TOUCHVOICES : UN EXEMPLE D'UTILISATION DU WEB COMME OUTIL D'INTERACTION ET DE DEVELOPPEMENT MULTI-PLATEFORME (DEMO)

Pierre-Adrien THEO
Etudiant Master RIM
CIEREC – EA3068
Université Lyon-Saint-Etienne
pierreadrien.theo@gmail.com

RÉSUMÉ

C'est dans cadre de la performance *ElectroPhilia* organisée par la promotion 2016 du Master RIM que le projet *TouchVoices* s'est développé. Il s'agit d'un outil de transformation de la voix pour un chanteur improvisateur, développé sur une table tactile. Peu à peu, le cahier des charges s'est précisé et il a semblé intéressant de permettre au public une fois rentré chez lui de pouvoir exploiter le projet.

Pour répondre à ce cahier des charges, le projet *TouchVoices* a été développé essentiellement avec les outils Web que sont les langages HTML, CSS et JavaScript ainsi que le langage Faust.

Ce texte présente les différents outils qui ont été utilisés pour réaliser le projet aussi bien au niveau de l'interface interactive que de la couche audio. Il présente aussi les moyens utilisés pour communiquer directement de la page Web vers le logiciel Max et ainsi utiliser l'interface graphique comme contrôleur.

Grâce à ces outils, le projet *TouchVoices* peut être utilisé sur une large palette de plateformes permettant l'accès à un navigateur Web moderne.

1. INTRODUCTION

Aujourd'hui de nombreuses applications sont développées directement sur et pour le Web. Après l'utilisation des langages Flash ou Java, de plus en plus les développeurs se tournent maintenant vers le langage JavaScript et développent des outils de plus en plus performants pour les applications. Nous allons voir comment le projet *TouchVoices* a utilisé ces outils, en particulier pour la réalisation de l'interface utilisateur temps réel mais aussi pour le développement d'un outil compatibles avec la plupart des plateformes actuelles.

2. LES OBJECTIFS DE TOUCHVOICES

L'idée d'origine du projet *TouchVoices* était de réaliser un outil d'improvisation interactif et utilisable par tous. Cet outil devait permettre de mettre en boucle des séquences audio enregistrées en direct par le musicien, de les transformer et de les spatialiser.

Très vite, deux types de publics ont été choisis pour établir le cahier des charges :

- un public totalement novice en musique électronique qui devrait pouvoir utiliser l'outil sans connaissances particulières et depuis un maximum de plateformes (tablette, pc…),
- un public professionnel pouvant utiliser l'outil en live et faire évoluer la partie audio du projet en fonction des besoins.

L'outil Web répond parfaitement au besoin d'applications multiplateformes mais comporte d'autres nombreux avantages comme la prise en charge d'une grande partie des contrôleurs intégrés dans les plateformes actuelles (surfaces tactiles, accéléromètres, géolocalisation…) ainsi que la présence d'une très large communauté. C'est un outil très documenté et donc rapidement abordable pour la programmation.

Concernant la couche audio, deux options sont possibles. La première consiste à envoyer les données de l'interface graphique vers Max via WebSocket[1] pour permettre aux musiciens professionnels et programmeurs de faire évoluer la couche audio simplement. La deuxième consiste à réaliser toute la couche audio en langage Faust et de la compiler en JavaScript à l'aide de faust2asmjs pour l'intégrer totalement dans la page Web et en faire une application autonome.

3. LES OUTILS POUR L'INTERFACE GRAPHIQUE

Il a donc été choisi de travailler avec tous les outils courants du Web que sont les langages HTML, CSS et JavaScript. C'est avec ce dernier langage qu'une grande partie de la page est écrite. Encore une fois, le principal intérêt d'utiliser ces langages est l'accès à de nombreuses ressources.

