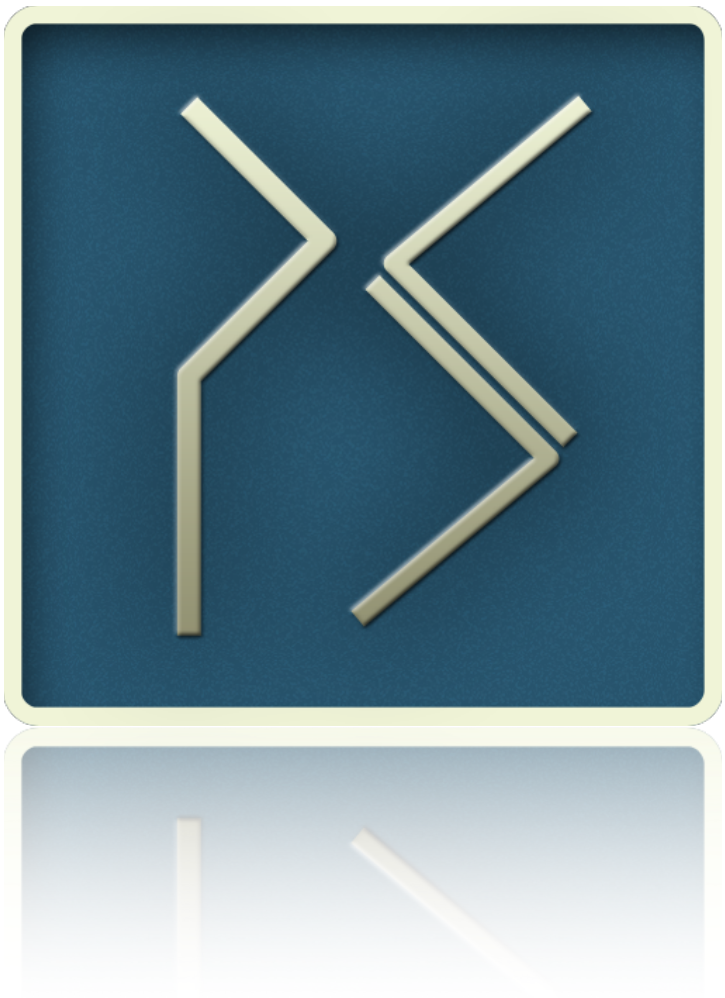


Pyo Synth - v0.1.2b

Manuel d'utilisation / User Manual

Table des matières

Manuel en français

1. Présentation de Pyo Synth

Pyo Synth permet de facilement contrôler ses scripts pyo à l'aide d'un contrôleur midi. Le principe est simple : chaque boîte de l'interface peut être assignée à un bouton de votre clavier midi pour ensuite contrôler n'importe quel objet audio du script. Pyo Synth met également plusieurs paramètres de votre contrôleur midi à votre disposition comme le «pitch bend» et la «modulation wheel» ainsi que tout le paramétrage du serveur de pyo. Quelques règles doivent être suivies pour permettre au programme de fonctionner correctement alors assurez-vous de lire au minimum l'introduction rapide.

2. Introduction rapide

Dans cette introduction, vous serez guidé au travers du fonctionnement de base de Pyo Synth avec un court tutoriel.

> Note : vérifiez que votre clavier midi soit bien branché avant de démarrer Pyo Synth, sans quoi le programme ne le reconnaîtra pas. Ceci est également valide pour la carte de son.

Pour débuter, il faut configurer le serveur de pyo. Cliquez sur le bouton rectangulaire en haut à droite de l'interface.

Ensuite, il faut choisir un script audio à exécuter. Avec votre installation de Pyo Synth vous avez quelques exemples de scripts dans le dossier «Pyo Synth Examples» qui se situe dans vos Documents. Choisissez «HarmoSynth.py». Maintenant, cliquez sur le bouton «Run» (⌘+R). Cette fonction exécute le script et démarre le serveur audio. À ce point, vous devriez avoir du son en jouant sur le clavier. Si vous utilisez le clavier d'ordinateur, n'oubliez pas de l'activer dans le menu principal ou en utilisant le raccourci clavier ⌘+K.

Maintenant, on peut commencer à faire des liens entre le contrôleur midi et le script. Cliquez sur la première boîte intitulée «Unused». La «Patch Window» apparaît et vous montre tous les objets audio présent dans le script et, en cliquant sur une flèche, vous verrez apparaître tous les les paramètres contrôlables de cet objet. Commençons par sélectionner l'attribut «transpo» de l'objet nommé «harmo - Harmonizer». Le paramètre est maintenant bel et bien lié à cette boîte, mais il reste encore à lier la boîte à un bouton de votre clavier midi. Pour ce faire, faites ⌘+Clique sur la boîte, touchez le contrôle que vous voulez assigner et ensuite cliquez à nouveau sur la boîte pour quitter le mode «Midi Learn».

