Package 'starsTileServer'

July 23, 2025

Title A Dynamic Tile Server for R

Version 0.1.1

Description
Makes it possible to serve map tiles for web maps (e.g. leaflet) based on a function or a stars object without having to render them in advance. This enables parallelization of the rendering, separating the data source and visualization location and to provide web services.

License GPL-3

Encoding UTF-8

RoxygenNote 7.2.1

URL https://bartk.gitlab.io/starsTileServer,
https://gitlab.com/bartk/starsTileServer

BugReports https://gitlab.com/bartk/starsTileServer/-/issues

Suggests knitr, rmarkdown, leaflet.extras, webshot, mapview, callr, magick, shiny, dplyr, magrittr, abind, testthat (>= 3.0.0), withr

VignetteBuilder knitr

Imports R6, leaflet, plumber, png, rlang, sf, stars, units, assertthat

Config/testthat/edition 3

Config/testthat/parallel true

NeedsCompilation no

Author Bart Kranstauber [aut, cre] (ORCID:
<https://orcid.org/0000-0001-8303-780X>)

Maintainer Bart Kranstauber <b.kranstauber@uva.nl>

Repository CRAN

Date/Publication 2022-08-22 21:50:02 UTC

Contents

`starsTileServer` *A R6 class that extends plumber to function as a tile server*

Description

Creates a tile server based on the R6 class `plumber::Plumber`. In can be created both with a `stars` grid as well as a function or list of functions. The main methods are `run` and `new`.

Super classes

`plumber::Hookable` -> `plumber::Plumber` -> `starsTileServer`

Methods

Public methods:

- `starsTileServer$new()`
- `starsTileServer$add_tile_endpoint()`
- `starsTileServer$get_grid()`
- `starsTileServer$get_attributes()`
- `starsTileServer$get_dimensions()`
- `starsTileServer$get_dimension_values_chr()`
- `starsTileServer$get_non_spatial_dimensions()`
- `starsTileServer$clone()`

Method `new()`**:** This method is used to initialize a new tile server

Usage:

```
starsTileServer$new(grid, colorFun = NULL, tileSize = 256, ...)
```

Arguments:

`grid` Either a stars grid, a path pointing towards a gridded file, a function or a list of named functions

`colorFun` a color function to use for coloring the map tiles, the function needs to be the same format as `leaflet::colorNumeric()`. It is important to specify a color function as it is important to keep the range of the color scale similar between tiles, therefore the minimum and maximum needs to be fixed. It can also be a list of color functions.

`tileSize` The size of the tile (generally 256 pixels, and not tested with other sizes)

`...` Arguments passed on to the plumber::Plumber, most important is the `port` number.

Details: If `grid` is a function it should take a stars grid as the first argument and return the grid with the same topology with values attached. Any other arguments to the function will be part of the API and will be passed to the function as characters.

Returns: An `starsTileServer` object.

Method `add_tile_endpoint()`**:** Add three endpoints to the tile server, to return both the tiles and the color scale used.

Usage:

```
starsTileServer$add_tile_endpoint(prefix, handlerFun, colorFun, params)
```

Arguments:

`prefix` the name to be used by the server for this tile server

`handlerFun` The function that handles the api request and returns the grid

`colorFun` The color function to use for example `leaflet::colorNumeric()`

`params` parameters passed on to the `new` method of plumber::PlumberEndpoint

Method `get_grid()`**:** return the grid used to initialize the function

Usage:

```
starsTileServer$get_grid()
```

Method `get_attributes()`**:** return the attributes of the stars grid

Usage:

```
starsTileServer$get_attributes()
```

Method `get_dimensions()`**:** return the names of the dimensions of the grid

Usage:

```
starsTileServer$get_dimensions()
```

Method `get_dimension_values_chr()`**:** return the values of a dimension as a character

Usage:

```
starsTileServer$get_dimension_values_chr(x)
```

Arguments:

`x` the name of the dimension

Method `get_non_spatial_dimensions()`**:** return all non spatial dimensions

Usage:

```
starsTileServer$get_non_spatial_dimensions()
```

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
starsTileServer$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

Examples

```
m <- matrix(1:20, nrow = 5, ncol = 4)
dim(m) <- c(x = 5, y = 4) # named dim
(s <- stars::st_as_stars(m))
sf::st_crs(s) <- 4326
starsTileServer$new(s)
# Working directly from a file
grid <- system.file("tif/L7_ETMs.tif", package = "stars")
```

```
starsTileServer$new(grid)
## Not run:
starsTileServer$new(s)$run()

## End(Not run)
```

`targetGrid` *Function to calculate a stars grid based on the x,y,z attributes that represents the coordinates of the map tile*

Description

Function to calculate a stars grid based on the x,y,z attributes that represents the coordinates of the map tile

Usage

```
targetGrid(x, y, z, tileSize = 256L)
```

Arguments

`x`	The location of the tile in the x dimension
`y`	The location of the tile in the y dimension
`z`	The zoom level
`tileSize`	The size of the tile generally 256 pixels

Details

This function is mostly useful for testing purpose.

Value

A stars grid with all values being zero with the dimensions matching those required for the tile specified

Examples

```
targetGrid(4, 2, 4)
targetGrid(4, 2, 4, 128)
```

Index