

Package ‘geohashTools’

October 3, 2019

Version 0.3.0

Title Tools for Working with Geohashes

Maintainer Michael Chirico <MichaelChirico4@gmail.com>

Depends R (>= 3.0.0)

Description Tools for working with Gustavo Niemeyer's geohash coordinate system, including API for interacting with other common R GIS libraries.

URL <https://github.com/MichaelChirico/geohashTools>

License MPL-2.0 | file LICENSE

Suggests rgdal, sf, sp, testthat, mockery

NeedsCompilation yes

Author Michael Chirico [aut, cre],
Dmitry Shkolnik [ctb]

Repository CRAN

Date/Publication 2019-10-03 12:20:02 UTC

R topics documented:

gh_decode	2
gh_encode	3
gh_neighbors	4
gis_tools	5
utils	6
Index	8

 gh_decode

Geohash decoding

Description

Convert geohash-encoded strings into latitude/longitude coordinates

Usage

```
gh_decode(geohashes, include_delta = FALSE, coord_loc = 'c')
```

Arguments

geohashes	character vector of input geohashes. There's no need for all inputs to be of the same precision.
include_delta	logical; should the cell half-width delta be included in the output?
coord_loc	character specifying where in the cell points should be mapped to; cell centroid is mapped by default; case-insensitive. See Details.

Details

coord_loc can be the cell's center ('c' or 'centroid'), or it can be any of the 8 corners (e.g. 's'/'south' for the midpoint of the southern boundary of the cell, or 'ne'/'northeast' for the upper-right corner).

Value

list with the following entries:

latitude	numeric vector of latitudes (y-coordinates) corresponding to the input geohashes, with within-cell position dictated by coord_loc
longitude	numeric vector of longitudes (x-coordinates) corresponding to the input geohashes, with within-cell position dictated by coord_loc
delta_latitude	numeric vector of cell half-widths in the y direction (only included if include_delta is TRUE)
delta_longitude	numeric vector of cell half-widths in the x direction (only included if include_delta is TRUE)

Author(s)

Michael Chirico

References

<http://geohash.org/> (Gustavo Niemeyer's original geohash service)

Examples

```
# Riddle me this
gh_decode('stq4s8c')

# Cell half-widths might be convenient to include for downstream analysis
gh_decode('tjmd79', include_delta = TRUE)
```

gh_encode

Geohash encoding

Description

Convert latitude/longitude coordinates into geohash-encoded strings

Usage

```
gh_encode(latitude, longitude, precision = 6L)
```

Arguments

latitude	numeric vector of input latitude (y) coordinates. Must be in $[-90, 90)$.
longitude	numeric vector of input longitude (x) coordinates. Should be in $[-180, 180)$.
precision	Positive integer scalar controlling the 'zoom level' – how many characters should be used in the output.

Details

precision is limited to at most 28. This level of precision encodes locations on the globe at a nanometer scale and is already more than enough for basically all applications.

Longitudes outside $[-180, 180)$ will be wrapped appropriately to the standard longitude grid.

Value

character vector of geohashes corresponding to the input. NA in gives NA out.

Author(s)

Michael Chirico

References

<http://geohash.org/> (Gustavo Niemeyer's original geohash service)

Examples

```
# scalar input is treated as a vector
gh_encode(2.345, 6.789)

# geohashes are left-closed, right-open, so boundary coordinates are
# associated to the east and/or north
gh_encode(0, 0)
```

gh_neighbors

Geohash neighborhoods

Description

Return the geohashes adjacent to input geohashes

Usage

```
gh_neighbors(geohashes, self = TRUE)
gh_neighbours(geohashes, self = TRUE)
```

Arguments

geohashes	character vector of input geohashes. There's no need for all inputs to be of the same precision.
self	Should the input also be returned as a list element? Convenient for one-line usage / piping

Details

North/south-pole adjacent geohashes are missing three of their neighbors; these will be returned as NA_character_.

Value

list with character vector entries in the direction relative to the input geohashes indicated by their name (e.g. value\$south gives all of the *southern* neighbors of the input geohashes).

