Package ‘highcharter’

October 13, 2022

Type Package
Version 0.9.4
Title A Wrapper for the 'Highcharts' Library
Description A wrapper for the 'Highcharts' library including shortcut functions to plot R objects. 'Highcharts' <https://www.highcharts.com/> is a charting library offering numerous chart types with a simple configuration syntax.
        https://github.com/jbkunst/highcharter
BugReports https://github.com/jbkunst/highcharter/issues
License MIT + file LICENSE
RoxygenNote 7.1.2
Encoding UTF-8
Depends R (>= 2.10)
Imports htmlwidgets, magrittr, purrr, rlist, assertthat, zoo, dplyr
        (>= 1.0.0), tibble (>= 1.1), stringr (>= 1.3.0), broom, xts,
        quantmod, tidyr, htmltools, jsonlite, igraph, lubridate, yaml,
        rlang (>= 0.1.1), rjson
Suggests knitr, rmarkdown, survival, ggplot2, httr, viridisLite,
        shiny, MASS, gapminder, forecast, geojsonio, testthat, covr,
        spelling
LazyData true
Language en-US
NeedsCompilation no
Author Joshua Kunst [aut, cre],
        Nuno Agostinho [ctb] (hchart.survfit, densities and
        hc_add_series_scatter),
        Danton Noriega [ctb] (hcaes_),
        Martin John Hadley [ctb] (hc_add_event_point improvement),
        Eduardo Flores [ctb] (First version hc_add_series_df_tidy),
        1
Dean Kilfoyle [ctb] (First version hc_add_series_boxplot),
Adline Dsilva [ctb] (First version Matrix heatmap),
Kamil Slowikowski [ctb] (<https://orcid.org/0000-0002-2843-6370>),
Christian Minich [ctb] (hcaes mutate_mapping improvement),
Jonathan Armond [ctb] (mutate_mapping bugfix),
David Breuer [ctb] (download_map_data quiet parameter),
Mauricio Vargas [ctb] (tests and gh-actions),
Michael Yan [ctb] (Motivational example for treemap/sunburst data helper),
Bart Oortwijn [ctb] (rjson option, hc_add_yAxis, and GH issues collaborator),
Paul Campbell [ctb] (additional proxy methods)

Maintainer  Joshua Kunst <jbkunst@gmail.com>
Repository  CRAN
Date/Publication  2022-01-03 16:40:05 UTC

R topics documented:

citytemp ............................................................... 5
citytemp_long ......................................................... 6
colorize ................................................................. 6
color_classes ......................................................... 7
color_stops ............................................................ 7
data_to_boxplot ..................................................... 8
data_to_hierarchical ................................................. 9
data_to_sankey ....................................................... 10
datetime_to_timestamp ............................................. 10
df_to_annotations_labels ........................................ 11
download_map_data .................................................. 11
export_hc .............................................................. 12
favorite_bars ....................................................... 13
favorite_pies ....................................................... 14
get_data_from_map ................................................... 14
get_hc_series_from_df .............................................. 15
globaltemp ........................................................... 15
hcaes ................................................................. 16
hcaes_string ........................................................ 16
hcboxplot ........................................................... 17
hchart ................................................................. 17
hchart.survfit ........................................................ 18
hciconarray .......................................................... 19
hcmap ................................................................. 20
hcparcords ............................................................ 21
hcpxy_add_point ..................................................... 22
hcpxy_add_series .................................................... 22
hcpxy_loading ........................................................ 23
hcpxy_redraw ........................................................ 23
<table>
<thead>
<tr>
<th>Topics Documented:</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>hcpxy_remove_point</td>
<td>24</td>
</tr>
<tr>
<td>hcpxy_remove_series</td>
<td>24</td>
</tr>
<tr>
<td>hcpxy_set_data</td>
<td>25</td>
</tr>
<tr>
<td>hcpxy_update</td>
<td>26</td>
</tr>
<tr>
<td>hcpxy_update_point</td>
<td>26</td>
</tr>
<tr>
<td>hcpxy_update_series</td>
<td>27</td>
</tr>
<tr>
<td>hcspark</td>
<td>27</td>
</tr>
<tr>
<td>hctreemap</td>
<td>28</td>
</tr>
<tr>
<td>hctreemap2</td>
<td>29</td>
</tr>
<tr>
<td>hc_add_annotation</td>
<td>30</td>
</tr>
<tr>
<td>hc_add_dependency</td>
<td>31</td>
</tr>
<tr>
<td>hc_add_dependency_fa</td>
<td>31</td>
</tr>
<tr>
<td>hc_add_event_point</td>
<td>32</td>
</tr>
<tr>
<td>hc_add_series</td>
<td>33</td>
</tr>
<tr>
<td>hc_add_series.character</td>
<td>33</td>
</tr>
<tr>
<td>hc_add_series.data.frame</td>
<td>34</td>
</tr>
<tr>
<td>hc_add_series.density</td>
<td>34</td>
</tr>
<tr>
<td>hc_add_series.forecast</td>
<td>35</td>
</tr>
<tr>
<td>hc_add_series.geo_json</td>
<td>35</td>
</tr>
<tr>
<td>hc_add_series.lm</td>
<td>36</td>
</tr>
<tr>
<td>hc_add_series.numeric</td>
<td>37</td>
</tr>
<tr>
<td>hc_add_series.ts</td>
<td>37</td>
</tr>
<tr>
<td>hc_add_series.xts</td>
<td>38</td>
</tr>
<tr>
<td>hc_add_series_list</td>
<td>38</td>
</tr>
<tr>
<td>hc_add_series_map</td>
<td>39</td>
</tr>
<tr>
<td>hc_add_theme</td>
<td>40</td>
</tr>
<tr>
<td>hc_annotations</td>
<td>41</td>
</tr>
<tr>
<td>hc_boost</td>
<td>42</td>
</tr>
<tr>
<td>hc_caption</td>
<td>44</td>
</tr>
<tr>
<td>hc_chart</td>
<td>45</td>
</tr>
<tr>
<td>hc_colorAxis</td>
<td>46</td>
</tr>
<tr>
<td>hc_colors</td>
<td>48</td>
</tr>
<tr>
<td>hc_credits</td>
<td>49</td>
</tr>
<tr>
<td>hc_drilldown</td>
<td>49</td>
</tr>
<tr>
<td>hc_elementId</td>
<td>51</td>
</tr>
<tr>
<td>hc_exporting</td>
<td>51</td>
</tr>
<tr>
<td>hc_labels</td>
<td>52</td>
</tr>
<tr>
<td>hc_legend</td>
<td>53</td>
</tr>
<tr>
<td>hc_loading</td>
<td>54</td>
</tr>
<tr>
<td>hc_mapNavigation</td>
<td>54</td>
</tr>
<tr>
<td>hc_motion</td>
<td>55</td>
</tr>
<tr>
<td>hc_navigator</td>
<td>55</td>
</tr>
<tr>
<td>hc_pane</td>
<td>56</td>
</tr>
<tr>
<td>hc_plotOptions</td>
<td>58</td>
</tr>
<tr>
<td>hc_rangeSelector</td>
<td>59</td>
</tr>
<tr>
<td>hc_responsive</td>
<td>59</td>
</tr>
<tr>
<td>hc_rm_series</td>
<td>60</td>
</tr>
<tr>
<td>hc_scrollbar</td>
<td>61</td>
</tr>
</tbody>
</table>
R topics documented:

hc_series ........................................ 62
hc_size ........................................ 62
hc_subtitle ..................................... 63
hc_theme .......................................... 64
hc_theme_538 .................................. 65
hc_theme_alone .................................. 66
hc_theme_bloom .................................. 66
hc_theme_chalk .................................. 67
hc_theme_darkunica .............................. 67
hc_theme_db ..................................... 68
hc_theme_economist .............................. 68
hc_theme_elementary .............................. 69
hc_theme_ffx ..................................... 69
hc_theme_flat .................................... 70
hc_theme_flatdark ............................... 70
hc_theme_ft ...................................... 71
hc_theme_ggplot2 ................................ 71
hc_theme_google ................................ 72
hc_theme_gridlight ............................... 72
hc_theme_handdrawn .............................. 73
hc_theme_hcrt .................................... 73
hc_theme_merge .................................. 74
hc_theme_monokai ................................ 74
hc_theme_null .................................... 75
hc_theme_sandsignika ............................ 75
hc_theme_smpl ................................... 76
hc_theme_sparkline .............................. 76
hc_theme_superheroes ............................ 77
hc_theme_tufte ................................... 77
hc_title .......................................... 78
hc_tooltip ....................................... 78
hc_xAxis ......................................... 79
hc_yAxis ......................................... 80
hc_yAxis_multiples .............................. 81
hc_zAxis .......................................... 82
hex_to_rgba ..................................... 84
highchart ........................................ 84
highchart2 ....................................... 85
highcharter ...................................... 86
highcharter-exports ............................. 86
highchartOutput .................................. 87
highchartProxy ................................... 87
highcharts_demo .................................. 88
hw_grid ............................................ 88
is.hexcolor ...................................... 89
is.highchart ..................................... 89
list_parse ........................................ 90
mountains_panorama .............................. 90
citytemp

City temperatures from a year in wide format

Description

This data comes from the https://www.highcharts.com/ examples.

Usage

citytemp

Format

A data frame with 12 observations and 5 variables.

Variables

- month: The months.
- tokyo: Tokyo’s temperatures.
- new_york: New York’s temperatures.
- berlin: Berlin’s temperatures.
- london: London’s temperatures.
citytemp_long  

City temperatures from a year in long format

Description

This data comes from the [https://www.highcharts.com/](https://www.highcharts.com/) examples.

Usage

citytemp_long

Format

A data frame with 36 observations and 3 variables.

Variables

- `month`: The months.
- `city`: City.
- `temp`: Temperatures.

---

colorize  

Create vector of color from vector

Description

Create vector of color from vector

Usage

colorize(x, colors = c("#440154", "#21908C", "#FDE725"))

Arguments

- `x`: A numeric, character or factor object.
- `colors`: A character string of colors (ordered) to colorize `x`

Examples

```r
colorize(runif(10))
colorize(LETTERS[rbinom(20, 5, 0.5)], c("#FF0000", "#00FFFF"))
```
**color_classes**

*Function to create dataClasses argument in hc_colorAxis*

**Description**

Function to create dataClasses argument in hc_colorAxis

**Usage**

```r
color_classes(breaks = NULL, colors = c("#440154", "#21908C", "#FDE725"))
```

**Arguments**

- `breaks` A numeric vector
- `colors` A character string of colors (ordered)

**Examples**

```r
color_classes(c(0, 10, 20, 50))
```

---

**color_stops**

*Function to create stops argument in hc_colorAxis*

**Description**

Function to create stops argument in hc_colorAxis

**Usage**

```r
color_stops(n = 10, colors = c("#440154", "#21908C", "#FDE725"))
```

**Arguments**

- `n` A numeric indicating how much quantiles generate.
- `colors` A character string of colors (ordered)

**Examples**

```r
color_stops(5)
```
**data_to_boxplot**  

Helper to transform data frame for boxplot highcharts format

### Description

Helper to transform data frame for boxplot highcharts format

### Usage

```r
data_to_boxplot(
  data, 
  variable, 
  group_var = NULL, 
  group_var2 = NULL, 
  add_outliers = FALSE, 
  ...
)
```

### Arguments

- **data**: The data frame containing variables.
- **variable**: The variable to calculate the box plot data.
- **group_var**: A variable to split calculation
- **group_var2**: A second variable to create separate series.
- **add_outliers**: A logical value indicating if outliers series should be calculated. Default to FALSE.
- **...**: Arguments defined in [https://api.highcharts.com/highcharts/plotOptions.

### Examples

```r
data(pokemon)

dat <- data_to_boxplot(pokemon, height)

highchart() %>%
  hc_xAxis(type = "category") %>%
  hc_add_series_list(dat)

dat <- data_to_boxplot(pokemon, height, type_1, name = "height in meters")

highchart() %>%
  hc_xAxis(type = "category") %>%
  hc_add_series_list(dat)

# Not run:
```
**data_to_hierarchical**

Helper to transform data frame for treemap/sunburst highcharts format

**Description**

Helper to transform data frame for treemap/sunburst highcharts format

**Usage**

```r
data_to_hierarchical(
  data,
  group_vars,
  size_var,
  colors = getOption("highchartr.color_palette")
)
```

**Arguments**

- **data**: data frame containing variables to organize each level of the treemap.
- **group_vars**: Variables to generate treemap levels.
- **size_var**: Variable to aggregate.
- **colors**: Color to chart every item in the first level.

