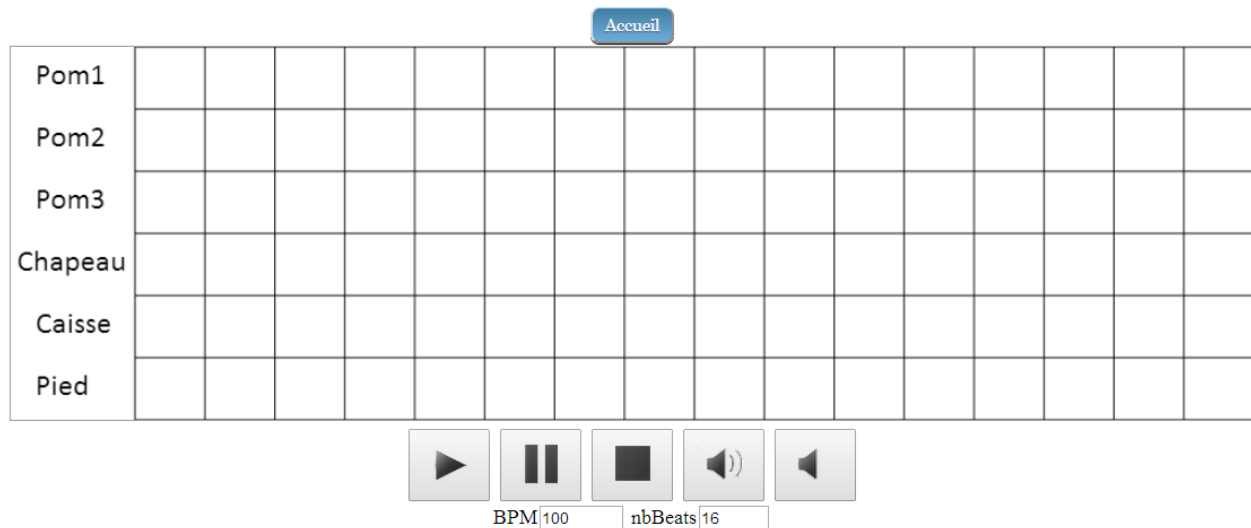


Lit une musique sur le le navigateur web (téléchargée avec Ajax/Xhr2) et affiche ses fréquences à 60 images/secondes

Nos applications

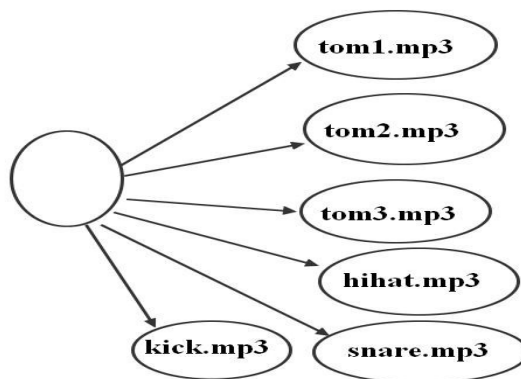
Boîte à Rythmes

Notre Boîte A Rythme



Nous avons par la suite tenté de faire notre propre boîte à rythme, pour commencer, les cases ont été dessinées, ensuite la ligne qui parcourt la piste ($BPS * \text{largeurPixelsUnBeat} = 1 \text{ seconde}$). La boîte à rythmes contient six sons (Pom1, Pom2, Pom3, Chapeau, Caisse, Pied), on crée un nœud web audio qui sera la source du graphe, on connecte la source (sample) au nœud de destination (speakers) et on joue. A chaque tours, il faudra reconstruire le graphe (web audio est optimisé pour ça).

Voici un schéma des sons de la boîte à rythmes:



Ensuite, nous avons implémenté la **gestion de N échantillons** sur la boîte à rythmes (en PHP), l'idée est simple, on peut rajouter des échantillons sur la boîte à rythmes, en fonction de ce nombre, on recalculera son interface.

