Package ‘rAmCharts’

December 6, 2019

Title JavaScript Charts Tool

Description Provides an R interface for using 'AmCharts' Library. Based on 'htmlwidgets', it provides a global architecture to generate 'JavaScript' source code for charts. Most of classes in the library have their equivalent in R with S4 classes; for those classes, not all properties have been referenced but can easily be added in the constructors. Complex properties (e.g. 'JavaScript' object) can be passed as named list. See examples at <http://datastorm-open.github.io/introduction_ramcharts/> and <http://www.amcharts.com/> for more information about the library. The package includes the free version of 'AmCharts' Library. Its only limitation is a small link to the web site displayed on your charts. If you enjoy this library, do not hesitate to refer to this page <http://www.amcharts.com/online-store/> to purchase a licence, and thus support its creators and get a period of Priority Support. See also <http://www.amcharts.com/about/> for more information about 'AmCharts' company.

Version 2.1.13

Maintainer Benoit Thieurmel <benoit.thieurmel@datastorm.fr>

License GPL (>= 2)

URL http://datastorm-open.github.io/introduction_ramcharts/

BugReports https://github.com/datastorm-open/rAmCharts/issues/

Depends R (>= 3.2.0)

Collate 'amChartsAPI.R' 'base_data.R' 'base_startupMessage.R' 'chart_amBarplot.R' 'chart_amBoxplot.R' 'chart_amBullet.R' 'chart_amCandlestick.R' 'chart_amFloatingBar.R' 'chart_amFunnel.R' 'chart_amGauge.R' 'chart_amHist.R' 'chart_amMekko.R' 'chart_amOHLC.R' 'chart_amPie.R' 'chart_amPlot.R' 'chart_amRadar.R' 'chart_amTimeSeries.R' 'chart_amWaterfall.R' 'chart_amWind.R' 'util_sharedGenerics.R' 'class_AmObject.R' 'class_AxisBase.R' 'class_ValueAxis.R' 'util_basicClassUnions.R' 'class_TrendLine.R' 'class_Title.R' 'class_AmChart.R' 'class_StockPanel.R' 'class_StockEvent.R' 'class_PeriodSelector.R' 'class_Label.R' 'class_Guide.R'
R topics documented:

'class_GaugeAxis.R' 'class_GaugeBand.R' 'class_GaugeArrow.R' 'class_DataSet.R' 'class_ChartScrollbar.R' 'class_ChartCursor.R' 'class_AmLegend.R' 'class_AmGraph.R' 'class_AmBalloon.R' 'classUnions.R' 'class_AmChart_constructors.R' 'class_CategoryAxis.R' 'class_AmChart_setters.R' 'class_AmChart_shinyUtils.R' 'class_AmStockChart.R' 'class_AmStockChart_setters.R' 'class_AxisBase_setters.R' 'class_ChartCursor_setters.R' 'class_ChartScrollbar_setters.R' 'class_DataSetSelector.R' 'class_DataSet_setters.R' 'class_GaugeArrow_setters.R' 'class_GaugeAxis_setters.R' 'class_Guide_setters.R' 'class_StockEvent_setters.R' 'class_StockPanel_setters.R' 'class_TrendLine_setters.R' 'shiny_examples.R' 'union_AmCharts.R' 'utils.R' 'utils_amOptions.R' 'utils_amTests.R'

**Imports** methods, htmlwidgets, htmltools, data.table (>= 1.9.6), graphics, utils, pipeR, knitr, grDevices, yaml, zoo

**Suggests** magrittr, shiny, testthat, shinydashboard, base64enc, lubridate, jsonlite

**RoxygenNote** 7.0.0

**LazyData** true

**NeedsCompilation** no

**Author** Benoit Thieurmel [aut, cre], Antanas Marcelionis [aut, cph] ('AmCharts' library in the directory htmlwidgets/lib/amcharts, refer to http://www.amcharts.com/), Jeffery Petit [aut, ctb], Elena Salette [aut, ctb], Titouan Robert [aut, ctb]

**Repository** CRAN

**Date/Publication** 2019-12-06 15:50:05 UTC

---

R topics documented:

- `addListener` .................................................................... 5
- `add_animate_dependency` ..................................................... 7
- `add_dataloader_dependency` .................................................. 8
- `add_export_dependency` ......................................................... 8
- `add_responsive_dependency` ................................................. 9
- `add_theme_dependency` ......................................................... 9
- `amAngularGauge` ............................................................... 10
- `AmBalloon-class` .............................................................. 11
- `amBarplot` ...................................................................... 12
- `amBoxplot` .................................................................... 15
<table>
<thead>
<tr>
<th>R topics documented:</th>
</tr>
</thead>
<tbody>
<tr>
<td>amBullet</td>
</tr>
<tr>
<td>amCandlestick</td>
</tr>
<tr>
<td>AmChart-class</td>
</tr>
<tr>
<td>amChartsAPI</td>
</tr>
<tr>
<td>amChartsOutput</td>
</tr>
<tr>
<td>amFloatingBar</td>
</tr>
<tr>
<td>amFunnel</td>
</tr>
<tr>
<td>AmGraph-class</td>
</tr>
<tr>
<td>amHist</td>
</tr>
<tr>
<td>AmLegend-class</td>
</tr>
<tr>
<td>amLines</td>
</tr>
<tr>
<td>amMekko</td>
</tr>
<tr>
<td>AmObject-class</td>
</tr>
<tr>
<td>amOHLC</td>
</tr>
<tr>
<td>amOptions</td>
</tr>
<tr>
<td>amPie</td>
</tr>
<tr>
<td>amPlot</td>
</tr>
<tr>
<td>amRadar</td>
</tr>
<tr>
<td>amSolidGauge</td>
</tr>
<tr>
<td>AmStockChart-class</td>
</tr>
<tr>
<td>amStockMultiSet</td>
</tr>
<tr>
<td>amTimeSeries</td>
</tr>
<tr>
<td>amWaterfall</td>
</tr>
<tr>
<td>amWind</td>
</tr>
<tr>
<td>api</td>
</tr>
<tr>
<td>AxisBase-class</td>
</tr>
<tr>
<td>CategoryAxis-class</td>
</tr>
<tr>
<td>ChartCursor-class</td>
</tr>
<tr>
<td>ChartScrollbar-class</td>
</tr>
<tr>
<td>controlShinyPlot</td>
</tr>
<tr>
<td>DataSet-class</td>
</tr>
<tr>
<td>DataSetSelector-class</td>
</tr>
<tr>
<td>data_AirPassengers</td>
</tr>
<tr>
<td>data_bar</td>
</tr>
<tr>
<td>data_candleStick1</td>
</tr>
<tr>
<td>data_candleStick2</td>
</tr>
<tr>
<td>data_fbar</td>
</tr>
<tr>
<td>data_funnel</td>
</tr>
<tr>
<td>data_gantt</td>
</tr>
<tr>
<td>data_gbar</td>
</tr>
<tr>
<td>data_gdp</td>
</tr>
<tr>
<td>data_mekko</td>
</tr>
<tr>
<td>data_pie</td>
</tr>
<tr>
<td>data_radar</td>
</tr>
<tr>
<td>data_stock1</td>
</tr>
<tr>
<td>data_stock_2</td>
</tr>
<tr>
<td>data_stock_3</td>
</tr>
<tr>
<td>data_waterfall</td>
</tr>
</tbody>
</table>
R topics documented:

