Package ‘randgeo’

October 14, 2022

Title  Generate Random 'WKT' or 'GeoJSON'

Description  Generate random positions (latitude/longitude),
              Well-known text ('WKT') points or polygons, or 'GeoJSON' points or
              polygons.

Version  0.3.0

License  MIT + file LICENSE

LazyData  true

URL  https://github.com/ropensci/randgeo

BugReports  https://github.com/ropensci/randgeo/issues

VignetteBuilder  knitr

Suggests  rmarkdown, knitr, testthat

RoxygenNote  6.0.1

NeedsCompilation  no

Author  Scott Chamberlain [aut, cre] (<https://orcid.org/0000-0003-1444-9135>),
        Noam Ross [aut],
        Samuel Bosch [aut]

Maintainer  Scott Chamberlain <myrmecocystus@gmail.com>

Repository  CRAN

Date/Publication  2018-05-18 23:34:28

R topics documented:

  randgeo-package                          ............................................. 2
  geo_linestring                           ............................................. 2
  geo_point                                ................................................. 3
  geo_polygon                              ................................................. 4
  rg_position                              ................................................. 5
  wkt_linestring                           ................................................. 5
  wkt_point                                ................................................. 6
  wkt_polygon                              ................................................. 7

Index  8
geo_linestring

Description

Random GeoJSON linestring

Usage

geo_linestring(count = 1, num_vertices = 10, max_length = 0.001, max_rotation = pi/8, bbox = NULL)
geo_point

Arguments

- **count**: (integer/numeric) number of Polygons. Default: 1
- **num_vertices**: (integer/numeric) how many coordinates each polygon will contain. Default: 10
- **max_length**: (integer/numeric) maximum distance that a vertex can be from its predecessor. Units are in degrees latitude (Approximately 69 miles or 111 km). Default: 0.001 (approximately 121 yards or 111 meters)
- **max_rotation**: (integer/numeric) the maximum number of radians that a line segment can turn from the previous segment. Default: pi / 8
- **bbox**: (integer/numeric) lat/long bounding box for the starting point of the line, numeric vector of the form west (long), south (lat), east (long), north (lat). optional

Value

GeoJSON; a list with one or more Linestrings in a FeatureCollection, with class geo_list - simple unclass() to remove the class

Examples

```
geo_linestring()
geo_linestring(10)
geo_linestring(bbox = c(50, 50, 60, 60))
```

---

**geo_point**

Random GeoJSON point

Usage

```
geo_point(count = 1, bbox = NULL)
```

Arguments

- **count**: (integer/numeric) number of points. Default: 1
- **bbox**: (integer/numeric) lat/long bounding box from which to generate positions; numeric vector of the form west (long), south (lat), east (long), north (lat). optional

Value

GeoJSON; a list with one or more Points in a FeatureCollection, with class geo_list - simple unclass() to remove the class
geo_polygon

Examples

geo_point()
geo_point(10)
geo_point(bbox = c(50, 50, 60, 60))

geo_polygon

Random GeoJSON polygon

Description

Random GeoJSON polygon

Usage

geo_polygon(count = 1, num_vertices = 10, max_radial_length = 10,
bbox = NULL)

Arguments

count (integer/numeric) number of Polygons. Default: 1
num_vertices (integer/numeric) how many coordinates each polygon will contain. Default: 10
max_radial_length (integer/numeric) maximum distance that a vertex can reach out of the center of the polygon. Units are in degrees latitude (Approximately 69 miles or 111 km). Default: 10
bbox (integer/numeric) lat/long bounding box for the centers of the polygons, numeric vector of the form west (long), south (lat), east (long), north (lat). optional

Value

GeoJSON; a list with one ore more Polygons in a FeatureCollection, with class geo_list - simple unclass() to remove the class

Examples

geo_polygon()
geo_polygon(10)
geo_polygon(bbox = c(50, 50, 60, 60))
**rg_position**

