Package ‘TargomoR’

December 7, 2019

Type     Package
Title    Interface to the 'Targomo' API
Version 0.2.0
Description Functions to provide an intuitive interface for retrieving travel time data from the 'Targomo' API (see <https://targomo.com/developers/> for details). Provides support for retrieving isochrone polygons, travel routes, times and distances. Also includes functions for easily adding the data to 'leaflet' maps, and functions for using the 'Targomo' map tiles.
License GPL-3
URL https://cwthom.github.io/TargomoR
BugReports https://github.com/cwthom/TargomoR/issues
Imports leaflet, magrittr, htr, jsonlite, geojsonsf, sf, tibble, xml2
Encoding UTF-8
LazyData true
RoxygenNote 6.1.1
Suggests testthat (>= 2.1.0), covr, knitr, rmarkdown
NeedsCompilation no
Author Chris Mason-Thom [aut, cre]
Maintainer Chris Mason-Thom <christmasmason@university.com>
Repository CRAN
Date/Publication 2019-12-06 23:40:08 UTC

R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>addTimeLegend</td>
<td>2</td>
</tr>
<tr>
<td>attribution</td>
<td>3</td>
</tr>
<tr>
<td>callTargomoAPI</td>
<td>4</td>
</tr>
<tr>
<td>capabilities</td>
<td>4</td>
</tr>
<tr>
<td>createRequestBody</td>
<td>5</td>
</tr>
<tr>
<td>createRequestURL</td>
<td>6</td>
</tr>
<tr>
<td>createRoutePopup</td>
<td>6</td>
</tr>
</tbody>
</table>
addTimeLegend

**Description**

Add Time Legend to Map

**Usage**

```r
addTimeLegend(map, palette, values, options, group)
```

**Arguments**

- **map**: A leaflet map
- **palette**: A colour palette (from `createTimePalette`)
- **values**: Values to use (travel times)
- **options**: A set of `timeLegendOptions`
- **group**: The layer group to add the legend to

**Value**

The leaflet map with the time legend in a control
### Description

Functions providing link to Targomo Attributions page, depending on plan.

### Usage

- `attributionFreeIframe()`
- `attributionOtherIframe()`
- `attributionLink()`
- `addTargomoAttribution(map, free_plan = TRUE, ...)`

### Arguments

- `map` A leaflet map
- `free_plan` Logical - is the Targomo plan you’re using free or paid?
- `...` Further arguments to pass to `leaflet::addControl` e.g. `position`

### Value

A link or iframe to the attributions page.

### Examples

```r
# load leaflet package
library(leaflet)

# add an attribution iframe to a map
leaflet() %>%
  addTargomoAttribution(free_plan = FALSE)

# return the attribution link
attributionLink()
```
callTargomoAPI *Call the Targomo API*

**Description**

Function to wrap around `httr::POST`, sending the request body to the API.

**Usage**

```r
callTargomoAPI(api_key = Sys.getenv("TARGOMO_API_KEY"),
               region = Sys.getenv("TARGOMO_REGION"), service, body,
               config = list(), verbose = FALSE, progress = FALSE,
               timeout = NULL)
```

**Arguments**

- `api_key`: The Targomo API key.
- `region`: The Targomo region.
- `service`: The Targomo service - 'polygon', 'route', or 'time'.
- `body`: A request body made with `createRequestBody`.
- `config`: Config options to pass to `httr::POST` e.g. proxy settings.
- `verbose`: Display info on the API call?
- `progress`: Display a progress bar?
- `timeout`: Timeout in seconds (leave NULL for no timeout/curl default).

**Value**

A `httr` response object with the API response (whether successful or not).

capabilities *Get Account Capabilities*

**Description**

Function to return a list of the capabilities of the API Key. Comes with a print method to print out the main results nicely in the console.