data_wind .................................................. 70
GaugeArrow-class .......................................... 71
GaugeAxis-class ............................................. 71
GaugeBand-class ............................................. 72
Generics functions .......................................... 72
getCurrentStockData ....................................... 74
getTransformTS .............................................. 75
Guide-class .................................................. 76
initialize,AmBalloon-method .............................. 77
initialize,AmChart-method .................................. 78
initialize,AmGraph-method .................................. 91
initialize,AmLegend-method .................................. 93
initialize,AmStockChart-method ............................ 94
initialize,CategoryAxis-method ............................ 99
initialize,ChartCursor-method .............................. 100
initialize,ChartScrollbar-method .......................... 101
initialize,DataSet-method ................................... 102
initialize,DataSetSelector-method ........................ 104
initialize,GaugeArrow-method ................................ 105
initialize,GaugeAxis-method ................................ 106
initialize,GaugeBand-method ................................ 107
initialize,Guide-method ..................................... 108
initialize,Label-method .................................... 109
initialize,PeriodSelector-method ........................... 111
initialize,StockEvent-method ............................... 112
initialize,StockPanel-method ............................... 113
initialize,Title-method .................................... 116
initialize,TrendLine-method .................................. 117
initialize,ValueAxis-method ................................ 119
Label-class .................................................. 120
listProperties ............................................... 121
PeriodSelector-class ........................................ 121
plot,AmCharts-method ....................................... 122
print,AmObject-method ...................................... 122
rAmCharts-shinymodules .................................... 123
rAmCharts-shinymodules-ts .................................. 124
renderAmCharts ............................................... 129
runExamples ............................................... 129
setExport .................................................... 130
show,AmChart-method ....................................... 131
show,AmObject-method ....................................... 131
show,AmStockChart-method .................................. 132
StockEvent-class ............................................ 132
stockGraph .................................................... 133
stockLegend ................................................... 133
StockPanel-class ............................................. 134
Title-class .................................................. 136
TrendLine-class ............................................... 136
Description

Update an object of inherited class `AxisBase`.

Usage

```r
## S4 method for signature 'AxisBase, GuideOrMissing'
addGuide(.Object, guide = NULL, ...)
```

Arguments

- `.Object` children class of `AxisBase`.
- `guide` (optional) `Guide`.
- `...` properties of `Guide` Argument for method `addGuide`.

Value

(possibly updated) `.Object` of class `AxisBase`.

Examples

```r
addGuide(.Object = valueAxis(), fillAlpha = .4, adjustBorderColor = TRUE, gridThickness = 1)
# equivalent to:
guide_obj <- guide(fillAlpha = .4, adjustBorderColor = TRUE, gridThickness = 1)
addGuide(.Object = valueAxis(), guide = guide_obj)
```

addListener

`AmObject` methods

Description

Methods for inherited classes.
addListener

Usage

addListener(.Object, name, expression)

## S4 method for signature 'AmObject,character,character'
addListener(.Object, name, expression)

resetProperties(.Object, ...)

## S4 method for signature 'AmObject'
resetProperties(.Object, ...)

setProperties(.Object, list_prop, ...)

## S4 method for signature 'AmObject'
setProperties(.Object, list_prop, ...)

Arguments

,Object AmObject.
name character, name of the listener.
expression character, associated function event.
... other properties
list_prop (Optional) list containing properties to set. The former properties will be over-written.

Details

Former properties will be overwritten.
Former properties will be overwritten.

Value

The updated object.

Examples

addListener(.Object = amPieChart(),
    name = "clickSlice",
    expression = "function(event){ alert('ok !'); }")

addListener(.Object = amLegend(),
    name = "select",
    expression = paste0("function onSelect (properties) {",
    "alert('selected nodes: ' + properties.nodes);",
    "}")
)

library(pipeR)
amPlot(runif(10)) %>% resetProperties("categoryAxis") %>% print(withDetail = FALSE)
library(pipeR)
# either you can set a list
ls <- list(categoryAxis = list(gridPosition = "start"), fontSize = 15)
amSerialChart() %>% setProperties(list = ls) %>% print()

# or you can set one or more properties
amPieChart() %>% setProperties(handDrawn = TRUE, fontSize = 15) %>% print()

# overwrite a property
amPieChart() %>% setProperties(fontSize = 15) %>% setProperties(fontSize = 12) %>% print()

# Carefull if you try to set a property which is a slot...
# in that case, use the setter methods 'setXX' or 'addXX' which check the validity
## Not run:
amPieChart()

## End(Not run)
amPieChart() %>% setExport()

---

**add_animate_dependency**

*Add animate dependency*

### Description

Add the 'animate' dependency to an htmlwidget. You can only manipulate the htmlwidget if you call the method 'plot' with an rAmChart.

### Usage

```r
add_animate_dependency(widget)
```

### Arguments

- `widget` An htmlwidget.

### Value

Return an updated htmlwidget with the dependency.
add_dataloader_dependency

Add dataloader dependency

Description

Add the 'dataloader' dependency to an htmlwidget. You can only manipulate the htmlwidget if you call the method 'plot' with an rAmChart.

Usage

add_dataloader_dependency(widget)

Arguments

widget An htmlwidget

Value

Return the updated htmlwidget.

add_export_dependency

Add the export dependency to an htmlwidget

Description

Add the 'export' dependency to an htmlwidget. You can only manipulate the htmlwidget if you call the method 'plot' with an rAmChart.

Usage

add_export_dependency(widget)

Arguments

widget An htmlwidget.

Value

Return the updated widget with the 'export' dependency.
add_responsive_dependency

Description

Add the 'responsive' dependency to an htmlwidget. You can only manipulate the htmlwidget if you call the method 'plot' with an rAmChart.

Usage

add_responsive_dependency(widget)

Arguments

widget An htmlwidget.

Value

Return an updated htmlwidget with the dependency.

add_theme_dependency

Description

Add the 'theme' dependency to an htmlwidget. You can only manipulate the htmlwidget if you call the method 'plot' with an rAmChart.

Usage

add_theme_dependency(
  widget,
  theme_js = c("light.js", "patterns.js", "dark.js", "chalk.js")
)

Arguments

widget An htmlwidget.
theme_js A character indicating the JS file dependency.

Value

Return the updated htmlwidget.
Examples

library(pipeR)
amPlot(1:10, theme = "dark")

---

amAngularGauge

### Plotting gauge using rAmCharts

#### Description

amAngularGauge computes a gauge of the given value.

#### Usage

```r
amAngularGauge(
  x,
  start = 0,
  end = 100,
  step = 20,
  bands = data.frame(start = numeric(), end = numeric(), color = character(), width =
    numeric(), stringsAsFactors = FALSE),
  text = "",
  textSize = 25,
  secondAxe = FALSE,
  start2 = 0,
  end2 = 100,
  step2 = 20,
  bands2 = data.frame(start = numeric(), end = numeric(), color = character(),
    stringsAsFactors = FALSE),
  ...
)
```

#### Arguments

- `x` numeric, value for which the angular gauge is desired.
- `start` numeric, minimum value allowed.
- `end` numeric, maximum value allowed.
- `step` numeric, intervals size.
- `bands` data.frame with 4 columns: start (numeric, minimal value for the band), end (numeric, maximal value for the band), color (character, color of the band, in hexadecimal) and width (numeric, width of the band). If the last column is not defined, it is automatically set to 10.
- `text` character, text legend.
AmBalloon-class

**textSize** numeric, text size.

**secondAxe** logical, TRUE if two axes are desired. Default is set to FALSE.

**start2** numeric, minimum value allowed for the second axe if secondAxe is TRUE.

**end2** numeric, maximum value allowed for the second axe if secondAxe is TRUE.

**step2** numeric, intervals size for the second axe if secondAxe is TRUE.

**bands2** data.frame with 4 columns : start (numeric, minimal value for the band), end (numeric, maximal value for the band), color (character, color of the band, in hexadecimal) and width (numeric, width of the band). If the last column is not defined, it is automatically set to 10.

