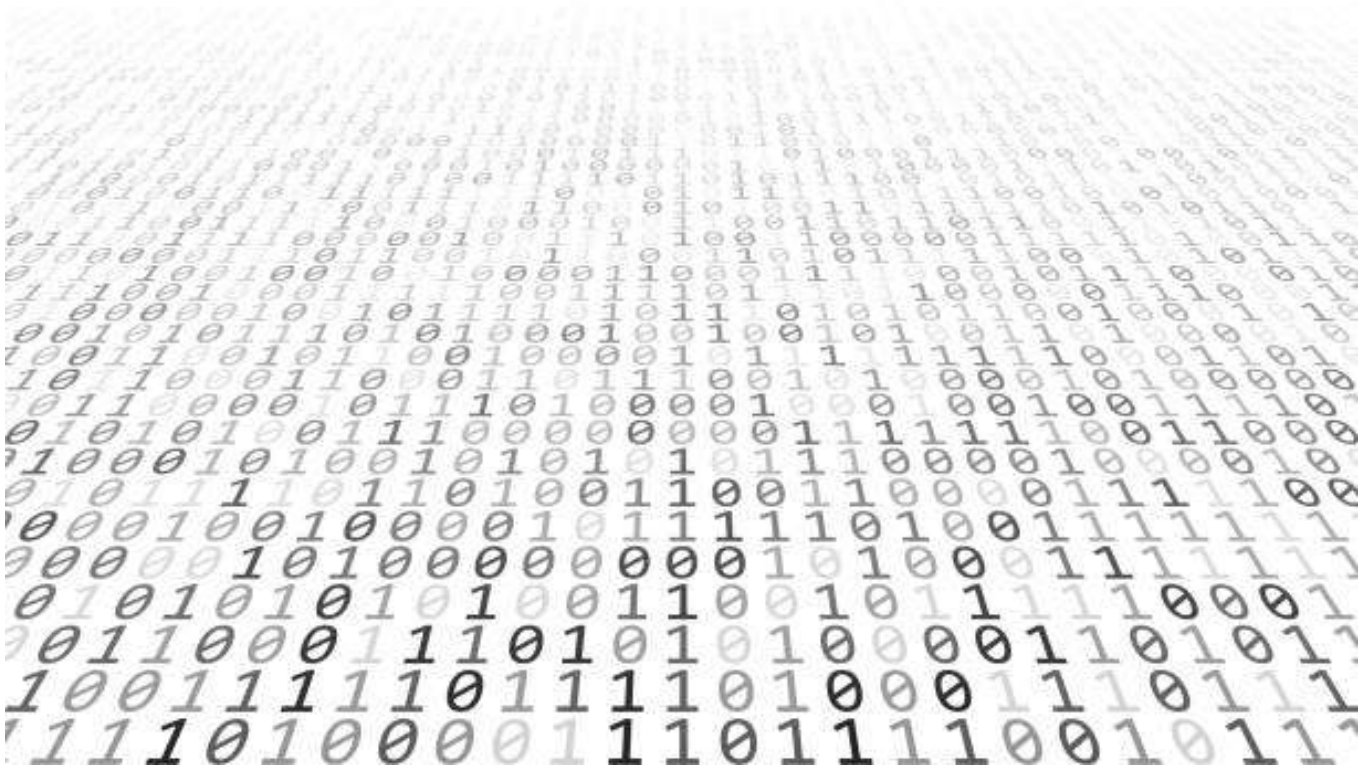
Εμμανουήλ Ροβίθης Ph.D.

Διδάσκων / Ερευνητής

Τμήμα Τεχνών Ήχου & Εικόνας

Ιόνιο Πανεπιστήμιο

Το Πρόγραμμα



Βασική Ιδέα

*Κάθε εντολή είναι ένα **αντικείμενο**.*

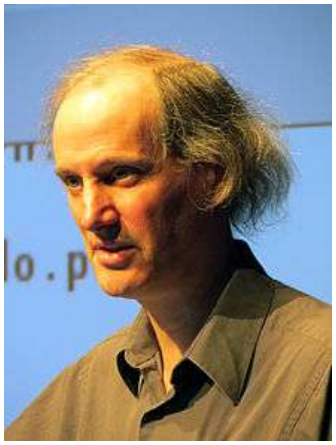
*Κάθε αντικείμενο **δέχεται** πληροφορία, την **επεξεργάζεται** και την **εξάγει**.*

*Ο τρόπος επεξεργασίας εξαρτάται από τις **παραμέτρους** του αντικειμένου.*

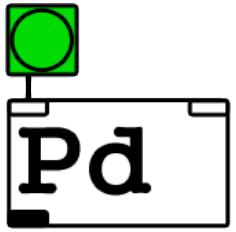
*Η **έξοδος** ενός αντικειμένου ενώνεται με την **είσοδο** ενός άλλου μέσω **καλώδιου**.*

Pure Data

- Οπτική γλώσσα προγραμματισμού
  - Χωρική γραμματική προγραμματιστικών στοιχείων
- Βασικός δημιουργός: Miller Puckette (δεκαετία 90)
- Δημιουργία διαδραστικών συστημάτων:
  - Ηλεκτρονική μουσική
  - Πολυμέσα
  - Αισθητήρες
  - LAN/Web συνδεσιμότητα σε-πραγματικό-χρόνο
- Σύνδεση με γλώσσα Max και πρόγραμμα Max/MSP



Pure Data



- Χρήσιμα Links

https://puredata.info

http://pd-tutorial.com/english/index.html

http://msp.ucsd.edu/Pd_documentation/

http://msp.ucsd.edu

- Έκδοση

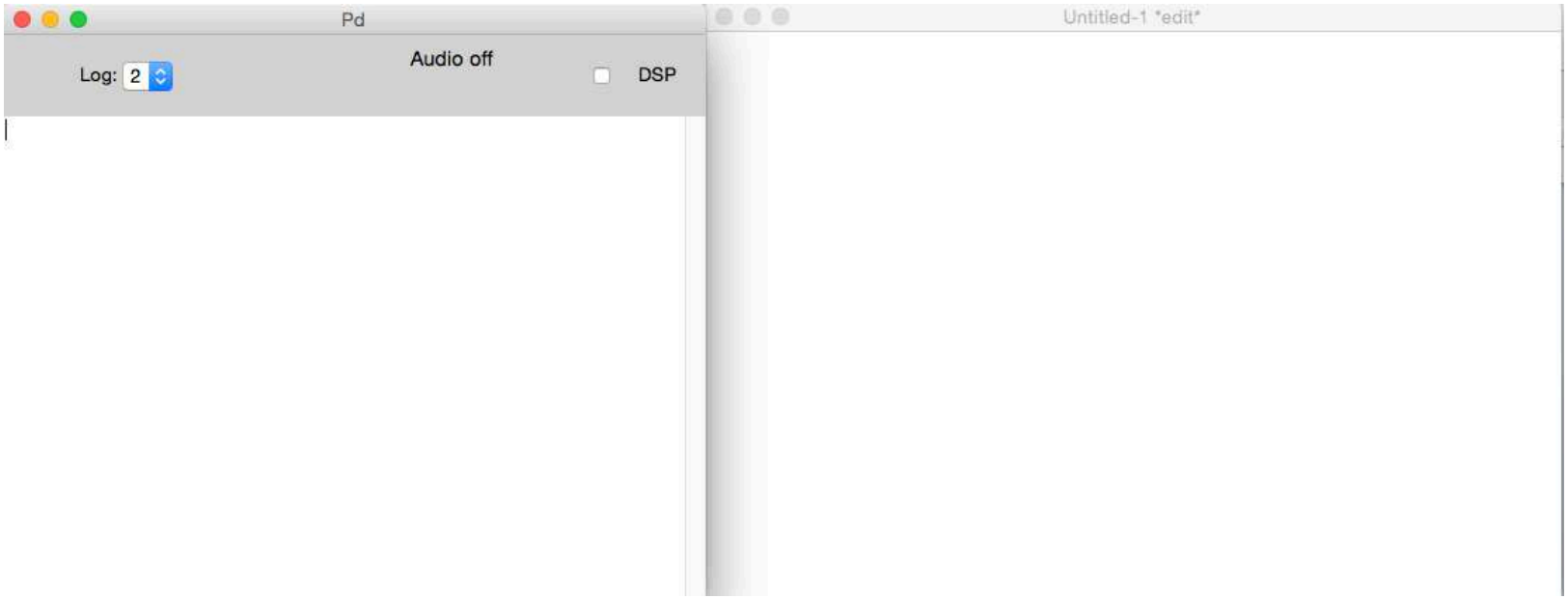
Vanilla 0.48-1

Προγραμματιστικό Περιβάλλον

Ο προγραμματισμός λαμβάνει χώρα στο παράθυρο του καμβά (**patcher** window),

ενώ στην κονσόλα εμφανίζονται μηνύματα του αντικειμένου **print** ή ειδοποιήσεις για σφάλματα του κώδικά μας.

Θα αναφερόμαστε στο patch μας ως “πρόγραμμα”



Επεξεργασία και Εκτέλεση Προγράμματος

Για να προγραμματίσουμε χρειάζεται να θέσουμε το παράθυρο του καμβά σε κατάσταση επεξεργασίας (**edit** mode). Έπειτα, το κλειδώνουμε σε κατάσταση λειτουργίας (**run** mode), ώστε να ενεργοποιήσουμε το περιεχόμενό του.

Η εναλλαγή γίνεται είτε από το μενού “edit”, είτε με τη συντόμευση ctrl+e / cmd+e

edit mode

run mode

Εισαγωγή Στοιχείων

Τα στοιχεία του προγράμματος εισάγονται από το μενού “Put”.

Παρατηρούμε πως εναλλακτικά μπορούμε να χρησιμοποιήσουμε συντομεύσεις.

Θεμελιώδη Στοιχεία

bang

για πραγματοποίηση μιας ενέργειας

message

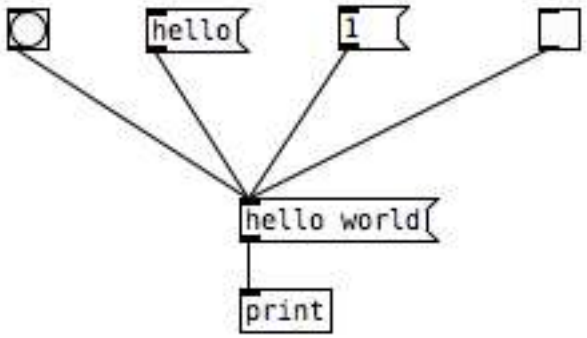
για μετάδοση μηνύματος

number box

για εισαγωγή αριθμού

toggle

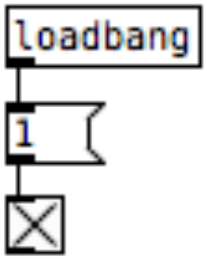
για διακόπτη 1 και 0



loadbang

Το αντικείμενο **loadbang** παράγει ένα bang κατά την αρχική φόρτωση του προγράμματος.

Είναι χρήσιμο για αρχικοποίηση των παραμέτρων του προγράμματός μας.



Εισαγωγή Αντικειμένων

Όταν εισάγουμε ένα αντικείμενο, καλούμαστε να γράψουμε το **όνομα** του στο κενό πεδίο εντός των διακεκομμένων ορίων.

Αν το αντικείμενο υπάρχει, **το πλαίσιο κλείνει**, και εμφανιζονται οι ανάλογες προς το αντικείμενο **είσοδοι** και **έξοδοι**.

Ζεστή και Κρύα Είσοδος

Τα περισσότερα αντικείμενα του Pure Data, έχουν **δύο ειδών εισόδους**:

- την κρύα (**cold inlet**), η οποία αποθηκεύει την εισαγόμενη πληροφορία χωρίς να πυροδοτεί περαιτέρω ενέργεια, και
- τη ζεστή (**hot inlet**), η οποία αποθηκεύει την εισαγόμενη πληροφορία και την προωθεί παρακάτω στον αλγόριθμο

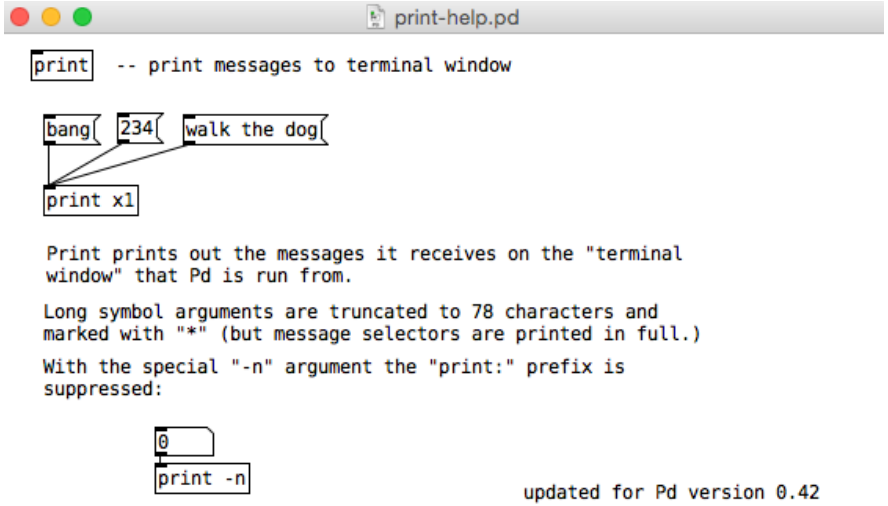
Η ζεστή είσοδος είναι η αριστερή

hot inlet cold inlet

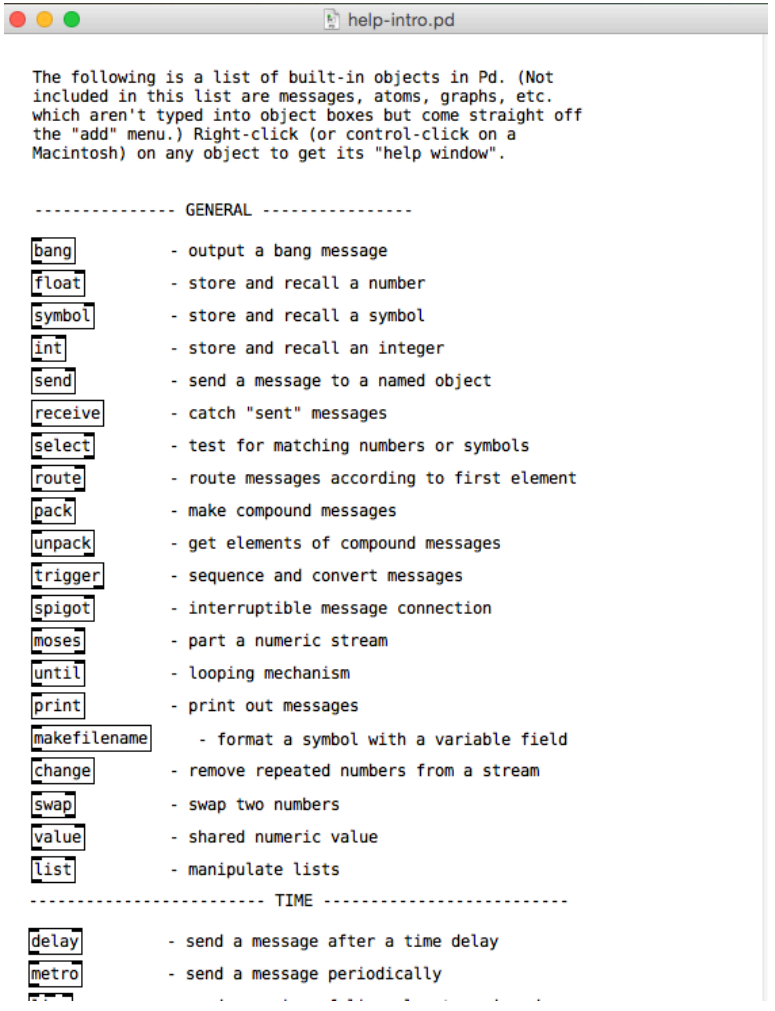
Βοήθεια

Κάθε αντικείμενο ξεχωριστά διαθέτει αρχείο βοήθειας (**help** file) (δεξί κλικ στο αντικείμενο.

Το αρχείο βοήθειας αποτελεί λειτουργικό patch, μπορούμε να το ξεκλειδώσουμε και διαχειριστούμε.



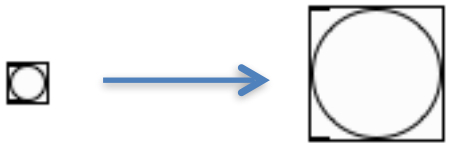
Με επιλογή help από τον καμβά, εμφανίζεται λίστα με το σύνολο των **διαθέσιμων αντικειμένων**

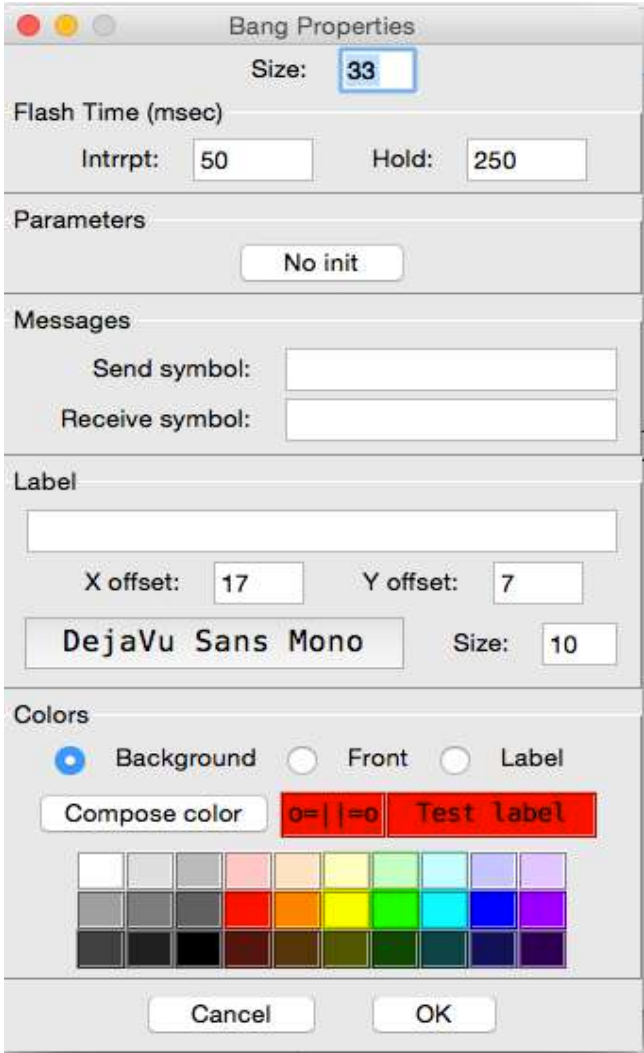


Ιδιότητες

Το χρώμα, το μέγεθος, και άλλα στοιχεία πολλών αντικειμένων,

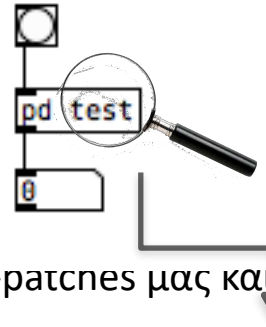
τροποποιούνται μέσα από το παράθυρο ιδιοτήτων τους (**properties**) (δεξί κλικ στο αντικείμενο)



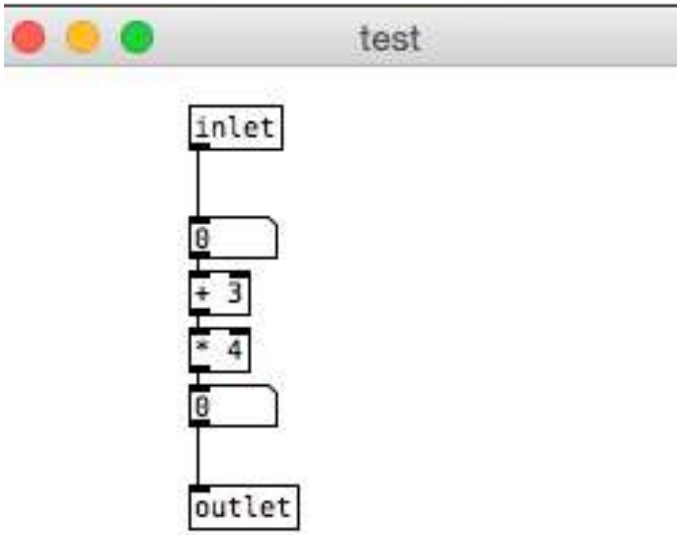


Υπο-προγράμματα

Μπορούμε να οργανώνουμε τμήματα του προγράμματός μας σε υπο-προγράμματα (**sub-patches**), αρκεί να εισάγουμε ένα νέο αντικείμενο με το πρόθεμα “**pd**” δίνοντας έπειτα από κενό ένα όνομα της αρεσκείας μας.



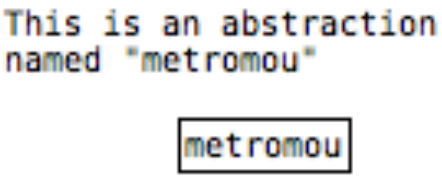
Η επικοινωνία ανάμεσα στα διάφορα επίπεδα των sub-patches μας και του καμβά μας πραγματοποιείται μέσω **εισόδων** και **εξόδων**, των αντικειμένων δηλαδή **inlet** και **outlet**



Αφαιρέσεις

Οι αφαιρέσεις (**abstractions)** είναι υπο-κατηγορία των **externals**, δηλαδή αντικειμένων που δεν περιέχονται εξαρχής στο Pure Data. Τα abstractions είναι patches αποθηκευμένα σε κάποιο path που ελέγχει το πρόγραμμα (pd>>preferences>>path) και μπορούμε να τα ανακαλούμε σε άλλα patches εισάγοντας σε ένα object απευθείας το όνομά τους,

Έτσι, μπορούμε να δημιουργήσουμε τη δική μας τράπεζα εργαλείων, αλλά και να συνεισφέρουμε στην κοινότητα.



Συμβουλή:

Με την προσθήκη σχολίων (**comments**) μπορούμε να κάνουμε τα patches μας πιο επεξηγηματικά και εύχρηστα.

Plugins

Άλλη υπο-κατηγορία των **externals** είναι τα plugins, προγράμματα συντεταγμένα σε άλλες γλώσσες και συμβατά με το Pd. Ανάλογα με το είδος του plugin υπάρχουν διαφορετικοί τρόποι να τα εισάγουμε στο πρόγραμμά μας. Συγκεκριμένα τα plugins που αφορούν την εξατομίκευση του GUI έχουν συνταχθεί στη γλώσσα Tcl/tk

http://www.tcl.tk

και χρειάζεται να είναι αποθηκευμένα σε κάποιο path που ελέγχει το πρόγραμμα (pd>>preferences>>path) για να φορτωθούν κατά την έναρξη.

Το plugin "**Ketea-theme**" μας επιτρέπει να αλλάξουμε χρώμα στα στοιχεία του προγραμματιστικού περιβάλλοντος

από: https://github.com/Monetus/Ketea-theme-plugin

(αφαιρέστε τα: obj_fill-plugin.tcl και object_db.dict)

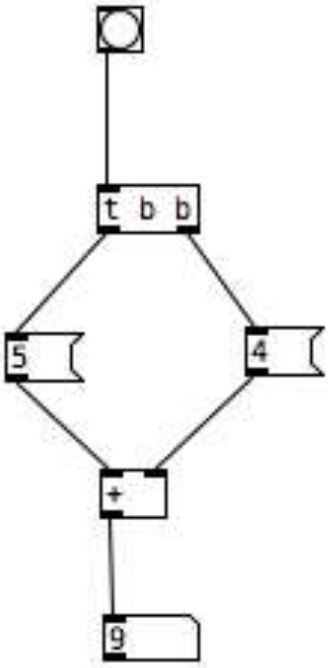
Plugins

Παράδειγμα tcl.tk script για εμφάνιση μηνύματος κατά την έναρξη, και αλλαγή χρώματος του background. Το κείμενο μετά τη δίεση είναι σχόλιο που αν ενεργοποιηθεί, εμφανίζει χρωματική παλέτα για επιλογή.

```
::pdwindow::post  "THIS IS AN EXAMPLE"
::tk_setPalette "green"
# ::tk_setPalette [tk_chooseColor]
```

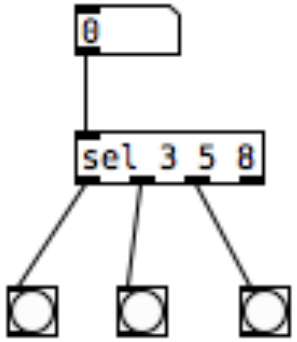
Τελεστές Ροής: trigger

Με το αντικείμενο **trigger** μπορούμε να ελέγξουμε τη ροή των δεδομένων, καθώς κατά την ενεργοποίηση αποστέλλει τα ορίσματά του με σειρά από δεξιά προς αριστερά



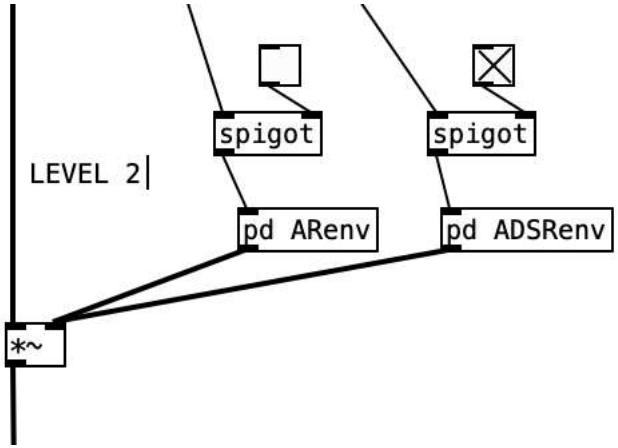
Τελεστές Ροής: select

Το αντικείμενο **select** ελέγχει την εισερχόμενη πληροφορία και, εάν αυτή ταιριάζει με κάποιο από τα ορίσματά του, παράγει **bang** στην αντίστοιχη έξοδο. Σε κάθε άλλη περίπτωση, η πληροφορία περνάει από την τελευταία έξοδο του αντικειμένου.



