Package ‘leafdown’

October 13, 2022

Type Package

Title Provides Drill Down Functionality for 'leaflet' Choropleths

Version 1.2.0

Description Provides drill down functionality for 'leaflet' choropleths in 'shiny' apps.

License GPL-3

Encoding UTF-8

LazyData true

Suggests covr, testthat, knitr, rmarkdown, shinytest, dplyr,
htmltools, raster, rmapshaper, shinycssloaders

RoxygenNote 7.1.2

Depends R (>= 3.5.0)

Imports R6, leaflet, magrittr, checkmate, shiny, shinyjs

VignetteBuilder knitr, shinycssloaders

NeedsCompilation no

Author Andreas Hofheinz [cre],
Peter Gandenberger [aut]

Maintainer Andreas Hofheinz <andreas.hofheinz@outlook.com>

Repository CRAN

Date/Publication 2022-09-19 06:06:11 UTC

R topics documented:

`assert_join_map_levels_by` *Check whether the given join_map_levels_by is valid*

Description

The join_map_levels_by must be a named vector of at most one element. The columns specified in the vector must be data slots of the spdfs in the spdfs_list.

Usage

```
assert_join_map_levels_by(join_map_levels_by, spdfs_list)
```

Arguments

`join_map_levels_by`	A named vector with the columns to join the map levels by.
`spdfs_list`	A list with the spdfs of all map levels.

Value

the join_map_levels_by in the right order

`assert_spdf_list` *Check whether the given spdf_list is a valid spdf_list and has all the required params.*

Description

The spdf_list must be a list of at most two elements. All elements must be a s4 class of type SpatialPolygonsDataFrame.

Usage

```
assert_spdf_list(spdfs_list)
```

Arguments

`spdfs_list`	A list with the spdfs of all map levels

Value

TRUE if spdf_list is valid.

`check_draw_ellipsis` *Checks for undesired arguments in ellipsis in $draw_leafdown method*

Description

Checks arguments in ellipsis for undesired inputs such as 'layerId' which may collide with internal structure of leafdown and returns a "cleaned" version of the arguments by removing or redefining problematic inputs. e.g. 'layerId' is removed from arg_list when set.

Usage

```
check_draw_ellipsis(...)
```

Arguments

`...`	Additional arguments given to `leaflet::addPolygons`

Value

List containing arguments in ... as elements

`gdp_2014_admin_districts` *GPD for administrative districts of Germany for 2014.*

Description

A dataset containing the GPD (gross domestic product) for 402 administrative districts of Germany for the year 2014.

Usage

```
gdp_2014_admin_districts
```

Format

A data frame with 402 rows and 2 variables:

Admin_District Name of the administrative district

GDP_2014 GDP for the year 2014, in euro

Source

Landatlas (www.landatlas.de). Ausgabe 2018. Hrsg.: Thuenen-Institut fuer Laendliche Raeume - Braunschweig 2018.

Note that in this package we have slightly adapted some names of the administrative districts for a better match.

`gdp_2014_federal_states` *GPD for federal states of Germany for 2014.*

Description

A dataset containing the GPD (gross domestic product) for all 16 federal states of Germany for the year 2014.

Usage

```
gdp_2014_federal_states
```

Format

A data frame with 16 rows and 2 variables:

Federal_State Name of the federal state

GDP_2014 GDP for the year 2014, in euro

Source

Arbeitskreis Volkswirtschaftliche Gesamtrechnungen der Laender: `https://www.deutschlandinzahlen.de`

`Leafdown` *Leafdown R6 Class*

Description

This class acts as a wrapper around a leafdown map.

Active bindings

`curr_sel_data` A `reactiveValue` containing a data.frame with the metadata and (if available) the corresponding values of all currently selected shapes.

`curr_data` The metadata and (if available) the corresponding values of all currently displayed shapes.

`curr_map_level` Index of the current map level. This corresponds to the position of the shapes in the `spdfs_list`. (i.e The highest-level is 1, the next is 2 and so on...).

`curr_poly_ids` The ids of all polygons of the current map level.

Methods

Public methods:

- `Leafdown$new()`
- `Leafdown$draw_leafdown()`
- `Leafdown$keep_zoom()`
- `Leafdown$add_data()`
- `Leafdown$drill_down()`
- `Leafdown$drill_up()`
- `Leafdown$toggle_shape_select()`
- `Leafdown$clone()`

Method `new()`**:** Initializes the leafdown object.

Usage:

```
Leafdown$new(spdfs_list, map_output_id, input, join_map_levels_by = NULL)
```

Arguments:

`spdfs_list` A list with the spdfs of all map levels. This cannot be changed later.

`map_output_id` The id from the shiny-ui used in the `leafletOutput("<<id>>")`. Used to observe for _shape_click events.

`input` The `input` from the shiny app.

`join_map_levels_by` A named vector of length `length(spdfs_list) - 1` with the columns by which the map levels should be joined. The first element defines how the first and second map levels should be joined, the second element does the same for the second and third map levels and so on. The name of an element defines the name of the join column in the upper map level and the actual element the join column of the lower map level. By default this is set to c("GID_0" = "GID_0", "GID_1" = "GID_1", ..., "GID_n" = "GID_n"), where n is `length(spdfs_list) - 1`.

