

Using the EMS JSON-to-CSV conversion tool

Overview

Raw data files created by devices adhering to the Equipment Monitoring System set of specifications (WHO/PQS/E006/DL01, WHO/PQS/E006/EM01, and WHO/PQS/E006/DS01) use Javascript Object Notation (JSON) format. This format is both machine and human readable, which makes it useful for both automated data systems and spot checking the data.

However, EMS devices can use a hierarchical data structure, with administrative information at the top of the file, and an array of timestamped records following. This structure avoids repetition of the administrative data in each record but complicates the use of the data file with common computer tools like spreadsheets. Microsoft Excel can import a JSON array into a spreadsheet; the process is straightforward with a plain array but involves several obscure steps when the data are hierarchical.

This tool is designed to extract the timestamped records from a file or files and create comma-separated values (CSV) files, which can be opened and manipulated with Excel or another spreadsheet application. It was originally developed by New Horizons and is currently maintained by Phoenix Global Health (Phoenix GH, <https://phoenixgh.org/>), a United States-based not-for-profit organization.

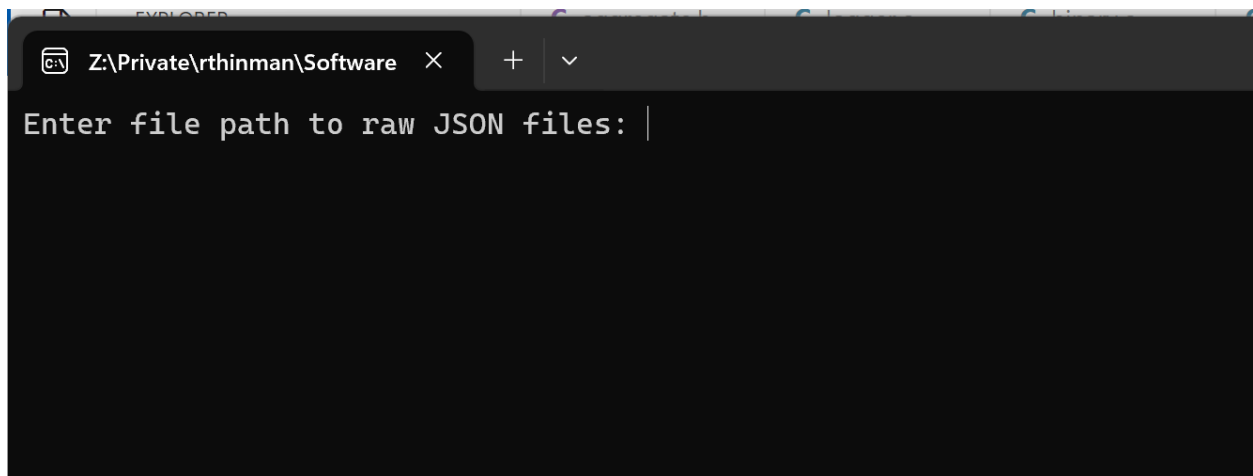
Installation

There is no installation needed, merely uncompress the archive to obtain the **ems-json-tool** folder. You may want to place the folder on your desktop or somewhere else easily accessible. The folder contains: these instructions, the Windows executable application, a folder of sample files, and an **ems-output** folder. The converted CSV files will be placed in the **ems-output** folder.

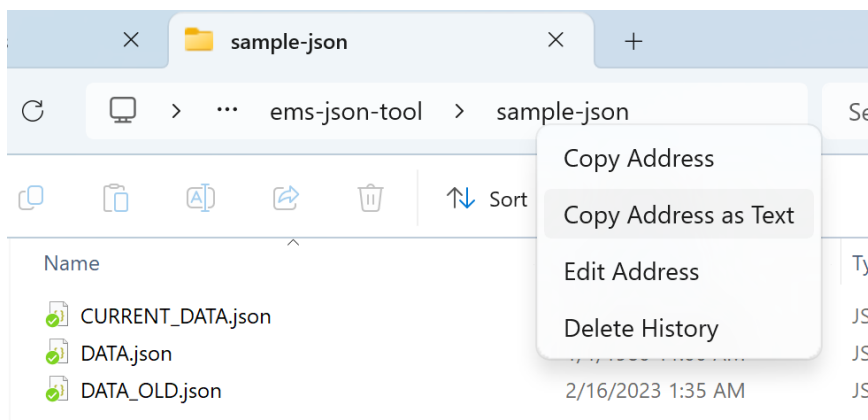
Using the tool

The application can convert a single file or convert all the JSON files in a folder that you specify. To use it:

