The ConvertWithMoss Manual

Jürgen Moßgraber

September 2, 2021

Contents

1 ConvertWithMoss

Converts multisamples in a specific source format to a different destination format. Furthermore, it can create multisample files from plain WAV files.

2 Installation

Run the matching installer for your operating system. After that you can start the application ConvertWithMoss.

IMPORTANT: Macos Mojave: Mojave prevents now software to run which is not authorized by Apple. But instead of telling you so, you get an error that the files are corrupted (so, your OS is lying to you now...).

To fix it open a console and enter the application folder:

```
cd /Applications/ConvertWithMoss.app
```

Then remove the evil flag with:

```
sudo xattr -rc .

[Enter your administrator password to execute the command]
```

Since this seems not to work for everybody, there is another solution:

Temporarily, disable the Gatekeeper with

```
sudo spctl --master-disable
```

Open the application (should work now). Close it and enable Gatekeeper again to feel safe...

```
sudo spctl --master-enable
```

The application should now run also with Gatekeeper enabled.

Finally, have fun.

3 Parameters respected for the conversion

The conversion process reads and write metadata (name, category, creator, description and keywords) if supported by the format. If the source format does not support the information a guessing algorithm is applied to the name.

Furthermore, samples can be grouped in velocity layers and key ranges. Each sample can have 1 or no loop and parameters for pitch and different playback parameters.

The converter does not support any sophisticated synthesizer parameters like envelopes, filters or modulation.

4 Usage

1. Select a source folder, which contains one or multiple folders with multisamples in the selected source format. The files can also be located in sub-folders.
2. Select the output folder where you want to create the multisamples. This folder must be empty. You can add a non-existing folder to the name, which then is automatically created. E.g. you could select the Desktop and then add a folder *Conversions*.
3. Press the *Convert* button to start the conversion. The progress is shown with notification messages in the log area, which you should check for potential errors like defect source files, skipped folder, etc. Alternatively, press *Analyse* to execute the same process as *Convert* but does not write any files. Use this to check for errors before finally running the conversion.

4.1 Options

- **Create folder structure**: If enabled, sub-folders from the source folder are created as well in the output folder. For example, if I select my whole "Sounds" folder, there are sub-folders like "Sounds\07 Synth\Lead\01W Emerson'70 Samples". In that case the output folder would contain e.g. "07 Synth\Lead\01W Emerson'70.multisample" if Bitwig multisample is selected as the destination format.
- **Add new files**: Starts the conversion even if the output folder is not empty but only adds files which are not already present.

5 Supported formats

The following multisample formats are supported as the source format:

1. WAV files (*.wav)
2. Bitwig Studio multisample (*.multisample)
3. SFZ (*.sfz)
4. SoundFont 2 (*.sf2)
5. DecentSampler (*.dspreset,* .dslibrary)

The following multisample formats are supported as the destination format:

1. Bitwig Studio multisample (*.multisample)
2. SFZ (*.sfz)
3. DecentSampler (*.dspreset,* .dslibrary)

5.1 Source formats

The following multisample formats can be the source of a conversion.

5.1.1 Plain WAV files

This is not a multisample format but a clever algorithm tries to detect the necessary information from each multisample file. It uses metadata found int the WAV file or from its' name.

WAV file can contain different sample formats. This converter supports (split) stereo uncompressed and IEEE float 32 bit formats.

All WAV files located in the same folder are considered as a part of one multisample. You can also select a top folder. If you do so, all sub-folders are checked for potential multisample folders. The algorithm tries to detect as much metadata as possible from the WAV files:

- Notes are first detected from the sample chunk in the wave file (if present). If this is not set different parser settings are tried on the file name to detect a note name (or MIDI note value).
- The category is tried to be extracted from the file name. If this fails it tries with the folder names (e.g. you might have sorted your lead sounds in a folder called *Lead*). Furthermore, several synonyms and abbreviations are considered (e.g. Solo as a synonym for Lead).
- Characterizations like *hard* are tried to be extracted with a similar algorithm as for the category.

5.1.1.1 Velocity layers Detected velocity layers will be equally distributed across the velocity range. E.g. if 2 layers are detected the first will be mapped to the velocity range of 0-63 and the second to 64-127.

- Detection pattern: Comma separated list of patterns to detect velocity layers. The pattern must contain a star character ("*"), which indicates the position which contains the layer number.
- Order of layer numbering: Enable to map velocity layers inversed. This means that the highest number will be mapped to the lowest velocity range.

5.1.1.2 Mono Splits

Stereo samples might be split up into 2 mono files (the left and right channel). This tool will combine them into a stereo file.

- Left channel detection pattern: Comma separated list of patterns to detect the left channel from the filename. E.g. "_L".
- Only WAV files in Mono or Stereo are supported.