**Usage**

```r
getTargomoCapabilities(api_key = Sys.getenv("TARGOMO_API_KEY"),
                       region = Sys.getenv("TARGOMO_REGION"), config = list(),
                       verbose = FALSE, progress = FALSE)
```

```r
## S3 method for class 'tgm_capabilities'
print(x, ...)
```
createRequestBody

Arguments

- **api_key**: Your Targomo API key - defaults to the TARGOMO_API_KEY environment variable
- **region**: Your Targomo region - defaults to the TARGOMO_REGION environment variable
- **config**: Config options to pass to httr::GET e.g. proxy settings
- **verbose**: Whether to print out information about the API call.
- **progress**: Whether to show a progress bar of the API call.
- **x**: A list, output of getTargomoCapabilities
- **...**: Further arguments to print

Value

A list of the capabilities of the given API key, in the given region

Examples

```r
caps <- getTargomoCapabilities()

# print default
print.default(caps)

# print using bespoke method
print(caps)
```

**createRequestBody**  
Create Request Body

Description

Function to create a request body using the sources and options given.

Usage

```r
createRequestBody(service, sources = NULL, targets = NULL, options)
```

Arguments

- **service**: The Targomo Service to create a body for - 'polygon', 'time', 'route'.
- **sources**: A processed sources object to pass to the API.
- **targets**: A processed targets object (optional).
- **options**: A processed options list.

Value

A JSON request body to be POST-ed to the API
createRequestURL  

*Create Request URL*

**Description**

Function to create the request URL.

**Usage**

```
createRequestURL(region, end_point)
```

**Arguments**

- `region`  
The Targomo region.
- `end_point`  
The API end_point.

**Value**

Character string, the URL of the chosen endpoint

createRoutePopup  

*Create Route Popups*

**Description**

Function for constructing popups on routes.

**Usage**

```
createRoutePopup(data, transit = FALSE, startEnd = transit)
```

**Arguments**

- `data`  
The route data from which to create the popup.
- `transit`  
Whether this is a transit route.
- `startEnd`  
Whether to show information on the start and end points.

**Value**

A HTML string for the route segment popup
**createTimePalette**

Create a Colour Palette for Time Service Results

**Description**

Create a Colour Palette for Time Service Results

**Usage**

```r
createTimePalette(palette, type, maxTime, bins, reverse)
```

**Arguments**

- `palette`: A colour palette e.g. "viridis", "Blues"
- `type`: Either "numeric" or "bin"
- `maxTime`: The maximum time value to consider
- `bins`: Either a single number of bins, or a vector of cut points.
- `reverse`: Whether to reverse the colour palette.

**Value**

A colour palette function for use with the time legend and markers

**deriveOptions**

Derive Options

**Description**

Function to create options in a nested list structure suitable to be turned into JSON.

**Usage**

```r
deriveOptions(options)
```

**Arguments**

- `options`: The output of `targomoOptions`

**Value**

List of options correctly structured for converting to JSON and passing to the API
deriveSources  Derive Sources/Targets

Description

Function to create the sources needed to query the Targomo API.

Usage

createIds(data = NULL, id = NULL)
createPoints(data = NULL, lat = NULL, lng = NULL, id = NULL)
deriveSources(points, options)
deriveTargets(points)

Arguments

data  The data object
id    The id vector or formula to resolve
lat, lng  The lat/lng vectors or formulae to resolve
points A processed data object (for sources/targets).
options A processed options object (for sources).

Value

A data.frame of sources/targets, with IDs attached.

draw-routes  Draw Routes

Description

Helper functions for drawing different routes.

Usage

drawRouteSegment(map, segment, drawOptions, type, group, ...)
drawWalk(map, segment, drawOptions, group, ...)
drawBike(map, segment, drawOptions, group, ...)
drawCar(map, segment, drawOptions, group, ...)
drawTransit(map, segment, drawOptions, group, ...)
### formatting

#### Format Edgeweights (times and distances)

**Description**

Functions to make the interface easier and more intuitive to use - they convert numeric edgeweights (e.g. 900) to character strings (e.g. "15min"), and vice versa.