Vous devez maintenant configurer la boîte de sorte que son ambitus de valeur corresponde bien à l'effet recherché (voir section 4). Dans ce cas, un facteur de transposition de -12 à 12 est désirable. Pour se faire, faites Shift+Clique sur la boîte et vous verrez les paramètres de la boîte s'afficher. Vous pouvez ici changer le titre du contrôle, son minimum et son maximum, activer/désactiver les modes «exponentiel», «portamento» ou «entiers seulement». Ici, il serait intéressant d'activer le mode «entiers seulement». Quand vous avez terminé avec la boîte, faites «Enter».

Voilà, vous avez vu l'essentiel de l'interface. Avant d'écrire votre propre script pour Pyo Synth, lisez la section sur la «Création et Édition de scripts» dans la section 7.

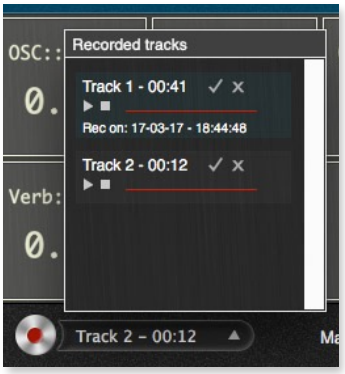
3. Interface principal

1. Boutons «Open» et «Save». Le bouton de sauvegarde écrit directement dans le script tous les paramètres des boîtes actives y comprit les valeurs courantes. Au moment de rouvrir le script, Pyo Synth entre en mode «Matching Preset» vous permettant d'amener chaque boîte à la valeur sauvegardée pour retrouver la sonorité d'origine.
2. Le métronome peut être démarré et arrêté avec son bouton on/off, mais même lorsqu'il est à «off» il continu de fonctionner en arrière plan. Vous pouvez donc vous en servir pour rythmer certains effets sans avoir à l'entendre. Pour récupérer le stream écrivez : pyosynth['click']. Pour changer le tempo, double-cliquer sur la valeur du tempo.
3. L'enveloppe ADSR peut être éditée directement avec la souris ou encore avec un contrôle midi. Faites ⌘+Clique sur un bouton pour entrer en mode «Midi Learn», touchez le contrôle à assigner, faites encore ⌘+Clique pour désactiver le mode «Midi Learn». Vous pouvez maintenant alterner entre le mode midi et souris en double-cliquant sur le bouton. Noter que le texte devient vert en mode midi.
4. La section de paramétrage du serveur comprend une zone d'infos rapide et le bouton pour accéder aux paramètres.
5. Section d'enregistrement avec gestion des pistes. (Voir section 5)
6. Voyant de saturation. (Peak) Ce voyant s'allumera si l'objet 'mix' du script sature. Il se peut, par contre, que le Vu mètre n'indique pas de saturation.

4. Paramétrage d'une ParamBox

Pour faire apparaître cette fenêtre, faites Shift+Clique sur la boîte voulue alors qu'elle est active. Ici, on peut modifier le nom, la valeur minimum et maximum, la précision d'affichage, le mode portamento, le mode exponentiel, le mode entiers seulement (tronque les décimales) ainsi que l'affichage des valeurs en dB.

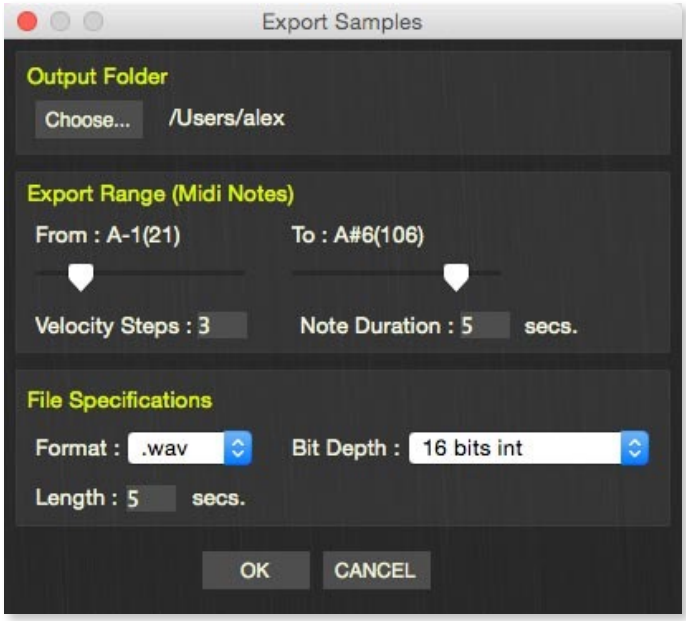
5. Fenêtre des pistes enregistrées



L'enregistrement de piste utilise l'objet mix du script, tout comme le voyant de «clipping». Il est important de gérer les niveaux avant le master, car il n'influ pas sur le niveau du signal enregistré (Il est donc possible que le voyant de clipping s'allume même si le Vu mètre n'indique pas de saturation). Les pistes ont une durée maximum de 5 minutes pour l'instant (une préférence sera ajoutée dans le futur). Les pistes enregitrées pendant la session actuelle seront affichées en gris, tandis que les pistes récupérées d'une session précédente seront affichées en bleu.