5.1.1.3 Options

- Prefer folder name: If enabled the name of the multisample will be extracted from the folder instead of the sample names.
- Default creator: The name which is set as the creator of the multisamples, if no creator tag could be found.
- Creator tag(s): Here you can set a number of creator names, which need to be separated by comma. You can also use this to look up other things. For example, I set the names of the synthesizers which I sampled. My string looks like: “01W,FM8,Pro-53,Virus B,XV” (without the quotes).
- Crossfade notes: You can automatically create crossfades between the different note ranges. This makes especially sense if you only sampled a couple of notes. Set the number of notes, which should be crossfaded between two samples (0-127). If you set a too high number the crossfade is automatically limited to the maximum number of notes between the two neighbouring samples.
- Crossfade velocities: You can automatically create crossfades between the different velocity layers. This makes especially sense if you sampled several sample layers with different velocity values. Set the number of velocity steps (0-127), which should be crossfaded between two samples. If you set a too high number the crossfade is automatically limited to the maximum number of velocity steps between the two neighbouring samples.
- Postfix text to remove: The algorithm automatically removes the note information to extract the name of the multisample but there might be further text at the end of the name, which you might want to remove. For example the multisamples I created with SampleRobot have a layer information like "_ms0_0". You can set a comma separated list of such postfix texts in that field.

5.1.2 Bitwig Studio multisample

The parser can read all information from Bitwig Studio multisamples except the layer color, select and parameter 1 to 3, which are not mappable.

A Bitwig multisample file is a zip archive which contains all samples in WAV format and a metadata file in XML format. This converter supports (split) stereo uncompressed and IEEE float 32 bit formats for the WAV files.

5.1.3 SFZ

The SFZ format is a file format to define how a collection of samples are arranged for performance. The goal behind the SFZ format is to provide a free, simple, minimalistic and expandable format to arrange, distribute and use audio samples with the highest possible quality and the highest possible performance flexibility (cited from https://sfzformat.com/).

The SFZ file contains only the description of the multisample. The related samples are normally kept in a separate folder. The converter supports only samples in WAV format encoded as (split) stereo uncompressed and IEEE float 32 bit format.

There are currently no metadata fields (category, creator, etc.) specified in the format. Therefore, the same guessing logic is applied as with plain WAV files.

5.1.4 SoundFont 2

The original SoundFont file format was developed in the early 1990s by E-mu Systems and Creative Labs. It was first used on the Sound Blaster AWE32 sound card for its General MIDI support.

A SoundFont can contain several presets grouped into banks. Presets refer to one or more instruments which are distributed over a keyboard by key and velocity ranges. The sample data contained in the file is in mono or split stereo with 16 or 24 bit.

The conversion process creates one destination file for each preset found in a SoundFont file. The mono files are combined into stereo files. If the left and right channel mono samples contain different loops, the loop of the left channel is used.

5.1.5 DecentSampler

The Decent Sampler plugin is a free (but closed source) sample player plugin that allows you to play sample libraries in the DecentSampler format (files with extensions: dspreset and dslibrary). See https://www.decentsamples.com/product/decent-sampler-plugin/ The format specification is available here: https://www.decentsamples.com/wp-content/uploads/2020/06/format-documentation.html#the-sample-element

A preset file contains a single preset. A dspreset file contains only the description of the multisample. The related samples are normally kept in a separate folder. Only WAV files are supported. A dslibrary file contains several dspreset files incl. the samples compressed in ZIP format.

5.2 Destination formats

The following multisample formats can be the destination of a conversion.

5.2.1 Bitwig Studio multisample

This format can be loaded in the Bitwig Sampler device. It supports multiple layers, key and velocity crossfades as well as several metadata information: creator, sound category and keywords.

5.2.2 SFZ

Writes a SFZ file (see above) and puts all samples in a sub-folder with the same name.

5.2.3 DecentSampler

Writes a dspreset or dslibrary file (see above) depending out the setting. Samples are stored in a sub-folder with the same name.

Further options:

- Make monophonic: Restricts the sound to 1 note, use e.g. for lead sounds.
- Add envelope: Create 4 knobs to edit the amplitude envelope.
- Add filter: Adds a low pass filter and creates a cutoff and resonance knob for it.
- Add reverb: Adds a reverb effect and creates two parameter knobs for it.

6 Changes

6.1 2.2.0

- New: DecentSampler creator got some options to choose which controls to create and to make the sound monophonic.
- Fixed: WAV detector: Upper velocity layer was not always 127.

6.2 2.1.1

- Fixed: WAV detector did not read loops from WAV files.

6.3 2.1

- Fixed: WAV detector did also deliver results for empty folders.
- Fixed: Setup for created DecentSampler Filter and Reverb is working now.

6.4 2.0

- New: Added reading and writing of DecentSampler preset and library files.
- New: Improved note detection from file names.
- Fixed: SFZ detector - global_label was not read.
- Fixed: SFZ parser - Comments at line end were not removed which conflicted with attribute values.
- Fixed: WAV detector - Crash if left and right mono sample had different lengths.
- Fixed: Creating folders for SFZ could raise an exception.
- Fixed: Source and destination tabs could be removed.