**Usage**

```r
classicEdgeWeight(edgeWeight, type)
prettyEdgeWeight(edgeWeight, type)
```

**Arguments**

**edgeWeight**

A time or distance, in numeric or string form.

**type**

Either 'time' or 'distance'.

**Value**

Either a numeric or formatted edgeweight

**Examples**

```r
classicEdgeWeight("1hr 30m", "time") # 5400
classicEdgeWeight("1ml", "distance") # 1609

prettyEdgeWeight(1245, "time") # 30min 45s
prettyEdgeWeight(1245, "distance") # 1km 245m
```
**getTargomoMapURL** *(Targomo Map Tiles URL)*

**Description**

Targomo Map Tiles URL

**Usage**

```r
getTargomoMapURL(style = "basic",
                  api_key = Sys.getenv("TARGOMO_API_KEY"))
```

**Arguments**

- **style**
  
  A valid Targomo Map Style - see `targomoMapStyles()`

- **api_key**
  
  Your Targomo API key - defaults to the TARGOMO_API_KEY environment variable

**Value**

The URL of the requested map tile

**Examples**

```r
getTargomoMapURL(style = "toner", api_key = NULL)
```

---

**getTargomoPolygons** *(Add Targomo Polygons to a Leaflet Map)*

**Description**

Functions for retrieving isochrone polygons from the Targomo API and adding drawing them on a leaflet map.

**Usage**

```r
getTargomoPolygons(source_data = NULL, source_lat = NULL, source_lng = NULL, options = targomoOptions(),
                    api_key = Sys.getenv("TARGOMO_API_KEY"),
                    region = Sys.getenv("TARGOMO_REGION"), config = list(),
                    verbose = FALSE, progress = FALSE, timeout = NULL)
```

```r
drawTargomoPolygons(map, polygons, drawOptions = polygonDrawOptions(),
                    group = NULL, ...)
```
addTargomoPolygons(map, source_data = NULL, source_lng = NULL, source_lat = NULL, options = targomoOptions(), drawOptions = polygonDrawOptions(), group = NULL, ..., api_key = Sys.getenv("TARGOMO_API_KEY"), region = Sys.getenv("TARGOMO_REGION"), config = list(), verbose = FALSE, progress = FALSE, timeout = NULL)

Arguments

source_data The data object from which source ppoints are derived.
source_lng, source_lat Vectors/one-sided formulas of longitude and latitude.
options A list of targomoOptions to call the API.
api_key Your Targomo API key - defaults to the TARGOMO_API_KEY environment variable
region Your Targomo region - defaults to the TARGOMO_REGION environment variable
config Config options to pass to httr::POST e.g. proxy settings
verbose Whether to print out information about the API call.
progress Whether to show a progress bar of the API call.
timeout Timeout in seconds (leave NULL for no timeout/curl default).
map A leaflet map.
polygons A polygons dataset returned by getTargomoPolygons, for drawing
drawOptions A list of polygonDrawOptions to determine how to show the resulting polygons on the map.
group The leaflet map group to add the polygons to. A single group is used for all the polygons added by one API call.
... Further arguments to pass to addPolygons

Value

For ‘get*’, an object of class "sf" containing the polygons. For ‘draw*’ and ‘add*’, the leaflet map returned with the polygons drawn on.