... see amOptions for more options.

References

See online documentation [https://datastorm-open.github.io/introduction_ramcharts/](https://datastorm-open.github.io/introduction_ramcharts/) and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

amAngularGauge(10)

---

**AmBalloon-class** *AmBalloon class*

**Description**

Creates the balloons (tooltips) of the chart. It follows the mouse cursor when you roll-over the data items. The framework generates the instances automatically, you just have to adjust the appearance to your needs.

**Details**

Run api("AmBalloon") for more information and all available properties.
amBarplot

Plotting bar chart using rAmCharts

Description

amBarplot computes a bar chart of the given values.

Usage

```r
amBarplot(
  x,
  y,
  data,
  xlab = "",
  ylab = "",
  ylim = NULL,
  groups_color = NULL,
  horiz = FALSE,
  stack_type = c("none", "regular", "100"),
  layered = FALSE,
  show_values = FALSE,
  depth = 0,
  dataDateFormat = NULL,
  minPeriod = ifelse(!is.null(dataDateFormat), "DD", ""),
  ...
)
```
Arguments

- **x** character, column name for x-axis or numeric, value of the corresponding column. It is optional if argument data has row names.
- **y** character, column name for y-axis or numeric vector of the corresponding column. If you want to display a grouped barchart or a stacked barchart, y is a vector of characters or numerics.
- **data** data.frame, dataframe with values to display. You can add a column "color" (character, colors in hexadecimal). You can also add a column "description" (character) containing the text you want to display when mouse is on the graphic ('<br>' for a new line). See data_bar and data_gbar.
- **xlab** character, label for x-axis.
- **ylab** character, label for y-axis.
- **ylim** limits for the y axis.
- **groups_color** character, vector of colors in hexadecimal, same length as y.
- **horiz** logical, TRUE for an horizontal chart, FALSE for a vertical one. If 'horiz' is set to TRUE, the setting 'labelRotation' will be ignored.
- **stack_type** character, "regular" if you wish stacked bars, "100" if you want 100 percent stacked bars. Default is set to "none".
- **layered** logical, TRUE for layered bars. If TRUE, stack_type must be set to "none".
- **show_values** logical, TRUE to display values.
- **depth** numeric, if > 0, chart is displayed in 3D. Value between 0 and 100.
- **dataDateFormat** character, default set to NULL. Even if your chart parses dates, you can pass them as strings in your dataframe - all you need to do is to set data date format and the chart will parse dates to date objects. Check this page for available formats. Please note that two-digit years (YY) as well as literal month names (MMM) are NOT supported in this setting.
- **minPeriod** Specifies the shortest period of your data. This should be set only if dataDateFormat is not 'NULL'. Possible period values: fff - milliseconds, ss - seconds, mm - minutes, hh - hours, DD - days, MM - months, YYYY - years. It’s also possible to supply a number for increments, i.e. '15mm' which will instruct the chart that your data is supplied in 15 minute increments.

... see amOptions for more options.

Details

**Notice about labels:** if the chart has many columns, several labels might be hidden. It depends on the width of the container where the chart is displayed. Zoom on the chart to see if the chart can contain all labels. If not, use the parameter labelRotation. You can also add a cursor to your chart...

Value

An object of class AmChart.
amBarplot

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

```r
# Data
data(data_bar)
data(data_gbar)

amBarplot(x = "country", y = "visits", data = data_bar, main = "example")

# Other examples available which can be time consuming depending on your configuration.

# fixed value axis
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar, ylim = c(0, 26))
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar, stack_type = "100")

# Test with label rotation
amBarplot(x = "country", y = "visits", data = data_bar, labelRotation = -45)

# Horizontal bar
amBarplot(x = "country", y = "visits", data = data_bar, horiz = TRUE, labelRotation = -45)

# 3D bar
amBarplot(x = "country", y = "visits", data = data_bar, depth = 15, labelRotation = -45)

# Display values
amBarplot(x = "country", y = "visits", data = data_bar, show_values = TRUE, labelRotation = -45)

# Grouped columns
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar)

# Parse dates
# Default label: first day of each year
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar,
dataDateFormat = "YYYY", minPeriod = "YYYY")

# Default label: first day of each month
amBarplot(x = "month", y = c("income", "expenses"), data = data_gbar,
dataDateFormat = "MM/YYYY", minPeriod = "MM")
amBarplot(x = "day", y = c("income", "expenses"), data = data_gbar,
dataDateFormat = "DD/MM/YYYY")
```
# Change groups colors
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar, 
groups_color = c("#87cefa", "#c7158"))

# Regular stacked bars
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar, stack_type = "regular")

# 100% stacked bars
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar, stack_type = "100")

# Layered bars
amBarplot(x = "year", y = c("income", "expenses"), data = data_gbar, layered = TRUE)

# Data with row names
dataset <- data.frame(get(x = "USArrests", pos = "package:datasets"))
amBarplot(y = c("Murder", "Assault", "UrbanPop", "Rape"), data = dataset, stack_type = "regular")

# Round values
amBarplot(x = "year", y = c("in", "ex"), data = data_gbar, precision = 0)

---

**amBoxplot**

Plotting boxplot using rAmCharts

**Description**

amBoxplot computes a boxplot of the given data values. Can be a vector, a data.frame, or a matrix.

**Usage**

amBoxplot(object, ...)

## Default S3 method:
amBoxplot(
  object,
  xlab = NULL,
  ylab = NULL,
  ylim = NULL,
  names = NULL,
  col = "#1e90ff",
  horiz = FALSE,
  ...
)

## S3 method for class 'data.frame'
amBoxplot(
  object,
  ...
id = NULL,
xlab = NULL,
ylab = NULL,
ylim = NULL,
col = NULL,
horiz = FALSE,
...
)

## S3 method for class 'matrix'
amBoxplot(
  object,
  use.cols = TRUE,
  xlab = NULL,
  ylab = NULL,
  ylim = NULL,
  col = NULL,
  horiz = FALSE,
  ...
)

## S3 method for class 'formula'
amBoxplot(
  object,
  data = NULL,
  id = NULL,
  xlab = NULL,
  ylab = NULL,
  ylim = NULL,
  col = NULL,
  horiz = FALSE,
  ...
)

Arguments

object a vector, data.frame, a matrix, or a formula.
...
see amOptions for more options.
xlab, ylab character, labels of the axis.
ylim numeric, y values range with sensible defaults.
names character, name on x-axis, if object is a vector.
col character, color(s) to be used to fill the boxplot.
horiz logical, TRUE to rotate chart.
id character, column name of id to identify outliers, if object is a dataframe.
use.cols logical, for matrix only. Set to TRUE to display boxplot based on columns.
data data.frame, from which the variables in formula should be taken.
Value
An object of class AmChart.

