herb sauce
by herb delux
Version 1.0 Documentation

How to Use

Introduction:

Why I did this:

I wanted to make 4.0 music (generally front facing (dual stereo), but not required) easily accessible, and to also support metadata (track info). With another stereo bus to mix with, you can split up your tracks/instruments. For example, now your kick can come out the 1,2 channels, and the bass out of the 3,4. This means those low end frequencies aren't competing anymore electrically (or digitally). You end up with way more dynamics/dynamic range, texture, soundstage and original fidelity. Instead of 8 tracks going into a 2 channel stereo output, you have 4 tracks into a stereo output, and another 4 into the other. You can also ride your limiters and compressors less, while still getting the loudness you want. Everything (processing, amps, speakers, etc) should hopefully be more in the optimal zone for performance and quality.

So I came up with the .herb file and this herb sauce software to try to make that happen. .herb files are PCM (pulse code modulation) encoded audio files, similar to a .wav file, with all integers written in big endian. Header info is unsigned integers, audio frames are signed. The header structure is also a bit different from a .wav file, and supports metadata (track info) tags.

.herb files technically support up to 16,777,215 channels of audio. This program plays them and creates them at 2.0 channels (stereo), 4.0 channels (quadraphonic), or 6.0 channels (hexaphonic). It can convert 1 (mono .wavs are treated as stereo), 2, 4, or 6 channel .wav files into 2, 4, or 6 channels .herb files. I can also convert 2x2 channel .wav files into a 4 channel .herb, or 3x2 channel .wav files into a 6 channel .herb.

It works in Windows, Mac (M1+ and Intel) and Linux (tested in a Debian based distro).

It doesn't need to install, and it doesn't save any settings (which can be a minor annoyance if your computer requires special settings). It doesn't connect to the internet or require admin priveledges or anything like that. Just double click it and it should run. It takes a few extra seconds to run in Mac for some reason.

1. .herb Player

This tab is used to open and play .herb files.

Note in the bottom right corner of the window, herb sauce will tell you how many output channels are available on your currently selected audio device (sound card). If you intend to listen to a 4 channel .herb file, you want at least 4 output channels available.

If there's 0 output channels listed (or not as many as you need) go to the Settings tab, click "Change audio device" and attempt to find an audio device with a proper amount of output channels. Make sure after submitting your selection to click "Save settings". You can always use the "Test 4.0 channels" or "Test 4.0 levels" buttons to check if your newly selected audio device is working properly. (More info on this later in this document).

If you click the "Open .herb file" button you can find and select the file you would like to listen to.

You'll notice the header information will appear below (the bit depth, header length, sample rate etc). If there was detectable metadata, that will appear even further below (track name, artists name, etc).

If there was metadata, you can also click the "Metadata items: 5" text, and it will open a text box with all the metadata that was found, so you can view it and event copy+paste it, should you want.

If you click "Play .herb file" it will begin playing the selected .herb file. If you're *not* in pre-buffer mode, the Play button will also turn into a Pause button while you're listening.

If you click "Stop playback" it will attempt to end playback.

Currently, herb sauce can only play .herb files that are 2 channels, 4 channel, or 6 channel. Assuming the files were made with this program, you can easily tell how many channels they have by the number before the file extension (e.g. name.2.herb, name.4.herb, name.6.herb).

Here is an example, using two mismatches pairs of speakers, of an intended 4.0 front facing audio setup would look like. Notice how it's symmetrical. The red would be connected to channels 1 and 2, and the black would be connected to output channels 3 and 4.

2. .herb Creator

This tab is used to convert 1 or more .wav files into a .herb file, and also append metadata (optional).

As mentioned earlier, herb sauce can play and also create .herb files with 2 channels, 4 channels, or 6 channels.

There are multiple ways to convert .wav's into .herbs. Here are those ways.

Note: Mono (1 channel) .wav files are treated as 2 channel files (the mono signal is duplicated into 2 separate channels).

