Package ‘osrmr’

October 14, 2022

Type Package
Title Wrapper for the 'OSRM' API
Version 0.1.36
Date 2021-05-31
Author Adrian Staempfli, Christoph Strauss
Maintainer Adrian Staempfli <adrian.staempfli@ost.ch>
Description Wrapper around the 'Open Source Routing Machine (OSRM)' API <http://project-osrm.org/>. osrmr works with API versions 4 and 5 and can handle servers that run locally as well as the 'OSRM' webserver.
License GPL-3
LazyData TRUE
Imports assertthat, bitops, rjson, R.utils, stringr
Suggests testthat, knitr, rmarkdown, microbenchmark
RoxygenNote 7.1.1
VignetteBuilder knitr
SystemRequirements To use the Localhost of OSRM, you need to build OSRM <https://github.com/Project-OSRM/osrm-backend/wiki/Building-OSRM> locally
NeedsCompilation no
Repository CRAN
Date/Publication 2021-05-31 09:40:02 UTC

R topics documented:

- decode_geom ......................................................... 2
- encoded_string_api_4 ........................................... 3
- encoded_string_api_5 ........................................... 3
- make_request ...................................................... 4
decode_geom

Transform encoded polylines to lat-lng data.frame.

Description


Usage

decode_geom(encoded, precision = stop("a numeric, either 5 or 6"))

Arguments

encoded A character containing encoded polylines

precision A numeric (either 5 or 6) to specify the precision of [lat,lng] encoding. (OSRM API v4 used precision 5 with "polyline", OSRM API v5 uses precision 6 with "polyline6")

Value
data.frame with lat and lng

Examples

decoded_api_4 <- decode_geom(osmr::encoded_string_api_4, precision = 5)
decoded_api_5 <- decode_geom(osmr::encoded_string_api_5, precision = 6)
decoded_api_4[1:3,]
#   lat  lng
# 1 47.10020 8.09970
# 2 47.09850 8.09207
# 3 47.09617 8.09118
decoded_api_5[1:3,]
#   lat  lng
# 1 47.10020 8.099703
# 2 47.09850 8.092074
# 3 47.09617 8.091181
assertthat::assert_that(all.equal(decoded_api_4, decoded_api_5, tolerance = 1e-6))
**Description**

`encoded_string_api_4`: An encoded route to illustrate the `'osrmr::decode_geom()'` function. After decoding all points on the route are available as wgs84 coordinates. Decoding varies on the API version of OSRMR. This version is decoded using API v4.

**Usage**

`encoded_string_api_4`

**Format**

A string containing an encoded polyline

---

**Description**

`encoded_string_api_5`: An encoded route to illustrate the `'osrmr::decode_geom()'` function. After decoding all points on the route are available as wgs84 coordinates. Decoding varies on the API version of OSRMR. This version is decoded using API v5.

**Usage**

`encoded_string_api_5`

**Format**

A string containing an encoded polyline
make_request  
*Run one server request for OSRM (online- or localhost)*

**Description**

In order to fail gracefully, this function handles errors and warnings if the asked server (online- or localhost) doesn’t work properly. In this case the error message is returned and connections are closed using `base::closeAllConnections()`.

**Usage**

`make_request(request)`

**Arguments**

- **request**: A character

**Details**

If the asked server doesn’t react within 1 second, a warning is thrown using `R.utils::withTimeout(..., timeout = 1)`

**Value**

A list. The dimension of the list depends on the request and whether the server reacted properly or not.

---

nearest  
*nearest accessible position*

**Description**

`nearest()` calculates the nearest position to the given coordinates which can be accessed by car. The coordinate-standard is WGS84. Attention: The OSRM API v4 is only working locally, but not with the `OSRM` webserver.

**Usage**

`nearest(lat, lng, api_version = 5, localhost = F, timeout = 0.001)`

**Arguments**

- **lat**: A numeric (-90 < lat < 90)
- **lng**: A numeric (-180 < lng < 180)
- **api_version**: A numeric (either 4 or 5)
- **localhost**: A logical (TRUE = localhost is used, FALSE = onlinehost is used)
- **timeout**: A numeric indicating the timeout between server requests (in order to prevent queue overflows). Default is 0.001s.
nearest_api_v4

Description

nearest_api_v4() calculates the nearest position to the given coordinates which can be accessed by car with the OSRM API 4. The coordinate-standard is WGS84. Attention: The OSRM API v4 is only working locally, but not with the 'OSRM' webserver.

