Bot IRC -> MIDI

aka **pouetBot**

(Page créée le 15 mai 2020, en cours de rédaction)

Comment transformer les profondes discussions d'IRC en « chœur de l'aube » ?
→ avec un bot logiciel qui transforme les messages écrits en notes MIDI.

Le bot en php écoute les discussions d'un canal IRC et transforme ce que chaque participant·e écrit en notes MIDI qui sont envoyées à un synthé logiciel (qysnth) qui génère les sons, ~~le résultat est streamé vers un serveur icecast~~, le résultat est streamé en utilisant un serveur WebRTC, généreusement public, dont la latence est plus adapté à une transcription en temps réel.
Tout ça forme une usine à gaz moderne et performante, mais qui pourrait sûrement trouver une forme différente plus optimale.

En pratique

Tous les caractères alphanumériques définissent une note.
Tous les caractères accentués et de ponctuation définissent un hit d'instrument de percussion.
Commencer une ligne par un chiffre (entre 1 et 9) définit le nombre de répétitions de la ligne.

Les commandes suivantes sont utilisables :

- **!change** : changement d'instrument au hasard
- **!instrument [NUMERO]** : changement d'instrument, NUMERO est compris entre 1 et 128, selon le standard General MIDI 1
- **!panique** : coupe toutes les notes (envoie ALL NOTES OFF et ALL SOUNDS OFF sur les 16 canaux MIDI)

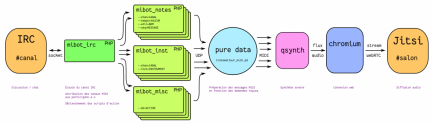
D'autres commandes sont utilisables pour le débuguage

Mise en pratique

Une fois les différents logiciels démarrés ainsi que la communication entre eux, démarrer une videoconférence webRTC sur une généreuse instance jitsi (par exemple, parmi cette liste d'instance Jitsi compilée par Framasoft), en coupant la caméra et en choisissant comme source sonore le “monitor of audio interne stéréo analogique”. Ce choix de source sonore vaut pour un ordinateur sous linux debian, il faudra l'adapter dans d'autres cas. Ce flux sonore correspond au “son qui sort de l'ordinateur”, toutes applications confondues.

Usine à gaz

Le script mibot_irc est en écoute sur le canal IRC choisi, chaque nouvelle contribution est analysée pour y chercher des commandes ou interpréter ce qui est écrit.
Selon les cas, des scripts sont appelés pour envoyer les messages adaptés à pure data qui les transformera en commande MIDI.
Au démarrage, ces scripts ouvrent un socket UDP et envoient les messages nécessaires, dans le cas d'un message à multiples caractères (la plupart des messages sont de ce type), il est envoyé, caractère après caractère au tempo voulu.
Une fois l'action terminé le socket est fermé et le script s'arrête et “se détruit”.

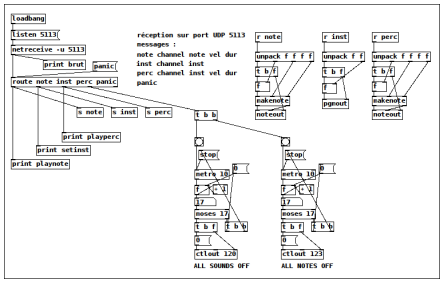


Démarrage des logiciel et connexions MIDI

```
qsynth &      # démarrer qsynth
sleep 5       # nécessaire en script pour laisser du temps à qsynth pour s'initialiser
puredata -open ./transmetteur_midi.pd -alsamidi -midiindev 1 -midioutdev 1 &
aconnect -i   # liste d'adresse des émetteurs MIDI
aconnect -o   # liste d'adresse des récepteurs MIDI
aconnect 128:1 129:0  # connecter la sortie MIDI de pure data à l'entrée MIDI de qsynth, à adapter selon le résultat des commandes
précédentes
# aseqdump -p 128:1   # si on veut afficher les messages MIDI qui sortent de pure data
php ./mibot_irc_001.php
```

Réception dans pure data et envoi des messages MIDI

transmetteur_midi.pd



Scripts

Les différents scripts sont téléchargeables dans ce dépôt : https://github.com/emoc/pouetBot

