Pure Data Reference Card

K. Barkati & N. Granieri – November 15, 2018

Modes

`ctrl-e` (or `cmd-e`) toggle between *run* mode (performance) and *edit* mode (programming); this affects how mouse clicks affect the patch.

General

`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 messages
`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
`list`	manipulate lists

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 messages

Math

`expr`	C-style expressions
`+ - * / 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
`wrap`	wrap a number to a range [0, 1)

I/O via MIDI, OSC, and FUDI

`notein ctlin pgmin bendin touchin polytouchin midiin sysexin midirealtimein`	MIDI input
`noteout ctlout pgmout bendout touchout polytouchout midiout`	MIDI output
`makenote`	schedule delayed "note off" message for a note-on
`stripnote`	strip note-off messages
`oscparse oscformat`	OSC messages to and from Pd lists
`fudiparse fudiformat`	FUDI messages to and from Pd lists

Arrays / 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
`table`	create a named table
`array`	general array creation and manipulation

Misc

`loadbang`	bang on load
`declare`	set search path and/or load libraries
`savestate`	mechanism for saving state of an abstraction
`netsend`	send messages over the internet
`netreceive`	receive them
`qlist`	message sequencer
`textfile`	file to message converter
`text`	general text handling
`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

`expr~ fexpr~`	C-style expressions
`+~ -~ *~ /~`	arithmetic on audio signals
`max~ min~`	maximum or minimum of 2 inputs
`clip~`	constrict signal to lie between two bounds
`sqrt~`	approximate (16-bit) square root
`rsqrt~`	reciprocal square root
`q8_rsqrt~ q8_sqrt~`	fast, cheap 8 bits versions
`wrap~`	wraparound (fractional part)
`fft~`	complex forward discrete Fourier transform
`ifft~`	complex inverse discrete Fourier transform
`rfft~`	real forward discrete Fourier transform
`rifft~`	real inverse discrete Fourier transform
`pow~ log~ exp~ abs~`	math
`framp~`	output a ramp for each block
`mtof~ ftom~ rmstodb~ dbtorms~`	acoustic conversions

General Audio Manipulation

`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
`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 (RMS amplitude in dB)
`hip~`	high pass filter
`lop~`	low pass filter
`bp~`	band pass filter
`biquad~`	raw filter (2 poles and 2 zeros)
`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~`	time-reversed `rzero~`
`cpole~, czero~, czero_rev~`	corresponding complex-valued filters

Audio Delay

`delwrite~`	write to a delay line
`delread~`	read from a delay line
`delread4~ vd~`	read with a time-varying delay time

Subwindows

`pd`	define a subwindow
`inlet`	add an inlet to a pd
`outlet`	add an outlet to a pd
`inlet~ outlet~`	signal versions of `inlet` and `outlet`
`clone`	make copies of a subpatch
`block~ switch`	specify block size and overlap, or, if invoked as "switch", also switch subpatches on and off

Data Templates

`struct`	define a data structure
`drawcurve, filledcurve`	draw a curve
`drawpolygon, filledpolygon`	draw a polygon
`drawtext drawsymbol`	draw text
`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
`scalar`	create a single scalar

Extra (patches and externals in pd/extra)

`sigmund~`	pitch tracker
`bonk~`	attack detector
`choice`	best match of list to templates
`hilbert~ complex-mod~`	phase quadrature / frequency shifting
`loop~`	phasor~ with S/H on its frequency input
`lrshift~`	left and right shift (useful with FFT objects)
`pd~ stout`	run another copy of Pd (for multiprocessing)
`rev1~ rev2~ rev3~`	reverberators
`bob~`	Moog resonant filter model