PURE DATA : initiation

Jérôme Abel
http://impala.utopia.free.fr

url : http://impala.utopia.free.fr/pd/patchs/PureData_Initiation_fr.pdf (2Mo)

Document de synthèse réalisé pour deux ateliers d'initiation niveau 1 et 2 à Mains d'Oeuvres, Paris, dans le cadre des formations proposées par l'association Art Sensitif, le 23 et 30 septembre 2006.

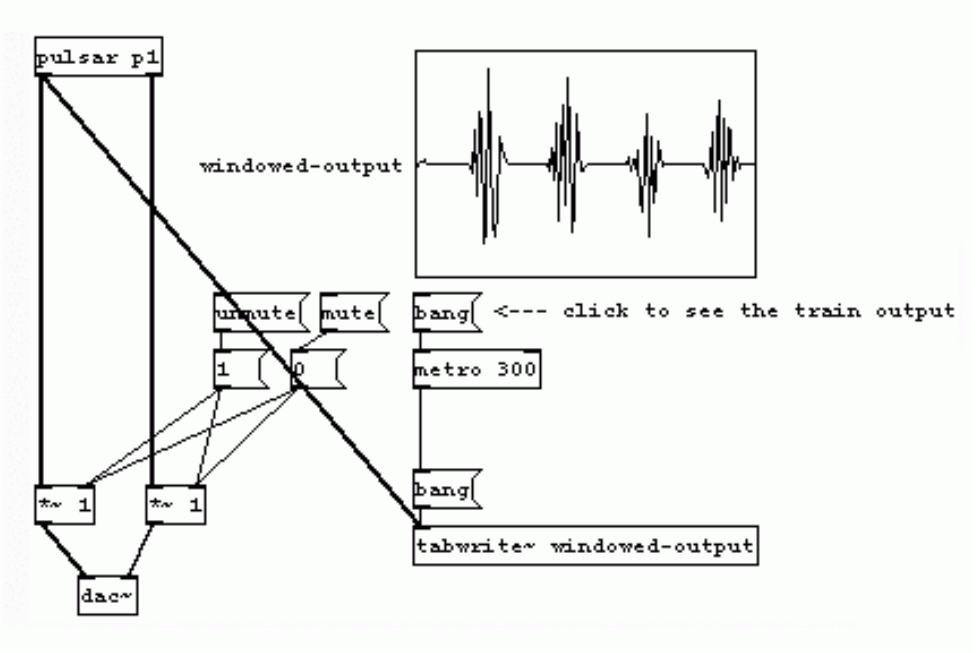
Le document a par la suite été revu et corrigé en février, mars 2007. La lecture de la deuxième partie, plus pratique, pourra s'accompagner du dossier compressé comprenant un atelier complet d'initiation composé de patchs pure data : http://impala.utopia.free.fr/pd/patchs/AtelierPd_2007.02_fr.zip (3,6 Mo)

Première partie

Qu'est-ce que c'est ?
Autour de pd
A quoi ça sert ?
Références
Entrées / Sorties
Ressources Entrées / Sorties
D'où ça sort ?
Qui en est le créateur ?
Qui sont ses fidèles ?
Quelle licence ?
Comment se le procurer ?
Différences avec un langage de programmation ?
Multi-plateforme ?
Différences avec MaxMSP ?
Où trouver des ressources ?
En France

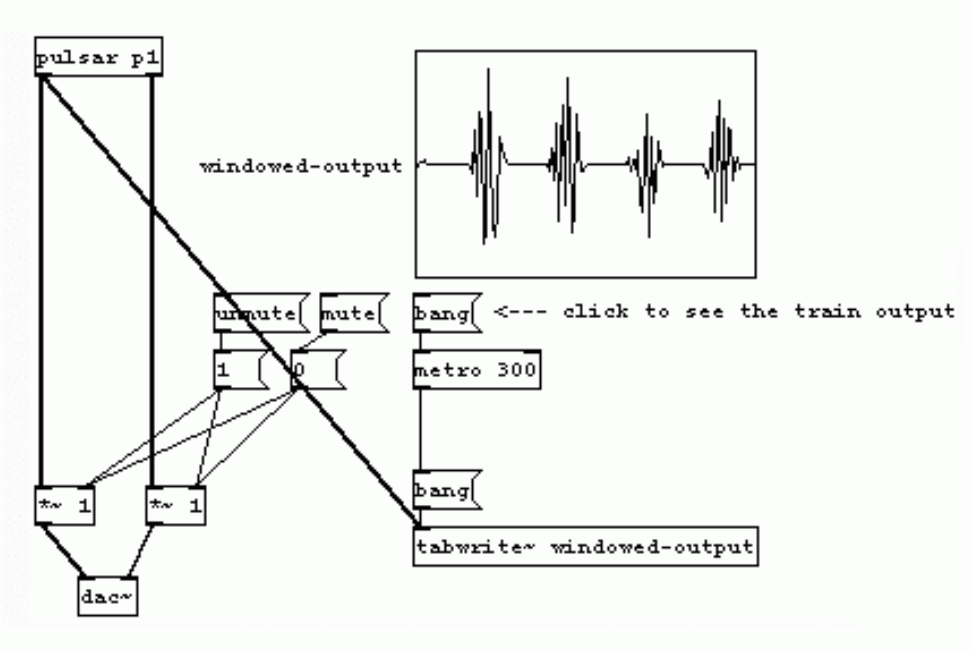
Qu'est-ce que c'est ?

A. Un logiciel pour faire des organigrammes
B. Un logiciel de montages électroniques ou de plomberie
C. Un environnement de programmation graphique en temps réel pour la création musicale et multimédia



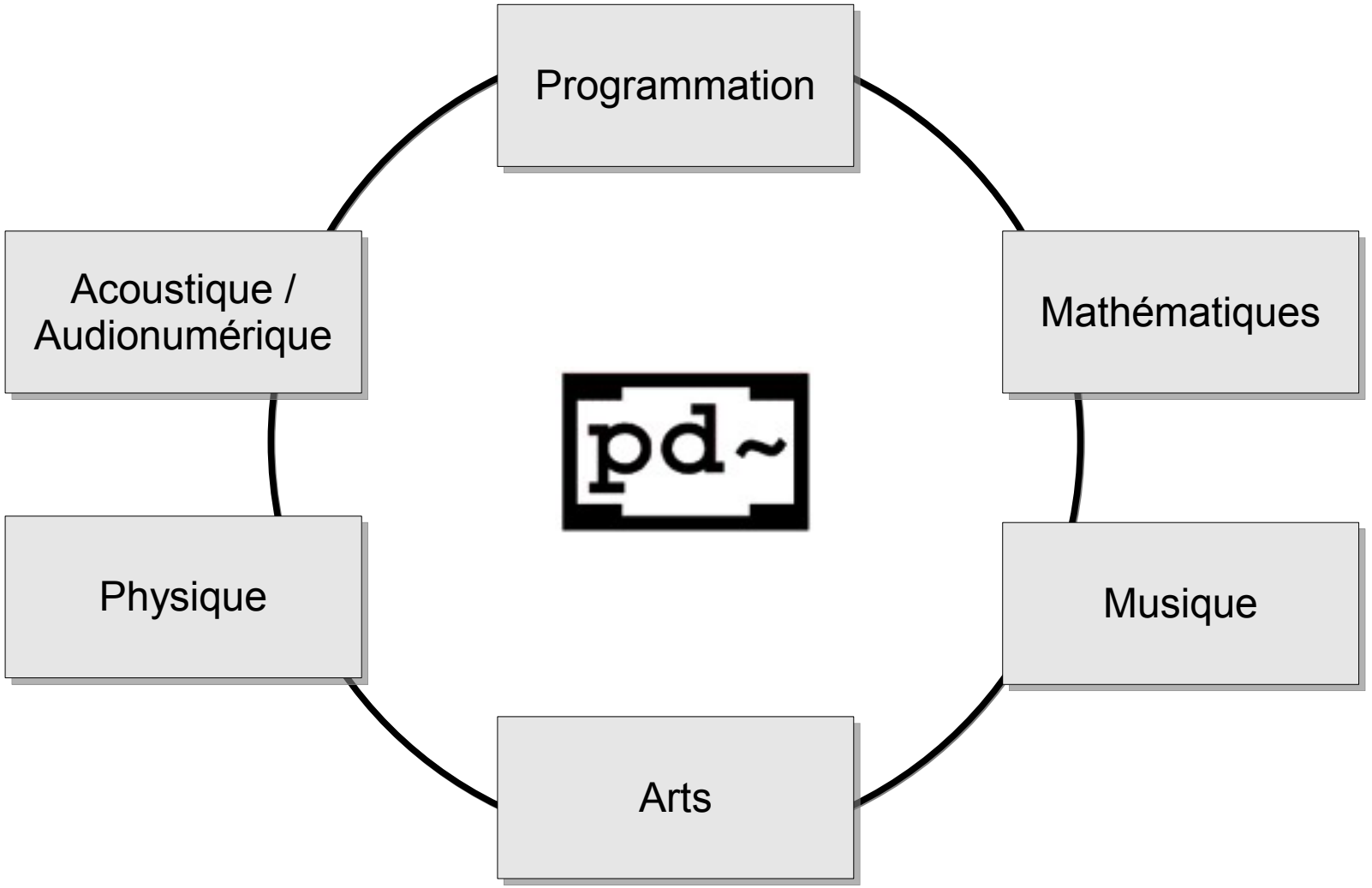
Qu'est-ce que c'est ?

A. Un logiciel pour faire des organigrammes
B. Un logiciel de montages électroniques ou de plomberie
C. Un environnement de programmation graphique en temps réel pour la création musicale et multimédia



Autour de pd

(Une diversité de connaissances qui pose des difficultés mais qui présente aussi un formidable intérêt)

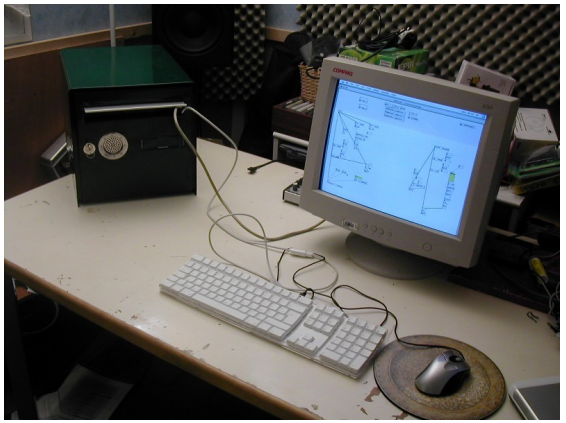


A quoi ça sert ?

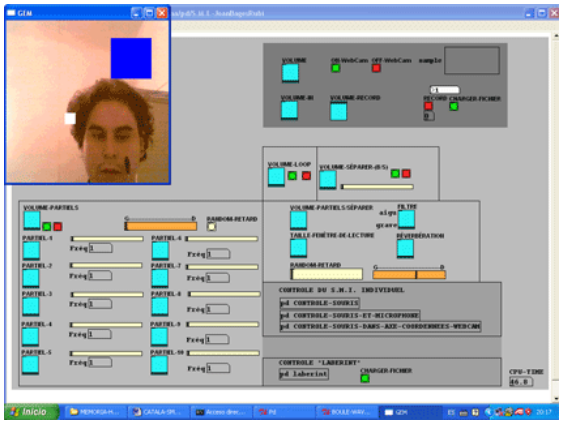
Un programme qui fait des programmes. Grande variété d'approches esthétiques, les seules limites, dit-on, sont celles de l'imagination. La difficulté n'est plus la construction de l'outil, mais sa définition, sa conception, en tenant compte d'impératifs musicaux, artistiques, techniques.

- **>** Modéliser des instruments électroniques (synthétiseurs, sampler, effets, séquenceur midi, ...)
- **>** Applications multimédias, interfaces, interactions (jeux, instruments, robotique, design d'interaction, stream...)
- **>** Concerts, performance, compositions, installations (vidéo / sonore), conception sonore (sound design)
- **>** Outil technique de mesures acoustiques
- **>** Outil pédagogique (acoustique, synthèse, ...)

A quoi ça sert ?