Start the application by double-clicking it. This will open a console window, and after a few seconds there will be a prompt for a filename or a folder name. See below.



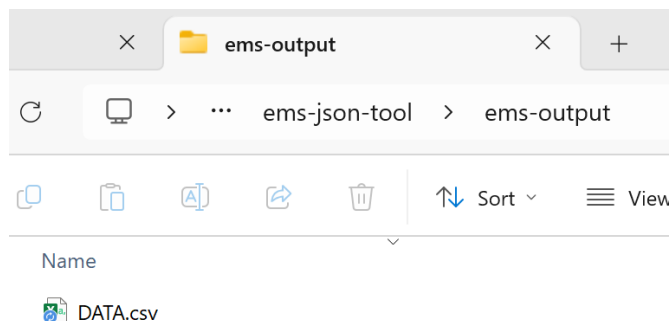
The easiest way to obtain the absolute path to the folder without a lot of typing is to navigate Windows explorer to the folder containing the file or files you wish to convert. Then right click on the folder name in the navigation bar and choose “Copy as Address as Text”. The process is shown below with the example folder included with the application.



Switch to the EMS JSON tool and paste (Ctrl-V, or right click in the window). If you are converting a single file, type the filename at the end of the path, as shown in the example below (“DATA.json”). Then hit <Enter>.



The tool will convert the file or files to CSV, using the same filename with the extension “.json” changed to “.csv”, and will place them in the **ems-output** folder. It will skip files that do not have the extensions “.json” or “.JSON”.



At this time, the tool will abort if it tries to open a file that is not JSON format or has been corrupted. This error can prevent it from converting any or all JSON files within a folder when being used in batch mode.

Each converted file will be a table with a header that contains all the data fields in the “records” portion of the source file. Missing or null fields will be empty. In addition, the final column, called `total_seconds`, will be the timestamp (RELT) converted from ISO 8601 duration format to the number of seconds that duration represents. This column can be used to compute time differences between records, if needed.

Note that data objects outside of the EMS “records” array, typically administrative information, are not converted and must be tracked separately.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	RELT	RTCW	TVC	DORV	TAMB	BLOG	CMPR	CMPS	DCCD	DCSV	SVA	LERR	total_seconds	
2	P15DT0H6I:P14DT21H		4.4	0	25.6	1826						TODO	1296407	
3	P15DT0H2:P14DT21H		4.4	0	25.8	1826						TODO	1297306	
4	P15DT0H3:P14DT21H		5.2	22	25.9	1826						TODO	1298206	
5	P15DT0H5:P14DT21H		5.3	0	25.9	1826						TODO	1299106	
6	P15DT1H6:P14DT21H		5.4	0	25.8	1826						TODO	1300006	
7	P15DT1H2:P14DT21H		5.5	0	25.8	1826						TODO	1300906	
8	P15DT1H3:P14DT21H		5.6	0	25.7	1826						TODO	1301806	

Useful post-tool manipulations in spreadsheets

We have found it useful to perform the following manipulations in Excel after opening the CSV file, and then to save it to Excel format rather than keep it as CSV:

- Add another column after the `total_seconds` column called `delta_seconds`, where each cell represents the interval between adjacent records (calculated by an Excel formula). In an ideal world, all values of `delta_seconds` would be 900 seconds.
- Add conditional formatting to the `delta_seconds` column that highlights cells if they are less than 900 seconds (yellow) or greater than 900 seconds (red). This makes it easier to find those cases.
- Add a column `comment` as a place to take notes.
- Add two rows at the top that contain the minimum and maximum values in their respective columns (and a third empty row that is just for space). This gives a sense of whether there

could be problematic data to investigate. Of course, this step is not necessary for text data columns.

