Package 'rAmCharts4'

October 14, 2022

Title Interface to the JavaScript Library 'amCharts 4'

Version 1.6.0

Maintainer Stéphane Laurent <laurent_step@outlook.fr>

Description Creates JavaScript charts. The charts can be included in 'Shiny' apps and R markdown documents, or viewed from the R console and 'RStudio' viewer. Based on the JavaScript library 'amCharts 4' and the R packages 'htmlwidgets' and 'reactR'. Currently available types of chart are: vertical and horizontal bar chart, radial bar chart, stacked bar chart, vertical and horizontal Dumbbell chart, line chart, scatter chart, range area chart, gauge chart, boxplot chart, pie chart, and 100% stacked bar chart.

URL <https://github.com/stla/rAmCharts4>

BugReports <https://github.com/stla/rAmCharts4/issues>

License GPL-3

Encoding UTF-8

Imports htmltools, htmlwidgets (>= 1.5.3), reactR, shiny, jsonlite, lubridate, minpack.lm, tools, base64enc, xml2, stringr, stats, grDevices

Suggests reshape2

RoxygenNote 7.2.1

NeedsCompilation no

Author Stéphane Laurent [aut, cre],
Antanas Marcelionis [ctb, cph] ('amCharts' library
(https://www.amcharts.com/)),
Terence Eden [ctb, cph] ('SuperTinyIcons' library
(https://github.com/edent/SuperTinyIcons/)),
Tom Alexander [ctb, cph] ('regression-js' library
(https://github.com/Tom-Alexander/regression-js))

Repository CRAN

Date/Publication 2022-09-22 10:10:02 UTC

R **topics documented:**

---

amAxisBreaks   *Axis breaks*

Description

Create an object defining the breaks on an axis.

Usage

```
amAxisBreaks(
  values = NULL,
  labels = NULL,
  interval = NULL,
  timeInterval = NULL
)
```

Arguments

values	positions of the breaks, a vector of values; for a date axis, this must be a vector of dates
labels	if values is given, the labels of the breaks; if NULL, the labels are set to the values
interval	for equally spaced breaks, the number of pixels between two consecutive breaks; ignored if values is given
timeInterval	for equally spaced breaks on a date axis, this option defines the interval between two consecutive breaks; it must be a string like "1 day", "7 days", "1 week", "2 months", ...; ignored if values or interval is given

---

amAxisLabels   *Axis labels*

Description

Create a list of settings for the labels of an axis.

Usage

```
amAxisLabels(
  color = NULL,
  fontSize = 18,
  fontWeight = "normal",
  fontFamily = NULL,
  rotation = 0,
  formatter = NULL
)
```

```
amAxisLabelsCircular(
  color = NULL,
  fontSize = 14,
  fontWeight = "normal",
  fontFamily = NULL,
  radius = NULL,
  relativeRotation = NULL
)
```

Arguments

`color`	color of the labels
`fontSize`	size of the labels
`fontWeight`	font weight of the labels, it can be `"normal"`, `"bold"`, `"bolder"`, `"lighter"`, or a number in `seq(100, 900, by = 100)`
`fontFamily`	font family of the labels
`rotation`	rotation angle
`formatter`	this option defines the format of the axis labels; this should be a number formatting string for a numeric axis, and a list created with `amDateAxisFormatter` for a date axis
`radius`	radius in percentage
`relativeRotation`	
	relative rotation angle

Value

A list of settings for the labels of an axis.

Note

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

---

amBarChart *HTML widget displaying a bar chart*

Description

Create a HTML widget displaying a bar chart.

Usage

```
amBarChart(
  data,
  data2 = NULL,
  category,
  values,
  valueNames = NULL,
  showValues = TRUE,
  hline = NULL,
  yLimits = NULL,
  expandY = 5,
  valueFormatter = "#.",
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  draggable = FALSE,
  tooltip = NULL,
  columnStyle = NULL,
  threeD = FALSE,
  bullets = NULL,
  alwaysShowBullets = FALSE,
  backgroundColor = NULL,
  cellWidth = NULL,
  columnWidth = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  legend = NULL,
  caption = NULL,
  image = NULL,
  button = NULL,
  cursor = FALSE,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

data	a dataframe
data2	NULL or a dataframe used to update the data with the button; its column names must include the column names of data given in values, it must have the same number of rows as data and its rows must be in the same order as those of data
category	name of the column of data to be used on the category axis

values	name(s) of the column(s) of data to be used on the value axis
valueNames	names of the values variables, to appear in the legend; NULL to use values as names, otherwise a named list of the form list(value1 = "ValueName1", value2 = "ValueName2", ...) where value1, value2, ... are the column names given in values and "ValueName1", "ValueName2", ... are the desired names to appear in the legend; these names can also appear in the tooltips: they are substituted to the string {name} in the formatting string passed on to the tooltip (see the second example)
showValues	logical, whether to display the values on the chart
hline	an optional horizontal line to add to the chart; it must be a named list of the form list(value = h, line = settings) where h is the "intercept" and settings is a list of settings created with amLine
yLimits	range of the y-axis, a vector of two values specifying the lower and the upper limits of the y-axis; NULL for default values
expandY	if yLimits = NULL, a percentage of the range of the y-axis used to expand this range
valueFormatter	a number formatting string; it is used to format the values displayed on the chart if showValues = TRUE, the values displayed in the cursor tooltips if cursor = TRUE, the labels of the y-axis unless you specify your own formatter in the labels field of the list passed on to the yAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example for the way to set a number formatter in the tooltip text)
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
draggable	TRUE/FALSE to enable/disable dragging of all bars, otherwise a named list of the form list(value1 = TRUE, value2 = FALSE, ...) to enable/disable the dragging for each bar corresponding to a column given in values
tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amTooltip; this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip
columnStyle	settings of the columns (the bars); NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amColumn; this can also be a single list of settings that will be applied to each column
threeD	logical, whether to render the columns in 3D
bullets	settings of the bullets; NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amCircle, amTriangle or amRectangle; this can also be a single list of settings that will be applied to each series

alwaysShowBullets

              logical, whether to always show the bullets; if FALSE, the bullets are shown only on hovering a column

backgroundColor

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

cellWidth   cell width in percent; for a simple bar chart, this is the width of the columns; for a grouped bar chart, this is the width of the clusters of columns; NULL for the default value

columnWidth   column width, a percentage of the cell width; set to 100 for a simple bar chart and use cellWidth to control the width of the columns; for a grouped bar chart, this controls the spacing between the columns within a cluster of columns; NULL for the default value

xAxis   settings of the category axis given as a list, or just a string for the axis title; the list of settings has three possible fields: a field title, a list of settings for the axis title created with [amText](), a field labels, a list of settings for the axis labels created with [amAxisLabels](), and a field adjust, a number defining the vertical adjustment of the axis (in pixels)

yAxis   settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with [amText](), a field labels, a list of settings for the axis labels created with [amAxisLabels](), a field adjust, a number defining the horizontal adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with [amLine]() and a field breaks to control the axis breaks, an R object created with [amAxisBreaks]()

scrollbarX   logical, whether to add a scrollbar for the category axis

scrollbarY   logical, whether to add a scrollbar for the value axis

legend   either a logical value, whether to display the legend, or a list of settings for the legend created with [amLegend]()

caption   NULL or FALSE for no caption, a formatted text created with [amText](), or a list with two fields: text, a list created with [amText](), and align, can be "left", "right" or "center"

image   option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with [amImage](), the field position can be "topleft", "topright", "bottomleft" or "bottomright", the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment

button   NULL for the default, FALSE for no button, or a list of settings created with [amButton](); this button is used to replace the current data with data2

cursor   option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor with default settings for the tooltips, or a list of settings created with [amTooltip]() to set the style of the tooltips, or a list with three possible fields: a field tooltip, a list of tooltip settings created with [amTooltip](), a field extraTooltipPrecision, an

integer, the number of additional decimals to display in the tooltips, and a field
`modifier`, which defines a modifier for the values displayed in the tooltips; a
modifier is some JavaScript code given as a string, which performs a modifica-
tion of a string named `text`, e.g. `modifier = "text = '>>>' + text;"`

width   the width of the chart, e.g. `"600px"` or `"80%"`; ignored if the chart is displayed
                    in Shiny, in which case the width is given in amChart4Output

height   the height of the chart, e.g. `"400px"`; ignored if the chart is displayed in Shiny,
                    in which case the height is given in amChart4Output

export   logical, whether to enable the export menu

chartId   a HTML id for the chart

elementId   a HTML id for the container of the chart; ignored if the chart is displayed in
                    Shiny, in which case the id is given by the Shiny id

Examples

```
# a simple bar chart ####

dat <- data.frame(
  country = c("USA", "China", "Japan", "Germany", "UK", "France"),
  visits = c(3025, 1882, 1809, 1322, 1122, 1114)
)

amBarChart(
  data = dat, data2 = dat,
  width = "600px",
  category = "country", values = "visits",
  draggable = TRUE,
  tooltip =
    "[bold font-style:italic #ffff00]{valueY.value.formatNumber('#,###.')}[/]",
  chartTitle =
    amText(text = "Visits per country", fontSize = 22, color = "orangered"),
  xAxis = list(title = amText(text = "Country", color = "maroon")),
  yAxis = list(
    title = amText(text = "Visits", color = "maroon"),
    gridLines = amLine(color = "orange", width = 1, opacity = 0.4)
  ),
  yLimits = c(0, 4000),
  valueFormatter = "#,###.",
  caption = amText(text = "Year 2018", color = "red"),
  theme = "material")


# bar chart with individual images in the bullets ####

dat <- data.frame(
  language = c("Python", "Julia", "Java"),
  users = c(10000, 2000, 5000),
  href = c(
    tinyIcon("python", "transparent"),
    tinyIcon("julia", "transparent"),
```

```
      tinyIcon("java", "transparent")
    )
  )

  amBarChart(
    data = dat,
    width = "700px",
    category = "language",
    values = "users",
    valueNames = list(users = "#users"),
    showValues = FALSE,
    tooltip = amTooltip(
      text = "{name}: [bold]valueY[/]",
      textColor = "white",
      backgroundColor = "#101010",
      borderColor = "silver"
    ),
    draggable = FALSE,
    backgroundColor = "seashell",
    bullets = amCircle(
      radius = 30,
      color = "white",
      strokeWidth = 4,
      image = amImage(
        href = "inData:href",
        width = 50, height = 50
      )
    ),
    alwaysShowBullets = TRUE,
    xAxis = list(title = amText(text = "Programming language")),
    yAxis = list(
      title = amText(text = "# users"),
      gridLines = amLine(color = "orange", width = 1, opacity = 0.4)
    ),
    yLimits = c(0, 12000),
    valueFormatter = "#.",
    theme = "material")


# a grouped bar chart ####

set.seed(666)
dat <- data.frame(
  country = c("USA", "China", "Japan", "Germany", "UK", "France"),
  visits = c(3025, 1882, 1809, 1322, 1122, 1114),
  income = rpois(6, 25),
  expenses = rpois(6, 20)
)

amBarChart(
  data = dat,
  width = "700px",
  category = "country",
```

```
      values = c("income", "expenses"),
      valueNames = list(income = "Income", expenses = "Expenses"),
      tooltip = amTooltip(
        textColor = "white",
        backgroundColor = "#101010",
        borderColor = "silver"
      ),
      draggable = list(income = TRUE, expenses = FALSE),
      backgroundColor = "#30303d",
      columnStyle = list(
        income = amColumn(
          color = "darkmagenta",
          strokeColor = "#cccccc",
          strokeWidth = 2
        ),
        expenses = amColumn(
          color = "darkred",
          strokeColor = "#cccccc",
          strokeWidth = 2
        )
      ),
      chartTitle = amText(text = "Income and expenses per country"),
      xAxis = list(title = amText(text = "Country")),
      yAxis = list(
        title = amText(text = "Income and expenses"),
        gridLines = amLine(color = "whitesmoke", width = 1, opacity = 0.4),
        breaks = amAxisBreaks(values = seq(0, 45, by = 5))
      ),
      yLimits = c(0, 45),
      valueFormatter = "#.#",
      caption = amText(text = "Year 2018"),
      theme = "dark")
```

---

amBoxplotChart   *HTML widget displaying a boxplot chart*

Description

Create a HTML widget displaying a boxplot chart.

Usage

```
amBoxplotChart(
  data,
  category,
  value,
  color = NULL,
  hline = NULL,
  yLimits = NULL,
  expandY = 5,
```

```
  valueFormatter = "#.",
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  tooltip = TRUE,
  bullets = NULL,
  backgroundColor = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  caption = NULL,
  image = NULL,
  cursor = FALSE,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

data	a dataframe
category	name of the column of data to be used for the category axis; this can be a date column
value	name of the column of data to be used for the value axis
color	the color of the boxplots; it can be given by the name of a R color, the name of a CSS color, e.g. "crimson" or "fuchsia", a HEX code like "#FF009A", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
hline	an optional horizontal line to add to the chart; it must be a named list of the form list(value = h, line = settings) where h is the "intercept" and settings is a list of settings created with amLine
yLimits	range of the y-axis, a vector of two values specifying the lower and the upper limits of the y-axis; NULL for default values
expandY	if yLimits = NULL, a percentage of the range of the y-axis used to expand this range
valueFormatter	a number formatting string; it is used to format the values displayed in the cursor tooltips, the labels of the y-axis unless you specify your own formatter in the labels field of the list passed on to the yAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example of amBarChart for the way to set a number formatter in the tooltip text)
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"

animated	Boolean, whether to animate the rendering of the graphic
tooltip	TRUE for the default tooltips, FALSE for no tooltip, otherwise a string for the text to display in the tooltip
bullets	settings of the bullets representing the outliers; NULL for default, otherwise a list created with amCircle, amTriangle or amRectangle
backgroundColor	
	a color for the chart background; it can be given by the name of a R color, the name of a CSS color, e.g. "lime" or "olive", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
xAxis	settings of the category axis given as a list, or just a string for the axis title; the list of settings has four possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the vertical adjustment of the axis (in pixels), and a field gridLines, a list of settings for the grid lines created with amLine
yAxis	settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the horizontal adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with amLine and a field breaks to control the axis breaks, an R object created with amAxisBreaks
scrollbarX	logical, whether to add a scrollbar for the category axis
scrollbarY	logical, whether to add a scrollbar for the value axis
caption	NULL or FALSE for no caption, a formatted text created with amText, or a list with two fields: text, a list created with amText, and align, can be "left", "right" or "center"
image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with amImage, the field position can be "topleft", "topright", "bottomleft" or "bottomright", the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment
cursor	option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor with default settings for the tooltips, or a list of settings created with amTooltip to set the style of the tooltips, or a list with three possible fields: a field tooltip, a list of tooltip settings created with amTooltip, a field extraTooltipPrecision, an integer, the number of additional decimals to display in the tooltips, and a field modifier, which defines a modifier for the values displayed in the tooltips; a modifier is some JavaScript code given as a string, which performs a modification of a string named text, e.g. modifier = "text = '>>>' + text;"
width	the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in amChart4Output
height	the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in amChart4Output
export	logical, whether to enable the export menu

chartId	a HTML id for the chart
elementId	a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
library(rAmCharts4)
set.seed(666)
dat <- data.frame(
  group = gl(4, 50, labels = c("A", "B", "C", "D")),
  y     = rt(200, df = 3)
)
amBoxplotChart(
  dat,
  category = "group",
  value = "y",
  color = "maroon",
  valueFormatter = "#.#",
  theme = "moonrisekingdom"
)
```

---

amButton   *Button*

Description

Create a list of settings for a button.

Usage

