# Sonic Visualiser - Feature #1840
## Import Audio Data

2018-06-25 01:13 PM - Chris Cannam

| | | | |
|---|---|---|---|
| **Status:** | Closed | **Start date:** | 2018-06-25 |
| **Priority:** | Normal | **Due date:** | |
| **Assignee:** | | **% Done:** | 0% |
| **Category:** | | **Estimated time:** | 0.00 hour |
| **Target version:** | | | |

**Description**

We have Export Audio Data; how about Import?

### History

**#1 - 2018-09-06 12:43 PM - Chris Cannam**

Implementing this in the import-audio-data branch. As of commit:17c58d48acfe we have the fundamentals. Still to do:

- Handle scaling for sample values that are in a different range than [-1,1) (+ normalisation)
- Make sure the right thing is done with the Recent Files menu and Save option (probably avoid catching the eye of either of them)
- Test some more
- Thoroughly re-test the CSV import for annotation layers in case of regressions

**#2 - 2018-09-06 01:41 PM - Chris Cannam**

Another thing to do:

- Refactor CSVFormatDialog into separate classes for the audio and annotation versions. At the moment they use conditionals within the same class, but it's increasingly apparent that they differ in more than they have in common.

**#3 - 2018-09-07 04:11 PM - Chris Cannam**

Question: should we use this feature when a CSV file or similar is supplied as the only command-line argument? (Probably)

**#4 - 2018-09-07 04:20 PM - Chris Cannam**

Refactored CSVFormatDialog as described.

Have added normalisation option to WavFileReader, for use by WritableWaveFileModel when cacheing data from a CSV file with unknown range.

Still to do:

- Pull out CSVSampleRange enum from CSVAudioFileReader into CSVFormat. This will involve changing the range return/update code in the CSV format deduction code, but we need it there because it will be CSVFileReader (and not any GUI class) that actually applies the necessary scaling.
- Make CSVFileReader apply the necessary scaling.
- Wire through a normalisation option from WritableWaveFileModel through to the WavFileReader that reads the cached data. Use that in place of scaling when the scale factor is unknown (RangeOther).
- Write tests for WavFileReader with normalisation enabled.

Plus any remaining things listed earlier.

#### #5 - 2018-09-07 04:56 PM - Chris Cannam

Have added the sample range deduction support to CSVFormat - but not yet updated CSVAudioFileReader to use it.

#### #6 - 2018-09-10 03:37 PM - Chris Cannam

Have wired the change all the way through to the main UI. We now have working audio import.

However, there's still a significant conceptual issue to deal with, which is about how this interacts with session handling. Like the recording feature, audio import creates a new audio file in a private location, which will need to be transported along with the session file if the user ever wants to communicate it to someone else. Unlike recording, the audio file that is created by import is currently a temporary one with a default temporary filename which disappears on exit, so re-opening a saved session file will not succeed, and passing the session file to someone else is effectively impossible.

I can think of three options:

1. Continue to create a new audio file in a private location, but (as recording does) make it persistent and its location predictable. The file's location is listed in the Layer Summary dialog, which probably nobody really knows about. For recorded audio, File -> Browse Recorded Audio Folder exposes the saved files, but it would seem really heavy to add a similar feature for imported audio. Potentially the imported audio could go in a special subdirectory of the recorded audio folder, but that's a bit obscure. Or we could rename the menu function to say "recorded and imported" but ew.
2. Ask the user somehow (during import) where to save the resulting file to. This is awkward but would have the advantage of making it quite explicit what was actually going on.
3. Do not refer to the newly-created audio file in the session at all - refer to the original CSV file, plus the parameters that were used when importing it, as if that were an audio file itself. This could be supported by a different model type, e.g. a DataBackedWaveFileModel in addition to the current R/O and R/W sorts. This would be the most "audacious" option. The really big risk is that, by introducing another type of audio file, we have another source of breakage with future changes to anything else in the session or audio file handling. (Note that even the existing WritableWaveFileModel doesn't truly exist in the session - when a session is reloaded, it is reloaded using a normal ReadOnlyWaveFileModel because there's no need for it to be writable any more. So nothing in the session handling needs to know about WritableWaveFileModel, which would not be the case for this new type.)

#### #7 - 2018-09-11 06:05 PM - Chris Cannam

I went for option 1, using the term "convert" in menus ("Convert Audio from Data File..." and "Browse Recorded and Converted Audio") because "import" already has a slightly different meaning in this context.

There is still a problem. When the file needs to be normalised (because the incoming sample data doesn't fit a neat range), this happens on read, being done by the WavFileReader - but the normalisation flag is in the WritableWaveFileModel, the ReadOnly variant doesn't actually know about normalisation (yet). So when the session is saved and reloaded, the file is no longer normalised. Also, the file can't then be opened in any other session because there is no general facility to normalise audio on load.

It would be far better to normalise on write, but that would mean making two passes over the CSV data, or else saving more than one audio file (the first one being temporary).

#### #8 - 2018-09-12 03:31 PM - Chris Cannam

Updated to normalise the audio file on write (with logic in WritableWaveFileModel). This should be easier to maintain than the alternatives.

Remaining:

- Make sure the right thing is done with the Recent Files menu and Save option (probably avoid catching the eye of either of them)
- Test some more
- Thoroughly re-test the CSV import for annotation layers in case of regressions

#### #9 - 2018-09-17 11:49 AM - Chris Cannam

Fixed the menu behaviour and merged the changes back to default in commit:64dbd75b9296. Subsequent commits add further tests for the CSV format deduction code. I think we're in pretty good shape.

#### #10 - 2018-09-17 11:50 AM - Chris Cannam

*- Status changed from New to Resolved*

#### #11 - 2018-12-19 09:40 AM - Chris Cannam

*- Status changed from Resolved to Closed*