Examples

# load leaflet package
library(leaflet)
l <- leaflet()

# get the polygons
p <- getTargomoPolygons(source_lat = 51.5007, source_lng = -0.1246, 
                        options = targomoOptions(travelType = "bike"))

# draw them on the map
l %>% drawTargomoPolygons(polygons = p, group = "BigBenBike")
# note could combine get... and draw... into one with add...

messageMultipleTravelModes

*Message if multiple Travel Modes supplied*

**Description**
Message if multiple Travel Modes supplied

**Usage**
messageMultipleTravelModes(tms)

**Arguments**
tms A vector of travel modes

**options**

*Set Targomo Options*

**Description**
This function sets the options to be passed to the API service. For full details of available options see `https://docs.targomo.com/core/`

**Usage**
targomoOptions(travelType = "bike", travelTimes = list(600, 1200, 1800), intersectionMode = "union", carRushHour = FALSE, walkSpeed = 5, walkUpHillAdjustment = 10, walkDownHillAdjustment = 0, bikeSpeed = 15, bikeUpHillAdjustment = 20, bikeDownHillAdjustment = -10, transitDate = NULL, transitTime = NULL, transitDuration = NULL, transitMaxWalkingTimeFromSource = NULL, transitMaxWalkingTimeToTarget = NULL, transitEarliestArrival = FALSE, transitMaxTransfers = NULL, edgeWeight = "time", maxEdgeWeight = 1800, elevation = FALSE, serializer = "geojson", srid = 4326, minPolygonHoleSize = NULL, buffer = NULL, simplify = NULL, quadrantSegments = NULL, decimalPrecision = NULL)
options

Arguments

- **travelType**
  What mode of transport to use - car, bike, walk or public transport.

- **travelTimes**
  A list of times - each time corresponds to a different polygon. Your API key will determine how many you can add. Acceptable formats are numeric (interpreted as seconds), or a string of the form .h.m.s. E.g. for 1 hour, "1h", 90 minutes = "1h30m" or "90m" etc.

- **intersectionMode**
  Whether to calculate the union or intersection of multiple sources.

- **carRushHour**
  Account for rush hour while driving.

- **walkSpeed**, **walkUpHillAdjustment**, **walkDownHillAdjustment**
  Settings for walking travel type.

- **bikeSpeed**, **bikeUpHillAdjustment**, **bikeDownHillAdjustment**
  Settings for cycling travel type.

- **transitDate**
  The date for public transport calculations (YYYYMMDD).

- **transitTime**
  The time in seconds since midnight to begin transit.

- **transitDuration**
  The duration of the transit timeframe (seconds or .h.m.s string).

- **transitMaxWalkingTimeFromSource**, **transitMaxWalkingTimeToTarget**
  Settings for transit travel type.

- **transitMaxTransfers**, **transitEarliestArrival**
  Further transit settings.

- **edgeWeight**
  Should calculations be in "time" or "distance"?

- **maxEdgeWeight**
  The max time or distance to search for routes/times in - acceptable formats are numeric (seconds or metres), or a string of the form .h.m.s for time, or .km.m.ml for distance, where ml represents miles (1609 metres).

- **elevation**
  Account for elevation?

- **serializer**
  Should be "geojson" or "json". See API for details.

- **srid**
  The spatial reference of the returned data.

- **minPolygonHoleSize**
  Minimum area of holes in returned polygons.

- **simplify**, **buffer**
  Parameters for manipulating the returned polygons.

- **quadrantSegments**, **decimalPrecision**
  Parameters for fine-tuning the returned polygons.

Value

A (filtered, no NULLs) list of options to pass to the API

Examples

targomoOptions(travelType = "transit")
targomoOptions(travelTimes = list("15m", "30m", "45m", "1h"), maxEdgeWeight = "1h")
**polygonDrawOptions**  
*Options for Drawing Polygons on the Map*

### Description

Function to return a list of the desired drawing options - you can set all the usual parameters of a call to `addPolygons`.

### Usage

```r
polygonDrawOptions(stroke = TRUE, weight = 5, color = c("red", "orange", "green"), opacity = 0.5, fill = TRUE, fillColor = color, fillOpacity = 0.2, dashArray = NULL, smoothFactor = 1, noClip = FALSE)
```

### Arguments

- **stroke**: Whether to draw the polygon borders.
- **weight**: Stroke width in pixels.
- **color**: Stroke colour.
- **opacity**: Stroke opacity.
- **fill**: Whether to fill the polygons in with colour.
- **fillColor**: The fill colour.
- **fillOpacity**: The fill opacity.
- **dashArray**: A string to define the stroke dash pattern.
- **smoothFactor**: How much to simplify polylines on each zoom level.
- **noClip**: Whether to disable polyline clipping.

