Package 'Rnvd3'

October 12, 2022

Type Package

Title An Incomplete Wrapper of the 'nvd3' JavaScript Library

Version 1.0.0

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Description Creates JavaScript charts with the 'nvd3' library. So far only the multibar chart, the horizontal multibar chart, the line chart and the line chart with focus are available.

License GPL-3

Encoding UTF-8

URL https://github.com/stla/Rnvd3

BugReports https://github.com/stla/Rnvd3/issues

Imports lubridate, data.table, htmlwidgets, lazyeval, viridisLite, htmltools, jsonlite, grDevices, utils

Suggests reshape2, shiny

RoxygenNote 7.1.1

NeedsCompilation no

Author Stéphane Laurent [aut, cre],
Novus Partners, Inc. [cph] ('nvd3.js' library),
Michael Bostock [cph] ('d3.js' library)

Repository CRAN

Date/Publication 2021-09-02 09:20:05 UTC

R topics documented:

---

hMultiBarChart | *Horizontal multibar chart*

Description

HTMLwidget displaying a horizontal multibar chart.

Usage

```
hMultiBarChart(
  data,
  formula,
  by,
  palette = "viridis",
  xAxisTitle = NULL,
  yAxisTitle = NULL,
  margins = list(b = 100, l = 100),
  duration = 1300,
  groupSpacing = 0.1,
  xAxisTitleDistance = 25,
  yAxisTitleDistance = -5,
  yAxisShowMaxMin = FALSE,
  yAxisTickFormat = ".0f",
  nticks = 5,
  xLabelsFontSize = "1rem",
  yLabelsFontSize = "1rem",
  showValues = FALSE,
  tooltipFormatters = list(value = NULL, header = NULL, key = NULL),
  tooltipTransitions = TRUE,
  tooltipShadow = TRUE,
  width = "100%",
  height = NULL,
  elementId = NULL
)
```

Arguments

data	dataframe containing the data used for the chart
formula	a two-sided formula like y ~ x, where "x" and "y" are two column names of data
by	string, the "by" variable; must be a column name of data
palette	this can be either the name of a viridis color palette, e.g. "viridis", "cividis" or "turbo" (see [viridis](#)), or a vector of colors, or a function that takes an integer argument (the required number of colors) and returns a character vector of colors (e.g. you can use [colorRampPalette](#))
xAxisTitle	a title for the x-axis; if NULL, the title is taken from the formula argument

yAxisTitle	a title for the y-axis; if NULL, the title is taken from the `formula` argument
margins	a named list defining the margins, with names `"t"`, `"r"`, `"b"` and `"l"`, for "top", "right", "bottom" and "left" respectively; you can specify only certain margins in the list to change just those parts
duration	duration of the transition, a number of milliseconds
groupSpacing	a number, controls the distance between groups of bars
xAxisTitleDistance	
	a number, controls the distance between the x-axis and its title
yAxisTitleDistance	
	a number, controls the distance between the y-axis and its title
yAxisShowMaxMin	
	Boolean, whether to show the min and the max on the y-axis
yAxisTickFormat	
	a d3 formatting string for the y-axis; see [d3.format](#)
nticks	integer, the number of ticks on the y-axis
xLabelsFontSize	
	a CSS measure, the font size of the labels on the x-axis
yLabelsFontSize	
	a CSS measure, the font size of the labels on the y-axis
showValues	Boolean, whether to show the values next to the bars
tooltipFormatters	
	formatters for the tooltip; each formatter must be NULL for the default formatting, otherwise a JavaScript function created with [JS](#); there are three possible formatters (see the example):

> **value** formatter for the y-value displayed in the tooltip
>
> **header** formatter for the tooltip header (this is the x-value)
>
> **key** formatter for the value of the 'by' variable

tooltipTransitions	
	Boolean, whether to style the tooltip with a fade effect
tooltipShadow	Boolean, whether to style the tooltip with a shadow
width	width of the chart container, must be a valid CSS measure
height	height of the chart container, must be a valid CSS measure
elementId	an id for the chart container; commonly useless

Value

A htmlwidget displaying a grouped/stacked bar chart.