Method `draw_leafdown()`**:** Draws the leaflet map on the current map level. All unselected parents will be drawn in gray.

Usage:

```
Leafdown$draw_leafdown(...)
```

Arguments:

`...` Additional arguments given to `leaflet::addPolygons`

Method `keep_zoom()`**:** Keeps the zoom after `drill_down` and `drill_up` events.

Usage:

```
Leafdown$keep_zoom(map, input)
```

Arguments:

`map` the map output from `draw_leafdown`

`input` the input object from the shiny app

Method `add_data()`**:** Adds the data to the currently displayed shapes. This includes the meta-data AND the values to be visualized in the map.

Usage:

`Leafdown$add_data(data)`

Arguments:

`data` The new data existing of the meta-data and the values to display in the map(color)

Method `drill_down()`**:** Drills down to the lower level if:

- there is a lower level (for now there are only two levels)
- at least one shape is selected to drill down on

This will not redraw the map. Also call `add_data` to add data for the new level and then `draw_leafdown` to redraw the map on the new level.

Usage:

`Leafdown$drill_down()`

Method `drill_up()`**:** Drills up to the higher level if:

- there is a higher level (for now there are only two levels)

This will not redraw the map. Also call `add_data` to add data for the new level and then `draw_leafdown` to redraw the map on the new level.

Usage:

`Leafdown$drill_up()`

Method `toggle_shape_select()`**:** Selects the shape with the given shape id, or unselects it if it was already selected.

Usage:

`Leafdown$toggle_shape_select(shape_id)`

Arguments:

`shape_id` the id of the shape to select, has to be a character and in the current map-level.

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

Usage:

`Leafdown$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## Not run:

library(leafdown)
library(leaflet)
library(shiny)
library(dplyr)
library(shinyjs)

ger1 <- raster::getData(country = "Germany", level = 1)
ger2 <- raster::getData(country = "Germany", level = 2)
```

```
spdfs_list <- list(ger1, ger2)

ui <- shiny::fluidPage(
  useShinyjs(),
  actionButton("drill_down", "Drill Down"),
  actionButton("drill_up", "Drill Up"),
  leafletOutput("leafdown")
)

server <- function(input, output) {
  my_leafdown <- Leafdown$new(spdfs_list, "leafdown", input)
  update_leafdown <- reactiveVal(0)

  observeEvent(input$drill_down, {
    my_leafdown$drill_down()
    update_leafdown(update_leafdown() + 1)
  })

  observeEvent(input$drill_up, {
    my_leafdown$drill_up()
   update_leafdown(update_leafdown() + 1)
 })

  output$leafdown <- renderLeaflet({
    update_leafdown()
    meta_data <- my_leafdown$curr_data
    curr_map_level <- my_leafdown$curr_map_level
    if (curr_map_level == 1) {
      data <- meta_data %>%
        left_join(gdp_2014_federal_states, by = c("NAME_1" = "Federal_State"))
    } else {
      data <- meta_data %>%
        left_join(gdp_2014_admin_districts, by = c("NAME_2" = "Admin_District"))
    }

    my_leafdown$add_data(data)
    my_leafdown$draw_leafdown(
     fillColor = ~ colorNumeric("Greens", GDP_2014)(GDP_2014), weight = 2, color = "grey"
    )
  })
}

shinyApp(ui, server)


## End(Not run)
```

`us_election_counties` *Results of the 2016 US Presidential Election - County Level*

Description

A dataset containing the results of the presidential election and census data (e.g. racial makeup, unemployment)

Usage

```
us_election_counties
```

Format

A data frame with 3,143 rows and 17 total columns

State Name of the State

ST Abbreviation of the State name

County Name of the County

Votes Total number of votes cast

Republicans2016 Percent of votes for the Republican Party

Democrats2016 Percent of votes for the Democratic Party

Green2016 Percent of votes for the Green Party

Libertarians2016 Percent of votes for the Libertarian Party

TotalPopulation Total Population of the county

Unemployment Percent of unemployment

White Percentage of Whites

Black Percentage of Blacks

Hispanic Percentage of Hispanics

Asian Percentage of Asians

Amerindian Percentage of Amerindians

Other Percentage of Other Races

NAME_2 The short County name, used for matching with the map

Source

https://github.com/Deleetdk/USA.county.data

`us_election_states` *Results of the 2016 US Presidential Election - State Level*

Description

A dataset containing the results of the presidential election and census data (e.g. racial makeup, unemployment)

Usage

```
us_election_states
```

Format

A data frame with 51 rows and 15 total columns

State Name of the State

ST Abbreviation of the State name

Votes Total number of votes cast

Republicans2016 Percent of votes for the Republican Party

Democrats2016 Percent of votes for the Democratic Party

Green2016 Percent of votes for the Green Party

Libertarians2016 Percent of votes for the Libertarian Party

TotalPopulation Total Population of the county

Unemployment Percent of unemployment

White Percentage of Whites

Black Percentage of Blacks

Hispanic Percentage of Hispanics

Asian Percentage of Asians

Amerindian Percentage of Amerindians

Other Percentage of Other Races

Source

https://github.com/Deleetdk/USA.county.data

Note: The data was aggregated from the county level

Index