[1] «WebSocket est une technologie évoluée qui permet d'ouvrir un canal de communication interactif entre un navigateur (côté client) et un serveur. Avec cette API, vous pouvez envoyer des messages à un serveur et recevoir ses réponses de manière événementielle sans avoir à aller consulter le serveur pour obtenir une réponse. » [1]

3.1. Couche interactive : Interact.js et Jquery.knob.js

La bibliothèque Javascript *Interact.js* [3] a permis de développer rapidement une interface compatible aussi bien pour des plateformes tactiles (multipoints) que non tactiles. Cette bibliothèque permet de reconnaître toutes les actions (clic, cliqué-déplacé…) réalisées sur objet CSS. Une fois ces actions reconnues, il devient possible de déclencher des fonctions et donc des actions dans la page Web (graphique, audio…).

Voici un exemple simple de mouvement d'un objet CSS :

```
interact('.cercle')
  .draggable({
    onmove: function (event) {
      var target = event.target,
      x =
(parseFloat(target.getAttribute
   ('data-x')) || 0) + event.dx,
      y =
(parseFloat(target.getAttribute
   ('data-y')) || 0) + event.dy;

      target.style.WebkitTransform =
        'translate(' + x + 'px, ' + y +
'px)';

     target.setAttribute('data-x', x);
     target.setAttribute('data-y', y);
        }
    }
  })
```

Ici, lorsqu'un mouvement de déplacement est fait sur un objet CSS de la classe *.cercle*, on ajoute les valeurs de placement en x et y respectivement aux attributs data-x et data-y de l'objet, on le déplace sur ces nouvelles valeurs et on met à jour les attributs. Cette commande est entièrement compatible multipoint ce qui est extrêmement important pour la portabilité du projet et le grand potentiel d'interactivité que cela crée.

Interact.js. propose ainsi de nombreuses commandes qui permettent rapidement la création d'une interface de contrôle complète. A cette bibliothèque, s'ajoute l'utilisation de la bibliothèque *Jquery.knob.js* [4] qui permet la création de potentiomètres qui sont intégrés dans l'interface de *TouchVoices*.

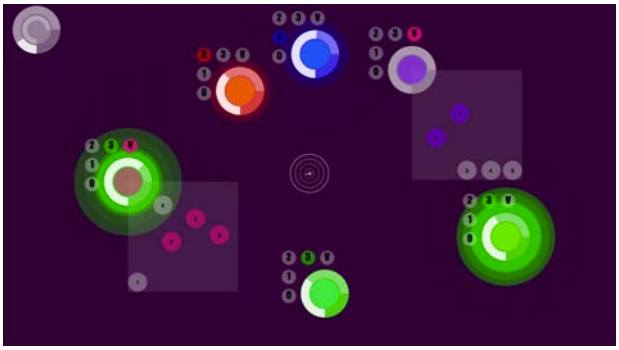

Figure 1 : Interface graphique de *TouchVoices*

3.2. Sous-couche : P5.js

La couche graphique a été créée via *P5.js* [5] une bibliothèque JavaScript développée à partir du logiciel Processing. Elle permet la création très rapide de graphismes simples. Comme dans la version stand-alone de Processing, une fonction *draw* est appelée à chaque *frame* pour générer un dessin. Dans le cas de *ToucheVoices,* le dessin est fixé par rapport à la place des objets CSS. A chaque frame, la fonction appelle les informations des attributs data-x et data-y qui représentent la place des objets pour générer son dessin.

Les fonctions « getAttributes » ou encore la bibliothèque *Jquery.js*[6] permet simplement d'appeler les attributs des objets CSS et d'en utiliser les valeurs.

Cette couche permet ici l'affichage de nombreux éléments graphiques cherchant à rendre l'interface plus intuitive (affichage du centre de l'espace de jeu, affichage du volume sonore présent dans l'objet…).

3.3. L'interface de *TouchVoices*

Dans le cas de *TouchVoices*, l'interface est constituée de 6 objets circulaires constitués d'un centre, un bouton rotatif, de 5 boutons fixes et d'une page d'effet attachée au nœud principal, comprenant 5 boutons.