**Examples**

```r
## Not run:
library(dplyr)
data(gapminder, package = "gapminder")
gapminder_2007 <- gapminder::gapminder %>%
  filter(year == max(year)) %>%
  mutate(pop_mm = round(pop / 1e6))
dout <- data_to_hierarchical(gapminder_2007, c(continent, country), pop_mm)
hchart(dout, type = "sunburst")
hchart(dout, type = "treemap")
## End(Not run)
```
data_to_sankey  

Helper to transform data frame for sankey highcharts format

Description
Helper to transform data frame for sankey highcharts format

Usage

data_to_sankey(data = NULL)

Arguments

data  A data frame

Examples

## Not run:
library(dplyr)
data(diamonds, package = "ggplot2")
diamonds2 <- select(diamonds, cut, color, clarity)
data_to_sankey(diamonds2)
hchart(data_to_sankey(diamonds2), "sankey", name = "diamonds")
## End(Not run)

datetime_to_timestamp  Date to timestamps

Description
Turn a date time vector to timestamp format

Usage

datetime_to_timestamp(dt)
dt_tstp(dt)

dt  Date or datetime vector
df_to_annotations_labels

Examples

datetime_to_timestamp(
  as.Date(c("2015-05-08", "2015-09-12"),
    format = "%Y-%m-%d"
  )
)

df_to_annotations_labels

Function to create annotations arguments from a data frame

Description

Function to create annotations arguments from a data frame

Usage

df_to_annotations_labels(df, xAxis = 0, yAxis = 0)

Arguments

df                      A data frame with x, y and text columns names.
xAxis                   Index (js 0-based) of the x axis to put the annotations.
yAxis                   Index (js 0-based) of the y axis to put the annotations.

Examples

df <- data.frame(text = c("hi", "bye"), x = c(0, 1), y = c(1, 0))
df_to_annotations_labels(df)

download_map_data

Helper function to download the map data form a url

Description

The urls are listed in https://code.highcharts.com/mapdata/.

Usage

download_map_data(url = "custom/world.js", showinfo = FALSE, quiet = FALSE)
export_hc

Arguments

url  The map’s url.
showinfo  Show the properties of the downloaded map to know how are the keys to add data in hcmap.
quiet  Boolean parameter to turn off download messages (on by default).

See Also

hcmap

Examples

```r
## Not run:
mpdta <- download_map_data("https://code.highcharts.com/mapdata/countries/us/us-ca-all.js")
mpdta <- download_map_data("https://code.highcharts.com/mapdata/countries/us/us-ca-all.js",
  quiet = TRUE
)
str(mpdta, 1)
## End(Not run)
```

---

export_hc  

Function to export js file the configuration options

Description

Function to export js file the configuration options

Usage

```r
export_hc(hc, filename = NULL, as = "is", name = NULL)
```

Arguments

hc  A Highcharts object.
filename  String of the exported file.
as  String to define how to save the configuration options. One of ‘is’, ‘container’, ‘variable’.
name  A variable used to put as name of the generated object if as is ‘variable’ and the css/js selector if as is container.
**favorite_bars**  

**Examples**

```r
fn <- "function(){
  console.log('Category: ' + this.category);
  alert('Category: ' + this.category);
}
"

hc <- highcharts_demo() %>%
  hc_plotOptions{
    series = list(
      cursor = "pointer",
      point = list(
        events = list(
          click = JS(fn)
        )
      )
    )
  }

## Not run:
export_hc(hc, filename = "~/hc_is.js", as = "is")
export_hc(hc, filename = "~/hc_vr.js", as = "variable", name = "objectname")
export_hc(hc, filename = "~/hc_ct.js", as = "container", name = "#selectorid")

## End(Not run)
```

<table>
<thead>
<tr>
<th>favorite_bars</th>
<th>Marshall's Favorite Bars</th>
</tr>
</thead>
</table>

**Description**

Data from *How I met Your Mother: Marshall's Favorite Bars*.

**Usage**

```r
favorite_bars
```

**Format**

A data frame with 5 observations and 2 variables.

**Variables**

- **bar**: Bar's name.
- **percent**: In percentage of awesomeness
**favorite_pies**  
*Marshall’s Favorite Pies*

### Description

Data from *How I met Your Mother: Marshall’s Favorite Pies*

### Usage

```
favorite_pies
```

### Format

A data frame with 5 observations and 2 variables.

### Variables

- **pie**: Bar’s name.
- **percent**: In percentage of tastiness

---

**get_data_from_map**  
*Helper function to get the data inside the map data The urls are listed in [https://code.highcharts.com/mapdata/](https://code.highcharts.com/mapdata/).*

### Description

Helper function to get the data inside the map data The urls are listed in [https://code.highcharts.com/mapdata/](https://code.highcharts.com/mapdata/).

### Usage

```
get_data_from_map(mapdata)
```

### Arguments

- **mapdata**: A list obtained from `download_map_data`.

### See Also

`download_map_data`

### Examples

```r
dta <- download_map_data("https://code.highcharts.com/mapdata/countries/us/us-ca-all.js")
get_data_from_map(dta)
```
get_hc_series_from_df

Auxiliar function to get series and options from tidy frame for hchart.data.frame

Description

This function is used in hchart.data.frame.

Usage

get_hc_series_from_df(data, type = NULL, ...)

Arguments

data A data.frame object.
type The type of chart. Possible values are line, scatter, point, column.
... Aesthetic mappings as x y group color low high.

Examples

highcharter:::get_hc_series_from_df(iris, type = "point", x = Sepal.Width)

globaltemp

Description

Temperature information by years.

Usage

globaltemp

Format

A data frame with 1992 observations and 4 variables.

Variables

- date: Date.
- lower: Minimum temperature.
- median: Median temperature.
- upper: Maximum temperature.
hcaes

Define aesthetic mappings. Similar in spirit to ggplot2::aes

Usage

hcaes(x, y, ...)

Arguments

x, y, ... List of name value pairs giving aesthetics to map to variables. The names for x and y aesthetics are typically omitted because they are so common; all other aesthetics must be named.

Examples

hcaes(x = xval, color = colorvar, group = grvar)

hcaes_string

Define aesthetic mappings using strings. Similar in spirit to ggplot2::aes_string

Usage

hcaes_string(x, y, ...)

Arguments

x, y, ... List of name value pairs giving aesthetics to map to variables. The names for x and y aesthetics are typically omitted because they are so common; all other aesthetics must be named.
Examples

hchart(mtcars, "point", hcaes_string("hp", "mpg", group = "cyl"))

hcaes_string(x = "xval", color = "colorvar", group = "grvar")

----------

hcboxplot

Shortcut to make a boxplot

Description

Shortcut to make a boxplot

Usage

hcboxplot(x = NULL, var = NULL, var2 = NULL, outliers = TRUE, ...)

Arguments

x A numeric vector.
var A string vector same length of x.
var2 A string vector same length of x.
outliers A boolean value to show or not the outliers.
... Additional arguments for the data series https://api.highcharts.com/highcharts/series.

Examples

## Not run:
hcboxplot(x = iris$Sepal.Length, var = iris$Species, color = "red")

## End(Not run)

----------

hchart

Create a highchart object from a particular data type

Description

hchart uses highchart to draw a particular plot for an object of a particular class in a single command. This defines the S3 generic that other classes and packages can extend.

Usage

hchart(object, ...)


Arguments

- object: A R object.

Details

Run methods(hchart) to see what objects are supported.

hchart.survfit  
Plot survival curves using Highcharts

Description

Plot survival curves using Highcharts

Usage

```r
## S3 method for class 'survfit'
hchart(
  object,
  ..., 
  fun = NULL,
  markTimes = TRUE,
  symbol = "plus",
  markerColor = "black",
  ranges = FALSE,
  rangesOpacity = 0.3
)
```

Arguments

- object: A survfit object as returned from the survfit function
- ...: Extra parameters to pass to hc_add_series function
- fun: Name of function or function used to transform the survival curve: log will put y axis on log scale, event plots cumulative events (f(y) = 1-y), cumhaz plots the cumulative hazard function (f(y) = -log(y)), and cloglog creates a complimentary log-log survival plot (f(y) = log(-log(y)) along with log scale for the x-axis.
- markTimes: Label curves marked at each censoring time? TRUE by default
- symbol: Symbol to use as marker (plus sign by default)
- markerColor: Color of the marker ("black" by default); use NULL to use the respective color of each series
- ranges: Plot interval ranges? FALSE by default
- rangesOpacity: Opacity of the interval ranges (0.3 by default)
Value

Highcharts object to plot survival curves

Examples

# Plot Kaplan-Meier curves
require("survival")
leukemia.surv <- survfit(Surv(time, status) ~ x, data = aml)
hchart(leukemia.surv)

# Plot the cumulative hazard function
lsurv2 <- survfit(Surv(time, status) ~ x, aml, type = "fleming")
hchart(lsurv2, fun = "cumhaz")

# Plot the fit of a Cox proportional hazards regression model
fit <- coxph(Surv(futime, fustat) ~ age, data = ovarian)
ovarian.surv <- survfit(fit, newdata = data.frame(age = 60))
hchart(ovarian.surv, ranges = TRUE)

---

**hciconarray**

**Shortcut to make icon arrays charts**

Description

Shortcut to make icon arrays charts

Usage

hciconarray(labels, counts, rows = NULL, icons = NULL, size = 4, ...)

Arguments

- labels: A character vector
- counts: A integer vector
- rows: A integer to set
- icons: A character vector same length (or length 1) as labels
- size: Font size
**hcmap**  
*Shortcut for create map from* https://code.highcharts.com/mapdata/collection.*

**Description**

Shortcut for create map from https://code.highcharts.com/mapdata/collection.

**Usage**

```r
hcmap(
  map = "custom/world",
  download_map_data = getOption("highcharter.download_map_data"),
  data = NULL,
  value = NULL,
  joinBy = NULL,
  ...
)
```

**Arguments**

- **map**
  String indicating what map to chart, a list from https://code.highcharts.com/mapdata/. See examples.

- **download_map_data**
  A logical value whether to download (add as a dependency) the map. Default TRUE via getOption("highcharter.download_map_data").

- **data**
  Optional data to make a choropleth, in case of use the joinBy and value are needed.

- **value**
  A string value with the name of the variable to chart.

- **joinBy**
  What property to join the map and df.

- **...**

**Examples**

```r
options(highcharter.download_map_data = TRUE)

# hcmap(nullColor = "DADADA")
# hcmap(nullColor = "DADADA", download_map_data = FALSE)

require(dplyr)
data("USArrests", package = "datasets")
USArrests <- mutate(USArrests, "woe-name" = rownames(USArrests))

# hcmap(
#   map = "countries/us/us-all", data = USArrests,
```
hcparcords

Shortcut to create parallel coordinates

Description

Shortcut to create parallel coordinates

Usage

hcparcords(df, ...)

Arguments

df
A data frame object.

... Additional shared arguments for the data series (https://api.highcharts.com/highcharts/series) for the hchar.data.frame function.

Examples

require(viridisLite)
n <- 15
hcparcords(head(mtcars, n), color = hex_to_rgba(magma(n), 0.5))

require(dplyr)
data(iris)
set.seed(123)

iris <- sample_n(iris, 60)

hcparcords(iris, color = colorize(iris$Species))
Description
Add point to a series of a higchartProxy object

Usage
hcpxy_add_point(
  proxy,
  id = NULL,
  point,
  redraw = TRUE,
  shift = FALSE,
  animation = TRUE
)

Arguments
proxy A higchartProxy object.
id A character vector indicating the id of the series to update.
point The point options. If options is a single number, a point with that y value is appended to the series. If it is an list, it will be interpreted as x and y values respectively. If it is an object, advanced options as outlined under series.data are applied
redraw Whether to redraw the chart after the point is added. When adding more than one point, it is highly recommended that the redraw option be set to false, and instead Highcharts.Chart#redraw is explicitly called after the adding of points is finished. Otherwise, the chart will redraw after adding each point.
shift If TRUE, a point is shifted off the start of the series as one is appended to the end.
animation Whether to apply animation, and optionally animation configuration.

Description
Add data to higchartProxy element

Usage
hcpxy_add_series(proxy, data = NULL, ...)

**hcpxy_loading**

**Arguments**

- **proxy**: A higchartProxy object.
- **data**: An R object supported by `hc_add_series` like data frame, ts, etc.
- **...**: Arguments defined in [https://api.highcharts.com/highcharts/plotOptions.series](https://api.highcharts.com/highcharts/plotOptions.series).

**hcpxy_loading**  
*Show or hide loading text for a higchartProxy object*

**Description**

Show or hide loading text for a higchartProxy object

**Usage**

```r
ehcpxy_loading(proxy, action = "show")
```

**Arguments**

- **proxy**: A higchartProxy object.
- **action**: Single-element character vector indicating to "show" or "hide" the loading text defined in `lang` options.