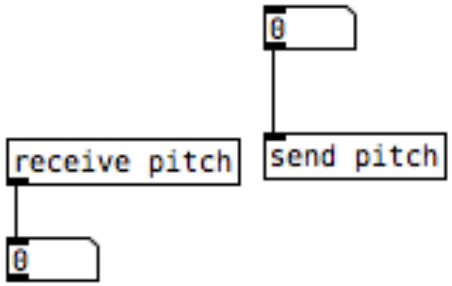
Τελεστές Ροής: spigot

Το αντικείμενο **spigot** λειτουργεί ως πύλη. Δεχόμενη την τιμή 1 στην κρύα της είσοδο ανοίγει τη ζεστή είσοδο για να περάσουν τα δεδομένα, ενώ δεχόμενη την τιμή 0 η δίοδος κλείνει.



send & receive

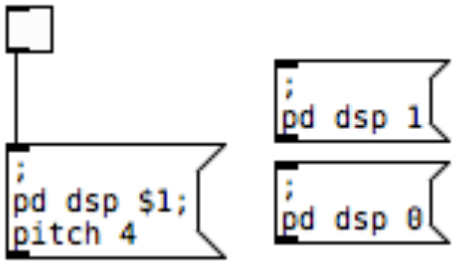
Τα αντικείμενα **send** και **receive** επιτρέπουν τη ροή δεδομένων ανάμεσα στα πολλαπλά επίπεδα ενός patcher αρκεί να μοιράζονται το ίδιο όνομα.



send & receive

Ροή πληροφορίας με μαζικό τρόπο προς όλα τα επίπεδα του patcher μας επιτυγχάνεται με το σύμβολο του ελληνικού ερωτηματικού ” **;** “ σε μήνυμα, ακολουθούμενο από το όνομα του αντικειμένου και τα στοιχεία προς αποστολή. Στο παρακάτω παράδειγμα θα σταλεί η τιμή 4 στο αντικείμενο **receive** με το όνομα pitch.

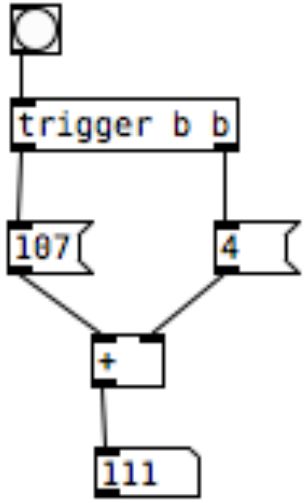
Με το πρόθεμα “ **pd** “ μπορούμε να στείλουμε δεδομένα στο ίδιο το σύστημα του προγράμματος, όπως στο παρακάτω παράδειγμα, όπου τα μηνύματα 1 και 0 ενεργοποιούν και απενεργοποιούν το dsp.



Αριθμητικοί Τελεστές

Οι αριθμητικοί τελεστές πραγματοποιούν αριθμητικές πράξεις ανάμεσα στις τιμές των εισόδων τους.

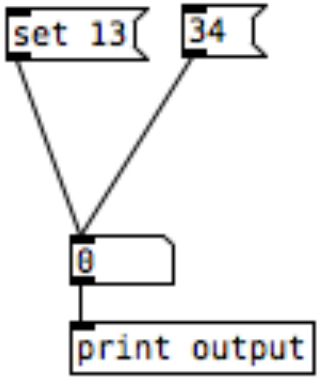
Προσοχή πως η τιμή στη ζεστή είσοδο θα πραγματοποιήσει την πράξη.



Εντολή Μηνύματος: set

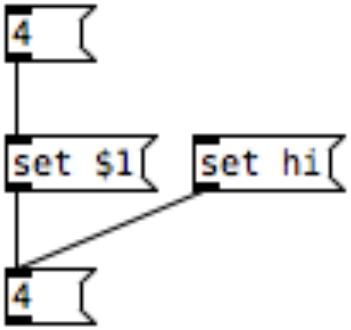
Εισάγοντας ένα μήνυμα με το πρόθεμα **set** η πληροφορία διαμοιράζεται χωρίς περαιτέρω πυροδότηση ενέργειας. Στο παρακάτω παράδειγμα, το **message 34** θα θέσει την τιμή στο **number box** και θα την στείλει στο αντικείμενο **print**, ενώ η εντολή **set 13** θα θέσει την τιμή, χωρίς να την προωθήσει στο επόμενο αντικείμενο.

Μήνυμα **set** χωρίς άλλο περιεχόμενο σβήνει το περιεχόμενο ενός άλλου **message**.



Μεταβλητές

Το σύμβολο **$** επιτρέπει τη δημιουργία μεταβλητών.



Προσαρμογή Εύρους

Για πιο σύνθετη προσαρμογή ενός εύρους τιμών σε ένα άλλο, χρησιμοποιούμε τη φόρμουλα:

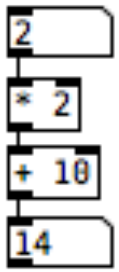
max = min + steps*x

Έστω πως θέλουμε να μεταφέρουμε γραμμικά το εύρος τιμών “0 έως 5” στο “10 έως 20”

Αρχικά χρησιμοποιούμε τη φόρμουλα για τον υπολογισμό του συντελεστή **x**

20 = 10 + 5x >> 10 = 5x >> x = 2

Έπειτα πολλαπλασιάζουμε κάθε τιμή του αρχικού εύρους με τον συντελεστή **x** και προσθέτουμε το επιθυμητό **min**



Σε περίπτωση που το αρχικό εύρος δεν αρχίζει από μηδενική τιμή,

π.χ. από “1 έως 6” σε “10 έως 20” χρειάζεται να μηδενίσουμε την αρχική τιμή πριν τον υπολογισμό

Προσαρμογή Εύρους

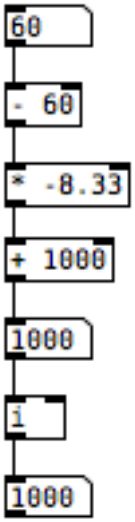
Η ίδια μέθοδος ισχύει και για αντίστροφη προσαρμογή, δηλαδή το **min** ενός εύρους τιμών να αντιστοιχιστεί με το **max** ενός άλλου.

Σε αυτή την περίπτωση, ο συντελεστής είναι αρνητικός.

Στο παρακάτω παράδειγμα μεταφέρεται γραμμικά το εύρος “60 έως 120”

στο εύρος τιμών “1000 έως 500”

```
60 - 120 >> 1000-500
500 = 1000 + 60x
-500 = 60x
-8.33 = x
```



Για αποφυγή δεκαδικών

χρησιμοποιούμε το αντικείμενο **i (integer)**,

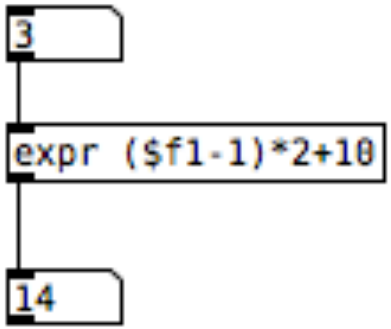
το οποίο προσαρμόζει την είσοδό του στον πιο κοντινό ακέραιο προς το μηδέν.

Τελεστές Αριθμητικών Πράξεων: expr

Το αντικείμενο **expr** συμπυκνώνει αριθμητικές πράξεις σε μια γραμμή εντολών και τις εκτελεί σε μεταβλητές. Στο παρακάτω παράδειγμα οι δύο προγραμματιστικές δομές είναι ταυτόσημες. Προσέξτε πως στο expr χρειάζεται να εισάγουμε τη μεταβλητή διαφορετικά απ' ό,τι συνήθως:

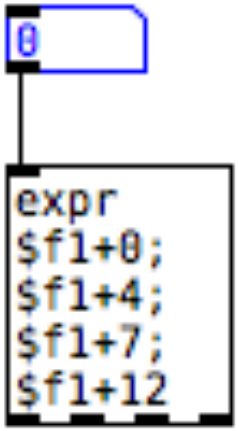
$f1

υποδηλώνοντας πως η αναμενόμενη μεταβλητή είναι δεκαδικός αριθμός (f = float)



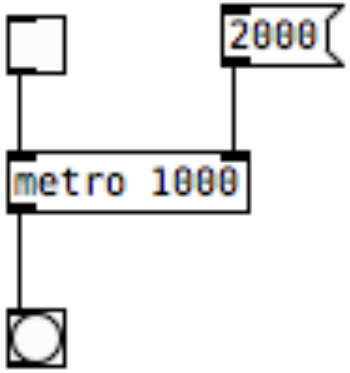
expr

Μπορούμε να εκτελέσουμε με το ίδιο **expr** διαφορετικές πράξεις με κάθε μια να παράγει το δικό της αποτέλεσμα. Διαχωρίζουμε κάθε πράξη με το σύμβολο “**;**” και δημιουργείται αυτόματα η αντίστοιχη έξοδος.



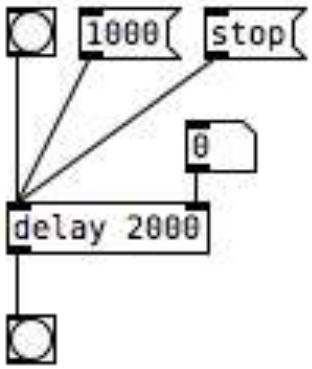
Τελεστές Χρόνου: metro

Το αντικείμενο **metro** παράγει **bangs** με συγκεκριμένη περιοδικότητα. Ως default κλίμακα, το **metro** δέχεται millisecs, είτε ως αρχικό όρισμα είτε στην κρύα είσοδο.



Τελεστές Χρόνου: delay

Το αντικείμενο **delay** καθυστερεί τα εισερχόμενα bangs για τον χρόνο του ορίσματός του (default σε milliseconds) ή για τον χρόνο που θα εισάγουμε στις εισόδους του αντικειμένου. Με **bang** ή **message** στη ζεστή είσοδο ενεργοποιείται η διαδικασία, με μήνυμα **stop** σταματάει.

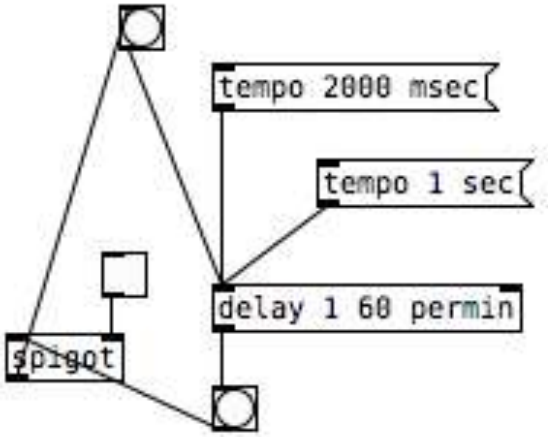


Τελεστές Χρόνου: tempo

Μπορούμε να ρυθμίσουμε το **delay** σύμφωνα με την επιθυμητή κλίμακα μέτρησης του χρόνου είτε με ορίσματα είτε στέλνοντας την εντολή **tempo**.

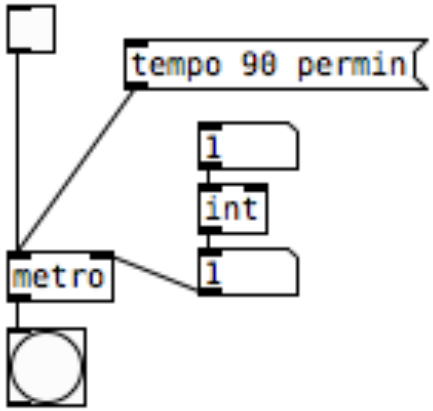
Στο παρακάτω παράδειγμα και με τους δύο τρόπους ορίζουμε τον χρόνο καθυστέρησης σε 1 δευτερόλεπτο (1 beat στην κλίμακα των 60 beats ανά λεπτό, 1 sec).

Η εντολή **tempo** συνδυάζεται και με msec, min, samp.



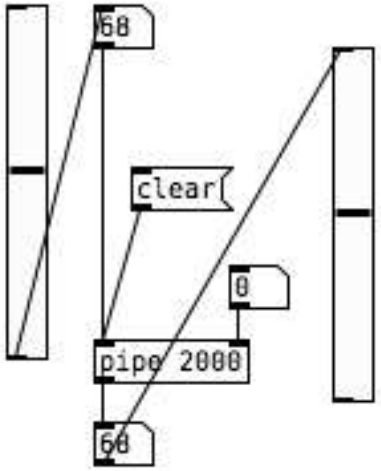
Τελεστές Χρόνου: tempo

Η εντολή **tempo** μπορεί να εφαρμοστεί και στο αντικείμενο **metro** θέτοντας την κλίμακα εκπομπής του, ενώ στη δεξιά είσοδο ορίζεται η μονάδα εκπομπής. Ο συνδυασμός τους διαμορφώνει τον ρυθμό του μετρονόμου, όπως στο παρακάτω παράδειγμα, όπου ο μετρονόμος θα εκπέμπει σε κάθε κτύπο με συχνότητα 90 κτύπους ανά λεπτό.



Τελεστές Χρόνου: pipe

Το αντικείμενο **pipe** καθυστερεί τιμές αντί για bangs. Στο παρακάτω παράδειγμα, κάθε κίνηση στον αριστερό ολισθητή (**vslider**) θα επαναληφθεί στον δεξιό μετά από 2 δευτερόλεπτα. Το μήνυμα **clear** σταματάει κάθε δρομολογημένη καθυστέρηση.

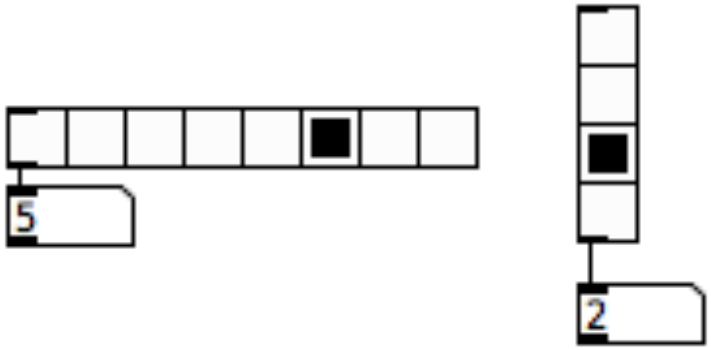


Διεπαφές: Κελιά

Τα αντικείμενα **Hradio** και **Vradio** δημιουργούν αντίστοιχα οριζόντια και κάθετα μενού με κελιά.

Το πλήθος των κελιών ορίζεται από τις ιδιότητες των αντικειμένων.

Κάθε κελί εκπέμπει και μια αντίστοιχη τιμή, αρχής γενομένης από το 0.

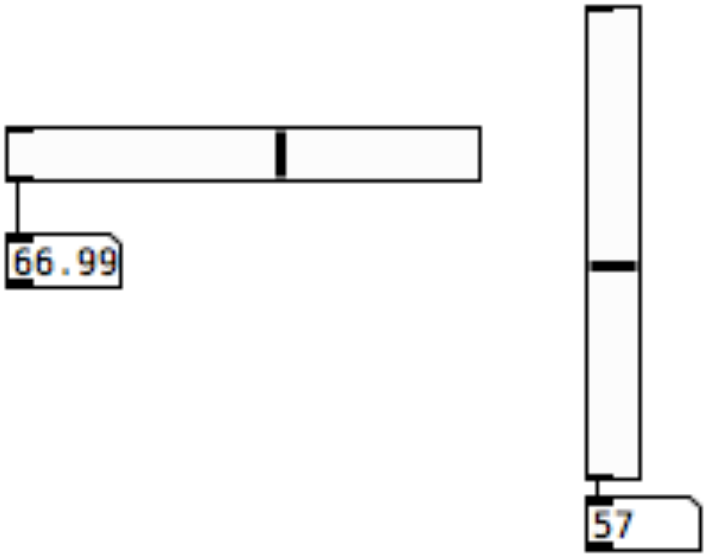


Διεπαφές: Ολισθητές

Τα αντικείμενα **Hslider** και **Vslider** δημιουργούν αντίστοιχα οριζόντιους και κάθετους ολισθητές.

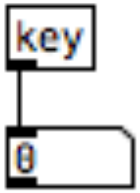
Το εύρος των εξαγόμενων τιμών ορίζεται από τις ιδιότητες των αντικειμένων.

Στην αρχική τους κατάσταση, οι ολισθητές εξάγουν ακέραιες τιμές 0 έως και 127



Διεπαφές: Πλήκτρα

Το αντικείμενο **key** αναφέρει την ASCII τιμή των πλήκτρων

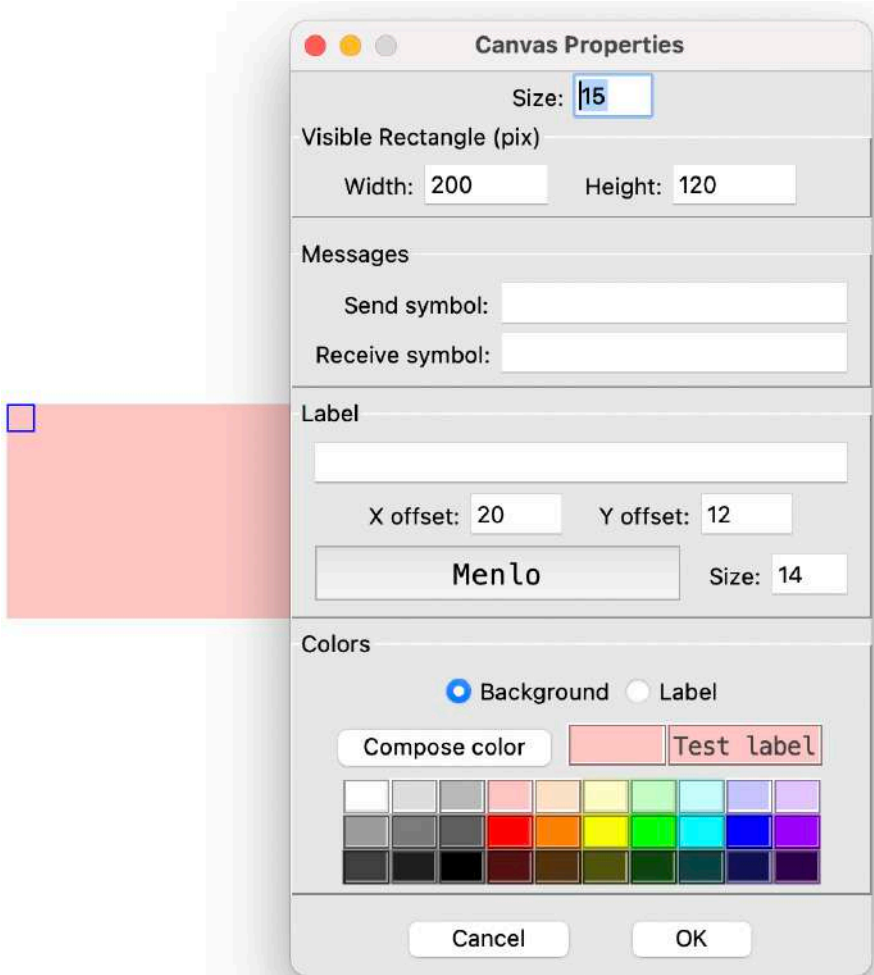


Περιβάλλον Διεπαφής: canvas

Το αντικείμενο **canvas** (menu: put > canvas) επιτρέπει την εισαγωγή “επιφανειών”, τις οποίες τροποποιούμε ως προς μέγεθος και χρώμα μέσω των ιδιοτήτων του αντικειμένου.

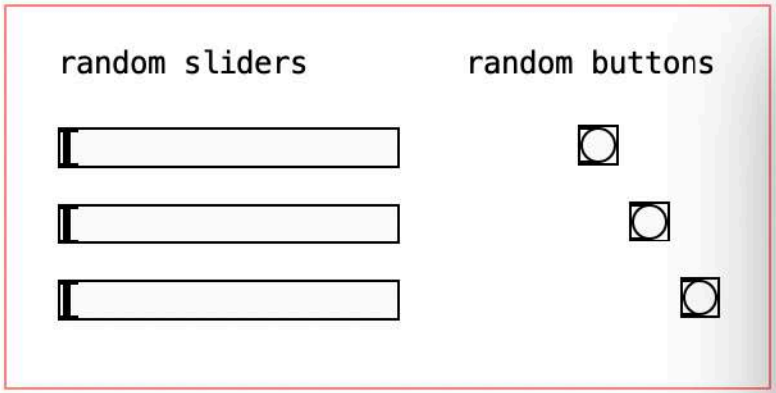
Προσοχή στο ότι το **canvas** θα αποκρύψει κάθε αντικείμενο που είχε δημιουργηθεί πριν την εισαγωγή του.

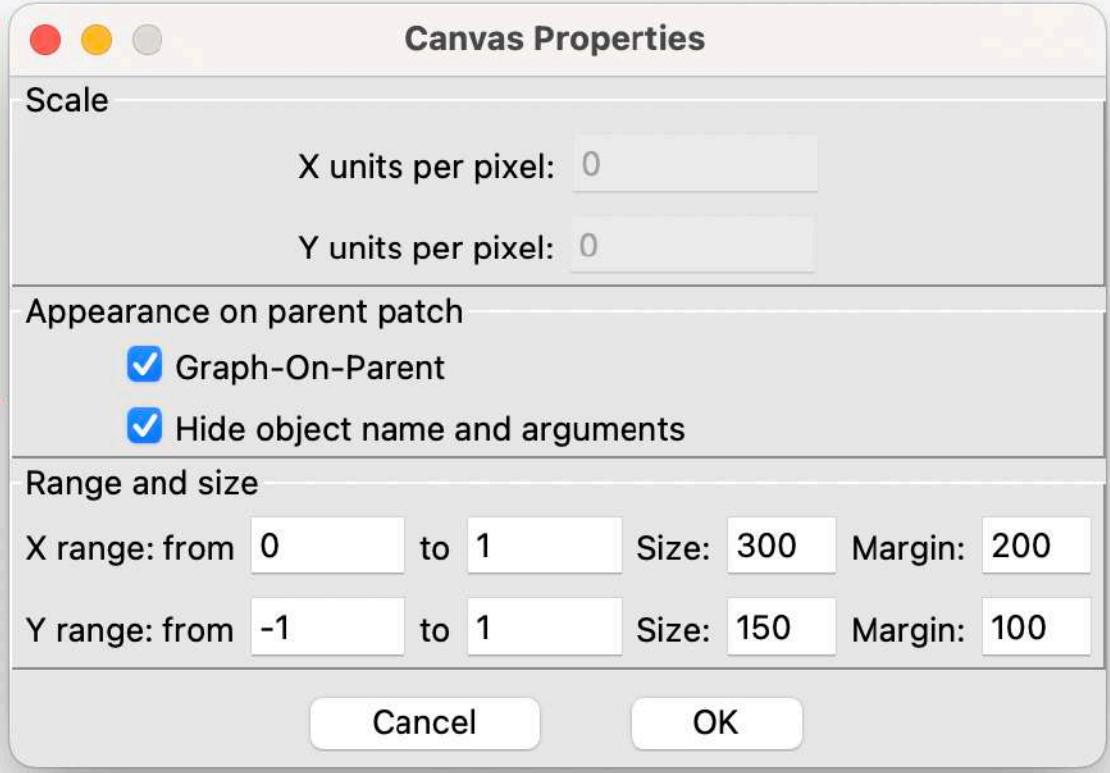
Για να στείλουμε το canvas στο background θα χρειαστεί τα αντικείμενα που θέλουμε να τοποθετήσουμε πάνω του να τα αποκόψουμε και επικολλήσουμε ξανά.