Examples

```
library(Rnvd3)
dat <- aggregate(breaks ~ wool + tension, data = warpbreaks, mean)
levels(dat[["tension"]]) <- c("Low", "Medium", "High")
```

```
hMultiBarChart(
  dat, breaks ~ wool, "tension", yAxisShowMaxMin = TRUE,
  yAxisTitle = "Mean of breaks", yAxisTickFormat = ".01f"
)

# the axis titles are small, let's make them bigger
library(htmltools)
CSS <- HTML(
  ".nvd3 .nv-axis.nv-x text.nv-axislabel,
   .nvd3 .nv-axis.nv-y text.nv-axislabel {
     font-size: 1rem;
  }"
)
prependContent(
  hMultiBarChart(
    dat, breaks ~ wool, "tension", yAxisShowMaxMin = TRUE,
    yAxisTitle = "Mean of breaks", yAxisTickFormat = ".01f"
  ),
  tags$style(CSS)
)
```

---

lineChart   *Line chart*

Description

Create a HTML widget displaying a line chart.

Usage

```
lineChart(
  data,
  xAxisTitle = "x",
  yAxisTitle = "y",
  margins = list(l = 90),
  duration = 500,
  useInteractiveGuideline = TRUE,
  xAxisTickFormat = ".0f",
  yAxisTickFormat = ".02f",
  xLabelsFontSize = "0.75rem",
  yLabelsFontSize = "0.75rem",
  legendPosition = "top",
  interpolate = "linear",
  xRange = NULL,
  yRange = NULL,
  rightAlignYaxis = FALSE,
  tooltipFormatters = list(value = NULL, header = NULL, key = NULL),
  tooltipTransitions = TRUE,
  tooltipShadow = TRUE,
```

```
  width = "100%",
  height = NULL,
  elementId = NULL
)
```

Arguments

data	data used for the chart; it must be a list created with [lineChartData](), or a list of such lists (for multiple lines)
xAxisTitle	string, the title of the x-axis
yAxisTitle	string, the title of the y-axis
margins	a named list defining the margins, with names `"t"`, `"r"`, `"b"` and `"l"`, for for "top", "right", "bottom" and "left" respectively; you can specify only certain margins in the list to change just those parts
duration	transition duration in milliseconds
useInteractiveGuideline	
	Boolean, a guideline and synchronized tooltips
xAxisTickFormat	
	a d3 formatting string for the ticks on the x-axis; a d3 *time* formatting string if the x-values are dates, see [d3.time.format]()
yAxisTickFormat	
	a d3 formatting string for the ticks on the y-axis
xLabelsFontSize	
	a CSS measure, the font size of the labels on the x-axis
yLabelsFontSize	
	a CSS measure, the font size of the labels on the y-axis
legendPosition	string, the legend position, `"top"` or `"right"`
interpolate	interpolation type, a string among `"linear"`, `"step-before"`, `"step-after"`, `"basis"`, `"basis-open"`, `"basis-closed"`, `"bundle"`, `"cardinal"`, `"cardinal-open"`, `"cardinal-closed"`, `"monotone"`
xRange	the x-axis range, a length two vector of the same type as the x-values, or NULL to derive it from the data
yRange	the y-axis range, a numeric vector of length 2, or NULL to derive it from the data
rightAlignYaxis	
	Boolean, whether to put the y-axis on the right side instead of the left
tooltipFormatters	
	formatters for the tooltip; each formatter must be NULL for the default formatting, otherwise a JavaScript function created with [JS](); there are three possible formatters (see the example):
	value formatter for the y-value displayed in the tooltip
	header formatter for the tooltip header (this is the x-value)
	key formatter for the value of the 'by' variable
tooltipTransitions	
	Boolean, whether to style the tooltip with a fade effect

tooltipShadow	Boolean, whether to style the tooltip with a shadow
width	width of the chart container, must be a valid CSS measure
height	height of the chart container, must be a valid CSS measure
elementId	an id for the chart container, usually useless

Value

A HTML widget displaying a line chart.