**hcpxy_redraw**  
*Redraw a higchartProxy object*

**Description**

Redraw a higchartProxy object

**Usage**

```r
ehcpxy_redraw(proxy)
```

**Arguments**

- **proxy**: A higchartProxy object.
**hcpxy_remove_point**  
*Remove point to a series of a higchartProxy object*

**Description**
Remove point to a series of a higchartProxy object

**Usage**
```
hcpxy_remove_point(proxy, id = NULL, i = NULL, redraw = TRUE)
```

**Arguments**
- **proxy**: A higchartProxy object.
- **id**: A character vector indicating the id of the series to update.
- **i**: The index of the point in the data array. Remember js is 0 based index.
- **redraw**: Whether to redraw the chart after the point is added. When adding more than one point, it is highly recommended that the redraw option be set to false, and instead Highcharts.Chart#redraw is explicitly called after the adding of points is finished. Otherwise, the chart will redraw after adding each point.

**hcpxy_remove_series**  
*Remove series to higchartProxy element*

**Description**
Remove series to higchartProxy element

**Usage**
```
hcpxy_remove_series(proxy, id = NULL, all = FALSE)
```

**Arguments**
- **proxy**: A higchartProxy object.
- **id**: A character vector indicating the id (or ids) of the series to remove.
- **all**: A logical value to indicate to remove or not all series. The values is used only when the value is TRUE.
**hcpxy_set_data**

Update data for a higchartProxy object

**Description**

Update data for a higchartProxy object

**Usage**

```r
hcpxy_set_data(
  proxy,  # A higchartProxy object.
  type,   # series type (column, bar, line, etc)
  data,   # dataframe of new data to send to chart
  mapping = hcaes(),  # how data should be mapped using hcaes()
  redraw = FALSE,     # boolean Whether to redraw the chart after the series is altered. If doing more operations on the chart, it is a good idea to set redraw to false and call hcpxy_redraw after.
  animation = NULL,   # boolean When the updated data is the same length as the existing data, points will be updated by default, and animation visualizes how the points are changed. Set false to disable animation, or a configuration object to set duration or easing.
  updatePoints = TRUE  # boolean When this is TRUE, points will be updated instead of replaced whenever possible. This occurs a) when the updated data is the same length as the existing data, b) when points are matched by their id’s, or c) when points can be matched by X values. This allows updating with animation and performs better. In this case, the original array is not passed by reference. Set FALSE to prevent.
)```

**Arguments**

- `proxy`: A higchartProxy object.
- `type`: series type (column, bar, line, etc)
- `data`: dataframe of new data to send to chart
- `mapping`: how data should be mapped using hcaes()
- `redraw`: boolean Whether to redraw the chart after the series is altered. If doing more operations on the chart, it is a good idea to set redraw to false and call hcpxy_redraw after.
- `animation`: boolean When the updated data is the same length as the existing data, points will be updated by default, and animation visualizes how the points are changed. Set false to disable animation, or a configuration object to set duration or easing.
- `updatePoints`: boolean When this is TRUE, points will be updated instead of replaced whenever possible. This occurs a) when the updated data is the same length as the existing data, b) when points are matched by their id’s, or c) when points can be matched by X values. This allows updating with animation and performs better. In this case, the original array is not passed by reference. Set FALSE to prevent.
**hcpxy_update**  
*Update options for a higchartProxy object*

**Description**

Update options for a higchartProxy object

**Usage**

```r
hcpxy_update(proxy, ...)
```

**Arguments**

- `proxy` A higchartProxy object.
- `...` Named options.

**hcpxy_update_point**  
*Update options series in a higchartProxy object*

**Description**

Update options series in a higchartProxy object

**Usage**

```r
hcpxy_update_point(proxy, id = NULL, id_point = NULL, ...)
```

**Arguments**

- `proxy` A higchartProxy object.
- `id` A character indicating the id of the series’ point to update.
- `id_point` A vector value indicating the point’s index to update, (0 based).
- `...` Arguments defined in [https://api.highcharts.com/class-reference/Highcharts.Point](https://api.highcharts.com/class-reference/Highcharts.Point). The arguments will be the same for each series. So if you want update data it is used this function sequentially for each point.
**hcpxy_update_series**  
*Update options series in a higchartProxy object*

**Description**  
Update options series in a higchartProxy object

**Usage**  
```r  
hcpxy_update_series(proxy, id = NULL, ...)  
```

**Arguments**

- **proxy**  
  A higchartProxy object.

- **id**  
  A character vector indicating the id (or ids) of the series to update.

- **...**  
  Arguments defined in [https://api.highcharts.com/highcharts/plotOptions.series](https://api.highcharts.com/highcharts/plotOptions.series). The arguments will be the same for each series. So if you want update data it is used this function sequentially for each series.

**hcspark**  
*Shortcut to make spkarlines*

**Description**  
Shortcut to make spkarlines

**Usage**

```r  
hcspark(x = NULL, type = NULL, ...)  
```

**Arguments**

- **x**  
  A numeric vector.

- **type**  
  Type sparkline: line, bar, etc.

- **...**  

**Examples**

```r  
set.seed(123)  
x <- cumsum(rnorm(10))  
hcspark(x)  
hcspark(x, "columnn")  
hcspark(c(1, 4, 5), "pie")  
hcspark(x, type = "area")  
```
hctreemap  

Shortcut for create treemaps

Description

This function helps to create highcharts treemaps from treemap objects from the package treemap. 
NOTE: This function is deprecated. Please use hctreemap2 instead.

Usage

hctreemap(tm, ...)

Arguments

tm  
A treemap object from the treemap package.

...  

Examples

```r
## Not run:
library("treemap")
library("viridis")
data(GNI2014)
head(GNI2014)

tm <- treemap(GNI2014,
index = c("continent", "iso3"),
vSize = "population", vColor = "GNI",
type = "comp", palette = rev(viridis(6)),
draw = FALSE)

hctreemap(tm, allowDrillToNode = TRUE, layoutAlgorithm = "squarified") %>%
  hc_title(text = "Gross National Income World Data") %>%
  hc_tooltip(pointFormat = "<b>{point.name}</b>: <br>
  Pop: {point.value:,.0f}<br>
  GNI: {point.valuecolor:,.0f}"
)

## End(Not run)
```
Description

This function helps create highcharts treemaps from data frames.

Usage

hctreemap2(data, group_vars, size_var, color_var = NULL, ...)

Arguments

data          data frame containing variables to organize each level of the treemap on

                  group_vars vector of strings containing column names of variables to generate treemap levels from. the first listed column will specify the top level of the treemap. the unique values in each of these columns must have no intersection (including NAs).

                  size_var string name of column containing numeric data to aggregate by

                  color_var string name of column containing numeric data to color by. defaults to same column as size_var

                  ... additional shared arguments for the data series (https://api.highcharts.com/highcharts/series).

Value

highchart plot object

Examples

```r
## Not run:
library(tidyverse)
library(highcharter)
library(RColorBrewer)
tibble(
  index1 = sample(LETTERS[1:5], 500, replace = T),
  index2 = sample(LETTERS[6:10], 500, replace = T),
  index3 = sample(LETTERS[11:15], 500, replace = T),
  value = rpois(500, 5),
  color_value = rpois(500, 5)
) %>%
  hctreemap2(
    group_vars = c("index1", "index2", "index3"),
    size_var = "value",
    color_var = "color_value",
    layoutAlgorithm = "squarified",
```
levelIsConstant = FALSE,
levels = list(
  list(level = 1, dataLabels = list(enabled = TRUE)),
  list(level = 2, dataLabels = list(enabled = FALSE)),
  list(level = 3, dataLabels = list(enabled = FALSE))
)
) %>%
hc_colorAxis(
  minColor = brewer.pal(7, "Greens")[1],
  maxColor = brewer.pal(7, "Greens")[7]
) %>%
hc_tooltip(pointFormat = "<b>{point.name}</b>:<br>
Value: {point.value:,.0f}<br>
Color Value: {point.colorValue:,.0f}"

## End(Not run)

---

**hc_add_annotation**  
*Helper to add annotations from data frame or list*

### Description

Helper to add annotations from data frame or list

### Usage

```r
hc_add_annotation(hc, ...)
```

```r
hc_add_annotations(hc, x)
```

### Arguments

- **hc**
  A highchart htmlwidget object.

- **...**
  Arguments defined in [https://api.highcharts.com/highcharts/annotations](https://api.highcharts.com/highcharts/annotations).

- **x**
  A list or a data.frame of annotations.

### Details

The `x` elements must have `xValue` and `yValue` elements
**hc_add_dependency**  
*Add modules or plugin dependencies to highcharts objects*

**Description**

Add modules or plugin dependencies to highcharts objects

**Usage**

```r
hc_add_dependency(hc, name = "plugins/annotations.js")
```

**Arguments**

- **hc**
  - A highchart htmlwidget object.
- **name**
  - The partial path to the plugin or module, example: "plugins/annotations.js"

**Details**

See vignette("modules")

**Examples**

```r
data(mpg, package = "ggplot2")

hchart(mpg, "point", hcaes(displ, hwy),
regression = TRUE,
regressionSettings = list(type = "polynomial", order = 5, hideInLegend = TRUE))
#>
hc_add_dependency("plugins/highcharts-regression.js")

hchart(mpg, "point", hcaes(displ, hwy, group = drv), regression = TRUE)
#>
hc_colors(c("#d35400", "#2980b9", "#2ecc71"))
#>
hc_add_dependency("plugins/highcharts-regression.js")
```

**hc_add_dependency_fa**  
*Helpers functions to get FontAwesome icons code*

**Description**

Helpers functions to get FontAwesome icons code
Usage

hc_add_dependency_fa(hc)
fa_icon(iconname = "circle")
fa_icon_mark(iconname = "circle")

Arguments

hc A highchart htmlwidget object.
iconname The icon's name

Description

When you use highcharter in a shiny app, for example renderHighcharter('my_chart'), you can access to the actions of the user using and then use the hc_add_event_point via the my_chart input (input$my_chart). That's a way you can use a chart as an input.

Usage

hc_add_event_point(hc, series = "series", event = "click")
hc_add_event_series(hc, series = "series", event = "click")

Arguments

hc A highchart htmlwidget object.
series The name of type of series to apply the event.
event The name of event: click, mouseOut, mouseOver. See https://api.highcharts.com/highcharts/plotOptions.areasplinerange.point.events.select for more details.

Note

Event details are accessible from hc_name_EventType, i.e. if a highchart is rendered against output$my_hc and and we wanted the coordinates of the user-clicked point we would use input$my_hc_click
hc_add_series

Adding data to highchart objects

Description

Adding data to highchart objects

Usage

hc_add_series(hc, data = NULL, ...)

Arguments

hc
A highchart htmlwidget object.

data
An R object like numeric, list, ts, xts, etc.

...
Arguments defined in https://api.highcharts.com/highcharts/plotOptions.series.

Examples

highchart() %>%
  hc_add_series(data = abs(rnorm(5)), type = "column") %>%
  hc_add_series(data = purrr::map(0:4, function(x) list(x, x)), type = "scatter", color = "orange")

hc_add_series.character

hc_add_series for character and factor objects

Description

hc_add_series for character and factor objects

Usage

## S3 method for class 'character'
hc_add_series(hc, data, ...)

## S3 method for class 'factor'
hc_add_series(hc, data, ...)

Arguments

hc
A highchart htmlwidget object.

data
A character or factor object.

...  
Arguments defined in https://api.highcharts.com/highcharts/plotOptions.series.
**hc_add_series.data.frame**

*hc_add_series for data frames objects*

**Description**

hc_add_series for data frames objects

**Usage**

```r
## S3 method for class 'data.frame'
hc_add_series(hc, data, type = NULL, mapping = hcaes(), fast = FALSE, ...)
```

**Arguments**

- **hc**: A highchart htmlwidget object.
- **data**: A data.frame object.
- **type**: The type of the series: line, bar, etc.
- **mapping**: The mapping, same idea as ggplot2.
- **fast**: convert to json during the composition of a highchart object
- **...**: Arguments defined in [https://api.highcharts.com/highcharts/chart](https://api.highcharts.com/highcharts/chart).

---

**hc_add_series.density**  
*hc_add_series for density objects*

**Description**

hc_add_series for density objects

**Usage**