6. Export du script en samples

Cette option, accessible par le menu intitulé «Menu» ou avec ⌘+E, permet d'exporter un script en samples note par note dans l'ambitus défini. Vous avez également le choix des étapes de vélocité si le script répond à l'intensité du jeu. Par exemple, en spécifiant 3 étapes de vélocité, Pyo Synth va générer toutes les notes aux vélocités 42, 84 et 127. «Note duration» définit la durée de la note midi tandis que «Length» définit la durée du fichier audio final. (Utile s'il y a de la réverb ou du délai) Notez que si vous utilisez un long reverb, il se peut qu'il y ait une partie du signal au début du sample suivant. Il est du devoir de l'utilisateur de s'assurer que la longueur du fichier soit suffisamment longue pour laisser le signal atteindre le silence.



7. Création et Édition de scripts

7.1. Règles de base

Pour que votre script pyo fonctionne avec Pyo Synth, vous devez respectez deux règles de base:

1. Le dernier objet de la chaîne audio doit s'appeler 'mix' et doit rassembler tous les objets audio que vous désirez envoyer aux haut-parleurs.
2. L'objet 'mix' doit être le seul qui soit envoyé aux haut-parleurs. (c-à-d le seul objet sur lequel vous appelez la méthode 'out()').

L'objet mix est celui qui sera utilisé pour la fonction d'enregistrement ainsi que pour le voyant de clipping (saturation), il se peut donc que le signal sature lors d'un enregistrement, et ce, même si le Vu mètre n'indique pas de clip puisque le Master Volume est appliqué en dernier.

7.2. Streams

Les streams associés au clavier midi peuvent être utilisés de la façon suivante : pyosynth['nom_du_paramètre']. Voici la liste complète des streams mis à votre disposition :

1. pitch : fréquences en Hz des notes jouées
2. amp : enveloppe d'amplitude de la note jouée (tient compte de l'enveloppe ADSR)
3. vel : stream de vélocité des notes (entre 0 et 1)
4. bend : roulette de pitch bend
5. mod : roulette de modulation
6. noteon : streams de triggers représentant les événements «note on»
7. noteoff : streams de triggers représentant les événements «note off»
8. freevoice : envoi un trigger quand une voix est libérée. C-à-d quand l'enveloppe d'amplitude atteint 0.
9. c1 à c16 : représente les 16 boîtes de Pyo Synth et permet l'assignation des contrôles du clavier midi directement dans le code.
10. click : stream de triggers du métronome. (Tempo défini par l'interface)

Vous pouvez vous fier aux exemples fournis avec Pyo Synth pour savoir comment utiliser les streams. Il y a également des tutoriels qui détaillent un peu plus le fonctionnement des fonctionalités.

7.3. Méthodes de configuration des boîtes

De plus, des méthodes peuvent être appelées pour configurer les boîtes de Pyo Synth à même le code. Ceci peut être pratique si vous désirez effectuer des changements de contrôle programmatiquement pendant que le script roule.

1. pyosynth.setScale(ctl, min, max)
   I. ctl : numéro du contrôle (de 1 à 16)
   II. min : valeur minimum du contrôle
   III. max : valeur maximum du contrôle

2. pyosynth.setExp(ctl, value)
   I. ctl : numéro du contrôle (de 1 à 16)
   II. value : booléen, active/désactive la courbe exponentielle

3. pyosynth.setPort(ctl, value)
   I. ctl : numéro du contrôle (de 1 à 16)
   II. value : booléen, active/désactive le portamento

4. pyosynth.setFloor(ctl, value)
   I. ctl : numéro du contrôle (de 1 à 16)
   II. value : booléen, active/désactive le mode «entiers seulement»

5. pyosynth.setDisplayDB(ctl, value)
   I. ctl : numéro du contrôle (de 1 à 16)

II. value : booléen, active/désactive le mode d'affichage des valeurs en dB

6. pyosynth.quickPreset(dict)
   I. dict : dictionnaire python contenant comme clés le numéro des contrôles que vous souhaitez modifier, et comme valeur soit : 1. un string représentant le nom de la boîte ou, 2: une liste contenant en première position un string représentant le nom de la boîte, et en deuxième position, la précision d'affichage de la valeur.