Puis nous avons tenté d'appliquer des **filtres** (qualité et fréquence), lorsque le graphe est reconstruit, il va chercher dans les formulaires la valeur de la fréquence et de la qualité si le filtre est activé. Le filtre est créé, puis relié à la source et à la destination.

```
//Création d'un filtre avec web audio
this.filter = contextSound.createBiquadFilter();
this.filter.type = 0; // LOWPASS
this.filter.frequency.value = 5000;

//On applique les filtres
this.changeFrequency(document.getElementById("freq"));
this.changeQuality(document.getElementById("qual"));

//On connecte le filtre à la destination
this.sourceNode.connect(this.filter);
this.filter.connect(contextSound.destination);
```

Enfin, nous avons affiché des informations sur le son joué (fréquences / spectre / wave), pour ce faire, on utilise un **analyser** relié au filtre et au nœud javascript présenté ci-dessous.

```
// On cree le noeud javascript pour générer les courbes
var javascriptNode = contextSound.createJavaScriptNode(1024, 1, 1);
javascriptNode.connect(contextSound.destination);

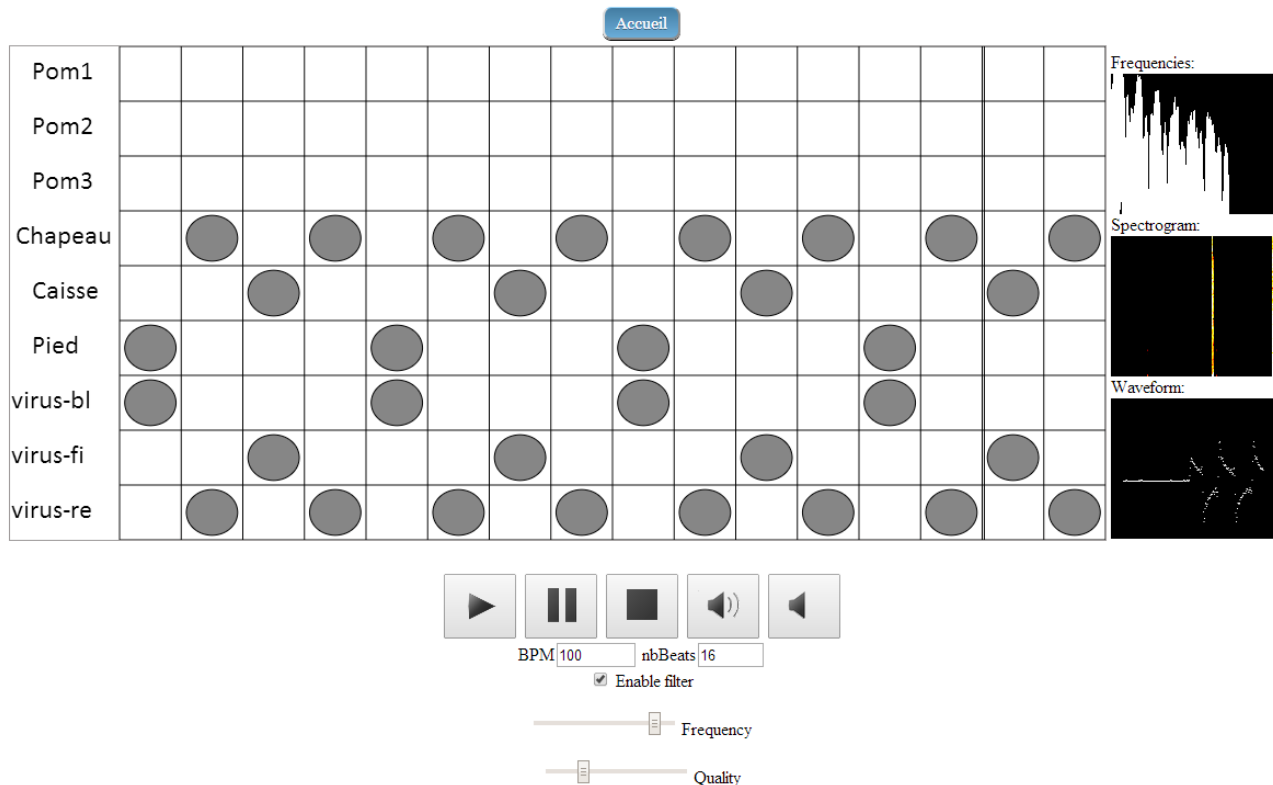
// Quand le noeud javascript est appelé, on utilise
// les informations de l'analyser pour dessiner le son
javascriptNode.onaudioprocess = function () {

    // get the average for the first channel
    var array = new Uint8Array(analyser.frequencyBinCount);
    analyser.getByteFrequencyData(array);

    drawSpectrum(array);
    ...
}
```