```r
## S3 method for class 'density'
hc_add_series(hc, data, ...)
```

**Arguments**

- **hc**: A highchart htmlwidget object.
- **data**: A density object.
- **...**: Arguments defined in [https://api.highcharts.com/highcharts/plotOptions.series](https://api.highcharts.com/highcharts/plotOptions.series).
hc_add_series.forecast

hc_add_series for forecast objects

Description

hc_add_series for forecast objects

Usage

## S3 method for class 'forecast'
hc_add_series(
  hc,
  data,
  addOriginal = FALSE,
  addLevels = TRUE,
  fillOpacity = 0.1,
  name = NULL,
  ...
)

Arguments

- **hc** A highchart htmlwidget object.
- **data** A forecast object.
- **addOriginal** Logical value to add the original series or not.
- **addLevels** Logical value to show predictions bands.
- **fillOpacity** The opacity of bands.
- **name** The name of the series.
- **...** Arguments defined in https://api.highcharts.com/highcharts/chart.

hc_add_series.geo_json

hc_add_series for geo_json & geo_list objects

Description

hc_add_series for geo_json & geo_list objects

Usage

## S3 method for class 'geo_json'
hc_add_series(hc, data, type = NULL, ...)

## S3 method for class 'geo_list'
hc_add_series(hc, data, type = NULL, ...)
hc_add_series.lm

Arguments

hc | A highchart htmlwidget object.
data | A geo_json or geo_list object.
type | Type of series. Can be 'mapline', 'mapoint'.
...
Arguments defined in https://api.highcharts.com/highcharts/plotOptions.series.

hc_add_series.lm | hc_add_series for lm and loess objects

Description

hc_add_series for lm and loess objects

Usage

## S3 method for class 'lm'
hc_add_series(
  hc,
data,
type = "line",
color = "#5F83EE",
fillOpacity = 0.1,
...
)

## S3 method for class 'loess'
hc_add_series(
  hc,
data,
type = "line",
color = "#5F83EE",
fillOpacity = 0.1,
...
)

Arguments

hc | A highchart htmlwidget object.
data | A lm or loess object.
type | The type of the series: line, spline.
color | A stringr color.
fillOpacity | fillOpacity to the confidence interval.
...
Arguments defined in https://api.highcharts.com/highcharts/chart.
**hc_add_series.numeric**  
*hc_add_series for numeric objects*

**Description**

*hc_add_series for numeric objects*

**Usage**

```r
## S3 method for class 'numeric'
hc_add_series(hc, data, ...)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `data` A numeric object
- `...` Arguments defined in [https://api.highcharts.com/highcharts/plotOptions.series](https://api.highcharts.com/highcharts/plotOptions.series)

**hc_add_series.ts**  
*hc_add_series for time series objects*

**Description**

*hc_add_series for time series objects*

**Usage**

```r
## S3 method for class 'ts'
hc_add_series(hc, data, ...)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `data` A time series ts object.
- `...` Arguments defined in [https://api.highcharts.com/highcharts/plotOptions.series](https://api.highcharts.com/highcharts/plotOptions.series)
**hc_add_series.xts**  
*hc_add_series for xts objects*

**Description**

hc_add_series for xts objects

**Usage**

```r
## S3 method for class 'xts'
hc_add_series(hc, data, ...)

## S3 method for class 'ohlc'
hc_add_series(hc, data, type = "candlestick", ...)
```

**Arguments**

- `hc`  
  A highchart htmlwidget object.
- `data`  
  A xts object.
- `...`  
  Arguments defined in https://api.highcharts.com/highcharts/plotOptions.series.
- `type`  
  The way to show the xts object. Can be 'candlestick' or 'ohlc'.

**hc_add_series_list**  
*Shortcut for data series from a list of data series*

**Description**

Shortcut for data series from a list of data series

**Usage**

```r
hc_add_series_list(hc, x)
```

**Arguments**

- `hc`  
  A highchart htmlwidget object.
- `x`  
  A list or a data.frame of series.
**hc_add_series_map**

**Examples**

```r
ds <- lapply(seq(5), function(x) {
    list(data = cumsum(rnorm(100, 2, 5)), name = x)
})

highchart() %>%
  hc_plotOptions(series = list(marker = list(enabled = FALSE))) %>%
  hc_add_series_list(ds)
```

---

**hc_add_series_map**  
*Add a map series*

**Description**

Add a map series

**Usage**

```r
hc_add_series_map(hc, map, df, value, joinBy, ...)
```

**Arguments**

- **hc**: A highchart htmlwidget object.
- **map**: A list object loaded from a geojson file.
- **df**: A data.frame object with data to chart. Code region and value are required.
- **value**: A string value with the name of the variable to chart.
- **joinBy**: What property to join the map and df
- **...**: Additional shared arguments for the data series ([https://api.highcharts.com/highcharts/series](https://api.highcharts.com/highcharts/series)).

**Details**

This function force the highchart object to be map type.

**Examples**

```r
library("dplyr")

data("USArrests", package = "datasets")
data("usgeojson")

USArrests <- mutate(USArrests, state = rownames(USArrests))

highchart() %>%
  hc_title(text = "Violent Crime Rates by US State") %>%
Add highcharts themes to a highchart object.

**Usage**

`hc_add_theme(hc, hc_thm)`

**Arguments**

- `hc`: A highchart object
- `hc_thm`: A highchart theme object ("hc_theme" class)

**Examples**

```r
highchart() %>%
  hc_add_theme()
```
Annotations options for highchart objects

Description
A basic type of an annotation. It allows to add custom labels or shapes. The items can be tied to points, axis coordinates or chart pixel coordinates.

Usage
hc_annotations(hc, ...)

Arguments
hc A highchart htmlwidget object.
...
Arguments defined in https://api.highcharts.com/highcharts/annotations.

Examples

# Ex 1
highchart() %>%
  hc_add_series(
    data = c(29.9, 71.5, 106.4, 129.2, 144.0, 176.0, 135.6, 148.5, 216.4, 194.1, 95.6, 54.4)
  ) %>%
  hc_xAxis(
    tickInterval = 0.5,
    gridLineWidth = 1
  ) %>%
  hc_annotations(
    list(
      labels =
        list(
          list(
            point = list(x = 3, y = 129.2, xAxis = 0, yAxis = 0),
            text = "x: {x}<br/>y: {y}"
          ),
          list(
            point = list(x = 9, y = 194.1, xAxis = 0, yAxis = 0),
            text = "x: {x}<br/>y: {y}"
          ),
          list(
        )
    )
)
```r
# Ex 2
df <- data.frame(
  x = 1:10,
  y = 1:10
)

highchart() %>%
  hc_add_series(data = df, hcaes(x = x, y = y), type = "area") %>%
  hc_annotations(
    list(
      labels = list(
        list(point = list(x = 5, y = 5, xAxis = 0, yAxis = 0), text = "Middle"),
        list(point = list(x = 1, y = 1, xAxis = 0, yAxis = 0), text = "Start")
      )
    )
  )
```

---

**hc_boost**

**Boost options for highchart objects**

**Description**

Options for the Boost module. The Boost module allows certain series types to be rendered by WebGL instead of the default SVG. This allows hundreds of thousands of data points to be rendered in milliseconds. In addition to the WebGL rendering it saves time by skipping processing and inspection of the data wherever possible. This introduces some limitations to what features are available in boost mode. See the docs for details. In addition to the global boost option, each series has a boostThreshold that defines when the boost should kick in. Requires the modules/boost.js module.

**Usage**

```
hc_boost(hc, ...)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `...` Arguments defined in [https://api.highcharts.com/highcharts/boost](https://api.highcharts.com/highcharts/boost).
**Examples**

```
# Ex 1
options(highcharter.rjson = FALSE)

n <- 50000

x <- sin(4 * 2 * pi * seq(n) / n) + rnorm(n) / 10

x <- round(x, 3)

plot(x)

hc1 <- highchart() %>%
  hc_chart(zoomType = "x") %>%
  hc_add_series(data = x) %>%
  hc_title(text = "No boost") %>%
  hc_boost(
    enabled = FALSE # Default
  )

hc1

# Boost is a stripped-down renderer-in-a-module for Highcharts. It bypasses
# some of the standard Highcharts features (such as animation), and focuses
# on pushing as many points as possible as quickly as possible.

hc2 <- highchart() %>%
  hc_chart(zoomType = "x") %>%
  hc_add_series(data = x) %>%
  hc_title(text = "With boost") %>%
  hc_boost(enabled = TRUE)

hc2

# # Ex 2
# library(MASS)
#
# n <- 20000
#
# sigma <- matrix(c(10,3,3,2),2,2)
# sigma
#
# mvr <- round(mvrnorm(n, rep(c(0, 0)), sigma), 2)
# vx <- ceiling(1+abs(max(mvr[, 1])))
# vy <- ceiling(1+abs(max(mvr[, 2])))
#
# # unnamed list
# ds <- list_parse2(as.data.frame(mvr))
# ```
# highchart() %>%
# hc_chart(zoomType = "xy") %>%
# hc_xAxis(min = -vx, max = vx) %>%
# hc_yAxis(min = -vy, max = vy) %>%
# hc_add_series(
#   data = ds, #list
#   type = "scatter",
#   name = "A lot of points!",
#   color = 'rgba(0,0,0,0.1)',
#   marker = list(radius = 2)
# ) %>%
# hc_boost(
#   enabled = TRUE
# )

# dat <- as.data.frame(mvr)
# names(dat) <- c("x", "y")

# highchart() %>%
# hc_chart(zoomType = "xy") %>%
# hc_xAxis(min = -vx, max = vx) %>%
# hc_yAxis(min = -vy, max = vy) %>%
# hc_add_series(
#   data = dat,
#   type = "scatter",
#   hcaes(x, y),
#   name = "A lot of points!",
#   color = 'rgba(0,0,0,0.1)',
#   marker = list(radius = 2)
# ) %>%
# hc_boost(enabled = TRUE)

# Ex3
# N <- 1000000
# n <- 5
# s <- seq(n)
# s <- s/(max(s) + min(s))
# s <- round(s, 2)

# series <- s %>%
#   purrr::map(~ stats::arima.sim(round(N/n), model = list(ar = .x)) + .x * n * 20) %>%
#   purrr::map(as.vector) %>%
#   purrr::map(round, 2) %>%
#   purrr::map(~ list(data = .x))

# highchart() %>%
# hc_add_series_list(series) %>%
# hc_chart(zoomType = "x") %>%
# hc_boost(enabled = TRUE)

---

hc_caption

Caption options for highcharter objects
**hc_chart**

**Description**

The chart’s caption, which will render below the chart and will be part of exported charts. The caption can be updated after chart initialization through the Chart.update or Chart.caption.update methods.

**Usage**

```
hc_caption(hc, ...)
```

**Arguments**

- `hc`: A highchart htmlwidget object.
- `...`: Arguments defined in https://api.highcharts.com/highcharts/caption.

**Examples**

```r
highchart() %>%
  hc_title(text = "Chart with a caption") %>%
  hc_subtitle(text = "This is the subtitle") %>%
  hc_xAxis(categories = c("Apples", "Pears", "Banana", "Orange")) %>%
  hc_add_series(
    data = c(1, 4, 3, 5),
    type = "column",
    name = "Fruits"
  ) %>%
  hc_caption(
    text = "<b>The caption renders in the bottom, and is part of the exported chart.</b><br>
    Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum."'
  )
```

---

**hc_chart**  
*Chart options for highcharter objects*

**Description**

General options for the chart.

**Usage**

```
hc_chart(hc, ...)
```
Arguments

hc  
A highchart htmlwidget object.

Arguments defined in https://api.highcharts.com/highcharts/chart.

Examples

hc <- highchart() %>%
  hc_xAxis(categories = month.abb) %>%
  hc_add_series(name = "Tokyo", data = sample(1:12)) %>%
  hc_add_series(name = "London", data = sample(1:12) + 10)

hc

hc %>%
  hc_chart(
    type = "column",
    options3d = list(enabled = TRUE, beta = 15, alpha = 15)
  )

hc %>%
  hc_chart(
    borderColor = "#EBBA95",
    borderRadius = 10,
    borderWidth = 2,
    backgroundColor = list(
      linearGradient = c(0, 0, 500, 500),
      stops = list(
        list(0, "rgb(255, 255, 255)",
        list(1, "rgb(200, 200, 255)"")
      )
    )
  )

hc_colorAxis

ColorAxis options for highcharter objects

Description

A color axis for series. Visually, the color axis will appear as a gradient or as separate items inside the legend, depending on whether the axis is scalar or based on data classes. For supported color formats, see the docs article about colors. A scalar color axis is represented by a gradient. The colors either range between the minColor and the maxColor, or for more fine grained control the colors can be defined in stops. Often times, the color axis needs to be adjusted to get the right color spread for the data. In addition to stops, consider using a logarithmic axis type, or setting min and max to avoid the colors being determined by outliers. When dataClasses are used, the ranges are subdivided into separate classes like categories based on their values. This can be used for
ranges between two values, but also for a true category. However, when your data is categorized, it may be as convenient to add each category to a separate series. Color axis does not work with: sankey, sunburst, dependencywheel, networkgraph, wordcloud, venn, gauge and solidgauge series types. Since v7.2.0 colorAxis can also be an array of options objects. See the Axis object for programmatic access to the axis.

