

# Package ‘csvy’

August 1, 2018

**Type** Package

**Title** Import and Export CSV Data with a YAML Metadata Header

**Version** 0.3.0

**Date** 2018-07-31

**Description**

Support for import from and export to the CSVY file format. CSVY is a file format that combines the simplicity of CSV (comma-separated values) with the metadata of other plain text and binary formats (JSON, XML, Stata, etc.) by placing a YAML header on top of a regular CSV.

**URL** <https://github.com/leeper/csvy>

**BugReports** <https://github.com/leeper/csvy/issues>

**Imports** tools, data.table, jsonlite, yaml

**Suggests** testthat, datasets

**License** GPL-2

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Thomas J. Leeper [aut, cre] (<<https://orcid.org/0000-0003-4097-6326>>),  
Alexey N. Shiklomanov [aut] (<<https://orcid.org/0000-0003-4022-5979>>),  
Jonathan Carroll [aut] (<<https://orcid.org/0000-0002-1404-5264>>)

**Maintainer** Thomas J. Leeper <[thosjleeper@gmail.com](mailto:thosjleeper@gmail.com)>

**Repository** CRAN

**Date/Publication** 2018-08-01 04:40:02 UTC

## R topics documented:

colclass_dict . . . . .	2
csvy . . . . .	2
get_yaml_header . . . . .	2
read_csvy . . . . .	3
read_metadata . . . . .	3
write_csvy . . . . .	4
write_metadata . . . . .	5

**Index****6**


---

colclass_dict	<i>Dictionary of column classes for reading data</i>
---------------	--

---

**Description**

Dictionary of column classes for reading data

**Usage**

```
colclass_dict
```

**Format**

An object of class character of length 7.

---

csvy	<i>Import and Export CSV Data With a YAML Metadata Header</i>
------	---

---

**Description**

CSVY is a file format that combines the simplicity of CSV (comma-separated values) with the metadata of other plain text and binary formats (JSON, XML, Stata, etc.). The **CSVY file specification** is simple: place a YAML header on top of a regular CSV. The csvy package implements this format using two functions: [write\\_csvy](#) and [read\\_csvy](#).

---

get_yaml_header	<i>Retrieve YAML header from file</i>
-----------------	---------------------------------------

---

**Description**

Note that this assumes only one Yaml header, starting on the first line of the file.

**Usage**

```
get_yaml_header(file, yaml_rxp = "^\\#*---[[:space:]]*$", verbose = TRUE)
```

**Arguments**

file	A character string or R connection specifying a file.
yaml_rxp	Regular expression for parsing YAML header
verbose	Logical. If TRUE, print warning if no header found.

**Value**

Character vector of lines containing YAML header, or 'NULL' if no YAML header found.

---

read_csvy	<i>Import CSVY data</i>
-----------	-------------------------

---

**Description**

Import CSVY data as a data.frame

**Usage**

```
read_csvy(file, metadata = NULL, stringsAsFactors = FALSE,  
          detect_metadata = TRUE, ...)
```

**Arguments**

file	A character string or R connection specifying a file.
metadata	Optionally, a character string specifying a YAML (“.yaml”) or JSON (“.json”) file containing metadata (in lieu of including it in the header of the file).
stringsAsFactors	A logical specifying whether to treat character columns as factors. Passed to <a href="#">read.csv</a> or <a href="#">fread</a> depending on the value of method. Ignored for method = 'readr' which never returns factors.
detect_metadata	A logical specifying whether to auto-detect a metadata file if none is specified (and if no header is found).
...	Additional arguments passed to <a href="#">fread</a> .

**See Also**

[write\\_csvy](#)

**Examples**

```
read_csvy(system.file("examples", "example1.csvy", package = "csvy"))
```

---

read_metadata	<i>Read metadata</i>
---------------	----------------------

---

**Description**

Read csvy metadata from an external .yaml/.yml or .json file

**Usage**

```
read_metadata(file)
```

**Arguments**

file                    full path of file from which to read the metadata.

**Value**

the metadata as a list

---

write\_csvy                    *Export CSVY data*

---

**Description**

Export data.frame to CSVY

**Usage**

```
write_csvy(x, file, metadata = NULL, sep = ",", dec = ".",
  comment_header = if (is.null(metadata)) TRUE else FALSE,
  name = deparse(substitute(x)), metadata_only = FALSE, ...)
```

**Arguments**

x                    A data.frame.

file                A character string or R connection specifying a file.

metadata            Optionally, a character string specifying a YAML (“.yaml”) or JSON (“.json”) file to write the metadata (in lieu of including it in the header of the file).

sep                A character string specifying a between-field separator. Passed to [fwrite](#).

dec                A character string specifying a within-field separator. Passed to [fwrite](#).

comment\_header    A logical indicating whether to comment the lines containing the YAML front matter. Default is TRUE.

name                A character string specifying a name for the dataset.

metadata\_only    A logical indicating whether only the metadata should be produced (no CSV component).

...                Additional arguments passed to [fwrite](#).

**See Also**

[write\\_csvy](#)

**Examples**

```
library("datasets")
write_csvy(head(iris))

# write yaml w/o comment charaters
write_csvy(head(iris), comment_header = FALSE)
```

---

write_metadata	<i>Write csvy metadata</i>
----------------	----------------------------

---

**Description**

Write csvy metadata to an external .yaml/.yml or .json file

**Usage**

```
write_metadata(metadata_list = NULL, file = NULL)
```

**Arguments**

metadata_list	metadata to be stored. Must be valid as per <code>yaml::as.yaml()</code> or <code>jsonlite::write_json()</code> for that particular output type.
file	full path of file in which to save the metadata.

**Value**

NULL (invisibly)

# Index

## \*Topic **datasets**

colclass\_dict, 2

colclass\_dict, 2

csvy, 2

csvy-package (csvy), 2

fread, 3

fwrite, 4

get\_yaml\_header, 2

jsonlite::write\_json(), 5

read.csv, 3

read\_csvy, 2, 3

read\_metadata, 3

write\_csvy, 2-4, 4

write\_metadata, 5

yaml::as.yaml(), 5