array contient un tableau de données des fréquences, la fonction **drawSpectrum** crée l'affichage des fréquences en dessinant des rectangles blancs sur fond noir de taille **200 – value**, value étant la valeur d'une case du tableau de données des fréquences. Tout cela est censé se dérouler sur un nœud, mais comme on a un instrument à échantillons, le flux audio n'est pas continu, il se peut qu'entre deux temps, on prenne la mesure et que le résultat du tableau soit nul (que des 0), pour pallier à ça, il a été décidé de ne dessiner la courbe à l'unique condition que le tableau ne soit pas nul.

Mais rien ne vaut une illustration en image:



Synthétiseur

Après l'étude du synthétiseur de démonstration, nous avons toutes les connaissances théoriques sur le fonctionnement d'un synthétiseur pour refaire le notre.

Ce dernier permet d'utiliser divers filtres de traitement de signal:

-Contrôle du volume en modifiant l'amplitude du signal sonore.

```
ctxGain.gain.value = $( "#volume" ).slider( "value" ) / 100;
```

-Contrôle de l'acoustique, qui crée une réverbération plus ou moins prononcée.

```
var revNode = context.createConvolver();
revNode.buffer = buffer2;
```

-Le Shape, qui permet de choisir la forme de l'onde sonore. (Sinusoidale, carré, en dent de scie ou en triangle)

```

switch (valShape) {
  case 0:
    oscillator.type = oscillator.SINE;
    break;
  case 1:
    oscillator.type = oscillator.SQUARE;
    break;
  case 2:
    oscillator.type = oscillator.SAWTOOTH;
    break;
  case 3:
    oscillator.type = oscillator.TRIANGLE;
    break;
}

```

-Contrôle de la fréquence de l'onde sonore.

```
oscillator.frequency.value = $( "#freq" ).val() * 10;
```

-La coupure, ou passe-bas, permet de définir le seuil de fréquence pour couper les notes. Par exemple, un filtre Passe-bas dont on aura fixé la fréquence de coupure à 8 KHz réduira progressivement le spectre sonore situé au-dessus de 8 KHz sans toucher les fréquences situées en-dessous.