### Value

A list of options governing how the polygons appear on the map

### Examples

```r
# show the list
polygonDrawOptions()
```
Process API responses

Description
Functions to turn a successful request into data - either polygons, routes or times.

Usage
- catchBadResponse(response)
- processResponse(response, service)
- processPolygons(payload)
- getRouteFeatures(route)
- processRoutes(payload)
- processTimes(payload)

Arguments
- response: A response object from callTargomoAPI.
- service: The Targomo API service being called - polygon, route or time.
- payload: The http::content of the response.
- route: A single element of the returned routes list.

Process Capabilities

Description
This function takes the raw JSON list of capabilities and converts then into a formatted list.

Usage
- processCapabilities(response)

Arguments
- response: The API response object

Value
The formatted list, of class ’tgm_capabilities’
routeDrawOptions

Options for Drawing Routes on the Map

Description

Function to return a list of the desired drawing options - you can set colours, line weights and dash styles for each transport type, whether to show the source and target markers, and whether to show transfers between different modes of transport.

Usage

routeDrawOptions(showMarkers = TRUE, showTransfers = TRUE, 
walkColour = "green", walkWeight = 5, walkDashArray = "1,10", 
carColour = "blue", carWeight = 5, carDashArray = NULL, 
bikeColour = "orange", bikeWeight = 5, bikeDashArray = NULL, 
transitColour = "red", transitWeight = 5, transitDashArray = NULL, 
transferColour = "blue", transferRadius = 10)

Arguments

showMarkers Whether to show the source/target markers.
showTransfers whether to highlight transfers between different modes of transport.
wakColour, bikeColour, carColour, transitColour
    Set the line colours.
walkWeight, bikeWeight, carWeight, transitWeight
    Set the line weights.
wakDashArray, bikeDashArray, carDashArray, transitDashArray
    Set the dash styles.
transferColour Set the colour of transfer markers.
transferRadius Set the size of transfer markers.

Value

A list of options governing how the routes are drawn on the map.

Examples

# show the list
routeDrawOptions()
Add Targomo Routes to a Leaflet Map

Description
This function takes source and target data, together with options for the API and drawing options, and returns the map with the requested routes.

Usage
getTargomoRoutes(source_data = NULL, source_lat = NULL, source_lng = NULL, target_data = NULL, target_lat = NULL, target_lng = NULL, source_id = NULL, target_id = NULL, options = targomoOptions(), api_key = Sys.getenv("TARGOMO_API_KEY"), region = Sys.getenv("TARGOMO_REGION"), config = list(), verbose = FALSE, progress = FALSE, timeout = NULL)

drawTargomoRoutes(map, routes, drawOptions = routeDrawOptions(), group = NULL, ...)

addTargomoRoutes(map, source_data = NULL, source_lat = NULL, source_lng = NULL, source_id = NULL, target_data = NULL, target_lat = NULL, target_lng = NULL, target_id = NULL, options = targomoOptions(), drawOptions = routeDrawOptions(), group = NULL, api_key = Sys.getenv("TARGOMO_API_KEY"), region = Sys.getenv("TARGOMO_REGION"), config = list(), verbose = FALSE, progress = FALSE, timeout = NULL)

Arguments
source_data, target_data
The source and target points for your routes - supported types are data.frame matrix and objects from the sf and sp packages.

source_lat, source_lng
Columns identifying the latitude and longitude columns in your sourcedata, or numeric vectors of equal length.

target_lat, target_lng
As for source_lat, source_lng but for target data.