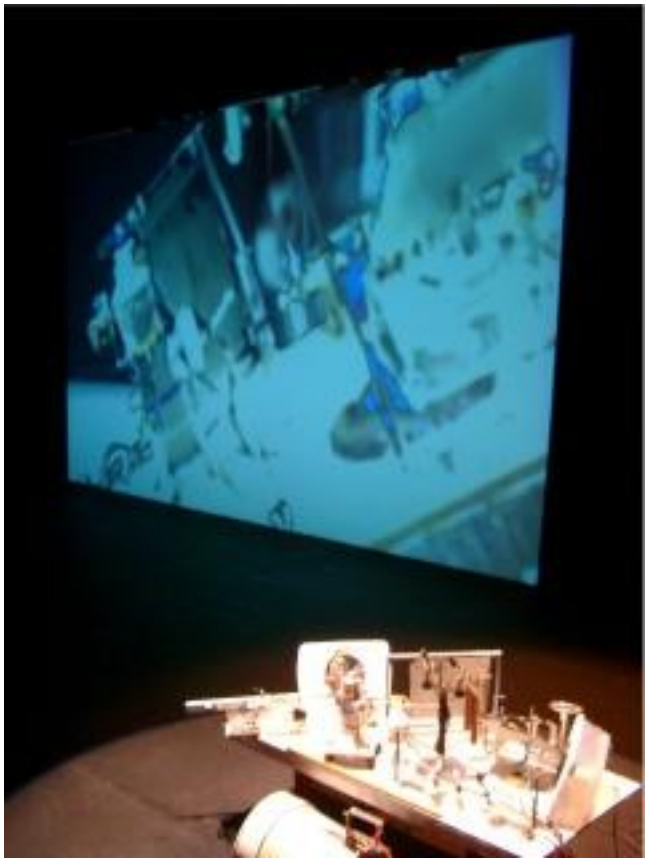
Boîte aux lettres sonore/interactive (Hacking)



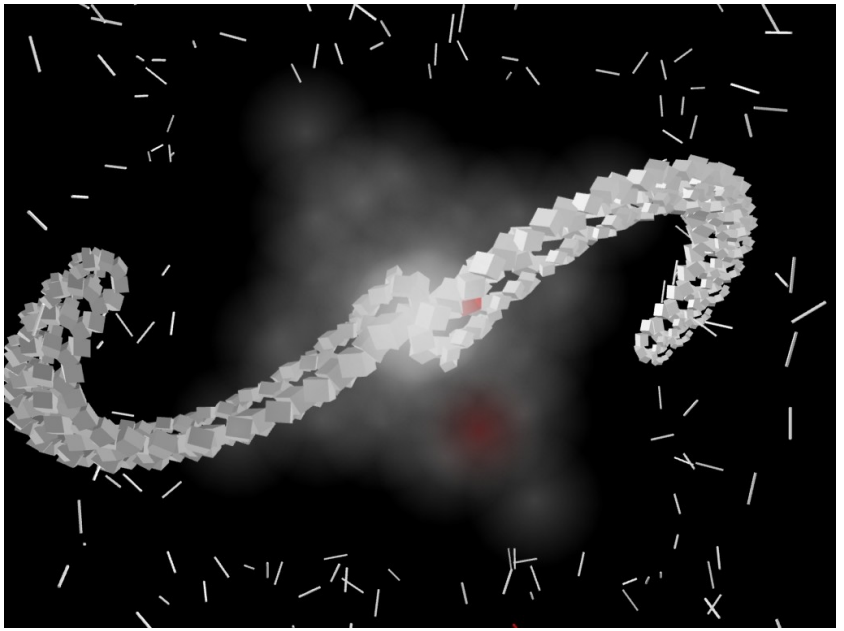
Streaming audio/vidéo

Performance / Instrument

A quoi ça sert ?

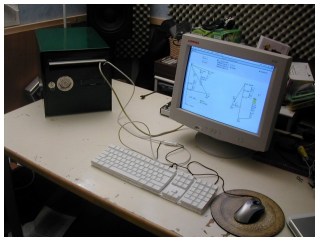


Robotique

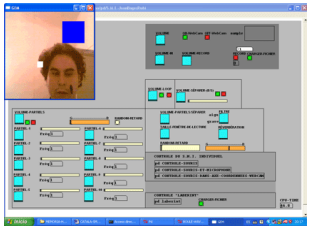


Live 3D/son

Références

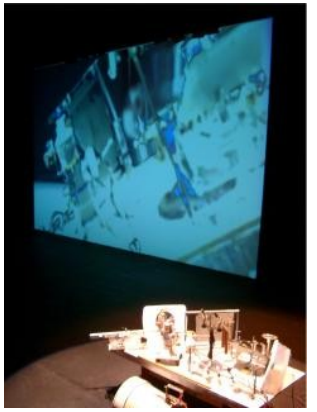


Impala Utopia (Jérôme Abel)
Unité 01
http://impala-utopia.org

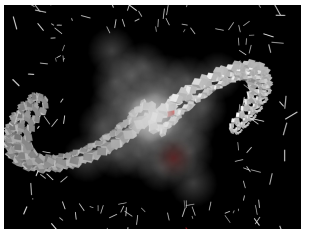


Joan Bagès I Rubi
D.E.A Systèmes Musicaux Interactifs avec interface physique
http://www.techn.upf.es/master/mad02/~m2308/pagina-wab-personal-JoanBagesRubi/cat/escolta.htm

IO team(Josh Steiner, Jacob Zweig, George Campbell, Patina Mendez, Steven Mc Donald, Nevin Cheung, Star Morin, Jon Nelson, Hans-Cristoph Steiner, Megan Keene, Candice Lucado, Aaron Young, Pasha Donelly, Adrian Mulvaney, JD Hillard)
Improbable orchestra
http:/xiphoidprocess.com/io



Team?
Robot Rock Band
http://robotrockband.com



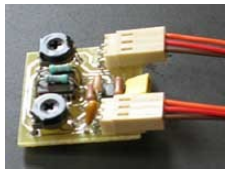
Chdh (Cyrille Henry, Damien Henry, Nicolas Montgermont)
image d'une performance
http://chdh.free.fr

Entrées / Sorties *(non exhaustif)*

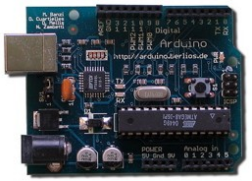
Entrées / Sorties *(non exhaustif)*

Entrées / Sorties *(non exhaustif)*

Capteurs



Interfaces capteurs



Logiciels

Usb > Midi

Interfaces actionneurs

Actionneurs

Passage entre différents mondes

Physique > analogique (électricité) > numérique (binaire) > analogique (électricité) > physique

Ressources Entrées / Sorties

C.U.I (Create USB Interface)
http://www.create.ucsb.edu/%7Edano/CUI/
http://ciam.dyndns.org/~vitamin/tof/cuiad/

Arduino + Wiring
http://hardware.processing.org/

Interfaces capteurs + actionneurs + documentations (électroniques, logiciels, ...)
http://www.interface-z.com/

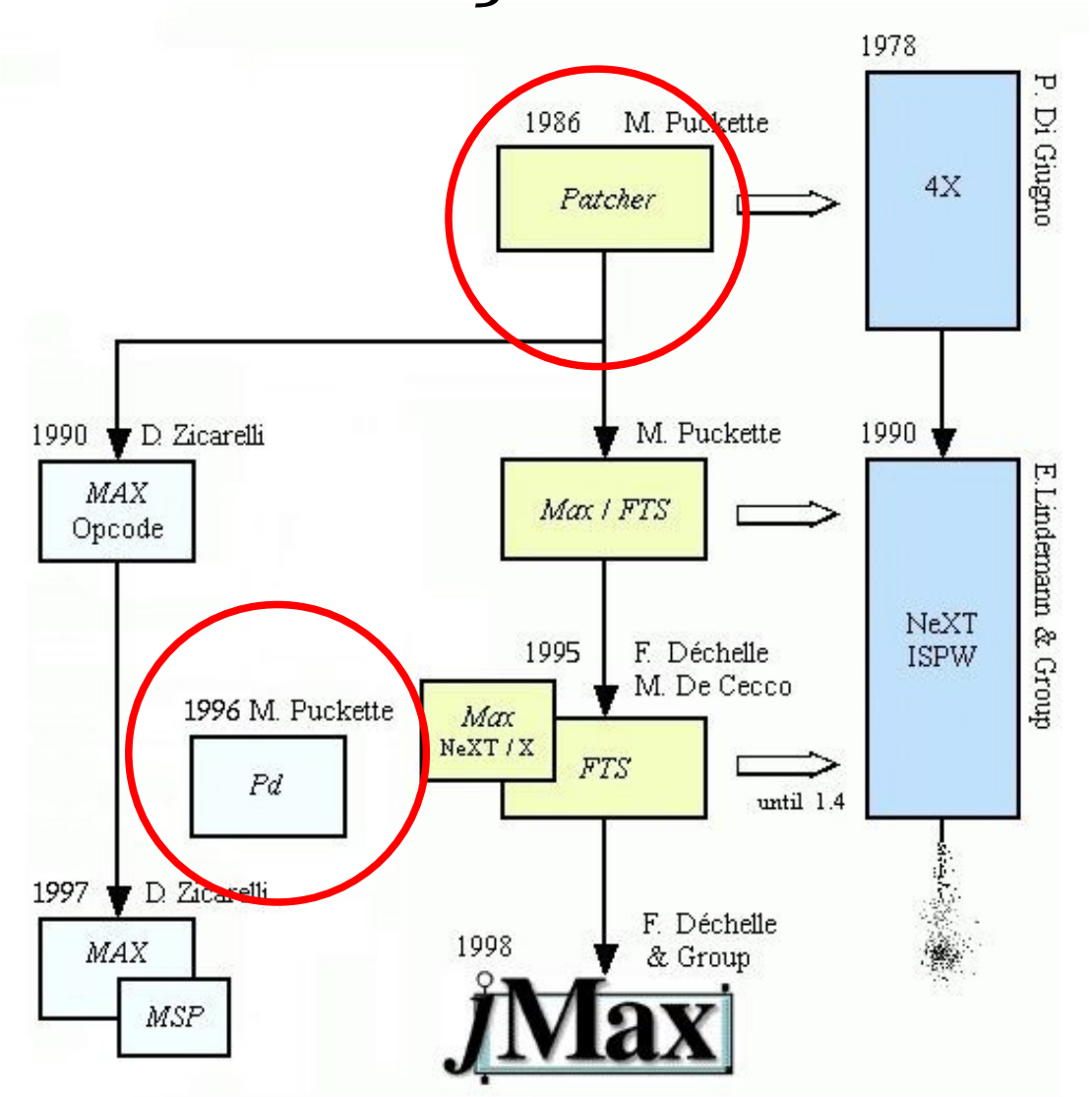
Liens Emmanuel FLETY - IRCAM Development Blog
http://recherche.ircam.fr/equipes/temps-reel/movement/flety/static.php?page=static050310-210011
The WiSe Box Project, The EtherSense Project, AtoMIC Pro : Analog to MIDI Interface, Eobody

Do It Yourselft (D.I.Y) APO 33 (joystick, clavier)
http://www.apo33.org/aposite/index.php/Chaos_Micromedias_project

Liens D.I.Y
http://del.icio.us/giair/DIY

Centre de Ressources d'Art Sensitif à Mains d'Œuvres
http://www.craslab.org/

D'où ça sort ?


1978
P. Di Giugno
4X
1986 M. Puckette
Patcher
1990 D. Zicarelli
MAX
Opcode
M. Puckette
Max / FTS
1990
E.Lindemann & Group
NeXT
ISPW
1995 F. Déchelle
M. De Cecco
Max
NeXT / X
FTS
until 1.4
1996 M. Puckette
Pd
1997 D. Zicarelli
MAX
MSP
1998 F. Déchelle
& Group
jMax

D'où ça sort ?

Il tire son origine de l'éditeur ***Patcher*** écrit par Miller Puckette en **1988**. Le logiciel a été cédé à la société américaine Opcode, où il a été réécrit par David Zicarelli sous le nouveau nom de Max/MSP. Miller Puckette a décidé de reprendre la conception de Patcher pour faire un nouveau logiciel dans le but d'obtenir un logiciel libre et transportable à des fins musicales en temps réel.

Il est distribué gratuitement sur le Web, maintenu par Miller Puckette, maintenant directeur associé du CRCA (Center for Research in Computing and the Arts) de l'Université de Californie. **De nombreux développeurs se sont par la suite joints au projet**.

Qui en est le créateur?

Miller Puckette, l'enfant prodige, propage la bonne nouvelle. Nous lui devons obéissance, notre corps et notre âme lui appartiennent.

Qui sont ses fidèles ?

Rencontres sous forme de conférences, d'ateliers, de formations, de festivals.
A noter : la 1ère convention internationale à Graz (Autriche) en 2004. La prochaine convention aura lieu à Montréal en Août 2007. Constat sociologique, comme toujours, il y a une très faible représentation de femmes et de pays du Sud.

Quelle licence ?