Περιβάλλον Διεπαφής: graph-on-parent

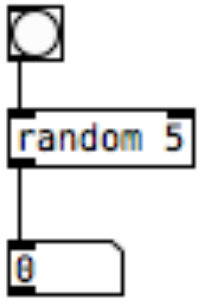
Μπορούμε σε ένα **sub-patch** να ανοίξουμε ένα “παράθυρο προβολής” προς ένα επίπεδο πάνω. Στις ιδιότητες του κενού καμβά ενεργοποιούμε την επιλογή “**Graph-On-Parent**” και εμφανίζουμε ένα πλαίσιο. Ορίζουμε τις διαστάσεις (size-x, size-y) και τη θέση (margin-x, margin-y) του πλαισίου, και μεταφέρουμε τα αντικείμενα που θέλουμε να φαίνονται στον χρήστη, όταν κλείσουμε το **sub-patch**.





Τυχαιότητα: random

Το αντικείμενο **random** παράγει τυχαίες τιμές από το μηδέν έως την τιμή κατά μια μονάδα μικρότερη του ορίσματός του. Στο παρακάτω παράδειγμα, με κάθε **bang**, το **random** θα παράγει μια τιμή από 0 έως 4 με ίση πιθανότητα εμφάνισης.



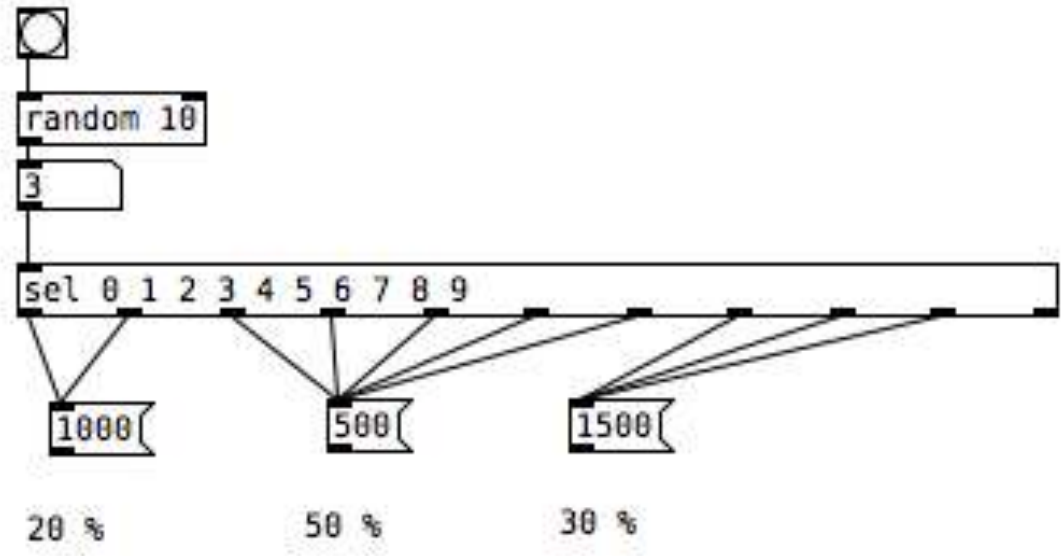
Πιθανοκρατικές Δομές: random + select

Συνδυάζοντας τα αντικείμενα **random** και **select** δημιουργούμε πιθανοκρατικές δομές.

Το **random** παράγει τυχαίες τιμές, τις οποίες ελέγχει το **select**, και εμείς ορίζουμε τί θα ενεργοποιείται σε κάθε έλεγχο.

Στο παρακάτω παράδειγμα, το **random** παράγει 10 διαφορετικές τιμές, από τις οποίες:

- 2 ενεργοποιούν την τιμή 1000, συνεπώς με 20% πιθανότητα,
- 5 την τιμή 500, συνεπώς με 50% πιθανότητα, και
- 3 την τιμή 1500, συνεπώς με 30% πιθανότητα



Πιθανοκρατικές Δομές: σχεσιακοί τελεστές

Ένας ακόμη τρόπος για πιθανοκρατική οργάνωση είναι μέσω **σχεσιακών τελεστών.**

Πρόκειται για αντικείμενα που ελέγχουν 2 τιμές ως προς μια συνθήκη.

Παράγουν την τιμή 1, εάν η δήλωση υπό έλεγχο είναι αληθής,

και την τιμή 0, εάν είναι ψευδής.

Στο παρακάτω παράδειγμα, το αντικείμενο **random** παράγει τιμές από 0 έως 9,

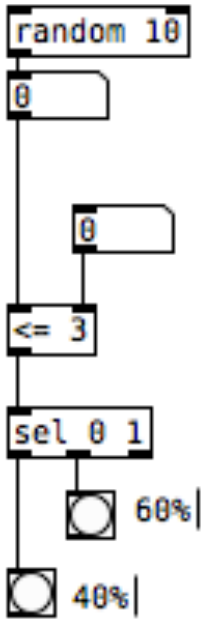
οι οποίες ελέγχονται από έναν σχεσιακό εκτελεστή '**<=**'

και διαχωρίζονται στο εύρος τιμών από 0 έως 3 (μικρότερο-ίσο του 3)

που έχει 40% πιθανότητα να προκύψει,

και στο εύρος τιμών από 4 εώς 9 (μεγαλύτερο του 3)

με πιθανότητα εμφάνισης 60%



Πιθανοκρατικές Δομές: λογικοί τελεστές

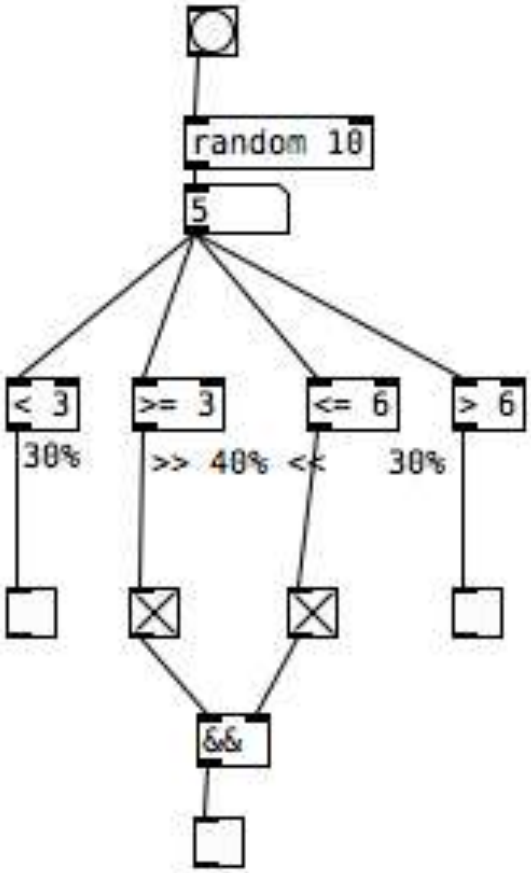
Για την απομόνωση ενδιάμεσου εύρους τιμών

ελέγχουμε τα αποτελέσματα σχεσιακών τελεστών μέσω **λογικών τελεστών**.

Ο λογικός τελεστής **&&** εξάγει την τιμή 1,

μόνο όταν δεχτεί και στις δύο εισόδους μη-μηδενικές τιμές.

Σε κάθε άλλη περίπτωση εξάγει την τιμή 0.

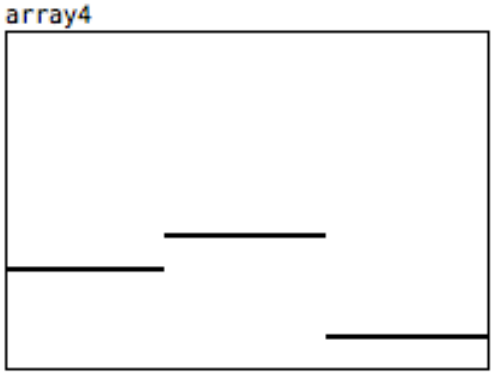


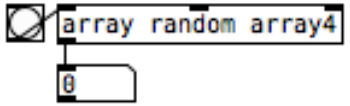
Πιθανοκρατικές Δομές: πίνακας array

Για εύκολη διαχείριση πιθανοκρατικών δομών σε πραγματικό χρόνο χρησιμοποιούμε το αντικείμενο **array**. Θα δούμε και παρακάτω πως το αντικείμενο **array** είναι ουσιαστικά ένας πίνακας αποθήκευσης και αναπαράστασης τιμών

Εισάγουμε το **array** (μενού>put>array) και το ονοματίζουμε, ώστε να κατευθύνουμε την εντολή **array random** σε αυτό. Κάθε **bang** παράγει μια τυχαία τιμή από το μέγεθος (x-indexes 0, 1, 2) του **array** "array4" σύμφωνα με την πιθανοκρατική σχέση που ορίζουν οι y-τιμές που έχουν ανατεθεί στα x-indexes.

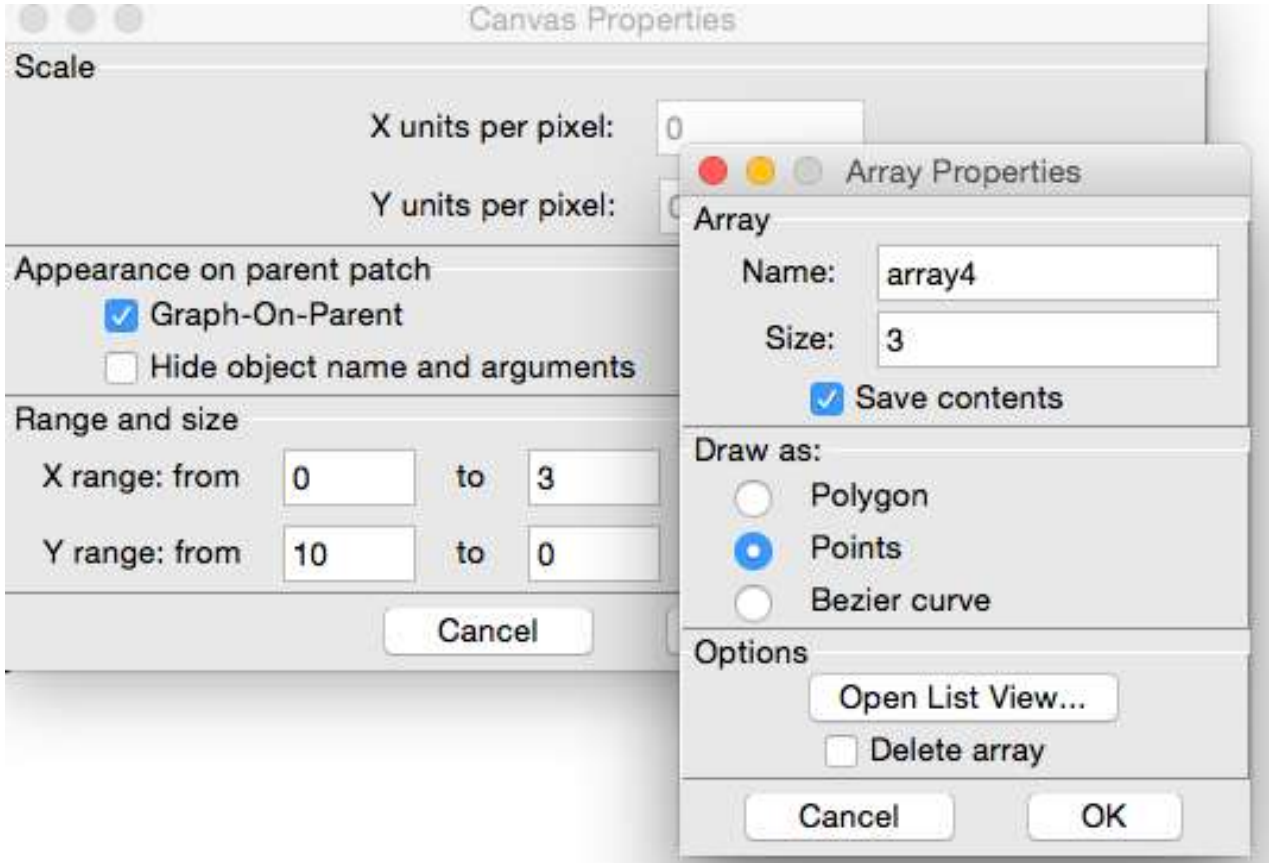
Συνεπώς, μια θέση a (άξονας-x) με τιμή i (άξονας-y), θα έχει διπλάσια πιθανότητα να προκύψει σε σχέση με μια θέση b με τιμή i/2.





Πιθανοκρατικές Δομές: πίνακας array

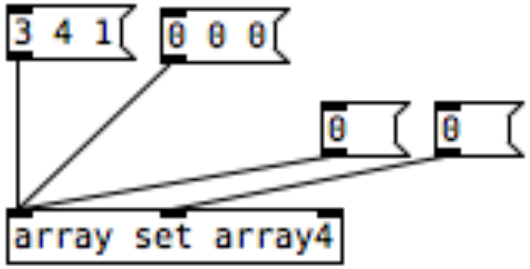
Το μέγεθος του **array**, δηλαδή το πλήθος των τιμών στον άξονα-x, και το εύρος του **array**, δηλαδή οι πιθανές τιμές που μπορούν να τους ανατεθούν στον άξονα-y, ορίζονται μέσα απο τα **properties** του αντικειμένου στα **size** και **Y range** αντίστοιχα.



Πιθανοκρατικές Δομές: πίνακας array

Εκτός από το να σχεδιάζουμε το περιεχόμενο του **array** απ' ευθείας στο διαθέσιμο πλαίσιο, μπορούμε να εισάγουμε τα περιεχόμενά του και με **messages** και **number boxes** μέσω της εντολής **array set**, κάτι που μας επιτρέπει την αποτύπωση με ακρίβεια προετοιμασμένων δεδομένων, αλλά και άλλες διαδικασίες, όπως τον μηδενισμό του πίνακα. Έτσι, μπορούμε να εισάγουμε μέσω **message** μια λίστα τιμών, οι οποίες θα αποδοθούν αυτόματα στις αντίστοιχες θέσεις του **array** ξεκινώντας από τη θέση 0.

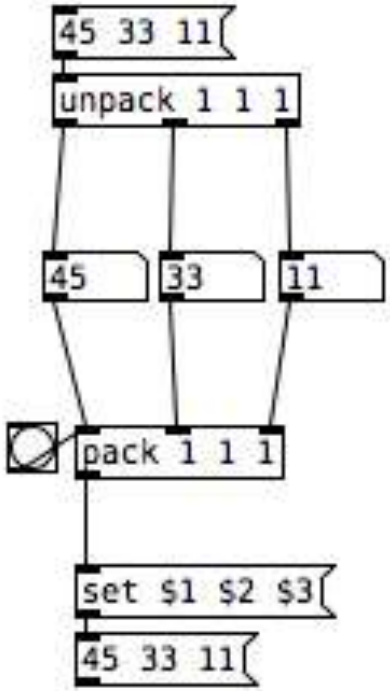
Εναλλακτικά, μπορούμε να επιλέγουμε μια επιθυμητή θέση στη δεύτερη είσοδο του αντικειμένου και να της αποδίδουμε έπειτα μια τιμή στην πρώτη είσοδο.



Λίστες

Μπορούμε να συνδυάσουμε περισσότερα μηνύματα σε μια λίστα διαχωρίζοντας τα με κενό.

Τα αντικείμενα **pack** και **unpack** συναρμολογούν και αποσυναρμολογούν αντίστοιχα λίστες από και στα επιμέρους τους συστατικά. Τα ορίσματά τους αντιστοιχούν στις αναμενόμενες εισόδους, εδώ στο παράδειγμα σε ακέραιους αριθμούς.



MIDI δεδομένα

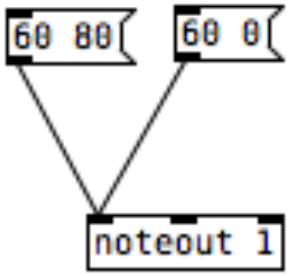
Το Pd μπορεί να δέχεται και να εκπέμπει δεδομένα **midi**.

Συγκεκριμένα, το αντικείμενο **noteout** δέχεται μηνύματα αποτελούμενα από:

μια **midi νότα** και μια **midi ένταση**.

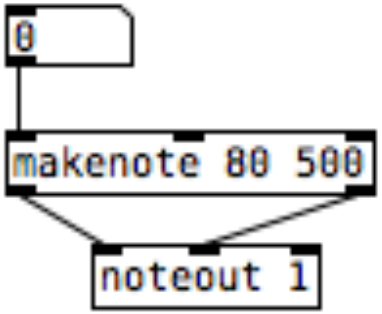
Αν η νότα δεν προβλέπει εκ φύσεως σίγαση στην εξέλιξή της, χρειάζεται να αποσταλεί ένα δεύτερο **message** με το ίδιο midi νούμερο και μηδενική ένταση.

Το όρισμα στο αντικείμενο σηματοδοτεί το midi κανάλι, όπου θα αποσταλεί η πληροφορία



MIDI δεδομένα

Το αντικείμενο **makenote** εξαλείφει την ανάγκη αποστολής δύο μηνυμάτων για κάθε νότα, αφού δημιουργεί αυτόματα μια midi νότα με ένταση σύμφωνη με το πρώτο του όρισμα, και μια ακόμη midi νότα με μηδενική ένταση, όταν παρέλθει η διάρκεια που έχει εισαχθεί ως δεύτερο όρισμα (σε mlsecs)

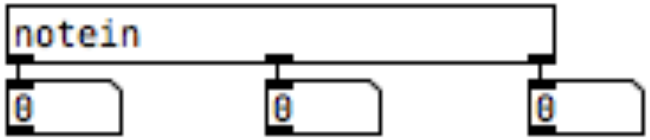


MIDI δεδομένα

Το αντικείμενο **pgmout** ελέγχει το midi οργάνου που θα αναπαράγει τα δεδομένα

MIDI δεδομένα

Το αντικείμενο **notein** αναφέρει midi δεδομένα (νότα, ένταση, κανάλι) από τις συνδεδεμένες midi συσκευές.

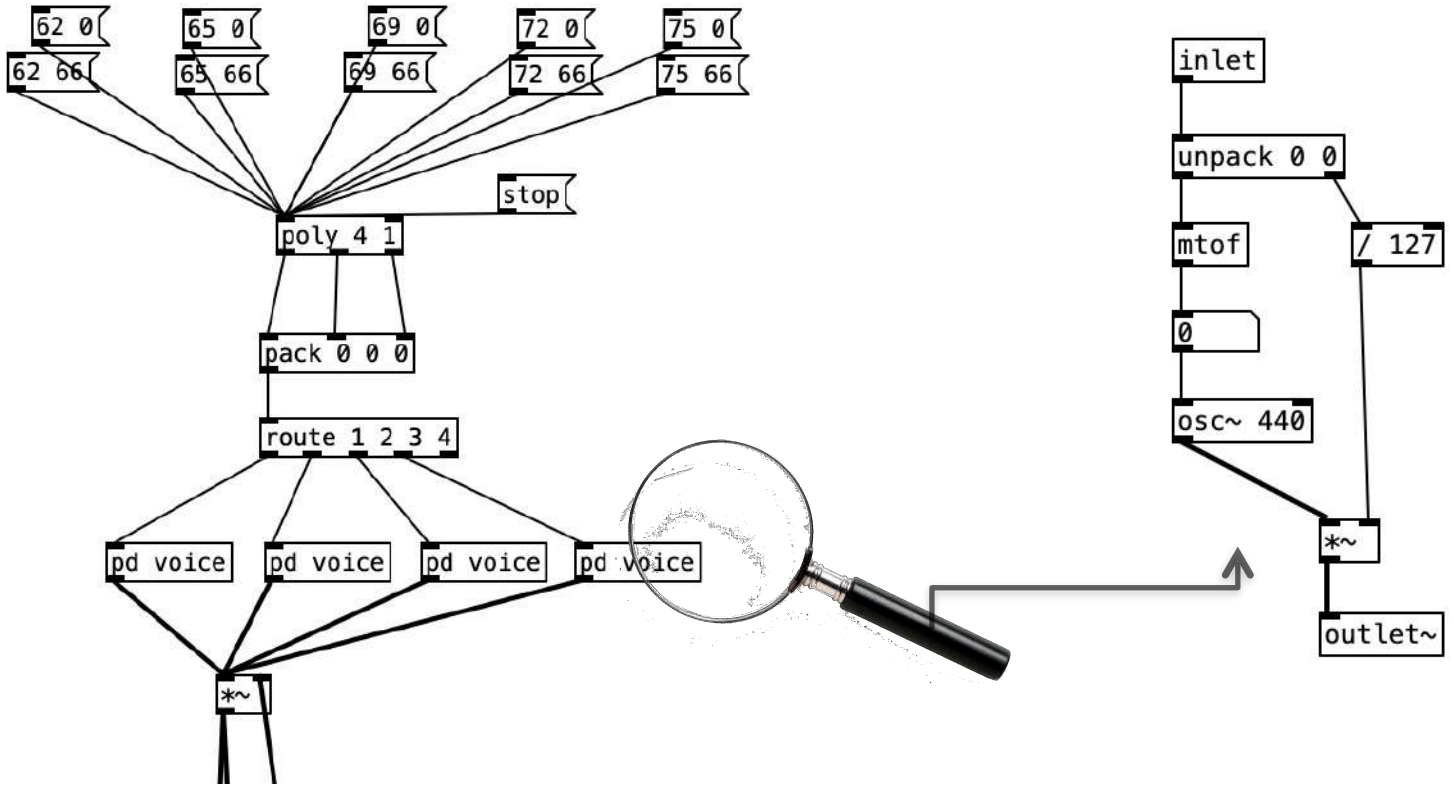


MIDI δεδομένα

Για διαμόρφωση και δοκιμή των midi ρυθμίσεων, δηλαδή για τη σύνδεση hardware ή software midi συσκευών στην είσοδο και έξοδο του Pure Data, αναφερόμαστε στις menu επιλογές **MIDI Settings** και **Test Audio and MIDI** αντίστοιχα.

Πολυφωνία

Το αντικείμενο **poly** δέχεται ζευγάρια τιμών (pitch-velocity), τα οποία αποσυναρμολογεί προσθέτοντας έναν αριθμητικό δείκτη. Συνδυαζόμενο με **pack**, για επανασυναρμολόγηση σε λίστα με 3 στοιχεία, και **route**, για οδήγηση της λίστας σε αντίστοιχη έξοδο ανάλογα με το πρώτο της στοιχείο, μπορούμε να οδηγήσουμε διαφορετικά midi συμβάντα σε ξεχωριστές φωνές.



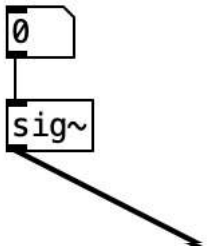
Σήματα Ήχου

Σε αντίθεση με τα σήματα ελέγχου, τα οποία ενεργούν μόνο όταν τα καλεί ο χρήστης, στα σήματα ήχου (audio) η ροή της πληροφορίας είναι συνεχής.

Όλα τα αντικείμενα που επεξεργάζονται σήματα ήχου φέρουν το σύμβολο “**~**” και τα καλώδια τους είναι πιο χοντρά από αυτά των αντικειμένων ελέγχου.

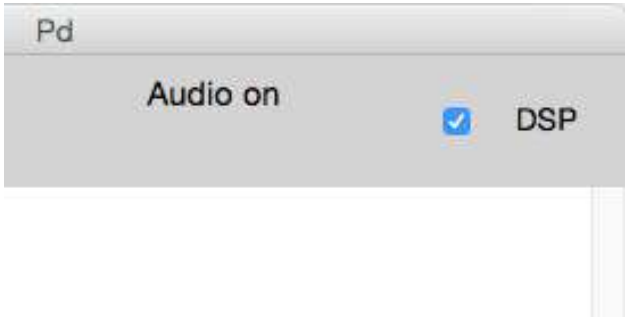
Πολλά αντικείμενα συναντώνται και στις δύο μορφές, ελέγχου και ήχου.

Στο παράδειγμα, ένα **number box** στέλνει σήμα ελέγχου στο αντικείμενο **sig~**, το οποίο διαβιβάζει την πληροφορία ως σήμα ήχου.



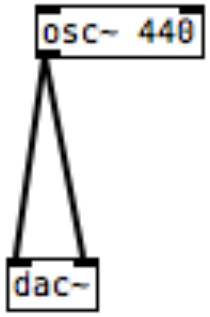
Σήματα Ήχου

Για να ακούμε τα σήματα ήχου χρειάζεται να είναι ενεργοποιημένο το το **dsp** του προγράμματος (**d**igital **s**ignal **p**rocessing).



Επίσης, στη κατάληξη του προγράμματος μας το σήμα πρέπει να οδηγείται στο αντικείμενο **dac~** (digital-analog converter).

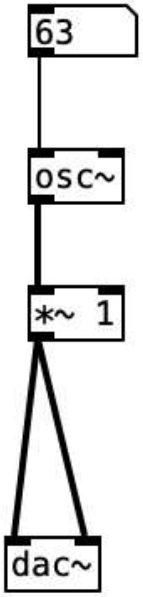
Η αριστερή και δεξιά είσοδος του αντικειμένου αντιπροσωπεύουν το αριστερό και δεξί κανάλι εξόδου αντίστοιχα.