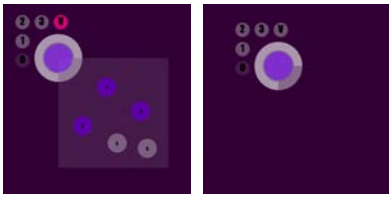

Figure 2 : Un objet avec et sans fenêtre d'effets

3.3.1. L'objet central

Le centre de cet objet central permet, lorsqu'il est "clique-déplacé", de déplacer l'ensemble de l'objet. Ce déplacement influe directement sur la spatialisation sonore de l'objet. Au centre de la fenêtre un dessin symbolise le centre de l'espace de jeu.
Autour de ce centre, est créé un bouton rotatif permettant de régler le volume sonore de l'objet.

3.3.2. Les boutons

En haut à gauche de chaque objet, se trouvent 5 boutons. Le bouton « 1 » permet de couper l'entrée microphone dans l'objet ou de couper la lecture de la boucle le cas échéant. Le bouton « 2 » permet l'activation de l'entrée micro dans l'objet, le « 3 » lance l'enregistrement d'une boucle et le « 4 » permet de faire jouer la boucle enregistrée. Enfin, le bouton « V » permet d'afficher la fenêtre d'effet de chaque objet. Chaque bouton a une couleur particulière permettant rapidement de voir dans quel mode se trouve l'objet. Cette information de « mode » est directement utilisé pour modifier l'affichage graphique de la couche réalisée sur P5.js.

3.3.3. La fenêtre d'effet

Chaque bouton contrôle un effet de l'objet. Par un simple clic sur le bouton, l'effet est activé. La position en x/y du bouton dans la fenêtre d'effet permet l'édition des paramètres de l'effet.
Ici, chaque effet porte un numéro et non un nom ou une description. C'est par le déplacement des objets que l'utilisateur se fera une idée de l'effet et non par une compréhension technique de l'effet.

3.4. La table Biin

Toute la partie graphique du projet *TouchVoices* a été adaptée et optimisée pour fonctionner sur une table Mosaïque de la société BIIN. Il s'agit d'un ordinateur implanté dans une table et fonctionnant avec un écran tactile multipoint de 46 pouces. La table utilise le système d'exploitation Windows 8.1 et un processeur i7. Ce matériel rare et très performant a été mis à la disposition du projet par le CIEREC. Elle a été acquise dans le cadre des projets de recherche PACAP [7] sur la captation du geste et FEEVER [8] sur la construction d'outils ubiquitaires pour le traitement du signal. Cet outil a permis de transformer le projet *TouchVoices* en un réel instrument interactif et optimisé pour le concert.

4. MISE EN PLACE DE LA COUCHE AUDIO

Le projet devait intégrer deux versions, une première pouvant interagir avec Max pour une plus grande souplesse de programmation lors des phases de développement et d'exploitation en live et une seconde intégralement Web réalisée sous Faust grâce à faust2asmjs.

4.1. Les interactions Web <> Max

Pour interagir directement depuis la page Web vers Max, l'objet [ol.wsserver] développé par Oliver Larkin de l'université de York [9], permet la création d'un serveur local directement dans un patch Max. Oliver Larkin propose quelques exemples et une bibliothèque graphique simple permettant de créer des pages Web de contrôle. Dans le cadre de *TouchVoices,* l'interface graphique proposée semblait trop simpliste et il a fallu détourner quelque peu l'objet original pour transférer les informations de la couche graphique Web vers Max.

Cet objet fonctionne grâce aux WebSockets, outils de communication permettant la communication entre un serveur et un client. Il faut donc ouvrir dans notre page Web (outil de contrôle) une connexion WebSocket vers le serveur. Ici, toute la phase de programmation du serveur n'est pas nécessaire puisqu'elle est directement faite dans l'objet Max. A chaque ouverture d'une nouvelle connexion WebSocket, le serveur est informé et la communication est ouverte.