La licence qui protège ce logiciel permet un **travail collaboratif** en le rendant accessible au niveau des sources et en permettant de le distribuer gratuitement. Pure Data est étrangement en copyright, mais il est libre d'utilisation (**logiciel libre**) pour n'importe quel usage. « *Standard Improved BSD License* »

Une large **communauté** de part le monde participe à ce projet. Elle se compose de **développeurs** et d'**utilisateurs**. Les premiers créent de nouveaux objets en langage C ou contribuent à améliorer le fonctionnement général du programme. Les seconds l'utilisent à des fins artistiques, scientifiques ou pédagogiques. Ils peuvent aussi contribuer en faisant des remarques aux développeurs ou en aidant à leur manière (documentations, ...).

Comment se le procurer ?

A. une connexion internet suffit
B. une connexion internet et une carte bleue
C. une connexion internet, une carte bleue, et la récitation de mantras.

Comment se le procurer ?

A. une connexion internet suffit

B. une connexion internet et une carte bleue

C. une connexion internet, une carte bleue, et la récitation de mantras.

Différences avec un langage de programmation ?

Ce n'est pas vraiment un langage de programmation en tant que tel, mais un environnement d'implémentation. Il emprunte à la programmation classique certaines notions : noms d'objets, de méthodes, certains types de données (int, float, ...). C'est un système conçu de façon modulaire. Chaque utilisateur peut adapter le logiciel selon ses besoins, la copie de bout de code est très facile, ce qui permet de faire des prototypes assez rapidement.

Il exploite un langage de programmation non procédural avancé (programmation orientée objet) qui permet à l'utilisateur d'effectuer des **modifications de code en temps réel sans que celui-ci doive chaque fois être recompilé**.

Alors que pour un langage de programmation, il y a trois étapes :

1. enregistrer un fichier (fichier source)
2. le compiler dans un terminal (fichier binaire compréhensible par la machine)
3. le lancer dans un terminal.

Différences avec un langage de programmation ?

Cependant, la **non-linéarité** de cette approche peut parfois causer des difficultés dans la conception des patchs, surtout à l'étape de composition. Elle peut notamment entraîner de la confusion dans la compréhension et dans la gestion de l'ordre des opérations. On peut arranger ça, on le verra plus tard.
(> voir *http://www.earcatching.com/pdconv/index.htm : convention de programmation pour travailler à plusieurs.*).

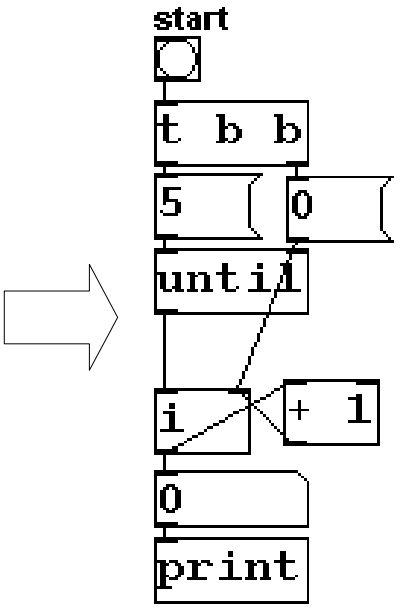
Exemple : une boucle
but : afficher 0,1,2,3,4 dans un terminal

En Java :

```
for (int i = 0; i < 5; i++) {
   System.out.println(i);
}
```

Dans Pure Data :

Problème d'organisation des fils.
Problème de lisibilité, d'ordre.
Ca commence où, qui fait quoi ?



Multi-plateforme ?

C'est un environnement **portable** avec une architecture à deux composants, client/serveur : l'interface graphique-utilisateur **TCL/TK** et le moteur d'exécution temps réel ("le serveur"). Ils peuvent ainsi être développés séparément.

TCL/TK est un kit d'outils gratuit pour créer des interface-utilisateur, portable sur d'autres système d'exploitation. La communication entre les deux se fait via un protocole indépendant du matériel, qui peut utiliser une connexion réseau comme couche de transport (le firewall doit permettre cette liaison)

Le fait d'etre **multi-plateforme** implique quelques contraintes. Un moteur graphique compatible est peu performant puisqu'il n'est pas spécifiquement optimisé pour un type de machines.

Différences avec MaxMSP ?

(voir : ./pd/doc/1.manual/x5.htm#s3)
Le point commun c'est Miller Puckette, qui a collaboré au développement de ce type de logiciel à l'Ircam. Ils appartiennent à la même famille « *patcher programming languages* » (Max/FTS, ISPW Max, Max/MSP, jMax, etc.)

Au départ, il y a Max, un système midi. Ensuite, a été ajouté la partie MSP (Max Signal Processing), qui permet de générer du son en temps réel.

Frères ennemis ou faux jumeaux ?

> Il est possible d'exporter les connaissances acquises de l'un vers l'autre, mais pas les fichiers. En théorie, ca peut marcher en nommant les fichiers « .pat ».
> Les objets de base sont identiques, mais certains ne se nomment pas de la même façon.
> Max bénéficie de plus de dix ans de recherches et d'ajouts. Plus complet.
> Pd dispose d'objets singuliers notamment pour des opérations de réseau qui n'ont pas d'équivalent sur Max.
> Max est commercialisé par Cycling 74, Pd est un logiciel libre (état d'esprit, conviction politique, éthique)
> MAXMSP / Jitter : prix : 1000 €, prix étudiant : 600 € (max seul 600€ ou 300€ pour étudiants), Pd Gem est offert, Jitter est vendu séparément (mais il existe aussi des applications libres, ARGO : http://perso.orange.fr/Paresys/ARGO)
> Ordre des instructions droite gauche dans l'emplacement du patch. Dans pure data, ordre des instructions = ordre de création
> Max a une **meilleure documentation** (c'est ce qu'on essaie de réparer ici)
> **Interface plus conviviale**. (couleurs, outils).
> Certains professeurs utilisant Max dans les écoles, ne proposent pas la solution gratuite de pure data (= lobby ou ignorance ?)
> **Avec max on peut créer des VST/ RTAS plugin et des applications standalone** (= autonome, évite d'installer le logiciel)
> La gestion des traits de connections dans Max en traits droits et anguleux permet une **meilleure visibilité**. (dans pd la fonction menu Edit > tidy up n'est pas encore vraiment efficace)

D'autres logiciels sont basés sur ce principe modulaire (patcher) : Bidule, Isadora, Reaktor, Eyes web, Modular (Clavia), vvvv, ...

Où trouver des ressources ?

Je vous conseille de commencer avec Wikipedia :

* Wikipedia (fr) : http://fr.wikipedia.org/wiki/Pure_Data

* Miller Puckette (en) : http://www-crca.ucsd.edu/~msp/
* Communauté (en) : http://www.puredata.info/
* Installeurs de Hans (en) : http://at.or.at/hans/pd/installers.html
* Forum, documentation (fr) : http://artengine.ca/~idecibel/
* Forum (en) : http://puredata.hurleur.com/
* Webring (en, fr) : http://pd.klingt.org/webring/

Pour des questions plus avancées :

* Mailing lists : http://lists.puredata.info/listinfo

En France

(Liste non exhaustive)

* Art Sensitif, CrasLab
* Art Labo, Labomedia
* Apo33
* Goto10
* Chdh
* Locus Sonus
* Beaux Arts Aix-En-Provence
* Ircam
* Impala Utopia
* Idecibel
* MakeArt
* Mal au pixel
* Instants chavirés
* Quebec (alexandre Quessy, Pdhtml (montreal, ...)
* Interface-Z
* CICM, Beaux Arts, Universités (Paris 8)
* ...

Deuxième Partie

Installation
Organisation
Démarrage
Ressources dans Pd
Le patch
Terminologie

Installation

Deux liens pour télécharger le logiciel. Deux versions existent : Pd et **Pd extended**. Il faut savoir que le noyau, c'est-à-dire Pd avec les objets de base, on dit objets natifs, est sur le site de Miller Puckette. Il est développé séparément des librairies qui gravitent autour de Pd. Pour débuter, il vaut mieux privilégier la version *extended.* Elle incorpore une version stable de pure data plus ancienne avec beaucoup plus d'objets donc de fonctionnalités.

> http://www.puredata.info/downloads
Pd-extended :

Zexy 1.3, librairie.
Iemlib 1.15, librairie de l'univesristé IEM..
Cyclone 0.1 49, libraire d'objets tirés de Max/Msp.
Gem 0.9, librairie vidéo.
Pmpd 0.5, librairie d'objets permettant la modélisation de physique mécanique.
Xsample 0.3.0pre20, librairie d'objets permettant des fonctions avancées de lecture et d'écriture de fichiers sonores.
OSC, librairie d'objets permettant la communication par protocol OSC.

> Hans-Christoph Steiner : http://at.or.at/hans/pd/installers.html

Installation

Le fichier d'installation

Windows (.exe) : lancer l'installeur, choisir répertoire (depuis la version 0.37 ou 0.38 les espaces dans le chemin de ce répertoire ne posent plus de problèmes.)

MacOSX (.dmg): déplacer Pd.app où vous voulez
site de ressources pour MacOsX : http://homepage.mac.com/atl/pd/links.html

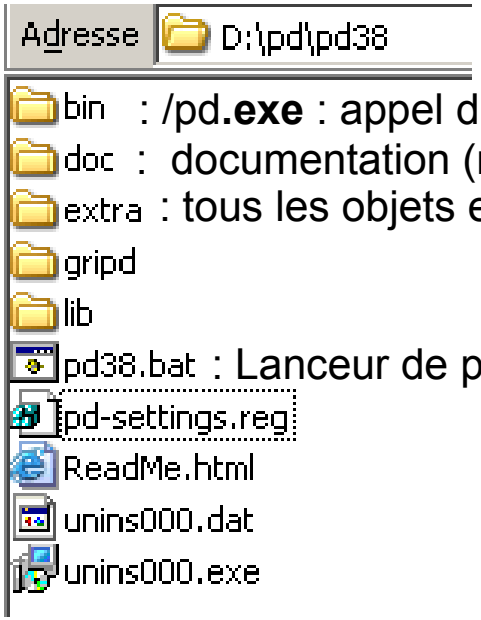
Linux : apt-get install puredata + planet CCRMA

>>> Distributions Linux orientées multimédia : Demudi (Agnula), Pure Dyne (Goto10), Ubuntu, Apodio (Apo 33), Dyne:bolic.

Organisation *(des fichiers)*

Répertoire principal :
Linux, le répertoire est **/usr/lib/pd/**
Mac OsX, le répertoire est **/usr/local/lib/pd/**
Windows : le répertoire est choisi lors de l'installation

Pour Mac, il faut souvent procéder d'une autre façon :
1. ouvrir le dossier "application"
2. trouver le lanceur pure data
3. click droit (ctrl + click) puis "afficher le contenu du paquet"
4. ouvrir le dossier "contents/ressources/extra"



bin : /pd**.exe** : appel du programme
doc : documentation (manuel, exemples control, audio, fft; références (aides d' objets), ...)
extra : tous les objets et les librairies (paquet d'objets), des fois, il s'appelle « externs »
pd38.bat : Lanceur de pure data, il se termine par **.bat** pour windows

Les objets prennent l'extension :
** **.dll** pour Windows*
** **.pd_darwin** pour Mac*
** **.pd_linux** pour Linux*

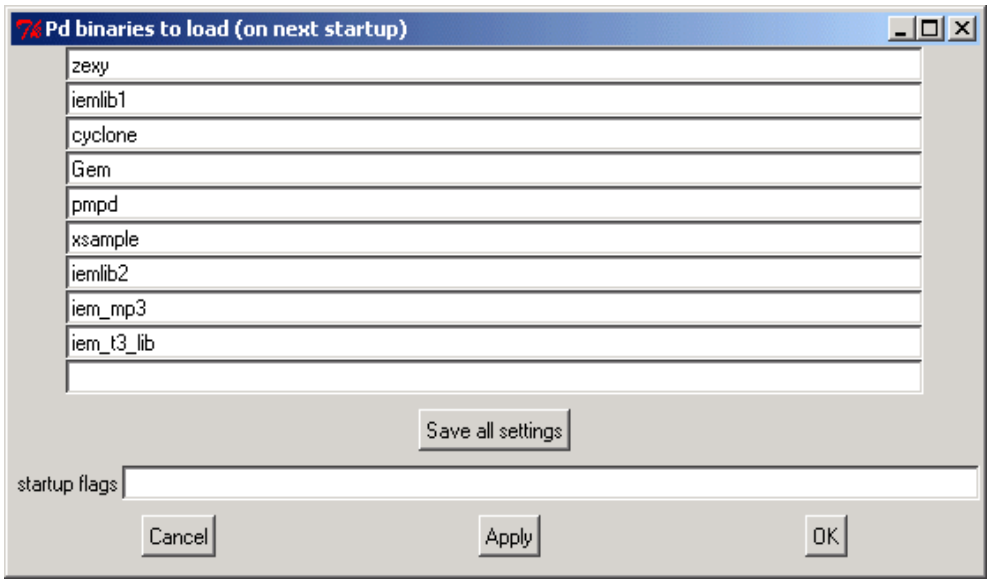
Démarrage

Le démarrage est effectué soit par le lanceur du programme (son icône en général), soit par un fichier où est enregistré les ***options de démarrage*** *(« startup options », « configuration flags »). Pour gérer ces options de démarrage, il y a deux façons soit lancer le programme et utiliser le menu, soit configurer soi-même le fichier. Au début, il vaut mieux privilégier le menu.*

1. Menu

File > Startup
File > Path

Ordre à respecter :
Apply > Save all settings > Ok



Dans cet exemple, on peut voir qu'il y a déjà les librairies *zexy, iemlib1, cyclone, Gem, pmpd, xsample, iemlib2, iem_mp3, iem_t3_lib.*

Démarrage

2. Fichier

Où trouver ce fichier de configuration ?