source_id, target_id
Formulas or vectors of IDs to give to your source and target points. These will be used to match back to the input data if applicable.

options
A list of targomoOptions to send to the API.

api_key
Your Targomo API key - defaults to the TARGOMO_API_KEY environment variable.

region
Your Targomo region - defaults to the TARGOMO_REGION environment variable.
routes

- **config**: Config options to pass to `httr::POST` e.g. proxy settings
- **verbose**: Whether to print out information about the API call.
- **progress**: Whether to show a progress bar of the API call.
- **timeout**: Timeout in seconds (leave NULL for no timeout/curl default).
- **map**: A leaflet map
- **routes**: A list of route segments provided by `getTargomoRoutes`.
- **drawOptions**: A list of `routeDrawOptions` to determine how to show the resulting routes on the map.
- **group**: The leaflet map group to add the routes to. One group is used for all map elements being drawn per call to the API.
- **...**: Further arguments to pass to `addPolylines`

**Value**

For `get*`, a list of objects of class "sf" containing the routes For `draw*` and `add*`, the leaflet map returned with the routes drawn on.

**See Also**

`draw-routes`

**Examples**

```r
# load leaflet package
library(leaflet)
l <- leaflet()

# get route from Big Ben to Tower Bridge
r <- getTargomoRoutes(source_lat = 51.5007, source_lng = -0.1246,
                      target_lat = 51.5055, target_lng = -0.0754,
                      options = targomoOptions(travelType = c("bike", "transit")))

# draw the routes on the map
l %>% drawTargomoRoutes(routes = r)

# note, could combine get.. and draw... into one with add...
```
**setTargomoVariables**  
*Set Targomo Environment Variables*

**Description**

This helper function allows you to set your API key in either a global or local .Renviron file, for ease of use. All of the TargomoR functions which call the Targomo API require an API key, and use the TARGOMO_API_KEY environment variable by default. Similarly you can set your default region.

**Usage**

```r
setTargomoVariables(api_key = NULL, region = NULL, overwrite = FALSE, global = FALSE)
```

**Arguments**

- `api_key`  
  Your Targomo API key
- `region`  
  Your preferred Targomo default region
- `overwrite`  
  Whether to overwrite an existing setting
- `global`  
  If TRUE, write to a global .Renviron in Sys.getenv("HOME")

**Details**

For available regions, see here: [https://targomo.com/developers/resources/availability/](https://targomo.com/developers/resources/availability/)

**Value**

Invisibly, the API key - this function is called for its side effects

**Examples**

```r
## Not run:
# write to a global file at Sys.getenv("HOME")
setTargomoVariables(api_key = "YOUR_SECRET_KEY", region = "asia", overwrite = TRUE, global = TRUE)
## End(Not run)
```

**targomoAPI**  
*Targomo API base URL*

**Description**

Targomo API base URL

**Usage**

```r
targomoAPI()
```
### tidy-capabilities

**Helper functions for tidying up capabilities response**

**Description**

These functions tidy up the raw capabilities lists.

**Usage**

- tidyGeneral(general)
- tidyTransit(transit)
- tidySpeeds(speeds)

**Arguments**

- general, transit, speeds
  - The parts of the list

**Value**

Lists of capabilities (data.frames or vectors)

---

### tiles

**Add Targomo Basemaps to a Leaflet Map**

**Description**

This function wraps round leaflet::addTiles to provide access to the Targomo basemaps.