```
amButton(label, color = NULL, position = 0.9, marginRight = 10)
```

Arguments

label	label of the button, a character string or a list created with amText for a formatted label
color	button color
position	the vertical position of the button: 0 for bottom, 1 for top
marginRight	right margin in pixels

Value

A list of settings for a button.

---

amColumn   *Columns style*

Description

Create a list of settings for the columns of a bar chart.

Usage

```
amColumn(
  color = NULL,
  opacity = NULL,
  strokeColor = NULL,
  strokeWidth = 4,
  cornerRadius = 8
)
```

Arguments

color	color of the columns; this can be a color adapter
opacity	opacity of the columns, a number between 0 and 1
strokeColor	color of the border of the columns; this can be a color adapter
strokeWidth	width of the border of the columns
cornerRadius	radius of the corners of the columns

Value

A list of settings for usage in `amBarChart` or `amHorizontalBarChart`

Note

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

---

amDateAxisFormatter   *Date axis formatter*

Description

Create a list of settings for formatting the labels of a date axis, to be passed on to the `formatter` argument of `amAxisLabels`.

Usage

```
amDateAxisFormatter(
  day = c("dd", "MMM dd"),
  week = c("dd", "MMM dd"),
  month = c("MMM", "MMM yyyy")
)
```

Arguments

`day, week, month`

> vectors of length two, the first component is a formatting string for the dates within a period, and the second one is a formatting string for the dates at a period change; see Formatting date and time

Value

A list of settings for formatting the labels of a date axis.

---

amDumbbellChart   *HTML widget displaying a Dumbbell chart*

Description

Create a HTML widget displaying a Dumbbell chart.

Usage

```
amDumbbellChart(
  data,
  data2 = NULL,
  category,
  values,
  seriesNames = NULL,
  hline = NULL,
  yLimits = NULL,
  expandY = 5,
  valueFormatter = "#.",
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  draggable = FALSE,
  tooltip = NULL,
  segmentsStyle = NULL,
  bullets = NULL,
  backgroundColor = NULL,
  xAxis = NULL,
  yAxis = NULL,
```

```
    scrollbarX = FALSE,
    scrollbarY = FALSE,
    legend = NULL,
    caption = NULL,
    image = NULL,
    button = NULL,
    cursor = FALSE,
    width = NULL,
    height = NULL,
    export = FALSE,
    chartId = NULL,
    elementId = NULL
)
```

Arguments

data	a dataframe
data2	NULL or a dataframe used to update the data with the button; its column names must include the column names of data given in values, it must have the same number of rows as data and its rows must be in the same order as those of data
category	name of the column of data to be used for the category axis
values	a character matrix with two columns; each row corresponds to a series and provides the names of two columns of data to be used as the limits of the segments
seriesNames	a character vector providing the names of the series to appear in the legend; its length must equal the number of rows of the values matrix: the n-th component corresponds to the n-th row of the values matrix
hline	an optional horizontal line to add to the chart; it must be a named list of the form list(value = h, line = settings) where h is the "intercept" and settings is a list of settings created with amLine
yLimits	range of the y-axis, a vector of two values specifying the lower and the upper limits of the y-axis; NULL for default values
expandY	if yLimits = NULL, a percentage of the range of the y-axis used to expand this range
valueFormatter	a number formatting string; it is used to format the values displayed in the cursor tooltips, the labels of the y-axis unless you specify your own formatter in the labels field of the list passed on to the yAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example of amBarChart for the way to set a number formatter in the tooltip text)
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
draggable	TRUE/FALSE to enable/disable dragging of all bullets, otherwise a named list of the form list(value1 = TRUE, value2 = FALSE, ...)

tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with [amTooltip](#); this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip
segmentsStyle	settings of the segments; NULL for default, otherwise a named list of the form list(series1 = settings1, series2 = settings2, ...) where series1, series2, ... are the names of the series provided in seriesNames and settings1, settings2, ... are lists created with [amSegment](#); this can also be a single list of settings that will be applied to each series
bullets	settings of the bullets; NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with [amCircle](#), [amTriangle](#) or [amRectangle](#); this can also be a single list of settings that will be applied to each series
backgroundColor	
	a color for the chart background; it can be given by the name of a R color, the name of a CSS color, e.g. ″lime″ or ″olive″, an HEX code like ″#ff009a″, a RGB code like ″rgb(255,100,39)″, or a HSL code like ″hsl(360,11,255)″
xAxis	settings of the category axis given as a list, or just a string for the axis title; the list of settings has four possible fields: a field title, a list of settings for the axis title created with [amText](#), a field labels, a list of settings for the axis labels created with [amAxisLabels](#), a field adjust, a number defining the vertical adjustment of the axis (in pixels), and a field gridLines, a list of settings for the grid lines created with [amLine](#)
yAxis	settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with [amText](#), a field labels, a list of settings for the axis labels created with [amAxisLabels](#), a field adjust, a number defining the horizontal adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with [amLine](#) and a field breaks to control the axis breaks, an R object created with [amAxisBreaks](#)
scrollbarX	logical, whether to add a scrollbar for the category axis
scrollbarY	logical, whether to add a scrollbar for the value axis
legend	either a logical value, whether to display the legend, or a list of settings for the legend created with [amLegend](#)
caption	NULL or FALSE for no caption, a formatted text created with [amText](#), or a list with two fields: text, a list created with [amText](#), and align, can be ″left″, ″right″ or ″center″
image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with [amImage](#), the field position can be ″topleft″, ″topright″, ″bottomleft″ or ″bottomright″, the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment
button	NULL for the default, FALSE for no button, or a list of settings created with [amButton](#); this button is used to replace the current data with data2

cursor   option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor with default settings for the tooltips, or a list of settings created with amTooltip to set the style of the tooltips, or a list with three possible fields: a field tooltip, a list of tooltip settings created with amTooltip, a field extraTooltipPrecision, an integer, the number of additional decimals to display in the tooltips, and a field modifier, which defines a modifier for the values displayed in the tooltips; a modifier is some JavaScript code given as a string, which performs a modification of a string named text, e.g. modifier = "text = '>>>' + text;"

width   the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in amChart4Output

height   the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in amChart4Output

export   logical, whether to enable the export menu

chartId   a HTML id for the chart

elementId   a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
dat <- data.frame(
  x = c("T0", "T1", "T2"),
  y1 = c(7, 15, 10),
  y2 = c(20, 25, 23),
  z1 = c(5, 10, 5),
  z2 = c(25, 20, 15)
)

amDumbbellChart(
  width = "500px",
  data = dat,
  draggable = TRUE,
  category = "x",
  values = rbind(c("y1","y2"), c("z1","z2")),
  seriesNames = c("Control", "Treatment"),
  yLimits = c(0, 30),
  segmentsStyle = list(
    "Control" = amSegment(width = 2),
    "Treatment" = amSegment(width = 2)
  ),
  bullets = list(
    y1 = amTriangle(strokeWidth = 0),
    y2 = amTriangle(rotation = 180, strokeWidth = 0),
    z1 = amTriangle(strokeWidth = 0),
    z2 = amTriangle(rotation = 180, strokeWidth = 0)
  ),
  tooltip = amTooltip("upper: {openValueY}\nlower: {valueY}", scale = 0.75),
  xAxis = list(
    title = amText(
      "timepoint",
      fontSize = 17, fontWeight = "bold", fontFamily = "Helvetica"
```

```
        )
      ),
      yAxis = list(
        title = amText(
          "response",
          fontSize = 17, fontWeight = "bold", fontFamily = "Helvetica"
        ),
        gridLines = amLine("silver", width = 1, opacity = 0.4)
      ),
      legend = amLegend(position = "right", itemsWidth = 15, itemsHeight = 15),
      backgroundColor = "lightyellow",
      theme = "dataviz"
  )
```

---

amFont   *Font*

Description

Create a list of settings for a font.

Usage

```
amFont(fontSize = NULL, fontWeight = "normal", fontFamily = NULL)
```

Arguments

fontSize	font size, must be given as a character string like "10px" or "2em", or a numeric value, the font size in pixels
fontWeight	font weight, it can be "normal", "bold", "bolder", "lighter", or a number in seq(100, 900, by = 100)
fontFamily	font family

Value

A list of settings for a font.

Note

There is no option for the font style.

amGaugeChart   *HTML widget displaying a gauge chart*

Description

Create a HTML widget displaying a gauge chart.

Usage

```
amGaugeChart(
  score,
  minScore,
  maxScore,
  scorePrecision = 0,
  gradingData,
  innerRadius = 70,
  labelsRadius = (100 - innerRadius)/2,
  axisLabelsRadius = 19,
  chartFontSize = 11,
  labelsFont = amFont(fontSize = "2em", fontWeight = "bold"),
  axisLabelsFont = amFont(fontSize = "1.2em"),
  scoreFont = amFont(fontSize = "6em"),
  scoreLabelFont = amFont(fontSize = "2em"),
 hand = amHand(innerRadius = 45, width = 8, color = "slategray", strokeColor = "black"),
  gridLines = FALSE,
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  backgroundColor = NULL,
  caption = NULL,
  image = NULL,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

score	gauge value, a number between `minScore` and `maxScore`
minScore	minimal score
maxScore	maximal score
scorePrecision	an integer, the number of decimals of the score to be displayed

gradingData	data for the gauge, a dataframe with three required columns: label, lowScore, and highScore, and an optional column color; if the column color is not present, then the colors will be derived from the theme
innerRadius	inner radius of the gauge given as a percentage, between 0 (the gauge has no width) and 100 (the gauge is a semi-disk)
labelsRadius	radius for the labels given as a percentage; use the default value to get centered labels
axisLabelsRadius	
	radius for the axis labels given as a percentage
chartFontSize	reference font size, a numeric value, the font size in pixels; this font size has an effect only if you use the relative CSS unit em for other font sizes
labelsFont	a list of settings for the font of the labels created with [amFont](#), but the font size must be given in pixels or in em CSS units (no other units are accepted)
axisLabelsFont	a list of settings for the font of the axis labels created with [amFont](#)
scoreFont	a list of settings for the font of the score created with [amFont](#)
scoreLabelFont	a list of settings for the font of the score label created with [amFont](#)
hand	a list of settings for the hand of the gauge created with [amHand](#)
gridLines	a list of settings for the grid lines created with [amLine](#), or a logical value: FALSE for no grid lines, TRUE for default grid lines
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with [amText](#), or a list with two fields: text, a list of settings created with [amText](#), and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
backgroundColor	
	a color for the chart background; it can be given by the name of a R color, the name of a CSS color, e.g. "aqua" or "indigo", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
caption	NULL or FALSE for no caption, a formatted text created with [amText](#), or a list with two fields: text, a list created with [amText](#), and align, can be "left", "right" or "center"
image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with [amImage](#), the field position can be "topleft", "topright", "bottomleft" or "bottomright", the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment
width	the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in [amChart4Output](#)
height	the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in [amChart4Output](#)
export	logical, whether to enable the export menu
chartId	a HTML id for the chart
elementId	a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Note

In Shiny, you can change the score of a gauge chart with the help of updateAmGaugeChart.

Examples

```
library(rAmCharts4)

gradingData <- data.frame(
  label = c("Slow", "Moderate", "Fast"),
  color = c("blue", "green", "red"),
  lowScore = c(0, 100/3, 200/3),
  highScore = c(100/3, 200/3, 100)
)

amGaugeChart(
  score = 40, minScore = 0, maxScore = 100, gradingData = gradingData
)
```

---

amHand   *Gauge hand*

Description

Create a list of settings for the hand of a gauge chart.

Usage

```
amHand(innerRadius, width, color, strokeColor)
```

Arguments

innerRadius	inner radius of the hand, given as a percentage
width	width of the base of the hand in pixels, a positive number
color	color of the hand
strokeColor	stroke color of the hand

Value

A list of settings for the hand of a gauge chart.

---

amHorizontalBarChart   *HTML widget displaying a horizontal bar chart*

Description

Create a HTML widget displaying a horizontal bar chart.

Usage

```
amHorizontalBarChart(
  data,
  data2 = NULL,
  category,
  values,
  valueNames = NULL,
  showValues = TRUE,
  vline = NULL,
  xLimits = NULL,
  expandX = 5,
  valueFormatter = "#.",
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  draggable = FALSE,
  tooltip = NULL,
  columnStyle = NULL,
  threeD = FALSE,
  bullets = NULL,
  alwaysShowBullets = FALSE,
  backgroundColor = NULL,
  cellWidth = NULL,
  columnWidth = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  legend = NULL,
  caption = NULL,
  image = NULL,
  button = NULL,
  cursor = FALSE,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

data	a dataframe
data2	NULL or a dataframe used to update the data with the button; its column names must include the column names of data given in values, it must have the same number of rows as data and its rows must be in the same order as those of data
category	name of the column of data to be used on the category axis
values	name(s) of the column(s) of data to be used on the value axis
valueNames	names of the values variables, to appear in the legend; NULL to use values as names, otherwise a named list of the form list(value1 = "ValueName1", value2 = "ValueName2", ...) where value1, value2, ... are the column names given in values and "ValueName1", "ValueName2", ... are the desired names to appear in the legend; these names can also appear in the tooltips: they are substituted to the string {name} in the formatting string passed on to the tooltip (see the second example)
showValues	logical, whether to display the values on the chart
vline	an optional vertical line to add to the chart; it must be a named list of the form list(value = v, line = settings) where v is the "intercept" and settings is a list of settings created with amLine
xLimits	range of the x-axis, a vector of two values specifying the left and the right limits of the x-axis; NULL for default values
expandX	if xLimits = NULL, a percentage of the range of the x-axis used to expand this range
valueFormatter	a number formatting string; it is used to format the values displayed on the chart if showValues = TRUE, the values displayed in the cursor tooltips if cursor = TRUE, the labels of the x-axis unless you specify your own formatter in the labels field of the list passed on to the xAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example of amBarChart for the way to set a number formatter in the tooltip text)
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
draggable	TRUE/FALSE to enable/disable dragging of all bars, otherwise a named list of the form list(value1 = TRUE, value2 = FALSE, ...) to enable/disable the dragging for each bar corresponding to a column given in values
tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amTooltip; this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip

columnStyle	settings of the columns; NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with [amColumn](); this can also be a single list of settings that will be applied to each column
threeD	logical, whether to render the columns in 3D
bullets	settings of the bullets; NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with [amCircle](), [amTriangle]() or [amRectangle](); this can also be a single list of settings that will be applied to each series
alwaysShowBullets	
	logical, whether to always show the bullets; if FALSE, the bullets are shown only on hovering a column
backgroundColor	
	a color for the chart background; a color can be given by the name of a R color, the name of a CSS color, e.g. "aqua" or "indigo", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
cellWidth	cell width in percent; for a simple bar chart, this is the width of the columns; for a grouped bar chart, this is the width of the clusters of columns; NULL for the default value
columnWidth	column width, a percentage of the cell width; set to 100 for a simple bar chart and use cellWidth to control the width of the columns; for a grouped bar chart, this controls the spacing between the columns within a cluster of columns; NULL for the default value
xAxis	settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with [amText](), a field labels, a list of settings for the axis labels created with [amAxisLabels](), a field adjust, a number defining the vertical adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with [amLine](), and a field breaks to control the axis breaks, an R object created with [amAxisBreaks]()
yAxis	settings of the category axis given as a list, or just a string for the axis title; the list of settings has three possible fields: a field title, a list of settings for the axis title created with [amText](), a field labels, a list of settings for the axis labels created with [amAxisLabels](), and a field adjust, a number defining the horizontal adjustment of the axis (in pixels)
scrollbarX	logical, whether to add a scrollbar for the value axis
scrollbarY	logical, whether to add a scrollbar for the category axis
legend	FALSE for no legend, TRUE for a legend with default settings, or a list of settings created with [amLegend]()
caption	NULL or FALSE for no caption, a formatted text created with [amText](), or a list with two fields: text, a list created with [amText](), and align, can be "left", "right" or "center"
image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with [amImage](), the field position can be "topleft", "topright",