Windows :
1. Soit dans le répertoire d'installation vu précédemment, c'est le fichier qui se termine par **.bat**. S'il n'y en a pas, vous pouvez le créer suivant quelques règles que nous verrons juste après.
2. Ou alors, si vous avez choisi l'installeur de Hans Christoph Steiner, il faut double-cliquer sur `pd-settings.reg` pour importer les configurations dans le registre (accès aux librairies). On peut voir le résultat dans le "registry", (*Démarrer > Exécuter > Tapez regedit*), clé (*platform dependant way*) : "*HKEY_LOCAL_MACHINE\SOFTWARE\Pd*".

MacOsX : Fichiers dont le nom contient "org" et "puredata" dans le dossier "`Library/Preferences/`" de votre dossier d'utilisateur ("home directory") : *`~/Library/Preferences/org.puredata.pd.plist`*

Linux : *`~/.pdsettings`*. **"~"** est un raccourci pour le dossier "home"; les fichiers commençant par un point (**"."**) sont la plupart du temps cachés.

Démarrage

2. Fichier.bat pour Windows

Les fichiers qui se terminent par **.bat** sont une suite de commandes DOS. On peut les modifier (clic droit + modifier) avec un logiciel de traitement de texte basique comme bloc-notes. Pour le lancement de pure data, on peut en créer plusieurs selon ses besoins (un pour la vidéo, un pour une installation, ...)

Exemple d'un fichier : (tout sur la même ligne)

```
".\bin\pd.exe" -lib audience~ -font 10 -path ".\doc\vasp" -lib
fastools -lib cyclone -lib ext13 -lib maxlib  -lib iemlib1 -lib
iemlib2 -lib iem_t3_lib -lib iem_mp3 -lib mjLib -lib motex -lib
OSC -lib percolate -lib pdogg -lib vasp -lib xeq -lib xsample
-lib zexy -lib Gem -listdev %1 %2 %3 %4 %5 %6 %7
```

ATTENTION : Comme dans tous les programmes informatiques, chaque caractère est important. Un espace a un sens, un point virgule aussi, etc...

Démarrage

2. Fichier.bat pour Windows

Explications de ces commandes :
*(>>> pour toutes les options, voir **./pd/doc/1.manual/x3.htm#s4**)*

`"D:\pd\pd38\bin\pd.exe"` : chemin pour trouver le programme (en premier)

`-lib audience~` : charger une librairie (ici elle s'appelle audience~)

`-font 10` : taille de police

`-path ".\doc\vasp"` : chemin pour charger des abstractions

`-listdev %1 %2 %3 %4 %5 %6 %7` : listing du matériel audio, midi

ATTENTION : Comme dans tous les programmes informatiques, chaque caractère est important. Un espace a un sens, un point virgule aussi, etc...

Démarrage

<u>Annexe :</u> chemin absolu et chemin relatif

`./bin/pd.exe` : On va chercher le fichier à partir du répertoire où on est situé. C'est indiqué par le point « . ». C'est donc un chemin relatif, il dépend de la position où on se trouve.

`C:/pd/bin/pd.exe` : On cherche le fichier à partir de la racine, du début de l'arborescence des dossiers de la machine. C'est indiqué par le slash « / ». C'est donc un chemin absolu, où que l'on soit dans l'arborescence de l'ordinateur, ce chemin sera toujours correct.

`/usr/bin/pd` : La même chose sous sytème UNIX (Linux, MacOsX)

Démarrage

2. Fichier avec Linux et MacOsX

> Créer un fichier avec un éditeur de texte :

```
#vim /home/utilisateur/.pdrc
=> ~/.pdrc pour Linux,
=> /Users/utilisateur/.pdrc pour Mac

#vim /home/utilisateur/lanceur
```

Les fichiers commençant par un « . » sont cachés par défaut

> Quelques fois, il faut donner toutes les permissions au dossier ou au fichier :

```
# chmod -R a+rw /home/utilisateur/
# chmod a+rw /home/utilisateur/.pdrc
```

> Rafraîchir le fichier :

```
# source /home/giair/.pdrc
# source /home/giair/lanceur
```

Fichier « .pdrc » (*saut de ligne*) :

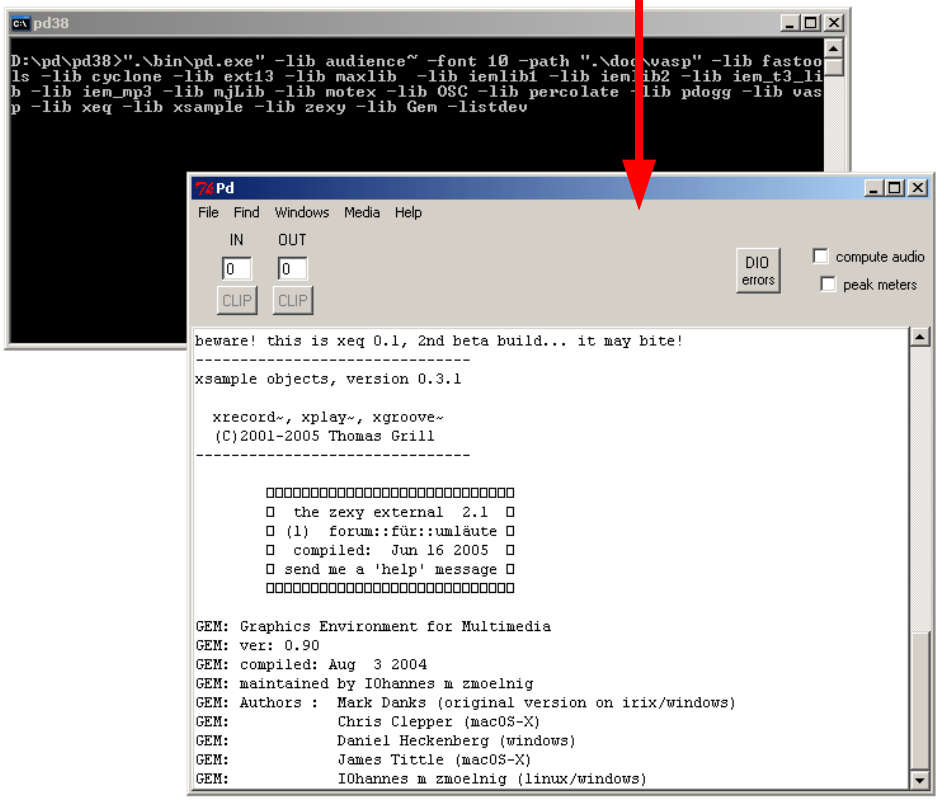
```
-path /usr/lib/pd/externs
-path /usr/lib/pd/externs/cyclone
-path /usr/lib/pd/externs
-path /usr/lib/pd/iemabs
-lib iemlib1:iemlib2
-lib cyclone
-lib zexy
```

Fichier exécutable « lanceur » (*tout à la ligne*) :

```
#!bin/sh
 /usr/bin/pd -jack -r 44100 ...
```

Démarrage

Deux fenêtres s'ouvrent (pd version 0.38):
- Le **Terminal** (fenêtre noire).
- La fenêtre principale avec menu qui fait aussi office de **fenêtre de sortie** (utile pour visualiser des informations, les erreurs)



On retrouve la configuration de démarrage écrite dans la fenêtre noire et le résultat dans la fenêtre principale.

*Quand ce message apparaît **« Can't load library »** dans la fenêtre de sortie : le chemin d'accès à la librairie n'est pas correct*
=> vérifiez bien les espaces, guillements, chaque caractère dans File > Path ou File > Startup

Ressources dans Pd

Documentation : `./pd/doc/1.manual/` ou Menu Help > Manual

Documentation début : `./pd/doc/2.control.examples/`

Documentation audio : `./pd/doc/3.audio.examples/`

Liste d'objets : `./pd/doc/5.reference/0.INTRO.txt` ou Clic Droit + Help dans une zone blanche du patch

Tous les objets et librairies : `./pd/doc/5.reference/`

Documentations des librairies : `./pd/doc/manuals/`
(Gem, Iem, Maxlib, Vasp, pmpd, ...)

Le patch

Métaphore du réseau, une sorte de **mécano cybernétique** (dixit Roland Cahen). Conception modulaire, systémique, dorénavant assimilée. Conception d'un ensemble d'instructions, de fonctions à l'aide du visuel.

<u>Quelques similitudes :</u>

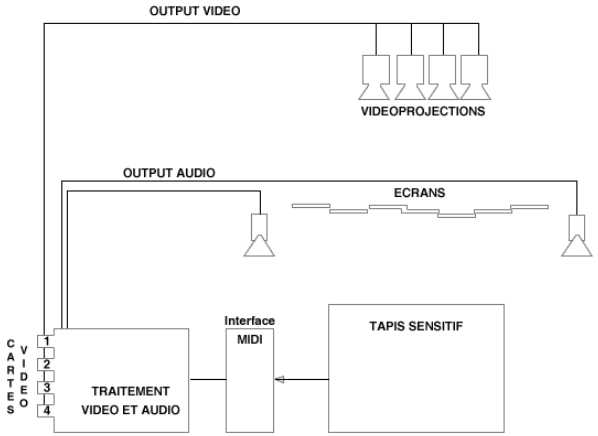


Schéma de performance artistique (de Chdh)

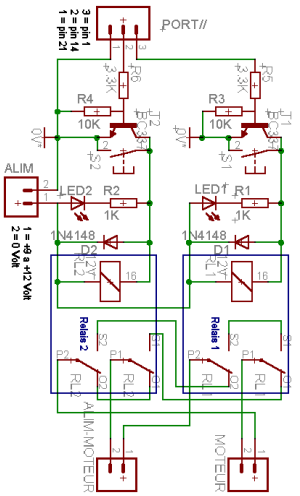
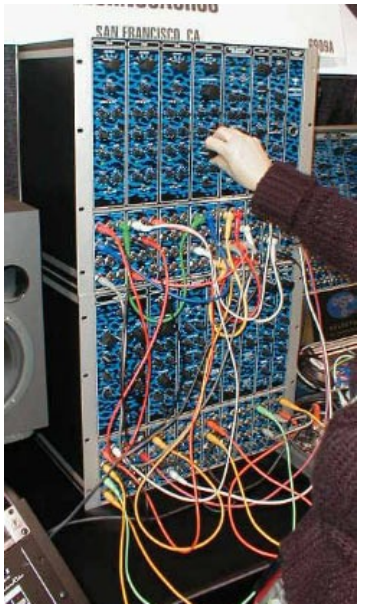


Schéma électrique

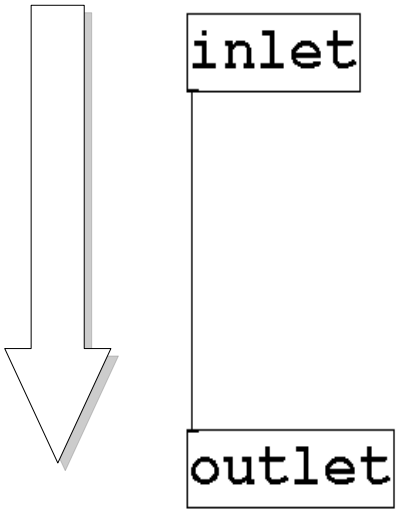
Plomberie

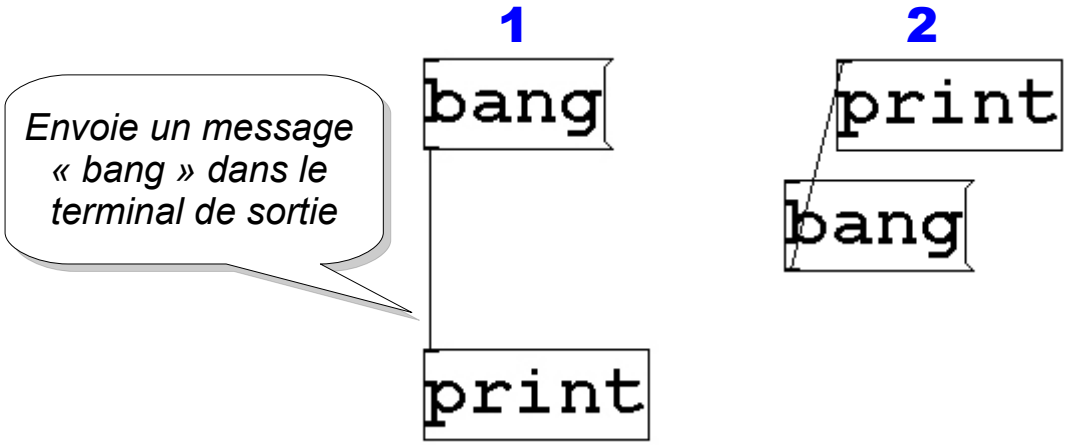


Synthétiseurs modulaires

Le patch

Pour commencer il faut créer un nouveau patch : **Menu File > New**. Un nouveau patch est une page blanche. On y ajoute des objets **Menu Put**, et on les connecte entre eux avec des fils **(quand la main devient un cercle)**. L'information parcours le fil de haut (inlet) en bas (outlet), comme de l'eau dans un tube, soumise à la gravitation.