Usage

hc_colorAxis(hc, ...)

Arguments

hc A highchart htmlwidget object.
...
Arguments defined in https://api.highcharts.com/highcharts/colorAxis.

Examples

library(dplyr)
data(mpg, package = "ggplot2")

mpgman2 <- mpg %>%
  group_by(manufacturer, year) %>%
dplyr::summarise(
  n = dplyr::n(),
  displ = mean(displ)
)

mpgman2

hchart(
  mpgman2, "column", hcaes(x = manufacturer, y = n, group = year),
  colorKey = "displ",
  # color = c("#FCA50A", "#FCFFA4"),
  name = c("Year 1999", "Year 2008")
) %>%
  hc_colorAxis(min = 0, max = 5)

# defaults to yAxis
hchart(iris, "point", hcaes(Sepal.Length, Sepal.Width)) %>%
  hc_colorAxis(
    minColor = "red",
    maxColor = "blue"
  )

# Ex2
n <- 5
stops <- data.frame(
    q = 0:n / n,
    c = c("#440154", "#414487", "#2A788E", "#22A884", "#7AD151", "#FDE725"),
    stringsAsFactors = FALSE
)

stops <- list_parse2(stops)

M <- round(matrix(rnorm(50 * 50), ncol = 50), 2)

hchart(M) %>%
    hc_colorAxis(stops = stops)

# Ex3
# hchart(volcano) %>%
#   hc_colorAxis(stops = stops, max = 200)

---

**hc_colors**

*Colors options for highcharter objects*

**Description**

An array containing the default colors for the chart’s series. When all colors are used, new colors are pulled from the start again.

**Usage**

```r
hc_colors(hc, colors)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `colors` A vector of colors.

**Examples**

```r
library(viridisLite)

cols <- viridis(3)
cols <- substr(cols, 0, 7)

highchart() %>%
    hc_add_series(data = sample(1:12)) %>%
    hc_add_series(data = sample(1:12) + 10) %>%
    hc_add_series(data = sample(1:12) + 20) %>%
    hc_colors(cols)
```
**hc_credits**

*Credits options for highcharter objects*

**Description**

Highchart by default puts a credits label in the lower right corner of the chart. This can be changed using these options.

**Usage**

```r
hc_credits(hc, ...)
```

**Arguments**

- `hc`: A highchart htmlwidget object.
- `...`: Arguments defined in [https://api.highcharts.com/highcharts/credits](https://api.highcharts.com/highcharts/credits).

**Examples**

```r
highchart() %>%
  hc_xAxis(categories = citytemp$month) %>%
  hc_add_series(name = "Tokyo", data = sample(1:12)) %>%
  hc_credits(
    enabled = TRUE,
    text = "htmlwidgets.org",
    href = "http://www.htmlwidgets.org/
  )
```

**hc_drilldown**

*Drilldown options for highcharter objects*

**Description**

Options for drill down, the concept of inspecting increasingly high resolution data through clicking on chart items like columns or pie slices. The drilldown feature requires the drilldown.js file to be loaded, found in the modules directory of the download package, or online at [code.highcharts.com/modules/drilldown.js](http://code.highcharts.com/modules/drilldown.js).

**Usage**

```r
hc_drilldown(hc, ...)
```

**Arguments**

- `hc`: A highchart htmlwidget object.
- `...`: Arguments defined in [https://api.highcharts.com/highcharts/drilldown](https://api.highcharts.com/highcharts/drilldown).
Examples

```r
library(highcharter)
library(dplyr)
library(purrr)

df <- tibble(
    name = c("Animals", "Fruits"),
    y = c(5, 2),
    drilldown = tolower(name)
)
df

hc <- highchart() %>%
    hc_title(text = "Basic drilldown") %>%
    hc_xAxis(type = "category") %>%
    hc_legend(enabled = FALSE) %>%
    hc_plotOptions(
        series = list(
            borderWidth = 0,
            dataLabels = list(enabled = TRUE)
        )
    ) %>%
    hc_add_series(
        data = df,
        type = "column",
        hcaes(name = name, y = y),
        name = "Things",
        colorByPoint = TRUE
    )

dfan <- data.frame(
    name = c("Cats", "Dogs", "Cows", "Sheep", "Pigs"),
    value = c(4, 3, 1, 2, 1)
)
dffru <- data.frame(
    name = c("Apple", "Organes"),
    value = c(4, 2)
)

dsan <- list_parse2(dfan)
dsfru <- list_parse2(dffru)

hc <- hc %>%
    hc_drilldown(
        allowPointDrilldown = TRUE,
        series = list(
            list(

hc_elementId

id = "animals",
data = dsan
),
list(
  id = "fruits",
data = dsfru
)
)
)

hc

<table>
<thead>
<tr>
<th>hc_elementId</th>
<th>Setting elementId</th>
</tr>
</thead>
</table>

**Description**

Function to modify the id for the container.

**Usage**

hc_elementId(hc, id = NULL)

**Arguments**

- **hc**: A highchart htmlwidget object.
- **id**: A string

**Examples**

```r
hchart(rnorm(10)) %>%
  hc_elementId("newid")
```

hc_exporting

**Exporting options for highchart objects**

**Description**

Options for the exporting module. For an overview on the matter, see the docs.

**Usage**

hc_exporting(hc, ...)

Arguments

hc  A highchart htmlwidget object.
...
Arguments defined in https://api.highcharts.com/highcharts/exporting.

Examples

```r
date <- c(1:12)
hc <- highchart() %>%
  hc_xAxis(categories = month.abb) %>%
  hc_add_series(name = "Tokyo", data = sample(1:12)) %>%
  hc_exporting(
    enabled = TRUE, # always enabled
    filename = "custom-file-name"
  )
```

hc_labels  

Labels options for highcharter objects

Description

HTML labels that can be positioned anywhere in the chart area. This option is deprecated since v7.1.2. Instead, use annotations that support labels.

Usage

hc_labels(hc, ...)

Arguments

hc  A highchart htmlwidget object.
...
Arguments defined in https://api.highcharts.com/highcharts/labels.

Examples

```r
date <- c(1:12)
hc <- highchart() %>%
  hc_add_series(data = sample(1:12)) %>%
  hc_labels(
    items = list(
      list(
        html = "<p>Some <b>important</b><br>text</p>",
        style = list(
          left = "150%",
          top = "150%"
        )
      )
    )
  )
```
**hc_legend**

*Legend options for highcharter objects*

---

**Description**

The legend is a box containing a symbol and name for each series item or point item in the chart. Each series (or points in case of pie charts) is represented by a symbol and its name in the legend. It is possible to override the symbol creator function and create custom legend symbols.

**Usage**

`hc_legend(hc, ...)`

**Arguments**

- `hc`: A highchart htmlwidget object.
- `...`: Arguments defined in [https://api.highcharts.com/highcharts/legend](https://api.highcharts.com/highcharts/legend).

**Details**

A Highmaps legend by default contains one legend item per series, but if a colorAxis is defined, the axis will be displayed in the legend. Either as a gradient, or as multiple legend items for dataClasses.

**Examples**

```r
highchart() %>%
  hc_xAxis(categories = month.abb) %>%
  hc_add_series(name = "Tokyo", data = sample(1:12)) %>%
  hc_add_series(name = "London", data = sample(1:12) + 10) %>%
  hc_add_series(name = "Other City", data = sample(1:12) + 20) %>%
  hc_legend(
    align = "left",
    verticalAlign = "top",
    layout = "vertical",
    x = 0,
    y = 100
  )
```
hc_loading  

**Description**

The loading options control the appearance of the loading screen that covers the plot area on chart operations. This screen only appears after an explicit call to chart.showLoading(). It is a utility for developers to communicate to the end user that something is going on, for example while retrieving new data via an XHR connection. The "Loading..." text itself is not part of this configuration object, but part of the lang object.

**Usage**

```r
hc_loading(hc, ...)
```

**Arguments**

- **hc** A highchart htmlwidget object.
- **...** Arguments defined in [https://api.highcharts.com/highcharts/loading](https://api.highcharts.com/highcharts/loading).

**Examples**

```r
highcharts_demo() %>%
  hc_loading(
    hideDuration = 1000,
    showDuration = 1000
  )
```

---

hc_mapNavigation  

**Description**

Mapnavigation options for highcharter objects

**Usage**

```r
hc_mapNavigation(hc, ...)
```

**Arguments**

- **hc** A highchart htmlwidget object.
Examples

```r
hcmap(download_map_data = TRUE) %>%
  hc_mapNavigation(
    enabled = TRUE,
    enableMouseWheelZoom = TRUE,
    enableDoubleClickZoom = TRUE
  )
```

---

**hc_motion**  
*Setting Motion options to highcharts objects*

**Description**

The Motion Highcharts Plugin adds an interactive HTML5 player to any Highcharts chart (Highcharts, Highmaps and Highstock).

**Usage**

```r
hc_motion(hc, enabled = TRUE, startIndex = 0, ...)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `enabled` Enable the motion plugin.
- `startIndex` start index, default to 0.
- `...` Arguments defined in [https://github.com/TorsteinHonsi/Motion-Highcharts-Plugin/wiki](https://github.com/TorsteinHonsi/Motion-Highcharts-Plugin/wiki).

---

**hc_navigator**  
*Navigator options for highchart objects*

**Description**

The navigator is a small series below the main series, displaying a view of the entire data set. It provides tools to zoom in and out on parts of the data as well as panning across the dataset.

**Usage**

```r
hc_navigator(hc, ...)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `...` Arguments defined in [https://api.highcharts.com/highstock/navigator](https://api.highcharts.com/highstock/navigator).
Examples

```r
highchart(type = "stock") %>%
  hc_add_series(AirPassengers) %>%
  hc_rangeSelector(selected = 4) %>%
  hc_navigator(
    outlineColor = "gray",
    outlineWidth = 2,
    series = list(
      color = "red",
      lineWidth = 2,
      type = "areaspline", # you can change the type
      fillColor = "rgba(255, 0, 0, 0.2)"
    ),
    handles = list(
      backgroundColor = "yellow",
      borderColor = "red"
    )
  )
```

**hc_pane**  
*Pane options for highcharter objects*

Description

The pane serves as a container for axes and backgrounds for circular gauges and polar charts.

Usage

```r
hc_pane(hc, ...)
```

Arguments

- `hc`  
  A `highchart` `htmlwidget` object.
- `...`  
  Arguments defined in [https://api.highcharts.com/highcharts/pane](https://api.highcharts.com/highcharts/pane).

Examples

```r
highchart() %>%
  hc_chart(
    type = "gauge",
    plotBackgroundColor = NULL,
    plotBackgroundImage = NULL,
    plotBorderWidth = 0,
    plotShadow = FALSE
  ) %>%
  hc_title(
    text = "Speedometer"
  )
```
```r
hc_pane

```hc_pane

    startAngle = -150,
    endAngle = 150,
    background = list(list(
        backgroundColor = list(
            linearGradient = list(x1 = 0, y1 = 0, x2 = 0, y2 = 1),
            stops = list(
                list(0, "#FFF"),
                list(1, "#333")
            )
        ),
        borderWidth = 0,
        outerRadius = "109%"
    ), list(
        backgroundColor = list(
            linearGradient = list(x1 = 0, y1 = 0, x2 = 0, y2 = 1),
            stops = list(
                list(0, "#333"),
                list(1, "#FFF")
            )
        ),
        borderWidth = 1,
        outerRadius = "107%"
    ), list(
        # default background
    ), list(
        backgroundColor = "#DDD",
        borderWidth = 0,
        outerRadius = "105%",
        innerRadius = "103%"
    ))

```hc_add_series

    data = list(80), name = "speed", tooltip = list(valueSuffix = " km/h")

```hc_yAxis

    min = 0,
    max = 200,
    minorTickInterval = "auto",
    minorTickWidth = 1,
    minorTickLength = 10,
    minorTickPosition = "inside",
    minorTickColor = "#666",
    tickPixelInterval = 30,
    tickWidth = 2,
    tickPosition = "inside",
    tickLength = 10,
    tickColor = "#666",
    labels = list(
        step = 2,
        rotation = "auto"
    ),
```
hc_plotOptions

Plotoptions options for highcharter objects

Description

The plotOptions is a wrapper object for config objects for each series type. The config objects for each series can also be overridden for each series item as given in the series array. Configuration options for the series are given in three levels. Options for all series in a chart are given in the plotOptions.series object. Then options for all series of a specific type are given in the plotOptions of that type, for example plotOptions.line. Next, options for one single series are given in the series array.

Usage

hc_plotOptions(hc, ...)

Arguments

hc A highchart htmlwidget object.
...

Arguments defined in https://api.highcharts.com/highcharts/plotOptions.

Examples

highchart() %>%
  hc_add_series(
    data = c(29.9, 71.5, 106.4, 129.2, 144.0, 176.0, 135.6, 148.5, 216.4, 194.1, 95.6, 54.4)
  ) %>%
  hc_plotOptions(
    line = list(
      color = "blue",
      marker = list(
        fillColor = "white",
        lineWidth = 2,
        lineColor = NULL
      )
    )
  )
hc_rangeSelector

**Rangeselector options for highchart objects**

**Description**

The range selector is a tool for selecting ranges to display within the chart. It provides buttons to select preconfigured ranges in the chart, like 1 day, 1 week, 1 month etc. It also provides input boxes where min and max dates can be manually input.

**Usage**

```r
hc_rangeSelector(hc, ...)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `...` Arguments defined in https://api.highcharts.com/highstock/rangeSelector.