	"bottomleft" or "bottomright", the field hjust defines the horizontal adjust-ment, and the field vjust defines the vertical adjustment
button	NULL for the default, FALSE for no button, or a list of settings created with amButton; this button is used to replace the current data with data2
cursor	option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor with default settings for the tooltips, or a list of settings created with amTooltip to set the style of the tooltips, or a list with three possible fields: a field tooltip, a list of tooltip settings created with amTooltip, a field extraTooltipPrecision, an integer, the number of additional decimals to display in the tooltips, and a field modifier, which defines a modifier for the values displayed in the tooltips; a modifier is some JavaScript code given a string, which performs a modification of a string named text, e.g. modifier = "text = '>>>' + text;"
width	the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in amChart4Output
height	the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in amChart4Output
export	logical, whether to enable the export menu
chartId	a HTML id for the chart
elementId	a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
# a simple horizontal bar chart ####

dat <- data.frame(
  country = c("USA", "China", "Japan", "Germany", "UK", "France"),
  visits = c(3025, 1882, 1809, 1322, 1122, 1114)
)

amHorizontalBarChart(
  data = dat, data2 = dat,
  width = "600px", height = "550px",
  category = "country", values = "visits",
  draggable = TRUE,
  tooltip = "[font-style:italic #ffff00]{valueX}[/]",
  chartTitle =
    amText(text = "Visits per country", fontSize = 22, color = "orangered"),
  xAxis = list(
    title = amText(text = "Country", color = "maroon"),
    gridLines = amLine(opacity = 0.4, width = 1, dash = "3,1")
  ),
  yAxis = list(title = amText(text = "Visits", color = "maroon")),
  xLimits = c(0, 4000),
  valueFormatter = "#,###",
  caption = amText(text = "Year 2018", color = "red"),
  theme = "moonrisekingdom")
```

```
# a grouped horizontal bar chart ####

set.seed(666)
dat <- data.frame(
  country = c("USA", "China", "Japan", "Germany", "UK", "France"),
  visits = c(3025, 1882, 1809, 1322, 1122, 1114),
  income = rpois(6, 25),
  expenses = rpois(6, 20)
)

amHorizontalBarChart(
  data = dat,
  width = "700px",
  category = "country",
  values = c("income", "expenses"),
  valueNames = list(income = "Income", expenses = "Expenses"),
  tooltip = amTooltip(
    text = "[bold]{name}:\n{valueX}[/]",
    textColor = "white",
    backgroundColor = "#101010",
    borderColor = "silver"
  ),
  draggable = list(income = TRUE, expenses = FALSE),
  backgroundColor = "#30303d",
  columnStyle = list(
    income = amColumn(
      color = "darkmagenta",
      strokeColor = "#cccccc",
      strokeWidth = 2
    ),
    expenses = amColumn(
      color = "darkred",
      strokeColor = "#cccccc",
      strokeWidth = 2
    )
  ),
  chartTitle = amText(text = "Income and expenses per country"),
  yAxis = list(title = amText(text = "Country")),
  xAxis = list(
    title = amText(text = "Income and expenses"),
    gridLines = amLine(color = "whitesmoke", width = 1, opacity = 0.4)
  ),
  xLimits = c(0, 41),
  valueFormatter = "#.#",
  caption = amText(text = "Year 2018"),
  theme = "dark")
```

amHorizontalDumbbellChart

HTML widget displaying a horizontal Dumbbell chart

Description

Create a HTML widget displaying a horizontal Dumbbell chart.

Usage

```
amHorizontalDumbbellChart(
  data,
  data2 = NULL,
  category,
  values,
  seriesNames = NULL,
  vline = NULL,
  xLimits = NULL,
  expandX = 5,
  valueFormatter = "#.",
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  draggable = FALSE,
  tooltip = NULL,
  segmentsStyle = NULL,
  bullets = NULL,
  backgroundColor = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  legend = NULL,
  caption = NULL,
  image = NULL,
  button = NULL,
  cursor = FALSE,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

data	a dataframe
data2	NULL or a dataframe used to update the data with the button; its column names must include the column names of data given in values, it must have the same number of rows as data and its rows must be in the same order as those of data
category	name of the column of data to be used for the category axis
values	a character matrix with two columns; each row corresponds to a series and provides the names of two columns of data to be used as the limits of the segments

seriesNames	a character vector providing the names of the series to appear in the legend; its length must equal the number of rows of the values matrix: the n-th component corresponds to the n-th row of the values matrix
vline	an optional vertical line to add to the chart; it must be a named list of the form list(value = v, line = settings) where v is the "intercept" and settings is a list of settings created with amLine
xLimits	range of the x-axis, a vector of two values specifying the left and right limits of the x-axis; NULL for default values
expandX	if xLimits = NULL, a percentage of the range of the x-axis used to expand this range
valueFormatter	a number formatting string; it is used to format the values displayed in the cursor tooltips, the labels of the x-axis unless you specify your own formatter in the labels field of the list passed on to the xAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example of amBarChart for the way to set a number formatter in the tooltip text)
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
draggable	TRUE/FALSE to enable/disable dragging of all bullets, otherwise a named list of the form list(value1 = TRUE, value2 = FALSE, ...)
tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amTooltip; this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip
segmentsStyle	settings of the segments; NULL for default, otherwise a named list of the form list(series1 = settings1, series2 = settings2, ...) where series1, series2, ... are the names of the series provided in seriesNames and settings1, settings2, ... are lists created with amSegment; this can also be a single list of settings that will be applied to each series
bullets	settings of the bullets; NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amCircle, amTriangle or amRectangle; this can also be a single list of settings that will be applied to each series
backgroundColor	
	a color for the chart background; it can be given by the name of a R color, the name of a CSS color, e.g. "lime" or "olive", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
xAxis	settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the horizontal

adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with amLine and a field breaks to control the axis breaks, an R object created with amAxisBreaks

yAxis   settings of the category axis given as a list, or just a string for the axis title; the list of settings has four possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the vertical adjustment of the axis (in pixels), and a field gridLines, a list of settings for the grid lines created with amLine

scrollbarX   logical, whether to add a scrollbar for the value axis

scrollbarY   logical, whether to add a scrollbar for the category axis

legend   either a logical value, whether to display the legend, or a list of settings for the legend created with amLegend

caption   NULL or FALSE for no caption, a formatted text created with amText, or a list with two fields: text, a list created with amText, and align, can be "left", "right" or "center"

image   option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with amImage, the field position can be "topleft", "topright", "bottomleft" or "bottomright", the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment

button   NULL for the default, FALSE for no button, or a list of settings created with amButton; this button is used to replace the current data with data2

cursor   option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor with default settings for the tooltips, or a list of settings created with amTooltip to set the style of the tooltips, or a list with three possible fields: a field tooltip, a list of tooltip settings created with amTooltip, a field extraTooltipPrecision, an integer, the number of additional decimals to display in the tooltips, and a field modifier, which defines a modifier for the values displayed in the tooltips; a modifier is some JavaScript code given as a string, which performs a modification of a string named text, e.g. modifier = "text = '>>>' + text;"

width   the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in amChart4Output

height   the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in amChart4Output

export   logical, whether to enable the export menu

chartId   a HTML id for the chart

elementId   a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
set.seed(666)
lwr <- rpois(20, 5)
dat <- data.frame(
```

```
    comparison = paste0("Ctrl vs. ", LETTERS[1:20]),
    lwr = lwr,
    upr = lwr + rpois(20, 10)
)

amHorizontalDumbbellChart(
  width = "500px", height = "450px",
  data = dat,
  draggable = TRUE,
  category = "comparison",
  values = rbind(c("lwr", "upr")),
  xLimits = c(0, 30),
  segmentsStyle = amSegment(width = 1, color = "darkred"),
  bullets = amCircle(strokeWidth = 0, color = "darkred"),
  tooltip = amTooltip("left: {valueX}\nright: {openValueX}", scale = 0.75),
  xAxis = list(
    title = amText(
      "difference",
      fontSize = 17, fontWeight = "bold", fontFamily = "Helvetica"
    ),
    gridLines = amLine("darkblue", width = 2, opacity = 0.8, dash = "2,2"),
    breaks = amAxisBreaks(c(0,10,20,30))
  ),
  yAxis = list(
    title = amText(
      "comparison",
      fontSize = 17, fontWeight = "bold", fontFamily = "Helvetica"
    ),
    labels = amAxisLabels(fontSize = 15),
    gridLines = amLine(color = "red", width = 1, opacity = 0.6, dash = "1,3")
  ),
  backgroundColor = "lightsalmon"
)
```

---

amImage   *Image*

Description

Create a list of settings for an image.

Usage

```
amImage(href, width, height, opacity = 1)
```

Arguments

href   a link to an image file or a base64 string representing an image; you can get such
                a string with [tinyIcon](), or you can create it from a file with base64enc::dataURI;
                this option can also be a string of the form "inData:DATAFIELD" where DATAFIELD

is the name of a column of the data - this is useful to have different images in the bullets

width, height	dimensions of the image
opacity	opacity of the image, a number between 0 and 1

Value

A list of settings for an image.

---

amLegend	*Legend*

Description

Create a list of settings for a legend.

Usage

```
amLegend(
  position = "bottom",
  maxHeight = NULL,
  scrollable = FALSE,
  maxWidth = 220,
  itemsWidth = 20,
  itemsHeight = 20
)
```

Arguments

position	legend position
maxHeight	maximum height for a horizontal legend (position = "bottom" or position = "top")
scrollable	whether a vertical legend should be scrollable
maxWidth	maximum width for a vertical legend (position = "left" or position = "right"); set it to NULL for no limit
itemsWidth	width of the legend items
itemsHeight	height of the legend items

Value

A list of settings for a legend.

---

amLine   *Line style*

Description

Create a list of settings for a line.

Usage

```
amLine(
  color = NULL,
  opacity = 1,
  width = 3,
  dash = NULL,
  tensionX = NULL,
  tensionY = NULL
)
```

Arguments

color   line color

opacity   line opacity, a number between 0 and 1

width   line width

dash   string defining a dashed/dotted line; see Dotted and dashed lines

tensionX, tensionY

                    parameters for the smoothing; see Smoothed lines for the meaning of these pa-
                    rameters

Value

A list of settings for a line.

Note

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

amLineChart   *HTML widget displaying a line chart*

Description

Create a HTML widget displaying a line chart.

Usage

```
amLineChart(
  data,
  data2 = NULL,
  xValue,
  yValues,
  yValueNames = NULL,
  hline = NULL,
  vline = NULL,
  xLimits = NULL,
  yLimits = NULL,
  expandX = 0,
  expandY = 5,
  Xformatter = ifelse(isDate, "yyyy-MM-dd", "#."),
  Yformatter = "#.",
  trend = FALSE,
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  draggable = FALSE,
  tooltip = NULL,
  bullets = NULL,
  alwaysShowBullets = FALSE,
  lineStyle = NULL,
  backgroundColor = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  legend = NULL,
  caption = NULL,
  image = NULL,
  button = NULL,
  cursor = FALSE,
  zoomButtons = FALSE,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
```