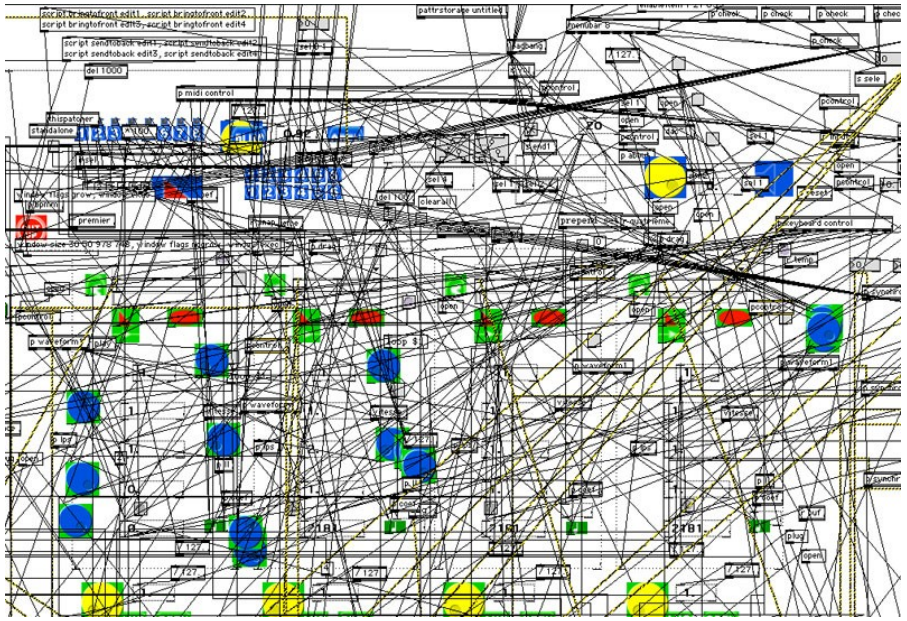




Ces deux configurations font exactement la même chose, mais la première est plus simple à lire.

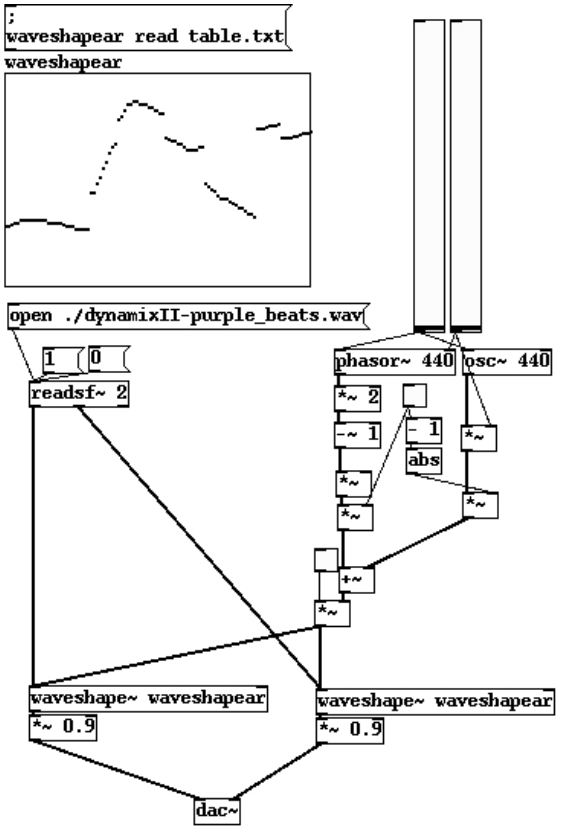
Le patch

La construction d'un patch est imprégnée d'une certaine esthétique du bordel, des connections dans tous les sens. Ca impressionne certainement, mais ça rend la **lisibilité** difficile. Avec la pratique on peut éviter certains pièges mais certains sont inévitables de toutes les façons.
>>> http://www.earcatching.com/pdconv/index.htm : convention de programmation pour travailler à plusieurs.
>>> http://www.piksel.no/pwiki/DesireData/ : projet qui propose des améliorations ergonomiques à Pure Data.



Application Max/Msp
http://www.tinytool.org
(Wilfried > Amnésie)

Bonne lisibilité

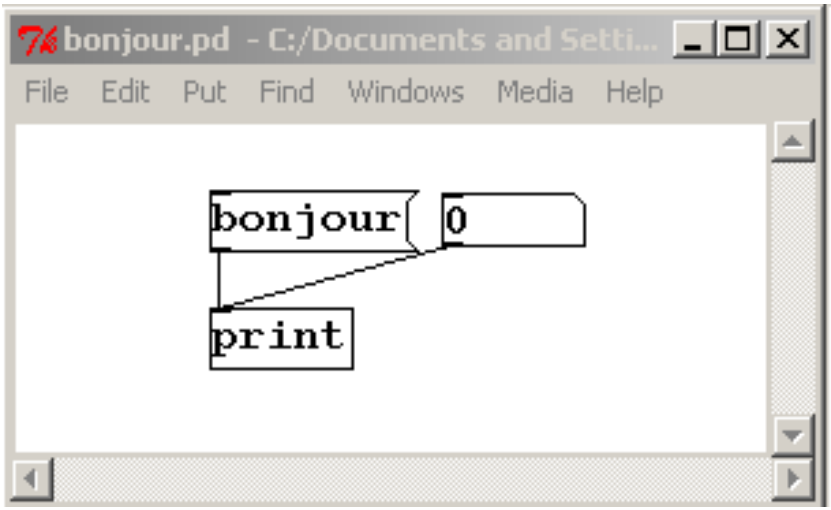


Le patch

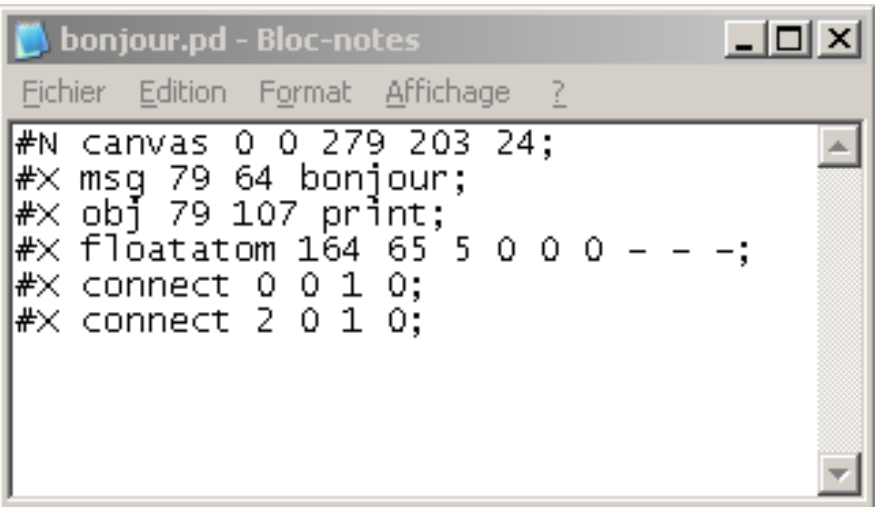
Les documents pd s'appellent des ***patchs*** ou moins couramment des ***canvas***. Dans Pure Data, ils ont l'apparence sympathique d'objets assemblés, et ouverts avec un éditeur de texte, c'est simplement du texte. Dans le texte ci-dessous, on voit en première ligne le mot ***canvas*** suivi de l'appel de création des objets et des connections, à ma connaissance, c'est tout ce qu'il y a. Cette économie d'information rend les fichiers très légers. Le poids maximum d'un fichier dans la documentation du logiciel fait dans les 20 Ko. Je présente cet aspect textuel car cela nous sert à s'échanger des patchs sur internet quand ils apparraissent seulement en version texte. Cela peut paraître surprenant, mais en copiant le texte ci-dessous, et en renommant l'extension du fichier en **.pd**, on obtient bien un fichier pure data valide !

Une astuce de recherche de patchs Pure Data sur internet, utilise le terme canvas, car il se retrouve sur tous les débuts de fichiers pd en texte (merci à Hu_koala).
Sur google, tapez : **.pd #N canvas**