You can convert:
-a 2 channel (or mono) .wav into a 2 channel .herb,
-a 4 channel .wav into a 4 channel .herb,
-a 6 channel .wav into a 6 channel .herb,
-2x 2 channel .wavs into a 4 channel .herb,
-or 3x 2 channel .wavs into a 6 channel .herb.

Metadata is optional. The way metadata structuring works in the header I think is very elastic (wtf that mean). If you want to add track info (metadata) click the "Add track info (optional)" button. Make sure to submit it with the "Submit track info" button that appears in the next window.

Note: You don't have to fill out all of the fields. Feel free to only fill out the ones you want.

Once you have added (or not added) track info, pick a file name. You don't need to add the .herb, herb sauce will handle that. If you accidentally add .herb, it should detect that and prevent the double file extension.

Click "Create .herb file".

Give it a moment (or many) to create the file. It will end up being exported into the same folder as the one herb sauce is stored in (or at least the herb sauce executable you ran for this session). So if you opened herb sauce from the Desktop, the .herb file will be saved to the Desktop. You can change the output folder location (see below).

Note: If you want to change the output folder location, go to the Settings tab, click "Select custom .herb output directory", select a new directory, and then click "Save settings" near the bottom right of the Settings tab. Now all .herb files you create will be output to this custom folder.

Once complete, text will appear below the "Create .herb file" button showing the info about the .herb file that was created.

3. Settings

Note: upon loading, herb sauce utilizes the Sounddevice package in Python for sending the audio stream to your soundcard or audio interface. However, some people (depending on their computer's specs and operating system) may find better performance using the Soundcard package. If you check the "Use Soundcard package" check box, and then hit "Save settings", herb sauce will close and re-open one second later, utilizing the Soundcard package instead.

This tab is used to modify all sorts of settings for audio playback, .wav file conversion, .herb file output location, color scheme and testing your 4.0 channels and levels using sine wave.

The "Change audio device" button will open a window that shows all your available audio devices that herb sauce can see. Find one that has a minimum of the number of output channels you need. For example, if you want to listen to 4.0 (quadraphonic) .herb files, you need at least 4 output channels available.

Once you select your device, click "Submit". Then click the "Save settings" button.

Blocksize:

Changing the blocksize can potentially improve audio performance when in the default mode (sounddevice mode, with pre-buffer off). This particular setting will change automatically once you make a selection, and will be applied to the next time you start playback of a song.

You can use the "Test 4.0 levels" or "Test 4.0 channels" buttons to test if the audio device is functioning properly.

Note: Sometimes "fake" audio devices that exist only digitally in your operating system will list many output channels, but not be usable for listening to music or audio. In that case, select another device that says it has enough output channels for your needs, and try again.

Under the "Output channels" section, you'll see the "Custom channel mapping" check box. If you check that and then click "Save settings", custom channel mapping will be activated.

Note: Custom channel mapping works for the following setups: Sounddevice package with pre-buffer enable, Soundcard package with pre-buffer disabled, Soundcard package with pre-buffer enabled. (Doesn't affect anything for Sounddevice package with pre-buffer disabled, which is the default setup when you first open the program. In that setup, the program will default to outputting from 1-2 and 3-4 no matter what).

You can then use the drop down menus for 2.0 channels, 4.0 channels, and 6.0 channels to set custom channels for output from your soundcard or audio interface. Make sure to click "Save settings" to save your choices.

On the right hand side, you'll see "Manual .wav header length override", with a small text box below it.

Most .wav files have a 44 byte header, some have a 46 byte header, and occasionally you find .wavs with other byte-length headers that don't conform to normal standards. herb sauce is pretty good (from my testing) as detecting 44 and 46 byte headers, but it's not perfect. If you're having trouble converting .wav files into .herb files properly, you can set a custom offset for the header length of the .wav files you are trying to read.

You may have to analyze the .wav file yourself to determine the actual header length. Once you know it (you can even try to guess to if you want and see what happens), type it into the text box.