```
    elementId = NULL
)
```

Arguments

data	a dataframe
data2	NULL or a dataframe used to update the data with the button; its column names must include the column names of data given in yValues as well as the column name given in xValue; moreover it must have the same number of rows as data and its rows must be in the same order as those of data
xValue	name of the column of data to be used on the x-axis
yValues	name(s) of the column(s) of data to be used on the y-axis
yValueNames	names of the variables on the y-axis, to appear in the legend; NULL to use yValues as names, otherwise a named list of the form list(yvalue1 = "ValueName1", yvalue2 = "ValueName2", ...) where yvalue1, yvalue2, ... are the column names given in yValues and "ValueName1", "ValueName2", ... are the desired names to appear in the legend
hline	an optional horizontal line to add to the chart; it must be a named list of the form list(value = h, line = settings) where h is the "intercept" and settings is a list of settings created with [amLine](#)
vline	an optional vertical line to add to the chart; it must be a named list of the form list(value = v, line = settings) where v is the "intercept" and settings is a list of settings created with [amLine](#)
xLimits	range of the x-axis, a vector of two values specifying the left and right limits of the x-axis; NULL for default values
yLimits	range of the y-axis, a vector of two values specifying the lower and the upper limits of the y-axis; NULL for default values
expandX	if xLimits = NULL, a percentage of the range of the x-axis used to expand this range
expandY	if yLimits = NULL, a percentage of the range of the y-axis used to expand this range
Xformatter	a [number formatting string](#) if xValue is set to a numeric column of data; it is used to format the values displayed in the cursor tooltips if cursor = TRUE, the labels of the x-axis unless you specify your own formatter in the labels field of the list passed on to the xAxis option, and the values displayed in the tooltips unless you specify your own tooltip text; if xValue is set to a date column of data, this option should be set to a [date formatting string](#), and it has an effect only on the values displayed in the tooltips (unless you specify your own tooltip text); formatting the dates on the x-axis is done via the labels field of the list passed on to the xAxis option
Yformatter	a [number formatting string](#); it is used to format the values displayed in the cursor tooltips if cursor = TRUE, the labels of the y-axis unless you specify your own formatter in the labels field of the list passed on to the yAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example of [amBarChart](#) for the way to set a number formatter in the tooltip text)

trend   option to request trend lines and to set their settings; FALSE for no trend line, oth-
erwise a named list of the form list(yvalue1 = trend1, yvalue2 = trend2,
...) where trend1, trend2, ... are lists with the following fields:

> method the modelling method, can be "lm", "lm.js", "nls", "nlsLM", or
> "loess"; "lm.js" performs a polynomial regression in JavaScript, its ad-
> vantage is that the fitted regression line is updated when the points of the
> line are dragged

> formula a formula passed on to the modelling function for methods "lm",
> "nls" or "nlsLM"; the lefthandside of this formula must always be y, and
> its righthandside must be a symbolic expression depending on x only, e.g.
> y ~ x, y ~ x + I(x^2), y ~ poly(x,2)

> interval effective for methods "lm" and "lm.js" only; a list with five possible
> fields: type can be "confidence" or "prediction", level is the confi-
> dence or prediction level (number between 0 and 1), color is the color of
> the shaded area, opacity is the opacity of the shaded area (number between
> 0 and 1), tensionX and tensionY to control the smoothing (see [amLine](#))

> order the order of the polynomial regression when method = "lm.js"

> method.args a list of additional arguments passed on to the modelling function
> defined by method for methods "nls", "nlsLM" or "loess", e.g. method.args
> = list(span = 0.3) for method "loess"

> style a list of settings for the trend line created with [amLine](#)

> it is also possible to request the same kind of trend lines for all series given by the
> yValues argument, by passing a list of the form list("_all" = trendconfig),
> e.g. list("_all" = list(method = "lm", formula = y ~ 0+x, style = amLine()))

chartTitle   chart title, it can be NULL or FALSE for no title, a character string, a list of settings
created with [amText](#), or a list with two fields: text, a list of settings created with
[amText](#), and align, can be "left", "right" or "center"

theme   theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom",
"frozen", "spiritedaway", "patterns", "microchart"

animated   Boolean, whether to animate the rendering of the graphic

draggable   TRUE/FALSE to enable/disable dragging of all lines, otherwise a named list of
the form list(yvalue1 = TRUE, yvalue2 = FALSE, ...) to enable/disable the
dragging for each series corresponding to a column given in yValues

tooltip   settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a
named list of the form list(yvalue1 = settings1, yvalue2 = settings2,
...) where settings1, settings2, ... are lists created with [amTooltip](#); this
can also be a single list of settings that will be applied to each series, or a just a
string for the text to display in the tooltip

bullets   settings of the bullets; NULL for default, otherwise a named list of the form
list(yvalue1 = settings1, yvalue2 = settings2, ...) where settings1,
settings2, ... are lists created with [amCircle](#), [amTriangle](#) or [amRectangle](#);
this can also be a single list of settings that will be applied to each series

alwaysShowBullets
         logical, whether the bullets should always be visible, or visible on hover only

lineStyle   settings of the lines; NULL for default, otherwise a named list of the form list(yvalue1 = settings1, yvalue2 = settings2, ...) where settings1, settings2, ... are lists created with [amLine](#); this can also be a single list of settings that will be applied to each line

backgroundColor
                   a color for the chart background; it can be given by the name of a R color, the name of a CSS color, e.g. "teal" or "fuchsia", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"

xAxis   settings of the x-axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with [amText](#), a field labels, a list of settings for the axis labels created with [amAxisLabels](#), a field adjust, a number defining the vertical adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with [amLine](#), and a field breaks to control the axis breaks, an R object created with [amAxisBreaks](#)

yAxis   settings of the y-axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with [amText](#), a field labels, a list of settings for the axis labels created with [amAxisLabels](#), a field adjust, a number defining the horizontal adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with [amLine](#), and a field breaks to control the axis breaks, an R object created with [amAxisBreaks](#)

scrollbarX   logical, whether to add a scrollbar for the x-axis

scrollbarY   logical, whether to add a scrollbar for the y-axis

legend   FALSE for no legend, TRUE for a legend with default settings, or a list of settings created with [amLegend](#)

caption   NULL or FALSE for no caption, a formatted text created with [amText](#), or a list with two fields: text, a list created with [amText](#), and align, can be "left", "right" or "center"

image   option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with [amImage](#), the field position can be "topleft", "topright", "bottomleft" or "bottomright", the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment

button   NULL for the default, FALSE for no button, or a list of settings created with [amButton](#); this button is used to replace the current data with data2

cursor   option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor for both axes with default settings for the axes tooltips, otherwise a named list with four possible fields: a field axes to specify the axes for which the cursor is requested, can be "x", "y", or "xy", a field tooltip to set the style of the axes tooltips, this must be a list of settings created with [amTooltip](#), a field extraTooltipPrecision, a named list of the form list(x = i, y = j) where i and j are the desired numbers of additional decimals for the tooltips on the x-axis and on the y-axis respectively, and a field modifier, a list with two possible fields, x and y, which defines modifiers for the values displayed in the tooltips;

a modifier is some JavaScript code given a string, which performs a modification of a string named `text`, e.g. `"text = '[font-style:italic]' + text + '[/]';"`; see the first example for an example of `modifier`

zoomButtons   a Boolean value, or a list created with [amZoomButtons](#)

width   the width of the chart, e.g. `"600px"` or `"80%"`; ignored if the chart is displayed in Shiny, in which case the width is given in [amChart4Output](#)

height   the height of the chart, e.g. `"400px"`; ignored if the chart is displayed in Shiny, in which case the height is given in [amChart4Output](#)

export   logical, whether to enable the export menu

chartId   a HTML id for the chart

elementId   a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
# a line chart with a numeric x-axis ####

set.seed(666)
dat <- data.frame(
  x = 1:10,
  y1 = rnorm(10),
  y2 = rnorm(10)
)

amLineChart(
  data = dat,
  width = "700px",
  xValue = "x",
  yValues = c("y1", "y2"),
  yValueNames = list(y1 = "Sample 1", y2 = "Sample 2"),
  trend = list(
    y1 = list(
      method = "lm.js",
      order = 3,
      style = amLine(color = "lightyellow", dash = "3,2")
    ),
    y2 = list(
      method = "loess",
      style = amLine(color = "palevioletred", dash = "3,2")
    )
  ),
  draggable = list(y1 = TRUE, y2 = FALSE),
  backgroundColor = "#30303d",
  tooltip = amTooltip(
    text = "[bold]({valueX},{valueY})[/]",
    textColor = "white",
    backgroundColor = "#101010",
    borderColor = "whitesmoke"
  ),
  bullets = list(
```

```
      y1 = amCircle(color = "yellow", strokeColor = "olive"),
      y2 = amCircle(color = "orangered", strokeColor = "darkred")
    ),
    alwaysShowBullets = TRUE,
    cursor = list(
      extraTooltipPrecision = list(x = 0, y = 2),
      modifier = list(
        y = c(
          "var value = parseFloat(text);",
          "var style = value > 0 ? '[#0000ff]' : '[#ff0000]';",
          "text = style + text + '[/]';"
        )
      )
    ),
    lineStyle = list(
      y1 = amLine(color = "yellow", width = 4),
      y2 = amLine(color = "orangered", width = 4)
    ),
    chartTitle = amText(
      text = "Gaussian samples",
      color = "whitesmoke",
      fontWeight = "bold"
    ),
    xAxis = list(title = amText(text = "Observation",
                               fontSize = 21,
                               color = "silver",
                               fontWeight = "bold"),
                 labels = amAxisLabels(fontSize = 17),
                 breaks = amAxisBreaks(
                   values = 1:10,
                   labels = sprintf("[bold %s]%d[/]", rainbow(10), 1:10))),
    yAxis = list(title = amText(text = "Value",
                               fontSize = 21,
                               color = "silver",
                               fontWeight = "bold"),
                 labels = amAxisLabels(color = "whitesmoke",
                                       fontSize = 14),
                 gridLines = amLine(color = "whitesmoke",
                                    opacity = 0.4,
                                    width = 1)),
    yLimits = c(-3, 3),
    Yformatter = "#.00",
    caption = amText(text = "[font-style:italic]try to drag the yellow line![/]",
                     color = "yellow"),
    theme = "dark")


# line chart with a date x-axis ####

library(lubridate)

set.seed(666)
dat <- data.frame(
```

```
    date = ymd(180101) + days(0:60),
    visits = rpois(61, 20)
)

amLineChart(
  data = dat,
  width = "750px",
  xValue = "date",
  yValues = "visits",
  draggable = TRUE,
  chartTitle = "Number of visits",
  xAxis = list(
    title = "Date",
    labels = amAxisLabels(
      formatter = amDateAxisFormatter(
        day = c("dt", "[bold]MMM[/] dt"),
        week = c("dt", "[bold]MMM[/] dt")
      )
    ),
    breaks = amAxisBreaks(timeInterval = "7 days")
  ),
  yAxis = "Visits",
  xLimits = range(dat$date) + c(0,7),
  yLimits = c(0, 35),
  backgroundColor = "whitesmoke",
  tooltip = paste0(
    "[bold][font-style:italic]{dateX.value.formatDate('yyyy/MM/dd')}[/]",
    "\nvisits: {valueY}[/]"
  ),
  caption = amText(text = "Year 2018"),
  theme = "material")


# smoothed lines ####

x <- seq(-4, 4, length.out = 100)
dat <- data.frame(
  x = x,
  Gauss = dnorm(x),
  Cauchy = dcauchy(x)
)

amLineChart(
  data = dat,
  width = "700px",
  xValue = "x",
  yValues = c("Gauss", "Cauchy"),
  yValueNames = list(
    Gauss = "Standard normal distribution",
    Cauchy = "Cauchy distribution"
  ),
  draggable = FALSE,
  tooltip = FALSE,
```

```
      lineStyle = amLine(
        width = 4,
        tensionX = 0.8,
        tensionY = 0.8
      ),
      xAxis = list(title = amText(text = "x",
                                  fontSize = 21,
                                  color = "navyblue"),
                   labels = amAxisLabels(
                     color = "midnightblue",
                     fontSize = 17)),
      yAxis = list(title = amText(text = "density",
                                  fontSize = 21,
                                  color = "navyblue"),
                   labels = FALSE),
      theme = "dataviz")
```

---

amPercentageBarChart   *HTML widget displaying a 100% stacked bar chart*

Description

Create a HTML widget displaying a 100% stacked bar chart.

Usage

```
amPercentageBarChart(
  data,
  category,
  values,
  valueNames = NULL,
  hline = NULL,
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  backgroundColor = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  legend = TRUE,
  caption = NULL,
  image = NULL,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

data	a dataframe
category	name of the column of data to be used on the category axis
values	names of the columns of data to be used on the value axis
valueNames	names of the values variables, to appear in the legend; NULL to use values as names, otherwise a named list of the form list(value1 = "ValueName1", value2 = "ValueName2", ...) where value1, value2, ... are the column names given in values and "ValueName1", "ValueName2", ... are the desired names to appear in the legend; these names also appear in the tooltips.
hline	an optional horizontal line to add to the chart; it must be a named list of the form list(value = h, line = settings) where h is the "intercept" and settings is a list of settings created with amLine
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
backgroundColor	
	a color for the chart background; a color can be given by the name of a R color, the name of a CSS color, e.g. "rebeccapurple" or "fuchsia", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
xAxis	settings of the category axis given as a list, or just a string for the axis title; the list of settings has three possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, and a field adjust, a number defining the vertical adjustment of the axis (in pixels)
yAxis	settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the horizontal adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with amLine and a field breaks to control the axis breaks, an R object created with amAxisBreaks
scrollbarX	logical, whether to add a scrollbar for the category axis
scrollbarY	logical, whether to add a scrollbar for the value axis
legend	either a logical value, whether to display the legend, or a list of settings for the legend created with amLegend
caption	NULL or FALSE for no caption, a formatted text created with amText, or a list with two fields: text, a list created with amText, and align, can be "left", "right" or "center"

image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with amImage, the field position can be "topleft", "topright", "bottomleft" or "bottomright", the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment
width	the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in amChart4Output
height	the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in amChart4Output
export	logical, whether to enable the export menu
chartId	a HTML id for the chart
elementId	a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
library(rAmCharts4)

dat <- data.frame(
  category = c("A", "B", "C"),
  v1 = c(1, 2, 3),
  v2 = c(9, 5, 7)
)

amPercentageBarChart(
  dat,
  category = "category",
  values = c("v1", "v2"),
  valueNames = c("Value1", "Value2"),
  yAxis = "Percentage",
  theme = "dataviz",
  legend = amLegend(position = "right")
)
```

---

amPieChart   *HTML widget displaying a pie chart*

Description

Create a HTML widget displaying a pie chart.

Usage

```
amPieChart(
  data,
  category,
  value,
```

```
    innerRadius = 0,
    threeD = FALSE,
    depth = ifelse(variableDepth, 100, 10),
    colorStep = 3,
    variableRadius = FALSE,
    variableDepth = FALSE,
    chartTitle = NULL,
    theme = NULL,
    animated = TRUE,
    backgroundColor = NULL,
    legend = TRUE,
    caption = NULL,
    image = NULL,
    width = NULL,
    height = NULL,
    export = FALSE,
    chartId = NULL,
    elementId = NULL
)
```

Arguments

data	a dataframe
category	name of the column of data to be used as the category
value	name of the column of data to be used as the value
innerRadius	the inner radius of the pie chart in percent
threeD	whether to render a 3D pie chart
depth	for a 3D chart, this parameter controls the height of the slices
colorStep	the step in the color palette
variableRadius	whether to render slices with variable radius
variableDepth	for a 3D chart, whether to render slices with variable depth
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with [amText](), or a list with two fields: text, a list of settings created with [amText](), and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
backgroundColor	a color for the chart background; it can be given by the name of a R color, the name of a CSS color, e.g. "lime" or "olive", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
legend	either a logical value, whether to display the legend, or a list of settings for the legend created with [amLegend]()
caption	NULL or FALSE for no caption, a formatted text created with [amText](), or a list with two fields: text, a list created with [amText](), and align, can be "left", "right" or "center"
```

image                 option to include an image at a corner of the chart; NULL or FALSE for no image,
                      otherwise a named list with four possible fields: the field image (required) is a
                      list created with [amImage](), the field position can be "topleft", "topright",
                      "bottomleft" or "bottomright", the field hjust defines the horizontal adjust-
                      ment, and the field vjust defines the vertical adjustment

width                 the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed
                      in Shiny, in which case the width is given in [amChart4Output]()

height                the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny,
                      in which case the height is given in [amChart4Output]()

export                logical, whether to enable the export menu

chartId               a HTML id for the chart

elementId             a HTML id for the container of the chart; ignored if the chart is displayed in
                      Shiny, in which case the id is given by the Shiny id

## Examples

```r
library(rAmCharts4)
dat <- data.frame(
  country = c(
    "Lithuania", "Czechia", "Ireland", "Germany", "Australia", "Austria"
  ),
  value = c(260, 230, 200, 165, 139, 128)
)
amPieChart(
  data = dat,
  category = "country",
  value = "value",
  variableRadius = TRUE
)

# shiny app demonstrating the options ####
library(rAmCharts4)
library(shiny)

dat <- data.frame(
  country = c(
    "Lithuania", "Czechia", "Ireland", "Germany", "Australia", "Austria"
  ),
  value = c(260, 230, 200, 165, 139, 128)
)

ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      sliderInput(
        "innerRadius", "Inner radius", min = 0, max = 60, value = 0, step = 20
      ),
      checkboxInput("variableRadius", "Variable radius", TRUE),
      checkboxInput("threeD", "3D"),
      conditionalPanel(
```

```
      "input.threeD",
      checkboxInput("variableDepth", "Variable depth")
    )
  ),
  mainPanel(
    amChart4Output("piechart", height = "500px")
  )
  )
)

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

  piechart <- reactive({
    amPieChart(
      data = dat,
      category = "country",
      value = "value",
      innerRadius = input[["innerRadius"]],
      threeD = input[["threeD"]],
      variableDepth = input[["variableDepth"]],
      depth = ifelse(input[["variableDepth"]], 300, 10),
      variableRadius = input[["variableRadius"]],
      theme = "dark"
    )
  })

  output[["piechart"]] <- renderAmChart4({
    piechart()
  })
}

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

---

amRadialBarChart   *HTML widget displaying a radial bar chart*

Description

Create a HTML widget displaying a radial bar chart.

