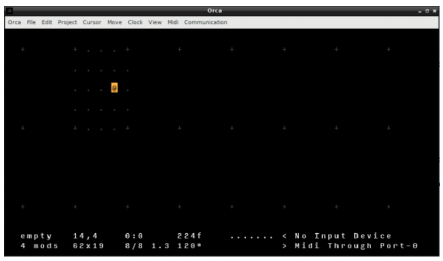
musique, livecoding, vcv-rack, em

Orca

Orca est un langage ésotérique de création de séquenceurs, capable d'envoyer des informations de contrôle en MIDI, OSC ou UDP à des logiciels musicaux ou audiovisuels. Il permet de construire des séquences complexes par un langage spécifique composé d'instructions réduites à une lettre. C'est un environnement de création pour le livecoding audio-visuel, qui ressemble à un croisement entre l'assembleur et le jeu de la vie de Conway

- https://hundredrabbits.itch.io/orca
- https://github.com/hundredrabbits/Orca

Comment l'utiliser



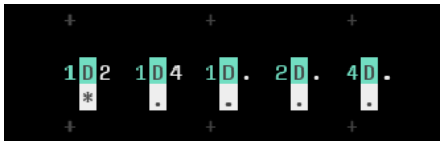
En démarrant Orca, on se retrouve devant une interface minimale composée d'une grille avec des petits points et de quelques lignes de console au bas de cette grille qui affichent des informations : dimensions, position, nombre de frames écoulées, informations sur les instructions.

Sur chaque petit point de cette grille, on peut placer un caractère qui représente une instruction du séquenceur (comme une fonction d'un langage d'un programmation). Chacune de ces instructions peut être modifiée selon plusieurs paramètres représentés par les points autour de la l'instruction. En plaçant le curseur sur ces points de modulation, leur rôle est indiqué sur la console.

Les notes sont représentées selon la notation anglaise.

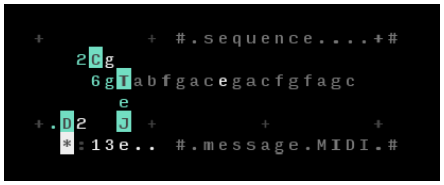


exemple avec l'instruction D (Delay)



L'instruction **D** (comme Delay), déclenche un «bang» quand le modulo de la frame vaut zéro. Ce bang, symbolisé par un astérisque, permet de déclencher d'autres évènements.
En plaçant le curseur sur le D, le nom de la fonction s'affichera sur la console au bas de la fenêtre, en le plaçant à gauche ou à droite du D sur les points colorés, c'est le nom des paramètres qui s'affichera sur la console.

exemple d'une séquence de 16 notes envoyées en MIDI

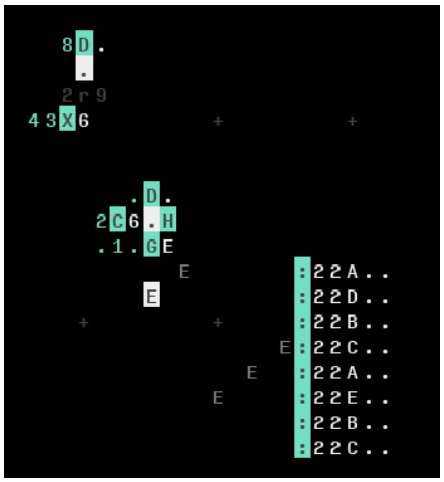


L'instruction **T** permet de définir un motif de x notes, ici il y en a 16 que l'on voit à droite de **T**. Ce nombre est défini à gauche (g), encore plus à gauche on a la position actuelle.

La position actuelle dans le motif est définie par la sortie de l'instruction **C** qui boucle sur 16 valeurs (g, à droite).

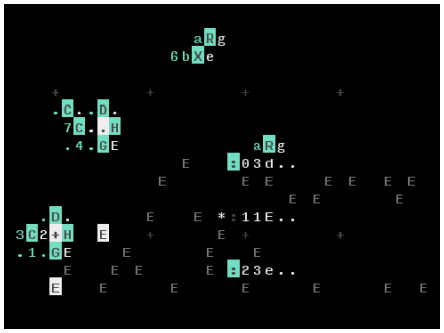
La sortie de **T** est reportée dans un message MIDI (:13x) déclenché par un bang, le message est envoyé sur le canal MIDI 1, et envoie la note du motif sur l'octave 3! **J** sert uniquement à reporter son entrée (en haut) sur sa sortie (en bas).

exemple de séquenceur à 8 pas



Chaque pas envoie une note midi différente sur le même canal (on pourrait aussi varier la vélocité et la durée des notes). Les pas sont lus en une boucle dont la taille varie de temps en temps, c'est le rôle du petit module en haut à gauche de définir le nombre de pas lus!

exemple avec des générateurs de bang



Petites infos utiles

Orca fonctionne en base 36!