References
See online documentation `https://datastorm-open.github.io/introduction_ramcharts/` and `amChartsAPI`

See Also
amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

```r
## Not run:
# 'numeric' (default)
amBoxplot(rnorm(100))

# 'formula'
amBoxplot(count ~ spray, data = InsectSprays)

# 'formula', two group
data <- InsectSprays
data$group <- c("H", "F")
amBoxplot(count ~ spray + group, data = data, col = c("purple", "darkblue"))

# 'matrix'
x <- matrix(nrow = 10, ncol = 5, rnorm(50))
amBoxplot(x)

# 'data.frame'
amBoxplot(iris[, 1:4])

## End(Not run)
# Other examples available which can be time consuming depending on your configuration.

don <- data.frame(a = 1:10, b = 1:5)
amBoxplot(don, ylim = c(0,15))

# --- matrix
x <- matrix(nrow = 10, ncol = 5, rnorm(50))
amBoxplot(x) # on columns
colnames(x) <- LETTERS[1:5]
amBoxplot(x) # with names
amBoxplot(x, use.cols = FALSE, col = c("blue", "red"))
```
### amBullet

**Plotting bullet chart using rAmCharts**

### Description

amBullet computes a bullet chart of the given value.

### Usage

```r
amBullet(
  value,
  min = 0,
  max = 100,
  val_color = "#000000",
  limit = 85,
  limit_color = "#000000",
  steps = TRUE,
  label = "",
  horiz = TRUE,
  rates,
  ...
)
```
Arguments

value        numeric, value to display.
min          numeric, minimum value allowed.
max          numeric, maximum value allowed.
val_color    character, color of the bar value, in hexadecimal.
limit        numeric, target value.
limit_color  character, color of the target line.
steps        logical, default set to TRUE.
label        character, label of the bullet.
horiz        logical, TRUE (default) for an horizontal bullet chart, FALSE for a vertical one.
rates        data.frame with 4 columns: name (character), min (numeric), max (numeric), and color (character, color in hexadecimal).
...           see amOptions for more options.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

amBullet(value = 65)

# Other examples available which can be time consuming depending on your configuration.

# Remove steps for background
amBullet(value = 65, steps = FALSE)

# Tune the colors with name or HTML code
amBullet(value = 65, val_color = "purple", limit_color = "#3c8dbc")

# Change the orientation
amBullet(value = 65, steps = FALSE, horiz = FALSE)

# Add text
amBullet(value = 65, label = "Evaluation")

# Change min and max values
amBullet(value = 65, min = 20, max = 90)
amCandlestick

Plotting candlestick chart using rAmCharts

Description

amCandlestick computes a candlestick chart of the given value.

Usage

amCandlestick(
  data,
  xlab = "", ylab = "", horiz = FALSE,
  positiveColor = "#7f8da9",
  negativeColor = "#db4c3c",
  names = c("low", "open", "close", "high"),
  dataDateFormat = NULL,
  minPeriod = ifelse(!is.null(dataDateFormat), "DD", ""),
  ...
)

Arguments

data data.frame, dataframe with at least 5 columns: category, open (numeric), close (numeric), low (numeric), high (numeric). See data_candleStick1 and data_candleStick2.
xlab character, label for x-axis.
ylab character, label for y-axis.
horiz logical, TRUE for an horizontal chart, FALSE for a vertical one.
positiveColor character, color for positive values (in hexadecimal).
negativeColor character, color for negative values (in hexadecimal).
names character, names for the tooltip. Default set to c("low", "open", "close", "high").
dataDateFormat character, default set to NULL. Even if your chart parses dates, you can pass them as strings in your dataframe - all you need to do is to set data date format and the chart will parse dates to date objects. Check this page for available formats. Please note that two-digit years (YY) as well as literal month names (MMM) are NOT supported in this setting.
amCandlestick

minPeriod character, minPeriod Specifies the shortest period of your data. This should be set only if dataDateFormat is not NULL. Possible period values: fff - milliseconds, ss - seconds, mm - minutes, hh - hours, DD - days, MM - months, YYYY - years. It's also possible to supply a number for increments, i.e. '15mm' which will instruct the chart that your data is supplied in 15 minute increments.

... see amOptions for more options.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

data("data_candleStick2")
amCandlestick(data = data_candleStick2)

# Change colors
amCandlestick(data = data_candleStick2, positiveColor = "black", negativeColor = "green")

# Naming the axes
amCandlestick(data = data_candleStick2, xlab = "categories", ylab = "values")

# Rotate the labels for x axis
amCandlestick(data = data_candleStick2, labelRotation = 90)

# Change names
amCandlestick(data = data_candleStick2, names = c("min", "begin", "end", "max"))

# Horizontal chart :
amCandlestick(data = data_candleStick2, horiz = TRUE)

# Parse date
amCandlestick(data = data_candleStick2, dataDateFormat = "YYYY-MM-DD")

# Datass over months
amCandlestick(data = data_candleStick2, dataDateFormat = "YYYY-MM-DD", minPeriod = "MM")

# Decimal precision
if (requireNamespace("pipeR", quietly = TRUE)) {
  require(pipeR)
  amCandlestick(data = data_candleStick2, horiz = TRUE)
  setProperties(precision = 2)
}

---

### Description

Defines the AmChart properties.

### Details

API for plotting AmChart with R.

### Slots

- **allLabels** list of **Label**. Example of a label object, with all possible properties: `label(x = 20, y = 20, text = "this is a label", align = "left", size = 12, color = "#CC0000", alpha = 1, rotation = 0, bold = TRUE, url = "http://www.amcharts.com"). Run api("Label") for more informations.

- **arrows** list of **GaugeArrow**. Only valid for gauge charts. Run api("GaugeArrow") for more informations.

- **axes** list of **GaugeAxis** properties. Only valid for gauge charts. Run api("GaugeAxis") for more informations.

- **balloon** **AmBalloon**. Creates the balloons (tooltips) of the chart. It follows the mouse cursor when you roll-over the data items. The framework generates the instances automatically you just have to adjust the appearance to your needs. Run api("AmBalloon") for more informations.

- **categoryAxis** **CategoryAxis**. Read-only. Chart creates category axis itself. If you want to change some properties, you should get this axis from the chart and set properties to this object. Run api("CategoryAxis") for more informations.

- **categoryField** character. Category field name indicates the name of the field in your dataProvider object which will be used for category axis values.

- **ChartCursor** **ChartCursor**. Chart's cursor. Run api("ChartCursor") for more informations.

- **ChartScrollbar** **ChartScrollbar**. Chart's scrollbar. Run api("ChartScrollbar") for more informations.

- **creditsPosition** character, specifies position of the amCharts' website link. Allowed values are: "top-left", "top-right", "bottom-left" and "bottom-right".

- **dataProvider** data.frame, containing the data.
graphs list of **AmGraph**. Creates the visualization of the data in following types: line, column, step line, smoothed line, olhc and candlestick. Run `api("AmGraph")` for more informations.

graph **AmGraph**. Only valid for Gantt charts. Gantt chart actually creates multiple graphs (separate for each segment). Properties of this graph are passed to each of the created graphs - this allows you to control the look of segments. Run `api("AmGraph")` for more informations.

guides list of **Guide**. Instead of adding guides to the axes, you can push all of them to this array. In case guide has category or date defined, it will automatically be assigned to the category axis, otherwise to the first value axis, unless you specify a different valueAxes for the guide. Run `api("Guide")` for more informations.

legend **AmLegend**. Legend of a chart. Run `api("AmLegend")` for more informations.

segmentsField character. Segments field in your data provider. Only valid for Gantt Charts.

subChartProperties list. Only valid for Drilldown charts.


titles list of **Title**. Run `api("Title")` for more informations.

trendLines list of **TrendLine** objects added to a chart. You can add trend lines to a chart using this list or access already existing trend lines. Run `api("TrendLine")` for more informations.

type character. Possible types are: "serial", "pie", "radar", "xy", "radar", "funnel", "gauge", "stock". See details about using argument type. (type map is in development).

valueAxes list of **ValueAxis**. Chart creates one value axis automatically, so if you need only one value axis, you don’t need to create it. Run `api("ValueAxis")` for more informations.

valueAxis **ValueAxis**. Only valid for Gantt Charts. Set it’s type to "date" if your data is date or time based. Run `api("ValueAxis")` for more informations.

valueScrollbar **ChartScrollbar**. Value scrollbar, enables scrolling value axes.