Usage

```
amRadialBarChart(
  data,
  data2 = NULL,
  category,
  values,
  valueNames = NULL,
```

```
        showValues = TRUE,
        innerRadius = 50,
        yLimits = NULL,
        expandY = 5,
        valueFormatter = "#.",
        chartTitle = NULL,
        theme = NULL,
        animated = TRUE,
        draggable = FALSE,
        tooltip = NULL,
        columnStyle = NULL,
        bullets = NULL,
        alwaysShowBullets = FALSE,
        backgroundColor = NULL,
        cellWidth = NULL,
        columnWidth = NULL,
        xAxis = NULL,
        yAxis = NULL,
        scrollbarX = FALSE,
        scrollbarY = FALSE,
        legend = NULL,
        caption = NULL,
        image = NULL,
        button = NULL,
        cursor = FALSE,
        width = NULL,
        height = NULL,
        export = FALSE,
        chartId = NULL,
        elementId = NULL
    )
```

Arguments

`data`	a dataframe
`data2`	NULL or a dataframe used to update the data with the button; its column names must include the column names of `data` given in `values`, it must have the same number of rows as `data` and its rows must be in the same order as those of `data`
`category`	name of the column of `data` to be used on the category axis
`values`	name(s) of the column(s) of `data` to be used on the value axis
`valueNames`	names of the values variables, to appear in the legend; NULL to use `values` as names, otherwise a named list of the form `list(value1 = "ValueName1", value2 = "ValueName2", ...)` where `value1, value2, ...` are the column names given in `values` and `"ValueName1", "ValueName2", ...` are the desired names to appear in the legend; these names can also appear in the tooltips: they are substituted to the string `{name}` in the formatting string passed on to the tooltip (see the second example of [`amBarChart`](#))

showValues	logical, whether to display the values on the chart
innerRadius	inner radius of the chart, a percentage (between 0 and 100 theoretically, but in practice it should be between 30 and 70)
yLimits	range of the y-axis, a vector of two values specifying the lower and the upper limits of the y-axis; NULL for default values
expandY	if yLimits = NULL, a percentage of the range of the y-axis used to expand this range
valueFormatter	a number formatting string; it is used to format the values displayed on the chart if showValues = TRUE, the values displayed in the cursor tooltips if cursor = TRUE, the labels of the y-axis unless you specify your own formatter in the labels field of the list passed on to the yAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example for the way to set a number formatter in the tooltip text)
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
draggable	TRUE/FALSE to enable/disable dragging of all bars, otherwise a named list of the form list(value1 = TRUE, value2 = FALSE, ...) to enable/disable the dragging for each bar corresponding to a column given in values
tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amTooltip; this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip
columnStyle	settings of the columns; NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amColumn; this can also be a single list of settings that will be applied to each column
bullets	settings of the bullets; NULL for default, otherwise a named list of the form list(value1 = settings1, value2 = settings2, ...) where settings1, settings2, ... are lists created with amCircle, amTriangle or amRectangle; this can also be a single list of settings that will be applied to each series
alwaysShowBullets	
	logical, whether to always show the bullets; if FALSE, the bullets are shown only on hovering a column
backgroundColor	
	a color for the chart background; a color can be given by the name of a R color, the name of a CSS color, e.g. "lime" or "fuchsia", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
cellWidth	cell width in percent; for a simple bar chart, this is the width of the columns; for a grouped bar chart, this is the width of the clusters of columns; NULL for the default value

columnWidth	column width, a percentage of the cell width; set to 100 for a simple bar chart and use `cellWidth` to control the width of the columns; for a grouped bar chart, this controls the spacing between the columns within a cluster of columns; NULL for the default value
xAxis	settings of the category axis given as a list, or just a string for the axis title; the list of settings has three possible fields: a field `title`, a list of settings for the axis title created with [amText](#), a field `labels`, a list of settings for the axis labels created with [amAxisLabelsCircular](#), and a field `adjust`, a number defining the vertical adjustment of the axis (in pixels)
yAxis	settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field `title`, a list of settings for the axis title created with [amText](#), a field `labels`, a list of settings for the axis labels created with [amAxisLabels](#), a field `adjust`, a number defining the horizontal adjustment of the axis (in pixels), a field `gridLines`, a list of settings for the grid lines created with [amLine](#) and a field `breaks` to control the axis breaks, an R object created with [amAxisBreaks](#)
scrollbarX	logical, whether to add a scrollbar for the category axis
scrollbarY	logical, whether to add a scrollbar for the value axis
legend	either a logical value, whether to display the legend, or a list of settings for the legend created with [amLegend](#)
caption	NULL or FALSE for no caption, a formatted text created with [amText](#), or a list with two fields: `text`, a list created with [amText](#), and `align`, can be `"left"`, `"right"` or `"center"`
image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field `image` (required) is a list created with [amImage](#), the field `position` can be `"topleft"`, `"topright"`, `"bottomleft"` or `"bottomright"`, the field `hjust` defines the horizontal adjustment, and the field `vjust` defines the vertical adjustment
button	NULL for the default, FALSE for no button, or a list of settings created with [amButton](#); this button is used to replace the current data with `data2`
cursor	option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor with default settings for the tooltips, or a list of settings created with [amTooltip](#) to set the style of the tooltips, or a list with three possible fields: a field `tooltip`, a list of tooltip settings created with [amTooltip](#), a field `extraTooltipPrecision`, an integer, the number of additional decimals to display in the tooltips, and a field `modifier`, which defines a modifier for the values displayed in the tooltips; a modifier is some JavaScript code given a string, which performs a modification of a string named `text`, e.g. `modifier = "text = '>>>' + text;"`
width	the width of the chart, e.g. `"600px"` or `"80%"`; ignored if the chart is displayed in Shiny, in which case the width is given in [amChart4Output](#)
height	the height of the chart, e.g. `"400px"`; ignored if the chart is displayed in Shiny, in which case the height is given in [amChart4Output](#)
export	logical, whether to enable the export menu
chartId	a HTML id for the chart
elementId	a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
# a grouped radial bar chart ####

set.seed(666)
dat <- data.frame(
  country = c("USA", "China", "Japan", "Germany", "UK", "France"),
  visits = c(3025, 1882, 1809, 1322, 1122, 1114),
  income = rpois(6, 25),
  expenses = rpois(6, 20)
)

amRadialBarChart(
  data = dat, data2 = dat,
  width = "600px", height = "600px",
  category = "country",
  values = c("income", "expenses"),
  valueNames = list(income = "Income", expenses = "Expenses"),
  showValues = FALSE,
  tooltip = amTooltip(
    textColor = "white",
    backgroundColor = "#101010",
    borderColor = "silver"
  ),
  draggable = TRUE,
  backgroundColor = "#30303d",
  columnStyle = list(
    income = amColumn(
      color = "darkmagenta",
      strokeColor = "#cccccc",
      strokeWidth = 2
    ),
    expenses = amColumn(
      color = "darkred",
      strokeColor = "#cccccc",
      strokeWidth = 2
    )
  ),
  chartTitle = "Income and expenses per country",
  xAxis = list(
    labels = amAxisLabelsCircular(
      radius = -82, relativeRotation = 90
    )
  ),
  yAxis = list(
    labels = amAxisLabels(color = "orange"),
    gridLines = amLine(color = "whitesmoke", width = 1, opacity = 0.4),
    breaks = amAxisBreaks(values = seq(0, 40, by = 10))
  ),
  yLimits = c(0, 40),
  valueFormatter = "#.#",
  caption = amText(
    text = "Year 2018",
```

```
      fontFamily = "Impact",
      fontSize = 18
    ),
    theme = "dark")


  # just for fun ####

  dat <- data.frame(
    cluster = letters[1:6],
    y1 = rep(10, 6),
    y2 = rep(8, 6),
    y3 = rep(6, 6),
    y4 = rep(4, 6),
    y5 = rep(2, 6),
    y6 = rep(4, 6),
    y7 = rep(6, 6),
    y8 = rep(8, 6),
    y9 = rep(10, 6)
  )

  amRadialBarChart(
    data = dat,
    width = "500px", height = "500px",
    innerRadius = 10,
    category = "cluster", values = paste0("y", 1:9),
    showValues = FALSE,
    tooltip = FALSE, draggable = FALSE,
    backgroundColor = "black",
    columnStyle = amColumn(strokeWidth = 1, strokeColor = "white"),
    cellWidth = 96,
    xAxis = list(labels = FALSE),
    yAxis = list(labels = FALSE, gridLines = FALSE),
    yLimits = c(0, 10),
    legend = FALSE,
    theme = "kelly")
```

---

amRangeAreaChart   *HTML widget displaying a range area chart*

Description

Create a HTML widget displaying a range area chart.

Usage

```
amRangeAreaChart(
  data,
  data2 = NULL,
  xValue,
```

```
  yValues,
  areas = NULL,
  hline = NULL,
  vline = NULL,
  xLimits = NULL,
  yLimits = NULL,
  expandX = 0,
  expandY = 5,
  Xformatter = ifelse(isDate, "yyyy-MM-dd", "#."),
  Yformatter = "#.",
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  draggable = FALSE,
  tooltip = NULL,
  bullets = NULL,
  alwaysShowBullets = FALSE,
  lineStyle = NULL,
  backgroundColor = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  legend = NULL,
  caption = NULL,
  image = NULL,
  button = NULL,
  cursor = FALSE,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

`data`	a dataframe
`data2`	NULL or a dataframe used to update the data with the button; its column names must include the column names of `data` given in `yValues`, it must have the same number of rows as `data` and its rows must be in the same order as those of `data`
`xValue`	name of the column of `data` to be used on the x-axis
`yValues`	a character matrix with two columns; each row corresponds to a range area and provides the names of two columns of `data` to be used as the limits of the range area
`areas`	an unnamed list of list of settings for the range areas; the n-th inner list of settings corresponds to the n-th row of the `yValues` matrix; each list of settings has three

	possible fields: name for the legend label, color for the color of the range area, and opacity for the opacity of the range area, a number between 0 and 1
hline	an optional horizontal line to add to the chart; it must be a named list of the form list(value = h, line = settings) where h is the "intercept" and settings is a list of settings created with amLine
vline	an optional vertical line to add to the chart; it must be a named list of the form list(value = v, line = settings) where v is the "intercept" and settings is a list of settings created with amLine
xLimits	range of the x-axis, a vector of two values specifying the left and right limits of the x-axis; NULL for default values
yLimits	range of the y-axis, a vector of two values specifying the lower and upper limits of the y-axis; NULL for default values
expandX	if xLimits = NULL, a percentage of the range of the x-axis used to expand this range
expandY	if yLimits = NULL, a percentage of the range of the y-axis used to expand this range
Xformatter	a number formatting string if xValue is set to a numeric column of data; it is used to format the values displayed in the cursor tooltips if cursor = TRUE, the labels of the x-axis unless you specify your own formatter in the labels field of the list passed on to the xAxis option, and the values displayed in the tooltips unless you specify your own tooltip text; if xValue is set to a date column of data, this option should be set to a date formatting string, and it has an effect only on the values displayed in the tooltips (unless you specify your own tooltip text); formatting the dates on the x-axis is done via the labels field of the list passed on to the xAxis option
Yformatter	a number formatting string; it is used to format the values displayed in the cursor tooltips if cursor = TRUE, the labels of the y-axis unless you specify your own formatter in the labels field of the list passed on to the yAxis option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example of amBarChart for the way to set a number formatter in the tooltip text)
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with amText, or a list with two fields: text, a list of settings created with amText, and align, can be "left", "right" or "center"
theme	theme, NULL or one of "dataviz", "material", "kelly", "dark", "moonrisekingdom", "frozen", "spiritedaway", "patterns", "microchart"
animated	Boolean, whether to animate the rendering of the graphic
draggable	TRUE/FALSE to enable/disable dragging of all lines, otherwise a named list of the form list(yvalue1 = TRUE, yvalue2 = FALSE, ...) to enable/disable the dragging for each series corresponding to a column given in yValues
tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form list(yvalue1 = settings1, yvalue2 = settings2, ...) where settings1, settings2, ... are lists created with amTooltip; this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip

bullets   settings of the bullets; NULL for default, otherwise a named list of the form
                     list(yvalue1 = settings1, yvalue2 = settings2, ...) where settings1,
                     settings2, ... are lists created with [amCircle](), [amTriangle]() or [amRectangle]();
                     this can also be a single list of settings that will be applied to each series

alwaysShowBullets
                     logical, whether the bullets should always be visible, or visible on hover only

lineStyle   settings of the lines; NULL for default, otherwise a named list of the form list(yvalue1
                     = settings1, yvalue2 = settings2, ...) where settings1, settings2, ...
                     are lists created with [amLine](); this can also be a single list of settings that will be
                     applied to each line

backgroundColor
                     a color for the chart background

xAxis   settings of the x-axis given as a list, or just a string for the axis title; the list of
                     settings has five possible fields: a field title, a list of settings for the axis title
                     created with [amText](), a field labels, a list of settings for the axis labels created
                     with [amAxisLabels](), a field adjust, a number defining the vertical adjustment
                     of the axis (in pixels), a field gridLines, a list of settings for the grid lines
                     created with [amLine](), and a field breaks to control the axis breaks, an R object
                     created with [amAxisBreaks]()

yAxis   settings of the y-axis given as a list, or just a string for the axis title; the list
                     of settings has five possible fields: a field title, a list of settings for the axis
                     title created with [amText](), a field labels, a list of settings for the axis labels
                     created with [amAxisLabels](), a field adjust, a number defining the horizontal
                     adjustment of the axis (in pixels), a field gridLines, a list of settings for the
                     grid lines created with [amLine](), and a field breaks to control the axis breaks, an
                     R object created with [amAxisBreaks]()

scrollbarX   logical, whether to add a scrollbar for the x-axis

scrollbarY   logical, whether to add a scrollbar for the y-axis

legend   FALSE for no legend, TRUE for a legend with default settings, or a list of settings
                     created with [amLegend]()

caption   NULL or FALSE for no caption, a formatted text created with [amText](), or a list
                     with two fields: text, a list created with [amText](), and align, can be "left",
                     "right" or "center"

image   option to include an image at a corner of the chart; NULL or FALSE for no image,
                     otherwise a named list with four possible fields: the field image (required) is a
                     list created with [amImage](), the field position can be "topleft", "topright",
                     "bottomleft" or "bottomright", the field hjust defines the horizontal adjust-
                     ment, and the field vjust defines the vertical adjustment

button   NULL for the default, FALSE for no button, or a list of settings created with
                     [amButton](); this button is used to replace the current data with data2

cursor   option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor
                     for both axes with default settings for the axes tooltips, otherwise a named list
                     with four possible fields: a field axes to specify the axes for which the cur-
                     sor is requested, can be "x", "y", or "xy", a field tooltip to set the style of
                     the axes tooltips, this must be a list of settings created with [amTooltip](), a field

extraTooltipPrecision, a named list of the form list(x = i, y = j) where
i and j are the desired numbers of additional decimals for the tooltips on the x-
axis and on the y-axis respectively, and a field modifier, a list with two possible
fields, x and y, which defines modifiers for the values displayed in the tooltips;
a modifier is some JavaScript code given as a string, which performs a modifi-
cation of a string named text, e.g. "text = '[font-style:italic]' + text +
'[/]';"; see the example for an example of modifier

width   the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed
             in Shiny, in which case the width is given in amChart4Output

height   the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny,
             in which case the height is given in amChart4Output

export   logical, whether to enable the export menu

chartId   a HTML id for the chart

elementId   a HTML id for the container of the chart; ignored if the chart is displayed in
             Shiny, in which case the id is given by the Shiny id