Ταλαντώσεις: Ημιτονοειδής Ταλάντωση

Το αντικείμενο **osc~** παράγει απλή ημιτονοειδή ταλάντωση, δηλαδή ταλάντωση χωρίς αρμονικές, στη συχνότητα που αναφέρει το αρχικό του όρισμα ή που εισάγουμε στη ζεστή του είσοδο.

Για να ελέγξουμε την ένταση του σήματος, τροποποιούμε το πλάτος του μέσω αντικειμένου πολλαπλασιασμού σήματος ***~** .

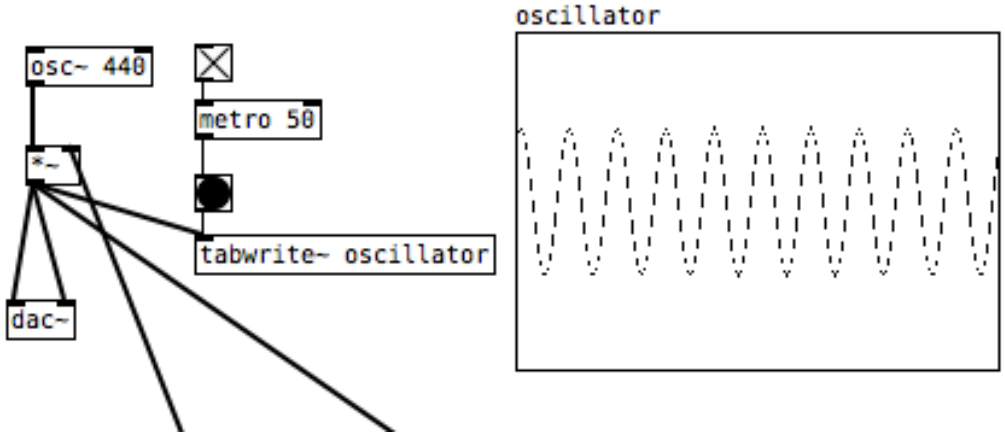


Παρατήρηση Κυματομορφής

Μπορούμε να παρατηρήσουμε οπτικά την κυματομορφή του σήματος μέσω του αντικειμένου **array**.

Το αντικείμενο **array** είναι ουσιαστικά ένας πίνακας αποθήκευσης τιμών και έχει πολλές διαφορετικές χρήσεις.

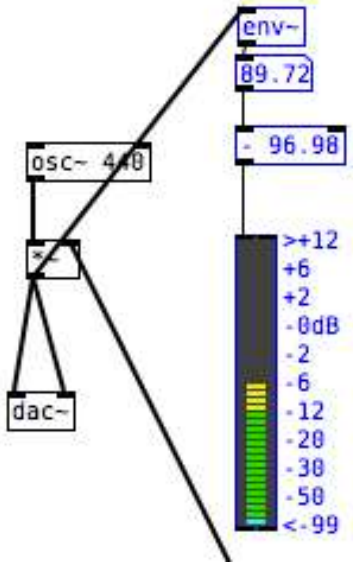
Σε μια από αυτές συνδυάζεται με το αντικείμενο **tabwrite~** και σε δεδομένη συχνότητα που ορίζουμε μέσω **metro** αναφέρει το περιεχόμενό του.



Παρατήρηση Στάθμης

Παρατήρηση σχετική με τη στάθμη του σήματος επιτυγχάνεται μέσω του αντικείμενου **env~**, το οποίο υπολογίζει και κανονικοποιεί την RMS τιμή (Root Mean Square) ενός σήματος σε dB, σε συνδυασμό με το αντικείμενο **vu** για απεικόνιση των τιμών.

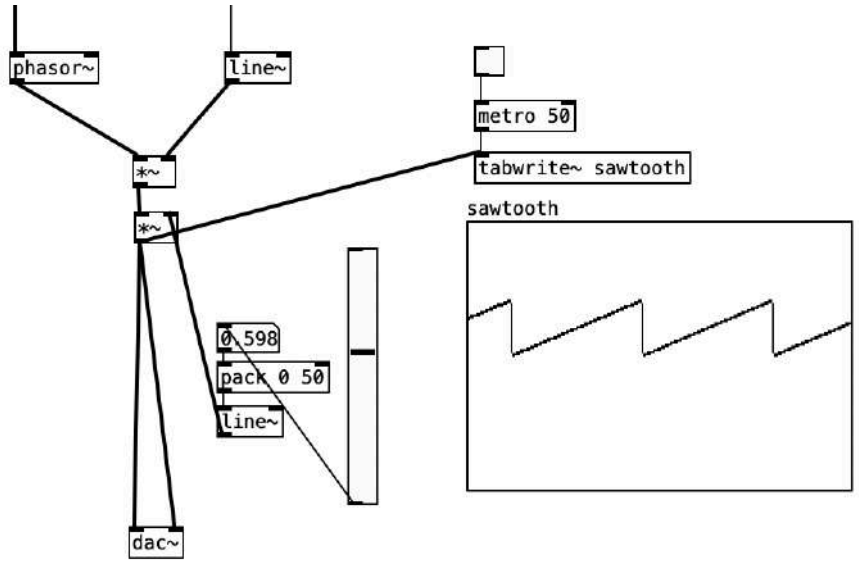
Παρεμβάλλουμε την πράξη αφαίρεσης, ώστε η μέγιστη τιμή πλάτους του σήματος μας (εδώ: 96,98) να αντιστοιχιστεί με την τιμή 0 του **vu**.



Ταλαντώσεις: Πριονωτή Ταλάντωση

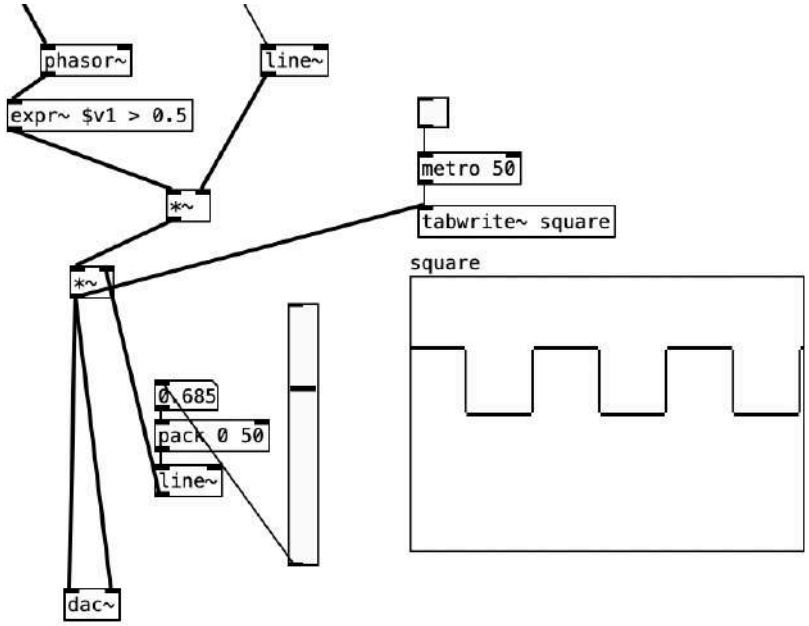
Το αντικείμενο **phasor~** παράγει μια γραμμική μετάβαση από τιμή σήματος 0 σε τιμή σήματος 1 και έπειτα απότομη μετάβαση στο 0 για επανάληψη της κίνησης.

Χρησιμοποιούμε το **phasor~** ως γεννήτρια πριονωτής ταλάντωσης, δηλαδή ταλάντωσης με όλες τις αρμονικές παρούσες με ενέργεια αντιστρόφως ανάλογη προς τη θέση τους. Δηλαδή, η αρμονική 2 θα έχει “ένταση” 1/2, η αρμονική 3 “ένταση” 1/3, κ.ο.κ.



Ταλαντώσεις: Τετράγωνη Ταλάντωση

Με μια μικρή προσαρμογή μετατρέπουμε την πριονωτή ταλάντωση σε τετράγωνη. Συγκεκριμένα, με το αντικείμενο **expr~** ορίζουμε κάθε τιμή σήματος μεγαλύτερη της τιμής 1.5 να κβαντίζεται στην τιμή 1, και οι υπόλοιπες τιμές σήματος στο 0.



Έλεγχος Έντασης

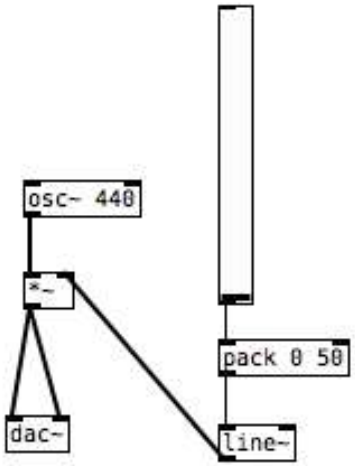
Για χειροκίνητο έλεγχο της έντασης μπορούμε να χρησιμοποιήσουμε έναν **ολισθητή** με τιμές από 0 έως 1. Όμως, προς αποφυγή παραμορφώσεων χρειάζεται να μεταβαίνουμε ομαλά από τη μια τιμή στην επόμενη.

Αυτό επιτυγχάνεται μέσω του αντικειμένου **line~**, το οποίο δέχεται λίστες τιμών στη μορφή:

"τιμή προορισμού - χρόνος μετάβασης"

Στο παράδειγμα, με το αντικείμενο **pack** δημιουργούμε τέτοιες λίστες με χρόνο μετάβασης 50 mlsecs.

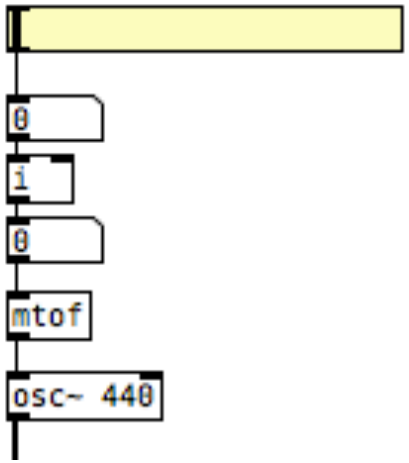
Προσέξτε πως, επειδή η ροή των σημάτων ήχου είναι συνεχής, η σύνδεση του **line~** με το ***~** είναι εφικτή μόνο, αν το αντικείμενο πολλαπλασιασμού δεν έχει αρχικό όρισμα!



Έλεγχος Συχνότητας

Το αντικείμενο **mtof** μετατρέπει αριθμητικές τιμές midi σε συχνότητα. Μέ έναν οριζόντιο ολισθητή (**hslider**) μπορούμε να στέλνουμε τέτοιες τιμές σε συγκεκριμένο εύρος που ορίζουμε από τις ιδιότητες του αντικειμένου.

Για παράδειγμα, οι τιμές 60 έως 72 αντιστοιχούν στις νότες από C4 έως C5. Αν θέλουμε να αποφύγουμε glissandi και να έχουμε διακριτά βήματα στον υπολογισμό των συχνοτήτων, παρεμβάλουμε το αντικείμενο **i** (integer) που κβαντίζει έναν δεκαδικό στον κοντινότερο ακέραιο προς το 0.



Περιβάλλουσες

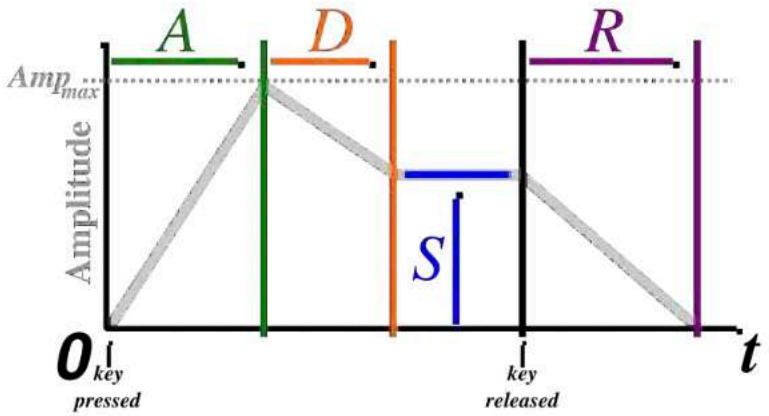
Οι **περιβάλλουσες** αποτυπώνουν συμπεριφορές ηχητικών παραμέτρων στον χρόνο.

Πολύ συνηθισμένη είναι η χρήση τους για δυναμικό έλεγχο της έντασης ενός ήχου. Η διαδικασία ονομάζεται μορφοποίηση έντασης.

Ένα τυπικό πρότυπο περιβάλλουσας είναι το **ADSR**:

(να σημειωθεί πως Attack, Decay, Release είναι χρόνος, ενώ Sustain στάθμη)

- Attack
  - Πόσο γρήγορα ο ήχος φθάνει τη μέγιστή του τιμή
- Decay
  - Πόσο γρήγορα ο ήχος πέφτει από την μέγιστη τιμή στην τιμή Sustain
- Sustain
  - Ο ήχος διατηρείται σταθερός
- Release
  - Ο χρόνος που απαιτείται μέχρις ότου ο ήχος σβήσει (σε σχέση με την παύση διέγερσής του)



Περιβάλλουσες

Στο Pure Data υπάρχουν διάφοροι τρόποι να δημιουργήσουμε περιβάλλουσες.

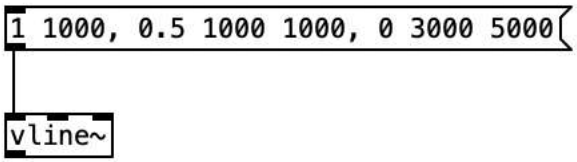
Όπως είδαμε, το αντικείμενο **line~** (και **line**) δέχεται ζεύγη τιμών.

Πιο σύνθετες μεταβάσεις δημιουργούμε με το αντικείμενο **vline~**.

Αυτό δέχεται πρώτα ένα αρχικό ζεύγος τιμών για την πρώτη μετάβαση, και έπειτα, χωρισμένα με κόμμα, πλήθος άλλων μεταβάσεων στη μορφή:

"νέος προορισμός - χρόνος μετάβασης - χρόνος έναρξης της μετάβασης από την αρχή της εντολής"

Στο παράδειγμα θα γίνουν 3 κινήσεις:

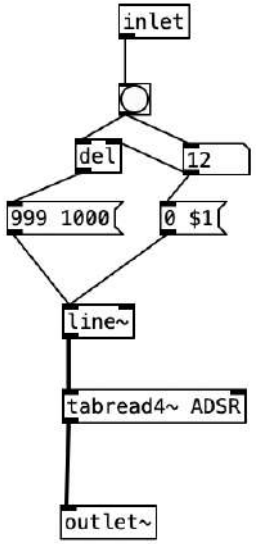


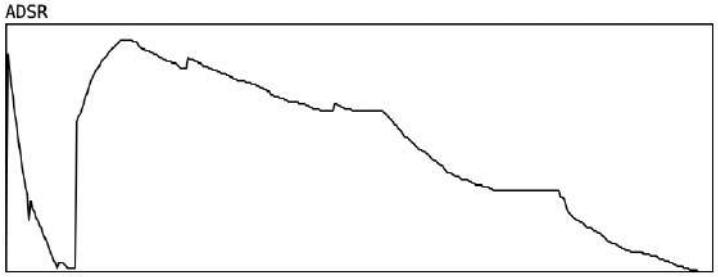
1. Μετάβαση στην τιμή 1 μέσα σε 1 sec
2. Μετά από 1 sec από την έναρξη - συνεπώς αμέσως μετά το τέλος της 1ης κίνησης - νέα μετάβαση στην τιμή 0.5 μέσα σε 1 sec
3. Μετά από 5 sec από την έναρξη - συνεπώς 3 sec μετά το τέλος της 2ης κίνησης - νέα μετάβαση στην τιμή 0 μέσα σε 3 sec

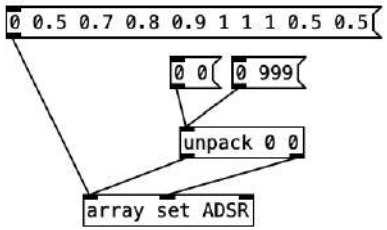
Να σημειωθεί πως οι μεταβάσεις των **line**, **line~**, και **vline~** είναι γραμμικές (linear ramps)

Περιβάλλουσες

Εναλλακτικά, μπορούμε να σχεδιάζουμε και να τροποποιούμε σε πραγματικό χρόνο περιβάλλουσες. Αυτή τη φορά το αντικείμενο **array** αξιοποιείται ως πίνακας εισαγωγής -και όχι προβολής- δεδομένων. Το αντικείμενο **tabread4~** διαβάζει τον πίνακα εξάγοντας για κάθε index (άξονας x) που δέχεται στην είσοδο την τιμή του άξονα y που της αντιστοιχεί. Για να σαρωθεί ο πίνακας με σειρά χρησιμοποιούμε **line~** σύμφωνο με το πλήθος σημείων του πίνακα (εδώ:99). Για να ορίσουμε κάποιες τιμές με ακρίβεια χρησιμοποιούμε **array set**, το οποίο στη μεσαία είσοδο δέχεται τη θέση εγγραφής και στη ζεστή είσοδο την αντίστοιχη τιμή.

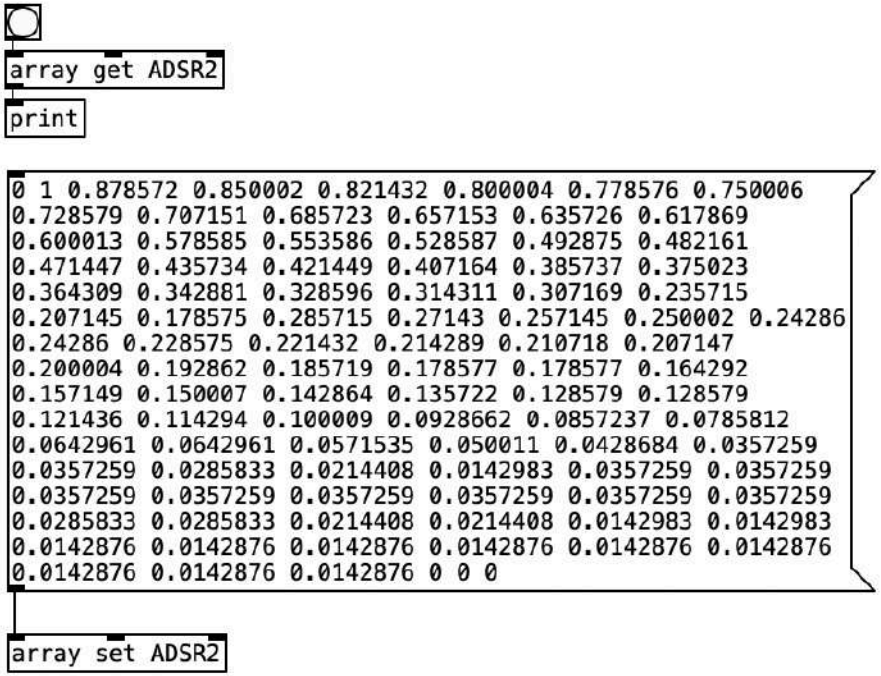






Περιβάλλουσες

Για να αποθηκεύσουμε μια περιβάλλουσα ως preset, εξάγουμε τα σημεία του **array** στην κονσόλα μέσω **print** και τα αντιγράφουμε σε **message**, το οποίο μέσω **set** ανακαλούμε στο **array** κατά βούληση.



Σύνθεση Κυματοπινάκων (Wavetable Synthesis)

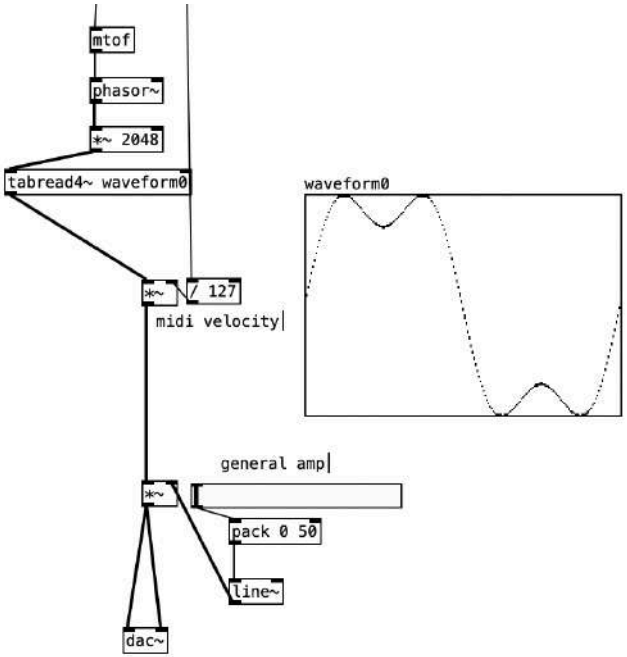
Η τεχνική της Σύνθεσης Κυματοπινάκων βασίζεται στην αρχή πως κάθε σύνθετο σήμα μπορεί να αναλυθεί σε αλληλεπιδρώντα ημίτονα/συνημίτονα (θεώρημα Fourier). Σε σχέση με τη γεννήτρια ταλαντώσεων **phasor~**, η Σύνθεση Κυματοπινάκων παρουσιάζει τα εξής πλεονεκτήματα:

- Ελέγχουμε το πόσες αρμονικές θα συμπεριλάβουμε, ώστε περιορίζουμε το φαινόμενο Aliasing
- Προσομοιώνουμε ακριβέστερα τη φυσική διαδικασία παραγωγής ήχου, αφού στη φύση δεν υπάρχουν ευθείες μεταβάσεις, αλλά συχνοτικές αλληλεπιδράσεις, ώστε προκύπτει πιο “ζεστό” άκουσμα

Σύνθεση Κυματοπινάκων (Wavetable Synthesis)

Στη Σύνθεση Κυματοπινάκων γράφουμε σε ένα **array** τιμές, οι οποίες θα διαβαστούν από αντικείμενο **tabread4~** σε συχνότητα που ορίζει αντικείμενο **phasor~**.

Να σημειωθεί πως εδώ δεν χρησιμοποιούμε το **phasor~** ως γεννήτρια, αλλά ως εργαλείο σάρωσης του πίνακα, αφού το **array~** παράγει περιοδικές μεταβάσεις από το 0 έως το 1, τις οποίες πολλαπλασιάζουμε με το πλήθος σημείων του πίνακα (εδώ: 2048)



Σύνθεση Κυματοπινάκων (Wavetable Synthesis)

Τον πίνακα μας τον γεμίζουμε με τις επιθυμητές αρμονικές με τον εξής τρόπο:

Αρχικά, το σύμβολο “**;**” σε message στέλνει στον πίνακα που θα ονοματίσουμε την εντολή **sinesum**, ότι δηλαδή θα ακολουθήσει ένα σύνολο αρμονικών. Έπειτα δηλώνουμε το πλήθος των σημείων του πίνακα (εδώ:2048) και τέλος τις αρμονικές, αρχής γενομένης από τη θεμελιώδη συχνότητα με ένταση 1 και τις υπόλοιπες αρμονικές με τις σχετικές τους εντάσεις.

Παρατηρήστε στα παρακάτω παραδείγματα πως η πριονωτή κυματομορφή ακολουθεί τον αλγόριθμο “*όλες οι αρμονικές με παρουσία αντιστρόφως ανάλογη προς τη θέση τους (1/h)*” και η τετράγωνη κυματομορφή τον αλγόριθμο “*μόνο οι μονές αρμονικές με παρουσία αντιστρόφως ανάλογη προς τη θέση τους στο τετράγωνο* $(1/h^2)$”.

Τέλος, η εντολή **normalize** κανονικοποιεί το άθροισμα της ενέργειας των αρμονικών στην τιμή που θα ορίσουμε (εδώ: 1).

sinewave – only 1 harmonic (fundamental)

```
;
waveform0 sinesum 2048 1
```

sawtooth wave with 4 harmonics

```
;
waveform0 sinesum 2048 1 0.5 0.33 0.25;
waveform0 normalize 1
```

square wave with 4 harmonics

```
;
waveform0 sinesum 2048 1 0 0.33 0;
waveform0 normalize 1
```

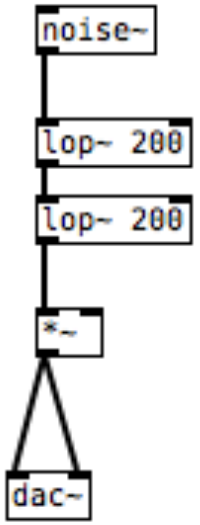
Συχνοτικά Φίλτρα

Το αντικείμενο **noise~** είναι μια γεννήτρια λευκού θορύβου, δηλαδή θορύβου, όπου η συχνοτική ενέργεια καταλαμβάνει ισομερώς το φάσμα.

Συχνά συνδυάζεται με τα αντικείμενα **lop~**, **hip~** και **bp~** , τα οποία αποτελούν συχνοτικά φίλτρα: χαμηλοπερατό, υψηλοπερατό και εύρους ζώνης αντίστοιχα.

Με τη σειραϊκή τοποθέτηση δύο ή περισσότερων ίδιων αντικειμένων αυξάνεται η τάξη του φίλτρου.