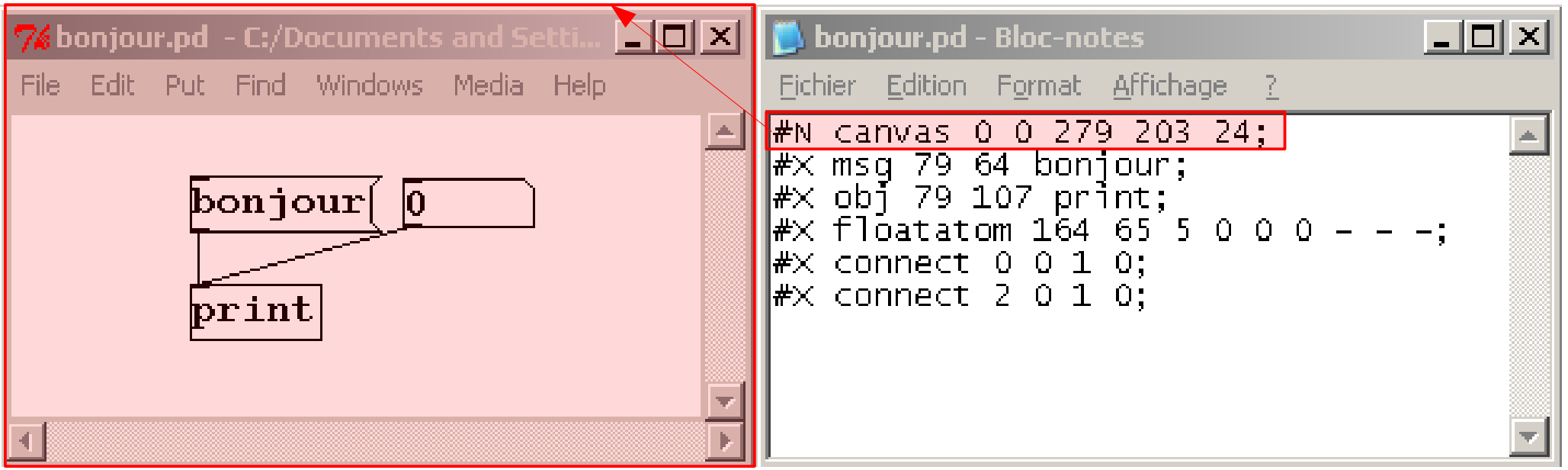
Version pure data



Version textuelle

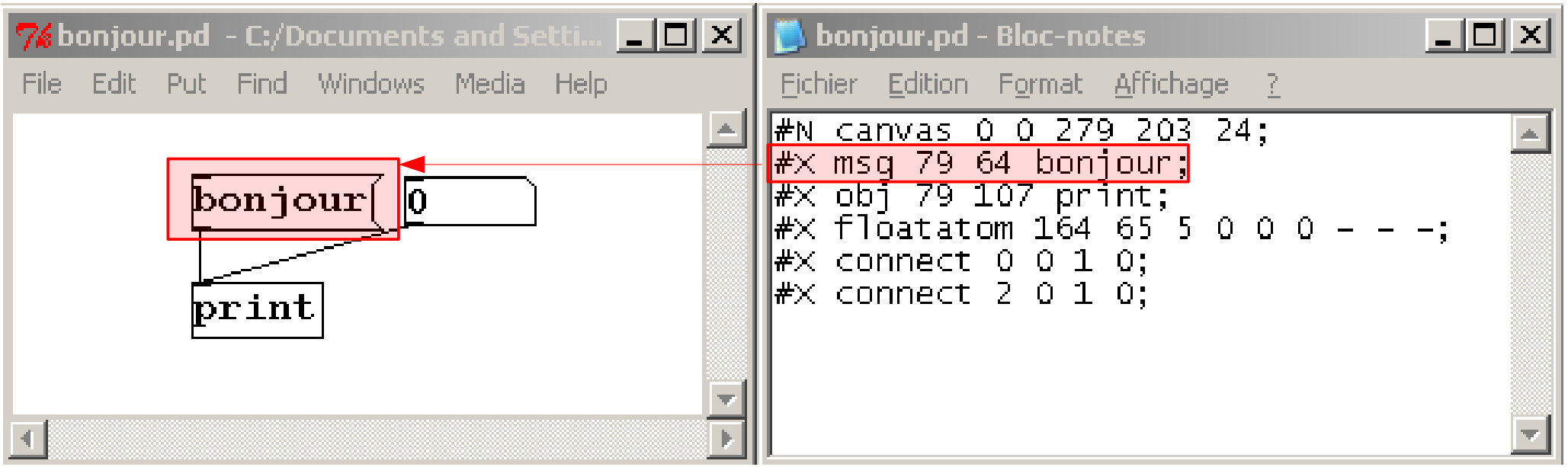


Le patch



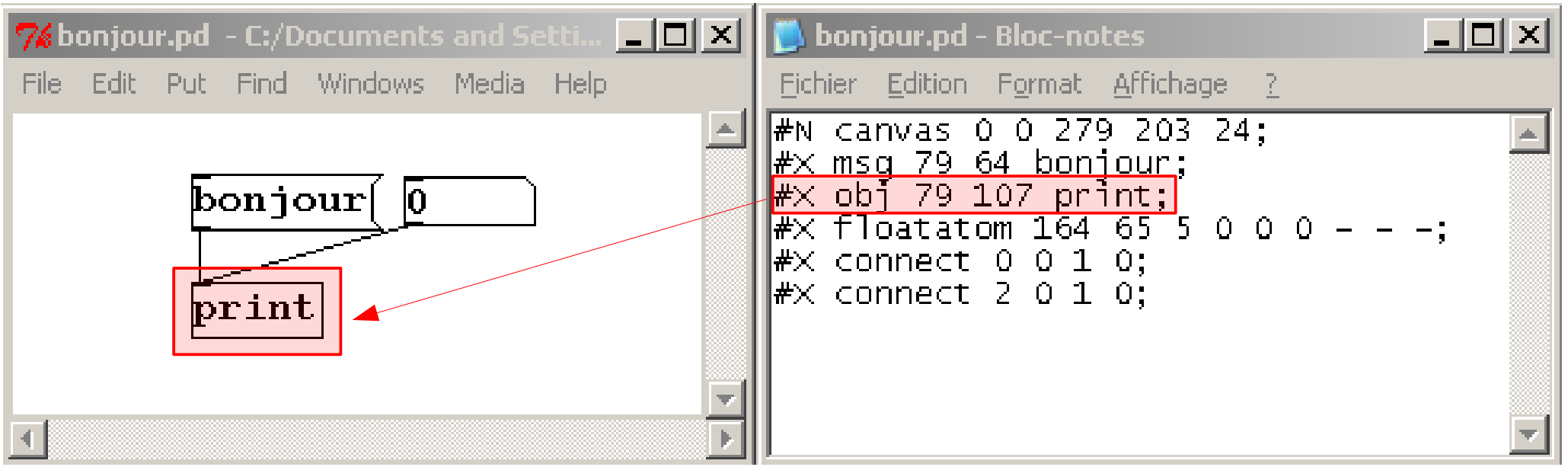
#N Canvas = nouveau patch
de **279** pixels de largeur et de **203** pixels de hauteur

Le patch



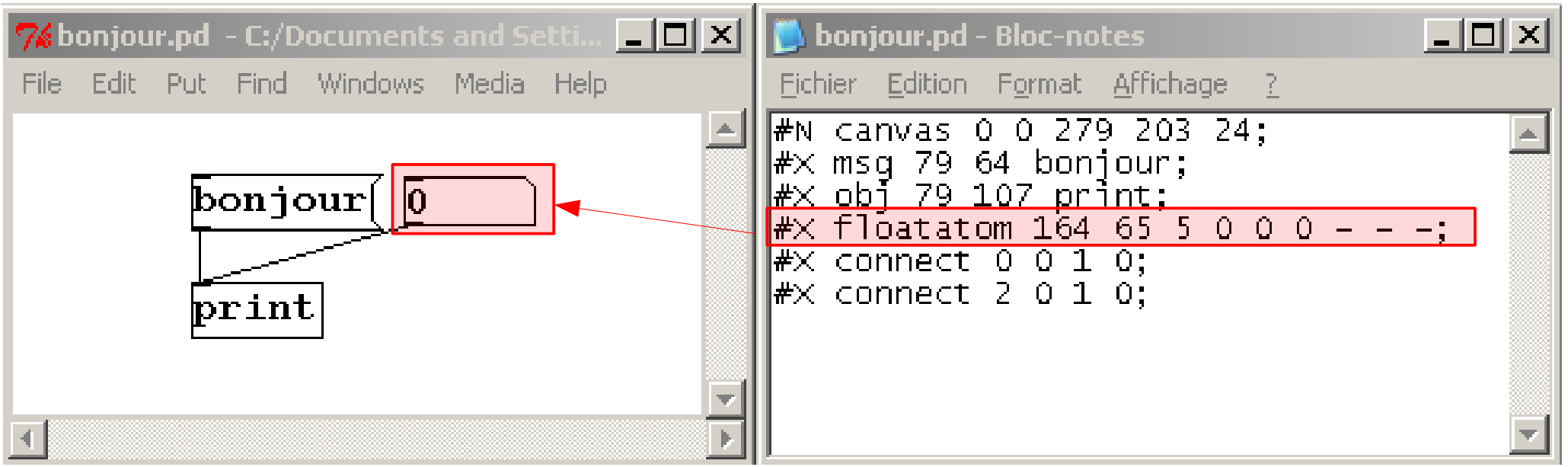
Boîte de type message (**msg**) qui s'intitule **bonjour**
située à **79** pixels du bord gauche et à **64** pixels du bord haut

Le patch



Boîte de type objet (**obj**) qui s'intitule **print**
située à **79** pixels du bord gauche et à **107** pixels du bord haut

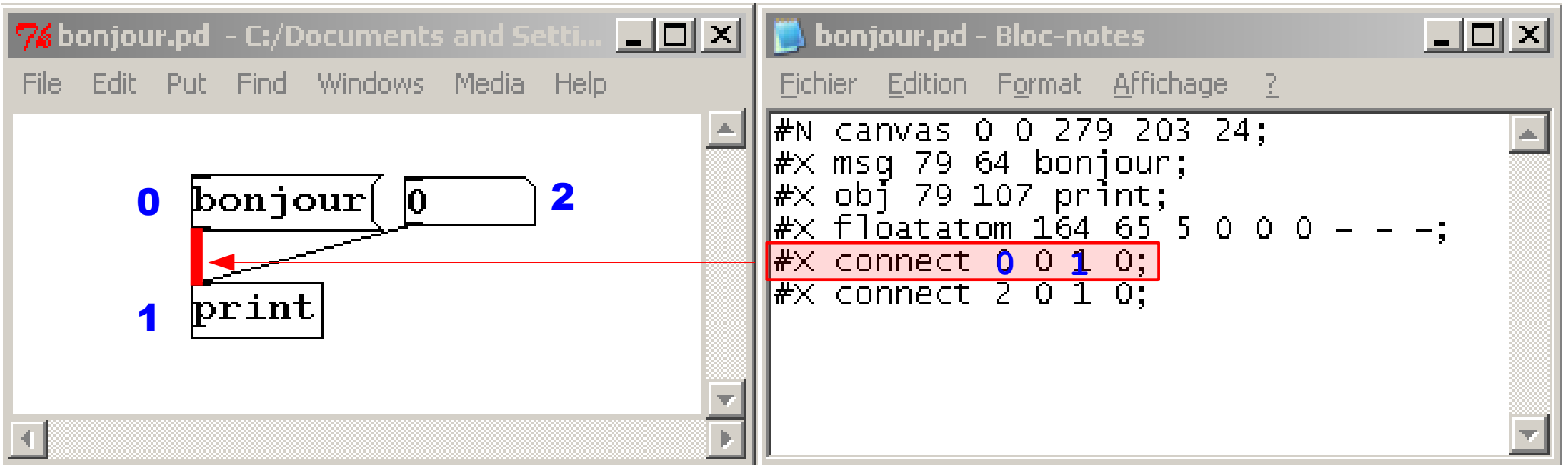
Le patch



Boîte de type chiffres (**floatatom**)
située à **164** pixels du bord gauche et à **65** pixels du bord haut

Le patch

En bleu, c'est l'ordre de création des éléments.

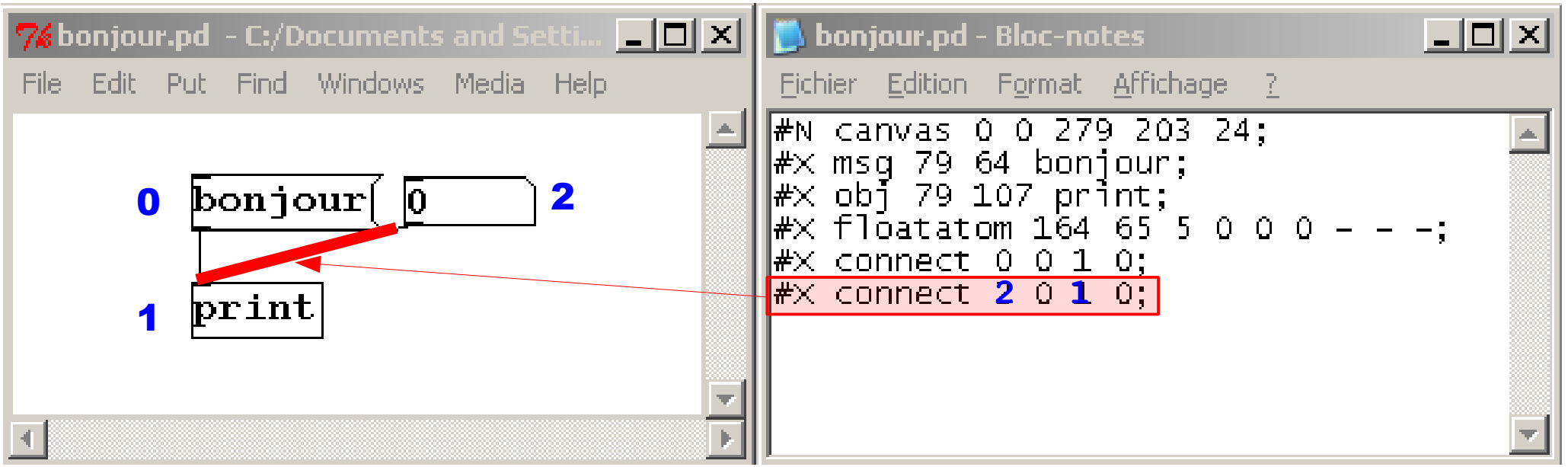


Connection (**connect**)
entre le premier élément créé (**0**) et le deuxième (**1**)

Le deuxième chiffre après l'identifiant de l'élément, c'est l'identifiant de la sortie pour l'élément d'où part la connection et l'identifiant d'entrée pour l'élément qui reçoit la connection (ici c'est 0 dans les deux cas, car les éléments n'ont qu'une seule sortie et qu'une seule entrée)