**Random position**

**Description**
Random position

**Usage**

```r
gg_position(count = 1, bbox = NULL)
```

**Arguments**

- `count` (integer/numeric) number of positions. Default: 1
- `bbox` (integer/numeric) lat/long bounding box from which to generate positions; numeric vector of the form west (long), south (lat), east (long), north (lat). optional

**Value**
A list, each element is a numeric vector length two of long, lat

**Examples**

```r
rg_position()
gg_position(10)
gg_position(100)
gg_position(bbox = c(50, 50, 60, 60))
```

# coerce to data.frame
```r
stats::setNames(
  do.call("rbind.data.frame", rg_position(10)),
  c("lng", "lat")
)
```

---

**wkt_linestring**

**Random WKT linestring**

**Description**
Random WKT linestring

**Usage**

```r
wkt_linestring(count = 1, num_vertices = 10, max_length = 1e-04,
  max_rotation = pi/8, bbox = NULL, fmt = 7)
```
Arguments

- **count**: (integer/numeric) number of Polygons. Default: 1
- **num_vertices**: (integer/numeric) how many coordinates each polygon will contain. Default: 10
- **max_length**: (integer/numeric) maximum number of decimal degrees (1 degree = approximately 69 miles or 111 km) that a vertex can be from its predecessor. Default: 0.0001
- **max_rotation**: (integer/numeric) the maximum number of radians that a line segment can turn from the previous segment. Default: \(\pi / 8\)
- **bbox**: (integer/numeric) lat/long bounding box for the starting point of the line, numeric vector of the form west (long), south (lat), east (long), north (lat). optional
- **fmt**: (integer/numeric) number of digits. Default: 7

Value

WKT; a character vector with one or more LINESTRING strings

Examples

```r
wkt_linestring()
wkt_linestring(10)
wkt_linestring(num_vertices = 4)
wkt_linestring(bbox = c(50, 50, 60, 60))
```

---

**wkt_point**

Random WKT point

Usage

```r
wkt_point(count = 1, bbox = NULL, fmt = 7)
```

Arguments

- **count**: (integer/numeric) number of points. Default: 1
- **bbox**: (integer/numeric) lat/long bounding box from which to generate positions; numeric vector of the form west (long), south (lat), east (long), north (lat). optional
- **fmt**: (integer/numeric) number of digits. Default: 7

Value

WKT; a character vector with one or more POINT strings
**wkt_polygon**

**Examples**

- `wkt_point()`
- `wkt_point(10)`
- `wkt_point(100)`
- `wkt_point(fmt = 5)`
- `wkt_point(fmt = 6)`
- `wkt_point(fmt = 7)`
- `wkt_point(bbox = c(50, 50, 60, 60))`

**Description**

Random WKT polygon

**Usage**

```
wkt_polygon(count = 1, num_vertices = 10, max_radial_length = 10,
            bbox = NULL, fmt = 7)
```

**Arguments**

- `count` (integer/numeric) number of Polygons. Default: 1
- `num_vertices` (integer/numeric) how many coordinates each polygon will contain. Default: 10
- `max_radial_length` (integer/numeric) maximum distance that a vertex can reach out of the center of the polygon. Units are in degrees latitude (Approximately 69 miles or 111 km). Default: 10
- `bbox` (integer/numeric) lat/long bounding box for the centers of the polygons, numeric vector of the form west (long), south (lat), east (long), north (lat). optional
- `fmt` (integer/numeric) number of digits. Default: 7

**Value**

WKT; a character vector with one or more POLYGON strings

**Examples**

- `wkt_polygon()`
- `wkt_polygon(num_vertices = 3)`
- `wkt_polygon(num_vertices = 4)`
- `wkt_polygon(num_vertices = 100)`
- `wkt_polygon(10)`
- `wkt_polygon(bbox = c(50, 50, 60, 60))`
Index

* package
  randgeo-package, 2

geo_linestring, 2
geo_point, 3
geo_point(), 2
geo_polygon, 4
geo_polygon(), 2

randgeo (randgeo-package), 2
randgeo-package, 2
rg_position, 5
rg_position(), 2

wkt_linestring, 5
wkt_point, 6
wkt_point(), 2
wkt_polygon, 7
wkt_polygon(), 2