Το παρακάτω παράδειγμα παρουσιάζει ένα χαμηλοπερατό φίλτρο 2ης τάξεως, με όριο τα 200Hz

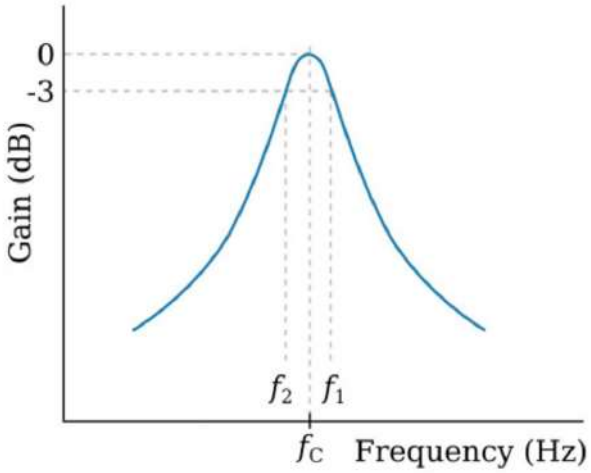


Συχνοτικά Φίλτρα

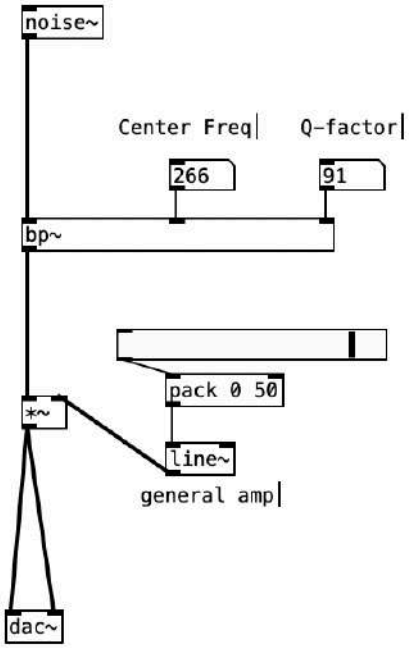
Συγκεκριμένα το φίλτρο εύρους ζώνης **bp~** δέχεται ως ορίσματα:

• μία κεντρική συχνότητα (Center Frequency), γύρω από την οποία συντελείται η συχνοτική αποσιώπηση, και

• έναν συντελεστή Q (Quality Factor) που δείχνει πόσο “τέλειο” είναι το φίλτρο, τουτέστιν όσο μεγαλύτερος ο συντελεστής Q, τόσο πιο οξεία η ενίσχυση της κεντρικής συχνότητας σε συνδυασμό με αποσιώπηση των γειτονικών.

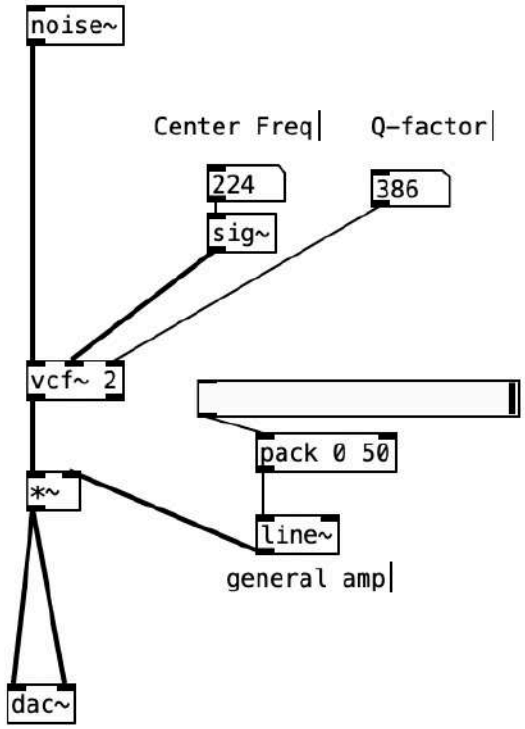


$$Q = \frac{f_C}{f_1 - f_2} = \frac{\text{center frequency}}{3\text{ dB bandwidth}}$$



Συχνοτικά Φίλτρα

Το αντικείμενο **vcf~** είναι ταυτόχρονα φίλτρο εύρους ζώνης και χαμηλοπερατό, από αριστερή και δεξιά έξοδο αντίστοιχα, του οποίου η κεντρική συχνότητα οδηγείται από σήμα, παρέχοντας έτσι ιδιαίτερες δυνατότητες για επεξεργασία σε πραγματικό χρόνο.

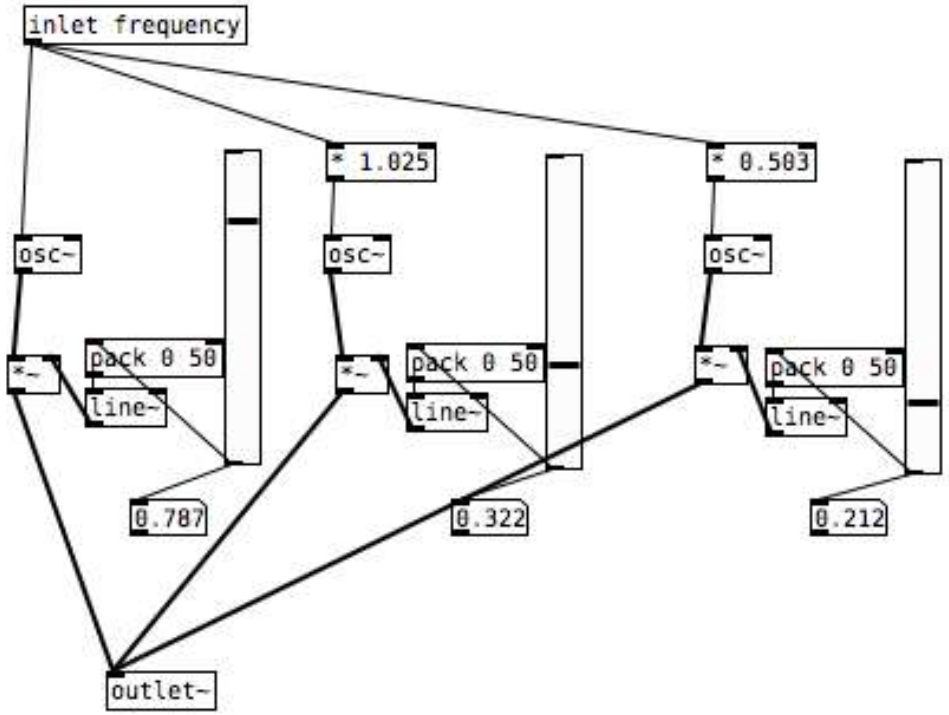


Αθροιστική Σύνθεση

Στο Pure Data, Αθροιστική Σύνθεση επιτυγχάνεται, όταν τα σήματα περισσότερων ταλαντωτών κατευθύνονται στο ίδιο σημείο, πχ στην ίδια είσοδο ενός πολλαπλασιαστή σήματος (***~**) ή στην ίδια έξοδο σήματος (**outlet~**) του προγράμματός μας.

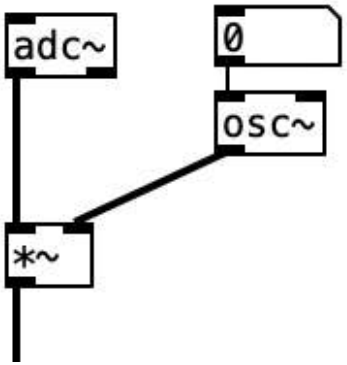
Με πολλαπλασιαστές σήματος (***~**) μπορούμε να ελέγχουμε ανεξάρτητα και την ένταση κάθε ταλαντωτή, ενώ με πολλαπλασιαστές ελέγχου (*****) μπορούμε να ορίζουμε τη σχέση της συχνότητας του αρχικού ταλαντωτή ως προς αυτές των υπόλοιπων.

Στο παράδειγμα, ο δεύτερος ταλαντωτής είναι «ξεκούρδιστος» ως προς τον πρώτο (*** 1.025**), και ο τρίτος περίπου μια οκτάβα χαμηλότερος (***0.503**).



Διαμόρφωση Δακτυλίου

Ως απλή μορφή της Διαμόρφωσης Πλάτους, η Διαμόρφωση Δακτυλίου επιτυγχάνεται μέσω του πολλαπλασιασμού του σήματος ενός Φορέα (**carrier**) με το σήμα ενός Διαμορφωτή (**modulator**).



Διαμόρφωση Πλάτους

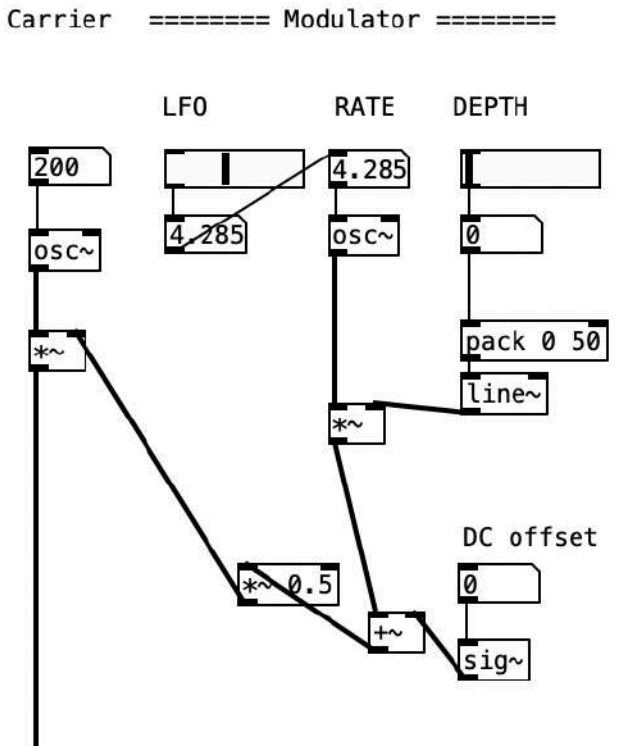
Σε πιο σύνθετη εφαρμογή Διαμόρφωσης Πλάτους, μπορούμε να ορίσουμε τις εξής παραμέτρους:

- **rate** : η συχνότητα του Διαμορφωτή ορίζει το ρυθμό μεταβολής της έντασης του Φορέα
- **depth** (αξίες από 0 έως 1) : το πλάτος του Διαμορφωτή ορίζει το πλάτος της Διαμόρφωσης, ουσιαστικά τον βαθμό επίδρασης του εφέ
- **DC offset** : η πρόσθεση σταθερής τάσης στο σήμα του Διαμορφωτή ορίζει τον βαθμό παρουσίας της αρχικής συχνότητας του Φορέα στο τελικό αποτέλεσμα

Όταν η τιμή της παραμέτρου rate είναι έως 10 Hz

(Low Frequency Oscillator - **LFO**)

τότε έχουμε το φαινόμενο του **tremolo**.

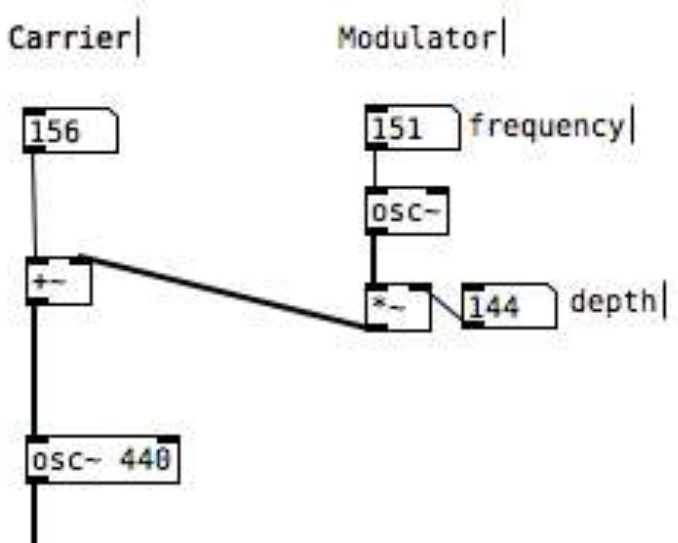


Διαμόρφωση Συχνότητας

Η Διαμόρφωση Συχνότητας επιτυγχάνεται μέσω της πρόσθεσης του σήματος ενός Φορέα (**carrier**) με το σήμα ενός Διαμορφωτή (**modulator**). Κατά τη διαδικασία,

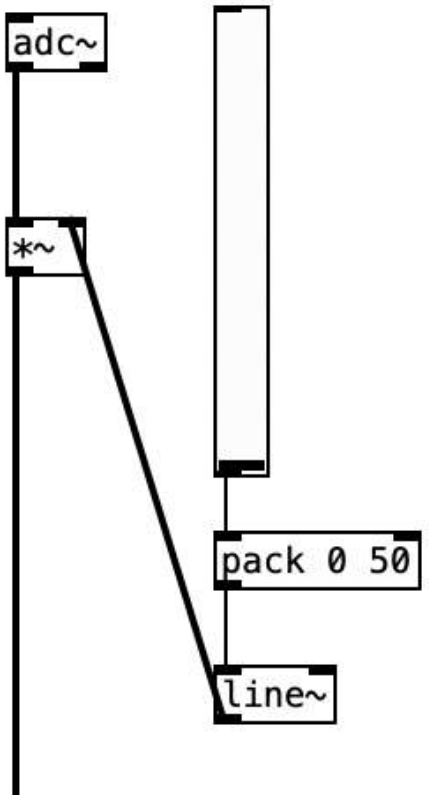
- **frequency** : η συχνότητα του Διαμορφωτή ορίζει το ρυθμό μεταβολής της συχνότητας του Φορέα
- **depth** : το πλάτος του Διαμορφωτή το εύρος της επίδρασης, +- γύρω από τη συχνότητα του Φορέα

Με μικρές τιμές (**LFO έως 10 Hz**) του ρυθμού Διαμόρφωσης έχουμε το φαινόμενο του **vibrato**.



Ηχογράφηση/Εισαγωγή Δείγματος

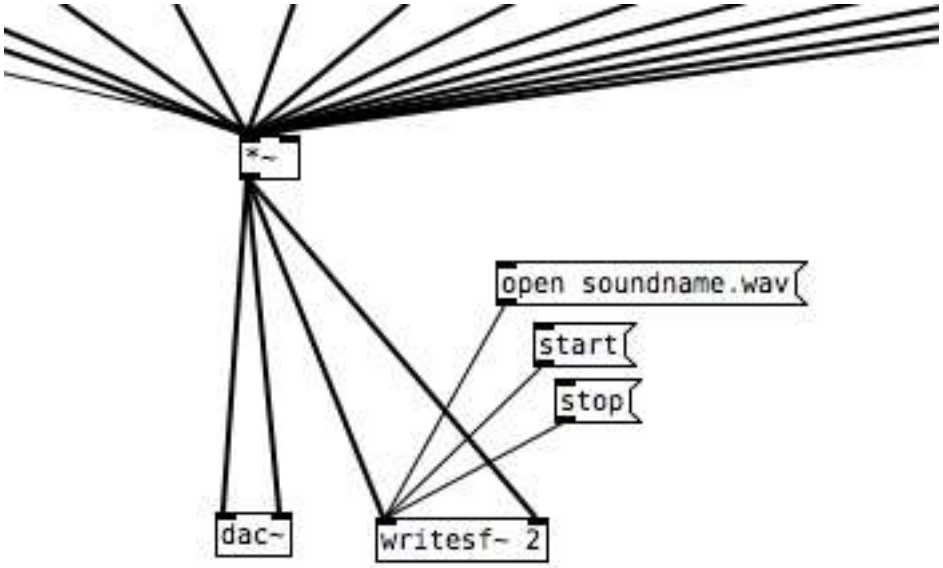
Εισάγουμε ήχο στο πρόγραμμα προς ηχογράφηση με το αντικείμενο **adc~** (Analog to Digital Converter). Ουσιαστικά λειτουργεί ως μικρόφωνο, αντλώντας πληροφορία από την πηγή που έχουμε ορίσει στις Ρυθμίσεις (menu: Media > Audio Settings)



Ηχογράφηση/Εισαγωγή Δείγματος

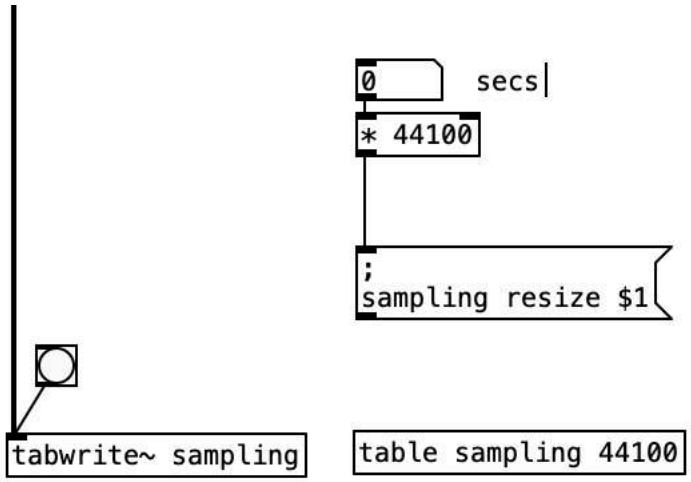
Η πρώτη επιλογή είναι να καταγράψουμε τον ήχο στον σκληρό δίσκο του υπολογιστή, λύση που προτιμάται συνήθως για μεγάλα αρχεία που δεν μετακινούμε.

Χρησιμοποιούμε το αντικείμενο **writesf~**. Αρχικά δημιουργούμε το νέο αρχείο με την εντολή **open** και όρισμα το επιθυμητό όνομα. Θα δημιουργηθεί αυτόματα στο directory του patch μας. Οι εντολές **start** και **stop** ενεργοποιούν και τερματίζουν αντίστοιχα τη ροή ηχητικών δεδομένων προς ηχογράφηση.



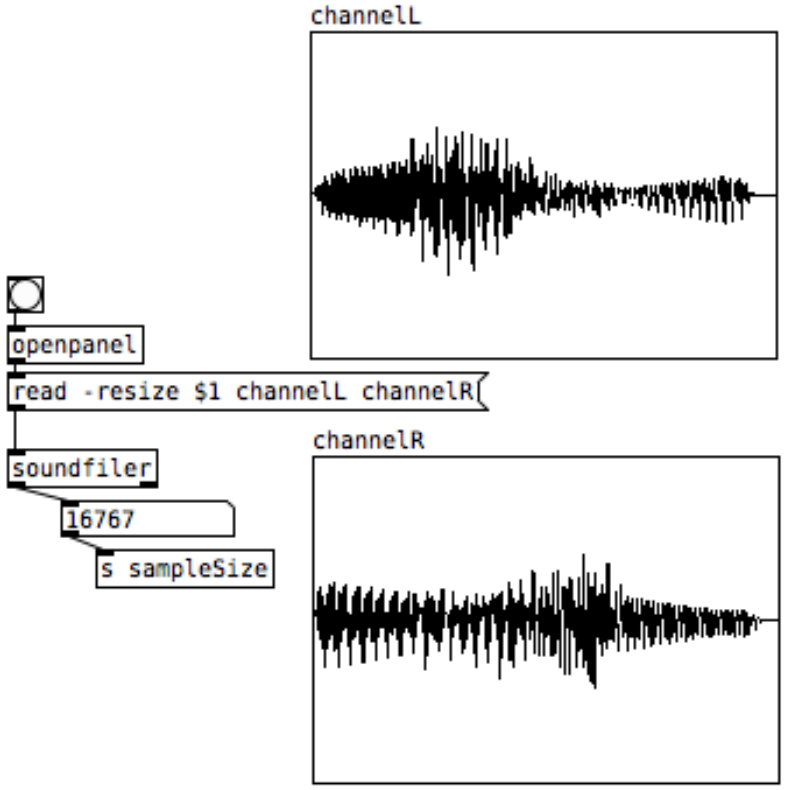
Ηχογράφηση/Εισαγωγή Δείγματος

Εναλλακτική, πιο ευέλικτη επιλογή είναι να ηχογραφήσουμε σε ένα “buffer”, δηλαδή σε έναν προσωρινό αποθηκευτικό χώρο. Θα δούμε πως υπάρχουν διάφοροι τρόποι να δημιουργήσουμε ένα buffer. Στο παρακάτω παράδειγμα αξιοποιούμε το αντικείμενο **table**, δίνοντας ένα όνομα (sampling) και ένα μέγεθος (44100sample = 1sec σε standard δειγματοληψία). Έπειτα, ξεκινούμε την ηχογράφηση με **bang** στο αντικείμενο **tabwrite~** παραπέμποντας στο όνομα του buffer. Η ηχογράφηση θα διαρκέσει μόνο για τη “διάρκεια” του buffer, την οποία μπορούμε να αλλάξουμε με την εντολή **resize**.



Ηχογράφηση/Εισαγωγή Δείγματος

Άλλος τρόπος για δημιουργία buffer είναι μέσω **array**, στο οποίο εισάγουμε τις τιμές πλάτους του στοχευόμενου αρχείου με το αντικείμενο **soundfiler**. Η εντολή μηνύματος **read** με το "flag" **–resize $1** προσαρμόζει το μέγεθος του buffer στο μέγεθος του αρχείου που εισάγουμε.



Αναπαραγωγή Δείγματος

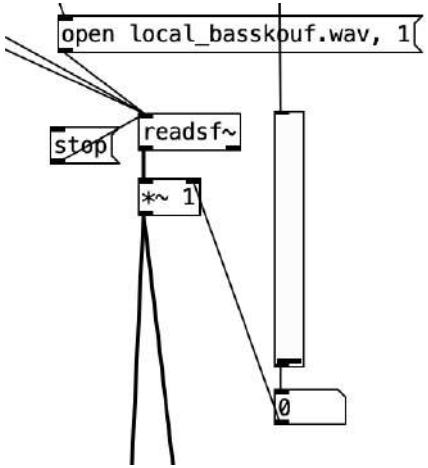
Για αναπαραγωγή αρχείου από τον σκληρό δίσκο χρησιμοποιούμε το αντικείμενο **readsf~**, στο οποίο πρέπει να σταλούν διαδοχικά ένα μήνυμα **open** με το πλήρες όνομα του αρχείου, αν βρίσκεται στο ίδιο directory, αλλιώς με το πλήρες path, και έπειτα ένα μήνυμα **start ή η τιμή 1**.

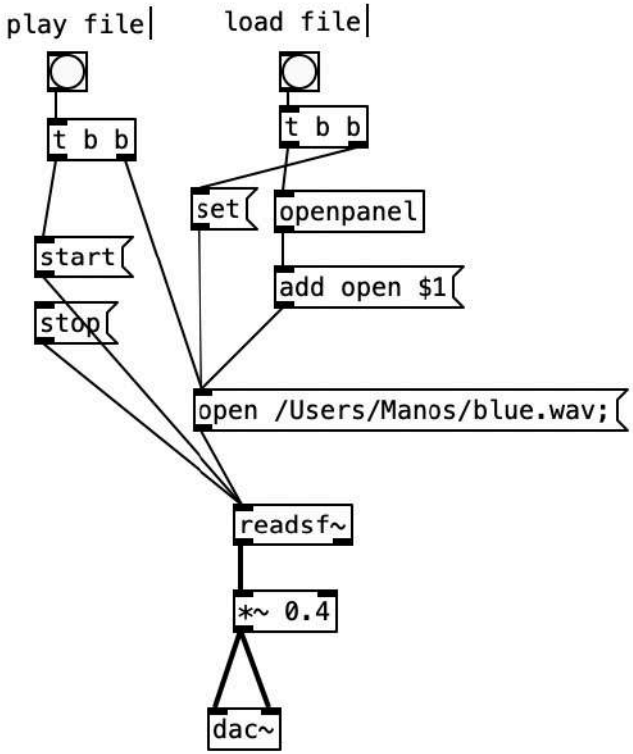
Το πλήρες path το βρίσκουμε με το αντικείμενο **openpanel**, το οποίο μας εμφανίζει ένα file explorer.

Προσέξτε πως η εντολή μηνύματος **add** συνδυάζει περισσότερα μηνύματα σε ένα.

Αν το ηχητικό αρχείο βρίσκεται στο ίδιο directory με το patch μας, χρειάζεται μόνο το όνομα χωρίς το path.

Μάλιστα, αξιοποιώντας το κόμμα που επιτρέπει την αποστολή πολλών μηνυμάτων σε σειρά, μπορούμε να αναπαράγουμε το αρχείο με ένα μόνο μήνυμα.