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string. Run `runShinyExamples()` for examples.

otherProperties list containing other available properties not yet implemented in the package.

value numeric.

See Also


Examples

```
# Run runShinyExamples() for examples.
```
### amChartsAPI

**View full API of amcharts.js**

#### Description

View full API of amcharts.js

#### Usage

```r
amChartsAPI(viewer = getOption("viewer"))
```

#### Arguments

- **viewer**: Set to NULL to open in a browser

#### References


#### Examples

```r
## Not run:
amChartsAPI()
amChartsAPI(NULL)
## End(Not run)
```

### amChartsOutput

**SHINY**

#### Description

Widget output function for use in Shiny

#### Usage

```r
amChartsOutput(outputId, type = NULL, width = "100\%", height = "400px")
```

#### Arguments

- **outputId**: character, output variable to read the chart from.
- **type**: character, indicating the chart type.
- **width**: character, the width of the chart container.
- **height**: character, the height of the chart container.
amFloatingBar - Plotting floating bar chart using rAmCharts

Description

amFloatingBar computes a floating bar chart of the given values.

Usage

amFloatingBar(
  x,
  y_inf,
  y_sup,
  data,
  xlab = "",
  ylab = "",
  groups_color = NULL,
  horiz = FALSE,
  show_values = FALSE,
  depth = 0,
  dataDateFormat = NULL,
  minPeriod = ifelse(!is.null(dataDateFormat), "DD", ""),
  ...
)

Arguments

x character, column name for x-axis or numeric value of the corresponding column. It is optional if argument data has row names.

y_inf character, column name for the lower value or numeric vector of the corresponding column.

y_sup character, column name for the upper value or numeric vector of the corresponding column.

data data.frame, dataframe with values to display. You can add a column "color" (character, colors in hexadecimal). You can also add a column "description" (character) containing the text you want to display when mouse is on the graphic ('<br>' for a new line). See data_fbar.

xlab character, label for x-axis.

ylab character, label for y-axis.

groups_color character, vector of colors in hexadecimal, same length as y_inf or y_sup.

horiz logical, TRUE for an horizontal chart, FALSE for a vertical one. If 'horiz' is set to TRUE, the setting 'labelRotation' will be ignored.

show_values logical, TRUE to display values.

depth numeric, if > 0, chart is displayed in 3D. Value between 0 and 100.
dataDateFormat character, default set to NULL. Even if your chart parses dates, you can pass them as strings in your dataframe - all you need to do is to set data date format and the chart will parse dates to date objects. Check this page for available formats. Please note that two-digit years (YY) as well as literal month names (MMM) are NOT supported in this setting.

minPeriod Specifies the shortest period of your data. This should be set only if dataDateFormat is not 'NULL'. Possible period values: fff - milliseconds, ss - seconds, mm - minutes, hh - hours, DD - days, MM - months, YYYY - years. It's also possible to supply a number for increments, i.e. '15mm' which will instruct the chart that your data is supplied in 15 minute increments.

... see amOptions for more options.

Details

**Notice about labels:** if the chart has many columns, several labels might be hidden. It depends on the width of the container where the chart is displayed. Zoom on the chart to see if the chart can contain all labels. You can also add a cursor to your chart...

Value

An object of class AmChart.

References

See online documentation [https://datastorm-open.github.io/introduction_ramcharts/](https://datastorm-open.github.io/introduction_ramcharts/) and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

```r
# Load data
data(data_fbar)
data(data_gbar)

amFloatingBar(x = "country", y_inf = "visits_inf", y_sup = "visits_sup",
data = data_fbar, labelRotation = -45)

amFloatingBar(x = "year", y_inf = "expenses", y_sup = "income", data = data_gbar,
dataDateFormat = "YYYY", minPeriod = "YYYY", zoom = TRUE)

# Other examples available which can be time consuming depending on your configuration.
library(pipeR)

# Reference example : column chart
```
amFunnel

Plotting funnel chart using rAmCharts

Description

amFunnel computes a funnel chart of the given value.
Usage

amFunnel(
  data,  
  inverse = FALSE, 
  neck_height = NULL, 
  neck_width = NULL, 
  depth = 0, 
  label_side = "right", 
  margin_right = 200, 
  margin_left = 200, 
  ... 
)

Arguments

data frame of at least 2 columns: value (numeric, positive), and description (character). You can add a third column "color" (character, colors in hexadecimal) see data_funnel.

inverse logical, if TRUE, the funnel chart will be inversed.

neck_height numeric, value between 0 and 100: if a bottleneck is desired, this value determines its height. Default to NULL.

neck_width numeric, value between 0 and 100: if a bottleneck is desired, this value determines its width. Default to NULL.

depth numeric, if > 0, chart is displayed in 3D, only for pyramid chart (without a bottleneck). Value between 0 and 100.

label_side character, label position: "right" or "left".

margin_right numeric, margin at the right side.

margin_left numeric, margin at the left side.

... see amOptions for more options.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

data(data_funnel)
amFunnel(data = data_funnel, inverse = TRUE)
Other examples available which can be time consuming depending on your configuration.

Change the orientation and legend side

```r
amFunnel(data = data_funnel, inverse = FALSE,
         label_side = "left", margin_right = 15, margin_left = 160)
```

Basic example: Funnel chart

```r
amFunnel(data = data_funnel, neck_height = 30, neck_width = 40)
```

3D pyramid

```r
amFunnel(data = data_funnel, depth = 50, inverse = TRUE)
```

---

**AmGraph-class**

**AmGraph class**

### Description

Creates the visualization of the data in following types: line, column, step line, smoothed line, ohlc and candlestick.

### Details

Run `api("AmGraph")` for more details and all available properties.

### Slots

- **balloonText** character. Balloon text. You can use tags like `[[value]]`, `[[description]]`, `[[percent]]`, `[[open]]`, `[[category]]` or any other field name from your data provider. HTML tags can also be used.

- **title** character. Graph title.

- **type** character. Type of the graph. Possible values are: "line", "column", "step", "smoothedLine", "candlestick", "ohlc". XY and Radar charts can only display "line" other graphs.

- **valueField** character. Name of the value field in your dataProvider.

- **listeners** "list" containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string. See examples for details.

- **otherProperties** "list" containing other available properties not yet implemented in the package.

- **value** numeric.
amHist  

Plotting histogram

Description

amHist computes a histogram of the given data values.

Usage

amHist(x, ...)