- It can be helpful to change the column order if the time objects (RELT, RTCW, and potentially ABST) are not near the left of the file.
- Use the Excel View>Freeze Panes function to freeze the top rows so that the minimum, maximum, and object names are always visible. One can also freeze the time columns on the left side.

Now the file can be scrolled horizontally to cover all the data objects and vertically to look through time.

The **sample-json** folder has several files from a New Horizons Demo EMS Logger (<https://docs.2to8.cc/ems-hardware/demo-ems-logger/>), which was monitoring the door and vaccine chamber temperature of an Aucma CFD-50 refrigerator. Note that this logger lacked sensors for mains power or compressor runtime, so the installation does not fully conform to DL01.

The file with the specification-compliant filename (002100265547501820383131_CURRENT_DATA_P863DT12H3M51S.json) contains about 36 hours of data from March 2025, including a period when firmware was updated. You will also find a similarly named Excel file, which was produced by converting the JSON file to CSV and applying the Excel changes described above. This Excel file serves as a template – you can copy and paste the `delta_seconds` column into newly generated files to easily calculate `delta_seconds` and apply conditional formatting.

Tips for checking logger operation

The suggestions below were assembled during development and stabilization of the New Horizons Demo EMS Logger. The conditions described indicate possible problems with an EMS logger that should be investigated further.

Checks with CSV/Excel

- Look for changes in the RTCW value. In an ideal world, RTCW would remain 0 (PT0S) for the life of the logger. During product development, it is reasonably common for it to change, but RTCW should rarely change once the product reaches the test lab or into the field.
- Look for cases where `delta_seconds` $\neq$ 900 (or another appropriate interval, such as 600 seconds).

Sometimes one sample will be late or early, so there might be (for example) an 890-second interval for one record and then a 910-second interval for the next record. The 910-second interval is not strictly compliant with DS01 but may be a transient phenomenon; it should be investigated for possible software problems. Long gaps (for example, 3600 seconds) should be thoroughly investigated.

- A case of two consecutive records with the same RELT value is non-compliant, as RELT must be strictly increasing. These cases can easily be found because `delta_seconds` will be 0.

- Sometimes a record will come early – and this is not a problem. Some possible causes are interim records when an alarm condition occurs (for instance, in a combined data logger + EMD), or during firmware updates.
- Look for compressor runtime (CMPR) $\neq 0$ when power availability (SVA) = 0 for an AC fridge. It would be very surprising for the compressor to run when there is no AC power.
- Look for vaccine chamber temperature (TVC) dropping when the compressor is not running (i.e., CMPR = 0). Depending on fridge architecture there might be reasonable explanations, but this situation is worth a closer inspection.
- Look for rapid changes in TVC. These could indicate a sensor problem, especially if the door has not been opened.
- Compare the values in the PDF summary document with the more detailed raw data derived from the JSON. Do they correlate?

Direct checks of the JSON

Because the JSON-to-CSV converter and Excel's internal CSV reader mask certain errors, it is also helpful to look at the raw JSON directly for these kinds of issues:

- Dates (e.g., production dates) formatted without leading zeroes, e.g. 2025-7-5 for July 5th, 2025. Excel tolerates this format and converts it to a date, but strict compliance with ISO 8601 requires leading zeroes.
- Duplicate data fields (e.g. "LMFR":"Acme","LMFR":"Acme"). The Python library's JSON->CSV conversion process will ignore one of the fields. And while many consumers (e.g., cloud data systems) would likely handle this duplication gracefully, it is better not to include duplicates.
- As noted above, data outside the “records” array is removed in the conversion process, so you should review these fields in the JSON directly.

Suggestions for improvement

Please send suggestions for future features to Phoenix GH.

Document Revision History

Date	Person	Changes
2024-11-21	Rod Hinman	Created
2024-12-02	Rod Hinman	Update for v2 of the application
2026-02-05	Rod Hinman	Add suggestions for further manipulation, data checks, and the change to Phoenix GH.