L'objet Max reçoit les messages codés de la manière suivante : numéro du client puis message. Cela permet d'isoler l'information venant d'un client uniquement. De la même manière, des informations peuvent partir de Max vers tous les clients ou uniquement vers un client précis.

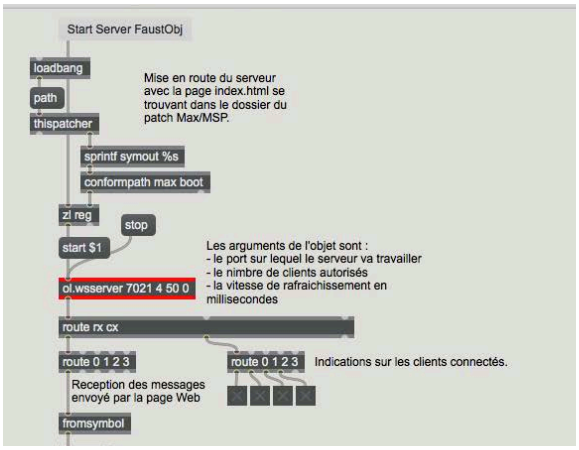

Figure 3 : Partir serveur du patch Max

4.2. La version FAUST

Faust fait partie du projet depuis le début de la conception de la couche audio. La plupart des effets est codée et optimisée en Faust grâce à FaustGen~ dans Max pour pouvoir les éprouver de manière très souple.

Une fois la couche audio entièrement finie, elle est transformée en patch Faust totalement indépendant. Chaque paramètre devant être utilisé dans le projet final est affiché sous la forme de curseurs et le patch est ensuite compilé sous la forme d'un fichier js à l'aide de la commande faust2asmjs[2].

Une fois le fichier intégré dans la page Web, toutes les commandes présentes sous forme de contrôleur dans la version .dsp deviennent éditables grâce à un adressage simple et la commande JavaScript « setvalue ».

Si la WebAudioAPI présente elle aussi des effets intégrée, l'utilisation de Faust dans le cadre de ce projet permet de conserver une couche audio identique entre la version Max et la version Web. Seul le procédé de compilation change.

4.3. La couche audio de *TouchVoices*

Le langage Faust ainsi que l'utilisation d'une correspondance a permis de développer différentes couches audio en fonction des besoins. Lors de la compilation pour le Web, une seule couche audio à été conservée. Il sera ici question uniquement de cette dernière regroupant les effets principaux.

[2] Pour un complément d'information sur l'utilisation de Faust et Asmjs, consulter http://jim2015.oicrm.org/actes/JIM15_Cipierre_T_et_al.pdf [10]

Figure 4 : Chaine audio de TouchVoices

4.3.1. Les effets

Le projet se voulant simple et accessible au grand public, il utilise des effets dont les modifications sont facilement audibles. Le langage est livré avec des bibliothèques d'effets très performantes [11], la plupart des effets utilisés est directement issue de ces bibliothèques :

- Boucleur : utilisation des tables
- Saturation : cubicnl (effect.lib)
- Munger : utilisation des tables
- Filtre : moog_vcf (filter.lib)
- Delay : sdelay (music.lib)
- Reverb : mono_freeverb (effect.lib)

4.3.2. La spatialisation

Pour la spatialisation du projet, deux options sont mises en place en fonction des versions mais les deux utilisent la même bibliothèque. Il s'agit de la bibliothèque HOA développée par une équipe du CICM [12]. Cette bibliothèque comporte de nombreux avantages. Elle est compatible pour Max ainsi que pour Faust, mais surtout elle dispose d'un mode d'écoute binaural.
Ainsi pour la version grand public, version WEB, le son est diffusé en écoute binaural pour permettre une écoute au casque de la spatialisation.
Pour la version concert du projet, la diffusion s'est faite en 4 canaux spatialisés toujours par la bibliothèque HOA.