7. pyosynth.setTempo(value)
   I. value : int ou float, valeur du tempo en bpm

8. pyosynth.getTempo(sig=True)
   I. sig : booléen, si vrai retourne un objet Sig avec comme valeur le tempo actuel. La valeur varira donc avec les changements de tempo par l'interface.

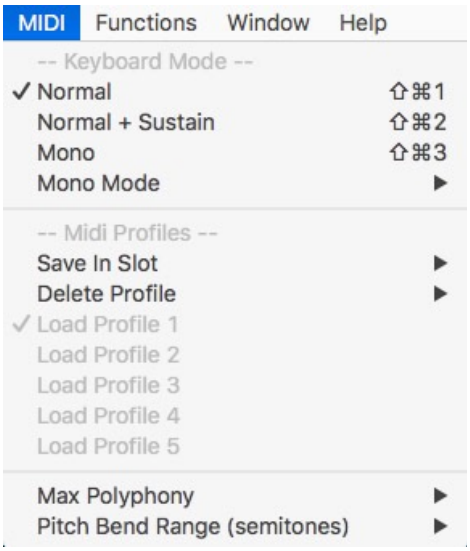
9. pyosynth.getTempoTime(sig=True)
   I. sig : booléen, si vrai retourne un objet Sig avec comme valeur la durée d'un temps (60./tempo) La valeur varira donc avec les changements de tempo par l'interface.

10. pyosynth.getTempoFreq(sig=True)
    I. sig : booléen, si vrai retourne un objet Sig avec comme valeur la fréquence du tempo (1./(60./tempo)) La valeur varira donc avec les changements de tempo par l'interface.

11. pyosynth.getPoly()
    I. retourne la polyphonie maximale actuelle de Pyo Synth

12. pyosynth.setFunctions(flist)
    I. flist : liste contenant soit des callables, des tuples ou un mix des deux. Les tuples doivent être définis comme suis: (callable, string). String étant le titre affiché dans le menu.

8. Menu MIDI



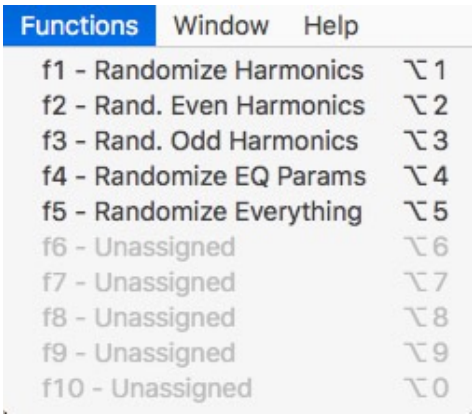
Le menu MIDI vous permet de paramétrer votre clavier MIDI selon vos besoins. Le mode du clavier peut être modifié ici: Normal, Normal + Sustain (permet d'utiliser la pédale de sustain avec un clavier midi), Mono ainsi que ses trois modes (priorité à la note la plus récente, la plus haute ou la plus basse). Les options Normal et Normal + Sustain n'ont pas d'influence sur le Virtual Keyboard puisque le sustain est toujours supporté.

Les Midi Profiles vont sauvegarder les numéros de contrôles assignés aux ParamBox de Pyo Synth et comprend aussi les boutons de l'ADSR. Cela dit, vous pouvez vous servir du mode Midi Learn pour assigner vos contrôler aux ParamBox et ensuite sauvegarder ces numéros dans un Midi Profile. Il est

donc facile de passer d'un clavier MIDI à un autre sans avoir à modifier votre script pyo.

Note: les midi profiles, le mode mono, la polyphonie et le pitch bend range sont sauvegardés dans les préférences de Pyo Synth. Le mode du keyboard n'est pas encore sauvegarder.

9. Menu Functions



Si vous désirez faire des modifications plus complexes dans votre script pendant qu'il s'exécute, passer par le menu functions. Les fonctions seront disponible soit par le menu ou par des raccourcis clavier.

Vous n'avez qu'à définir des fonctions python dans votre script et les passer à la méthode pyosynth.setFunctions(). La méthode peut recevoir une liste de callables, de tuples ou un mix des deux. Le tuples doit contenir le nom du callable suivi d'un string représentant le titre qui s'affichera dans le menu.