## S3 method for class 'numeric'
amHist(
  x,
  col = "#1e90ff",
  border = "#1e90ff",
  freq = TRUE,
  plot = TRUE,
  labels = FALSE,
  xlab, ylab,
  ylim,
  control_hist,
  ...
)

Arguments

x  numeric, a vector of values for which the histogram is desired.
...
see amOptions for more options.
col  character, a color to be used to fill the bars.
border  character, a color for the borders.
freq  logical, if TRUE, the histogram graphic is a representation of frequencies, the counts component of the result; if FALSE, probability densities, component density, are plotted (so that the histogram has a total area of one). Defaults to TRUE if and only if breaks are equidistant (and probability is not specified).
plot  logical, if TRUE (default), an histogram is plotted, otherwise a list of breaks and counts is returned. In the second case, a warning is used if (typically graphical) arguments are specified that only apply to the plot = TRUE case.
labels  logical, set to TRUE to display labels. Default set to FALSE. Additionally draw labels on top of bars. if TRUE, draw the counts or rounded densities; if labels is a character, draw itself.
xlab, ylab  character, labels of the axis.
ylim  numeric, the range of y values with sensible defaults.
control_hist  (optional) named list() containing parameters to compute the histogram.
Value

An object of class AmChart.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

amHist(x = rnorm(100))

# Other examples available which can be time consuming depending on your configuration.

x <- replicate(1000, {
  if (round(runif(1))) {
    rnorm(1)
  } else {
    rnorm(1, mean = 5)
  }
})

# Without plot
amHist(x = x, plot = FALSE)

# With options
amHist(x = x, border = "blue")
amHist(x = x, col = "lightblue", control_hist = list(breaks = 100))
amHist(x = x, col = "grey")
amHist(x = x, col = "gray")
amHist(x = x, main = "Histogram", ylab = "y-axis", xlab = "x-axis", col = "red")
amHist(x = x, main = "Histogram", ylab = "y-axis", xlab = "x-axis", ylim = c(10, 15))
amHist(x = x, main = "Histogram", ylab = "y-axis", xlab = "x-axis")

# Options for computing the histogram
amHist(x = x, control_hist = list(breaks = "Scott"))
AmLegend-class  

Description
Creates the legend for the chart, automatically adapts the color settings of the graphs.

Details
Run `api("AmLegend")` for more information and all available properties.

Slots
- `useGraphSettings` logical. If TRUE, border color will be changed when user rolls-over the slice, graph, etc, instead of background color.
- `listeners` list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.
- `otherProperties` list containing other available properties not yet implemented in the package.
- `value` numeric.

Author(s)
datastorm-open

---

amLines

Description
amLines adds a serie to an existing serial chart.

Usage
```r
amLines(
  chart,
  x = NULL,
  y = NULL,
  type = c("points", "line", "smoothedLine", "both", "step"),
  col = "#0066cc",
  title,
  fill_alphas = 0,
  balloon = T
)
```
**amMekko**

**Arguments**

- **chart** \(\text{AmChart}.\) Chart you wish to add the new serie.
- **x** numeric, equivalent to y, deprecated.
- **y** numeric.
- **type** (optionnal) character. Possible values are: "l" for line, "p" for points, "b" for both, "sl" for smoothed line, "st" for step
- **col** character, color of the new serie.
- **title** character, name of the new serie, used when legend is enabled.
- **fill_alphas** a numeric between 0 and 1 for printed area.
- **balloon** logical, add balloon with value or not

**Note**

It is supposed here that x or y corresponds to the y-axis, and the x-axis is automatically linked to the x values of the chart "chart". That is why it makes sense to give the y argument.

**Examples**

```r
if (requireNamespace("pipeR", quietly = TRUE)) {
  require(pipeR)
  amPlot(x = rnorm(100), type = "sl") %>>%
    amLines(x = rnorm(100), type = "p")
}

amPlot(x = rnorm(100), type = "sl")
amLines(x = rnorm(100), col = "blue")
amLines(x = rnorm(100), type = "sl")
amLines(x = rnorm(100), type = "p")

# For an XY chart
x <- sort(rnorm(100))
y1 <- rnorm(100, sd = 10)
y2 <- rnorm(100, sd = 10)
y3 <- rnorm(100, sd = 10)
amPlot(x = x, y = y1)
amLines(x = y2, col = "blue")
amLines(x = y3, type = "p")
```

---

**amMekko**

*Plotting mekko chart (quali vs quali)*

**Description**

amMekko computes a mekko chart of the given values.
Usage

```r
amMekko(
  x,
  y,
  data,
  xlab = "",
  ylab = "",
  groups_color = NULL,
  horiz = FALSE,
  show_values = FALSE,
  ...,
)
```

Arguments

- **x**: character, column name for x-axis.
- **y**: character, column name for y-axis.
- **data**: data.frame, dataframe with values to display. See `data_mekko`
- **xlab**: character, label for x-axis.
- **ylab**: character, label for y-axis.
- **groups_color**: character vector of colors in hexadecimal, same length as the number of y modalities.
- **horiz**: logical, TRUE for an horizontal chart, FALSE for a vertical one.
- **show_values**: logical, TRUE to display values.
- **...**: see `amOptions` for more options.

References

See online documentation [https://datastorm-open.github.io/introduction_ramcharts/](https://datastorm-open.github.io/introduction_ramcharts/) and `amChartsAPI`

See Also

- `amOptions`, `amBarplot`, `amBoxplot`, `amHist`, `amPie`, `amPlot`, `amTimeSeries`, `amStockMultiSet`, `amBullet`, `amRadar`, `amWind`, `amFunnel`, `amAngularGauge`, `amSolidGauge`, `amMekko`, `amCandlestick`, `amFloatingBar`, `amOHLC`, `amWaterfall`

Examples

```r
data(data_mekko)
amMekko(x = "var1", y = "var2", data = data_mekko)
```

# Other examples available which can be time consuming depending on your configuration.
library(pipeR)

# Horizontal
AmObject-class

\begin{verbatim}
  amMekko(x = "var1", y = "var2", data = data_mekko, horiz = TRUE)

  # Display values
  amMekko(x = "var1", y = "var2", data = data_mekko, show_values = TRUE)
\end{verbatim}

AmObject-class  AmObject class

**Description**

This is a virtual class for representing any Am** class

**Slots**

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.

otherProperties list containing other available properties not yet implemented in the package.

value numeric

**Author(s)**

datastorm-open

---

amOHLC  Plotting OHLC chart

**Description**

amOHLC computes an OHLC chart of the given value.

**Usage**

\begin{verbatim}
  amOHLC(
    data,
    xlab = "",
    ylab = "",
    horiz = FALSE,
    zoom = TRUE,
    positiveColor = "#7f8da9",
    negativeColor = "#db4c3c",
    names = c("low", "open", "close", "high"),
    dataDateFormat = NULL,
    minPeriod = ifelse(!is.null(dataDateFormat), "DD", ""),
    ...
  )
\end{verbatim}
Arguments

- **data** (data.frame): dataframe with at least 5 columns: category, open (numeric), close (numeric), low (numeric), high (numeric).
- **xlab** (character): label for x-axis.
- **ylab** (character): label for y-axis.
- **horiz** (logical): TRUE for an horizontal chart, FALSE for a vertical one.
- **zoom** (logical): default set to TRUE: a cursor is added to the chart.
- **positiveColor** (character): color for positive values (in hexadecimal).
- **negativeColor** (character): color for negative values (in hexadecimal).
- **names** (character): names for the tooltip. Default to c("low", "open", "close", "high").
- **dataDateFormat** (character): default set to NULL. Even if your chart parses dates, you can pass them as strings in your dataframe - all you need to do is to set data date format and the chart will parse dates to date objects. Check this page for available formats. Please note that two-digit years (YY) as well as literal month names (MMM) are NOT supported in this setting.
- **minPeriod** (character): minPeriod Specifies the shortest period of your data. This should be set only if dataDateFormat is not 'NULL'. Possible period values: fff - milliseconds, ss - seconds, mm - minutes, hh - hours, DD - days, MM - months, YYYY - years. It’s also possible to supply a number for increments, i.e. '15mm' which will instruct the chart that your data is supplied in 15 minute increments.

