

Package ‘geojsonsf’

July 22, 2025

Type Package

Title GeoJSON to Simple Feature Converter

Version 2.0.3

Date 2022-05-31

Description Converts Between GeoJSON and simple feature objects.

License MIT + file LICENSE

URL <https://github.com/SymbolixAU/geojsonsf>

BugReports <https://github.com/SymbolixAU/geojsonsf/issues>

Encoding UTF-8

LazyData true

Depends R (>= 3.3.0)

SystemRequirements C++11

LinkingTo geometries, jsonify (>= 1.1.1), rapidjsonr (>= 1.2.0), Rcpp,
sfheaders (>= 0.2.2)

Imports Rcpp

RoxygenNote 7.1.0

Suggests covr, jsonify, knitr, rmarkdown, tinytest

VignetteBuilder knitr

NeedsCompilation yes

Author David Cooley [aut, cre]

Maintainer David Cooley <dcooley@symbolix.com.au>

Repository CRAN

Date/Publication 2022-05-30 23:20:02 UTC

Contents

df_geojson	2
geojson_sf	3
geojson_sfc	5
geojson_wkt	6
geo_melbourne	7
sfc_geojson	7
sf_geojson	8
Index	10

df_geojson	<i>df to GeoJSON</i>
------------	----------------------

Description

Converts data.frame objects to GeoJSON. Each row is considered a POINT

Usage

```
df_geojson(  
  df,  
  lon,  
  lat,  
  z = NULL,  
  m = NULL,  
  atomise = FALSE,  
  simplify = TRUE,  
  digits = NULL,  
  factors_as_string = TRUE  
)
```

Arguments

df	data.frame
lon	column of df containing the longitude data
lat	column of df containing the latitude data
z	column of df containing the Z attribute of the GeoJSON
m	column of df containing the M attribute of the GeoJSON. If supplied, you must also supply z
atomise	logical indicating if the data.frame should be converted into a vector of GeoJSON objects
simplify	logical indicating if data.frame without property columns should simplify (TRUE) into a vector of GeoJSON, or (FALSE). If atomise is TRUE this argument is ignored.

digits integer specifying the number of decimal places to round numerics. numeric values are coerced using `as.integer`, which may round-down the value you supply. Default is NULL - no rounding

factors_as_string logical indicating if factors should be treated as strings. Defaults to TRUE.

Value

vector of GeoJSON

Examples

```
df <- data.frame(lon = c(1:5, NA), lat = c(1:5, NA), id = 1:6, val = letters[1:6])
df_geojson( df, lon = "lon", lat = "lat")
df_geojson( df, lon = "lon", lat = "lat", atomise = TRUE)

df <- data.frame(lon = c(1:5, NA), lat = c(1:5, NA) )
df_geojson( df, lon = "lon", lat = "lat")
df_geojson( df, lon = "lon", lat = "lat", simplify = FALSE)

df <- data.frame(lon = c(1:5), lat = c(1:5), elevation = c(1:5) )
df_geojson( df, lon = "lon", lat = "lat", z = "elevation")
df_geojson( df, lon = "lon", lat = "lat", z = "elevation", simplify = FALSE)

df <- data.frame(lon = c(1:5), lat = c(1:5), elevation = c(1:5), id = 1:5 )
df_geojson( df, lon = "lon", lat = "lat", z = "elevation")
df_geojson( df, lon = "lon", lat = "lat", z = "elevation", atomise = TRUE)

## to sf objects
geo <- df_geojson( df, lon = "lon", lat = "lat", z = "elevation")
sf <- geojson_sf( geo )
```

geojson_sf

Geojson to sf

Description

Converts GeoJSON to an 'sf' object

Usage

```
geojson_sf(
  geojson,
  expand_geometries = FALSE,
  input = NULL,
  wkt = NULL,
  crs = NULL,
```

```

    proj4string = NULL,
    buffer_size = 1024
  )

```

Arguments

geojson	string or vector of GeoJSON, or a URL or file pointing to a geojson file
expand_geometries	logical indicating whether to unnest GEOMETRYCOLLECTION rows. see details
input	user input for coordinate reference system object
wkt	well-known text for coordinate reference system object
crs	deprecated. coordinate reference system. See Details
proj4string	deprecated. proj4string. See Details
buffer_size	size of buffer used when reading a file from disk. Defaults 1024

Details

specifying `expand_geometries = TRUE` will expand individual GEOMETRYCOLLECTION geometries to their own row in the resulting 'sf' object. If the geometries are part of a Feature (i.e., with properties), the properties will be repeated on each row.

The GEOMETRYCOLLECTION information is not kept when using `expand_geometries = TRUE`. Therefore, it is not possible to reconstruct the GEOMETRYCOLLECTION after unnesting it.

Geojson specification RFC7946 <https://tools.ietf.org/html/rfc7946#page-12> says all CRS should be the World Geodetic System 1984 (WGS 84) [WGS84] datum, with longitude and latitude units of decimal degrees. This is equivalent to the coordinate reference system identified by the Open Geospatial Consortium (OGC) URN `urn:ogc:def:crs:OGC::CRS84`

`geojson_sfc` and `geojson_sf` automatically set the CRS to WGS 84. The fields `input` and `wkt` let you to overwrite the defaults.