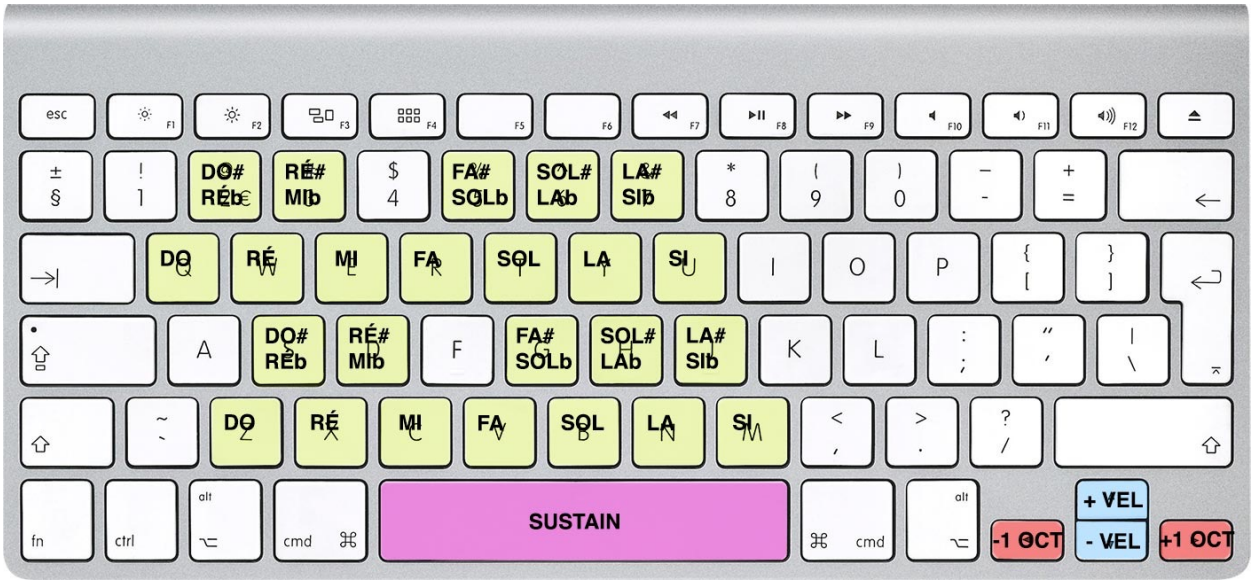
La valeur de retour de la fonction sera affichée dans l'interpréteur python du projet. Il est donc facile de récupérer des données générés aléatoirement par exemple.

10. Utilisation du clavier de l'ordinateur comme clavier midi

Dans la fenêtre de paramétrage du serveur, vous pouvez choisir le clavier d'ordinateur comme source d'entrée midi. Lorsqu'aucun appareil midi est présent, cette option est choisie par défaut.

Une fois le script démarré, allez dans le menu et cliquez sur 'Enable computer keyboard' (⌘+K) pour activer l'option. Fiez-vous à la figure ci-dessous pour connaître le mapping des notes et des fonctions.

Note: le clavier d'ordinateur est assez limité et il y a un maximum de touches qui peuvent être captées en même temps. Certaines combinaisons de touches ne sont alors pas possible. La touche sustain peut aider dans ces cas.



English manual

1. Pyo Synth Introduction

Pyo Synth allows you to easily control pyo scripts with a midi keyboard. Every knob on the MIDI keyboard can be assigned to a ParamBox and then control any audio object in the script. Pyo Synth makes all common MIDI streams available and lets you easily configure the server settings in the server panel. The main feature is the preset capability which saves the state of all ParamBoxes directly in the script. When rerunning your script, you can retrieve the exact sound you tweaked during the last session. Some basic rules need to be followed for the program to work as designed so make sure to read the Quick Start guide at the least.

2. Quick Start Guide

In this guide, we'll cover all the basic features of Pyo Synth by going through a short tutorial.

> Note : make sure your MIDI keyboard and soundcard are plugged before starting Pyo Synth, otherwise the program won't recognize them.

To start off, we need to configure the server. Click on the rectangle button on the top right corner of the application to open the server panel and set everything as you wish.

Now we need a script to execute. Pyo Synth includes a few examples and tutorials that can be found in the «Pyo Synth Examples» folder in your Documents folder. Choose « HarmoSynth.py» and run it by clicking on the Run button next to the name of the script (or use ⌘+R). This function executes the script and starts the audio server. At this point, you should get sound when playing on the keyboard. If you're using the computer keyboard, you need to enable it first in the main menu or using the shortcut ⌘+K.

We can now start to link objects in the script to knobs on your MIDI keyboard. Clicking on the first «Unused» ParamBox will show you the «Patch Window» through which you can create links. It shows you a list of the audio objects in the script and by expanding one in particular (click on the arrow) you get the list of parameters that can be linked to a knob on your controller. Lets choose the parameter «transpo» of the object named «harmo - Harmonizer». The parameter is now linked to the ParamBox but we still need to link a knob to the ParamBox. To achieve this, we use the MIDI learn feature by clicking on the ParamBox while holding the command key on your keyboard (⌘+Click) and then turning the knob we want to assign to it. Command click again to exit MIDI learn.