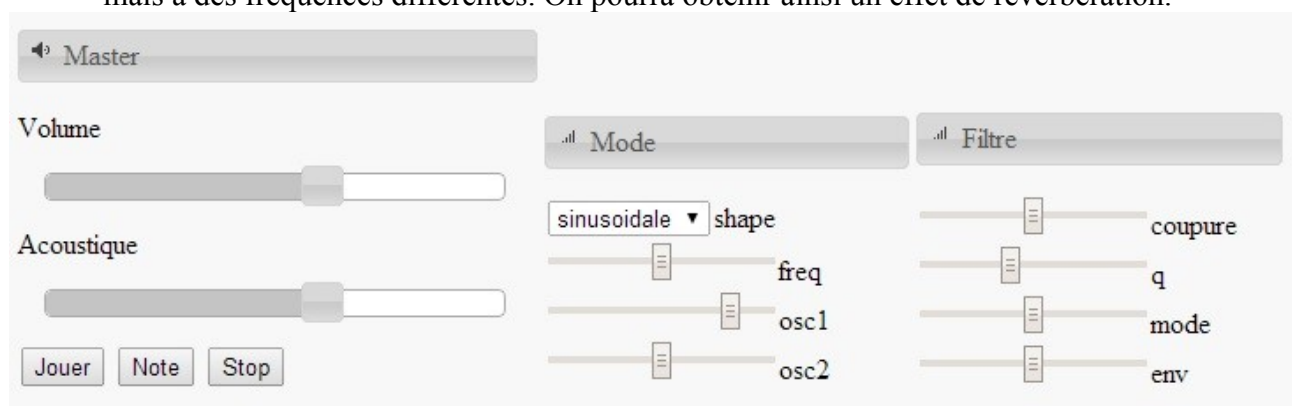
```
filter.frequency.value = 100*Math.pow(2.0, 3 * ($("#coupure").val()/100));
```

-Le Q, ou résonance, boost le spectre sonore dans les environs de la fréquence de coupure.

```
filter.Q.value = 1 + 30*$( "#q" ).val();
```

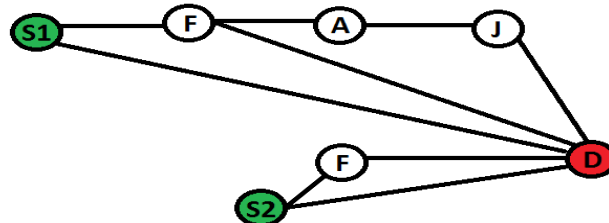
-Le Env, qui filtre les attaques d'une note. (permet d'avoir une note sans attaque, jusqu'à une longue attaque)

Ce synthétiseur est équipé de deux oscillateurs, travaillant ici sur la même forme d'onde mais à des fréquences différentes. On pourra obtenir ainsi un effet de réverbération.



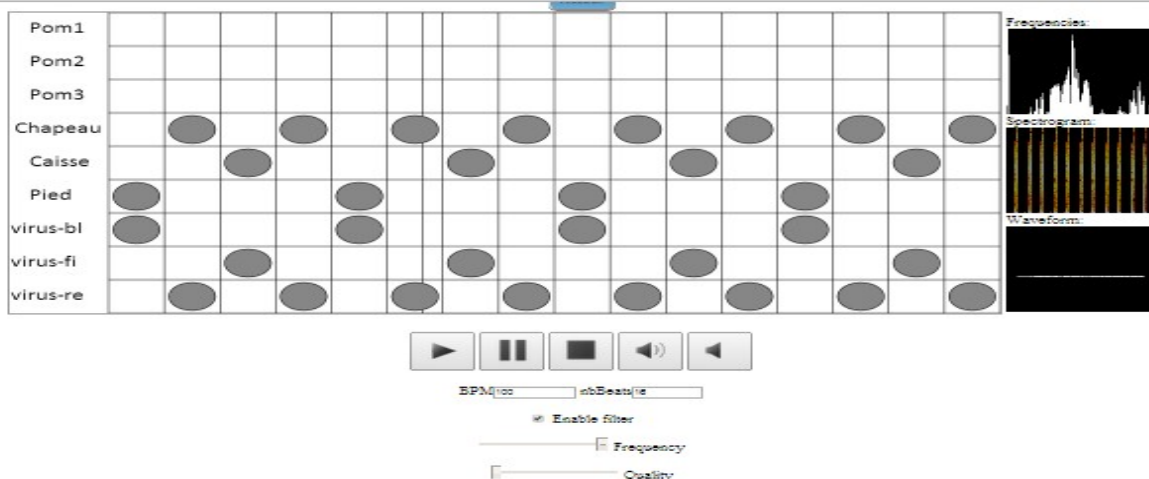
Application joignant 2 instruments

Nous voilà enfin arrivés à l'étape finale de notre projet, avoir deux instruments de musique au sein de la même page. Pour réaliser cette opération, il faut avoir en tête à quoi ressemble le graphe représentant nos deux instruments.