Note

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

Examples

```
set.seed(666)
x <- 1:20
dat <- data.frame(
  x = x,
  y1 = rnorm(20, sd = 1.5),
  y2 = rnorm(20, 10, sd = 1.5),
  z1 = rnorm(20, x+5, sd = 1.5),
  z2 = rnorm(20, x+15, sd = 1.5)
)

amRangeAreaChart(
  data = dat,
  width = "700px",
  xValue = "x",
  yValues = rbind(c("y1", "y2"), c("z1", "z2")),
  xLimits = c(1, 20),
  draggable = TRUE,
  backgroundColor = "#30303d",
  tooltip = list(
    y1 = amTooltip(
      text = "[bold]upper: {openValueY}\nlower: {valueY}[/]",
      textColor = "yellow",
      backgroundColor = "darkmagenta",
      backgroundOpacity = 0.8,
      borderColor = "rebeccapurple",
      scale = 0.9
```

```
    ),
    y2 = amTooltip(
      text = "[bold]upper: {valueY}\nlower: {openValueY}[/]",
      textColor = "yellow",
      backgroundColor = "darkmagenta",
      backgroundOpacity = 0.8,
      borderColor = "rebeccapurple",
      scale = 0.9
    ),
    z1 = amTooltip(
      text = "[bold]upper: {openValueY}\nlower: {valueY}[/]",
      textColor = "white",
      backgroundColor = "darkred",
      backgroundOpacity = 0.8,
      borderColor = "crimson",
      scale = 0.9
    ),
    z2 = amTooltip(
      text = "[bold]upper: {valueY}\nlower: {openValueY}[/]",
      textColor = "white",
      backgroundColor = "darkred",
      backgroundOpacity = 0.8,
      borderColor = "crimson",
      scale = 0.9
    )
  ),
  bullets = list(
    y1 = amCircle(color = "yellow", strokeColor = "olive"),
    y2 = amCircle(color = "yellow", strokeColor = "olive"),
    z1 = amCircle(color = "orangered", strokeColor = "darkred"),
    z2 = amCircle(color = "orangered", strokeColor = "darkred")
  ),
  alwaysShowBullets = FALSE,
  lineStyle = list(
    y1 = amLine(color = "yellow", width = 3, tensionX = 0.8, tensionY = 0.8),
    y2 = amLine(color = "yellow", width = 3, tensionX = 0.8, tensionY = 0.8),
    z1 = amLine(color = "orangered", width = 3, tensionX = 0.8, tensionY = 0.8),
    z2 = amLine(color = "orangered", width = 3, tensionX = 0.8, tensionY = 0.8)
  ),
  areas = list(
    list(name = "y1-y2", color = "blue", opacity = 0.2),
    list(name = "z1-z2", color = "red", opacity = 0.2)
  ),
  cursor = list(
    tooltip = amTooltip(
      backgroundColor = "silver"
    ),
    extraTooltipPrecision = list(x = 0, y = 2),
    modifier = list(y = "text = parseFloat(text).toFixed(2);")
  ),
  chartTitle = amText(text = "Range area chart",
                      color = "whitesmoke",
                      fontWeight = "bold"),
```

```
                xAxis = list(title = amText(text = "Observation",
                                            fontSize = 20,
                                            color = "silver"),
                             labels = amAxisLabels(color = "whitesmoke",
                                                   fontSize = 17),
                             adjust = 5),
                yAxis = list(title = amText(text = "Value",
                                            fontSize = 20,
                                            color = "silver"),
                             labels = amAxisLabels(color = "whitesmoke",
                                                   fontSize = 17),
                             gridLines = amLine(color = "antiquewhite",
                                                opacity = 0.4, width = 1)),
     Xformatter = "#",
     Yformatter = "#.00",
     image = list(
       image = amImage(
         href = tinyIcon("react", backgroundColor = "transparent"),
         width = 40, height = 40
       ),
       position = "bottomleft", hjust = 2, vjust = -2
     ),
     theme = "dark")
```

---

amScatterChart   *HTML widget displaying a scatter chart*

Description

Create a HTML widget displaying a scatter chart.

Usage

```
amScatterChart(
  data,
  data2 = NULL,
  xValue,
  yValues,
  yValueNames = NULL,
  hline = NULL,
  vline = NULL,
  xLimits = NULL,
  yLimits = NULL,
  expandX = 0,
  expandY = 5,
  Xformatter = ifelse(isDate, "yyyy-MM-dd", "#."),
  Yformatter = "#.",
  trend = FALSE,
  chartTitle = NULL,
```

```
    theme = NULL,
    animated = TRUE,
    draggable = FALSE,
    tooltip = NULL,
    pointsStyle = NULL,
    backgroundColor = NULL,
    xAxis = NULL,
    yAxis = NULL,
    scrollbarX = FALSE,
    scrollbarY = FALSE,
    legend = NULL,
    caption = NULL,
    image = NULL,
    button = NULL,
    cursor = FALSE,
    zoomButtons = FALSE,
    width = NULL,
    height = NULL,
    export = FALSE,
    chartId = NULL,
    elementId = NULL
)
```

Arguments

data	a dataframe
data2	NULL or a dataframe used to update the data with the button; its column names must include the column names of data given in yValues as well as the column name given in xValue; moreover it must have the same number of rows as data and its rows must be in the same order as those of data
xValue	name of the column of data to be used on the x-axis
yValues	name(s) of the column(s) of data to be used on the y-axis
yValueNames	names of the variables on the y-axis, to appear in the legend; NULL to use yValues as names, otherwise a named list of the form list(yvalue1 = "ValueName1", yvalue2 = "ValueName2", ...) where yvalue1, yvalue2, ... are the column names given in yValues and "ValueName1", "ValueName2", ... are the desired names to appear in the legend
hline	an optional horizontal line to add to the chart; it must be a named list of the form list(value = h, line = settings) where h is the "intercept" and settings is a list of settings created with [amLine](#)
vline	an optional vertical line to add to the chart; it must be a named list of the form list(value = v, line = settings) where v is the "intercept" and settings is a list of settings created with [amLine](#)
xLimits	range of the x-axis, a vector of two values specifying the left and the right limits of the x-axis; NULL for default values
yLimits	range of the y-axis, a vector of two values specifying the lower and the upper limits of the y-axis; NULL for default values

expandX	if `xLimits = NULL`, a percentage of the range of the x-axis used to expand this range
expandY	if `yLimits = NULL`, a percentage of the range of the y-axis used to expand this range
Xformatter	a number formatting string if `xValue` is set to a numeric column of `data`; it is used to format the values displayed in the cursor tooltips if `cursor = TRUE`, the labels of the x-axis unless you specify your own formatter in the `labels` field of the list passed on to the `xAxis` option, and the values displayed in the tooltips unless you specify your own tooltip text; if `xValue` is set to a date column of `data`, this option should be set to a date formatting string, and it has an effect only on the values displayed in the tooltips (unless you specify your own tooltip text); formatting the dates on the x-axis is done via the `labels` field of the list passed on to the `xAxis` option
Yformatter	a number formatting string; it is used to format the values displayed in the cursor tooltips if `cursor = TRUE`, the labels of the y-axis unless you specify your own formatter in the `labels` field of the list passed on to the `yAxis` option, and the values displayed in the tooltips unless you specify your own tooltip text (see the first example of `amBarChart` for the way to set a number formatter in the tooltip text)
trend	option to request trend lines and to set their settings; `FALSE` for no trend line, otherwise a named list of the form `list(yvalue1 = trend1, yvalue2 = trend2, ...)` where `trend1, trend2, ...` are lists with the following fields:
	method the modelling method, can be `"lm"`, `"lm.js"`, `"nls"`, `"nlsLM"`, or `"loess"`; `"lm.js"` performs a polynomial regression in JavaScript, its advantage is that the fitted regression line is updated when the points are dragged
	formula a formula passed on to the modelling function for methods `"lm"`, `"nls"` or `"nlsLM"`; the lefthandside of this formula must always be `y`, and its righthandside must be a symbolic expression depending on `x` only, e.g. `y ~ x, y ~ x + I(x^2), y ~ poly(x,2)`
	interval effective for methods `"lm"` and `"lm.js"` only; a list with five possible fields: type can be `"confidence"` or `"prediction"`, level is the confidence or prediction level (number between 0 and 1), color is the color of the shaded area, opacity is the opacity of the shaded area (number between 0 and 1), tensionX and tensionY to control the smoothing (see `amLine`)
	order the order of the polynomial regression when `method = "lm.js"`
	method.args a list of additional arguments passed on to the modelling function defined by `method` for methods `"nls"`, `"nlsLM"` or `"loess"`, e.g. `method.args = list(span = 0.3)` for method `"loess"`
	style a list of settings for the trend line created with `amLine`
	it is also possible to request the same kind of trend lines for all series given by the `yValues` argument, by passing a list of the form `list("_all" = trendconfig)`, e.g. `list("_all" = list(method = "lm", formula = y ~ 0+x, style = amLine()))`
chartTitle	chart title, it can be `NULL` or `FALSE` for no title, a character string, a list of settings created with `amText`, or a list with two fields: text, a list of settings created with `amText`, and align, can be `"left"`, `"right"` or `"center"`

theme	theme, NULL or one of ″dataviz″, ″material″, ″kelly″, ″dark″, ″moonrisekingdom″, ″frozen″, ″spiritedaway″, ″patterns″, ″microchart″
animated	Boolean, whether to animate the rendering of the graphic
draggable	TRUE/FALSE to enable/disable dragging of all lines, otherwise a named list of the form list(yvalue1 = TRUE, yvalue2 = FALSE, ...) to enable/disable the dragging for each series corresponding to a column given in yValues
tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form list(yvalue1 = settings1, yvalue2 = settings2, ...) where settings1, settings2, ... are lists created with amTooltip; this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip
pointsStyle	settings of the points style; NULL for default, otherwise a named list of the form list(yvalue1 = settings1, yvalue2 = settings2, ...) where settings1, settings2, ... are lists created with amCircle, amTriangle or amRectangle; this can also be a single list of settings that will be applied to each series
backgroundColor	a color for the chart background; it can be given by the name of a R color, the name of a CSS color, e.g. ″aqua″ or ″indigo″, an HEX code like ″#ff009a″, a RGB code like ″rgb(255,100,39)″, or a HSL code like ″hsl(360,11,255)″
xAxis	settings of the x-axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the vertical adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with amLine, and a field breaks to control the axis breaks, an R object created with amAxisBreaks
yAxis	settings of the y-axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the horizontal adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with amLine, and a field breaks to control the axis breaks, an R object created with amAxisBreaks
scrollbarX	logical, whether to add a scrollbar for the x-axis
scrollbarY	logical, whether to add a scrollbar for the y-axis
legend	FALSE for no legend, TRUE for a legend with default settings, or a list of settings created with amLegend
caption	NULL or FALSE for no caption, a formatted text created with amText, or a list with two fields: text, a list created with amText, and align, can be ″left″, ″right″ or ″center″
image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with amImage, the field position can be ″topleft″, ″topright″, ″bottomleft″ or ″bottomright″, the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment

button	NULL for the default, FALSE for no button, or a list of settings created with [amButton](); this button is used to replace the current data with data2
cursor	option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor for both axes with default settings for the axes tooltips, otherwise a named list with four possible fields: a field axes to specify the axes for which the cursor is requested, can be "x", "y", or "xy", a field tooltip to set the style of the axes tooltips, this must be a list of settings created with [amTooltip](), a field extraTooltipPrecision, a named list of the form list(x = i, y = j) where i and j are the desired numbers of additional decimals for the tooltips on the x-axis and on the y-axis respectively, and a field modifier, a list with two possible fields, x and y, which defines modifiers for the values displayed in the tooltips; a modifier is some JavaScript code given a string, which performs a modification of a string named text; see the first example of [amLineChart]() for an example of modifier
zoomButtons	a Boolean value, or a list created with [amZoomButtons]()
width	the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in [amChart4Output]()
height	the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in [amChart4Output]()
export	logical, whether to enable the export menu
chartId	a HTML id for the chart
elementId	a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
# iris data: petal widths ####
dat <- iris
dat$obs <- rep(1:50, 3)
dat <- reshape2::dcast(dat, obs ~ Species, value.var = "Petal.Width")

amScatterChart(
  data = dat,
  width = "700px",
  xValue = "obs",
  yValues = c("setosa", "versicolor", "virginica"),
  draggable = FALSE,
  backgroundColor = "#30303d",
  pointsStyle = list(
    setosa = amCircle(color = "orange", strokeColor = "red"),
    versicolor = amCircle(color = "cyan", strokeColor = "blue"),
    virginica = amCircle(color = "palegreen", strokeColor = "darkgreen")
  ),
  tooltip = "obs: {valueX}\nvalue: {valueY}",
  chartTitle = amText(text = "Iris data", color = "whitesmoke"),
  xAxis = list(title = amText(text = "Observation",
                             fontSize = 21,
                             color = "silver"),
             labels = amAxisLabels(color = "whitesmoke",
```

```
                                             fontSize = 17)),
  yAxis = list(title = amText(text = "Petal width",
                               fontSize = 21,
                               color = "silver"),
               labels = amAxisLabels(color = "whitesmoke",
                                      fontSize = 14),
               gridLines = amLine(color = "whitesmoke",
                                   opacity = 0.4, width = 1)),
  Xformatter = "#",
  Yformatter = "#.0",
  caption = amText(text = "[font-style:italic]rAmCharts4[/]",
                   color = "yellow"),
  theme = "dark")


# iris data: petal widths vs petal lengths

dat <- iris
dat$obs <- rep(1:50, 3)
dat <-
  reshape2::dcast(dat, obs + Petal.Length ~ Species, value.var = "Petal.Width")

amScatterChart(
  data = dat,
  width = "700px",
  xValue = "Petal.Length",
  yValues = c("setosa", "versicolor", "virginica"),
  draggable = FALSE,
  backgroundColor = "#30303d",
  pointsStyle = list(
    setosa = amCircle(color = "orange", strokeColor = "red"),
    versicolor = amCircle(color = "cyan", strokeColor = "blue"),
    virginica = amCircle(color = "palegreen", strokeColor = "darkgreen")
  ),
  tooltip = list(
    setosa = amTooltip(
      text = "length: {valueX}\nwidth: {valueY}",
      backgroundColor = "orange",
      borderColor = "red",
      textColor = "black"
    ),
    versicolor = amTooltip(
      text = "length: {valueX}\nwidth: {valueY}",
      backgroundColor = "cyan",
      borderColor = "blue",
      textColor = "black"
    ),
    virginica = amTooltip(
      text = "length: {valueX}\nwidth: {valueY}",
      backgroundColor = "palegreen",
      borderColor = "darkgreen",
      textColor = "black"
    )
```

