Package: olctools (via r-universe)

September 21, 2024

Type Package

Title Open Location Code Handling in R

Version 0.3.0

Date 2016-05-07

Author Oliver Keyes

Maintainer Oliver Keyes <ironholds@gmail.com>

Description 'Open Location Codes' (https://openlocationcode.com/) are a Google- created standard for identifying geographic locations. olctools provides utilities for validating, encoding and decoding entries that follow this standard.

License MIT + file LICENSE

Suggests testthat, knitr

LinkingTo Rcpp

Imports Rcpp

VignetteBuilder knitr

RoxygenNote 5.0.1

URL https://github.com/Ironholds/olctools

BugReports https://github.com/ironholds/olctools/issues

Repository https://ironholds.r-universe.dev

RemoteUrl https://github.com/ironholds/olctools

RemoteRef HEAD

RemoteSha abf5ab587c9361357898ceb88822d4e220034a67

Contents

`decode_olc` *Decode Open Location Codes into Latitude and Longitude Pairs*

Description

`decode_olc` takes Open Location Codes and, if they're valid (see `validate_full`) returns the minium, centred and maximum latitude and longitude for those coordinates.

Usage

```
decode_olc(olcs)
```

Arguments

`olcs`	a vector of Open Location Codes, generated through `encode_olc` or an equivalent tool.

See Also

`encode_olc` for the opposite operation, and `shorten_olc` to convert "full" Open Location Codes to "short" Open Location Codes.

Examples

```
decode_olc("7FG49Q00+")
```

`encode_olc` *Encode Latitude and Longitude Pairs as Open Location Codes*

Description

`encode_olc` creates Open Location Codes from latitude and longitude values, of a specified length.

Usage

```
encode_olc(lats, longs, length)
```

Arguments

`lats`	a numeric vector of latitudes.
`longs`	a numeric vector of longitudes, equivalent in size to `lats`
`length`	the length you want the resulting OLCs to be. The conventional lengths are 10 or 11, with any number above 8 and any *even* number below it being acceptable. `length` should consist of either a single value, if you want all codes to be calculated to the same length, or a vector of values the same size as `lats` and `longs` if you want to pre-set values.

See Also

decode_olc for the opposite operation, and shorten_olc to convert "full" Open Location Codes to "short" Open Location Codes.

Examples

```
encode_olc(20.375, 2.775,6)
```

---

olctools *Tools for handling Open Location Codes*

Description

Open Location Codes are a Google-created standard for identifying geographic locations. olctools provides utilities for validating, encoding and decoding entries that follow this standard.

---

recover_olc *Recover Full Open Location Codes From Shortened Codes*

Description

shorten_olc (and other sources) shorten a code, reducing the space it occupies. They also limit its ability to be translated back into latitude/longitude pairs. recover_olc recovers a full code from a shortened one, allowing it to be decoded with decode_olc. *Some* loss of accuracy or precision is expected - and as it finds the closest match to the coordinates rather than to the original code, the characters may be very different.

Usage

```
recover_olc(olcs, lats, longs)
```

Arguments

olcs	a vector of short open location codes, generated with shorten_olc or through any other means.
lats	a numeric vector of latitudes.
longs	a numeric vector of longitudes, equivalent in size to lats.

Examples

```
# Shorten an OLC and then recover the nearest full code. Note the actual characters differ.
shortened_code <- shorten_olc("8FVC9G8F+6X", 47.5, 8.5);
recovered_code <- recover_olc(shortened_code, 47.4, 8.6);
```

shorten_olc *Shorten Full Open Location Codes*

Description

One of the things that makes OLCs useful is that they can shortened - you can trim characters off them, saving space without substantially compromising the accuracy. `shorten_olc` takes full-length OLCs (generated with `encode_olc` or any other way) and shortens them.

Usage

```
shorten_olc(olcs, lats, longs)
```

Arguments

`olcs`	a vector of open location codes, generated with `encode_olc` or through any other means.
`lats`	a numeric vector of latitudes.
`longs`	a numeric vector of longitudes, equivalent in size to `lats`.

See Also

`encode_olc` to create full Open Location Codes.

Examples

```
#Encode an OLC and then shorten it
olc <- encode_olc(51.3708675,-1.217765625, 12)
validate_full(olc)
# [1] TRUE

olc <- shorten_olc(olc, 51.3708675,-1.217765625)
validate_short(olc)
# [1] TRUE
```

validate_olc *Check the Validity of Open Location Codes*

Description

These functions allow a useR to check whether OLCs they've been provided are valid or not. `valid_short` identifies whether a vector of OLCs are valid "short" codes; `valid_long` identifies whether OLCs are valid "long" codes, and `valid_full` identifies whether OLCs are valid, full stop.

Usage

```
validate_olc(codes)

validate_short(codes)

validate_full(codes)
```

Arguments

codes	a character vector containing Open Location Codes.

Value

a vector of TRUE and FALSE values, where TRUE corresponds to a valid code and FALSE an invalid.

See Also

decode_olc and encode_olc for creating and resolving valid Open Location Codes.

Examples

```
#Validate that a particular OLC is valid
validate_olc("WC2345+G6g")
#[1] TRUE

#It is! Is it a short?
validate_short("WC2345+G6g")
#[1] TRUE
#Yep!

#So it's not full?
validate_full("WC2345+G6g")
#[1] FALSE
#Nope!
```

Index