The order is self (if self = TRUE), southwest, south, southeast, west, east, northwest, north, northeast (reflecting an easterly, then northerly traversal of the neighborhood).

Author(s)

Michael Chirico

References

<http://geohash.org/> (Gustavo Niemeyer's original geohash service)

Examples

```
gh_neighbors('d7q8u4')
```

```
gis_tools
```

Helpers for interfacing geohashes with sp/sf objects

Description

These functions smooth the gateway between working with geohashes and geospatial information built for the major geospatial packages in R, [sp](#) and [sf](#).

Usage

```
gh_to_sp(geohashes)
gh_to_spdf(...)
gh_to_sf(...)

gh_covering(SP, precision = 6L, minimal = FALSE)

## Default S3 method:
gh_to_spdf(geohashes, ...)

## S3 method for class 'data.frame'
gh_to_spdf(gh_df, gh_col = 'gh', ...)
```

Arguments

geohashes	character vector of geohashes to be converted to polygons.
...	Arguments for subsequent methods.
SP	A Spatial object (requires <code>bbox</code> and <code>proj4string</code> methods, and <code>over</code> if <code>minimal</code> is <code>TRUE</code>)
precision	integer specifying the precision of geohashes to use, same as gh_encode
minimal	logical; if <code>FALSE</code> , the output will have all geohashes in the bounding box of <code>SP</code> ; if <code>TRUE</code> , any geohashes not intersecting <code>SP</code> will be removed.
gh_df	<code>data.frame</code> which 1) contains a column of geohashes to be converted to polygons and 2) will serve as the data slot of the resultant SpatialPolygonsDataFrame object.
gh_col	character column name saying where the geohashes are stored in <code>gh_df</code> .

Details

`gh_to_sp` relies on the [gh_decode](#) function. Note in particular that this function accepts any length of geohash (geohash-6, geohash-4, etc.) and is agnostic to potential overlap, though duplicates will be caught and excluded.

`gh_to_spdf.data.frame` will use `match.ID = FALSE` in the call to `SpatialPolygonsDataFrame`. Please file an issue if you'd like this to be more flexible.

`gh_to_sf` is just a wrapper of `st_as_sf` around `gh_to_spdf`; as such it requires both `sp` and `sf` packages to work.

Value

For `gh_to_sp`, a `SpatialPolygons` object.

For `gh_to_spdf`, a `SpatialPolygonsDataFrame` object.

For `gh_to_sf`, a `sf` object.

Examples

```
# get the neighborhood of this geohash in downtown Apia as an sp object
downtown = '2jtc5x'
apia_nbhd = unlist(gh_neighbors(downtown))
apia_sp = gh_to_sp(apia_nbhd)

# all geohashes covering a random sampling within Apia:
apia_covering = gh_covering(smp <- sp::spsample(apia_sp, 10L, 'random'))

apia_sf = gh_to_sf(apia_nbhd)
```

utils

Geohash utilities

Description

Various common functions that arise when working often with geohashes

Usage

```
gh_delta(precision)
```

Arguments

`precision` integer precision level desired.

Value

Length-2 numeric vector; the first element is the *latitude* (y-coordinate) half-width at the input precision, the second element is the *longitude* (x-coordinate).

Note

Caveat coder: not much is done in the way of consistency checking since this is a convenience function. So e.g. real-valued "precision"s will give results.

Author(s)

Michael Chirico

References

<http://geohash.org/> (Gustavo Niemeyer's original geohash service)

Examples

gh_delta(6)

Index

`gh_covering` (`gis_tools`), 5
`gh_decode`, 2, 5
`gh_delta` (`utils`), 6
`gh_encode`, 3, 5
`gh_neighbors`, 4
`gh_neighbours` (`gh_neighbors`), 4
`gh_to_sf` (`gis_tools`), 5
`gh_to_sp` (`gis_tools`), 5
`gh_to_spdf` (`gis_tools`), 5
`gis_tools`, 5

`sf`, 5, 6
`sp`, 5
`Spatial`, 5
`SpatialPolygons`, 6
`SpatialPolygonsDataFrame`, 5, 6
`st_as_sf`, 6

`utils`, 6