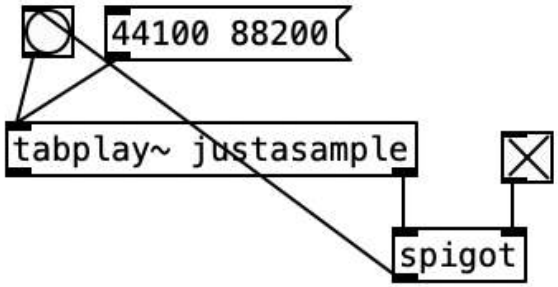




Αναπαραγωγή Δείγματος

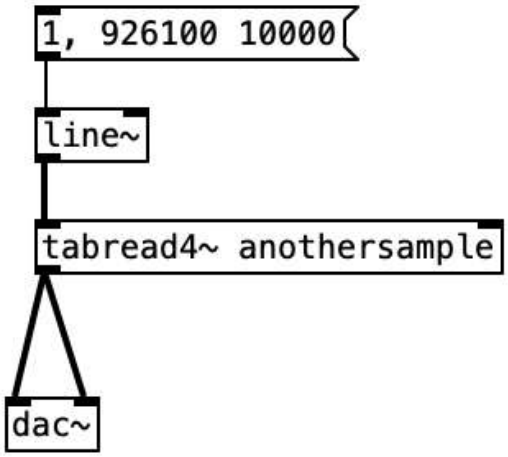
Για αναπαραγωγή αρχείου από buffer, μια απλή λύση είναι το αντικείμενο **tabplay~** με όρισμα το όνομα του στοχευόμενου buffer. Δεχόμενο **bang** αναπαράγει το περιεχόμενο του buffer από την αρχή, ενώ με λίστα δύο τιμών (samples) αναπαράγει το αντίστοιχο τμήμα.

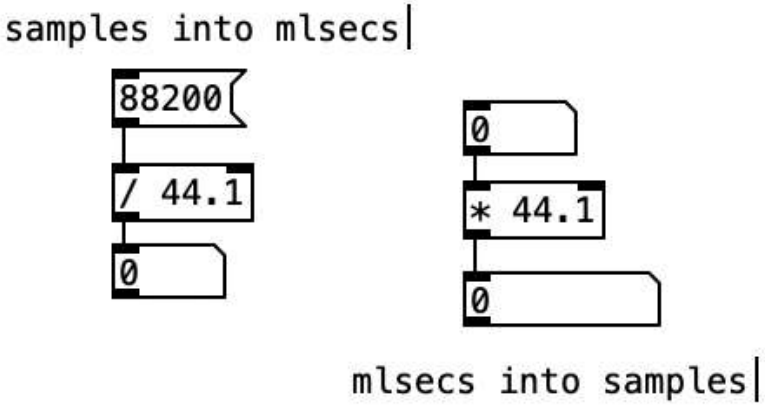
Τόσο το **tabplay~** όσο και το **readsf~** όταν ολοκληρώσουν την αναπαραγωγή στέλνουν **bang** από τη δεξιά έξοδο, κάτι που μπορούμε να αξιοποιήσουμε για τη δημιουργία βρόχου (loop).



Αναπαραγωγή Δείγματος

Άλλη επιλογή είναι το αντικείμενο **tabread4~**, το οποίο διαβάζει τα περιεχόμενα του buffer σύμφωνα με το μήνυμα μετάβασης που δέχεται μέσω **line~**. Τα πρώτα 2 στοιχεία του μηνύματος είναι σε samples (αφετηρία-προορισμός) και το 3ο στοιχείο ο χρόνος σε mlsecs για τη μετάβαση. Σημαντικό για να δημιουργούμε ενδιαφέρουσες μεταβάσεις είναι να εξοικειωθούμε με το μέγεθος σε samples και τη διάρκεια σε mlsecs ενός ηχητικού αρχείου. Παρακάτω βλέπουμε πώς μετατρέπουμε το ένα μέγεθος στο άλλο.



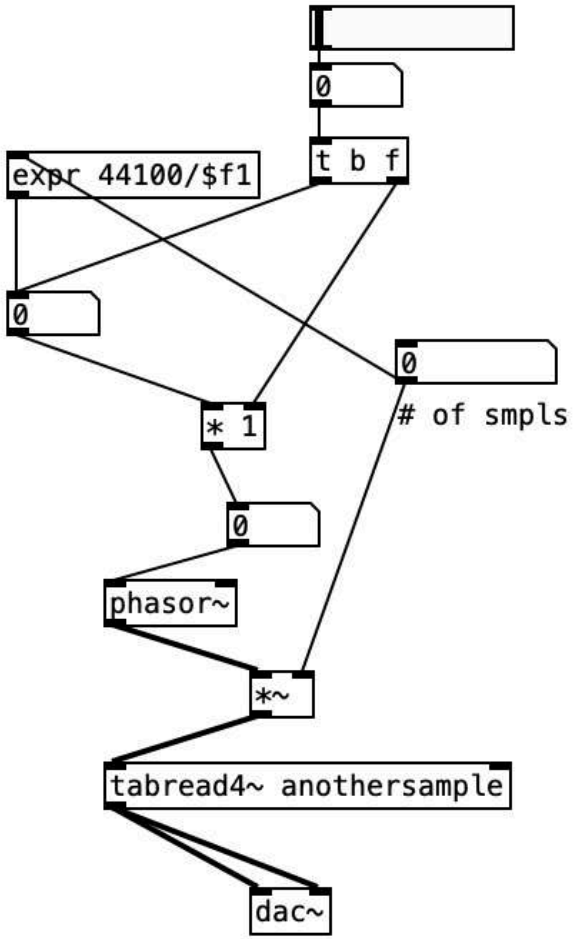


Αναπαραγωγή Δείγματος

Τέλος, μπορούμε να σαρώσουμε το buffer τροφοδοτώντας το **tabread4~** μέ **phasor~**, δηλαδή μέσω της γνωστής μας πια περιοδικής μετάβασης από το 0 έως το 1, πολλαπλασιαζόμενης με το σύνολο των δειγμάτων του buffer.

Για να βρούμε την απαραίτητη συχνότητα,
ώστε το **buffer** να διαβαστεί μία φορά
από την αρχή μέχρι το τέλος
διαιρούμε τη συχνότητα δειγματοληψίας (πχ εδώ **44100Hz**)
με το σύνολο των δειγμάτων.

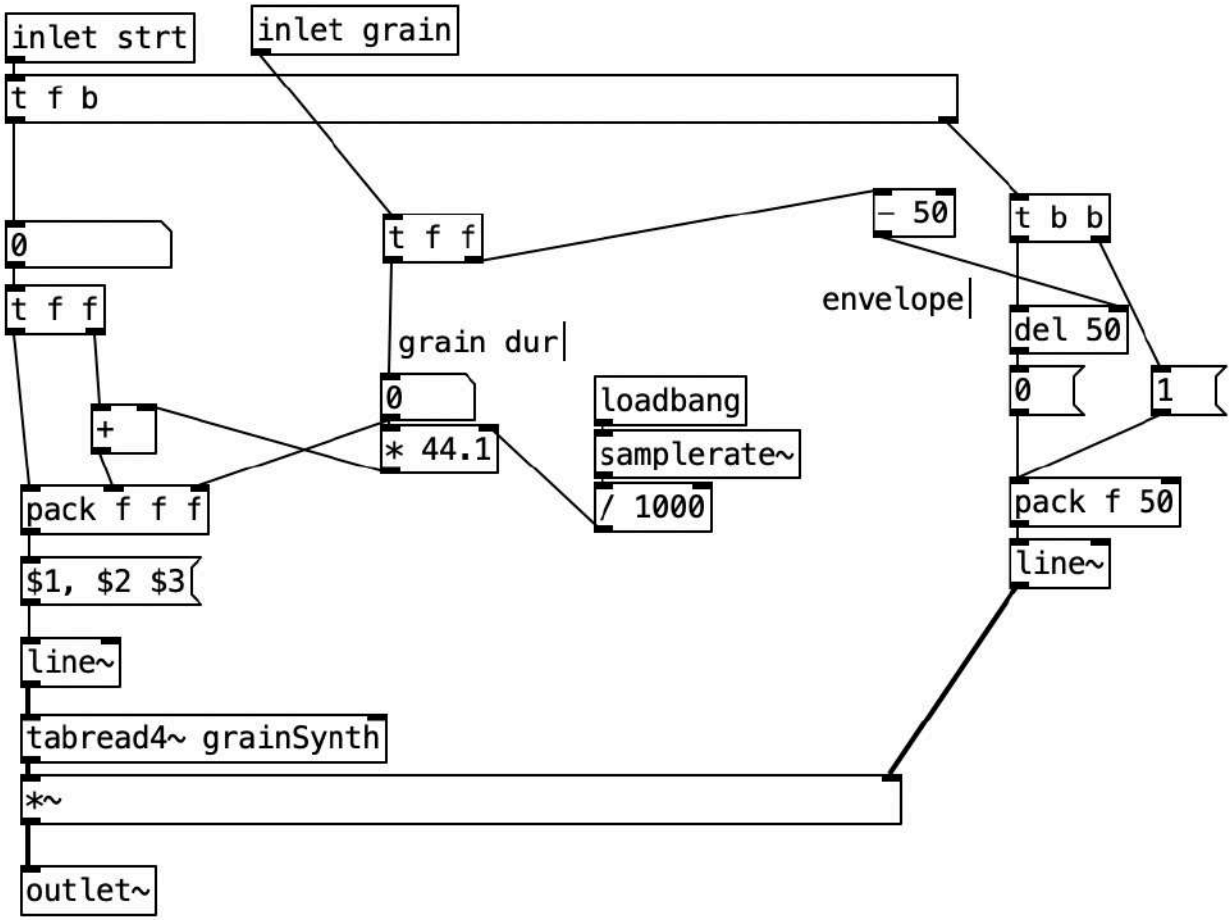
Πολλαπλασιάζοντας τη συχνότητα του **phasor~**
επηρεάζουμε την ταχύτητα αναπαραγωγής.
Αρνητικές τιμές διαβάζουν το buffer ανάποδα.



Μικροδομική Σύνθεση: ο Κόκκος

Στη Μικροδομική Σύνθεση οργανώνουμε ηχητικούς “κόκκους” σε σύνθετες δομές, όπως νέφη κτλ.

Το παρακάτω **abstraction** δείχνει τον προγραμματισμό ενός μεμονωμένου κόκκου.



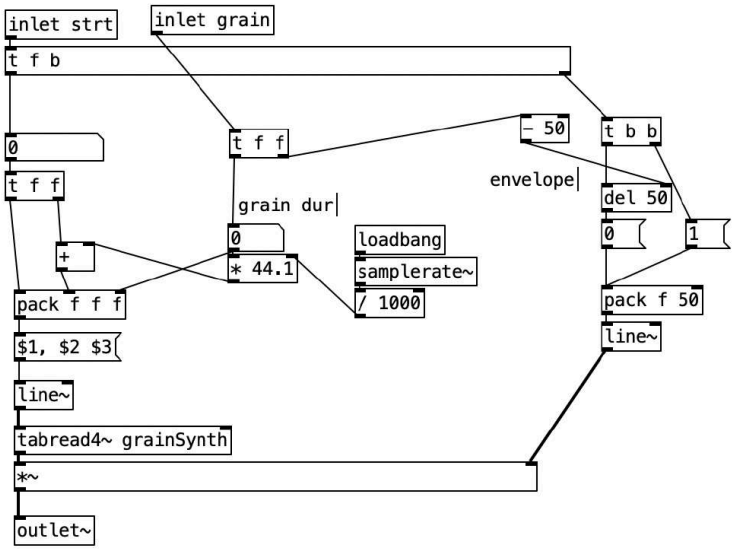
Μικροδομική Σύνθεση: ο Κόκκος

Το αντικείμενο **line~** δείχνει στο **tabread4~**, ποιό τμήμα του buffer να διαβάσει (σημείο έναρξης -samples-, σημείο λήξης -samples-, χρόνος μετάβασης -mlsecs-).

Στην αριστερή “στήλη” εισάγουμε το σημείο έναρξης του κόκκου ως πρώτο στοιχείο της λίστας μας (εξαιτίας του **trigger** το στοιχείο αυτό θα εισαχθεί τελευταίο στο **pack**, πυροδοτώντας τον κόκκο).

Στη μεσαία “στήλη” εισάγουμε το μέγεθος του κόκκου σε mlsecs ως τρίτο στοιχείο της λίστας, τον χρόνο δηλαδή της μετάβασης, και αφού το μετατρέψουμε σε samples το προσθέτουμε στο σημείο έναρξης, ώστε να καταχωρήσουμε το δεύτερο στοιχείο της λίστας, το σημείο λήξης του κόκκου.

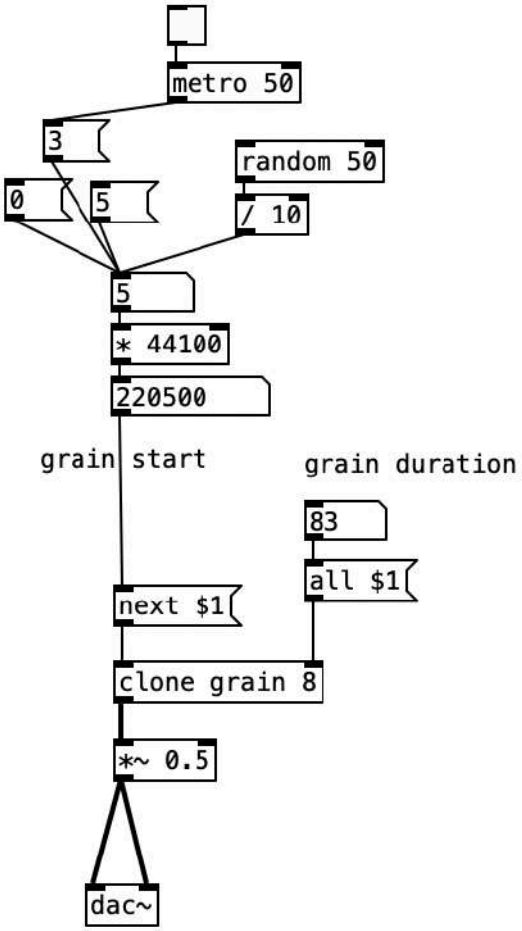
Η δεξιά “στήλη” εφαρμόζει έναν απλό μορφοποιητή έντασης.



Μικροδομική Σύνθεση: το Νέφος

Έχοντας αποθηκεύσει τον κόκκο μας ως **abstraction**, μπορούμε να τον πολλαπλασιάσουμε σε διαφορετικές “φωνές” με το αντικείμενο **clone** και ορίσματα το στοχευόμενο **abstraction** και των αριθμό των φωνών. Με το μήνυμα **next $1** στέλνουμε εκ περιτροπής στις φωνές σημεία έναρξης των κόκκων, ενώ με το μήνυμα **all $1** στέλνουμε σε όλες τις φωνές τη διάρκεια των κόκκων.

Με αλγοριθμική διαχείριση μπορούμε έπειτα να δημιουργήσουμε νέφη, αναπαράγωντας πχ μόνο έναν κόκκο (freeze fx),
ή πυροδοτώντας τους με τυχαία ανακατανομή,
ή συμπυκνώνοντάς τους σε επικαλύψεις και συνηχήσεις.



Μικροδομική Σύνθεση: Time Stretching

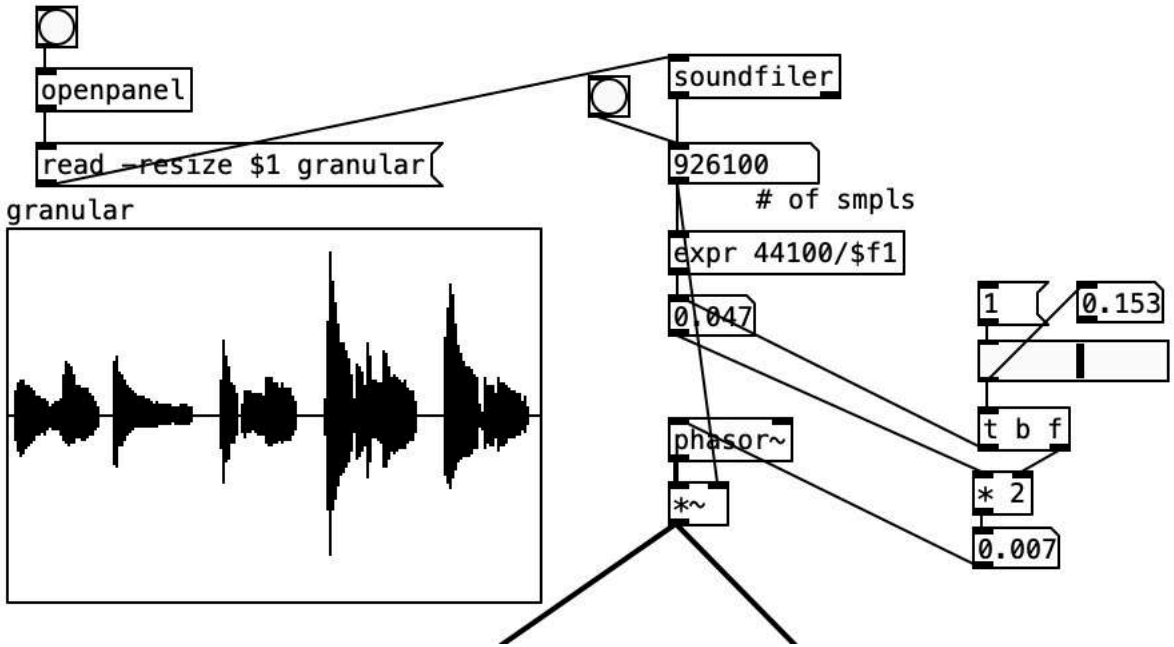
Time Stretching είναι η διαδικασία τροποποίησης της διάρκειας ενός ήχου χωρίς να επηρεαστεί το ύψος του συχνοτικού του περιεχομένου.

Θα δείξουμε την υλοποίηση Time Stretching εργαλείου μέσω Μικροδομικής Σύνθεσης σε 3 στάδια.

Η γενική ιδέα είναι πως θα πυροδοτούμε κόκκους με μεταβαλλόμενη ταχύτητα, κάθε κόκκος όμως θα αρχίζει από το χρονικό σημείο, στο οποίο βρίσκεται εκείνη τη στιγμή η κανονική αναπαραγωγή του **buffer**.

Μικροδομική Σύνθεση: Time Stretching

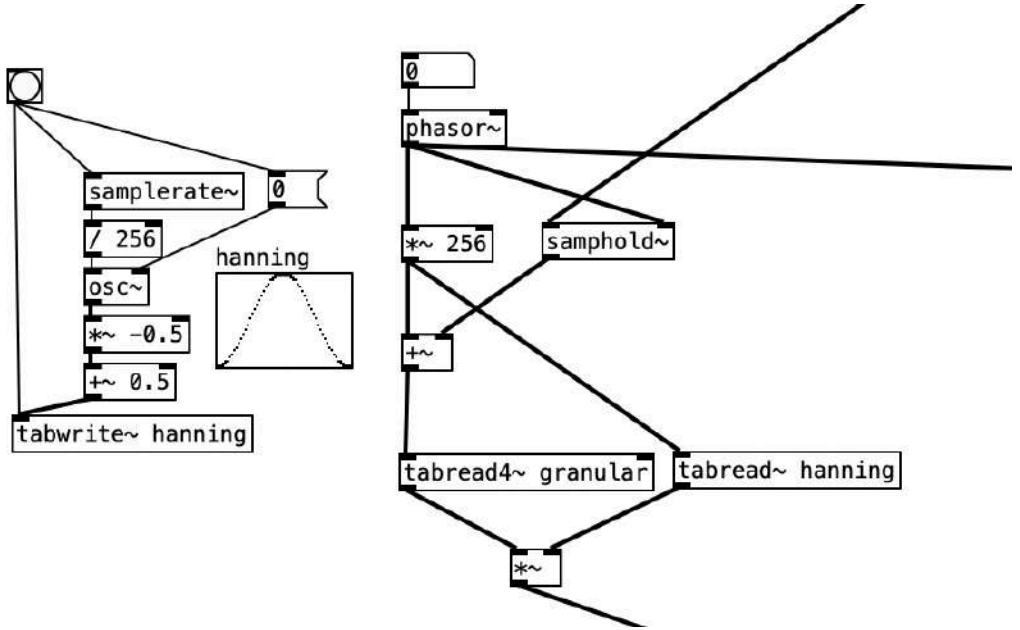
Συνεπώς, το **1ο στάδιο** είναι να έχουμε μια συνεχή ένδειξη ως προς το σημείο που βρίσκεται κάθε στιγμή η κανονική αναπαραγωγή του **buffer**. Όπως έχουμε δείξει, αυτό επιτυγχάνεται με σάρωση του **buffer** μέσω **phasor~**. Η πράξη του **expr** υπολογίζει τη συχνότητα του **phasor~** για κανονική αναπαραγωγή του **buffer** μία φορά σε μία περίοδο. Παρεμβάλλοντας πολλαπλασιαστή * μπορούμε να πειραματιστούμε και με διαφορετικές ταχύτητες αναπαραγωγής.



Μικροδομική Σύνθεση: Time Stretching

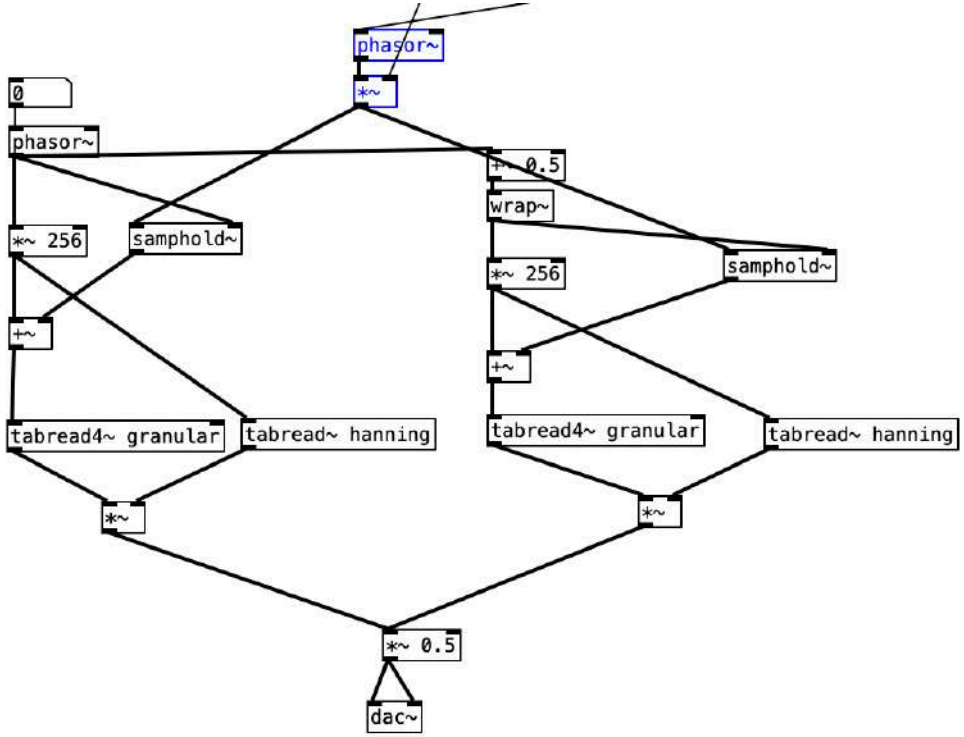
2ο στάδιο: Το **phasor~** της "ένδειξης" εισάγεται σε αντικείμενο **samphold~**, το οποίο εκπέμπει την αποθηκευμένη στη ζεστή είσοδο πληροφορία μόνο όταν η πληροφορία στην κρύα είσοδο πραγματοποιήσει πτώση, κάτι που με το **phasor~** συμβαίνει μόνο μια φορά στην αρχή κάθε περιόδου. Στην κρύα είσοδο του **samphold~** εισάγουμε νέο **phasor~**, αυτή τη φορά για τον κόκκο, εδώ μεγέθους 256 δειγμάτων. Μετά από κάθε περίοδο του **phasor~**, δηλαδή μετά από κάθε κόκκο, η επόμενη περίοδος ξεκινάει από το σημείο που βρίσκεται το **phasor~** της "ένδειξης".

Τέλος, κάθε κόκκος περνάει από μορφοποιητή έντασης τύπου "hanning" για εξομάλυνση του ακουστού αποτελέσματος.



Μικροδομική Σύνθεση: Time Stretching

3ο στάδιο: Η εξομάλυνση δημιουργεί κενά ανάμεσα στους κόκκους (λόγω του “fade-in” και “fade-out” του hanning). Μπορούμε να βελτιώσουμε το αποτέλεσμα γεμίζοντας το κενό μετά από κάθε κόκκο πυροδοτώντας ένα αντίγραφό του μισή περίοδο αργότερα. Η πρόσθεση **+~ 0.5** μεταφέρει την περίοδο του **phasor~** στις τιμές 0.5 έως 1.5, και το αντικείμενο **wrap~**, το οποίο εκπέμπει τη διαφορά μιας τιμής σήματος προς την κοντινότερη μικρότερη ακέραια τιμή, κρατάει τις τιμές στο εύρος από 0 έως 1 με την επιθυμητή διαφορά φάσης ως προς το σήμα εισόδου.