Usage

nearest_api_v4(lat, lng, address)

Arguments

lat, A numeric (-90 < lat < 90)
lng, A numeric (-180 < lng < 180)
address, A character specifying the serveraddress (local or online)

Value

A data.frame with lat and lng
## nearest_api_v5

### nearest accessible position for OSRM API v5

#### Description

nearest_api_v5() calculates the nearest position to the given coordinates which can be accessed by car with the OSRM API v5. The coordinate-standard is WGS84.

#### Usage

nearest_api_v5(lat, lng, address)

#### Arguments

- **lat**, A numeric (-90 < lat < 90)
- **lng**, A numeric (-180 < lng < 180)
- **address**, A character specifying the serveraddress (local or online)

#### Value

A data.frame with lat and lng

#### Examples

```r
## Not run:
osrmr:::nearest_api_v5(47,9, osrmr:::server_address(FALSE))
Sys.setenv("OSRM_PATH_API_5"="C:/OSRM_API5")
osrmr::run_server(Sys.getenv("OSRM_PATH_API_5"), "switzerland-latest.osrm")
osrmr:::nearest_api_v5(47,9, osrmr:::server_address(TRUE))
osrmr::quit_server()
Sys.unsetenv("OSRM_PATH_API_5")
## End(Not run)
```
### quit_server

**Quit local OSRM server**

#### Description

quit_server() quits your local OSRM server by using a taskkill shell command (depending on your OS).

#### Usage

quit_server()

#### Examples

```r
## Not run:
oosrmr::quit_server()
# NULL
## End(Not run)
```

### run_server

**Start local OSRM server**

#### Description

run_server() starts your local OSRM server by using a shell command (depending on your OS). A local (pre-built) version of the OSRM-engine must be on your device. (https://github.com/Project-OSRM/osrm-backend/wiki/Building-OSRM).

#### Usage

```r
r
run_server(map_name, osrm_path = Sys.getenv("OSRM_PATH"))
```

#### Arguments

- **map_name**: A character (name of your pre-built map - e.g. "switzerland-latest.osrm")
- **osrm_path**: A string pointing to your local OSRM installation. Default is the environment variable "OSRM_PATH".

#### Details

To start the server, it is necessary to know its location. If it was installed in C:/OSRM_APIx, it is easiest to set an environment variable which points to the folder via Sys.setenv(). Note: You need to set the variable in each session.

#### Value

- **error_code** A character
server_address

Examples

## Not run:
Sys.setenv("OSRM_PATH"="C:/OSRM_API4")
osrmr::run_server("switzerland-latest.osrm")
# 0
Sys.setenv("OSRM_PATH"="C:/OSRM_API5")
osrmr::run_server("switzerland-latest.osrm")
# 0
Sys.unsetenv("OSRM_PATH")
## End(Not run)

server_address

server_address() returns the URL address of the OSRM localhost or OSRM webserver, depending on the value of the variable 'use_localhost'. This is an internal function. The address is used as a part of a OSRM server-request.

Description

server_address() returns the URL address of the OSRM localhost or OSRM webserver, depending on the value of the variable 'use_localhost'. This is an internal function. The address is used as a part of a OSRM server-request.

Usage

server_address(use_localhost)

Arguments

use_localhost A logical, indicating whether to use the localhost or not.

Value

character, the address of an OSRM server

Examples

osrmr:::server_address(TRUE)
osrmr:::server_address(FALSE)
# [1] "http://router.project-osrm.org"
viaroute is a function that calculates route informations using OSRM. OSRM chooses the nearest point which can be accessed by car for the start- and end-destination. The coordinate-standard is WGS84. Attention: The OSRM API-4 is only working locally, but not with the onlinehost.

Usage

```r
viaroute(
    lat1, lng1, lat2, lng2, instructions, api_version, localhost, timeout = 0.001
)
```

Arguments

- `lat1`: A numeric (-90 < lat1 < 90) -> start-destination
- `lng1`: A numeric (-180 < lng1 < 180) -> start-destination
- `lat2`: A numeric (-90 < lat2 < 90) -> end-destination
- `lng2`: A numeric (-180 < lng2 < 180) -> end-destination
- `instructions`: A logical. If FALSE, only the traveltime (in seconds, as numeric) will be returned. If TRUE, more details of the route are returned (as list).
- `api_version`: A numeric (either 4 or 5)
- `localhost`: A logical (TRUE = localhost is used, FALSE = onlinehost is used)
- `timeout`: A numeric indicating the timeout between server requests (in order to prevent queue overflows). Default is 0.001s.

