

# Package ‘geojsonlint’

October 13, 2022

**Title** Tools for Validating 'GeoJSON'

**Description** Tools for linting 'GeoJSON'. Includes tools for interacting with the online tool <<http://geojsonlint.com>>, the 'Javascript' library 'geojsonhint' (<<https://www.npmjs.com/package/geojsonhint>>), and validating against a 'GeoJSON' schema via the 'Javascript' library (<<https://www.npmjs.com/package/is-my-json-valid>>). Some tools work locally while others require an internet connection.

**Version** 0.4.0

**License** MIT + file LICENSE

**URL** <https://github.com/ropensci/geojsonlint> (devel)  
<https://docs.ropensci.org/geojsonlint> (docs)

**BugReports** <https://github.com/ropensci/geojsonlint/issues>

**LazyData** true

**VignetteBuilder** knitr

**Encoding** UTF-8

**Depends** R (>= 2.10)

**Imports** crul (>= 0.7.0), jsonlite (>= 0.9.19), jsonvalidate (>= 1.0.0), V8

**Suggests** testthat, knitr, rmarkdown

**RoxygenNote** 7.0.2

**X-schema.org-applicationCategory** Geospatial

**X-schema.org-keywords** geojson, geospatial, lint, hint, data, schema

**X-schema.org-isPartOf** <https://ropensci.org>

**NeedsCompilation** no

**Author** Scott Chamberlain [aut, cre] (<<https://orcid.org/0000-0003-1444-9135>>),  
Andy Teucher [aut],  
Tom MacWright [cph] (geojsonhint Javascript library),  
Mads Kristensen [cph] (geojson schema)

**Maintainer** Scott Chamberlain <[myrmecocystus@gmail.com](mailto:myrmecocystus@gmail.com)>

**Repository** CRAN

**Date/Publication** 2020-02-13 06:20:11 UTC

## R topics documented:

geojsonlint-package . . . . .	2
as.location . . . . .	2
canada_cities . . . . .	3
geojson_hint . . . . .	3
geojson_lint . . . . .	4
geojson_validate . . . . .	5
gj_write . . . . .	6
states . . . . .	7
us_cities . . . . .	7
<b>Index</b>	<b>8</b>

---

geojsonlint-package      **GeoJSON Linting**

---

### Description

#### GeoJSON Linting

### Package API

- [geojson\\_hint\(\)](#) - Checks validity of geojson using the Javascript library geojsonhint
- [geojson\\_validate\(\)](#) - Checks validity of geojson using a GeoJSON schema and the Javascript library is-my-json-valid

### Author(s)

Scott Chamberlain <myrmecocystus@gmail.com>

Andy Teucher <andy.teucher@gmail.com>

---

as.location                      *Convert a path or URL to a location object.*

---

### Description

Convert a path or URL to a location object.

### Usage

```
as.location(x, ...)
```

### Arguments

x	Input.
...	Ignored.

**Examples**

```
## Not run:
# A file
file <- system.file("examples", "zillow_or.geojson",
  package = "geojsonlint")
as.location(file)

# A URL
url <- paste0("https://raw.githubusercontent.com/glynnbird/",
  "usstatesgeojson/master/california.geojson")
as.location(url)

## End(Not run)
```

---

canada\_cities

*This is the same data set from the maps library, named differently*


---

**Description**

This database is of Canadian cities of population greater than about 1,000. Also included are province capitals of any population size.

**Format**

A list with 6 components, namely "name", "country.etc", "pop", "lat", "long", and "capital", containing the city name, the province abbreviation, approximate population (as at January 2006), latitude, longitude and capital status indication (0 for non-capital, 1 for capital, 2 for provincial

---

geojson\_hint

*Validate GeoJSON using geojsonhint Javascript library*


---

**Description**

Validate GeoJSON using geojsonhint Javascript library

**Usage**

```
geojson_hint(x, inform = FALSE, error = FALSE)
```

**Arguments**

x	Input, a geojson character string, json object, or file or url pointing to one of the former
inform	(logical) When geojson is invalid, return reason why (TRUE) or don't return reason (FALSE). Default: FALSE
error	(logical) Throw an error on parse failure? If TRUE, then function returns TRUE on success, and stop with the error message on error. Default: FALSE

**Details**

Uses the Javascript library <https://www.npmjs.com/package/geojsonhint> via the **V8** package

**Value**

TRUE or FALSE. If `inform=TRUE` an attribute of name `errors` is added with error information