Examples

```
library(Rnvd3)

dat1 <-
  lineChartData(x = ~ 1:100, y = ~ sin(1:100/10), key = "Sine wave", color = "lime")
dat2 <-
  lineChartData(x = ~ 1:100, y = ~ sin(1:100/10)*0.25 + 0.5,
                key = "Another sine wave", color = "red")
dat <- list(dat1, dat2)

lineChart(dat)

# with a date x-axis ####
dat1 <-
  lineChartData(
    x = ~ Sys.Date() + 1:100, y = ~ sin(1:100/10), key = "Sine wave", color = "lime"
  )
dat2 <-
  lineChartData(x = ~ Sys.Date() + 1:100, y = ~ sin(1:100/10)*0.25 + 0.5,
                key = "Another sine wave", color = "darkred")
dat <- list(dat1, dat2)

lineChart(
  dat,
  margins = list(t = 100, r = 100, b = 100, l = 100),
  xAxisTickFormat = "%Y-%m-%d"
)

# with a datetime x-axis

dat <- data.frame(
  x  = Sys.time() + (1:300),
  y1 = sin(1:300/10),
  y2 = sin(1:300/10)*0.25 + 0.5
)
dat1 <-
  lineChartData(x = ~x, y = ~y1, data = dat, key = "Sine wave", color = "lime")
dat2 <-
  lineChartData(x = ~x, y = ~y2, data = dat,
                key = "Another sine wave", color = "darkred")
dat12 <- list(dat1, dat2)
```

```
lineChart(
  dat12,
  margins = list(t = 100, r = 100, b = 100, l = 100),
  xAxisTickFormat = "%H:%M:%S",
  xAxisTitle = "Time", yAxisTitle = "Energy"
)
```

---

lineChartData *Line chart data*

Description

Make line chart data.

Usage

```
lineChartData(x, y, data = NULL, key, color, area = FALSE)
```

Arguments

x	a right-sided formula giving the variable on the x-axis, numeric or date type
y	a right-sided formula giving the variable on the x-axis, numeric type
data	dataframe containing the data for the chart; if not NULL, the variables passed to x and y are searched among the columns of data
key	string, the title of the line chart
color	string, the color of the line chart
area	Boolean, whether to turn the line chart into a filled area chart

Value

A list, for usage in [lineChart](#).

Note

The color can be given by the name of a R color, the name of a CSS color, e.g. "lime" or "fuchsia", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)".

lineFocusChart   *Line chart with focus*

Description

Create a HTML widget displaying a line chart with a focus tool.

Usage

```
lineFocusChart(
  data,
  xAxisTitle = "x",
  yAxisTitle = "y",
  margins = list(l = 90),
  duration = 500,
  useInteractiveGuideline = FALSE,
  xAxisTickFormat = ".0f",
  yAxisTickFormat = ".02f",
  xLabelsFontSize = "0.75rem",
  yLabelsFontSize = "0.75rem",
  legendPosition = "top",
  interpolate = "linear",
  xRange = NULL,
  yRange = NULL,
  rightAlignYaxis = FALSE,
  tooltipFormatters = list(value = NULL, header = NULL, key = NULL),
  tooltipTransitions = TRUE,
  tooltipShadow = TRUE,
  width = "100%",
  height = NULL,
  elementId = NULL
)
```

Arguments

data	data used for the chart; it must be a list created with [lineChartData](#), or a list of such lists (for multiple lines)
xAxisTitle	string, the title of the x-axis
yAxisTitle	string, the title of the y-axis
margins	a named list defining the margins, with names "t", "r", "b" and "l", for "top", "right", "bottom" and "left" respectively; you can specify only certain margins in the list to change just those parts
duration	transition duration in milliseconds
useInteractiveGuideline	
	Boolean, a guideline and synchronized tooltips

xAxisTickFormat

> a d3 formatting string for the ticks on the x-axis; a d3 *time* formatting string if the x-values are dates, see d3.time.format

yAxisTickFormat

> a d3 formatting string for the ticks on the y-axis

xLabelsFontSize

> a CSS measure, the font size of the labels on the x-axis

yLabelsFontSize

> a CSS measure, the font size of the labels on the y-axis

legendPosition   string, the legend position, `"top"` or `"right"`

interpolate   interpolation type, a string among `"linear"`, `"step-before"`, `"step-after"`, `"basis"`, `"basis-open"`, `"basis-closed"`, `"bundle"`, `"cardinal"`, `"cardinal-open"`, `"cardinal-closed"`, `"monotone"`

xRange   the x-axis range, a length two vector of the same type as the x-values, or `NULL` to derive it from the data

yRange   the y-axis range, a numeric vector of length 2, or `NULL` to derive it from the data

rightAlignYaxis

> Boolean, whether to put the y-axis on the right side instead of the left

tooltipFormatters

> formatters for the tooltip; each formatter must be `NULL` for the default formatting, otherwise a JavaScript function created with [JS](JS); there are three possible formatters (see the example):
>
> **value** formatter for the y-value displayed in the tooltip
>
> **header** formatter for the tooltip header (this is the x-value)
>
> **key** formatter for the value of the 'by' variable

tooltipTransitions

> Boolean, whether to style the tooltip with a fade effect

tooltipShadow   Boolean, whether to style the tooltip with a shadow

width   width of the chart container, must be a valid CSS measure

height   height of the chart container, must be a valid CSS measure

elementId   an id for the chart container, usually useless

Value

A HTML widget displaying a line chart with a focus tool.