Value

a numeric or a list (depending on instructions)

Examples

```r
# direct examples of the online API
## Not run:
## link <- "http://router.project-osrm.org/route/v1/driving/8.1,47.1;8.3,46.9?steps=false"
```
a <- rjson::fromJSON(file = link)

# example with onlinehost API5
osmr:::viaroute(47.1, 8.1, 46.9, 8.3, FALSE, 5, FALSE)

# examples with localhost API4/API5
Sys.setenv("OSRM_PATH"="C:/OSRM_API4")
osmr::run_server("switzerland-latest.osrm")
osmr::viaroute(47.1, 8.1, 46.9, 8.3, FALSE, 4, TRUE)
osmr::quit_server()
Sys.unsetenv("OSRM_PATH")

Sys.setenv("OSRM_PATH"="C:/OSRM_API5")
osmr::run_server("switzerland-latest.osrm")
osmr::viaroute(47.1, 8.1, 46.9, 8.3, FALSE, 5, TRUE)
osmr::quit_server()
Sys.unsetenv("OSRM_PATH")
## End(Not run)

### viaroute_api_v4 travel time or full information of a route for OSRM API 4

#### Description
For a given start- and end-destination, viaroute() calculates route informations using OSRM API 4. OSRM chooses the nearest point which can be accessed by car for the start and destination. The coordinate-standard is WGS84. Attention: The OSRM API-4 is only working locally, but not with the onlinehost.

#### Usage
viaroute_api_v4(lat1, lng1, lat2, lng2, instructions, address)

#### Arguments
- **lat1**: A numeric (-90 < lat1 < 90) -> start-destination
- **lng1**: A numeric (-180 < lng1 < 180) -> start-destination
- **lat2**: A numeric (-90 < lat2 < 90) -> end-destination
- **lng2**: A numeric (-180 < lng2 < 180) -> end-destination
- **instructions**: A logical. If FALSE, only the traveltime (in seconds, as numeric) will be returned. If TRUE, more details of the route are returned (as list).
- **address**: A character specifying the serveraddress (local or online)

#### Value
- A numeric or a list (depending on parameter instructions)
viaroute_api_v5

Examples

```r
## Not run:
Sys.setenv("OSRM_PATH"="C:/OSRM_API4")
osmr::run_server("switzerland-latest.osrm")
osmr:::viaroute_api_v4(47,9,48,10, FALSE, osmr:::server_address(TRUE))
osmr:::quit_server()
Sys.unsetenv("OSRM_PATH")
## End(Not run)
```

viaroute_api_v5  travel time or full information of a route for OSRM API 5

Description

For a given start- and end-destination, viaroute() calculates route informations using OSRM API 5. OSRM chooses the nearest point which can be accessed by car for the start and destination. The coordinate-standard is WGS84. Attention: The OSRM API-4 is only working locally, but not with the onlinehost.

Usage

```r
viaroute_api_v5(lat1, lng1, lat2, lng2, instructions, address)
```

Arguments

- `lat1`: A numeric (-90 < lat1 < 90) -> start-destination
- `lng1`: A numeric (-180 < lng1 < 180) -> start-destination
- `lat2`: A numeric (-90 < lat2 < 90) -> end-destination
- `lng2`: A numeric (-180 < lng2 < 180) -> end-destination
- `instructions`: A logical. If FALSE, only the traveltime (in seconds, as numeric) will be returned. If TRUE, more details of the route are returned (as list).
- `address`: A character specifying the serveraddress (local or online)

Value

a numeric or a list (depending on parameter instructions)

Examples

```r
## Not run:
# example with onlinehost
osmr:::viaroute_api_v5(47, 9, 48, 10 , FALSE, osmr:::server_address(FALSE))

# example with localhost
Sys.setenv("OSRM_PATH"="C:/OSRM_API5")
osmr::run_server("switzerland-latest.osrm")
osmr:::viaroute_api_v5(47, 9, 48, 10 , FALSE, osmr:::server_address(TRUE))
```
osrmr::quit_server()
Sys.unsetenv("OSRM_PATH")
## End(Not run)
Index

* datasets
  encoded_string_api_4, 3
  encoded_string_api_5, 3

decode_geom, 2

encoded_string_api_4, 3
encoded_string_api_5, 3

make_request, 4

nearest, 4
nearest_api_v4, 5
nearest_api_v5, 6

quit_server, 7

run_server, 7

server_address, 8

viaroute, 9
viaroute_api_v4, 10
viaroute_api_v5, 11