Script de test

Script de base pour envoyer des messages aléatoires en UDP à pure data

mibot_test.php (cliquer pour afficher le code)

mibot_test.php

```
#!/usr/bin/php -q
<?php
/*
  Messages envoyés
    note {channel} {note} {vel} {dur}    : jouer la {note} sur ce {channel} MIDI avec une vélocité {vel} et une durée {dur}
    inst {channel} {inst}                : changer l'instrument du {channel} par {inst}
    perc {channel} {inst} {vel} {dur}    : jouer une percussion sur le {channel} ...

    channel : entre 1 et 16  (canal MIDI)
    note    : entre 0 et 127 (note à jouer)
    vel     : entre 0 et 127 (vélocité)
    dur     : entier (durée)

    note : le canal MIDI de percussion étant toujours le 10 (sauf exception...), il est fixé, c'est la seule différence entre perc
et note...
*/

// Créer un socket sur le port UDP 5113 de la machine locale
$fp = stream_socket_client("udp://127.0.0.1:5113", $errno, $errstr);

$compteur = 1;

if (!$fp) {
    echo "ERREUR : $errno - $errstr<br />\n";
} else {

  while (1) {
    $rnd = rand(0,99);
    $message = "";
    // Créer un message parmi les 3 types afin de tester que tout fonctionne
    if ($rnd < 33) {
      $message = "note " . rand(1,16) . " " . rand(0,127) . " " . rand(0, 127) . " " . (rand(20, 100) * 10) . ";\n";
    } else if ($rnd >= 33 && $rnd < 66) {
      $message = "inst " . rand(1,16) . " " . rand(0,127) . ";\n";
    } else {
      $message = "perc 10 " . rand(0,127) . " " . rand(0, 127) . " " . (rand(20, 100) * 10) . ";\n";
    }
    echo "envoi # $compteur : $message";
```

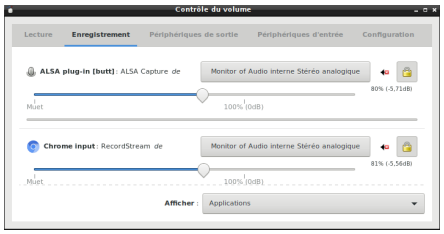
```
            fwrite($fp, $message);   // Envoyer le message sur le socket
            sleep(1);                // Petite pause d'1 seconde
            $compteur ++;
        }
        //fclose($fp);
    }
    ?>
```

Mise en place du stream

Démarrer Butt (dans cette configuration, Butt est utilisé pour enregistrer l'audio produit localement, en parallèle de la diffusion en stream.)
Démarrer Chromium

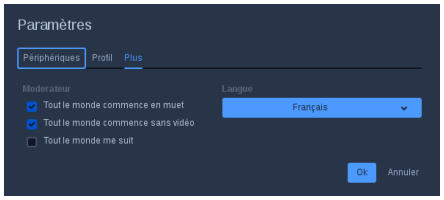
Réglages pulseaudio

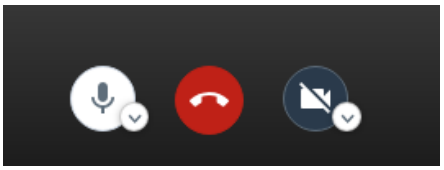
Dans l'onglet enregistrement, choisir “Monitor of Audio interne stéréo analogique” pour que le son utilisé ne soit pas celui du micro, mais le mix des sons produits.



Réglages jitsi

Pour l'émetteur, démarrer jitsi sans caméra et choisir que les participant·e·s démarrent aussi sans caméra ni micro.
Régler dans les paramètres de son “Audio interne stéréo analogique”





Ça fonctionne avec très peu de latence MAIS beaucoup de compression… Il faudrait trouver une alternative

Bugs

Buffer overflow! Le buffer du socket est dépassé par la taille de certains messages, les morceaux de messages restant seront interprêtés aussi, de manière imprévisible!
En particulier pour la gestion des 16 slots correspondant à chaque canal MIDI
Certains messages d'erreur du serveur IRC, déclenchés par ces dépassements sont invisibles sur le canal, mais sont interprêtés.
Beaucoup de notes restent en suspend… La fonction !panique est très utile!

Améliorations possibles

Corriger les bugs!

Une commande pour changer le tempo ? du swing ?