The next step is to configure the ParamBox so that it sends appropriate values to the audio object. In this case, a range of -12 to 12 would be acceptable. Hold the shift key and click on the ParamBox to show its parameters. In this window you can edit the name, range, precision of the display, activate/deactivate the exponential mode, portamento mode, etc. In this case, let's enable the «Int» mode, which clips all values to the lowest integer value (floor operation). When you're done with the window, just hit «Enter» or click on the box again.

There you go! You've seen enough to start having fun with Pyo Synth. Before writing your own script though, I suggest you read section 7, which explains what little tricks must be used to make Pyo Synth work well. You can also learn a bit more by checking the tutorials.

3. User Interface

1. Open and Save buttons. The Save button will save a preset in the current script. When rerunning the script, it will load all the parameters associated with the ParamBoxes and start the Match Mode. This allows to recall a specific sound.
2. The metronome can be started and stopped using the On/Off button. Note that the metronome is always running in the background, this way you can create rhythmic effects without having to hear the metronome. Rerieve its stream using pyosynth['click'] syntax. To modify the tempo value, double-click on the tempo text.
3. The ADSR envelope can be edited directly with the mouse or with your MIDI keyboard. Start MIDI learn by doing ⌘+Click on it and turn the knob you want to assign to it, then ⌘+Click again to disengage MIDI learn. You can switch between mouse edit or MIDI edit by double-

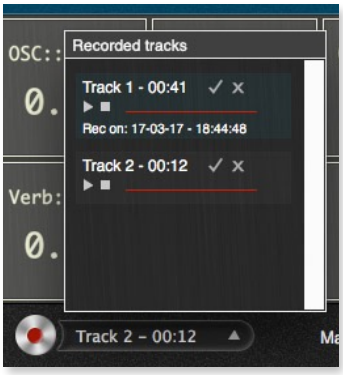
clicking on the ADSR button you wish to control. Note that the text becomes green when a button is controlled through MIDI or during MIDI learn.

4. This section displays information on the server setup, namely, the name of the output device, the sampling rate, buffersize and the number of channels. The rectangle button shows/hides the server setup panel.
5. Recording section and tracks window. (See section 5)
6. Vu meter and clipping light. The clipping light is monitoring the mix object and the Vu meter shows the total output of the server after the master volume is applied. This means that the mix object could clip, thus lighting up the clipping light, while the Vu meter shows no signs of clipping.

4. ParamBox parameters

To display the ParamBox settings window, click on the box while holding shift down (Shift+Click). You can edit the name, minimum, maximum and the precision (only affects the displayed value). You can also enable/disable portamento (0.05s by default), exponential mode (practical for frequencies), the integer only mode (floor operation) and the display values as dB option. All the information in this window is saved in the preset.

5. Recorded tracks window

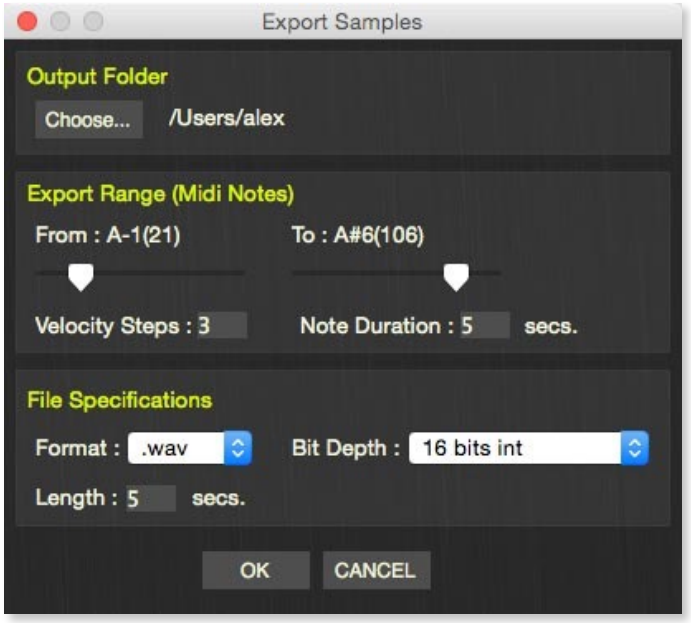


This module uses the mix object in your pyo script to record. Tracks have a maximum of 5 minutes recording time for now (this will become available as an option in the future). Tracks recorded in the current session will be shown in gray, while tracks recovered from a previous session will be shown in blue with additional information.