... see `amOptions` for more options.

References

See online documentation [https://datastorm-open.github.io/introduction_ramcharts/](https://datastorm-open.github.io/introduction_ramcharts/) and `amChartsAPI`

See Also

`amOptions`, `amBarplot`, `amBoxplot`, `amHist`, `amPie`, `amPlot`, `amTimeSeries`, `amStockMultiSet`, `amBullet`, `amRadar`, `amWind`, `amFunnel`, `amAngularGauge`, `amSolidGauge`, `amMekko`, `amCandlestick`, `amFloatingBar`, `amOHLC`, `amWaterfall`

Examples

```r
data("data_candleStick2")
amOHLC(data = data_candleStick2)
```

# Other examples available which can be time consuming depending on your configuration.
if (requireNamespace("pipeR", quietly = TRUE)) {
  require(pipeR)
}

# Change colors
amOHLC(data = data_candleStick2, positiveColor = "green", negativeColor = "red")
\# Naming the axes
amOHLC(data = data_candleStick2, xlab = "categories", ylab = "values")

\# Rotate the labels for x axis
amOHLC(data = data_candleStick2, labelRotation = 90)

\# Change names
amOHLC(data = data_candleStick2, names = c("min", "begin", "end", "max"))

\# Use amOptions
amOHLC(data = data_candleStick2, zoom = FALSE)

---

### Description

amOptions sets the most common options for chart customization. You can set other properties with the method `setProperties`. See details for exception.

### Usage

```r
amOptions(
  chart,
  theme = c(\"none\", \"light\", \"dark\", \"patterns\", \"chalk\"),
  legend = FALSE,
  legendPosition = \"right\",
  legendAlign = \"left\",
  export = FALSE,
  exportFormat = character(),
  creditsPosition = \"top-left\",
  main = character(),
  mainColor = \"#000000\",
  mainSize = 15,
  zoom = FALSE,
  scrollbar = FALSE,
  scrollbarHeight = 20,
  valuescrollbar = FALSE,
  valuescrollbarHeight = 20,
  labelRotation = 0,
  ...
)
```
Arguments

- **chart**: `AmChart`
- **theme**: character, possible values are: "none", "light", "dark", "patterns", "chalk", default set to "none".
- **legend**: logical, default FALSE. TRUE to add a legend to the chart.
- **legendPosition**: character, possible values are: "left", "right", "top" or "bottom", default set to "right".
- **legendAlign**: character, controls the legend alignment. Possible values are: "left", "right" or "center", default set to "left". Only used if legend = TRUE.
- **export**: logical, default set to FALSE. TRUE to display export feature.
- **exportFormat**: character, desired export format. Possible values are: "JPG", "PNG", "SVG", "CSV", "JSON", "PDF", "XLSX", "PRINT".
- **creditsPosition**: character, controls credits position. Possible values are: "top-left", "top-right", "bottom-left" or "bottom-right", default set to "top-left".
- **main**: character, chart's title.
- **mainColor**: character, main color (in hexadecimal), default set to "#000000".
- **mainSize**: numeric, main size, default set to 15.
- **zoom**: logical, TRUE to add a chart cursor, default set to FALSE.
- **scrollbar**: logical, default FALSE. TRUE to display scrollbar.
- **scrollbarHeight**: numeric, height in pixels, must be > 0.
- **valuescrollbar**: logical, default FALSE. TRUE to display valuescrollbar.
- **valuescrollbarHeight**: numeric, height in pixels, must be > 0.
- **labelRotation**: numeric, rotation angle of a label. Only horizontal axis' values can be rotated. Value must be between -90 and 90.

... Other properties added to the chart using setProperties.

Details

**Exception:**

- It's not possible to export a gauge chart data as CSV.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

- amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall
Examples

```r
library(pipeR)
data(data_pie)

# Export
amPie(data = data_pie)
amOptions(export = TRUE)

# Legend
amPie(data = data_pie)
amOptions(legend = TRUE)

# Legend position
amPie(data = data_pie)
amOptions(legend = TRUE, legendPosition = "bottom")

# Credits position
amPie(data = data_pie)
amOptions(creditsPosition = "bottom-right")

# Theme
amPie(data = data_pie)
amOptions(theme = "chalk")

# Title
amPie(data = data_pie)
amOptions(main = "Social network", mainColor = "#FFFFFF", mainSize = 40, theme = "chalk")

# Custom exemple
amPie(data = data_pie)
amOptions(main = "Social network", mainColor = "#FFFFFF", mainSize = 40,
theme = "dark", legend = TRUE, legendPosition = "bottom",
creditsPosition = "bottom-right")
```

---

`amPie`  
*Plotting pie chart*

**Description**

`amPie` computes a pie chart of the given value.

**Usage**

`amPie(data, show_values = TRUE, depth = 0, inner_radius = 0, ...)`
amPlot

Plot serial data

Description

amPlot computes a plot of the given data values (can be a vector, dataframe or formula).

Arguments

data frame, dataframe with at least 2 columns: label (character), value (numeric). See data_pie You can add a third column "color" (character, colors in hexadecimal).
show_values logical, TRUE to display values.
depth numeric, if > 0, chart is displayed in 3D, value between 0 and 100
inner_radius numeric, value between 0 and 100
...
see amOptions for more options.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

data("data_pie")
amPie(data = data_pie)

# Other examples available which can be time consuming depending on your configuration.

# Don't display values
amPie(data = data_pie, show_values = FALSE)

# 3D pie
amPie(data = data_pie, depth = 10)

# Donut chart
amPie(data = data_pie, inner_radius = 50)

# All parameters
amPie(data = data_pie, inner_radius = 50, depth = 10, show_values = FALSE)
Usage

amPlot(x, ...)

## Default S3 method:
amPlot(x, ...)

## S3 method for class 'numeric'
amPlot(
x,
y,
in bullet = c("round", "diamond", "square", "bubble", "yError", "xError", "triangleLeft", "triangleRight", "triangleUp", "triangleDown"),
type = c("points", "line", "smoothedLine", "step", "both"),
col = "#0066cc",
fill_alphas = 0,
weights = NULL,
precision = 2,
title = NULL,
id,
error,
xlab,
ylab,
lty,
cex,
lwd,
xlim,
ylim,
...
)

## S3 method for class 'character'
amPlot(
x,
y,
in bullet = c("round", "diamond", "square", "bubble", "yError", "xError", "triangleLeft", "triangleRight", "triangleUp", "triangleDown"),
type = c("points", "line", "smoothedLine", "step", "both"),
col = "#0066cc",
fill_alphas = 0,
weights = NULL,
precision = 2,
parsedates = FALSE,
title = NULL,
dataDateformat,
id,
error,
xlab,
ylab,
### Arguments

**x**
- the coordinates of points in the plot: numeric, data.frame, or formula.

**y**
- numeric, the y coordinates of points in the plot, optional if x is an appropriate structure.

**bullet**
- character, point shape. Possible values are: "diamond", "square", "bubble", "yError", "xError", "round", "triangleLeft", "triangleRight", "triangleUp", "triangleDown". Default set to "round".