**Usage**

```r
addTargomoTiles(map, style = "basic",
    api_key = Sys.getenv("TARGOMO_API_KEY"), layerId = NULL,
    group = NULL, ...)
targomoMapStyles()
```
timeDrawOptions

Arguments

- **map**: A leaflet map
- **style**: A valid Targomo Map Style - see `targomoMapStyles()`
- **api_key**: Your Targomo API key - defaults to the `TARGOMO_API_KEY` environment variable
- **layerId**: The layer id to pass to `leaflet::addTiles`
- **group**: The layer group to pass to `leaflet::addTiles`
- **...**: Further options to pass to `leaflet::addTiles` e.g. `options`

Value

The leaflet map with the requested map tiles

Examples

```r
# load leaflet package
library(leaflet)

# add basic style to map
leaflet() %>% addTargomoTiles(style = "basic")

# add dark blue style to map (without labels)
leaflet() %>% addTargomoTiles(style = "darkblue-nolabels")

# list Targomo Map Styles
targomoMapStyles()
```

timeDrawOptions

Options for Drawing Times on the Map

Description

Options for Drawing Times on the Map

Usage

```r
timeDrawOptions(palette = "viridis", type = "numeric",
                maxTime = 1800, reverse = FALSE, bins = c(600, 1200),
                legend = TRUE, legendOptions = timeLegendOptions(), radius = 10,
                stroke = TRUE, weight = 3, color = "black", opacity = 0.5,
                fill = TRUE, fillOpacity = 0.5)
```
Arguments

- **palette**: A colour palette name e.g. "viridis"
- **type**: Either "numeric" or "bin"
- **maxTime**: The max time to allow for
- **reverse**: Whether to reverse the colour palette.
- **bins**: A number of bins or a vector of cut points (only used for the bin palette)
- **legend**: Whether to automatically add a legend.
- **legendOptions**: A `timeLegendOptions` object.
- **radius**: The marker radius.
- **stroke**: Whether to draw the marker border.
- **weight**: Stroke width in pixels.
- **color**: Stroke colour.
- **opacity**: Stroke opacity.
- **fill**: Whether to fill the polygons in with colour.
- **fillOpacity**: The fill opacity.

Value

A list of options governing how time markers are drawn on the map

Examples

```r
# show the list
timeDrawOptions()
```

---

timeLegendOptions  
*time Legend Options*

Description

Time Legend Options

Usage

```r
timeLegendOptions(position = "topright", title = "Travel Times", layerId = NULL)
```

Arguments

- **position**: One of c("topright", "topleft", "bottomright", "bottomleft").
- **title**: The legend title.
- **layerId**: The legend layer ID.
Value

A list of options governing how the time legend appears on the map

Examples

```r
# show the list
timeLegendOptions()
```

Description

This function takes source and target data, together with options for the API and drawing options, and returns the map with the requested travel time data.

Usage

```r
getTargomoTimes(source_data = NULL, source_lat = NULL, source_lng = NULL, target_data = NULL, target_lat = NULL, target_lng = NULL, source_id = NULL, target_id = NULL, options = targomoOptions(), api_key = Sys.getenv("TARGOMO_API_KEY"), region = Sys.getenv("TARGOMO_REGION"), config = list(), verbose = FALSE, progress = FALSE, timeout = NULL)
drawTargomoTimes(map, times, drawOptions = timeDrawOptions(), group = NULL, ...)
addTargomoTimes(map, source_data = NULL, source_lat = NULL, source_lng = NULL, target_data = NULL, target_lat = NULL, target_lng = NULL, source_id = NULL, target_id = NULL, options = targomoOptions(), drawOptions = timeDrawOptions(), group = NULL, ..., api_key = Sys.getenv("TARGOMO_API_KEY"), region = Sys.getenv("TARGOMO_REGION"), config = list(), verbose = FALSE, progress = FALSE, timeout = NULL)
```

Arguments

- `source_data`, `target_data`  
The source and target points for your travel times - supported types are data.frame matrix and objects from the sf and sp packages.
- `source_lat`, `source_lng`  
  One-sided formulas identifying the latitude and longitude columns in your source data, or numeric vectors of equal length.
- `target_lat`, `target_lng`  
  As for `source_lat`, `source_lng` but for target data.
source_id, target_id

Formulas or vectors of IDs to give to your source and target points. These will be used to match back to the input data if applicable.