Le patch

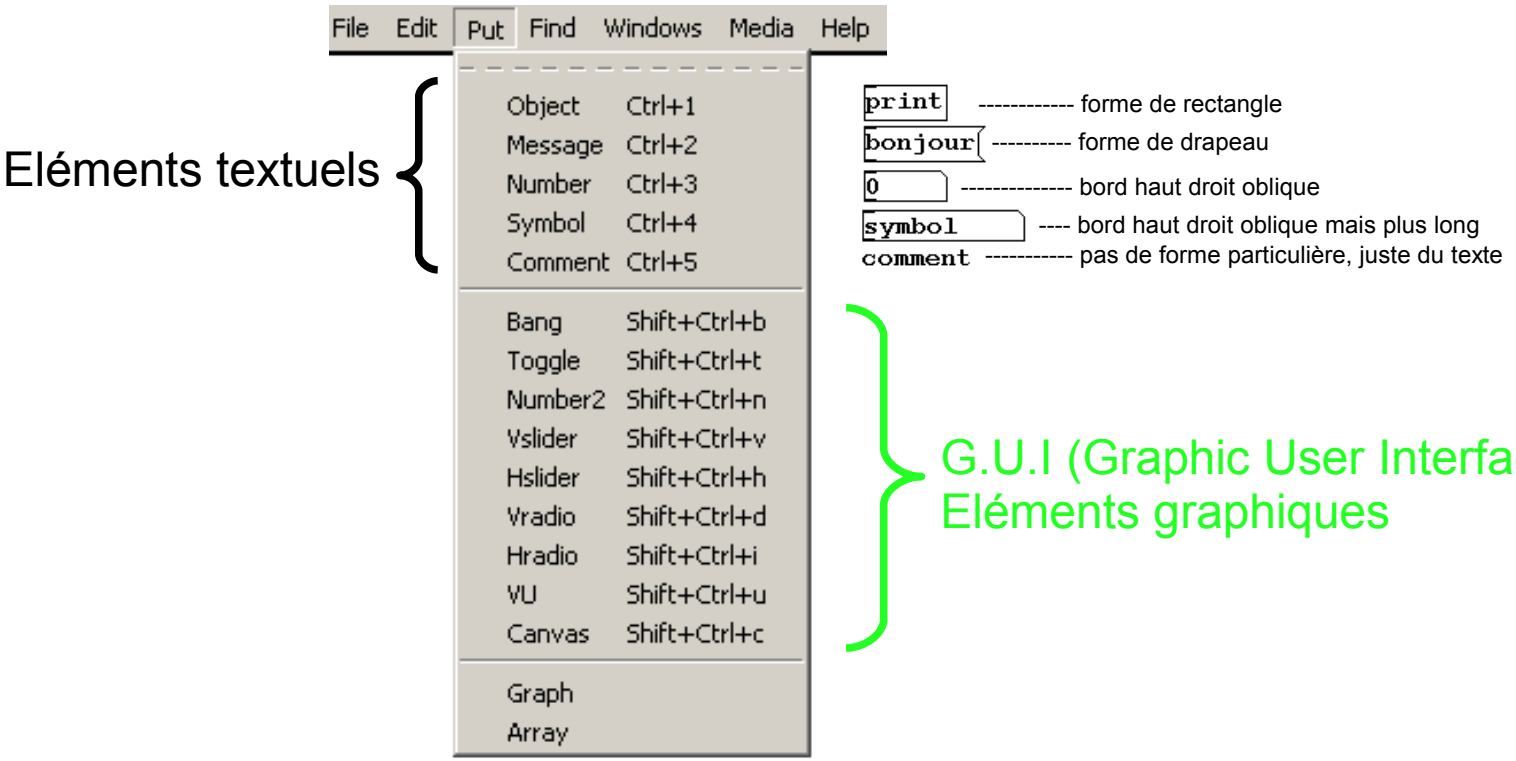
En bleu, c'est l'ordre de création des éléments.



Connection (**connect**)
entre le troisième élément créé (**2**) et le deuxième (**1**)

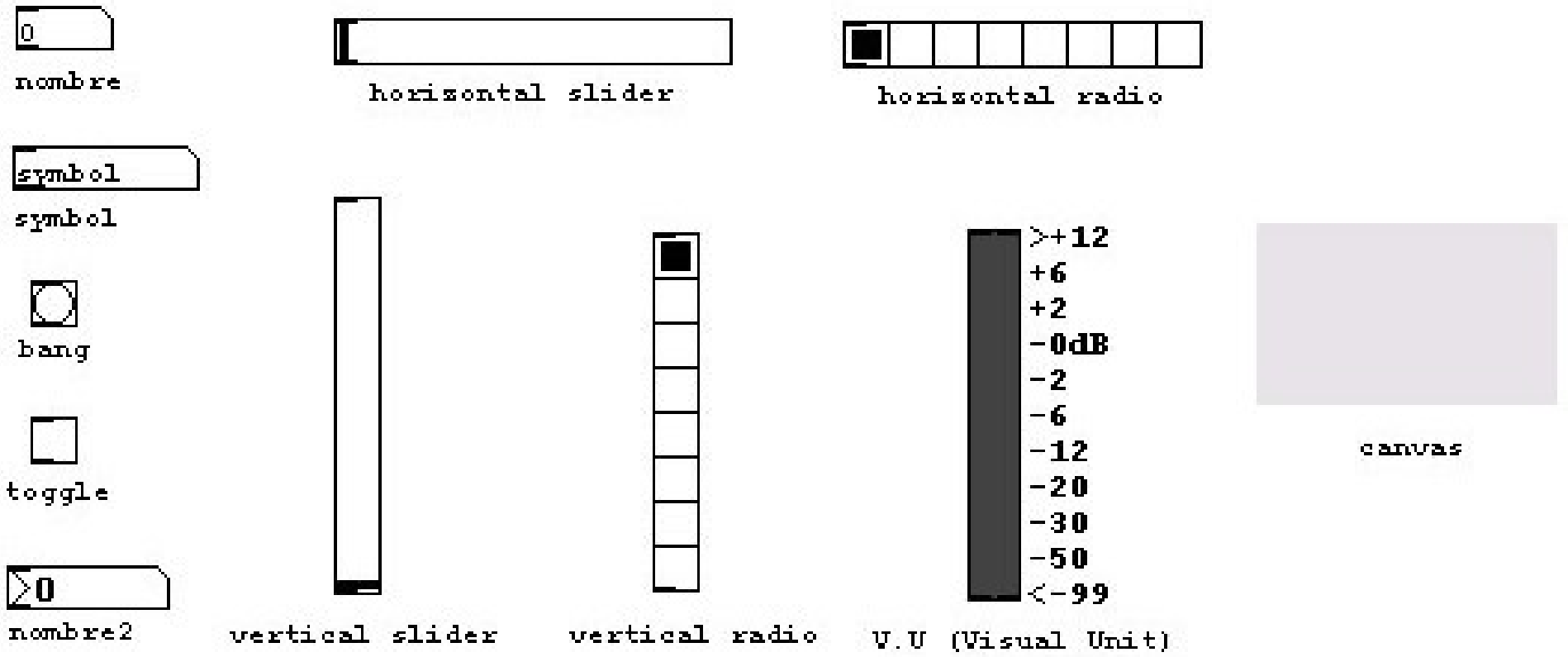
Terminologie

***ELEMENTS** (atoms) **:** En un mot, c'est tout ce qui n'est pas un fil, c'est-à-dire les boîtes. Il y a différents types d'éléments : objets, messages, symboles (=String), G.U.I, commentaires, array (=tableau), ... Pour les distinguer, il y a des repères visuels sur le côté droit de chaque élément.*



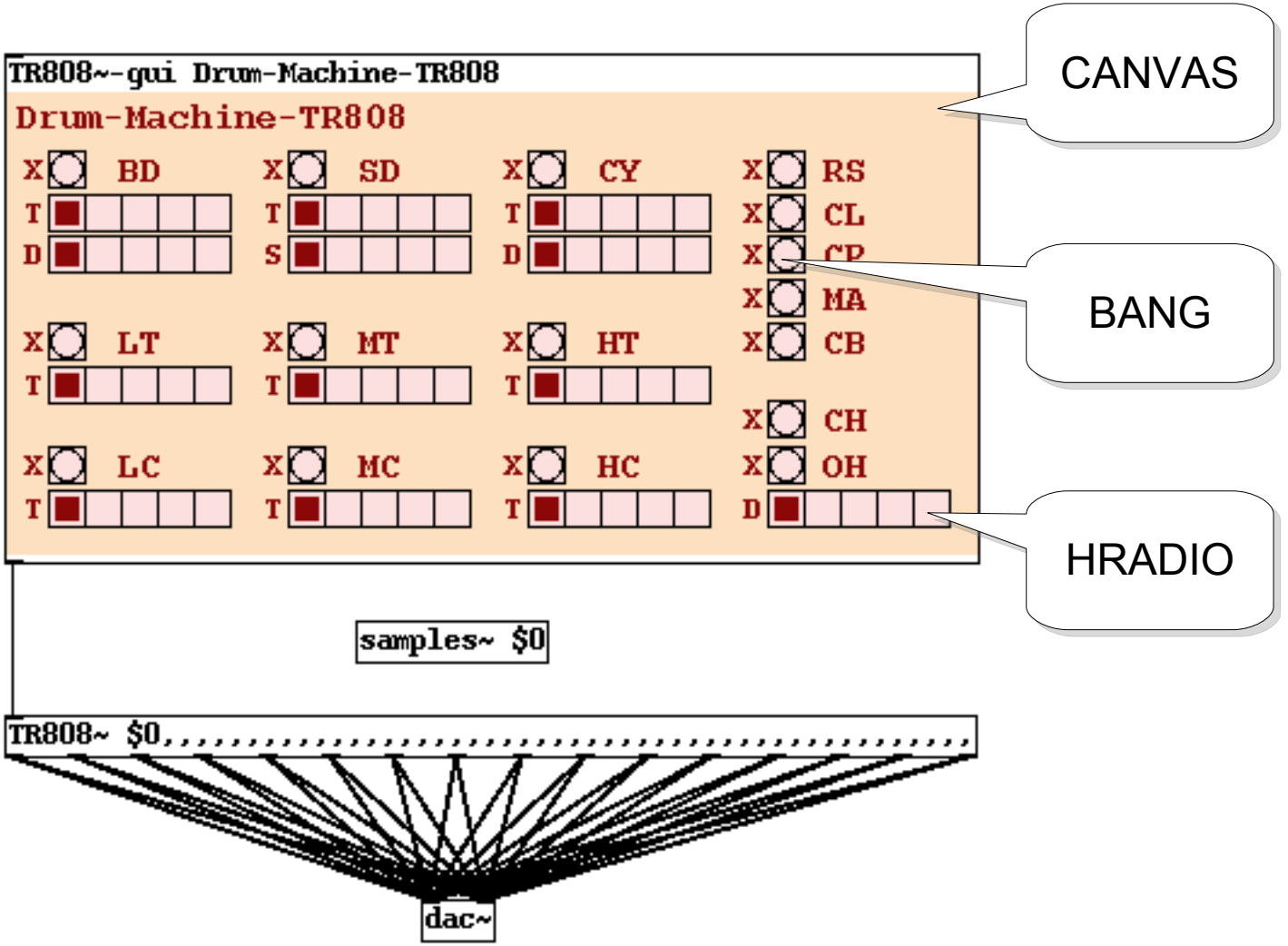
Terminologie

***G.U.I** (Graphic User Interface) : Permet de faire des interfaces de contrôle et de visualiser les informations*



Terminologie

G.U.I *: exemple d'un patch avec interface de contrôle (TR808 Claudius Maximus)*

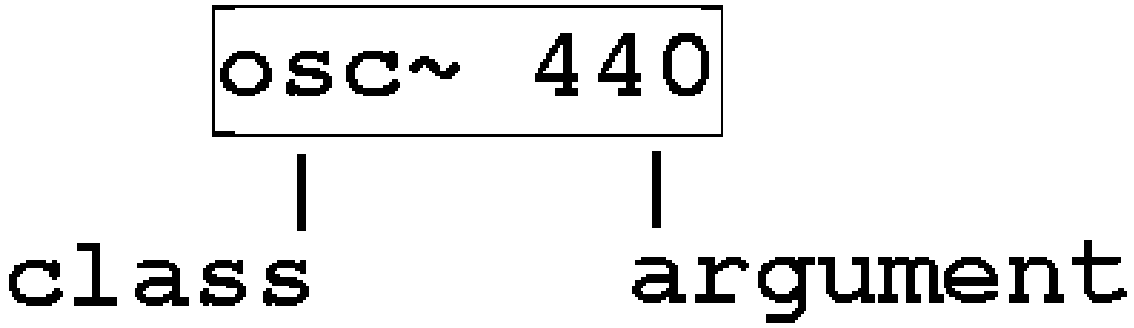


Terminologie

OBJETS : *Il faut faire la distinction entre les objets natifs et les objets externes. Ces derniers viennent des librairies. Les objets sont différents d'une abstraction, car il sont écrits en C et on ne peut pas les ouvrir directement. En effet, la seule façon de voir à l'intérieur et de les modifier c'est avec le code source en C.*

Il faut bien distinguer : ***mot clé invariable*** *(nom de l'objet) et* ***arguments*** *(ou paramètres)* ***variables*** *et relatifs à l'objet. Le mot clé est de type classe comme en programmation classique (fonctions ou méthodes)****.***

Il faut donc connaître le nom des objets. La liste des objets :
`./pd/doc/5.reference/0.INTRO.txt` *ou Clic Droit + Help dans une zone blanche du patch*
Tous les objets et librairies : `./pd/doc/5.reference/`