5. UN AUTRE EXEMPLE D'UTILISATION : CRD BOURGOIN JALLIEU

Dans le cadre d'un atelier d'improvisation avec transformation électroacoustique organisé par Phillippe Moenne-Loccoz au sein du conservatoire à rayonnement départemental de Bourguoin-Jallieu, il m'a été demandé de réaliser des outils de transformation de la voix extrêmement simples d'utilisation pouvant allier le geste au son. Cet atelier devait commencer début mars 2016 et le projet devait être entièrement prêt à l'emploi au moment des premières séances. En effet, le public étant constitué d'élèves du conservatoire non spécialistes et débutants, et les séances étant très courtes, tous problèmes informatiques étaient à bannir.

Nous sommes partis du principe que la plupart des étudiants étaient munis d'un smartphone et que les accéléromètres intégrés dans ces derniers seraient la solution la moins onéreuse et la plus simple à mettre en place pour capter les mouvements des chanteurs. L'idée a donc très rapidement été de développer un outil Faust compilé pour Android sous forme d'application. Mais cela présente deux problèmes principaux :

- Ne connaissant pas le système d'exploitation des smartphones des étudiants qui seraient sur le projet, il nous paraissait dommage de mettre de côté les étudiants munis de téléphone tournant sous iOS et sous Windows-Phone.
- Lors de la prestation, les transformations seraient générées et diffusées grâce à un patch Max déjà existant et le son devait être capté par de vrais microphones et non de simples microphones de smartphone. Malgré tout, l'interface devait être identique entre les deux versions.

Il a donc été choisi de développer deux versions du projet.

La première version est hébergée sur le Web et est totalement autonome. Cela permet aux étudiants de tester l'outil de transformation en dehors des séances de travail (moyennant une connexion internet suffisamment rapide).

Une deuxième version serait intégrée dans le patch Max avec l'objet Wsserver présenté plus haut et chaque étudiant connecté pourrait piloter ses transformations exactement de la même manière qu'avec l'outil Web mais sur un système professionnel. La possibilité de compiler Faust aussi bien en JavaScript pour le Web que pour Max pour la version « concert » permet de garder des transformations toujours identiques.

6. PERSPECTIVES

Le projet *TouchVoices* a permis de mettre en place grâce aux outils Web une interface intuitive, totalement dédiée au projet. Un tel projet permet aussi d'exploiter les ressources de la Web Audio API et d'en éprouver les limites. On pourra citer parmi ces limites les temps de latence encore importants et aléatoires, et une compatibilité partielle des navigateurs web avec l'accès à l'entrée microphone du périphérique.

Néanmoins, la possibilité de créer des interactions entre du traitement du son et une interface Web (à l'aide de Max ou de FAUST) ouvre la voie vers de nombreux projets. Actuellement, plusieurs projets sont en cours de développement comme une interface de contrôle d'objets OpenGL développée sous Jitter via Web Socket ainsi qu'une œuvre numérique Web évoluant dans une interface 3D grâce à la bibliothèque *three.js* [13].

7. REFERENCES

[1] https://developer.mozilla.org/fr/docs/WebSockets

[2] http://asmjs.org

[3] http://interactjs.io

[4] https://github.com/aterrien/jQuery-Knob

[5] http://p5js.org

[6] https://jquery.com/

[7] http://musinf.univ-st-etienne.fr/pacap.html

[8] http://feever.fr/

[9] https://github.com/olilarkin/wsserver

[10] Thomas Cipierre, Laurent Pottier, « Développement multiplateforme et temps réel d'un clavecin synthetisé par modèles physiques avec Faust » http://jim2015.oicrm.org/actes/JIM15_Cipierre_T_et_al.pdf

[11] http://lac.linuxaudio.org/2012/papers/25.pdf

[12] http://www.mshparisnord.fr/hoalibrary/

[13] http://threejs.org