6. Exporting a script as samples

This option, available through the main menu or by pressing ⌘+E, allows you to export your current script in separate samples using the defined parameters. You can choose a range of midi notes to export and in how many velocity steps (if your script reacts to velocity). For example, defining 3 velocity steps would render velocities 42, 84 and 127. «Note duration» represents the length of the sustained note while the «Length» defined in the file specifications is the actual length of the resulting sound file. This allows you to account for long reverb tails,

since this is not handled by the program. The tail of the previous note could be rendered in the next if the length of the sound file wasn't long enough.



7. Creating and editing scripts

7.1. Basic rules

In order for your script to properly work with Pyo Synth, you need to follow those two simple rules:

1. The last object of the audio processing chain must be called 'mix' and be the actual mix down of all objects in the script.
2. The 'mix' object must be the only one sent to the speakers. (That is, the only one on which to call the out() method)

This is because the object is used in the recording module and in the clipping light module.

7.2. Streams

The streams are associated to the MIDI keyboard and can be retrieved with the following syntax: pyosynth['name_of_the_stream']. Here is the complete list of streams available:

1. pitch : frequency in Hz of the notes being played
2. amp : amplitude envelope as defined by the ADSR knobs
3. vel : velocity stream (values range between 0 and 1)
4. bend : pitch bend wheel
5. mod : modulation wheel
6. noteon : trigger stream representing note on events
7. noteoff : trigger stream representing note off events
8. freevoice : sends a trigger when a voice is freed. That is, when the amplitude of the envelope reaches 0.

9. c1 à c16 : represents the 16 controls available in Pyo Synth. Allows for direct assignation through code
10. click : trigger stream sent by the metronome at the tempo defined through the interface

I suggest you check out the examples and tutorials if you want more info on streams and see them in action.

7.3. User available methods

Some methods are made available to you if you wish to setup the ParamBoxes directly in the code. Combined with the functions menu, you could create multiple «pages» of controls and switch between them.

1. pyosynth.setScale(ctl, min, max)
   I. ctl : number of the control (from 1 to 16)
   II. min : minimum value for the control
   III. max : maximum value for the control

2. pyosynth.setExp(ctl, value)
   I. ctl : number of the control (from 1 to 16)
   II. value : boolean, activate/deactivate exponential mode

3. pyosynth.setPort(ctl, value)
   I. ctl : number of the control (from 1 to 16)
   II. value : boolean, activate/deactivate portamento

4. pyosynth.setFloor(ctl, value)
   I. ctl : number of the control (from 1 to 16)
   II. value : boolean, activate/deactivate the integer only mode (floor function)

5. pyosynth.setDisplayDB(ctl, value)
   I. ctl : number of the control (from 1 to 16)
   II. value : booleann, activate/deactivate the display of dB values

6. pyosynth.quickPreset(dict)
   I. dict : python dictionary in which the keys represent the control number, and the values either 1) a string representing the name of the ParamBox or 2) a list containning the name of the ParamBox followed by the precision of the displayed value.

7. pyosynth.setTempo(value)
   I. value : int or float, the tempo value in bpm

8. pyosynth.getTempo(sig=True)
   I. sig : boolean, when True, returns a Sig object with the current tempo as a value. The value will follow changes made through the UI.

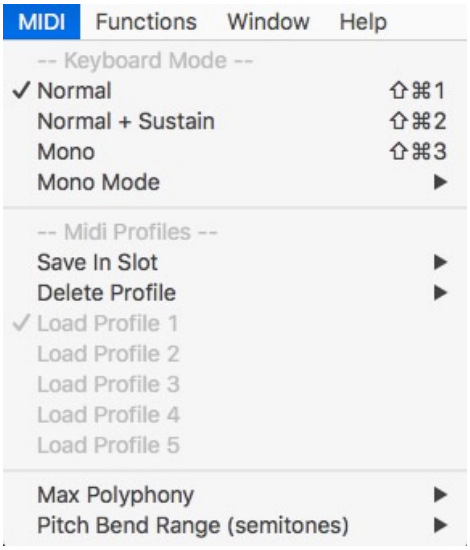
9. pyosynth.getTempoTime(sig=True)
   I. sig : booléan, when True, return a Sig object with the length of a beat (60./tempo) as a value. The value will follow changes made through the UI.

10. pyosynth.getTempoFreq(sig=True)
    I. sig : boolean, when True, returns a Sig object with the frequency of the beats as a value. (1. / (60./tempo)) The value will follow changes made through the UI.

11. pyosynth.getPoly()
    I. returns the maximum polyphony

12. pyosynth.setFunctions(flist)
    I. flist : list containing either callables, tuples or a mix of both. The tuples must be defined as follow: (callable, string). String being the name of the function displayed in the menu.