Make sure to click the "Save settings" button to save the change.

The "Select custom .herb output directory" button will open up a file browser window. Here you can select the folder you want .herb files to output to when you create them.

Note: Be default, herb sauce will output .herb files to whatever folder the executable was run from.

Once you have selected the custom directory you want to save .herb files to, click the "Save settings" button to save the change.

(From above) Upon loading, herb sauce utilizes the Sounddevice package in Python for sending the audio stream to your soundcard or audio interface. However, some people (depending on their computer's specs and operating system) may find better performance using the Soundcard package. If you check the "Use Soundcard package" check box, and then hit "Save settings", herb sauce will close and re-open one second later, utilizing the Soundcard package instead.

"Delux mode on/off" will toggle a color theme change for the program.

The "Donate!" button opens a new window with links to my websites, code, a donation email via PayPal, and a little About section to learn more about the program on a basic level. The links are also provided at the end of this document.

The "How to use" button opens a new window with quasi-basic instructions on how to use the program. It also lists key bindings. See key bindings below.

The key bindings are:

-Control-O: Open .herb file
-Control-P: Play .herb file
-Control-S: Stop playback
-Spacebar: Press/click selected item or button
-TAB: Change selected item (cycles through selectable items in the current tab one by one)

Testing the program:

There are two button on the Settings tab related to testing the program. The first is "Test 4.0 channels". When you click this, a small popup will appear with information regarding how this test works. When you click "OK", the test begins shortly after. This particular test is designed to help you gauge whether

your soundcard settings, channel output settings, and speaker connections are set up and working properly.

The next test is “Test 4.0 levels”. This too will open up a popup with basic information about how this test works. This test is designed to test whether all of your speakers or amps are gain-staged properly. Each speaker should sound the same volume level as the others.

Each test uses sine waves generated by herb sauce to work. No .herb files required for this test.

The “Save settings” button saves changes made to your settings.

At the bottom of the Settings tab, you will also find your currently selected audio device listed with its name, and its max number of output channels.

Links:

https://github.com/herbdelux/herbsauce

If you want to support me:

https://deluxeherbs.net/artists/herbdelux/herbdelux.html
https://herbdeluxmusic.bandcamp.com

You can find .herb file documentation on the next page.

License:

The program herb sauce and the .herb lossless audio file format are released into the world, with the intention of being free and open source. Hopefully they are used for good purposes and not bad ones.

For further information regarding the licenses of the packages I import in my code, please check with their respective documentation, licenses, or developers.

The butter and peppers logo is a real picture I took with my phone of real peppers that I grew, with a real stick of butter that I bought, on a cheeseboard I have. I am making it public domain, since I use it throughout the open source herb sauce software.

herb

.herb File Format Documentation
by herb delux
Version 1.0

Introduction:

This document serves as a breakdown of the header structure and frame writing structure of a .herb audio file.

.herb audio files are lossless uncompressed PCM (pulse code modulation) encoded audio files, similar to a .wav file, written with all header and frames values using big endian. Header items are unsigned integers (when applicable) and frames are signed. .herb files also have a different header structure, and support metadata (track info) tags, as you can see below.

Because the header length is written into the header itself, even if you write a program that has problems processing the metadata (track info) or finding the final b'herb', you can still read the important information (header length, bit depth, number of channels, and sample rate) so that you can at least find and play the audio frames.

Header structure:

Header is a minimum of 50 bytes without metadata (track info). With metadata it is variable in length.

4 bytes: b'herb'

A byte string of 'herb' in lowercase. This symbolizes that the file is a .herb file and it's header is beginning.

4 bytes: b'pcmb'

This byte string represents that the file uses PCM (Pulse code modulation) for the audio with big endian for its header items and audio frames. This is intended to never change. It also represents the beginning of the main and most important part of the header: the information about bit depth, sample rate, number of channels, etc.