Examples

```

## character string of GeoJSON

## load 'sf' for print methods
# library(sf)
geojson <- '{ "type" : "Point", "coordinates" : [0, 0] }'
geojson_sf(geojson)

## Not run:
## GeoJSON at a url
myurl <- "http://eric.clst.org/assets/wiki/uploads/Stuff/gz_2010_us_050_00_500k.json"
sf <- geojson_sf(myurl)

## End(Not run)

```

geojson_sfc*Geojson to sfc*

Description

Extracts geometries from GeoJSON and returns an 'sfc' object

Usage

```
geojson_sfc(  
  geojson,  
  expand_geometries = FALSE,  
  input = NULL,  
  wkt = NULL,  
  crs = NULL,  
  proj4string = NULL,  
  buffer_size = 1024  
)
```

Arguments

geojson	string or vector of GeoJSON, or a URL or file pointing to a geojson file
expand_geometries	logical indicating whether to unnest GEOMETRYCOLLECTION rows. see details
input	user input for coordinate reference system object
wkt	well-known text for coordinate reference system object
crs	deprecated. coordinate reference system. See Details
proj4string	deprecated. proj4string. See Details
buffer_size	size of buffer used when reading a file from disk. Defaults 1024

Details

specifying `expand_geometries = TRUE` will expand individual GEOMETRYCOLLECTION geometries to their own row in the resulting 'sf' object. If the geometries are part of a Feature (i.e., with properties), the properties will be repeated on each row.

The GEOMETRYCOLLECTION information is not kept when using `expand_geometries = TRUE`. Therefore, it is not possible to reconstruct the GEOMETRYCOLLECTION after unnesting it.

Geojson specification RFC7946 <https://tools.ietf.org/html/rfc7946#page-12> says all CRS should be the World Geodetic System 1984 (WGS 84) [WGS84] datum, with longitude and latitude units of decimal degrees. This is equivalent to the coordinate reference system identified by the Open Geospatial Consortium (OGC) URN `urn:ogc:def:crs:OGC::CRS84`

`geojson_sfc` and `geojson_sf` automatically set the CRS to WGS 84. The fields `input` and `wkt` let you to overwrite the defaults.

Examples

```
## character string of GeoJSON

## load 'sf' for print methods
# library(sf)
geojson <- '{ "type":"Point","coordinates":[0,0] }'
geojson_sfc(geojson)

geojson <- '[
  { "type":"Point","coordinates":[0,0]},
  { "type":"LineString","coordinates":[[0,0],[1,1]]}
]'
geojson_sfc( geojson )

## Not run:
## GeoJSON at a url
myurl <- "http://eric.clst.org/assets/wiki/uploads/Stuff/gz_2010_us_050_00_500k.json"
sf <- geojson_sfc(myurl)

## End(Not run)
```

geojson_wkt

Geojson to WKT

Description

Converts GeoJSON to Well-Known Text

Usage

```
geojson_wkt(geojson)
```

Arguments

geojson string or vector of GeoJSON, or a URL or file pointing to a geojson file

Value

data.frame with a 'geometry' column of well-known text

Examples

```
geojson <- '{ "type" : "Point", "coordinates" : [0, 0] }'
geojson_wkt(geojson)
```

geo_melbourne	<i>geo_melbourne</i>
---------------	----------------------

Description

GeoJSON data of Melbourne’s Inner suburbs.

Usage

geo_melbourne

Format

An object of class geojson (inherits from json) of length 1.

sfc_geojson	<i>sfc to GeoJSON</i>
-------------	-----------------------

Description

Converts ‘sfc’ objects to GeoJSON

Usage

sfc_geojson(sfc, digits = NULL)

Arguments

- sfc simple feature collection object
- digits integer specifying the number of decimal places to round numeric coordinates. numeric values are coerced using as.integer, which may round-down the value you supply. Default is NULL - no rounding

Value

vector of GeoJSON

Examples

```
## Not run:
library(sf)
sf <- sf::st_sfc(list(sf::st_point(c(0,0)), sf::st_point(c(1,1))))
sfc_geojson(sf)

## End(Not run)
```

sf_geojson

*sf to GeoJSON***Description**

Converts 'sf' objects to GeoJSON

Usage

```
sf_geojson(
  sf,
  atomise = FALSE,
  simplify = TRUE,
  digits = NULL,
  factors_as_string = TRUE
)
```

Arguments

sf	simple feature object
atomise	logical indicating if the sf object should be converted into a vector of GeoJSON objects
simplify	logical indicating if sf objects without property columns should simplify (TRUE) into a vector of GeoJSON, or return a Featurecollection with empty property fields (FALSE). If atomise is TRUE this argument is ignored.
digits	integer specifying the number of decimal places to round numerics. numeric values are coerced using <code>as.integer</code> , which may round-down the value you supply. Default is NULL - no rounding
factors_as_string	logical indicating if factors should be treated as strings. Defaults to TRUE.

Value

vector of GeoJSON

Examples

```
## Not run:
library(sf)
sf <- sf::st_sf(geometry = sf::st_sfc(list(sf::st_point(c(0,0)), sf::st_point(c(1,1)))))
sf$id <- 1:2
sf_geojson(sf)
sf_geojson(sf, atomise = T)

ls <- st_linestring(rbind(c(0,0),c(1,1),c(2,1)))
mls <- st_multilinestring(list(rbind(c(2,2),c(1,3)), rbind(c(0,0),c(1,1),c(2,1))))
sfc <- st_sfc(ls,mls)
```



```
sf <- st_sf(sfc)
sf_geojson( sf )
sf_geojson( sf, simplify = FALSE )
```

```
## End(Not run)
```

Index

* **datasets**

geo_melbourne, [7](#)

df_geojson, [2](#)

geo_melbourne, [7](#)

geojson_sf, [3](#)

geojson_sfc, [5](#)

geojson_wkt, [6](#)

sf_geojson, [8](#)

sfc_geojson, [7](#)