Ecriture des notes : C (majuscule) représente un DO , c (minuscule), un DO# (1/2 ton au dessus)

Chaque lettre de l'alphabet correspond à une commande différente, en minuscule elle sera déclenchée sur demande par un «bang», en majuscule à chaque «frame».

CTRL + G : **afficher la liste des instructions**
ESPACE : play/pause sur le déroulement = arrêt du compteur de frame
CTRL + K : activer la ligne de commande, ce qui permet de rentrer ce type de commande

- play
- stop
- run (jouer uniquement une frame)
- bpm:180
- apm:180 : amener le BPM à la valeur demandée
- et bien d'autres : liste complète

CTRL + : : (dé)commenter un bloc après l'avoir sélectionné au clavier ou à la souris
CTRL + I : basculer entre mode caractère unique / mode texte (pour écrire des commentaires, par exemple)

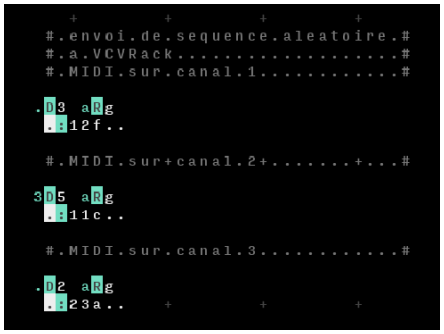
Installation sur Linux

Télécharger le fichier depuis https://hundredrabbits.itch.io/orca

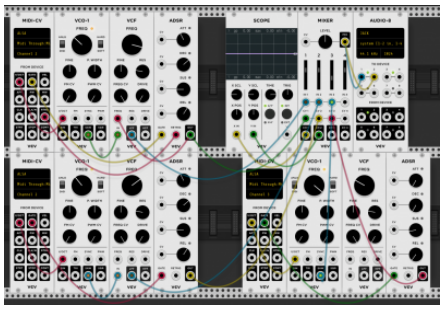
```
# debian 9.5 stretch @ kirin
# décompresser le fichier téléchargé, ici dans /home/emoc/orca-7.1.12
cd /home/emoc/orca-7.1.12
sudo chown root:root ./chrome-sandbox # il est nécessaire que root soit propriétaire de ce fichier
sudo chmod -R 4755 ./chrome-sandbox   # et qu'il ait tous les droits nécessaires
```

Configuration Orca + VCV Rack sur Linux

Code pour orca



patch VCVRack



exemple_orca_vcv.vcv

Pour cet exemple Orca est utilisé avec le “Midi Through Port-0” pour envoyer ses messages MIDI.
Une fois Orca démarré avec le code ci-dessus, on peut vérifier que les messages sont bien transmis

```
aseqdump -l         # affiche la liste des ports MIDI disponibles, 14:0 est le "Midi Through Port-0" utilisé par Orca
aseqdump -p 14:0    # affiche les infos MIDI qui transitent sur ce port (trouvé précédemment)
```

Dans VCVRack, régler le module MIDI-CV avec les caractéristiques suivantes

- MIDI Driver : ALSA
- MIDI Device : Midi Through Port-0 14:0
- MIDI Channel : 1, 2, 3 selon l'instrument

Ressources

tutoriel interactif bien bien : https://metasyn.srht.site/learn-orca/
D'autres exemples : https://git.sr.ht/~rabbits/orca-examples
vidéo de présentation en français : https://www.youtube.com/watch?v=ihFidWCWu9Y
vidéo d'initiation en anglais : https://www.youtube.com/watch?v=RaI_TuISSJE
fil de discussion sur le forum lines : https://llllllll.co/t/orca-livecoding-tool/17689/
interview de Devine Lu Linvega, auteur d'Orca : https://futureofcoding.org/episodes/045

«c'est encore en vidéo dont on se rend le mieux compte du truc» : tunnellers * dunes * pong * convey * delay with animated velocity * dub hotel lounge * armadillidiids * overload * time crystal * invaders * invaders * 15092019 * cascade modulo sequencing * shift registrer melodies * glitch

Article extrait de : http://www.lesporteslogiques.net/wiki/ - **WIKI Les Portes Logiques**
Adresse : **http://www.lesporteslogiques.net/wiki/ressource/logiciel/orca?rev=1698589080**
Article mis à jour: **2023/10/29 15:18**