**Examples**

```r
c <- highchart(type = "stock") %>%
  hc_add_series(AirPassengers)

hc <-

hc %>%
  hc_rangeSelector(enabled = FALSE)

hc %>%
  hc_rangeSelector(
    verticalAlign = "bottom",
    selected = 4
  )
```

---

hc_responsive

**Responsive options for highchart objects**

**Description**

Allows setting a set of rules to apply for different screen or chart sizes. Each rule specifies additional chart options.

**Usage**

```r
hc_responsive(hc, ...)
```


**Arguments**

- **hc**  
  A highchart htmlwidget object.

**Examples**

```r
leg_500_opts <- list(enabled = FALSE)
leg_900_opts <- list(align = "right", verticalAlign = "middle", layout = "vertical")

# change the with of the container/windows to see the effect
highchart() %>%
  hc_add_series(data = cumsum(rnorm(100))) %>%
  hc_responsive(
    rules = list(
      # remove legend if there is no much space
      list(
        condition = list(maxWidth = 500),
        chartOptions = list(legend = leg_500_opts)
      ),
      # put legend on the right when there is much space
      list(
        condition = list(minWidth = 900),
        chartOptions = list(legend = leg_900_opts)
      )
    )
  )
```

---

**hc_rm_series**  
Removing series to highchart objects

**Description**

Removing series to highchart objects

**Usage**

```r
hc_rm_series(hc, names = NULL)
```

**Arguments**

- **hc**  
  A highchart htmlwidget object.

- **names**  
  The series's names to delete.
**hc_scrollbar**

Scrollbar options for highcharter objects

---

**Description**

The scrollbar is a means of panning over the X axis of a stock chart. Scrollbars can also be applied to other types of axes. Another approach to scrollable charts is the chart.scrollablePlotArea option that is especially suitable for simpler cartesian charts on mobile. In styled mode, all the presentational options for the scrollbar are replaced by the classes .highcharts-scrollbar-thumb, .highcharts-scrollbar-arrow, .highcharts-scrollbar-button, .highcharts-scrollbar-rifles and .highcharts-scrollbar-track.

**Usage**

```r
hc_scrollbar(hc, ...)
```

**Arguments**

- `hc` A highchart htmlwidget object.
- `...` Arguments defined in [https://api.highcharts.com/highstock/scrollbar](https://api.highcharts.com/highstock/scrollbar).

**Examples**

```r
highchart(type = "stock") %>%
    hc_add_series(AirPassengers) %>%
    hc_rangeSelector(selected = 4) %>%
    hc_scrollbar(
        barBackgroundColor = "gray",
        barBorderRadius = 7,
        barBorderWidth = 0,
        buttonBackgroundColor = "gray",
        buttonBorderWidth = 0,
        buttonArrowColor = "yellow",
        buttonBorderRadius = 7,
        rifleColor = "yellow",
        trackBackgroundColor = "white",
        trackBorderWidth = 1,
        trackBorderColor = "silver",
        trackBorderRadius = 7
    )
```
### hc_series

**Series options for highchart objects**

**Description**

Series options for specific data and the data itself. In TypeScript you have to cast the series options to specific series types, to get all possible options for a series.

**Usage**

```r
hc_series(hdc, ...)
```

**Arguments**

- `hc`: A highchart htmlwidget object.
- `...`: Arguments defined in [https://api.highcharts.com/highcharts/series](https://api.highcharts.com/highcharts/series).

**Examples**

```r
highchart() %>%
  hc_series(
    list(
      name = "Tokyo",
      data = c(7.0, 6.9, 9.5, 14.5, 18.4, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6)
    ),
    list(
      name = "London",
      data = c(3.9, 4.2, 5.7, 8.5, 11.9, 15.2, 17.0, 16.6, 14.2, 10.3, 6.6, 4.8)
    )
  )
```

### hc_size

**Changing the size of a highchart object**

**Description**

Changing the size of a highchart object

**Usage**

```r
hc_size(hdc, width = NULL, height = NULL)
```

**Arguments**

- `hc`: A highchart htmlwidget object.
- `width`: A numeric input in pixels.
- `height`: A numeric input in pixels.
Examples

```r
hc <- hchart(ts(rnorm(100)), showInLegend = FALSE)
hc_size(hc, 200, 200)
```

---

**hc_subtitle**

*Subtitle options for highchart objects*

**Description**

The chart’s subtitle. This can be used both to display a subtitle below the main title, and to display random text anywhere in the chart. The subtitle can be updated after chart initialization through the `Chart.setTitle` method.

**Usage**

```r
hc_subtitle(hc, ...)
```

**Arguments**

- `hc`: A `highchart` htmlwidget object.
- `...`: Arguments defined in https://api.highcharts.com/highcharts/subtitle.

**Examples**

```r
highchart() %>%
  hc_add_series(  
    data = c(7.0, 6.9, 9.5, 14.5, 18.2, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6),
    type = "column",
  ) %>%
  hc_subtitle(  
    text = "And this is a subtitle with more information",
    align = "left",
    style = list(color = "#2b908f", fontWeight = "bold")
  )
```
Creating highchart themes

Description

Highcharts is very flexible so you can modify every element of the chart. There are some exiting themes so you can apply style to charts with few lines of code.

Usage

hc_theme(...) 

Arguments

... A list of named parameters.

Details


Examples

hc <- highcharts_demo()

hc

thm <- hc_theme(
    colors = c("red", "green", "blue"),
    chart = list(
        backgroundColor = "#15C0DE"
    ),
    title = list(
        style = list(
            color = "#333333",
            fontFamily = "Erica One"
        )
    ),
    subtitle = list(
        style = list(
            color = "#666666",
            fontFamily = "Shadows Into Light"
        )
    ),
    legend = list(
        itemStyle = list(
            fontFamily = "Tangerine",
            color = "black"
        ),
    )
)
Description

Highcharts is very flexible so you can modify every element of the chart. There are some exiting themes so you can apply style to charts with few lines of code.

Usage

hc_theme_538(...)  
hc_theme_sparkline_vb(...)  
hc_theme_tufte2(...)

Arguments

... A named parameters to modify the theme.

Examples

highcharts_demo() %>%
  hc_add_theme(hc_theme_538())

highcharts_demo() %>%
  hc_add_theme(hc_theme_sparkline_vb())

highchart() %>%
  hc_chart(type = "column") %>%
  hc_add_series(data = round(1 + abs(rnorm(12)), 2), showInLegend = FALSE) %>%
  hc_xAxis(categories = month.abb) %>%
  hc_add_theme(hc_theme_tufte2())
**hc_theme_alone**

*Alone theme for highcharts*

**Description**

Alone theme for highcharts

**Usage**

```r
ehc_theme_alone(...)```

**Arguments**

... A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_alone())
```

---

**hc_theme_bloom**

*Bloomberg Graphics theme for highcharts*

**Description**

Bloomberg Graphics theme for highcharts

**Usage**

```r
ehc_theme_bloom(...)```

**Arguments**

... A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_bloom())
```
Chalk theme for highcharts

**Description**
Chalk theme for highcharts

**Usage**
```r
hc_theme_chalk(...)```

**Arguments**
... A named parameters to modify the theme.

Chalk theme for highcharts was inspired by [https://www.amcharts.com/demos/](https://www.amcharts.com/demos/).

**Examples**
```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_chalk())
```

---

Dark Unica theme for highcharts

**Description**
Dark Unica theme for highcharts

**Usage**
```r
hc_theme_darkunica(...)```

**Arguments**
... A named parameters to modify the theme.

**Examples**
```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_darkunica())
```
hc_theme_db

Description
Dotabuff theme for highcharts

Usage
hc_theme_db(...)

Arguments
... A named parameters to modify the theme.

Examples
highcharts_demo() %>%
  hc_add_theme(hc_theme_db())

hc_theme_economist

Description
Economist theme for highcharts

Usage
hc_theme_economist(...)  

Arguments
... A named parameters to modify the theme.

Examples
highcharts_demo() %>%
  hc_add_theme(hc_theme_economist())
hc_theme_elementary  Elementary (OS) theme for highcharts

Description
Elementary (OS) theme for highcharts was based on https://elementary.io

Usage
hc_theme_elementary(...)

Arguments
... A named parameters to modify the theme.

Examples

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_elementary())
```

hc_theme_ffx  Firefox theme for highcharts

Description
Firefox theme was inspired by https://mozilla.design/.

Usage
hc_theme_ffx(...)

Arguments
... A named parameters to modify the theme.

Examples

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_ffx())
```
**hc_theme_flat**  
*Flat theme for highcharts*

**Description**

Flat and flatdark theme is inspired by [https://github.com/chriskempson/base16](https://github.com/chriskempson/base16) and [https://github.com/Mikata-Project/ggthemr#flat](https://github.com/Mikata-Project/ggthemr#flat)

**Usage**

`hc_theme_flat(...)`

**Arguments**

...  
A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_flat())
```

---

**hc_theme_flatdark**  
*Flatdark theme for highcharts*

**Description**

Flatdark theme for highcharts

**Usage**

`hc_theme_flatdark(...)`

**Arguments**

...  
A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_flatdark())
```
hc_theme_ft

Financial Times theme for highcharts

Description

Financial Times theme for highcharts

Usage

hc_theme_ft(...)

Arguments

...  A named parameters to modify the theme.

Examples

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_ft())
```

hc_theme_ggplot2

ggplot2 theme for highcharts

Description

ggplot2 theme is based on https://ggplot2.tidyverse.org/.

Usage

hc_theme_ggplot2(...)

Arguments

...  A named parameters to modify the theme.

Examples

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_ggplot2())
```
hc_theme_google  

Google theme for highcharts

**Description**

Google theme for highcharts is based on [https://books.google.com/ngrams/](https://books.google.com/ngrams/).

**Usage**

hc_theme_google(...)  

**Arguments**

...  

A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_google())
```

---

hc_theme_gridlight  

Grid Light theme for highcharts

**Description**

Grid Light theme for highcharts

**Usage**

hc_theme_gridlight(...)  

**Arguments**

...  

A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_gridlight())
```
**hc_theme_handdrawn**

*Hand Drawn theme for highcharts*

**Description**


**Usage**

```r
hc_theme_handdrawn()
```

**Arguments**

... A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_handdrawn())
```

---

**hc_theme_hcrt**

*Highcharter theme for highcharts*

**Description**

hcrt theme is used for the documentation website.

**Usage**

```r
hc_theme_hcrt()
```

**Arguments**

... A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_hcrt())
```
hc_theme_merge  Merge themes

Description
Function to combine hc_theme objects.

Usage
hc_theme_merge(...)

Arguments
...  hc_theme objects.

Examples

```
thm <- hc_theme_merge(
  hc_theme_darkunica(),
  hc_theme(
    chart = list(
      backgroundColor = "transparent",
      divBackgroundImage = "http://cdn.wall-pix.net/albums/art-3Dview/00025095.jpg"
    ),
    title = list(
      style = list(
        color = "white",
        fontFamily = "Erica One"
      )
    )
  )
)
```

hc_theme_monokai  Monokai theme for highcharts

Description
Monokai is a well know text editor theme.

Usage
hc_theme_monokai(...)  

Arguments
...  A named parameters to modify the theme.
**hc_theme_null**

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_monokai())
```

---

**hc_theme_null**  
*Null theme for highcharts*

**Description**

For Null theme the axis are removed (visible = FALSE).

**Usage**

```r
hc_theme_null(...)```

**Arguments**

...  
A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_null())
```

---

**hc_theme_sandsignika**  
*Sand Signika theme for highcharts*

**Description**

Sand Signika theme for highcharts

**Usage**

```r
hc_theme_sandsignika(...)```

**Arguments**

...  
A named parameters to modify the theme.