Examples

```
library(Rnvd3)

dat1 <-
  lineChartData(x = ~ 1:100, y = ~ sin(1:100/10), key = "Sine wave", color = "lime")
dat2 <-
  lineChartData(x = ~ 1:100, y = ~ sin(1:100/10)*0.25 + 0.5,
                key = "Another sine wave", color = "red")
dat <- list(dat1, dat2)

lineFocusChart(dat)
```

multiBarChart	*Multibar chart*

Description

HTMLwidget displaying a multibar chart.

Usage

```
multiBarChart(
  data,
  formula,
  by,
  palette = "viridis",
  xAxisTitle = NULL,
  yAxisTitle = NULL,
  margins = list(b = 100, l = 70),
  duration = 1300,
  rotateLabels = 0,
  groupSpacing = 0.1,
  xAxisTitleDistance = 35,
  yAxisTitleDistance = -5,
  yAxisShowMaxMin = FALSE,
  yAxisTickFormat = ".0f",
  xLabelsFontSize = "1rem",
  yLabelsFontSize = "1rem",
  rightAlignYaxis = FALSE,
  reduceXticks = FALSE,
  staggerLabels = FALSE,
  wrapLabels = FALSE,
  useInteractiveGuideline = FALSE,
  tooltipFormatters = list(value = NULL, header = NULL, key = NULL),
  tooltipTransitions = TRUE,
  tooltipShadow = TRUE,
  radioButtonMode = FALSE,
  legendTitle = NULL,
  legendHjust = -20,
  width = "100%",
  height = NULL,
  elementId = NULL
)
```

Arguments

data	dataframe used for the chart
formula	a two-sided formula like y ~ x, where "x" and "y" are two column names of data

by	string, the "by" variable; must be a column name of `data`
palette	this can be either the name of a viridis color palette, e.g. `"viridis"`, `"cividis"` or `"turbo"` (see [viridis](#)), or a vector of colors, or a function that takes an integer argument (the required number of colors) and returns a character vector of colors (e.g. you can use [colorRampPalette](#))
xAxisTitle	a title for the x-axis; if NULL, the title is taken from the `formula` argument
yAxisTitle	a title for the y-axis; if NULL, the title is taken from the `formula` argument
margins	a named list defining the margins, with names `"t"`, `"r"`, `"b"` and `"l"`, for "top", "right", "bottom" and "left" respectively; you can specify only certain margins in the list to change just those parts
duration	duration of the transition, a number of milliseconds
rotateLabels	a number, the angle of rotation of the labels of the x-axis (in degrees)
groupSpacing	a number, controls the distance between groups of bars
xAxisTitleDistance	
	a number, controls the distance between the x-axis and its title
yAxisTitleDistance	
	a number, controls the distance between the y-axis and its title
yAxisShowMaxMin	
	Boolean, whether to show the min and the max on the y-axis
yAxisTickFormat	
	a d3 formatting string for the y-axis; see [d3.format](#)
xLabelsFontSize	
	a CSS measure, the font size of the labels on the x-axis
yLabelsFontSize	
	a CSS measure, the font size of the labels on the y-axis
rightAlignYaxis	
	Boolean, whether to put the y-axis on the right side instead of the left
reduceXticks	Boolean, whether to reduce the ticks on the x-axis so that the x-labels are less likely to overlap
staggerLabels	Boolean, whether to make the x-labels stagger at different distances from the axis so they're less likely to overlap
wrapLabels	Boolean, whether to split long x-labels by new lines in order to prevent overlapping
useInteractiveGuideline	
	Boolean, other kind of tooltips: sets the chart to use a guideline and floating tooltip instead of requiring the user to hover over specific hotspots
tooltipFormatters	
	formatters for the tooltip; each formatter must be NULL for the default formatting, otherwise a JavaScript function created with [JS](#); there are three possible formatters (see the example):

value formatter for the y-value displayed in the tooltip

header formatter for the tooltip header (this is the x-value)

key formatter for the value of the 'by' variable

tooltipTransitions	
	Boolean, whether to style the tooltip with a fade effect
tooltipShadow	Boolean, whether to style the tooltip with a shadow
radioButtonMode	
	Boolean, whether to authorize only one selection in the legend (i.e. only one level of the 'by' variable)
legendTitle	a title for the legend, or NULL for no title
legendHjust	horizontal adjustment of the legend title
width	width of the chart container, must be a valid CSS measure
height	height of the chart container, must be a valid CSS measure
elementId	an id for the chart container; commonly useless

Value

A htmlwidget displaying a grouped/stacked bar chart.