S1 représente notre boîte à rythmes (important: si F relié à D, S1 ne l'est pas et vice-versa). S2 représente le synthétiseur de manière simplifiée.

Pour intégrer deux instruments, il suffit donc simplement de les relier tous les deux au nœud de destination (la sortie audio).



4. Gestion de projet

Wikipédia

Nous avons réalisé un wikipédia ainsi qu'une page d'aide afin de fournir de plus amples explications sur le travail réalisé.



GitHub

Un GitHub repository a été créé afin d'héberger notre site web. Nous permettant également de voir les versions de chaque fichier et ainsi travailler en groupe sans se gêner ou se ralentir.

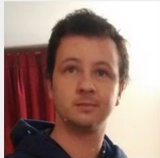
The screenshot shows the GitHub interface for the repository 'DonovanOliver / TER2'. The top navigation bar includes the GitHub logo, a search bar, and links to 'Explore', 'Gist', 'Blog', and 'Help'. The repository name 'DonovanOliver / TER2' is displayed, along with 'PUBLIC' status and interaction buttons: 'Unwatch' (3), 'Star' (0), and 'Fork' (1). Below the repository name, the 'TER2 / Commits' section is visible, showing a list of commits on the 'master' branch. The commits are grouped by date:

- Jun 17, 2014**
 - Commit: **PrésentationV2** (hash: 4cf10d2cf6) by pb402385, authored 20 hours ago. Includes a 'Browse code' link.
- Jun 16, 2014**
 - Commit: **Ajout Gestion multi instrument** (hash: 27e93607f6) by pb402385, authored yesterday at 21:54. Includes a 'Browse code' link.
- Jun 15, 2014**
 - Commit: **button** (hash: 40a507ce60) by DonovanOliver, authored 2 days ago. Includes a 'Browse code' link.
 - Commit: **Boite a rythme fun V2** (hash: 3d20553ed2) by pb402385, authored 3 days ago. Includes a 'Browse code' link.
- Jun 14, 2014**
 - Commit: **Presentation version 1** (hash: 707f6b4986) by pb402385, authored 4 days ago. Includes a 'Browse code' link.

Gestionnaire de tickets

Nous avons également créé notre propre système de gestion des tickets, propre et clair.

Donovan Oliver



Date:

Temps Humain(en heures):

Commentaire:

77

Date	2014-03-13
Heures de Travail	5
Commentaire	Mise en apprentissage de javascript avec le tutoriel de Mr Buffa. http://mainline.essi.fr/JavaScriptSlides/index.html

Date	2014-03-20
Heures de Travail	7
Commentaire	Implémentation de Test.html, avec les interfaces graphique et Son (sans noeud).

Date	2014-03-27
Heures de Travail	6
Commentaire	Implémentation d'un son, sans le spectre. Avec les liens de Mr Buffa. http://jsbin.com/ebaylPa/24/edit https://webaudiodemos.appspot.com/input/index.html http://www.storiesinflight.com/jsfft/visualizer_webaudio/

Réunions régulières

Toutes les semaines nous nous sommes concertés entre nous ainsi qu'avec notre coordinateur afin de se partager le travail. A ce sujet, nous avons tous fait au préalable un travail de recherche au début, notamment sur l'Api Web audio, mais également en javascript car nous n'étions pas très à l'aise n'en ayant jamais fait pour la plupart d'entre nous.

Ensuite, nous avons implémenté sur notre site quelques démos trouvées sur le web. C'est à ce moment que notre coordinateur monsieur Buffa nous a demandé de développer nos propres applications, il a chargé Benjamin et Donovan du développement d'un instrument virtuel à base d'échantillons (boîte à rythmes) et Quentin et Emmanuel de développer un instrument virtuel de synthèse.