```
  ),
  chartTitle = amText(text = "Iris data", color = "silver"),
  xAxis = list(title = amText(text = "Petal length",
                              fontSize = 19,
                              color = "gold"),
               labels = amAxisLabels(color = "whitesmoke",
                                     fontSize = 17)),
  yAxis = list(title = amText(text = "Petal width",
                              fontSize = 19,
                              color = "gold"),
               labels = amAxisLabels(color = "whitesmoke",
                                     fontSize = 17),
               gridLines = amLine(color = "whitesmoke",
                                  opacity = 0.4, width = 1)),
  cursor = list(
    tooltip = amTooltip(backgroundColor = "lightgray"),
    extraTooltipPrecision = list(x = 1, y = 1)
  ),
  caption = amText(text = "[font-style:italic]rAmCharts4[/]",
                   color = "yellow"),
  theme = "dark")


# scatter chart with trend lines ####

Asym = 5; R0 = 1; lrc = -3/4
x <- seq(-.3, 5, len = 101)
y0 <- Asym + (R0-Asym) * exp(-exp(lrc)* x)

dat <- data.frame(
  x = x,
  y1 = y0 + rnorm(101, sd = 0.33),
  y2 = y0 + rnorm(101, sd = 0.33) + 2
)

amScatterChart(
  data = dat,
  width = "700px",
  xValue = "x",
  yValues = c("y1", "y2"),
  trend = list("_all" = list(
    method = "nls",
    formula = y ~ SSasymp(x, Asym, R0, lrc),
    style = amLine()
  )),
  draggable = FALSE,
  pointsStyle = list(
    y1 = amTriangle(
      width = 8,
      height = 8,
      strokeColor = "yellow",
      strokeWidth = 1
    ),
```

```
    y2 = amTriangle(
      width = 8,
      height = 8,
      strokeColor = "chartreuse",
      strokeWidth = 1,
      rotation = 180
    )
  ),
  chartTitle = amText(text = "Asymptotic regression model"),
  xAxis = "x",
  yAxis = "y",
  Xformatter = "#.###",
  Yformatter = "#.",
  theme = "kelly",
  zoomButtons = TRUE)
```

---

amSegment   *Segment style*

Description

Create a list of settings for a segment.

Usage

```
amSegment(color = NULL, width = 1)
```

Arguments

color   color of the segment; this can be a [color adapter](#)

width   width of the segment

Value

A list of settings for a segment.

Note

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

Description

Create a HTML widget displaying a stacked bar chart.

Usage

```
amStackedBarChart(
  data,
  data2 = NULL,
  category,
  stacks,
  seriesNames = NULL,
  colors = NULL,
  hline = NULL,
  yLimits = NULL,
  expandY = 5,
  valueFormatter = "#.",
  chartTitle = NULL,
  theme = NULL,
  animated = TRUE,
  tooltip = NULL,
  threeD = FALSE,
  backgroundColor = NULL,
  cellWidth = NULL,
  columnWidth = NULL,
  xAxis = NULL,
  yAxis = NULL,
  scrollbarX = FALSE,
  scrollbarY = FALSE,
  legend = NULL,
  caption = NULL,
  image = NULL,
  button = NULL,
  cursor = FALSE,
  width = NULL,
  height = NULL,
  export = FALSE,
  chartId = NULL,
  elementId = NULL
)
```

Arguments

data   a dataframe

data2	NULL or a dataframe used to update the data with the button; its column names must include the column names of `data` given in `series`, it must have the same number of rows as `data` and its rows must be in the same order as those of `data`
category	name of the column of `data` to be used on the category axis
stacks	a list of stacks; a stack is a character vector of the form `c("series3", "series1", "series2")`, and the first element of a stack corresponds to the bottom of the column
seriesNames	names of the series variables (the variables which appear in the stacks), to appear in the legend; NULL to use the variables given in `stacks` as names, otherwise a named list of the form `list(series1 = "SeriesName1", series2 = "SeriesName2", ...)` where `series1`, `series2`, ... are the column names given in `stacks` and `"SeriesName1"`, `"SeriesName2"`, ... are the desired names to appear in the legend; these names can also appear in the tooltips: they are substituted to the string `{name}` in the formatting string passed on to the tooltip
colors	colors of the bars; NULL for automatic colors based on the theme, otherwise a named list of the form `list(series1 = Color1, series2 = Color2, ...)` where `series1`, `series2`, ... are the column names given in `stacks`
hline	an optional horizontal line to add to the chart; it must be a named list of the form `list(value = h, line = settings)` where `h` is the "intercept" and `settings` is a list of settings created with [amLine](#)
yLimits	range of the y-axis, a vector of two values specifying the lower and the upper limits of the y-axis; NULL for default values
expandY	if `yLimits = NULL`, a percentage of the range of the y-axis used to expand this range
valueFormatter	a [number formatting string](#); it is used to format the values displayed in the cursor tooltips if `cursor = TRUE`, the labels of the y-axis unless you specify your own formatter in the `labels` field of the list passed on to the `yAxis` option, and the values displayed in the tooltips unless you specify your own tooltip text
chartTitle	chart title, it can be NULL or FALSE for no title, a character string, a list of settings created with [amText](#), or a list with two fields: `text`, a list of settings created with [amText](#), and align, can be `"left"`, `"right"` or `"center"`
theme	theme, NULL or one of `"dataviz"`, `"material"`, `"kelly"`, `"dark"`, `"moonrisekingdom"`, `"frozen"`, `"spiritedaway"`, `"patterns"`, `"microchart"`
animated	Boolean, whether to animate the rendering of the graphic
tooltip	settings of the tooltips; NULL for default, FALSE for no tooltip, otherwise a named list of the form `list(series1 = settings1, series2 = settings2, ...)` where `settings1`, `settings2`, ... are lists created with [amTooltip](#); this can also be a single list of settings that will be applied to each series, or a just a string for the text to display in the tooltip
threeD	logical, whether to render the columns in 3D
backgroundColor	
	a color for the chart background; a color can be given by the name of a R color, the name of a CSS color, e.g. `"rebeccapurple"` or `"fuchsia"`, an HEX code like `"#ff009a"`, a RGB code like `"rgb(255,100,39)"`, or a HSL code like `"hsl(360,11,255)"`

cellWidth	cell width in percent; for a simple bar chart, this is the width of the columns; for a grouped bar chart, this is the width of the clusters of columns; NULL for the default value
columnWidth	column width, a percentage of the cell width; set to 100 for a simple bar chart and use cellWidth to control the width of the columns; for a grouped bar chart, this controls the spacing between the columns within a cluster of columns; NULL for the default value
xAxis	settings of the category axis given as a list, or just a string for the axis title; the list of settings has three possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, and a field adjust, a number defining the vertical adjustment of the axis (in pixels)
yAxis	settings of the value axis given as a list, or just a string for the axis title; the list of settings has five possible fields: a field title, a list of settings for the axis title created with amText, a field labels, a list of settings for the axis labels created with amAxisLabels, a field adjust, a number defining the horizontal adjustment of the axis (in pixels), a field gridLines, a list of settings for the grid lines created with amLine and a field breaks to control the axis breaks, an R object created with amAxisBreaks
scrollbarX	logical, whether to add a scrollbar for the category axis
scrollbarY	logical, whether to add a scrollbar for the value axis
legend	either a logical value, whether to display the legend, or a list of settings for the legend created with amLegend
caption	NULL or FALSE for no caption, a formatted text created with amText, or a list with two fields: text, a list created with amText, and align, can be "left", "right" or "center"
image	option to include an image at a corner of the chart; NULL or FALSE for no image, otherwise a named list with four possible fields: the field image (required) is a list created with amImage, the field position can be "topleft", "topright", "bottomleft" or "bottomright", the field hjust defines the horizontal adjustment, and the field vjust defines the vertical adjustment
button	NULL for the default, FALSE for no button, or a list of settings created with amButton; this button is used to replace the current data with data2
cursor	option to add a cursor on the chart; FALSE for no cursor, TRUE for a cursor with default settings for the tooltips, or a list of settings created with amTooltip to set the style of the tooltips, or a list with three possible fields: a field tooltip, a list of tooltip settings created with amTooltip, a field extraTooltipPrecision, an integer, the number of additional decimals to display in the tooltips, and a field modifier, which defines a modifier for the values displayed in the tooltips; a modifier is some JavaScript code given as a string, which performs a modification of a string named text, e.g. modifier = "text = '>>>' + text;"
width	the width of the chart, e.g. "600px" or "80%"; ignored if the chart is displayed in Shiny, in which case the width is given in amChart4Output
height	the height of the chart, e.g. "400px"; ignored if the chart is displayed in Shiny, in which case the height is given in amChart4Output

export	logical, whether to enable the export menu
chartId	a HTML id for the chart
elementId	a HTML id for the container of the chart; ignored if the chart is displayed in Shiny, in which case the id is given by the Shiny id

Examples

```
library(rAmCharts4)

dat <- data.frame(
  year    = c("2004", "2005", "2006"),
  europe  = c(10, 15, 20),
  asia    = c( 9, 10, 13),
  africa  = c( 5,  6,  8),
  meast   = c( 7,  8, 12),
  namerica = c(12, 15, 19),
  samerica = c(10, 16, 14)
)

dat2 <- data.frame(
  year    = c("2004", "2005", "2006"),
  europe  = c( 7, 12, 16),
  asia    = c( 8, 13, 10),
  africa  = c( 7,  7, 10),
  meast   = c( 8,  6, 14),
  namerica = c(10, 17, 17),
  samerica = c(12, 18, 17)
)

stacks <- list(
  c("europe", "namerica"),
  c("asia", "africa", "meast", "samerica")
)

seriesNames <- list(
  europe = "Europe",
  namerica = "North America",
  asia = "Asia",
  africa = "Africa",
  meast = "Middle East",
  samerica = "South America"
)

amStackedBarChart(
  dat,
  data2 = dat2,
  category = "year",
  stacks = stacks,
  seriesNames = seriesNames,
  yLimits = c(0, 60),
  chartTitle = amText(
    "Stacked bar chart",
```

```
    fontFamily = "Trebuchet MS",
    fontSize = 30,
    fontWeight = "bold"
  ),
  xAxis = "Year",
  yAxis = "A quantity...",
  theme = "kelly",
  button = amButton("Update", position = 1),
  height = 450
)
```

---

amText   *Text*

Description

Create a list of settings for a text.

Usage

```
amText(
  text,
  color = NULL,
  fontSize = NULL,
  fontWeight = "normal",
  fontFamily = NULL
)
```

Arguments

text	the text to display, a character string
color	color of the text; it can be given by the name of a R color, the name of a CSS color, e.g. "crimson", an HEX code like "#ff009a", a RGB code like "rgb(255,100,39)", or a HSL code like "hsl(360,11,255)"
fontSize	size of the text
fontWeight	font weight of the text, it can be "normal", "bold", "bolder", "lighter", or a number in seq(100, 900, by = 100)
fontFamily	font family

Value

A list of settings for a text.

Note

There is no option for the font style; you can get an italicized text by entering text = "[font-style:italic]Your text[/]".

---

amTooltip   *Tooltip*

Description

Create list of settings for a tooltip.

Usage

```
amTooltip(
  text,
  textColor = NULL,
  textAlign = "middle",
  backgroundColor = NULL,
  backgroundOpacity = 0.6,
  borderColor = NULL,
  borderWidth = 2,
  pointerLength = 10,
  scale = 1,
  auto = FALSE
)
```

Arguments

text	text to display in the tooltip; this should be a formatting string
textColor	text color
textAlign	alignement of the text, can be "start", "middle", or "end"
backgroundColor	
	background color of the tooltip
backgroundOpacity	
	background opacity
borderColor	color of the border of the tooltip
borderWidth	width of the border of the tooltip
pointerLength	length of the pointer
scale	scale factor
auto	logical, whether to use automatic background color and text color

Value

A list of settings for a tooltip.

Note

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

---

amZoomButtons *Zoom buttons*

Description

Zoom buttons.

Usage

```
amZoomButtons(
  halign = "left",
  valign = "top",
  marginH = 5,
  marginV = 5,
  zoomFactor = 0.1
)
```

Arguments

halign	"left" or "right"
valign	"top" or "bottom"
marginH	horizontal margin
marginV	vertical margin
zoomFactor	zoom factor

Value

A list of parameters for zoom buttons, for usage in amLineChart or amScatterChart

---

rAmCharts4-adapters *Adapters*

Description

Adapters allow to have finer control of settings such as the colors of the columns of a bar chart or the colors of the points of a scatter chart.

Usage

```
amColorAdapterFromVector(colors)

amColorAdapterFromCuts(cuts, colors, value)
```

Arguments

`colors`	a vector of colors
`cuts`	a vector of cut points (sorted increasingly)
`value`	a mathematical expression of the variables X and Y given as JavaScript code; the simplest examples are ″X″ and ″Y″, a more elaborate example is ″Math.sqrt(X**2+Y**2)″ (don't forget that the power in JavaScript is '**', not '^'!); see the examples

Examples

```
# bar chart with individual colors ####

dat <- data.frame(
  country = c("USA", "China", "Japan", "Germany", "UK", "France"),
  visits = c(3025, 1882, 1809, 1322, 1122, 1114)
)

amBarChart(
  data = dat,
  width = "600px",
  category = "country", values = "visits",
  showValues = FALSE,
  tooltip = FALSE,
  columnStyle = amColumn(
    color = amColorAdapterFromVector(hcl.colors(6, "Viridis")),
    opacity = 0.7,
    strokeColor = amColorAdapterFromVector(hcl.colors(6, "Cividis")),
    strokeWidth = 4
  ),
  bullets = amCircle(
    color = amColorAdapterFromVector(hcl.colors(6, "Viridis")),
    opacity = 1,
    strokeColor = amColorAdapterFromVector(hcl.colors(6, "Cividis")),
    strokeWidth = 4,
    radius = 12
  ),
  alwaysShowBullets = TRUE,
  chartTitle =
    amText(text = "Visits per country", fontSize = 22, color = "orangered"),
  backgroundColor = "rgb(164,167,174)",
  xAxis = list(title = amText(text = "Country", color = "maroon")),
  yAxis = list(
    title = amText(text = "Visits", color = "maroon"),
    gridLines = amLine(color = "white", width = 1, dash = "3,3")
  ),
  yLimits = c(0, 4000),
  valueFormatter = "#,###.",
  caption = amText(text = "Year 2018", color = "red")
)


# usage example of amColorAdapterFromCuts ####
```

```
set.seed(314159)
dat <- data.frame(
  x = rnorm(200),
  y = rnorm(200)
)

amScatterChart(
  data = dat,
  width = "500px", height = "500px",
  xValue = "x", yValues = "y",
  xLimits = c(-3,3), yLimits = c(-3,3),
  draggable = FALSE,
  backgroundColor = "#30303d",
  pointsStyle = amCircle(
    color = amColorAdapterFromCuts(
      cuts = c(-2, -1, 1, 2),
      colors = c("red", "green", "blue", "green", "red"),
      value = "Y"
    ),
    opacity = 0.5,
    strokeColor = amColorAdapterFromCuts(
      cuts = c(-2, -1, 1, 2),
      colors = c("darkred", "darkgreen", "darkblue", "darkgreen", "darkred"),
      value = "Y"
    )
  ),
  xAxis = list(
    breaks = amAxisBreaks(seq(-3, 3, by=1)),
    gridLines = amLine(opacity = 0.3, width = 1)
  ),
  yAxis = list(
    breaks = amAxisBreaks(seq(-3, 3, by=1)),
    gridLines = amLine(opacity = 0.3, width = 1)
  ),
  tooltip = FALSE,
  caption = amText(text = "[font-style:italic]rAmCharts4[/]",
                   color = "yellow"),
  theme = "dark")