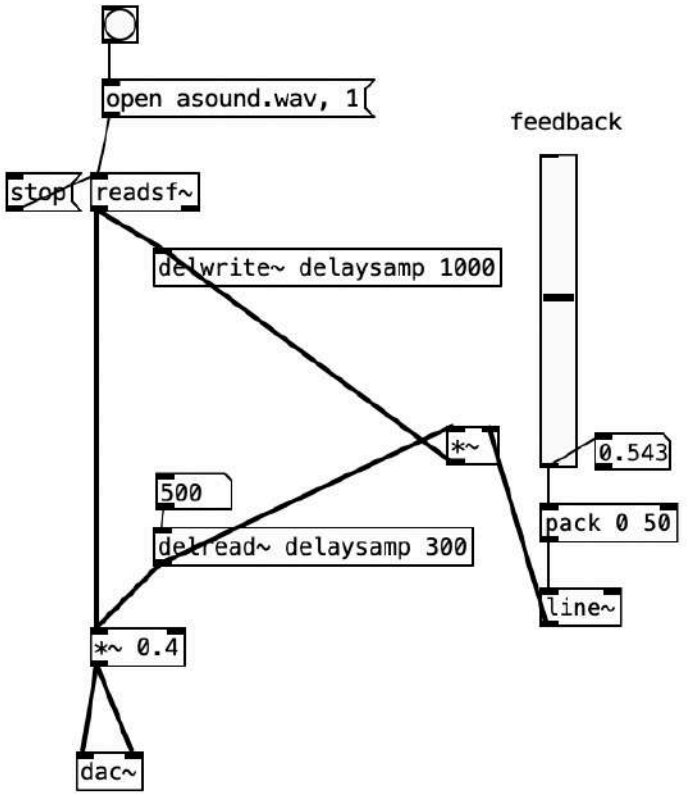
Σύνθεση με Σειρές Καθυστέρησης

Καθυστερούμε ένα σήμα σε ένα buffer με το αντικείμενο **delwrite~** και ορίσματα το όνομα του buffer (δεν χρειάζεται να το δημιουργήσουμε επιπλέον) και το μέγεθός του σε mlsecs.

Διαβάζουμε το καθυστερημένο σήμα με το αντικείμενο **delread~** με ορίσματα το ίδιο buffer και τον χρόνο καθυστέρησης.

Ανατροφοδοτώντας το καθυστερημένο σήμα στην αρχή της γραμμής καθυστέρησης (προσοχή: με τιμές μικρότερες του 1) περνάμε από το **delay** στο **echo**.

Επίσης, με τιμές χρόνου καθυστέρησης έως 10mlsecs δημιουργούμε **comb filtering**.

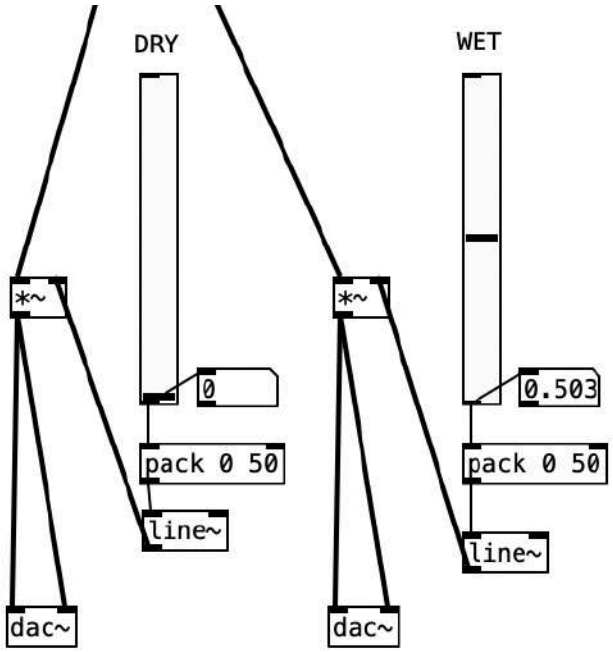


Σύνθεση με Σειρές Καθυστέρησης

Μπορούμε να ελέγχουμε την παρουσία του εφέ στο τελικό ακουστό αποτέλεσμα

(παράμετροι **dry** και **wet**)

στέλνοντας το αρχικό σήμα και το καθυστερημένο σήμα σε διαφορετικούς πολλαπλασιαστές σήματος (***~**)



Σύνθεση με Σειρές Καθυστέρησης

Για το εφέ της αντήχησης (**reverb**) το Pure Data έχει στοχευμένα αντικείμενα:

- Το **rev1~** δέχεται και εξάγει μονοφωνικό σήμα, και η μεσαία είσοδος καθορίζει τον χρόνο αντήχησης.
- Το **rev2~** δέχεται μονοφωνικό και εξάγει στερεοφωνικό σήμα.

Τα ορίσματά του κατά σειρά από αριστερά προς δεξιά καθορίζουν:

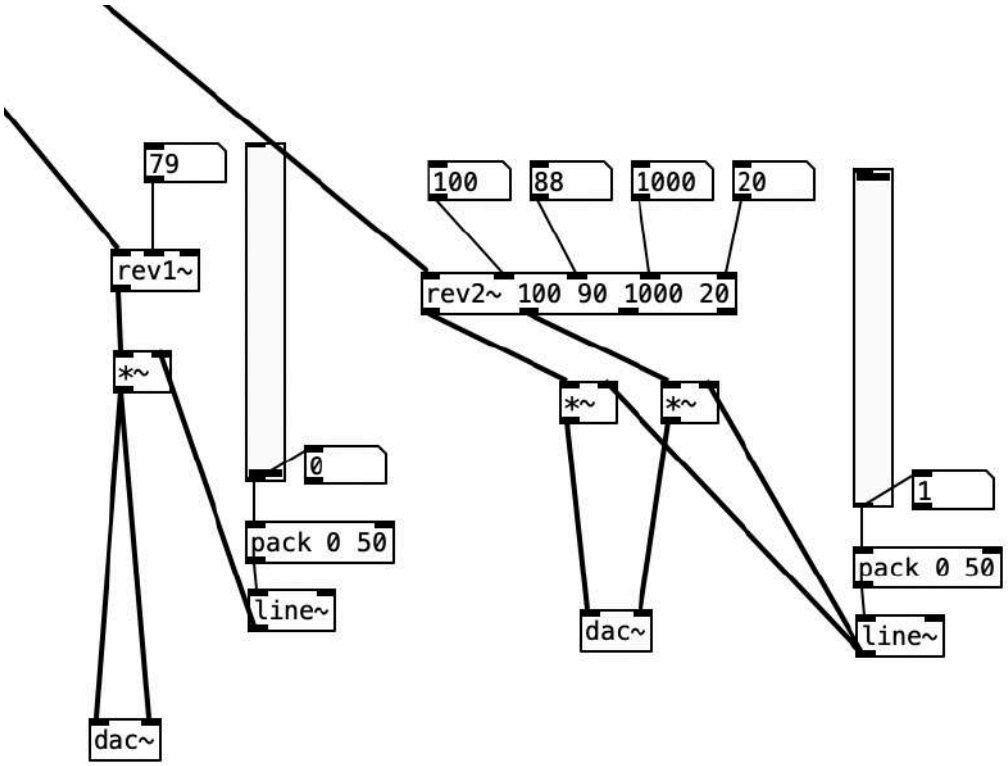
> την ένταση της αντήχησης,

> την ανατροφοδότηση,

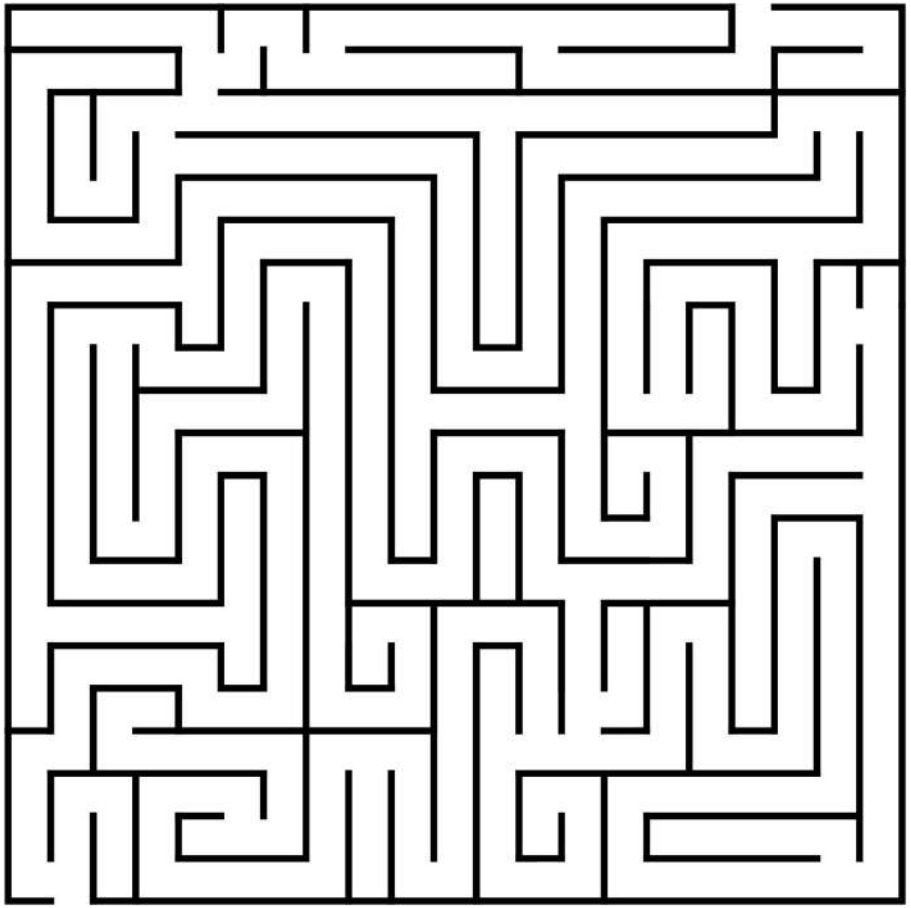
> το όριο του φίλτρου για δυναμική αποσιώπηση των υψηλών συχνοτήτων

> το μέγεθος της αποσιώπησης

Αντίστοιχα, το αντικείμενο **rev3~** δέχεται και εξάγει στερεοφωνικό σήμα



Ασκήσεις



Άσκηση 1: Ερώτημα

Εμφανίστε με μία ενέργεια το εξής μήνυμα στην κονσόλα:

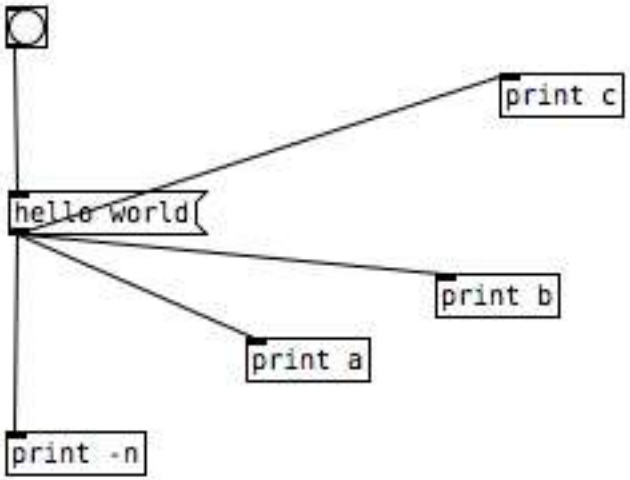
b: hello

c: hello

hello

a:hello

Άσκηση 1: Λύση



Έπειτα από πειραματισμο κατα τη συνοεση του μηνυματος με τα αντικείμενα print, αντιλαμβανόμαστε πως

η χρονική ροή των δεδομένων ακολουθεί τη σειρά της συνδεσμολογίας τους!

Επίσης, το όρισμα «-n» στο αντικείμενο **print** αποκρύπτει κάθε πρόθεμα του μηνύματος.

Άσκηση 2: Ερώτημα

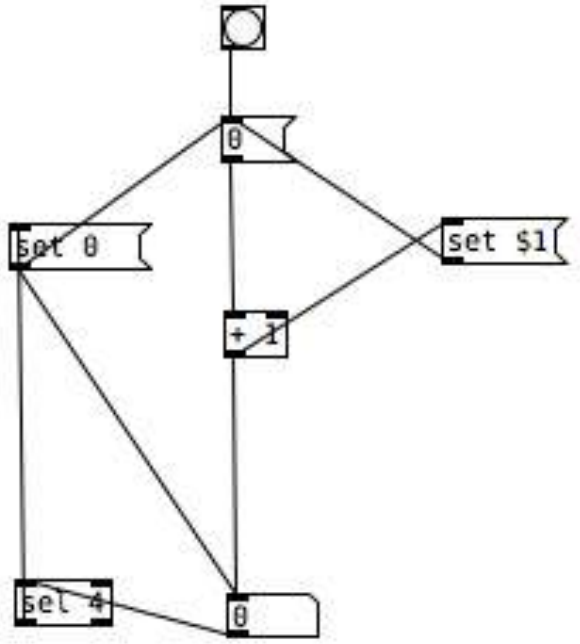
Δημιουργήστε έναν επαναλαμβανόμενο μετρητή 4 ενεργειών

Άσκηση 2: Λύση Α

Για να μετρήσουμε τα εισαγόμενα bangs, χρησιμοποιούμε το αντικείμενο της **πρόσθεσης** με το όρισμα 1. Κάθε αποτέλεσμα της πράξης επανατροφοδοτείται στην είσοδο του συστήματος, με χρήση του **set** για να αποφύγουμε ατέρμονη λούπα.

Έπειτα, με το αντικείμενο **select** και όρισμα το 4 μπορούμε να επανεκκινούμε το σύστημα κάθε 4 μετρήσεις.

Προσοχή στη σειρά της συνδεσμολογίας, ώστε το αποτέλεσμα της πρόσθεσης να πηγαίνει πρώτα στη μεταβλητή της ανατροφοδό και έπειτα να συνεχίζει στο υπόλοιπο πρόγραμμα!

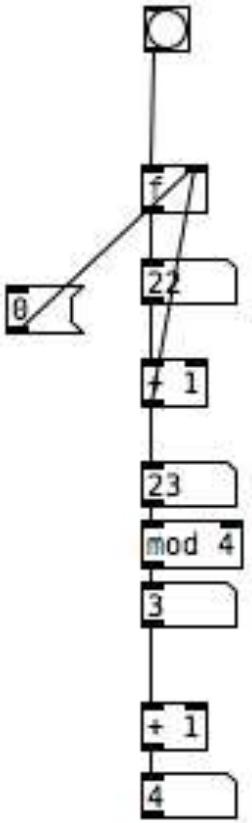


Άσκηση 2: Λύση Β

Εναλλακτικά μπορούμε να χρησιμοποιήσουμε το αντικείμενο **float (f)**, το οποίο αποθηκεύει μια αριθμητική τιμή. Η ευκολία εδώ είναι πως χωρίς τη χρήση του **set** μπορούμε να στέλνουμε το τρέχον αποτέλεσμα της πρόσθεσης στην κρύα είσοδο του **float**.

Επίσης, για την ομαδοποίηση των ενεργειών σε τετράδες, αξιοποιούμε το αντικείμενο **mod**, το οποίο διαιρεί την είσοδο με το όρισμα και αναφέρει το υπόλοιπο.

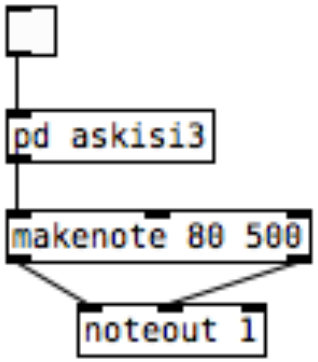
Προσοχή ότι το **mod** μετράει τις ενέργειες πάντα από το μηδέν!
Με τους κατάλληλους αριθμητικούς τελεστές
(σε αυτή την περίπτωση το αντικείμενο **+ 1** στο τέλος)
μπορούμε να προσαρμόσουμε το παραγόμενο στο επιθυμητό εύρος.



Άσκηση 3: Ερώτημα

Δημιουργήστε μια μελωδία με τυχαίες νότες από C4 έως C5 και τυχαίο διάστημα εκπομπής κάθε μισό, ένα, και ενάμιση δευτερόλεπτο με πιθανοκρατική κατανομή 50%, 20% και 30% αντίστοιχα.

Άσκηση 3: Λύση



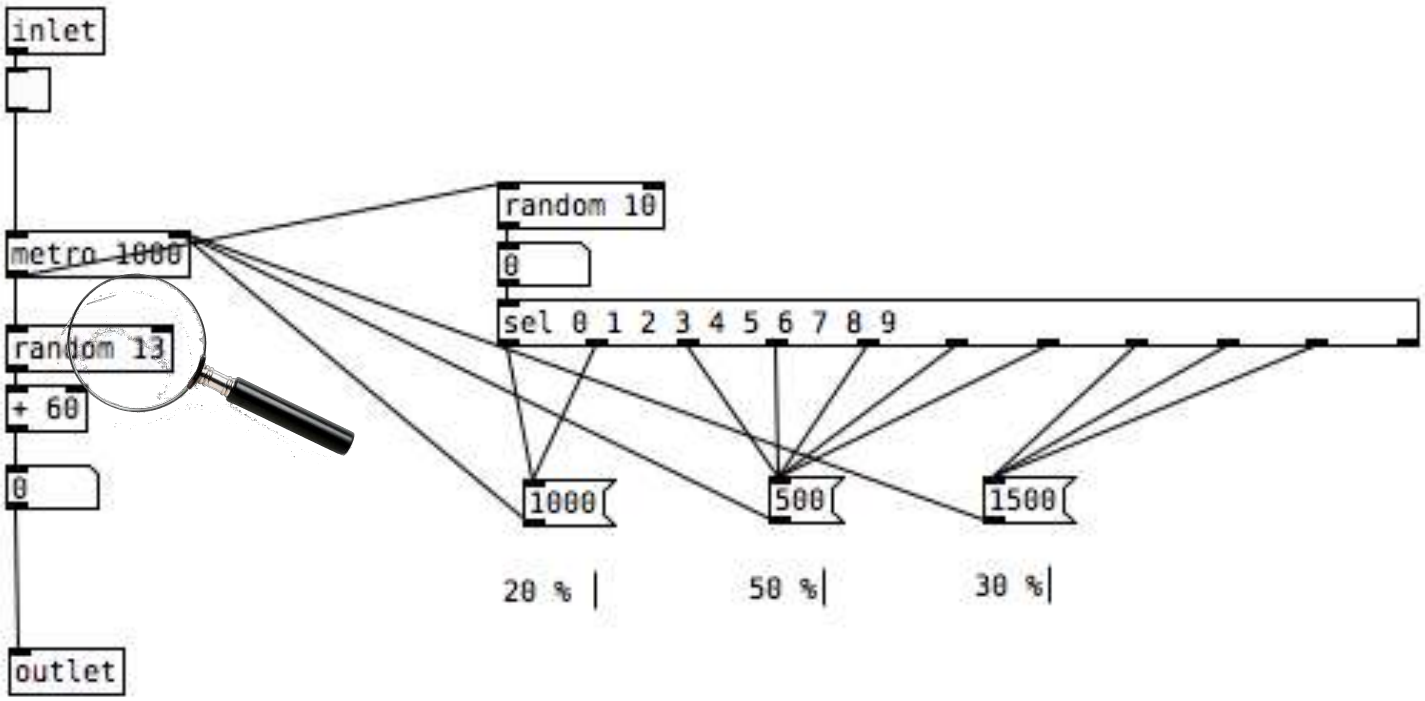
Αρχικά, θα τοποθετήσουμε το πρόγραμμα-ζητούμενο της Άσκησης
σε ένα υπο-πρόγραμμα (**pd Askisi3**),
με μία είσοδο, για να ενεργοποιεί ο χρήστης τη διαδικασία μέσω **toggle**,
και μία έξοδο, η οποία θα στέλνει midi-note τιμές στο αντικείμενο **maken**

Μέσα στο υπο-πρόγραμμα χρειαζόμαστε δύο πράγματα:

- μια γεννήτρια των επιθυμητών midi-note τιμών, και
- μια πιθανοκρατική δομή για την εμφάνιση τους.

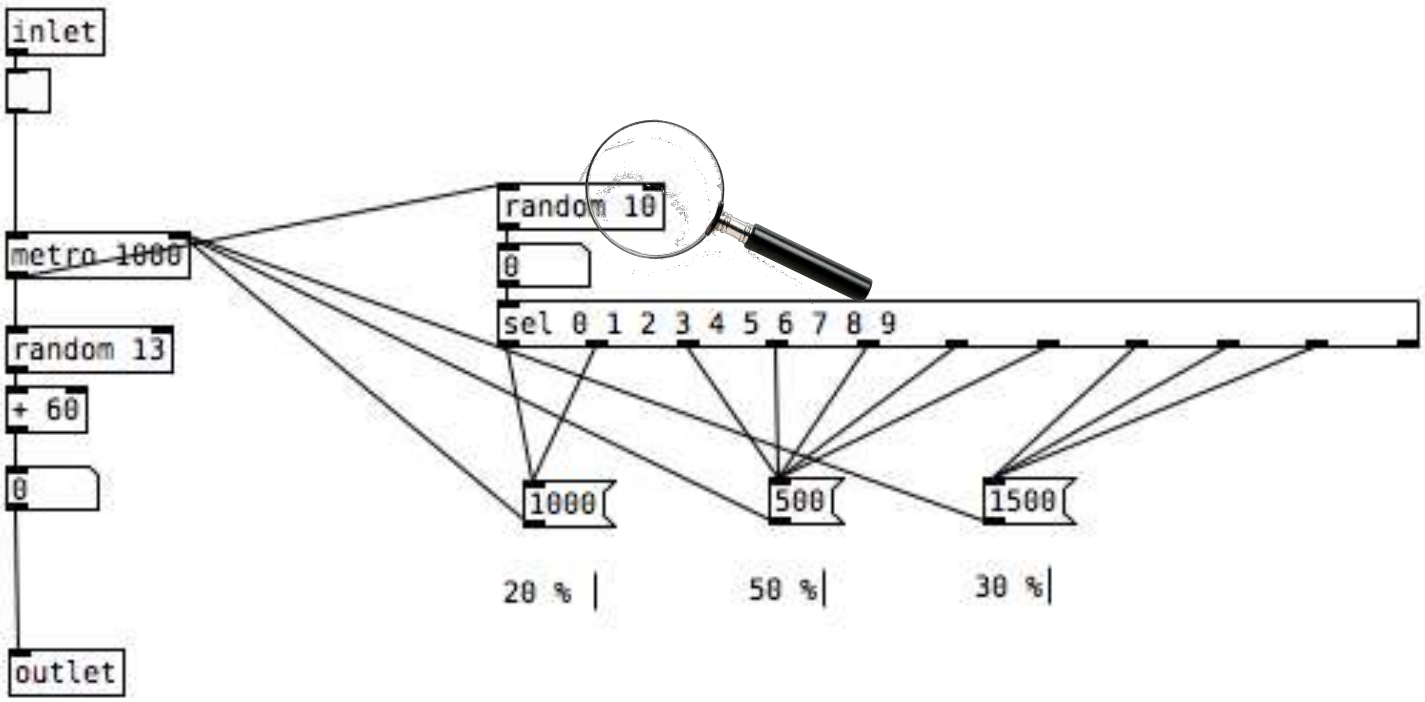
Άσκηση 3: Λύση

Αριστερά βλέπουμε τη γεννήτρια των midi-note τιμών μέσω του αντικειμένου **random**. Χρειαζόμαστε 13 διαφορετικές νότες, συνεπώς το **random** πρέπει να παράγει 13 τυχαίες τιμές, από 0 έως και 12. Τις οδηγούμε στο αντικείμενο πρόσθεσης **+ 60**, ώστε να επιτύχουμε το επιθυμητό εύρος 60 (C4) έως και 72 (C5).



Άσκηση 3: Λύση

Δεξιά βλέπουμε την πιθανοκρατική δομή, η οποία ελέγχει τον χρόνο εκπομπής **bang** του αντικειμένου **metro**. Ένα αντικείμενο **random** παράγει τυχαία τις τιμές 0 ώς και 9, οι οποίες πυροδοτούν τα μηνύματα 1000, 500, και 1500 (mlsecs) στο **metro**. Μέσω αντικειμένου **select** ορίζουμε την αναλογία 2/10, 5/10, και 3/10 αντίστοιχα.