4 **bytes: header length**

An integer, unsigned, in big endian, representing the total length of the header for this file. This allows you, after reading the sample rate, bit depth, and number of channels, to know exactly where the audio frames start (in case of metadata corruption or issues, etc).

3 bytes: number of channels

An integer, unsigned, in big endian, for the number of audio channels embedded in the file. 16,777,215 is thus the max number of channels supported.

2 byte: bit depth

An integer, unsigned, in big endian, for the bit depth of the file. 65,535 is thus the max number of bits per sample supported.

4 **bytes: Sample rate**

An integer, unsigned, in big endian, for the sample rate of the file. 4,294,967,295 is thus the max sample rate supported.

10 bytes: full size of the audio data (audio frames), not including the header.

An integer, unsigned, in big endian, representing the size in bytes of the audio data portion of the file. If you add this with the header length, in theory, you should get the full size of the .herb file. Thus, this value plus the header length in bytes should also give you the full file size in bytes.

10 bytes: empty space of 0's for future additions

Written as an integer, unsigned, in big endian, not that it matters at the moment.

4 bytes: b'tags'

Signifies the potential beginning of the metadata section (track info). If the next byte (number of metadata items) is zero, then there is no metadata. If the value (of the next byte) is 1 or more, then there are metadata items.

1 byte: number of metadata items

An integer, unsigned, in big endian, representing how many metadata items or entries there are coming up in the header. 255 items max.

IF THERE IS NO METADATA (the previous 1 byte was the integer 0):

4 bytes: b'leaf'

This byte string represents the end of the header, and the beginning of the audio frames.

IF THERE IS METADATA (the previous 1 byte was an integer greater than 0):

4 bytes: Byte string of metadata (pepper) section name.

Examples: b'trck', b'arti', b'albm', b'year', etc

1 byte: length of the actual metadata information (the track name itself, the artists name, etc)

An integer, unsigned, in big endian. This represents the length in bytes of the actual important part of the metadata.

This value is allowed to be zero.

N bytes: the actual meta data. (n = the integer from the 2 bytes prior, if not zero)

Examples: “The Official Sound Track” or “Blues Singer” etc.
This section is n number of bytes long, with n being the integer gathered while reading the prior two bytes.

4 bytes: Byte string of metadata section name (next item of metadata)

Examples: b’trck’, b’arti’, b’albm’, b’year’, etc

1 byte: length of the actual metadata information (the track name itself, the artists name, etc)

An integer, unsigned, in big endian. This represents the length in bytes of the actual important part of the metadata.

This value is allowed to be zero.

N bytes: the actual meta data. (n = the integer from the 2 bytes prior, if not zero)

Examples: “The Official Sound Track” or “Blues Singer” etc.
This section is n number of bytes long, with n being the integer gathered while reading the prior two bytes.

Then keep repeating for all metadata sections you want to add.

Note: *The herb sauce program adds in up to 5 optional meta data tag sections: track name, artist name, album name, year released, and track number.*

However, you don’t have to follow this, as long as you follow the structure laid out for a .herb file itself in this document. You can have up to 255 metadata items, including custom ones you make yourself. For example b’vart’ could be a section for “Various Artists” or something like that. See the end of this section for recommended metadata tags.

Note: You can calculate the length of the metadata tags section in bytes, by taking the header length value and subtracting 50 (since the bare minimum of header data is 50 bytes)

4 bytes: b'leaf'

This byte string represents the end of the header, and in this case the metadata section too, and the beginning of the audio frames. Remember, there is only one b'leaf'. Its location in the header depends on how many metadata items there are or aren't.