# other usage example of amColorAdapterFromCuts: linear gradient ####

set.seed(314159)
dat <- data.frame(
  x = rnorm(500),
  y = rnorm(500)
)

amScatterChart(
  data = dat,
  width = "500px", height = "500px",
  xValue = "x", yValues = "y",
```

```
    xLimits = c(-3,3), yLimits = c(-3,3),
    draggable = FALSE,
    backgroundColor = "#30303d",
    pointsStyle = amCircle(
      radius = 4,
      strokeWidth = 1,
      color = amColorAdapterFromCuts(
        cuts = seq(-3, 3, length.out = 121),
        colors = colorRampPalette(
          c("red","orangered","blue","white","blue","orangered","red")
        )(122),
        value = "X"
      ),
      opacity = 0.75,
      strokeColor = amColorAdapterFromCuts(
        cuts = seq(-3, 3, length.out = 121),
        colors = colorRampPalette(
          c("red","orangered","blue","white","blue","orangered","red")
        )(122),
        value = "X"
      )
    ),
    xAxis = list(
      breaks = amAxisBreaks(seq(-3, 3, by=1)),
      gridLines = amLine(opacity = 0.3, width = 1)
    ),
    yAxis = list(
      breaks = amAxisBreaks(seq(-3, 3, by=1)),
      gridLines = amLine(opacity = 0.3, width = 1)
    ),
    tooltip = FALSE,
    caption = amText(text = "[font-style:italic]rAmCharts4[/]",
                     color = "yellow"),
    theme = "dark")


# yet another usage example of amColorAdapterFromCuts: radial gradient

set.seed(314159)
dat <- data.frame(
  x = rnorm(1000),
  y = rnorm(1000)
)

amScatterChart(
  data = dat,
  width = "500px", height = "500px",
  xValue = "x", yValues = "y",
  xLimits = c(-3,3), yLimits = c(-3,3),
  draggable = FALSE,
  backgroundColor = "#30303d",
  pointsStyle = amCircle(
    radius = 4,
```

```
        strokeWidth = 1,
        color = amColorAdapterFromCuts(
          cuts = seq(0, 3, length.out = 121),
          colors = colorRampPalette(
            c("white","blue","orangered","red")
          )(122),
          value = "Math.sqrt(X**2+Y**2)"
        ),
        opacity = 0.75,
        strokeColor = amColorAdapterFromCuts(
          cuts = seq(0, 3, length.out = 121),
          colors = colorRampPalette(
            c("white","blue","orangered","red")
          )(122),
          value = "Math.sqrt(X**2+Y**2)"
        )
      ),
      xAxis = list(
        breaks = amAxisBreaks(seq(-3, 3, by=1)),
        gridLines = amLine(opacity = 0.3, width = 1)
      ),
      yAxis = list(
        breaks = amAxisBreaks(seq(-3, 3, by=1)),
        gridLines = amLine(opacity = 0.3, width = 1)
      ),
      tooltip = FALSE,
      caption = amText(text = "[font-style:italic]rAmCharts4[/]",
                       color = "yellow"),
      theme = "dark")
```

---

rAmCharts4-imports   *Objects imported from other packages*

Description

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

---

rAmCharts4-shapes   *Bullets*

Description

Create a list of settings for bullets, their shape and their style.

Usage

```
amTriangle(
  color = NULL,
  opacity = 1,
  width = 10,
  height = 10,
  strokeColor = NULL,
  strokeOpacity = 1,
  strokeWidth = 2,
  direction = "top",
  rotation = 0,
  image = NULL
)

amCircle(
  color = NULL,
  opacity = 1,
  radius = 6,
  strokeColor = NULL,
  strokeOpacity = 1,
  strokeWidth = 2,
  image = NULL
)

amRectangle(
  color = NULL,
  opacity = 1,
  width = 10,
  height = 10,
  strokeColor = NULL,
  strokeOpacity = 1,
  strokeWidth = 2,
  rotation = 0,
  cornerRadius = 3,
  image = NULL
)
```

Arguments

color	bullet color; this can be a [color adapter](#)
opacity	bullet opacity, a number between 0 and 1
width	bullet width
height	bullet height
strokeColor	stroke color of the bullet; this can be a [color adapter](#)
strokeOpacity	stroke opacity of the bullet, a number between 0 and 1
strokeWidth	stroke width of the bullet

direction	triangle direction
rotation	rotation angle
image	option to include an image in the bullet, a list created with amImage
radius	circle radius
cornerRadius	radius of the rectangle corners

Value

A list of settings for the bullets.

Note

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

---

rAmCharts4-shiny   *Shiny bindings for using* rAmCharts4 *in Shiny*

Description

Output and render functions for using the `rAmCharts4` widgets within Shiny applications and interactive Rmd documents.

Usage

```
amChart4Output(outputId, width = "100%", height = "400px")

renderAmChart4(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

outputId	output variable to read from
width, height	must be a valid CSS unit (like `"100%"`, `"400px"`, `"auto"`) or a number, which will be coerced to a string and have `"px"` appended
expr	an expression that generates a chart with amBarChart, amHorizontalBarChart, amLineChart, amScatterChart, amRangeAreaChart, amRadialBarChart, amDumbbellChart, amHorizontalDumbbellChart, amGaugeChart, amPieChart, or amPercentageBarChart
env	the environment in which to evaluate expr
quoted	whether expr is a quoted expression

Examples

```
library(rAmCharts4)
library(shiny)
library(lubridate)

ui <- fluidPage(
  br(),
  fluidRow(
    column(
      width = 8,
      amChart4Output("linechart", height = "500px")
    ),
    column(
      width = 4,
      tags$fieldset(
        tags$legend("Chart data"),
        verbatimTextOutput("chartData"),
      ),
      tags$fieldset(
        tags$legend("Change"),
        verbatimTextOutput("chartChange")
      )
    )
  )
)

server <- function(input, output){

  set.seed(666)
  dat <- data.frame(
    date = ymd(180101) + months(0:11),
    visits = rpois(12, 20),
    x = 1:12
  )

  output[["linechart"]] <- renderAmChart4({
    amLineChart(
      data = dat,
      data2 = dat,
      xValue = "date",
      yValues = "visits",
      draggable = TRUE,
      chartTitle = amText(
        text = "Number of visits",
        color = "crimson",
        fontWeight = "bold",
        fontFamily = "cursive"
      ),
      xAxis = list(
        title = "Date",
        labels = amAxisLabels(rotation = -45),
        breaks = amAxisBreaks(timeInterval = "1 month")
```

```
      ),
      yAxis = "Visits",
      yLimits = c(0, 35),
      backgroundColor = "whitesmoke",
      tooltip = "[bold][font-style:italic]{dateX}[/]\nvisits: {valueY}[/]",
      Yformatter = "#",
      caption = amText(
        text = "[bold font-size:22]Year 2018[/]",
        color = "fuchsia"
      ),
      button = amButton(
        label = amText("Reset data", color = "black"),
        color = "seashell",
        position = 0.95
      ),
      theme = "dataviz")
  })

  output[["chartData"]] <- renderPrint({
    input[["linechart"]]
  })

  output[["chartChange"]] <- renderPrint({
    input[["linechart_change"]]
  })

}

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

---

tinyIcon   *Icons*

Description

Icons for usage in amImage.

Usage

```
tinyIcon(icon, backgroundColor = NULL)

tinyIcons()

shinyAppTinyIcons()
```

Arguments

icon   name of an icon; `tinyIcons()` returns the list of available icons, and `shinyAppTinyIcons()`
                    runs a Shiny app which displays the available icons

backgroundColor

                    background color of the icon (possibly `"transparent"`)

Value

A base64 string that can be used in the `href` argument of [amImage](#).

Note

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

---

updateAmBarChart   *Update the data of a bar chart*

Description

Update the data of a bar chart in a Shiny app (vertical, horizontal, radial, or stacked bar chart).

Usage

```
updateAmBarChart(session, outputId, data)
```

Arguments

session   the Shiny `session` object

outputId   the output id passed on to [amChart4Output](#)

data   new data; if it is not valid, then nothing will happen (in order to be valid it must
                    have the same structure as the data passed on to [amBarChart](#) / [amHorizontalBarChart](#)
                    / [amRadialBarChart](#) / [amStackedBarChart](#)); in this case check the JavaScript
                    console, it will report the encountered issue

Examples

```
library(rAmCharts4)
library(shiny)

ui <- fluidPage(
  br(),
  actionButton("update", "Update", class = "btn-primary"),
  br(), br(),
  amChart4Output("barchart", width = "650px", height = "470px")
)
```

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

  set.seed(666)
  dat <- data.frame(
    country = c("USA", "China", "Japan", "Germany", "UK", "France"),
    visits = c(3025, 1882, 1809, 1322, 1122, 1114),
    income = rpois(6, 25),
    expenses = rpois(6, 20)
  )
  newdat <- data.frame(
    country = c("USA", "China", "Japan", "Germany", "UK", "France"),
    income = rpois(6, 25),
    expenses = rpois(6, 20)
  )

  output[["barchart"]] <- renderAmChart4({
    amBarChart(
      data = dat,
      category = "country",
      values = c("income", "expenses"),
      valueNames = list(income = "Income", expenses = "Expenses"),
      draggable = TRUE,
      backgroundColor = "#30303d",
      columnStyle = list(
        income = amColumn(
          color = "darkmagenta", strokeColor = "#cccccc", strokeWidth = 2
        ),
        expenses = amColumn(
          color = "darkred", strokeColor = "#cccccc", strokeWidth = 2
        )
      ),
      chartTitle = list(text = "Income and expenses per country"),
      xAxis = "Country",
      yAxis = "Income and expenses",
      yLimits = c(0, 41),
      valueFormatter = "#.#",
      caption = "Year 2018",
      theme = "dark")
  })

  observeEvent(input[["update"]], {
    updateAmBarChart(session, "barchart", newdat)
  })

}

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


# Survival probabilities ####
```

```
library(shiny)
library(rAmCharts4)

probs <- c(control = 30, treatment = 75) # initial probabilities

ui <- fluidPage(
  br(),
  sidebarLayout(
    sidebarPanel(
      wellPanel(
        tags$fieldset(
          tags$legend("Survival probability"),
          sliderInput(
            "control",
            "Control group",
            min = 0, max = 100, value = probs[["control"]], step = 1
          ),
          sliderInput(
            "treatment",
            "Treatment group",
            min = 0, max = 100, value = probs[["treatment"]], step = 1
          )
        )
      )
    ),
    mainPanel(
      amChart4Output("barchart", width = "500px", height = "400px")
    )
  )
)

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

  dat <- data.frame(
    group = c("Control", "Treatment"),
    alive = c(probs[["control"]], probs[["treatment"]]),
    dead  = 100 - c(probs[["control"]], probs[["treatment"]])
  )
  stacks <- list(
    c("alive", "dead")
  )
  seriesNames <- list(
    alive = "Alive",
    dead  = "Dead"
  )

  output[["barchart"]] <- renderAmChart4({
    amStackedBarChart(
      dat,
      category = "group",
      stacks = stacks,
      seriesNames = seriesNames,
      yLimits = c(0, 100),
```

```
        chartTitle = amText(
          "Survival probabilities",
          fontFamily = "Trebuchet MS",
          fontSize = 30,
          fontWeight = "bold"
        ),
        xAxis = "Group",
        yAxis = "Probability",
        theme = "dataviz"
      )
    })

    observeEvent(list(input[["control"]], input[["treatment"]]), {
      newdat <- data.frame(
        group = c("Control", "Treatment"),
        alive = c(input[["control"]], input[["treatment"]]),
        dead  = 100 - c(input[["control"]], input[["treatment"]])
      )
      updateAmBarChart(session, "barchart", newdat)
    })

  }

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

---

updateAmGaugeChart   *Update the score of a gauge chart*

Description

Update the score of a gauge chart in a Shiny app

Usage

```
updateAmGaugeChart(session, outputId, score)
```

Arguments

session	the Shiny session object
outputId	the output id passed on to amChart4Output
score	new value of the score

Examples

```
library(rAmCharts4)
library(shiny)
```

```
gradingData <- data.frame(
  label = c("Slow", "Moderate", "Fast"),
  lowScore = c(0, 100/3, 200/3),
  highScore = c(100/3, 200/3, 100)
)


ui <- fluidPage(
  sidebarLayout(
    sidebarPanel(
      sliderInput(
        "slider", "Score", min = 0, max = 100, value = 30
      )
    ),
    mainPanel(
      amChart4Output("gauge", height = "500px")
    )
  )
)

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

  output[["gauge"]] <- renderAmChart4({
    amGaugeChart(
      score = isolate(input[["slider"]]),
      minScore = 0, maxScore = 100, gradingData = gradingData,
      theme = "dataviz"
    )
  })

  observeEvent(input[["slider"]], {
    updateAmGaugeChart(session, "gauge", score = input[["slider"]])
  })

}

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

updateAmPercentageBarChart

Update the data of a 100% stacked bar chart

Description

Update the data of a 100% staced bar chart in a Shiny app (amPercentageBarChart).

Usage

```
updateAmPercentageBarChart(session, outputId, data)
```

Arguments

session	the Shiny session object
outputId	the output id passed on to amChart4Output
data	new data; if it is not valid, then nothing will happen (in order to be valid it must have the same structure as the data passed on to amPercentageBarChart); in this case check the JavaScript console, it will report the encountered issue

Examples

```
library(rAmCharts4)
library(shiny)

dat <- data.frame(
  country = c("Australia", "Canada", "France", "Germany"),
  "35-44" = c(2, 2, 3, 3),
  "45-54" = c(9, 5, 7, 6),
  "55+"   = c(8, 4, 6, 5),
  check.names = FALSE
)

newdat <- data.frame(
  country = c("Australia", "Canada", "France", "Germany"),
  "35-44" = c(3, 2, 3, 4),
  "45-54" = c(7, 3, 5, 5),
  "55+"   = c(7, 4, 5, 3),
  check.names = FALSE
)


ui <- fluidPage(
  br(),
  actionButton("update", "Update", class = "btn-primary"),
  br(), br(),
  amChart4Output("pbarchart", width = "650px", height = "470px")
)

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

  output[["pbarchart"]] <- renderAmChart4({
    amPercentageBarChart(
      dat,
      category = "country",
      values = c("35-44", "45-54", "55+"),
      chartTitle = "Profit by country and age breakdowns",
      xAxis = "Country",
      yAxis = "Profit",
      theme = "moonrisekingdom",
      legend = amLegend(position = "right")
    )
  })
```

```
  observeEvent(input[["update"]], {
    updateAmPercentageBarChart(session, "pbarchart", newdat)
  })

}

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

updateAmPieChart   *Update the data of a pie chart*

Description

Update the data of a pie chart in a Shiny app.

Usage

```
updateAmPieChart(session, outputId, data)
```

Arguments

session	the Shiny session object
outputId	the output id passed on to amChart4Output
data	new data; if it is not valid, then nothing will happen (in order to be valid it must have the same structure as the data passed on to amPieChart); in this case check the JavaScript console, it will report the encountered issue

Index