Note

In Shiny, you can style the axis titles with the help of CSS; see the shiny example. It is also possible outside of Shiny; see the second example below, where we set the CSS with the help of prependContent.

Examples

```
library(Rnvd3)
# in this example we use the tooltip formatters for styling only; we could
# achieve the same result with the help of CSS
dat <- reshape2::melt(
  apply(HairEyeColor, c(1, 2), sum), value.name = "Count"
)
multiBarChart(
  dat, Count ~ Eye, "Hair",
  tooltipFormatters = list(
    value = JS(
      "function(x){",
      "  return '<span style=\"color:red;\">' + x + '</span>';",
      "}"
    ),
    header = JS(
      "function(x){",
      "  return '<span style=\"color:green;\">' + x + '</span>';",
      "}"
    ),
    key = JS(
      "function(x){",
      "  return '<i style=\"color:blue;\">' + x + '</i>';",
      "}"
    )
  )
```

```
)

# style axis titles with CSS ####
library(htmltools)

CSS <- HTML(
  ".nvd3 .nv-axis.nv-x text.nv-axislabel,
   .nvd3 .nv-axis.nv-y text.nv-axislabel {
     font-size: 2rem;
     fill: red;
  }"
)

widget <- multiBarChart(
  dat, Count ~ Eye, "Hair", palette = "turbo"
)
prependContent(
  widget,
  tags$style(CSS)
)
```

Rnvd3-imports   *Objects imported from other packages*

Description

These objects are imported from other packages. Follow the links to their documentation: JS,
saveWidget, prependContent.

---

rnvd3-shiny   *Shiny bindings for rnvd3*

Description

Output and render functions for using rnvd3 widgets within Shiny applications and interactive Rmd
documents.

Usage

```
rnvd3Output(outputId, width = "100%", height = "400px")

renderRnvd3(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

`outputId`	output variable to read from
`width, height`	must be a valid CSS unit (like `'100%'`, `'400px'`, `'auto'`) or a number, which will be coerced to a string and have `'px'` appended
`expr`	an expression that generates a `rnvd3` widget
`env`	the environment in which to evaluate `expr`
`quoted`	is expr a quoted expression (with `quote()`)

Value

`rnvd3Output` returns an output element that can be included in a Shiny UI definition, and `renderRnvd3` returns a `shiny.render.function` object that can be included in a Shiny server definition.

Examples

```
library(Rnvd3)
library(shiny)

dat <- reshape2::melt(
  apply(HairEyeColor, c(1, 2), sum), value.name = "Count"
)

CSS <- HTML(
  "body {
    overflow: overlay;
  }
  /* style axis titles */
  .nvd3 .nv-axis.nv-x text.nv-axislabel,
   .nvd3 .nv-axis.nv-y text.nv-axislabel {
     font-size: 3rem;
     fill: red;
  }
  /* style the tooltip */
  .nvtooltip .value {
    color: red;
  }
  .nvtooltip .x-value {
    color: green;
  }
  .nvtooltip .key {
    color: blue;
    font-style: italic;
  }
  "
)

ui <- fluidPage(
  tags$head(tags$style(CSS)),
  br(),
  fluidRow(
```

```
    column(
      9,
      rnvd3Output("mychart", width = "100%", height = "500px")
    ),
    column(
      3,
      tags$h3("Chart state:"),
      verbatimTextOutput("state")
    )
  )
)

server <- function(input, output, session){

  output[["mychart"]] <- renderRnvd3({
    multiBarChart(
      dat, Count ~ Eye, "Hair", palette = "viridis",
      xLabelsFontSize = "2rem", yLabelsFontSize = "2rem"
    )
  })

  output[["state"]] <- renderPrint({
    input[["mychart_state"]]
  })

}

if(interactive()){
  shinyApp(ui, server)
}
```

Index