ELSE - EL Locus Solus' Externals

for the Pure Data programming language

Version: 1.0-0 RC-12 (release candidate #12) With Live Electronics Tutorial.

Released August 06th 2024

About ELSE

This version of ELSE needs **Pd 0.55-0** or above.

ELSE is a big library of externals that extends the performance Pure Data (Pd) - Miller S. Puckette's realtime computer music environment (download Pd from: http://msp.ucsd.edu/software.html).

ELSE provides a cohesive system for computer music, it also serves as a basis for an Live Electronics Tutorial by the same author, yours truly, Alexandre Torres Porres. This library's repository resides at https://github.com/porres/pd-else/. This tutorial is also found as part of the download of the ELSE library. Look for the 'Live-Electronics-Tutorial' folder inside it and also check its README on how to install it.

ELSE is also part of PlugData by Timothy Schoen, which is a fork of Pd that loads as a standalone or VST with a revamped GUI. ELSE has received collaboration from Tim and others involved with PlugData and many objects have been included in ELSE just so they are supported in PlugData. See: https://github.com/timothyschoen/PlugData

This project is still in an experimental phase (currently at a 'release candidate' phase), where changes

may occur and backwards compatibility is not guaranteed until a final and more stable release is available.

Downloading ELSE:

You can get ELSE from https://github.com/porres/pd-else/releases - where all releases are available, but ELSE is also found via Pd's external manager (In Pd, just go for Help => Find Externals and search for 'else'). In any case, you should download the folder to a place Pd automatically searches for, and the common place is the **~/Documents/Pd/externals** folder. Instructions on how to install and build ELSE are provided below.

Installing ELSE:

This library runs in Pd Vanilla and do I still need to say it doesn't run in the long dead "Pd Extended"? Neither its new (and unfortunate) reincarnations "Pd-L2ork/Purr Data". Nevertheless, it can run in other forks that are actually compatible to Vanilla, most notably, ELSE is included in the PlugData fork --> https://github.com/timothyschoen/PlugData and you don't need to install it there.

ELSE comes mostly as a set of separate binaries and abstractions, which work if you just add the 'else' folder to the path via "Preferences => Path" or via **[declare -path else]**. Nonetheless, you must also load ELSE as a library via "Preferences => Startup" or **[declare -lib else]**.

Loading the library loads an object browsing plugin. It also registers a loader that allows Pd externals written in Lua (with the "*.pd_lua" extension) to be loaded. Some objects in ELSE are coded in lua, like the [lua] object itself. Loading the library also prints information of what version of ELSE you have when you open Pd. Check else-help.pd for more details.

Building ELSE for Pd Vanilla:

ELSE relies on the build system called "pd-lib-builder" by Katja Vetter (check the project in: https://github.com/pure-data/pd-lib-builder). PdLibBuilder tries to find the Pd source directory at several common locations, but when this fails, you have to specify the path yourself using the pdincludepath variable. Example (assuming the unpacked Pd package is in ~/pd-0.54-1; for Windows/MinGW add `pdbinpath=~/pd-0.54-1/bin/`):

```
make pdincludepath=~/pd-0.55-0/src/
```

Installing with pdlibbuilder

Go to the pd-else folder and use "objectsdir" to set an *absolute* path for your build, something like:

```
make install objectsdir=~/else-build
```

Then move it from there to your preferred install folder for Pd and add it to the path. Cross compiling is also possible with something like this

```
make CC=arm-linux-gnueabihf-gcc target.arch=arm7l install objectsdir=~/else-build
```

Installing [sfload~], [play.file~], [sfont~], [sfz~], [plaits~] and [circuit~]

For technical reasons these objects reside in their own subdirectories of the ELSE source directory (note, however, that [sfload] and [play.file~] is on a single 'play.file~' folder). This means that a normal build as described above will skip them. You can install these separately and you can also check their subfolders for their own READMEs and instructions.

To install them with the rest of ELSE you can build from the toplevel source directory using the special 'sfont', 'sfz', 'plaits', 'play.file' and 'circuit' targets, such as:

```
make sfz plaits pdincludepath=~/pd-0.54-1/src/
```

```
make sfz-install plaits-install objectsdir=~/else-build
```

For [sfont~] you should do the same but you also must run a script in the sfont subfolder to copy the needed dependencies, check its readme for more details.