Άσκηση 4: Ερώτημα

Δημιουργήστε μια γεννήτρια αρπισμάτων, ματζόρε και μινόρε τρόπου. Δώστε στον χρήστη δυνατότητες ορισμού:

- *της θεμελιώδους νότας του αρπίσματος μέσω ολισθητή,*
- *του τρόπου αρπίσματος μέσω των πλήκτρων M (ματζόρε) και m (μινόρε),*
- *της ταχύτητας του αρπίσματος σε bpm*

Άσκηση 4: Λύση

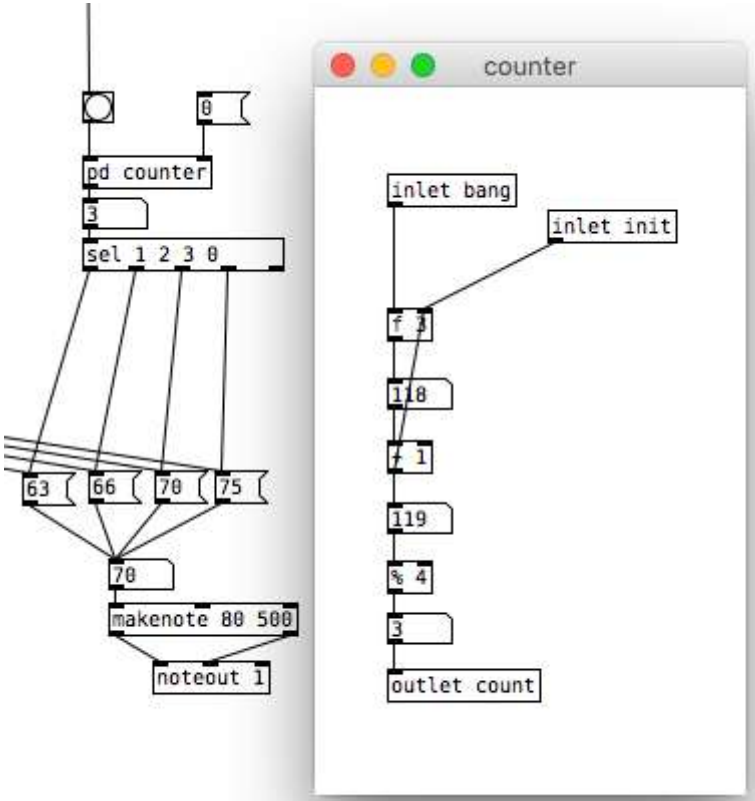
Για την υλοποίηση σύνθετων προγραμματιστικών δομών ενδείκνυται να δομούμε τη σκέψη μας σύμφωνα με μεμονωμένα modules που συναρμολογούν το τελικό επιθυμητό αποτέλεσμα. Αυτό διευκολύνει και τη διαδικασία ελέγχου για τυχόν λάθη, αφού προχωράμε στο επόμενο module όταν έχουμε βεβαιωθεί για τη σωστή λειτουργία του προηγούμενου.

Άσκηση 4: Λύση

Το κεντρικό μας module σαρώνει επαναληπτικά μια σειρά από **messages** με midi-note τιμές, και τις στέλνει στο αντικείμενο **makenote**. Αξιοποιούμε για τον “σαρωτή” μια προγραμματιστική δομή μετρητή που έχουμε υλοποιήσει σε προηγούμενη Άσκηση.

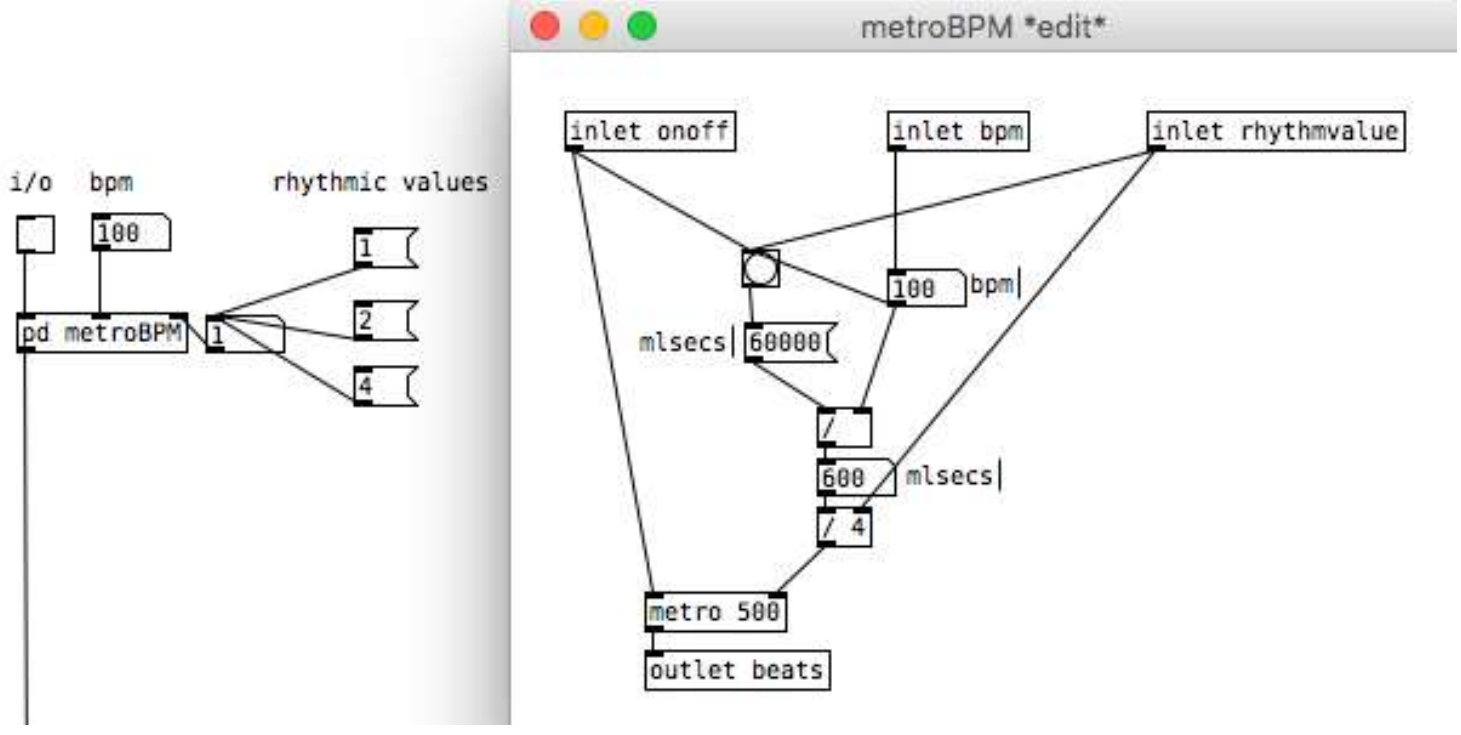
Προσοχή στο γεγονός πως στον συγκεκριμένο μετρητή χρησιμοποιείται το αντικείμενο **modulus**, το οποίο σε κάθε μέτρηση καταλήγει στην τιμή 0.

Το άρπισμα μας θα περιλαμβάνει 4 νότες, συνεπώς ο μετρητής μας θα επαναλαμβάνει 4 τιμές, κατά σειρά 1, 2, 3, και 0.



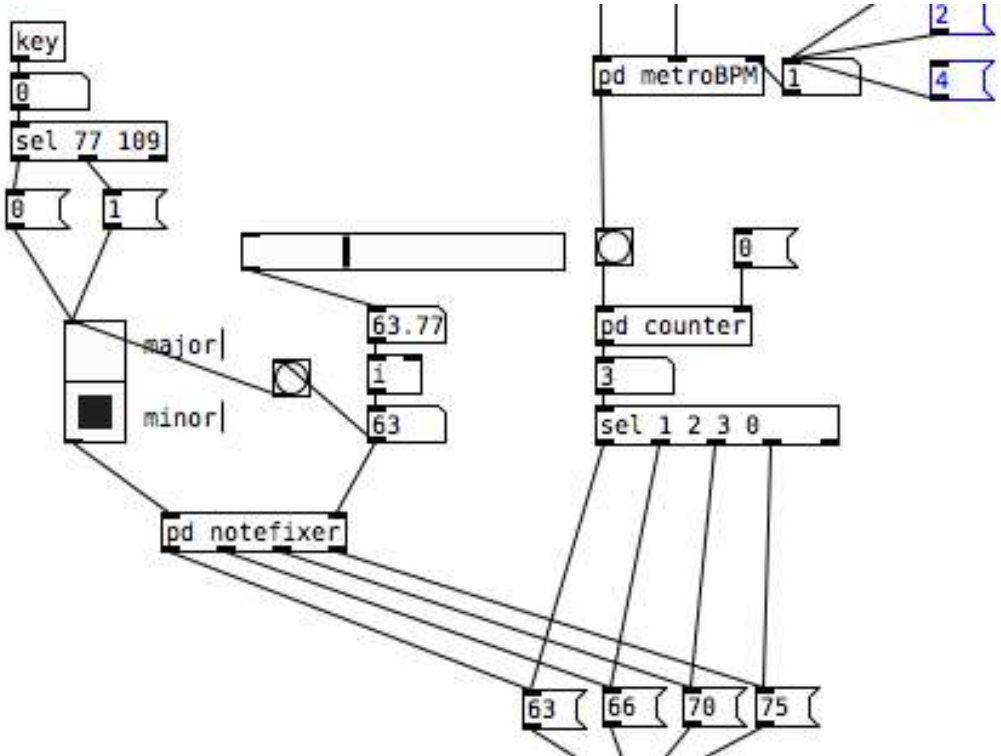
Άσκηση 4: Λύση

Το επόμενο module τροφοδοτεί το μετρητή μας με **bangs**. Χρησιμοποιούμε το αντικείμενο **metro**, όπου ο χρόνος εκπομπής υπολογίζεται βάσει bpm ως εξής: η τιμή 60 000 (τα mlsecs ενός λεπτού) διαιρούνται πρώτα με τα επιθυμητά bpm, και έπειτα με την επιθυμητή ρυθμική αξία. Στο παρακάτω παράδειγμα, έχει οριστεί ο μετρονόμος να εκπέμπει σε 100 bpm κάθε beat (ρυθμική αξία: 1/4). Προσοχή στο ότι κάθε αλλαγή πρέπει να επαναπυροδοτεί τη διαίρεση μέσω **bang**.



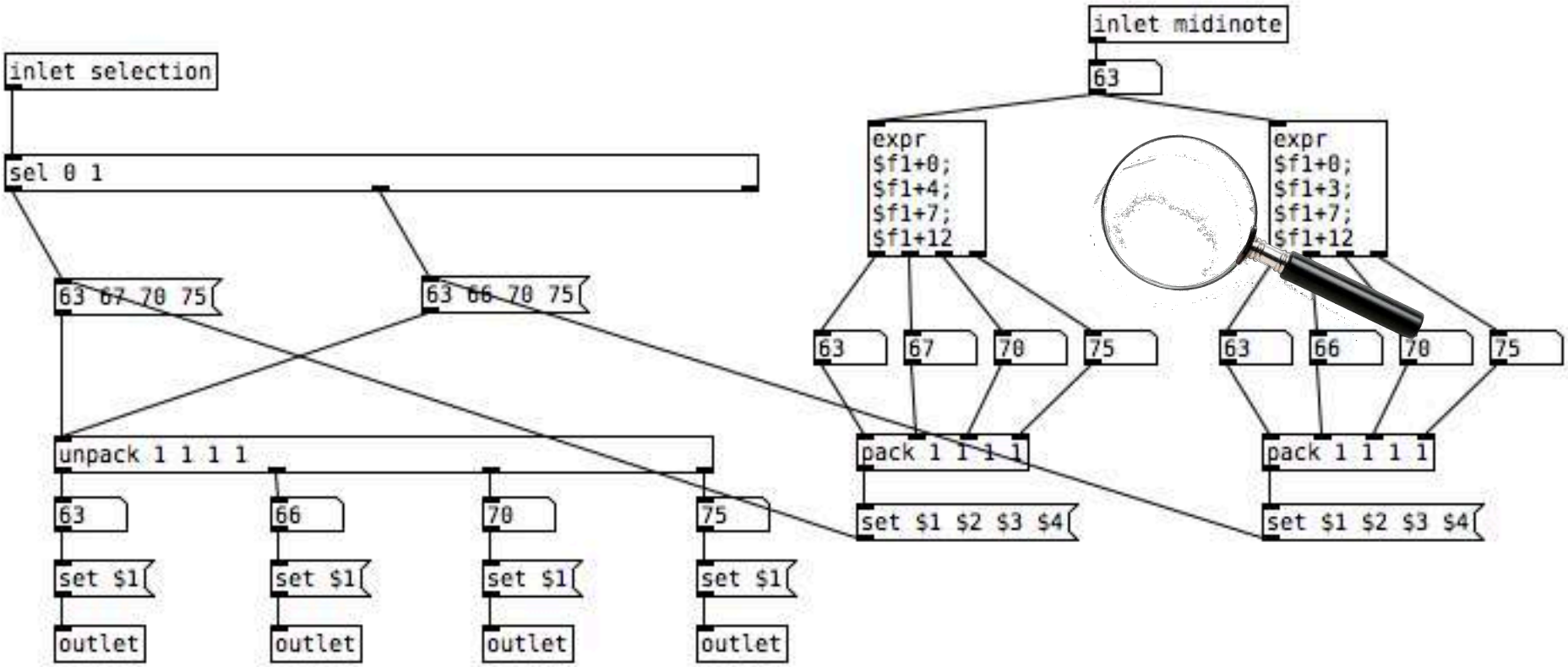
Άσκηση 4: Λύση

Χρειαζόμαστε τώρα ένα module που θα ορίζει ποιες midi-note τιμές θα σαρώνονται. Αυτό είναι το υπο-πρόγραμμα **pd notefixer**. Πριν δούμε το περιεχόμενό του, ας δούμε στο ανώτερο επίπεδο πως μέσω του αντικειμένου **key** ο χρήστης επιλέγει με “M” (ASCII τιμή: 77) το πρώτο κελί του **Vradio** και με “m” (ASCII τιμή: 109) το δεύτερο, ενώ με **Hslider** εισάγει στο υπο-πρόγραμμα τη θεμελιώδη midi νότα, η οποία πρώτα καταχωρείται και έπειτα πυροδοτεί με **bang** το **Vradio**.



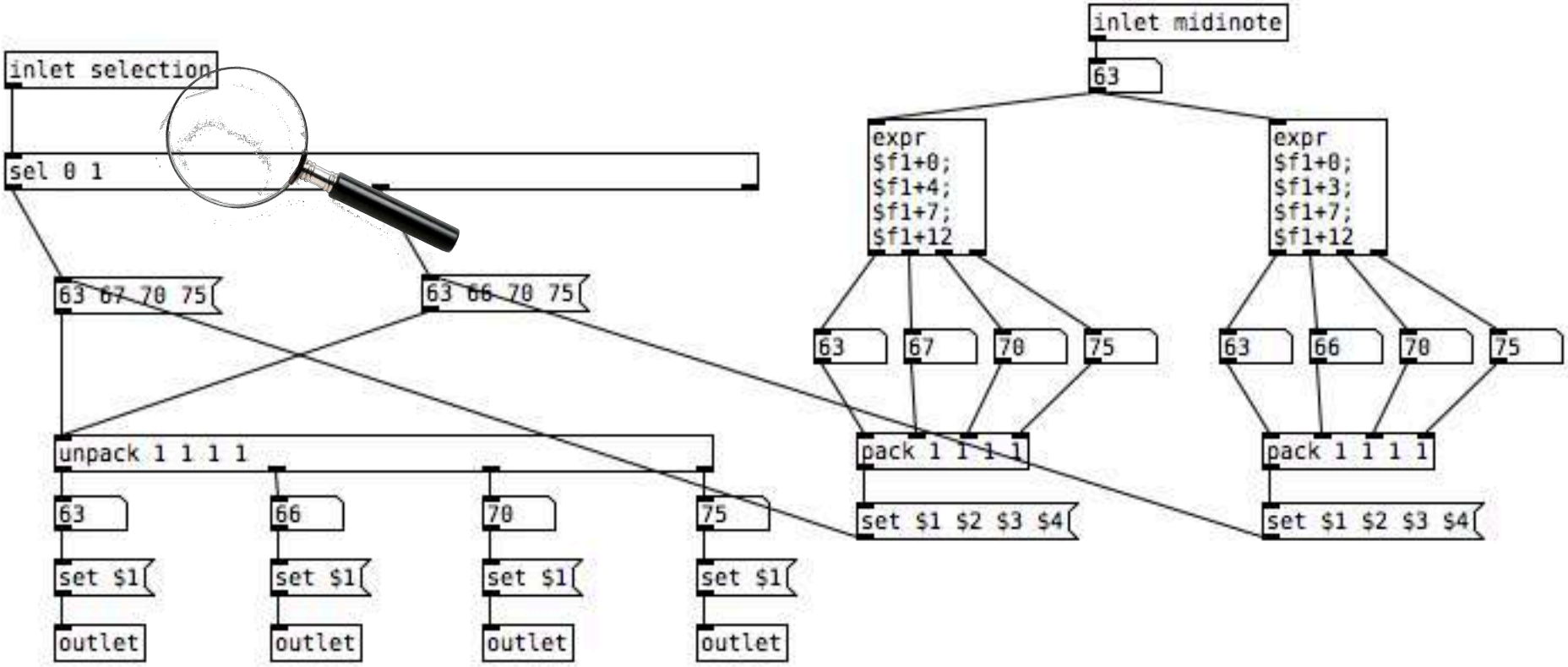
Άσκηση 4: Λύση

Μέσα στο υπο-πρόγραμμα **pd notefixer**, ας δούμε πρώτα δεξιά την τιμή midi-note να ενεργοποιεί τις πράξεις του **expr** για την παραγωγή των 4 νοτών των ματζόρε και μινόρε αρπισμάτων. Τα αποτελέσματα συλλέγονται σε λίστες μέσω των αντικειμένων **pack**, και μέσω **set** στέλνονται προς αναμονή πυροδότησης.



Άσκηση 4: Λύση

Αριστερά, κάθε επιλογή στο **Vradio** του άνω επιπέδου στέλνει την αντίστοιχη λίστα μέσω **unpack** ως μεμονωμένα στοιχεία στην αντίστοιχη έξοδο για τον σαρωτή του άνω επιπέδου. Κάνουμε χρήση **set**, ώστε να αποφύγουμε την άμεση πυροδότηση, καθώς θέλουμε να ενεργοποιείται η νότα μόνο τη στιγμή που τη σαρώνει ο μετρητής.



Άσκηση 5: Ερώτημα

Δημιουργήστε μια γεννήτρια ημιτονοειδούς ταλάντωσης .

Δώστε στον χρήστη τις εξής δυνατότητες ελέγχου της έντασης:

- *ευαισθησία στην παράμετρο midi velocity,*
- *περιβάλλουσα ADSR*
- *γενική ένταση του ηχητικού συμβάντος*

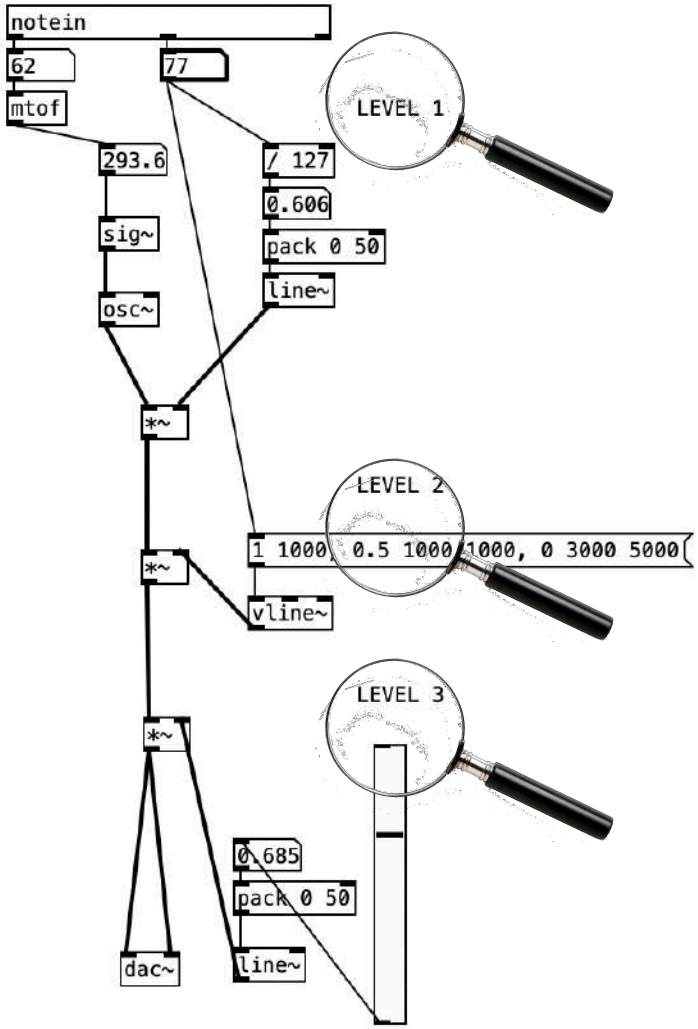
Άσκηση 5: Λύση

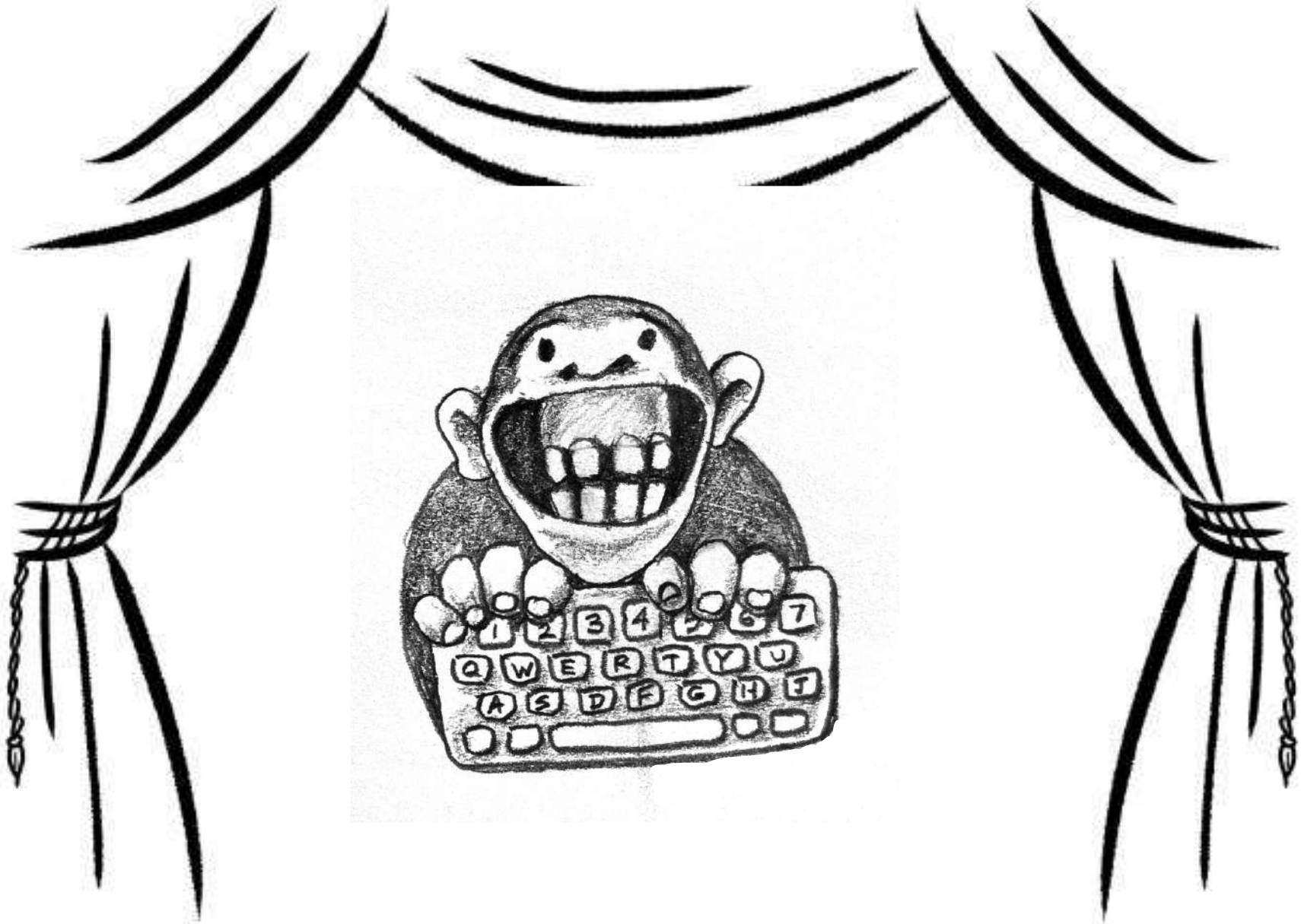
Κάθε επίπεδο ελέγχου της έντασης προϋποθέτει ξεχωριστό πολλαπλασιαστή σήματος. Συνεπώς θα χρησιμοποιηθούν 3 αντικείμενα ***~**.

- Στο πρώτο επίπεδο, η τιμή midi velocity (0-127) μεταφέρεται στο εύρος 0 -1 και τροφοδοτεί αντικείμενο **line~**

- Στο δεύτερο επίπεδο, ενεργοποιείται σειρά μεταβάσεων μέσω αντικειμένου **vline~**

- Στο τρίτο επίπεδο, ολισθητής ελέγχει τη γενική ένταση μέσω αντικειμένου **line~**




1 2 3 4 5 6 7
Q W E R T Y U
A S D F G H J