---------- GLUE ----------
bang - output a bang message
float - store and recall a number
symbol - store and recall a symbol
int - store and recall an integer
send - send a message to a named object
receive - catch "sent" messages
select - test for matching numbers or symbols
route - route messages according to first element
pack - make compound messages
unpack - get elements of compound messages
trigger - sequence and convert messagess
spigot - interruptible message connection
moses - part a numeric stream
until - looping mechanism
print - print out messages
makefilename - format a symbol with a variable field
change - remove repeated numbers from a stream
swap - swap two numbers
value - shared numeric value
---------- TIME ----------
delay - send a message after a time delay
metro - send a message periodically
line - send a series of linearly stepped numbers
timer - measure time intervals
cputime - measure CPU time
realtime -measure real time
pipe - dynamically growable delay line for numbers
---------- MATH ----------
+ - * / pow arithmetic
== != > < >= <= relational tests
& && | || % bit twiddling
mtof ftom powtodb rmstodb dbtopow dbtorms convert acoustical units
mod div sin cos tan atan atan2 sqrt log exp abs higher math
random lower math
max min greater or lesser of 2 numbers
clip force a number into a range
---------- MIDI ----------
notein ctlin pgmin bendin touchin polytouchin midiin sysexin - MIDI input
noteout ctlout pgmout bendout touchout polytouchout midiout - MIDI output
makenote - schedule a delayed "note off" message corresponding to a note-on
stripnote - strip "note off" messages
---------- TABLES----------
tabread - read a number from a table
tabread4 - read a number from a table, with 4 point interpolation
tabwrite - write a number to a table
soundfiler - read and write tables to soundfiles

---------- MISC ----------
loadbang - bang on load
serial - serial device control for NT only
netsend - send messages over the internet
netreceive - receive them
qlist - message sequencer
textfile - file to message converter
openpanel - "Open" dialog
savepanel - "Save as" dialog
bag - set of numbers
poly - polyphonic voice allocation
key, keyup - numeric key values from keyboard
keyname - symbolic key name
---------- AUDIO MATH ----------
+~ -~ *~ /~ arithmetic on audio signals
max~ min~ - maximum or minimum of 2 inputs
clip~ - constrict signal to lie between two bounds
q8_rsqrt~ - cheap reciprocal square root (beware -- 8 bits!)
q8_sqrt~ - cheap square root (beware -- 8 bits!)
wrap~ - wraparound (fractional part, sort of)
fft~ - complex forward discrete Fourier transform
ifft~ - complex inverse discrete Fourier transform
rfft~ - real forward discrete Fourier transform
rifft~ - real inverse discrete Fourier transform
framp~ - output a ramp for each block
mtof~, ftom~, rmstodb~, dbtorms~, rmstopow~, powtorms~ - acoustic conversions---------- AUDIO GLUE ----------
dac~ - audio output
adc~ - audio input
sig~ - convert numbers to audio signals
line~ - generate audio ramps
vline~ - deluxe line~
threshold~ detect signal thresholds
snapshot~ - sample a signal (convert it back to a number)
vsnapshot~ deluxe snapshot~
bang~ - send a bang message after each DSP block
samplerate~ get the sample rate
send~ - nonlocal signal connection with fanout
receive~ - get signal from send~
throw~ - add to a summing bus
catch~ - define and read a summing bus
block~ - specify block size and overlap
switch~ - switch DSP computation on and off
readsf~ - soundfile playback from disk
writesf~ - record sound to disk

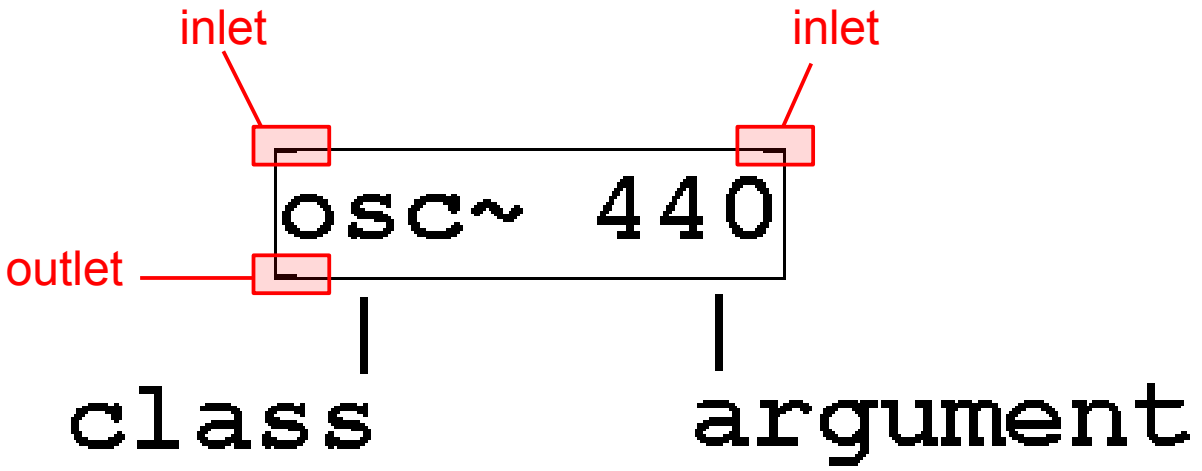
---------- AUDIO OSCILLATORS AND TABLES ----------
phasor~ - sawtooth oscillator
cos~ - cosine
osc~ - cosine oscillator
tabwrite~ - write to a table
tabplay~ - play back from a table (non-transposing)
tabread~ - non-interpolating table read
tabread4~ - four-point interpolating table read
tabosc4~ - wavetable oscillator
tabsend~ - write one block continuously to a table
tabreceive~ read one block continuously from a table
---------- AUDIO FILTERS ----------
vcf~ - voltage controlled filter
noise~ - white noise generator
env~ - envelope follower
hip~ - high pass filter
lop~ - low pass filter
bp~ - band pass filter
biquad~ - raw filter
samphold~ - sample and hold unit
print~ - print out one or more "blocks"
rpole~ - raw real-valued one-pole filter
rzero~ - raw real-valued one-zero filter
rzero_rev~ rzero~, time-reversed
cpole~, czero~, czero_rev - corresponding complex-valued filters
---------- AUDIO DELAY ----------
delwrite~ - write to a delay line
delread~ - read from a delay line
vd~ - read from a delay line at a variable delay time
---------- SUBWINDOWS ----------
pd - define a subwindow
table - array of numbers in a subwindow
inlet - add an inlet to a pd
outlet - add an outlet to a pd
inlet~, outlet~ - signal versions of inlet, outlet
---------- DATA TEMPLATES ----------
struct - define a data structure
drawcurve, filledcurve - draw a curve
drawpolygon, filledpolygon - draw a polygon
plot - plot an array field
drawnumber - print a numeric value

---------- ACCESSING DATA ----------
pointer - point to an object belonging to a template
get - get numeric fields
set - change numeric fields
element - get an array element
getsize - get the size of an array
setsize - change the size of an array
append - add an element to a list
sublist - get a pointer into a list which is an element of another scalar
scalar - draw a scalar on parent

Terminologie

OBJETS

*Chaque objet a des entrées (inlets) et sorties (outlets) différentes. En faisant **Clic droit + help**, on peut voir à quoi elles correspondent. Elles sont représentées par des petits rectangles noirs en haut et en bas de l'objet.*

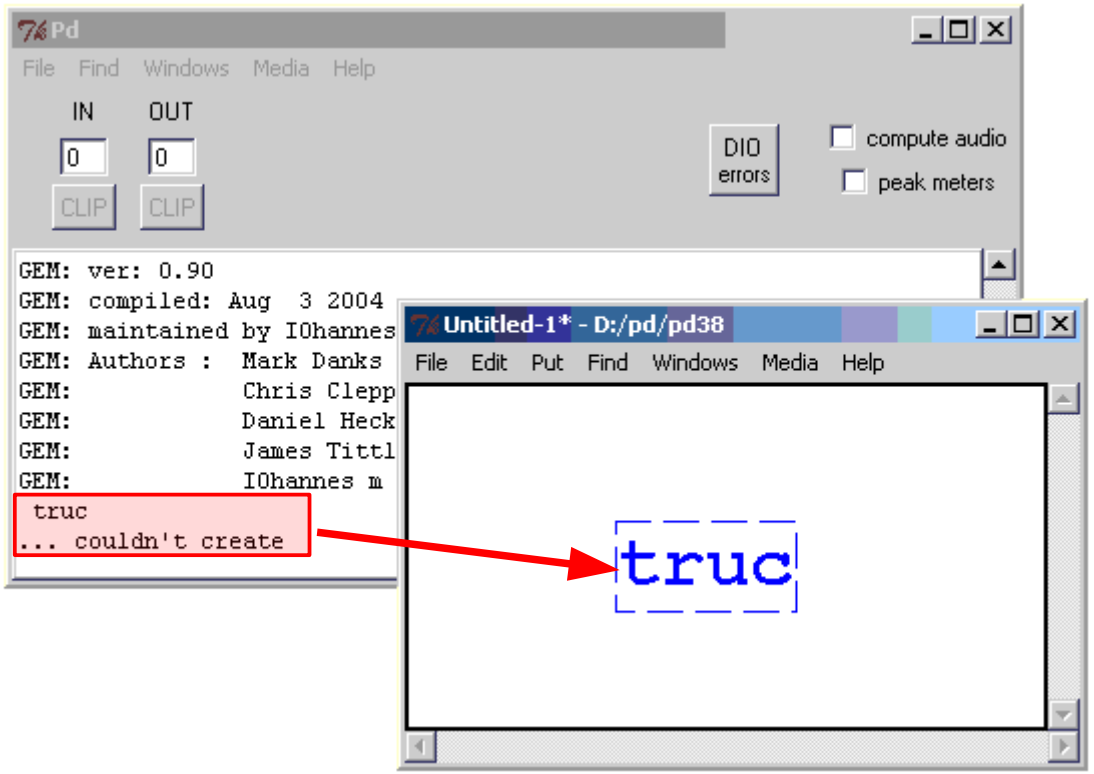


*Cet objet génère une oscillation périodique régulière de type cosinus (**osc~**). L'argument est **440**, il correspond à la fréquence de cette oscillation en période par seconde (Hertz). Cet objet a deux entrées et une sortie. (deux rectangles noirs en haut, et un en bas)*

Terminologie

OBJETS

*Quand un objet n'est pas reconnu, ces contours ne sont pas finis, des **tirets** l'entourent. Dans ce cas-là, un message apparaît dans la fenêtre de sortie : . . . `couldn't create`*



Terminologie

EXTERNS *(ou externals) : Ce sont des objets (écrits en C donc). Comme leur nom l'indique, ils sont exterieurs à la version de base de pure data, des ajouts. Pour les installer, la procédure la plus courante est de les copier dans le répertoire extra ou externs.*

Rappel des extensions des objets (natifs ou externes) : ***.pd_linux*** *ou* ***.pd_darwin*** *ou* ***.dll***

LIBRAIRIE *: Paquet d'externs et/ou d'abstractions. Copier le dossier où sont stockés les objets ou abstractions dans le répertoire extra ou externs. Rajouter si nécessaire le chemin, ou le nom de la librairie dans Menu > Path.*

http://ccrma.stanford.edu/planetccrma/software/pdworld.html : Pd Externals CCRMA

Terminologie

***ABSTRACTION** : C'est un nouveau patch dans lequel on a mis des objets, que l'on a enregistré et qui va être utile dans d'autres patchs. On crée un objet ctl+ shift + 1 dans un nouveau patch qui fera office de programme principal et on le nomme comme le fichier que l'on vient de sauvegarder. On peut regarder à l'intérieur et modifier le contenu d'une abstraction (clic droit + open ou double clic). C'est donc un fichier**.pd i**ndépendant. Si on la sauve dans le répertoire de travail, elle est reconnue tout de suite, sinon il faut la sauvegarder dans un répertoire et indiquer son chemin dans les options de démarrage : `-path ''D:\pd\MesAbstractions''` par exemple*

Voir : abstraction doc.

.

***PATCH, SOUS-PATCH et ABSTRACTION** : Quand on sauve un patch, on sauve le patch et ses sous patchs. On ne sauve pas les abstractions qui sont contenues dans le patch, elles doivent être sauvées indépendamment, en l'ouvrant et en la sauvant.*
Un patch peut aussi être considéré comme une abstraction dans la mesure où c'est un fichier.pd indépendant. On préfère utilisé le terme abstraction pour une fonction bien définie, le terme patch sera plutôt utilisé pour définir le programme principal.

Terminologie

SOUS-PATCH *: (créer un objet « **pd** nom_variable »). Fonctionne selon le principe des poupées russes, encapsulation, on isole une fonction pour permettre d'organiser le programme. Meilleure lisibilité.*

Dans cet exemple, les trois sous patchs sont tous contenus dans le patch principal sauvé sous le nom « ExemplesSousPatchs.pd ». Si on sauve dans n'importe quel sous patchs on sauve l'ensemble du patch. L'enregistrement sera visible dans la fenêtre de sortie. Les titres des fenêtres correspondent aux noms des sous patchs.