**Examples**

```r
highcharts_demo() %>%
  hc_add_theme(hc_theme_sandsignika())
```
**hc_theme_smpl**  
*Simple theme for highcharts*

**Description**

Theme smpl design is inspired by [https://github.com/hrbrmstr/hrbrmisc/blob/master/R/themes.r](https://github.com/hrbrmstr/hrbrmisc/blob/master/R/themes.r) and color by [https://materialui.co/flatuicolors](https://materialui.co/flatuicolors).

**Usage**

```r
hc_theme_smpl(...)  
```

**Arguments**

...  
A named parameters to modify the theme.

**Examples**

```r
everychart()  
  hc_add_theme(hc_theme_smpl())
```

---

**hc_theme_sparkline**  
*Sparkline theme for highcharts*

**Description**

Sparkline theme is based on [https://www.highcharts.com/demo/sparkline](https://www.highcharts.com/demo/sparkline) and this post [https://jkunst.com/blog/posts/2020-06-26-valuebox-and-sparklines/](https://jkunst.com/blog/posts/2020-06-26-valuebox-and-sparklines/).

**Usage**

```r
hc_theme_sparkline(...)  
```

**Arguments**

...  
A named parameters to modify the theme.

**Examples**

```r
everychart()  
  hc_add_theme(hc_theme_sparkline())
```
hc_theme_superheroes

Superheroes theme for highcharts

Description

The superheroes theme is inspired by https://public.tableau.com/profile/ryansmith#!/vizhome/HeroesofNewYork/SuperheroesinNewYork

Usage

hc_theme_superheroes(...)

Arguments

...  A named parameters to modify the theme.

Examples

highcharts_demo() %>%
hc_add_theme(hc_theme_superheroes())

hc_theme_tufte

Tufte theme for highcharts

Description

Tufte theme for highcharts

Usage

hc_theme_tufte(...)

Arguments

...  A named parameters to modify the theme.

Examples

n <- 15
dta <- data.frame(
  x = 1:n + rnorm(n),
  y = 2 * 1:n + rnorm(n)
)

highchart() %>%
  hc_chart(type = "scatter") %>%
  hc_add_series(data = list_parse(dta), showInLegend = FALSE) %>%
  hc_add_theme(hc_theme_tufte())

---
	hc_title

**Title options for highchart objects**

**Description**

The chart’s main title.

**Usage**

```r
hc_title(hc, ...)
```

**Arguments**

- `hc`: A highchart htmlwidget object.
- `...`: Arguments defined in [https://api.highcharts.com/highcharts/title](https://api.highcharts.com/highcharts/title).

**Examples**

```r
highchart() %>%
  hc_add_series(
    data = c(7.0, 6.9, 9.5, 14.5, 18.2, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6),
    type = "column"
  ) %>%
  hc_title(
    text = "This is a title with \textit{margin} and \textbf{Strong or bold text}\textbf{\textit{text}}",
    margin = 20,
    align = "left",
    style = list(color = "#22A884", useHTML = TRUE)
  )
```

---

**hc_tooltip**

**Tooltip options for highchart objects**

**Description**

Options for the tooltip that appears when the user hovers over a series or point.

**Usage**

```r
hc_tooltip(hc, ..., sort = FALSE, table = FALSE)
```
**hc_xAxis**

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hc</td>
<td>A highchart htmlwidget object.</td>
</tr>
<tr>
<td>...</td>
<td>Arguments defined in <a href="https://api.highcharts.com/highcharts/tooltip">https://api.highcharts.com/highcharts/tooltip</a>.</td>
</tr>
<tr>
<td>sort</td>
<td>Logical value to implement sort according to <a href="https://stackoverflow.com/a/16954666/829971">this point</a>.</td>
</tr>
<tr>
<td>table</td>
<td>Logical value to implement table in tooltip: <a href="https://stackoverflow.com/a/22327749/829971">https://stackoverflow.com/a/22327749/829971</a>.</td>
</tr>
</tbody>
</table>

**Examples**

```r
highchart() %>%
  hc_add_series(data = sample(1:12)) %>%
  hc_add_series(data = sample(1:12) + 10) %>%
  hc_tooltip(    
    crosshairs = TRUE,    
    borderWidth = 5,    
    sort = TRUE,    
    table = TRUE    
  )
```

**hc_xAxis**

Xaxis options for highcharter objects

**Description**

The X axis or category axis. Normally this is the horizontal axis, though if the chart is inverted this is the vertical axis. In case of multiple axes, the xAxis node is an array of configuration objects. See the Axis class for programmatic access to the axis.

**Usage**

```r
hc_xAxis(hc, ...)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>hc</td>
<td>A highchart htmlwidget object.</td>
</tr>
<tr>
<td>...</td>
<td>Arguments defined in <a href="https://api.highcharts.com/highcharts/xAxis">https://api.highcharts.com/highcharts/xAxis</a>.</td>
</tr>
</tbody>
</table>

**Details**

In Highmaps, the axis is hidden, but it is used behind the scenes to control features like zooming and panning. Zooming is in effect the same as setting the extremes of one of the axes.
Examples

```r
highchart() %>%
  hc_add_series(
    data = c(7.0, 6.9, 9.5, 14.5, 18.2, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6),
    type = "spline"
  ) %>%
  hc_yAxis(
    title = list(text = "y Axis at right"),
    ...) %>%
  hc_xAxis(
    title = list(text = "x Axis at top"),
    alternateGridColor = "#FDFFD5",
    opposite = TRUE,
    plotLines = list(
      list(
        label = list(text = "This is a plotLine"),
        color = "#FF0000",
        width = 2,
        value = 5.5
      )
    )
  )
```

---

**hc_yAxis**  
*Yaxis options for highcharter objects*

**Description**

The Y axis or value axis. Normally this is the vertical axis, though if the chart is inverted this is the horizontal axis. In case of multiple axes, the yAxis node is an array of configuration objects. See the Axis object for programmatic access to the axis.

**Usage**

```r
hc_yAxis(hc, ...)
```

**Arguments**

- **hc**  
  A highchart htmlwidget object.
- **...**  

**Examples**

```r
highchart() %>%
  hc_add_series(
    data = c(7.0, 6.9, 9.5, 14.5, 18.2, 21.5, 25.2, 26.5, 23.3, 18.3, 13.9, 9.6),
    type = "spline"
  ) %>%
  hc_yAxis(
    title = list(text = "y Axis at right"),
    ...) %>%
  hc_xAxis(
    title = list(text = "x Axis at top"),
    alternateGridColor = "#FDFFD5",
    opposite = TRUE,
    plotLines = list(
      list(
        label = list(text = "This is a plotLine"),
        color = "#FF0000",
        width = 2,
        value = 5.5
      )
    )
  )
```
Creating multiples yAxes to use with highcharts

Description

The Y axis or value axis. Normally this is the vertical axis, though if the chart is inverted this is the horizontal axis. Add yAxis allows to add multiple axes with a relative height between Y axes. Based upon the relative parameter the height of each Y axis is recalculated. Otherwise the parameters are as supported by Y axis.

Usage

hc_yAxis_multiples(hc, ...)

hc xAxis_multiples(hc, ...)

hc zAxis_multiples(hc, ...)

create_axis(
  naxis = 2,
  heights = 1,
  sep = 0.01,
  offset = 0,
  turnopposite = TRUE,
  ...
)

create_yaxis(...)

hc_add_yAxis(hc, ...)
hc_zAxis

Arguments

hc A highchart htmlwidget object.

... Arguments defined in https://api.highcharts.com/highcharts/yAxis.

naxis Number of axis an integer.

heights A numeric vector. This values will be normalized.

sep A numeric value for the separation (in percentage) for the panes.

offset A numeric value (in percentage).

turnopposite A logical value to turn the side of each axis or not.

Examples

highchart() %>%
  hc_yAxis_multiples(create_axis(naxis = 2, heights = c(2, 1))) %>%
  hc_add_series(data = c(1, 3, 2), yAxis = 0) %>%
  hc_add_series(data = c(20, 40, 10), yAxis = 1)

highchart() %>%
  hc_yAxis_multiples(create_axis(naxis = 3, lineWidth = 2, title = list(text = NULL))) %>%
  hc_add_series(data = c(1, 3, 2)) %>%
  hc_add_series(data = c(20, 40, 10), type = "area", yAxis = 1) %>%
  hc_add_series(data = c(200, 400, 500), yAxis = 2) %>%
  hc_add_series(data = c(500, 300, 400), type = "areaspline", yAxis = 2)

# Retrieve stock data to plot.
aapl <- quantmod::getSymbols("AAPL",
  src = "yahoo",
  from = "2020-01-01",
  auto.assign = FALSE
)

# Plot prices and volume with relative height.
highchart(type = "stock") %>%
  hc_title(text = "AAPLE") %>%
  hc_add_series(aapl, yAxis = 0, showInLegend = FALSE) %>%
  hc_add_yAxis(nid = 1L, title = list(text = "Prices"), relative = 2) %>%
  hc_add_series(aapl[, "AAPL.Volume"], yAxis = 1, type = "column", showInLegend = FALSE) %>%
  hc_add_yAxis(nid = 2L, title = list(text = "Volume"), relative = 1)

---

hc_zAxis Zaxis options for highcharter objects

Description

The Z axis or depth axis for 3D plots. See the Axis class for programmatic access to the axis.
Usage

hc_zAxis(hc, ...)

Arguments

hc  A highchart htmlwidget object.
...  Arguments defined in https://api.highcharts.com/highcharts/zAxis.

Examples

df <- data.frame(
  x = sample(1:5),
  y = sample(1:5),
  z = sample(1:5)
)

# Note the 3d require highchart2() due have the 3d module
highchart2() %>%
  hc_add_series(data = df, "scatter3d", hcaes(x = x, y = y, z = z)) %>%
  hc_chart(
    type = "scatter3d",
    options3d = list(
      enabled = TRUE,
      alpha = 20,
      beta = 30,
      depth = 200,
      viewDistance = 5,
      frame = list(
        bottom = list(
          size = 1,
          color = "rgba(0,0,0,0.05)"
        )
      )
    )
  )
) %>%
hc_zAxis(
  title = list(text = "Z axis is here"),
  startOnTick = FALSE,
  tickInterval = 2,
  tickLength = 4,
  tickWidth = 1,
  gridLineColor = "red",
  gridLineDashStyle = "dot"
)
hex_to_rgba

Transform colors from hexadecimal format to rgba hc notation

**Description**

Transform colors from hexadecimal format to rgba hc notation

**Usage**

```r
hex_to_rgba(x, alpha = 1)
```

**Arguments**

- `x`: colors in hexadecimal format
- `alpha`: alpha

**Examples**

```r
hex_to_rgba(x <- c("#440154", "#21908C", "#FDE725"))
```

---

highchart

Create a Highcharts chart widget

**Description**

This function creates a Highchart chart using htmlwidgets. The widget can be rendered on HTML pages generated from R Markdown, Shiny, or other applications.

**Usage**

```r
highchart(  
  hc_opts = list(),  
  theme = getOption("highchart.theme"),  
  type = "chart",  
  width = NULL,  
  height = NULL,  
  elementId = NULL,  
  google_fonts = getOption("highchart.google_fonts")
)
```
Arguments

hc_opts A list object containing options defined as https://api.highcharts.com/highcharts/.

theme A hc_theme class object.

type A character value to set if use Highchart, Highstock or Highmap. Options are "chart", "stock" and "map".

width A numeric input in pixels.

height A numeric input in pixels.

elementId Use an explicit element ID for the widget.

google_fonts A boolean value. If TRUE (default), adds a reference to the Google Fonts API to the HTML head, downloading CSS for the font families defined in the Highcharts theme from https://fonts.googleapis.com. Set to FALSE if you load your own fonts using CSS. This option as default is controlled by "highchart2.google_fonts" option.

---

highchart2 Create a Highcharts chart widget

---

Description

This widgets don’t support options yet.

Usage

```r
highchart2(
  hc_opts = list(),
  theme = getOption("highchart2.theme"),
  type = "chart",
  width = NULL,
  height = NULL,
  elementId = NULL,
  google_fonts = getOption("highchart2.google_fonts")
)
```

```r
highchartzero(
  hc_opts = list(),
  theme = NULL,
  width = NULL,
  height = NULL,
  elementId = NULL
)
```
Arguments

hc_opts A list object containing options defined as https://api.highcharts.com/highcharts/.
theme A hc_theme class object.
type A character value to set if use Highchart, Highstock or Highmap. Options are "chart", "stock" and "map".
width A numeric input in pixels.
height A numeric input in pixels.
elementId Use an explicit element ID for the widget.
google_fonts A boolean value. If TRUE (default), adds a reference to the Google Fonts API to the HTML head, downloading CSS for the font families defined in the Highcharts theme from https://fonts.googleapis.com. Set to FALSE if you load your own fonts using CSS.

Details

This function creates a Highchart chart using htmlwidgets. The widget can be rendered on HTML pages generated from R Markdown, Shiny, or other applications.

Description

Highcharts https://www.highcharts.com/ is a mature javascript charting library. Highcharts provide a various type of charts, from scatters to heatmaps or treemaps.

Author(s)

Joshua Kunst (@jbkunst)

Description

The following functions are imported and then re-exported from the highcharter package to avoid listing the magrittr as Depends of highcharter.
**highchartOutput**

**Description**

Widget output function for use in Shiny

**Usage**