Petit descriptif des taches:

Donovan:

- Mise en place du GitHub,
- Mise en place du site
- Mise en place du gestionnaire de tickets
- Mise en place des démos
- Création de la boîte à rythmes
- Création du Lecteur Audio montrant le spectre du son généré.
- Modification Rapport/Présentation

Benjamin:

- Création des pages du bouton "Divers" du site
 - permet l'ajout d'échantillon(s) sur le serveur
 - mini quizz mais abandonné faute de temps
- Création de la page d'aide (bouton "Wiki")
- Implémentation de l'animation de la page d'accueil
- Ajout des filtres sur la boîte à rythmes
- Ajout des analyzer (fréquences/spectre/wave)
- Ajout possible d'échantillon(s) sur la boîte à rythmes (PHP)
- Intégration de deux instruments sur une page
- Création Rapport/Présentation et finalisation.

Quentin:

- Mise en place et création de contenu du wikimédia
- Création d'un lecteur mp3/wav
- Étude du synthétiseur démo
- Création d'effet sonore de synthétiseur
- Modification Rapport/Présentation

Emmanuel:

- Rajout de contenu au wiki.
- Étude boîte à rythme.
- Étude du synthétiseur démo.
- Étude des concepts de traitement du signal.
- Modification Rapport/Présentation.
- Aide création d'effet sonore pour le synthétiseur.

Points faibles / Points forts

Points faibles:

- ▶ Manque de temps
- ▶ Jamais fait de Javascript auparavant
- ▶ Petits problèmes d'organisation au début
- ▶ WebAudio, une grosse API avec des concepts peu évidents
- ▶ Nous ne sommes pas musiciens et étions donc étrangers à pas mal de concepts
- ▶ Problème de rafraîchissement de l'affichage "wave" dans la boîte à rythmes
- ▶ Gestion du temps, du graphe (pas de stop/pause, on casse et on reconstruit le graphe à chaque fois, on avait pas compris au début)

Points forts:

- ▶ Gros progrès en JavaScript.
- ▶ On a eu des difficultés, notamment avec le fait que le son soit géré via un graphe, mais on pense qu'on a bien compris comment marche WebAudio maintenant
- ▶ Meilleures connaissances en musique désormais

5.Conclusion

Au travers de ces différentes applications, nous avons ainsi montré que l'API web audio fournit un panel d'outils très intéressants et innovants pour gérer le traitement de sons de manière interactive.

D'un point de vue pratique, nous avons appris les rudiments de la base en ce qui concerne l' API web audio, mais également en ce qui concerne l'audio en général (notamment le fonctionnement d'instruments comme le synthétiseur, la boîte à rythmes et les techniques de traitement employées pour analyser et modifier les sons), cela nous a permis de nous former à ces nouvelles possibilités qui seront probablement d'un impact majeur dans le web de demain.

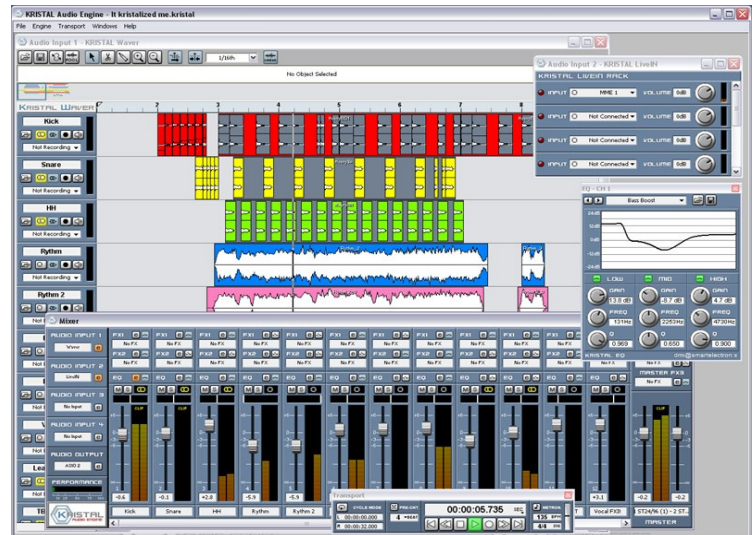
6. Perspectives et réflexions personnelles

Nous trouvons que nous n'avons vraiment pas eu assez de temps pour réaliser ce TER, nous aurions bien aimé pouvoir améliorer encore un peu nos instruments, afin de les achever.