More About ELSE

"EL Locus Solus" is run by yours truly, Alexandre Torres Porres, and it organizes cultural events/concerts and computer music courses (http://alexandre-torres.wixsite.com/el-locus-solus). The course uses Porres' Live Electronics Tutorial as a main textbook. The tutorial is here: https://github.com/porres/Live-Electronic-Music-Tutorial. This tutorial solely depends on the ELSE library and is a great didactic companion to this library. Both the library and the tutorial are provided as a single download, directly via Pure Data or GitHub.

The examples from the first incarnation of this tutorial were first developed for the now abandoned Pd Extended, making extensive use of the existing objects available in Pd Extended's libraries. Even though Pd Extended had many externals, there was the need at some point for something "else" - thus, this library emerged with the goal of providing more objects to include missing functionalities in the Pd Ecosystem.

But the library grew to encompass functionalities found in other Pd objects/libraries from old Pd Extended as well, with a different design and more functionalities. This was done in order to remove ALL the dependencies of the didactic material from these other libraries - with the goal to rely on just a single library that's alive (in active development) instead of many projects that are now long gone abandoned or not receiving much attention. I'm also involved in maintaining Cyclone, a legacy library for Pd (see: https://github.com/porres/pd-cyclone). But ELSE also superseeds cyclone for the purposes of this didactic material. See below in this document a list of alternatives to Cyclone provided by ELSE.

The goal of ELSE also outgrew the didactic material as it now includes objects not necessarily depicted in the computer music examples. Moreover, even basic elements from Pd Vanilla are being redesigned into new objects. So that's it, ELSE is becoming a quite big library and keeps growing and growing. A recent addition to it is 'M.E.R.D.A' a set of abstractions inspired by EuroRack modules.

ELSE has been in active development since early 2017 for real, but it hasn't stabilized into a final version yet. The aim is to do so as soon as PlugData's 1.0 version comes out! For now, it's at a "Release Candidate" stage of development, where changes may occur and backwards compatibility is not guaranteed until a final release is available.

Acknowledgements

Flávio Luis Schiavoni helped me out in a few things when I first started coding and collaborated with the objects: [median~] and [keyboard].

I'd also like to thank my Cyclone buddies Derek Kwan and Matt Barber, cause I started learning how to code externals with them as part of the cyclone team. Other developers of cyclone need to be praised, like Czaja, the original author, as I did steal quite a bit from cyclone into ELSE. I'd like to give a special thanks for Matt Barber for developing the "magic" in cyclone that I'm using here and also collaborating to ELSE with the objects: [float2bits], [brown~], [gray~], [perlin~], [pink~] and [blip~].

Lucarda is an active tester and has helped countless times with compilation issues for windows and more. Seb shader is a tcl/tk master that helped me a lot with this (which I know next to nothing) and is responsible for the [keycode] object.

Kudos and thanks to my buddy Esteban Viveros for helping with the compilation of ELSE for other systems as well as ELSE for Camomile and libpd projects, which opened the way for PlugData by Timothy Schoen, who's doing an amazing jaw dropping job with this project based on camomile that includes ELSE. Timothy has also helped me fix many ELSE related issues and has made incredible and countless contributions to ELSE. Other folks from the PlugData gang are being very helpful like Alex Mitchell and Amy. PlugData is a fork of Pd with a revamped GUI and comes with the ELSE library. See: https://github.com/timothyschoen/PlugData.

Current Object list (529 objects):

assorted

```
else
```

gui

```
knob numbox~ drum.seq bicoeff pad messbox mtx.ctl biplot zbiplot pic colors functio
```

time

```
chrono datetime
```

fft

```
hann~ bin.shift~
```

table

```
buffer tabgen tabreader tabreader~
```

tuning/notes

```
scales scale2freq scala autotune autotune2 makenote2 retune eqdiv cents2scale scale
```

patch/subpatch management

```
loadbanger args meter presets dollsym sender receiver retrieve dispatch var send2~
```

message management

```
format swap2 nmess unite separate symbol2any any2symbol changed hot initmess messag
```

list management

```
break order combine delete remove equal group iterate insert scramble sort reverse
```

file management

```
dir
```

midi

```
midi midi.learn midi.in midi.out sysrt.in sysrt.out ctl.in ctl.out touch.in touch.o
```

osc

```
osc.route osc.format osc.parse osc.send osc.receive
```

math functions

```
add add~ median avg mov.avg count gcd lcm frac.add frac.mul ceil ceil~ factor floor
```

math conversion

```
hex2dec dec2hex bpm car2pol car2pol~ cents2ratio cents2ratio~ ms2samps ms2samps~ db
```

math: constant values

```
sr~ nyquist~ pi e
```

logic

```
loop
```

audio multichannel tools

```
nchs~ sigs~ repeat~ select~ pick~ get~ sum~ merge~ unmerge~ slice~
```

