

Package ‘ndjson’

May 9, 2026

Type Package

Title Wicked-Fast Streaming 'JSON' ('ndjson') Reader

Version 0.9.1

Date 2025-03-06

Maintainer Bob Rudis <bob@rud.is>

Description Streaming 'JSON' ('ndjson') has one 'JSON' record per-line and many modern 'ndjson' files contain large numbers of records. These constructs may not be columnar in nature, but it is often useful to read in these files and ``flatten'' the structure out to enable working with the data in an R 'data.frame'-like context. Functions are provided that make it possible to read in plain 'ndjson' files or compressed ('gz') 'ndjson' files and either validate the format of the records or create ``flat'' 'data.table' structures from them.

URL <https://github.com/hrbrmstr/ndjson>

BugReports <https://github.com/hrbrmstr/ndjson/issues>

SystemRequirements zlib, C++17

NeedsCompilation yes

License MIT + file LICENSE

Encoding UTF-8

Suggests tinytest, covr

Depends R (>= 3.2.0)

Imports Rcpp, data.table, tibble

LinkingTo Rcpp

RoxygenNote 7.3.2

Author Bob Rudis [aut, cre] (ORCID: <<https://orcid.org/0000-0001-5670-2640>>),
Niels Lohmann [aut] (C++ json parser),
Deepak Bandyopadhyay [aut] (C++ gzstream),
Lutz Kettner [aut] (C++ gzstream),
Neal Fultz [ctb] (Rcpp integration),
Maarten Demeyer [ctb] (dtplyr cleanup)

Repository CRAN

Date/Publication 2025-03-06 22:20:02 UTC

Contents

flatten	2
ndjson	3
stream_in	3
validate	4
Index	5

flatten	<i>Flatten a character vector of individual JSON lines into a data.table</i>
---------	--

Description

Flatten a character vector of individual JSON lines into a data.table

Usage

```
flatten(x, cls = c("dt", "tbl"))
```

Arguments

x	character vector of individual JSON lines to flatten
cls	the package uses data.table::rbindlist for speed but that's not always the best return type for everyone, so you have option of keeping it a data.table or converting it to a tbl

Value

data.table or tbl

Examples

```
flatten('{"top":{"next":{"final":1,"end":true},"another":"yes"},"more":"no"}')
```

`ndjson`*Wicked-fast Streaming JSON ('ndjson) Reader*

Description

Streaming 'JSON' ('ndjson') has one 'JSON' record per-line and many modern 'ndjson' files contain large numbers of records. These constructs may not be columnar in nature, but it is often useful to read in these files and "flatten" the structure out to enable working with the data in an R 'data.frame'-like context. Functions are provided that make it possible to read in plain ndjson' files or compressed ('gz') 'ndjson' files and either validate the format of the records or create "flat" 'data.table' structures from them.

Author(s)

Bob Rudis (bob@rud.is)

See Also

Useful links:

- <https://github.com/hrbrmstr/ndjson>
- Report bugs at <https://github.com/hrbrmstr/ndjson/issues>

`stream_in`*Stream in & flatten an ndjson file into a data.table*

Description

Given a file of streaming JSON (ndjson) this function reads in the records and creates a flat `data.table` / `tbl` from it.

Usage

```
stream_in(path, cls = c("dt", "tbl"))
```

Arguments

<code>path</code>	path to file (supports "gz" files)
<code>cls</code>	the package uses <code>data.table::rbindlist</code> for speed but that's not always the best return type for everyone, so you have option of keeping it a <code>data.table</code> or converting it to a <code>tbl</code>

Value

`data.table` or `tbl`

References

<https://ndjson.org/>

Examples

```
f <- system.file("extdata", "test.json", package="ndjson")
nrow(stream_in(f))
```

```
gzf <- system.file("extdata", "testgz.json.gz", package="ndjson")
nrow(stream_in(gzf))
```

validate

Validate ndjson file

Description

Given a file of streaming JSON (ndjson) this function reads in the records and validates that they are all legal JSON records. If the verbose parameter is TRUE and errors are found, the line numbers of the errant records will be displayed.

Usage

```
validate(path, verbose = FALSE)
```

Arguments

path	path to file (supports "gz" files)
verbose	display verbose information (filename and line numbers with bad records)

Value

logical

References

<https://ndjson.org/>

Examples

```
f <- system.file("extdata", "test.json", package="ndjson")
validate(f)
```

```
gzf <- system.file("extdata", "testgz.json.gz", package="ndjson")
validate(gzf)
```

Index

`flatten`, [2](#)

`ndjson`, [3](#)

`ndjson-package (ndjson)`, [3](#)

`stream_in`, [3](#)

`validate`, [4](#)