Metadata tags:

Here are the recommended (but not required) metadata tags in the recommended (but not required) order.

b'trck' – Track name
b'arti' – Artist name
b'albm' – Album name
b'year' – Year released
b'tnum' – Track number
b'ofnm' – OF how many tracks total

b'comp' – Composer(s)
b'prod' – Producer(s)
b'feat' – Featuring artists
b'imag' – File name of PNG or JPG album art. Must exist in the same directory as the .herb file only
b'spos' – Speaker layout or positioning (hopefully will add to this down the line)
b'mixx' – Mixer engineer(s)
b'mast' – Mastering engineer(s)
b'gear' – Tools, software, or gear used
b'note' – Custom notes
b'not2' – Custom note 2

Feel free to create your own. You technically don't have to follow that order either, it's just recommended. You only need to follow the header structure laid out above (4 byte bytes string, 2 bytes for length of actual data, then actual data), and the frame writing/reading structure (reminder, .herb files are PCM encoded audio files).

For reading and parsing these tags I recommend using a dictionary as such (below is some pseudo Python code):

```
#b'tags'
bytetags = track.read(4)
#number of metadata items
tagnums = track.read(1)
tagnums = int.from_bytes(tagnums, 'big', signed=False)

tagdict = {
        b'trck' : " ",
        b'albm' : " ",
```

```
        }

for i in range(0, tagnums):

        tagval = track.read(4)
        taglength = track.read(1)
        taglength = int.from_bytes(tagength, 'big', signed=False)
        data = track.read(tagength)
        data = data.decode("utf-8")

        if tagval in tagdict.keys():
                tagict[pepperval] = data
        else:
                pass

        trackname = tagdict[b'trck']
        artistname = tagdict[b'arti']
        #continue as needed for the track info you want to parse or store
```

Audio Frames:

Audio frames are written similarly to a .wav file, except in a .herb file they are written in big endian. **(Big endian, signed integers)**

For a 2 channel .herb file at a bit depth of 24 (3 bytes), the frames are written: 3 bytes for sample #1 channel #1. Followed by 3 bytes for sample #1 channel #2. Then 3 bytes for sample #2 channel #1, followed by 3 bytes for sample #2 channel #2, and so on with that same pattern.

For a 4 channel .herb file at a bit depth of 24 (3 bytes), the frames are written: 3 bytes for sample #1 channel #1, 3 bytes for sample #1 channel #2, 3 bytes for sample #1 channel #3, 3 bytes for sample #1 channel number 4. Then 3 bytes for sample #2 channel #1, 3 bytes for sample #2 channel #2, 3 bytes for sample #2 channel #3, 3 bytes for sample #2 channel #4, and so on with that same pattern.

The same pattern of frame writing is used for .herb files regardless of how many channels there are in the file.

Short point about the "why" of this program and file type:

I wanted to make 4.0 music (generally front facing (dual stereo), but not required) easily accessible, and to also support metadata (track info). With another stereo bus to mix with, you can split up your tracks/instruments. For example, now your kick can come out the 1,2 channels, and the bass out of the 3,4. This means those low end frequencies aren't competing anymore electrically (or digitally). You end up with way more dynamics/dynamic range, texture, soundstage and original fidelity. Instead of 8

tracks going into a 2 channel stereo output, you have 4 tracks into a stereo output, and another 4 into the other. You can also ride your limiters and compressors less, while still getting the loudness you want. Everything (processing, amps, speakers, etc) should hopefully be more in the optimal zone for performance and quality.

(See how to support me and other license info on the next page!)

Other:

You can support me by purchasing my original music from Bandcamp or Deluxe herbs:

https://deluxeherbs.net/artists/herbdelux/herbdelux.html

https://herbdeluxmusic.bandcamp.com

For more information check out the GitHub page for my free open source .herb file player and creator called herb sauce at:

https://github.com/herbdelux/herbsauce

License:

The program herb sauce and the .herb lossless audio file format are released into the world, with the intention of being free and open source. Hopefully they are used for good purposes and not bad ones.

For further information regarding the licenses of the packages I import in my code, please check with their respective documentation, licenses, or developers.

The butter and peppers logo is a real picture I took with my phone of real peppers that I grew, with a real stick of butter that I bought, on a cheeseboard I have. I am making it public domain, since I use it throughout the open source herb sauce software.

herb