Objects Covered Cheatsheet

Part 3

Object []	What it does	Example Arguments (where applicable)
[biquad~]	Two-Pole, Two-Zero Filter, with a ton of modes. You *can* set coefficients yourself (but that's insane), or just easily connect it to [filtergraph~] to set parameters.	
[filtergraph~]	Graphical filter editor. Use an [attrui] in the first inlet to set the mode, and graphically edit the cutoff, gain, and Q values. Connect the left outlet to the left inlet of [biquad~] or [cascade~] to set them.	
[cascade~]	Cascaded series of biquad filters. Use with a [filtergraph~] and [attrui] in it's left inlet set to nfilters to set each band when using peaknotch mode. VERY similar to a graphical equalizer in that respect.	
[comb~]	Comb filter effect. Mixes the current input with earlier input/output to get a sweeping sound. Arguments are max delay, init delay, gain, feedforward, and feedback. All are floats. **DO NOT GO ABOVE 0.99 FOR FEEDFORWARD OR FEEDBACK**.	[comb~ 15 1 0.25 0. 0.25]
[teeth~]	Alternate comb filter effect. Arguments are feedforward delay, feedback delay, gain, feedforward gain, feedback gain. All are floats. **DO NOT GO ABOVE 0.99 FOR FEEDFORWARD OR FEEDBACK GAIN.**	[teeth~ 15 1 0.25 0. 0.25]
[allpass~]	Allpass filter - flat magnitude response but interesting phasing. Typically delays sharp transients. Arguments are max delay, initial delay and gain. All are floats.	[allpass~ 100 30. 0.5]
[reson~]	Resonant bandpass filter. Arguments are the initial gain, center frequency and Q (try values of 1-100). All are floats.	[reson~ 1. 200 23]
[lores~]	Resonant lowpass filter. Much more efficient than stacking a [biquad~] and a [reson~]! Arguments are cutoff frequency and resonance (0-1). All are floats.	[lores~ 220 0.8]
[onepole~]	Single-pole lowpass filter. Very simple IIR filter with 6 dB attenuation per octave. Argument is the center frequency (floats).	[onepole~ 330]
[svf~]	State-variable filter with four simultaneous outputs. Outputs lowpass, highpass, bandpass, and notch from left to right. Arguments are the center (cutoff) frequency and the resonance. Both are floats.	[svf~ 700 0.7]
[poly~]	Manages polyphony and voice allocation for creating polyphonic instruments. Arguments are the name of an abstraction and the number of voices to allocate.	[poly~ synth_core 6]
[midiformat]	Essentially the inverse of [midiparse], minus a final corresponding inlet for (midievent)'s. Right output is a (midievent), perfectly formatted for [poly~] or [vst~].	
[in] and [in~]	Message and signal input for a patcher loaded into [poly~] or [pfft~]. Argument is the inlet number.	[in 2] [in~ 37]
[out] and [out~]	Message and signal output for a patcher loaded into [poly~] or [pfft~]. Argument is the outlet number.	[out 1] [out~ 99]

[s] and [r]	Send and receive messages and non-audio data without patch cords. Can also be coded as [send] and [receive]. Each takes the name of a specific bus to use. If you use multiple instances of [send] with the same bus, the values are summed at the [receive]. If you use multiple [receive] instances and one [send], they all receive the same message.	[send foo] [receive foo]
[polymidiin]	Like [midiin], but for MPE data. Always used inside of a [poly~] instance.	
[mpeparse]	Like [midiparse], but for MPE data. Outputs Note/Velocity pairs, Poly Pressure, CC's, Program Changes, Aftertouch, Pitch Bend, Voice Number, Zone First Channel, Zone Index, and mpeevent (for [vst~] only).	
[scale]	ESSENTIAL OBJECT! Maps an input range of floats or ints to an output range. Arguments are the input-low, input-high, output-low, output-high. They can be ints or floats and can be used to convert between the two.	[scale 0 127 0. 1.]
[clip]	Limits values to a certain range. If it exceeds the maximum value, it keeps outputting the maximum. Arguments are min and max values.	Specify for floats - [clip 0. 50.]