Trouver une alternative utilisat·eur·rice-conviviale avec une meilleure qualité sonore, quelques pistes :

- mumble, https://fr.wikipedia.org/wiki/Mumble , c'est avant tout pour la voix mais on peut ptet régler les bitrates côté serveur et client et choisir le codec

    - pour mumble voir aussi https://wiki.mumble.info/wiki/3rd_Party_Applications
    - et un client web https://github.com/Johni0702/mumble-web (bien pour le côté convivial!)
- ninjam : https://www.cockos.com/ninjam/
- jamulus : http://llcon.sourceforge.net/ + https://ressources.labomedia.org/musique_en_reseau_jamulus
- jamtaba : https://jamtaba-music-web-site.appspot.com/
- jacktrip : https://ccrma.stanford.edu/groups/soundwire/software/jacktrip/index.html
- serveur webRTC bidouillé

Un MOOC du CCRMA sur le sujet (en anglais) :
https://online.stanford.edu/courses/sohs-music0001-online-jamming-and-concert-technology

Ressources

- La norme MIDI en détail et en français : https://www.sonelec-musique.com/electronique_theorie_midi_norme.html
- Wikikirc ou la sonification de Wikipedia : projet concurrent datant du début XXIe siècle, mené par Labomedia, déjà à la pointe du futur.
- php-irc-bot (github) : script minimal qui a servi de base pour ce bot, merci à Hans Koch.
- Qsynth : http://linuxmao.org/QSynth
- Fluidsynth : http://linuxmao.org/FluidSynth

Le coin des control freaks

(Utile aussi pour celleux qui veulent connaître le truc dans les tours de magie)

Percussions

carac.	MIDI	instrument	carac.	MIDI	instrument	carac.	MIDI	instrument
,	35	Bass Drum 2	.	49	Crash Cymbal 1	î	63	Open High Conga
;	36	Bass Drum 1	#	50	High Tom 1	ï	64	Low Conga
-	37	Side Stick	%	51	Ride Cymbal 1	|	65	High Timbale
!	38	Snare Drum 1	é	52	Chinese Cymbal	ù	67	High Agogo
*	39	Hand Clap	è	53	Ride Bell	ê	68	Low Agogo
?	40	Snare Drum 2	ç	54	Tambourine	ü	69	Cabasa
:	41	Low Tom 2	à	55	Splash Cymbal	ô	70	Maracas
/	42	Closed Hi-hat	$	56	Cowbell	ö	71	Short Whistle
&	43	Low Tom 1	@	57	Crash Cymbal 2	Ô	72	Long Whistle
(	44	Pedal Hi-hat	]	58	Vibra Slap	Û	73	Short Guiro
_	45	Mid Tom 2	ä	59	Ride Cymbal 2	Î	74	Long Guiro
)	46	Open Hi-hat	â	60	High Bongo	Ö	75	Claves
+	47	Mid Tom 1	}	61	Low Bongo	ï	76	High Wood Block
=	48	High Tom 2	{	62	Mute High Conga	Ü	77	Low Wood Block

Notes

carac.	MIDI	octave	note	carac.	MIDI	octave	note	carac.	MIDI	octave	note	carac.	MIDI	octave	note
A	21	1	La	Q	48	4	Do	g	76	6	Mi	w	103	8	Sol
B	23	1	Si	R	50	4	Ré	h	77	6	Fa	x	105	8	La
C	24	2	Do	S	52	4	Mi	i	79	6	Sol	y	107	8	Si
D	26	2	Ré	T	53	4	Fa	j	81	6	La	z	108	8	Do
E	28	2	Mi	U	55	4	Sol	k	83	6	Si	0	110	9	Ré
F	29	2	Fa	V	57	4	La	l	84	7	Do	1	112	9	Mi
G	31	2	Sol	W	59	4	Si	m	86	7	Ré	2	113	9	Fa
H	33	2	La	X	60	5	Do	n	88	7	Mi	3	115	9	Sol
I	35	2	Si	Y	62	5	Ré	o	89	7	Fa	4	117	9	La
J	36	3	Do	Z	64	5	Mi	p	91	7	Sol	5	119	9	Si
K	38	3	Ré	a	65	5	Fa	q	93	7	La	6	120	10	Do
L	40	3	Mi	b	67	5	Sol	r	95	7	Si	7	122	10	Ré
M	41	3	Fa	c	69	5	La	s	96	8	Do	8	124	10	Mi

carac.	MIDI	octave	note	carac.	MIDI	octave	note	carac.	MIDI	octave	note	carac.	MIDI	octave	note
N	43	3	Sol	d	71	5	Si	t	98	8	Ré	9	125	10	Fa
O	45	3	La	e	72	6	Do	u	100	8	Mi				
P	47	3	Si	f	74	6	Ré	v	101	8	Fa				

Cerise sur le gateau

Du midi (d) à la fréquence (f)

$$f = 2^{(d-69)/12} \cdot 440 \text{ Hz}$$