options

A list of `targomoOptions` to send to the API.

api_key

Your Targomo API key - defaults to the `TARGOMO_API_KEY` environment variable.

region

Your Targomo region - defaults to the `TARGOMO_REGION` environment variable.

config

Config options to pass to `httr::POST` e.g. proxy settings

verbose

Whether to print out information about the API call.

progress

Whether to show a progress bar of the API call.

timeout

Timeout in seconds (leave NULL for no timeout/curl default).

map

A leaflet map

times

A times dataset returned by `getTargomoTimes`

drawOptions

A list of `timeDrawOptions` to determine how to show the resulting times on the map.

group

The leaflet map group to add the times to. One group is used for all map elements being drawn per call to the API.

...

Further arguments to pass to `addCircleMarkers`

Value

For `get*`, an object of class "sf" containing the times. For `draw*` and `add*`, the leaflet map returned with the times drawn on as circle markers.

Examples

```r
# load leaflet package
library(leaflet)
l <- leaflet()

# create a source point (Big Ben) and some random targets
s <- data.frame(lat = 51.5007, lng = -0.1246, id = "BigBen")
t <- data.frame(lat = runif(min = 51.495, max = 51.5055, n = 100),
               lng = runif(min = -0.175, max = -0.075, n = 100))

# get the times
times <- getTargomoTimes(source_data = s, target_data = t,
                         options = targomoOptions(travelType = "car"))

# draw them on the map
l %>% drawTargomoTimes(times = times)
```
Index

addCircleMarkers, 24
addPolygons, 11, 14
addPolylines, 18
addTargomoAttribution (attribution), 3
addTargomoPolygons
  (getTargomoPolygons), 10
addTargomoRoutes (routes), 17
addTargomoTiles (tiles), 20
addTargomoTimes (times), 23
addTimeLegend, 2
attribution, 3
attributionFreeIframe (attribution), 3
attributionLink (attribution), 3
attributionOtherIframe (attribution), 3
callTargomoAPI, 4, 15
capabilities, 4
catchBadResponse (process), 15
createIds (deriveSources), 8
createPoints (deriveSources), 8
createRequestBody, 4, 5
createRequestURL, 6
createRoutePopup, 6
createTimePalette, 2, 7
deriveOptions, 7
deriveSources, 8
deriveTargets (deriveSources), 8
draw-routes, 8
drawBike (draw-routes), 8
drawCar (draw-routes), 8
drawRouteSegment (draw-routes), 8
drawTargomoPolygons
  (getTargomoPolygons), 10
drawTargomoRoutes (routes), 17
drawTargomoTimes (times), 23
drawTransit (draw-routes), 8
drawWalk (draw-routes), 8
formatting, 9
getRouteFeatures (process), 15
getTargomoCapabilities, 5
getTargomoCapabilities (capabilities), 4
getTargomoMapURL, 10
getTargomoPolygons, 10
getTargomoRoutes, 18
getTargomoRoutes (routes), 17
getTargomoTimes (times), 23
messageMultipleTravelModes, 12
numericEdgeWeight (formatting), 9
options, 12
polygonDrawOptions, 11, 14
prettyEdgeWeight (formatting), 9
print, 5
print.tgm_capabilities (capabilities), 4
process, 15
processCapabilities, 15
processPolygons (process), 15
processResponse (process), 15
processRoutes (process), 15
processTimes (process), 15
routeDrawOptions, 9, 16, 18
routes, 17
setTargomoVariables, 19
targomoAPI, 19
targomoMapStyles (tiles), 20
targomoOptions, 7, 11, 17, 24
targomoOptions (options), 12
tidy-capabilities, 20
tidyGeneral (tidy-capabilities), 20
tidySpeeds (tidy-capabilities), 20
tidyTransit (tidy-capabilities), 20
tiles, 20
timeDrawOptions, 21, 24
timeLegendOptions, 2, 22
times, 23