

ED SAMPLING SYSTEM V2 – ELENA DESIGN

*****PLEASE READ*****

THE NEW SAMPLING V2 LIBRARY AT PRESENT IS STILL CONSIDERED EXPERIMENTAL.
PLEASE TEST IT WELL!

THE OLD SAMPLING LIBRARY MODULES HAVE BEEN MOVED TO THE `_OBSOLETE/`
FOLDER AND **THEY WON'T BE UPDATED OR BUG-FIXED ANY MORE.**
AT PRESENT THEY AREN'T PLANNED FOR DELETION YET.
HOWEVER, YOU SHOULD PLAN A GRADUAL MIGRATION OF YOUR EXISTING PROJECTS
TO SAMPLING V2, ONCE IT CAN BE CONSIDERED STABLE.

ED Sampling System offers since its birth to SynthEdit users a BLOB-based modular system for sample acquisition, transmission, storage, processing and playback.

Starting from Elena Design's Module Pack ("EDM") v1.53, a brand new system has been devised, and which is implemented thru a new v2 sampling SEM library.

The new v2 Sampling System still adopts BLOBs for transmission, but is **NO LONGER COMPATIBLE WITH V1. AS SUCH, YOU MUST NOT CONNECT V1 AND V2 SAMPLING MODULES TOGETHER!**

Starting from v2, sample data are no longer transmitted directly: BLOBs now contain a simple control structure, which contains an identifier, the sample length and a *pointer to the actual remote sample data, which reside somewhere in memory.*

This choice was adopted for efficiency, to avoid a lot of pointless copy and caching operations for every BLOB pin, which resulted in massive CPU and RAM wastage, in particular for polyphonic modules and/or for long samples.

Internally, sample transmission and reception is provided by custom BLOB sample pin classes (both DSP and GUI), which take care of everything transparently.

THIRD PARTY DEVELOPERS ARE THUS NO LONGER ALLOWED TO EXPAND EDM SAMPLING SYSTEM WITH CUSTOM SAMPLING MODULES (IF EVER THEY DID AT ALL)

Samples are still transmitted on consistent audio block positions; however, **no more than one sample can be transmitted or processed for every audio block** (an audio block is usually 96 or 128 sample frames long, depending on your ASIO or DAW buffer size). This usually does not constitute a problem though, since samples are NOT meant to be "streamed".

The received samples are always read-only. This way, modules not meant to modify sample data (like oscillators, players, displays, analytics, ...) will just receive a pointer to the actual read-only

sample data; modules meant to modify samples instead shall perform an internal copy from the received pointer, and transmit the address of the local copy as result.

A special Patch Memory Sample is therefore required to store/retrieve samples in Presets, and to warrant DSP/GUI interfacing: **REGULAR PATCH MEMORY BLOB's CAN'T BE USED ANYMORE WITH THE NEW V2 SAMPLING SYSTEM!**

A special universal Patch Memory, which acts both as Patch Memory and Patch Memory Out, was judged the most convenient choice and has been provided.

A simpler volatile "bridge", to convert from DSP to GUI, mostly for display purposes, has been provided as well.

V2 BLOB samples can be safely switched, triggered or routed by any general purpose BLOB module as before.

At present, the only source of inefficiency left is the very Patch Memory Sample module, because of how SynthEdit manages Parameters and GUI/DSP communication. However, the day SynthEdit should provide a more efficient mechanism, the special Patch Memory Sample module can be simply adapted, without any need to modify any other Sampling v2 modules.

In theory a sample v2 length can range from 1 to 536,870,911 sample frames, but it is up to individual modules to limit the accepted size range, if needed.

Keep in mind that samples larger than 1,310,720 sample frames can NOT be stored in Patch Memory Samples at present (things might change in SynthEdit 1.6 though – see below).

ABOUT THE INFAMOUS 5MB SYNTHEDIT QUEUE LIMIT

At present, the infamous 5MB limit for transmitting Parameter values to DSP and for internal GUI/DSP pipeline communication still applies, and affects both the new Patch Memory Sample and DSP->GUI bridge.

Since Jeff on GitHub announced to have finally addressed this limit in SynthEdit 1.6, in vision of the future no more "palliative" measures are taken (i.e workaround attempts which however had their drawbacks, like the v1 Bridges), and sample loaders have no longer any option to limit the physical sample size.

Please do NOT ask to provide them again !

ABUSIVE USAGE

By choice, neither v1 nor v2 Sampling Systems were designed to work with arbitrarily long samples (i.e "recordings", or actual "audio tracks").

You are still advised not to abuse EDM Sampling System with operations on long samples which are better done with audio editing software or with your DAW.

In particular, EDM Sampling System does not support disk streaming, but every sample is loaded in memory entirely and saved to disc in one go.

Usage of very long samples may result in audio hiccups, performance drops or other unpredictable issues.

Please use EDM Sampling System judiciously. You are warned !

RANGE SELECTION - UNIFORMED RULES

All modules accepting a Range selection (Start, End) will now more strictly and consistently obey a standard behavior **unless specified otherwise**:

- Negative Start means invalid range (no selection)
- Start > End means invalid range as well (they will NOT be swapped internally)
- End is always clipped to sample length -1
- A negative End value is interpreted as relative to the sample length (e.g -1 = default = last sample)

POSSIBLE CRASHES DELETING GUI MODULES IN SE 1.4 - PLEASE READ

In Struct View (SE 1.4), when a GUI Sampling module which allocates its internal sample memory and which is connected to one or more receiving GUI Sampling modules gets deleted, a crash will likely occur, as soon as the receiving modules try to access the no longer existing sample data thru the original pointer. This does not seem to occur with SE 1.5, which evidently re-initializes all GUI modules every time a module is deleted or added. At present there is not an easy fix to this issue. PLEASE DO NOT REPORT SUCH CRASHES !

AVAILABLE ANTI ALIASING FILTERS

All modules which perform resampling (oscillators, players, resamplers, ...) now support true anti-aliasing for both upsampling (interpolation, to remove the “ghost” spectrum entering from the high end) and downsampling (decimation, to remove the reflected spectrum at high end), offering a choice from the following available filters:

- None: (nearest neighbor) no interpolation and anti-aliasing; really only useful to obtain deliberately a lo-fi effect; negligible CPU load
- Linear: (order 1+1 triangular kernel) very basic interpolation and aliasing reduction; consumes very little CPU

-Cubic: (order 2+2 cubic approximation of a windowed sinc) best compromise for moderate CPU load; very smooth interpolation, despite with some HF attenuation; aliasing in downsampling may still be slightly audible at high ratios (pitches) .

NOTE: an actual "Sinc 2" was not provided as order-2 choice, because the cubic approximation used, which is actually discontinue, for complicated mathematical reasons (different error distribution) presents with less artifacts than a short order-2 windowed sinc kernel, and in upsampling mode it is mathematically equivalent to a standard cubic interpolator

-Sinc 4: (order 4+4 true windowed sinc kernel) high quality interpolation with pretty no HF attenuation; aliasing no longer audible in downsampling; higher CPU load

-Sinc 8: (order 8+8 true windowed sinc kernel), extreme quality; very high CPU load

Keep in mind that, with exception of None, anti-aliasing is always performed when skipping samples (i.e when the increment of sample read position is > 1 , which happens when playing a sample at a higher pitch than its nominal one), whose effectiveness and CPU load depends on the selected filter. Always pay attention that CPU load will increase proportionally with the amount of skipped samples!