```r
highchartOutput(outputId, width = "100\%", height = "400px")
highchartOutput2(outputId, width = "100\%", height = "400px")
highchartOutputZ(outputId, width = "100\%", height = "400px")
```

**Arguments**

- `outputId` The name of the input.
- `width` A numeric input in pixels.
- `height` A numeric input in pixels.

**highchartProxy**

**Description**

Send commands to a Highcharts instance in a Shiny app

**Usage**

```r
highchartProxy(shinyId, session = shiny::getDefaultReactiveDomain())
```

**Arguments**

- `shinyId` Single-element character vector indicating the output ID of the chart to modify
- `session` The Shiny session object to which the map belongs; usually the default value will suffice.
Description

Chart a demo for testing themes

Usage

highcharts_demo()

Examples

highcharts_demo()

hw_grid

Lays out highchart widgets into a "grid", similar to grid.arrange from gridExtra.

Description

Lays out highchart widgets into a "grid", similar to grid.arrange from gridExtra.

Usage

hw_grid(
  ...,  
  ncol = NULL,  
  rowheight = NULL,  
  add_htmlgrid_css = TRUE,  
  browsable = TRUE
)

Arguments

... either individual highchart objects or a mixture of individual highchart objects and lists of highchart objects.
ncol how many columns in the grid
rowheight Height in px.
add_htmlgrid_css A logical value to add or not htmlgrid.css as dependency.
browsable Logical value indicating if the returned object is converted to an HTML object browsable using htmltools::browsable.
Examples

```r
charts <- lapply(1:9, function(x) {
  hchart(ts(cumsum(rnorm(100))))
})

if (interactive()) {
  hw_grid(charts, rowheight = 300)
}
```

is.hexcolor

Check if a string vector is in hexadecimal color format

Description

Check if a string vector is in hexadecimal color format

Usage

```r
is.hexcolor(x)
```

Arguments

- `x` A string vectors

Examples

```r
x <- c("#f0f0f0", "#FFF", "#99990000", "#00FFFFFF")

is.hexcolor(x)
```

is.highchart

Reports whether x is a highchart object

Description

Reports whether x is a highchart object

Usage

```r
is.highchart(x)
```

Arguments

- `x` An object to test
list_parse

Convert an object to list with identical structure

Description

This function is similar to rlist::list.parse but this removes names. NAs are removed for compatibility with rjson::toJSON.

Usage

list_parse(df)

list_parse2(df)

Arguments

df A data frame to parse to list

Examples

x <- data.frame(a = 1:3, type = c("A", "C", "B"), stringsAsFactors = FALSE)
list_parse(x)
list_parse2(x)

mountains_panorama

Visual comparison of Mountains Panorama

Description

This data comes from the https://www.highcharts.com/ examples: https://www.highcharts.com/demo/3d-area-multiple

Usage

mountains_panorama

Format

A data frame with 91 observations and 3 variables.

Variables

- place: The place.
- name: Name.
- heigth: Heigth.
mutate_mapping

Description
Modify data frame according to mapping

Usage
mutate_mapping(data, mapping, drop = FALSE)

Arguments
- data: A data frame object.
- mapping: A mapping from hcaes function.
- drop: A logical argument to drop variables or not. Default is FALSE

Examples

df <- head(mtcars)
mutate_mapping(data = df, mapping = hcaes(x = cyl, y = wt + cyl, group = gear))
mutate_mapping(data = df, mapping = hcaes(x = cyl, y = wt), drop = TRUE)

pokemon

Description
Information about 898 pokemon.

Usage
pokemon

Format
A data frame with 898 observations and 24 variables.
random_id

*Function to generate iids*

**Description**

Function to generate iids

**Usage**

random_id(n = 1, length = 10)

**Arguments**

- `n` Number of ids
- `length` Length of ids

---

renderHighchart

*Widget render function for use in Shiny*

**Description**

Widget render function for use in Shiny

**Usage**

renderHighchart(expr, env = parent.frame(), quoted = FALSE)

renderHighchart2(expr, env = parent.frame(), quoted = FALSE)

renderHighchartZ(expr, env = parent.frame(), quoted = FALSE)

**Arguments**

- `expr` A highchart expression.
- `env` A environment.
- `quoted` A boolean value.
**stars**

---

**stars**

---

**Description**

A sample using by Nadieh Bremer blocks. [http://bl.ocks.org/nbremer/eb0d1fd4118b731d069e2ff98dfadc47](http://bl.ocks.org/nbremer/eb0d1fd4118b731d069e2ff98dfadc47).

**Usage**

```r
codes
```

**Format**

A data frame with 404 observations and 6 variables.

**Variables**

- `bv`: BV
- `absmag`: Magnitude
- `lum`: Luminosity
- `temp`: Temperature
- `radiussun`: Radius
- `distance`: Distance

---

**str_to_id**

*String to 'id' format*

---

**Description**

Turn a string to id format used in treemaps.

**Usage**

```r
str_to_id(x)
str_to_id_vec(x)
```

**Arguments**

- `x` A vector string.

**Examples**

```r
str_to_id("A string _ with sd / sdg Underscores ")
```
tooltip_chart

Helper to create charts in tooltips.

Description
Helper to create charts in tooltips.

Usage
tooltip_chart(accesor = NULL, hc_opts = NULL, width = 250, height = 150)

Arguments
- `accesor`: A string indicating the name of the column where the data is.
- `width`: A numeric input in pixels indicating the width of the tooltip.
- `height`: A numeric input in pixels indicating the height of the tooltip.

Details
This function needs to be used in the `pointFormatter` argument inside of `hc_tooltip` function and `useHTML = TRUE` option.

Examples
```
require(dplyr)
require(purrr)
require(tidyr)
require(gapminder)
data(gapminder, package = "gapminder")

gp <- gapminder %>%
  arrange(desc(year)) %>%
  distinct(country, .keep_all = TRUE)

gp2 <- gapminder %>%
  nest(-country) %>%
  mutate(
    data = map(data, mutate_mapping, hcaes(x = lifeExp, y = gdpPercap), drop = TRUE),
    data = map(data, list_parse)
  ) %>%
  rename(ttdata = data)

gptot <- left_join(gp, gp2)

hc <- hchart(
```
```r
gptot,
"point",
hcaes(
  lifeExp,
  gdpPercap,
  name = country,
  size = pop,
  group = continent
)
) %>%
hc_yAxis(type = "logarithmic")

hc %>%
hc_tooltip(useHTML = TRUE, pointFormatter = tooltip_chart(accesor = "ttdata"))

hc %>%
hc_tooltip(useHTML = TRUE, pointFormatter = tooltip_chart(
  accesor = "ttdata",
  hc_opts = list(chart = list(type = "column"))
))

hc %>%
hc_tooltip(
  useHTML = TRUE,
  positioner = JS("function () { return { x: this.chart.plotLeft + 10, y: 10}; }"),
  pointFormatter = tooltip_chart(
    accesor = "ttdata",
    hc_opts = list(
      title = list(text = "point.country"),
      xAxis = list(title = list(text = "lifeExp")),
      yAxis = list(title = list(text = "gdpPercap"))
    )
  )
)

hc %>%
hc_tooltip(
  useHTML = TRUE,
  pointFormatter = tooltip_chart(
    accesor = "ttdata",
    hc_opts = list(
      legend = list(enabled = TRUE),
      series = list(list(color = "gray", name = "point.name"))
    )
  )
)

tooltip_table

Helper for make table in tooltips
```
Description

Helper to make table in tooltips for the pointFormat parameter in hc_tooltip

Usage

tooltip_table(x, y, title = NULL, img = NULL, ...)

Arguments

x A string vector with description text
y A string with accessors example: point.series.name, point.x
title A title tag with accessors or string
img Image tag
... html attributes for the table element

Examples

x <- c("Income:", "Genre", "Runtime")
y <- c(
"\${point.y}", "{point.series.options.extra.genre}",
"{point.series.options.extra.runtime}" 
)

tooltip_table(x, y)

unemployment US Counties unemployment rate

Description

This data comes from the highcharts and is used in highmaps examples.

Usage

unemployment

Format

A data.frame with 3 variables and 3.216 observations.

Variables

- code: The county code.
- name: The county name.
- value: The unemployment.
Description

This data comes from the https://code.highcharts.com/mapdata/countries/us/us-all-all.js and is used in highmaps examples.

Usage

uscountygeojson

Format

A list in geojson format.

Description

This data comes from the https://code.highcharts.com/mapdata/countries/us/us-all.js and is used in highmaps examples.

Usage

usgeojson

Format

A list in geojson format.
vaccines  

**Vaccines**

**Description**


**Usage**

vaccines

**Format**

A data frame with 3,876 observations and 3 variables.

**Variables**

- `year`: Year
- `state`: Name of the state
- `count`: Number of cases per 100,000 people. If the value is NA the count was 0.

weather  

**Weather**

**Description**

Temperature information of San Francisco.

**Usage**

weather

**Format**

A data frame with 365 observations and 4 variables.

**Variables**

- `date`: Day in date format.
- `min_temperaturec`: Minimum temperature.
- `max_temperaturec`: Maximum temperature.
- `mean_temperaturec`: Mean temperature.
worldgeojson

| worldgeojson | World map in Geojson format (list) |

**Description**

This data comes from the [https://code.highcharts.com/mapdata/custom/world.js](https://code.highcharts.com/mapdata/custom/world.js) and is used in highmaps examples.

**Usage**

worldgeojson

**Format**

A list in geojson format.
Index

* datasets
  citytemp, 5  
citytemp_long, 6  
favorite_bars, 13  
favorite_pies, 14  
globaltemp, 15  
mountains_panorama, 90  
pokemon, 91  
stars, 93  
unemployment, 96  
uscountygeojson, 97  
usgeojson, 97  
vaccines, 98  
weather, 98  
worldgeojson, 99

%>% (highcharter-exports), 86

citytemp, 5  
citytemp_long, 6  
color_classes, 7  
color_stops, 7  
colorize, 6  
create_axis (hc_yAxis_multiples), 81  
create_yaxis (hc_yAxis_multiples), 81

data_to_boxplot, 8  
data_to_hierarchical, 9  
data_to_sankey, 10  
datetime_to_timestamp, 10  
df_to_annotations_labels, 11  
download_map_data, 11, 14  
dt_tstp (datetime_to_timestamp), 10

export_hc, 12

fa_icon (hc_add_dependency_fa), 31  
fa_icon_mark (hc_add_dependency_fa), 31  
favorite_bars, 13  
favorite_pies, 14  
get_data_from_map, 14  

get_hc_series_from_df, 15  
globaltemp, 15

hc_add_annotation, 30  
hc_add_annotations (hc_add_annotation), 30  
hc_add_dependency, 31  
hc_add_dependency_fa, 31  
hc_add_event_point, 32  
hc_add_event_series  
  (hc_add_event_point), 32  
hc_add_series, 33  
hc_add_series.character, 33  
hc_add_series.data_frame, 34  
hc_add_series.density, 34  
hc_add_series.factor  
  (hc_add_series.character), 33  
hc_add_series.forecast, 35  
hc_add_series.geo_json, 35  
hc_add_series.geo_list  
  (hc_add_series.geo_json), 35  
hc_add_series.lm, 36  
hc_add_series.loess (hc_add_series.lm), 36  
hc_add_series.numeric, 37  
hc_add_series.ohlc (hc_add_series.xts), 38  
hc_add_series.ts, 37  
hc_add_series.xts, 38  
hc_add_series_list, 38  
hc_add_series_map, 39  
hc_add_theme, 40  
hc_add_yAxis (hc_yAxis_multiples), 81  
hc_annotations, 41  
hc_boost, 42  
hc_caption, 44  
hc_chart, 45  
hc_colorAxis, 46  
hc_colors, 48  
hc_credits, 49
list_parse2(list_parse), 90
mountains_panorama, 90
mutate_mapping, 91
pokemon, 91
random_id, 92
renderHighchart, 92
renderHighchart2(renderHighchart), 92
renderHighchartZ(renderHighchart), 92
stars, 93
str_to_id, 93
str_to_id_vec(str_to_id), 93
tagshighcharter-exports, 86
tooltip_chart, 94
tooltip_table, 95
unemployment, 96
uscountygeojson, 97
usgeojson, 97
vaccines, 98
weather, 98
worldgeojson, 99