Objects Covered Cheatsheet

Part 8

Object []	What it does	Example Arguments (where applicable)
[p]	Creates a subpatch in a patch. The argument is the name of the embedded patch. Unlike abstractions, subpatches cannot be reused between different patches, but can be used multiple times within the parent patch. VERY useful for "hiding the wires" from users.	[p foo]
[inlet] and [outlet]	Receive and send data (including audio, Jitter, etc.) to/from a subpatch. Each [inlet] creates a new in on the top of the [p], while [outlet]'s generate outs at the bottom. The left to right order inside the subpatch determines the numbering.	
[pfft~]	Like [poly~] but for spectral processing using FFT's. It uses the companion [fftin~] and [fftout~] objects to separate/recombine a signal into Real and Imaginary components of the complex coefficient [a +b*i*] where *i* is √-1. See Puckette's *The Theory and Technique of Electronic Music,* chapter 9 for excruciating details. Arguments are the subpatch to load, the FFT size, and the overlap factor. FFT size MUST be a power of 2, and the overlap size must be a whole number greater than 0.	[pfft~ spectral 512 8]
[gizmo~]	A pitch shifter that works *inside* [pfft~]. Pair with [transratio] to easily shift the pitch of complex sounds.	
[retune~]	Easy way to retune a monophonic signal. Send it a MIDI note/velocity pair (or use the [kslider]/[pack] method in the help file) to enable notes to quantize to. It requires a non-zero velocity to activate the retuning, and a zero velocity to turn it off.	
[transratio]	A built-in abstraction that converts MIDI to pitch ratios for [gizmo~].	
[t b f]	ESSENTIAL OBJECT! Trigger sends an input to many outputs formatted right to left as it's arguments. Argument types can be bang (b), integer (i), float (f), list (l), symbol (s), or fully named symbols (e.g. 'foo'). Argument placement determines the output place - and it ALWAYS outputs right to left. UTTERLY ESSENTIAL for accurate timing of events that occur simultaneously or near-simultaneously.	[t b i s b f]
[f]	Stores a float. If it comes in the left inlet, it stores and outputs. If it comes in the right inlet it just stores it until it receives a (bang) in the left inlet.	
[i]	Stores a integer. If it comes in the left inlet, it stores and outputs. If it comes in the right inlet it just stores it until it receives a (bang) in the left inlet.	
[bpatcher]	Embeds an abstraction with a visible UI shown in the parent patch. The abstraction must be in presentation mode (use the Inspector to toggle "Open in Presentation"), and the UI is placed starting in the top left corner. The only argument is the name of the abstraction. I find it VERY helpful to enable the border in the [bpatcher]'s Inspector.	[bpatcher foo]

[pattr]	Connects to an object via it's middle outlet to bind it to a [pattrstorage] as a named object so you can save the state of objects inside of [bpatcher]'s, subpatches, or abstractions, and access them through presets. The argument is a name. Best practice is to start the name with #1_ which creates a unique instance on load so that you can have multiple copies of the same abstraction and named variables with different saved values.	[pattr #1_volume]
[pattrstorage]	Stores and recalls [pattr] presets using the (store X) and (recall X) messages. It can interpolate between presets by sending it a changing float (either manually or through [line]). Arguments are a unique name, and the @savemode attribute. I recommend setting that to 2 (attempt autosave, if not prompt to save).	[pattrstorage test @savemode 2]