Monsieur Buffa nous pousse à finaliser le projet pour éventuellement préparer des démos pour la première conférence sur Web Audio qui se tiendra à l'IRCAM en Janvier 2015 et cela est assez appâtant, mais il nous faudra alors considérablement refaçonner nos instruments pour être à la hauteur d'un tel événement.

Notre application a vocation à être développée afin de créer un instrument complet de type MAO tel que ce que l'on peut trouver sur le marché.

Voici un aperçu d'instruments de type MAO que l'on peut trouver en vente sur internet:



ANNEXES:

API Web Audio:	https://dvcs.w3.org/hg/audio/raw-file/tip/webaudio/specification.html
Demo boîte à rythme:	http://www.rivieraproject.fr/BoiteARythme/
Notre boîte à rythme:	http://www.rivieraproject.fr/BoiteARythme/fun/indexV2.php
Demo Synthétiseur:	http://www.rivieraproject.fr/Piano/
Notre synthétiseur:	http://jsbin.com/gokukazo/22/edit
Demo micro:	http://www.rivieraproject.fr/Micro/
Son:	http://www.rivieraproject.fr/Son/
Wikipédia:	http://skarn.fr/TER/Wiki/index.php?title=Synth%C3%A9tiseur
Aide:	http://www.rivieraproject.fr/Wiki/
Multi-Instrument:	http://www.rivieraproject.fr/integrationInstruments/noob.php
Notre GitHub:	https://github.com/DonovanOliver/TER2
Gestionnaire de tickets:	http://www.rivieraproject.fr/Tickets/
Quizz:	http://www.rivieraproject.fr/Divers/Quizz/
Quizz:	Lit aléatoirement une musique stockée sur le serveur au format mp3, la joue, on attend simplement la réponse, une légèrement erronée peut être acceptée grâce à soundex (fonction utilisée pour simplifier les recherches dans les bases de données, où vous connaissez la prononciation d'un mot ou nom, mais pas son orthographe exacte).
Recorder.js:	https://github.com/mattdiamond/Recorderjs

Quelques définitions du projet:**def convolution:**

Le produit de convolution est un opérateur bilinéaire et un produit commutatif, généralement noté « \ast », qui à deux fonctions f et g sur un même domaine infini, fait correspondre une autre fonction « $f \ast g$ » sur ce domaine, qui en tout point de celui-ci est égale à l'intégrale sur l'entiereté du domaine (ou la somme si celui-ci est discret) d'une des deux fonctions autour de ce point, pondérée par l'autre fonction autour de l'origine - les deux fonctions étant parcourues en sens contraire l'une de l'autre (nécessaire pour garantir la commutativité).

def DOM:

Le Document Object Model (ou DOM) est un standard du W3C qui décrit une interface indépendante de tout langage de programmation et de toute plate-forme, permettant à des programmes informatiques et à des scripts d'accéder ou de mettre à jour le contenu, la structure ou le style de documents XML et HTML1. Le document peut ensuite être traité et les résultats de ces traitements peuvent être réincorporés dans le document tel qu'il sera présenté.

def PCM:

La modulation d'impulsion codée ou MIC, (en anglais Pulse Code Modulation, généralement abrégé en PCM1) est une représentation numérique non compressée d'un signal analogique via une technique d'échantillonnage. Cette technique est utilisée pour la voix en télécommunications (RTC ou VoIP)

def WebRTC:

Web Real-Time Communication, littéralement *communication web en temps réel*