analog circuitry emulation

```
circuit~
```

scripting

```
lua
```

fx: assorted

```
downsample~ conv~ chorus~ shaper~ crusher~ drive~ power~ flanger~ freq.shift~ pitch
```

fx: delay

```
del~ fbdelay~ ffdelay~ revdelay~ filterdelay~
```

fx: dynamics

```
compress~ duck~ expand~ noisegate~ norm~
```

fx: reverberation

```
allpass.rev~ comb.rev~ echo.rev~ mono.rev~ stereo.rev~ free.rev~ giga.rev~ plate.re
```

fx: filters

```
allpass.2nd~ allpass.filt~ bitnormal~ comb.filt~ lop.bw~ hip.bw~ biquads~ bandpass~
```

sampling, playing, granulation

```
player~ gran.player~ pvoc.player~ pvoc.live~ batch.rec~ bach.write~ rec.file~ play.
```

synthesis: synthesizers

```
pm2~ pm4~ pm6~ sfont~ sfz~ plaits~ synth~
```

synthesis: granular

```
grain.synth~
```

synthesis: physical modeling

```
pluck~
```

synthesis: oscillators

```
cosine~ impulse~ impulse2~ parabolic~ pulse~ saw~ saw2~ oscbank~ oscbank2~ oscnoise
```

synthesis: chaotic, stochastic, noise

```
white~ brown~ perlin~ crackle~ cusp~ fbsine~ fbsine2~ gbman~ gray~ henon~ ikeda~ la
```

control: mouse/keyboard

```
mouse canvas.mouse keycode keymap keypress
```

control: fade/pan/routing

```
euclid fader~ autofade~ autofade.mc~ autofade2~ autofade2.mc~ balance~ pan~ pan.mc~
```

control: sequencers

```
score score2 pattern list.seq sequencer sequencer~ phaseseq~ impseq~ rec rec2
```

control: envelopes

```
adsr~ asr~ decay~ decay2~ envelope~ envgen~
```

control: ramp/line/curve generators

```
{amp~ susloop~ function~ slew slew~ slew2 slew2~ lag~ lag2~ glide glide~ glide2 gli
```

control: random/stochastic

```
rand.f rand.f~ rand.i rand.i~ rand.list rand.u rand.dist rand.hist histogram markov
```

control: control rate lfo

```
lfo phasor pimp impulse pulse
```

control: triggers

```
above above~ bangdiv chance chance~ dust~ dust2~ gatehold~ gate2imp~ pimp~ pimpmul~
```

control: triggers, clock

```
clock metronome metronome~ polymetro polymetro~ speed tempo tempo~
```

analysis

```
changed~ changed2~ detect~ lastvalue~ median~ peak~ tap range range~ maxpeak~ rms~
```

M.E.R.D.A.

(Modular EuroRacks Dancing Along)

A submodule of ELSE by Porres (this is also by Porres).

Modules list (20 objects):

classic

```
adsr.m~ lfo.m~ seq8.m~ vca.m~ vcf.m~ vco.m~
```

FX

```
chorus.m~ delay.m~ drive.m~ flanger.m~ phaser.m~ plate.rev.m~ rm.m~
```

oscillators

```
gendyn.m~ plaits.m~ pluck.m~ pm6.m~
```

tools

```
presets.m sig.m~
```

extra

```
brane.m~
```

ALTERNATIVES TO CYCLONE

ELSE offers alternatives to objects from the Cyclone library (a library that clones objects from MAX/MSP). The objects that have no similar counterpart in ELSE (at least so far) are: anal / buddy / capture / capture~ / coll / cycle / decide / decode / frameaccum~ / framedelta~ / funbuff / funnel / flush / forward / kink~ / linedrive / prob / match / maximum / minimum / mousefilter / next / offer / peak / prob / pv / spray /

substitute / teeth~ / trough / universal / vectral~

But, here are some considerations: - maximum/minimum => [array max] array min - for [capture~] you can use [print~] - for [kink~] you can use function~ - [spike~] => [else/status~] + [else/detect~] => [threshold~] + [timer] - [anal] is usually used with [prob] for markov chains, but you can use [else/markov] instead - [teeth~] is just a comb reverberator which can be constructed with [else/ffdelay~] + [else/fbdelay~] - [forward] => just message boxes with ";" - [flush] => [poly] has a flush option, so does [else/voices] - for [coll], you can use [text] which miller considers is a better design, but it's simpler.