8. MIDI menu

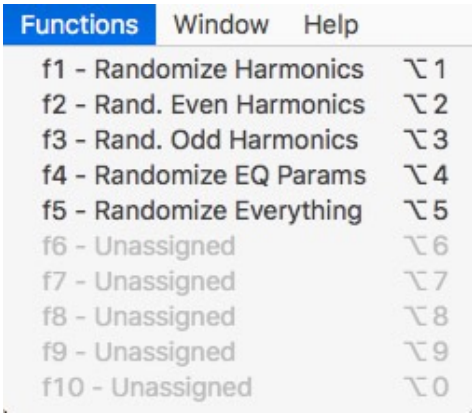


This menu is for setuping your MIDI keyboard. As you can see in the screenshot, there are three keyboard modes and also options to change the behavior of the mono mode (recent note, lowest note and highest note priority). The normal mode does not support the sustain pedal by default so a Normal + Sustain mode was created. Note that the Virtual Keyboard is always using the sustain option.

The Midi Profiles section is where you can save all the current setup for your keyboard. It stores the control numbers for all ParamBoxes as well as for the ADSR knobs. This way you can easily switch between MIDI keyboards without having to go through MIDI learn all over again.

Note: midi profiles, the mono mode, polyphony and pitch bend range are saved in Pyo Synth preferences. The keyboard mode is not yet saved anywhere.

9. Menu Functions



If you wish to dynamically call code during script execution, the Funtions menu can help you. These funtions are available through the menu or through keyboard shortcuts.

After defining the functions in your script, pass them to the pyosynth.setFunctions() method. It accepts either a list of callables, tuples or a mix of both. The tuple has to be as follows: (callable, string). The string represents the name of the function displayed in the menu.

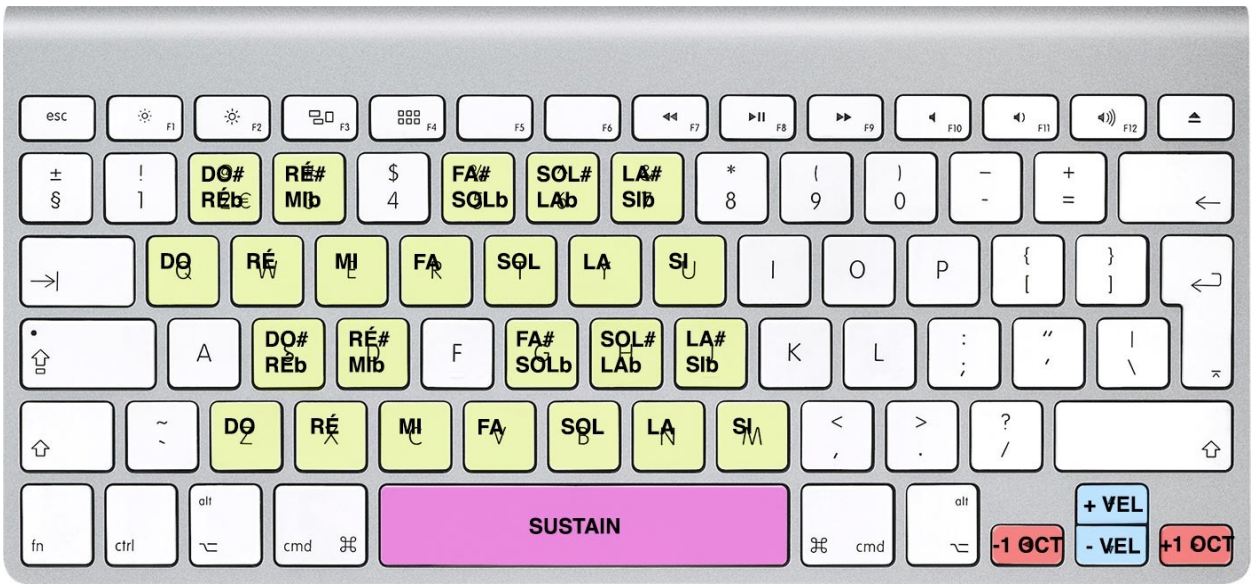
The return value of the functions will be displayed in the

python terminal. This makes it easy to print out randomly generated values that you may want to write down, for example.

10. Using the computer keyboard as a MIDI keyboard

In the server setup panel you can choose to use the computer keyboard in the MIDI input menu. When no other MIDI device is connected, it will be selected by default.

Once the script has been started, go in the main menu and click on «Enable computer keyboard» or use the shortcut (⌘+K). Refer to the image below to know the actual mapping of notes and functions.



Note: the computer keyboard is rather limited in the number of keys that can be pressed at once. This means that certain combinations of keys are impossible. Try using the sustain key to help workaround the issue.