**Examples**

```
geojson_hint('{ "type": "FooBar" }')
geojson_hint('{ "type": "FeatureCollection" }')
geojson_hint(
  '{"type":"Point","geometry":{"type":"Point","coordinates":[-80,40]},"properties":{}}'
)

# A file
file <- system.file("examples", "zillow_or.geojson", package = "geojsonlint")
geojson_hint(as.location(file))

# A URL
if (interactive()) {
  url <- "https://raw.githubusercontent.com/glynnbird/usstatesgeojson/master/california.geojson"
  geojson_hint(as.location(url))
}

# from json (jsonlite class)
x <- jsonlite::minify('{ "type": "FeatureCollection" }')
class(x)
geojson_hint(x)

# toggle whether reason for validation failure is given back
geojson_hint('{ "type": "FeatureCollection" }')
geojson_hint('{ "type": "FeatureCollection" }', inform = TRUE)

# toggle whether to stop with error message
geojson_hint('{ "type": "FeatureCollection" }')
geojson_hint('{ "type": "FeatureCollection" }', inform = TRUE)
if (interactive()) {
  geojson_hint('{ "type": "FeatureCollection" }', error = TRUE)
}
```

---

geojson\_lint

*Validate GeoJSON using geojsonlint.com web service*

---

**Description**

Validate GeoJSON using geojsonlint.com web service

**Usage**

```
geojson_lint(...)
```

**Arguments**

```
...          ignored
```

---

geojson_validate	<i>Validate GeoJSON using is-my-json-valid Javascript library</i>
------------------	---

---

**Description**

Validate GeoJSON using is-my-json-valid Javascript library

**Usage**

```
geojson_validate(x, inform = FALSE, error = FALSE, greedy = FALSE)
```

**Arguments**

x	Input, a geojson character string, json object, or file or url pointing to one of the former
inform	(logical) When geojson is invalid, return reason why (TRUE) or don't return reason FALSE). Default: FALSE
error	(logical) Throw an error on parse failure? If TRUE, then function returns NULL on success, and stop with the error message on error. Default: FALSE
greedy	(logical) Continue after the first error? TRUE or FALSE. Default: FALSE

**Details**

Sometimes you may get a response that your input GeoJSON is invalid, but get a somewhat unhelpful error message, e.g., no (or more than one) schemas match. See <https://github.com/ropensci/geojsonlint/issues/7#issuecomment-219881961>. We'll hopefully soon get this sorted out so you'll get a meaningful error message. However, this method is faster than the other two methods in this package, so there is that.

**Value**

TRUE or FALSE. If inform=TRUE an attribute of name errors is added with error information

**References**

<https://www.npmjs.com/package/is-my-json-valid>

**Examples**

```

# From a json character string
## good
geojson_validate('{ "type": "Point", "coordinates": [-100, 80]}')
## bad
geojson_validate(
  '{ "type": "Rhombus", "coordinates": [[1, 2], [3, 4], [5, 6]]}')

# A file
file <- system.file("examples", "zillow_or.geojson",
  package = "geojsonlint")
geojson_validate(x = as.location(file))

# A URL
if (interactive()) {
url <- "https://raw.githubusercontent.com/glynnbird/usstatesgeojson/master/california.geojson"
geojson_validate(as.location(url))
}

# toggle whether reason for validation failure is given back
geojson_validate('{ "type": "FeatureCollection" }')
geojson_validate('{ "type": "FeatureCollection" }', inform = TRUE)

# toggle whether to stop with error message
geojson_validate('{ "type": "FeatureCollection" }')
geojson_validate('{ "type": "FeatureCollection" }', inform = TRUE)
if (interactive()) {
geojson_validate('{ "type": "FeatureCollection" }', error = TRUE)
}

```

---

 gj\_write

*Write inputs to a geojson file*


---

**Description**

Write inputs to a geojson file

**Usage**

```
gj_write(x, file, ...)
```

**Arguments**

x	input character, json, or geojson
file	file path to write to
...	Further args

**Examples**

```
gj_write(x = '{"type": "Point", "coordinates": [-100, 80]}',  
        (file <- tempfile()))  
jsonlite::fromJSON(file)
```

---

states

*This is the same data set from the ggplot2 library*

---

**Description**

This is the same data set from the ggplot2 library

**Format**

A data.frame with 6 components, including "long", "lat", "group", "order", "region", and "subregion" columns specifying polygons for each US state.

---

us\_cities

*This is the same data set from the maps library, named differently*

---

**Description**

This database is of us cities of population greater than about 40,000. Also included are state capitals of any population size.

**Format**

A list with 6 components, namely "name", "country.etc", "pop", "lat", "long", and "capital", containing the city name, the state abbreviation, approximate population (as at January 2006), latitude, longitude and capital status indication (0 for non-capital, 1 for capital, 2 for state capital).

# Index

## \* data

canada\_cities, 3

states, 7

us\_cities, 7

as.location, 2

canada\_cities, 3

geojson\_hint, 3

geojson\_hint(), 2

geojson\_lint, 4

geojson\_validate, 5

geojson\_validate(), 2

geojsonlint (geojsonlint-package), 2

geojsonlint-package, 2

gj\_write, 6

states, 7

us\_cities, 7