Alternatives:

- 2d.wave~ => else/wt2d~ (sort of)
- +=~ / [plusequals~] => else/add~
- operators >~ / <~ / %~ / etc => else/op~ => [expr~]
- bitwise operators (bitand~ / bitor~, etc) => [expr~]
- accum => else/add - acos/acosh/acos~/acos~/asin/asin~/atan~/atan2~/asin/asinh~/cosh/cosh~/cosx~/sinh/sinh~/sinx~/tanh/tanh~/tanx~ => [expr]/[expr~]
- active => else/canvas.active
- append => [list append]
- atob/atodb~/dbtoa/dbtoa~ => db2lin/lin2db/db2lin~/lin2db/~
- bangbang => else/loadbanger => trigger
- borax / else/noteinfo
- average~ / avg~ => else/mov.avg~
- bondo => else/hot
- buffer~ => else/sample~
- buffir~ => else/conv~
- counter => else/count
- changed => else/changed~
- click~ => else/impseq~
- clip/clip~ => clip/clip~
- comb~ => else/comb.rev~
- count~ => else/ramp~
- cross~ => else/crossover~
- cycle~ => else/wavetable~ => tabosc4~
- curve~ => else/envgen~
- cartopol/poltocar/cartopol~/poltocar~ => else car2pol/pol2car/car2pol~/pol2car~
- degrade~ => else/crusher~
- drunk => else/drunkard
- delay~ => else/ffdelay~
- delta~ => [rzero~ 1]
- deltaclip~ => else/slew~ => slop~
- downsamp~ => else/downsample~
- edge~ => else/status~ => threshold~
- fromsymbol => else/symbol2any / else/separate
- grab => else/retrieve
- gate => else/router

- gate~ => else/xgate~
- histo => else/histogram
- index~ => else/ramp~
- iter => else/iterate
- join => else/merge
- listfunnel => else/order
- loadmess => else/initmess
- line~ => else/envgen~ => vline~
- lookup~ => else/shaper~
- lores~ => else/lowpass~
- mean => else/mov.avg
- matrix~ => else/mtx~
- maximum~/minimum~ => max~/min~/expr~ (totally unnecessary external)
- minmax~ => else/range~
- mstosamps~/samps2ms~ => else/ms2samps ms2samps~ samps2ms samps2ms~
- midiflush => else/panic
- midiformat/midiparse => midi in/out objects (else/note.in/note.out, etc)
- mtr => else/rec
- mousestate => else/mouse
- onebang => else/nmess
- onepole~ => lop~
- overdrive~ => else/drive~
- peakamp~ => else/peak~
- pak => else/pack2
- past => else/above
- peek~ => tabwrite
- phaseshift~ => else/allpass.2nd~
- phasewrap~ => else/wrap2~ => wrap~
- pink~ => else/pinknoise~
- play~ => else/tabplayer~
- poke~ => else/tabwriter~
- pong/pong~ => else/fold / else/wrap2 / else/fold~ / else/wrap2~
- pow~ => pow~ (totally unnecessary external)
- prepend => else/insert => [list prepend]
- round / round~ => else/quantizer / else/quantizer~
- rand~ => else/rampnoise~
- record~ => else/tabwriter~
- reson~ => else/bandpass~
- scale / scale~ => else/rescale / else/rescale~
- seq => else/midi
- speedlim => else/limit
- spell => [list fromsymbol]
- split => else/spread
- sprintf => else/format => makefilename
- sustain => else/suspedal
- switch => else/selector

- sah~ => else/sh~ => samphold~
- selector~ => else/xselect~
- slide~ => else/lag2~
- snapshot~ => else/s2f~ => snapshot~
- svf~ => else/svfilter~
- table => array
- tanh~ (again) => else/drive~
- thresh~ => else/schmitt~
- train~ => else/pulse~
- trapezoid~ => else/envelope~
- triangle~ => else/vsaw~
- trunc~ => else/trunc~
- thresh => else/combine
- togedge => else/status
- tosymbol => else/any2symbol / else/unite
- unjoin => else/unmerge
- urn => else/rand.seq
- uzi => else/loop
- xbendin/sbendin2/xbendout/xbenout2 => else/bend.in / else/bend.out
- xnotein/xnoteout => else/note.in / else/note.out
- wave~ => else/wavetable~
- zerox~ => else/zerocross~
- zl => several dedicated objects include functionalities from it, such as: else/group, else/scramble, else/sort, else/reverse, else/rotate, else/sum, else/slice and else/stream

GUI:

- scope~ => else/scope~ (not the same but quite similar)
- comment => else/note (actually, comment is deprecated and based on else's [note] now.