**type**
- character, type of plot. Possible values are: "l" for a line, "sl" for a smoothed line, "st" for step, "p" for points, and "b" for line and points. Default set to "p".

---

**amPlot**

```r
amPlot(
  x,
  y,
  bullet = "round",
  type = "p",
  col = "gray",
  weights = NULL,
  precision = 2,
  parseDates = FALSE,
  dataDateFormat = NULL,
  id,
  error,
  xlab,
  ylab,
  lty,
  cex,
  lwd,
  xlim,
  ylim,
  ...
)
```

## S3 method for class 'factor'

```r
amPlot(
  x,
  y,
  bullet = "round",
  type = "p",
  col = "gray",
  weights = NULL,
  precision = 2,
  parseDates = FALSE,
  dataDateFormat = NULL,
  id,
  error,
  xlab,
  ylab,
  lty,
  cex,
  lwd,
  xlim,
  ylim,
  ...
)
```

## S3 method for class 'data.frame'

```r
amPlot(x, columns, type = "l", precision = 2, xlab, ylab, fill_alphas = 0, ...)
```

## S3 method for class 'formula'

```r
amPlot(x, data, type = "p", fill_alphas = 0, xlab, ylab, main = "", ...)
```
col either a factor or a character, default set to "gray".
fill_alphas a numeric between 0 and 1 for printed area.
weights numeric, weights for x/y charts only. Small values are preferred for visibility.
precision numeric, precision you wish to display. Default set to 2.
title character, name of the new serie, used when legend is enabled.
id numeric, point id, for x/y charts only. Default 1:length(x).
error numeric, only when type is "xError" "yError" default NULL,
xlab character, label for x-axis.
ylab character, label for y-axis.
lty numeric, line type (dashes).
cex numeric, bullet size.
lwd numeric, line width
xlim numeric, x range.
ylim numeric, y range.
parseDates logical, default set to FALSE, if TRUE argument dataDateFormat has to be provided.
dataDateFormat character, default set to NULL. Even if your chart parses dates, you can pass them as strings in your dataframe - all you need to do is to set data date format and the chart will parse dates to date objects. Check this page for available formats. Please note that two-digit years (YY) as well as literal month names (MMM) are NOT supported in this setting.
columns (optional) either a vector of character containing the names of the series to draw, or a numeric vector of indices. By default all numeric columns will be drawn.
data dataset
main title

Value
Return an Amchart.

References
See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also
amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall
Examples

```r
## Not run:
#
# 'numeric':
amPlot(x = rnorm(100))

# add line
chart <- amPlot(x = rnorm(100), type = 'sl', legend = T)
amLines(chart, x = rnorm(100), type = "p")

# 'character':
start <- as.POSIXct('2015-01-01')
end <- as.POSIXct('2015-12-31')
date <- seq.POSIXt(from = start, to = end, by = 'day')
date <- format(date)
y <- rnorm(length(date))
amPlot(x = date, y = y, type = 'l', parseDates = TRUE, dataDateFormat = "YYYY-DD-MM")
# notice that by default 'parseDates = FALSE'

# 'data.frame'
amPlot(iris, col = colnames(iris)[1:2], type = c("l", "st"), zoom = TRUE, legend = TRUE)

# 'formula':
amPlot(Petal.Length + Sepal.Length ~ Sepal.Width, data = iris, legend = TRUE, zoom = TRUE)

## End(Not run)
```

# Other examples available which can be time consuming depending on your configuration.
library(data.table)
iris <- as.data.table(get("iris", "package:datasets"))
x <- rnorm(100)

# Simple scatter plot with title and color
# Also change type (set to "p" by default), available "l", "sl", "st", "p", "b"
amPlot(x = x, main = "Title", col = "lightblue", type = "b")
x <- sort(rnorm(100))
y <- runif(100)
weights <- runif(100, 0, 15)
amPlot(x = x, y = y, weights = weights)
```

---

**amRadar**

**Plotting radar**

**Description**

radar computes a radarplot of the given data values.
Usage

```r
amRadar(
  data,
  col = NULL,
  backTransparency = 0.5,
  type = "polygons",
  pch = "round",
  xlim = NULL,
  ...
)
```

Arguments

- **data** data.frame first column is named "label" (character), other columns are series of values, see `data_radar`.
- **col** character, color(s) of serie(s) hexadecimal like ":00FF00".
- **backTransparency** numeric, background transparency, between 0 and 1.
- **type** character, type of radar. Possible values are : "polygons" or "circle".
- **pch** character, points symbols. Possible values are : "round", "square", "triangleUp", "triangleDown", "triangleLeft", "triangleRight", "bubble", "diamond", "xError", "yError".
- **xlim** numeric, x range.
- **...** see `amOptions` for more options.

References

See online documentation [https://datastorm-open.github.io/introduction_ramcharts/](https://datastorm-open.github.io/introduction_ramcharts/) and `amChartsAPI`

See Also

- `amOptions`, `amBarplot`, `amBoxplot`, `amHist`, `amPie`, `amPlot`, `amTimeSeries`, `amStockMultiSet`, `amBullet`, `amRadar`, `amWind`, `amFunnel`, `amAngularGauge`, `amSolidGauge`, `amMekko`, `amCandlestick`, `amFloatingBar`, `amOHLC`, `amWaterfall`

Examples

```r
data("data_radar")
amRadar(data_radar)
```

# Other examples available which can be time consuming depending on your configuration.

```r
if (requireNamespace("pipeR", quietly = TRUE)) {
  require(pipeR)
```
# Change color
amRadar(data_rad, col = "#FF0000")
amRadar(data_rad, col = c("#0000FF", "#00FF00", "#FF0000"))

# Change backTransparency
amRadar(data_rad, backTransparency = 0.6)
amRadar(data_rad, backTransparency = c(0, 0.4, 0.6))

# Change type
amRadar(data_rad, type = "circles")

# Change pch
amRadar(data_rad, pch = "triangleRight")
amRadar(data_rad, pch = "triangleLeft")

# Min-Max
amRadar(data_rad, xlim = c(0, 8))
}

---

```r
amSolidGauge

Plotting solid gauge using rAmCharts
```

### Description

`amSolidGauge` computes a gauge of the given value.

### Usage

```r
amSolidGauge(
x,
min = 0,
max = 100,
type = "full",
width = 20,
color = "",
text = "",
textSize = 20,
...)
```

### Arguments

- `x`: numeric, value for which the angular gauge is desired.
- `min`: numeric, minimal possible value.
amSolidGauge

max numeric, maximal possible value.
type character, type of gauge: "full" or "semi".
width numeric, width of the gauge.
color character, hexadecimal color value or a vector of colors.
text character, text.
textSize numeric, text size.
...
see amOptions for more options.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

amSolidGauge(x = 65)

# Other examples available which can be time consuming depending on your configuration.
if (requireNamespace("pipeR", quietly = TRUE)) {
  require(pipeR)

  # Change min and max values
  amSolidGauge(x = 65, min = 0, max = 200)

  # Semi solid gauge
  amSolidGauge(x = 65, type = "semi")

  # Change width
  amSolidGauge(x = 65, width = 50)

  # Change color
  amSolidGauge(x = 65, color = "#2F4F4F")

  # Put a color scale
  amSolidGauge(x = 10, color = c("#00ff00", "#ff700", "#ff0000"))
  amSolidGauge(x = 35, color = c("#00ff00", "#ff700", "#ff0000"))
  amSolidGauge(x = 70, color = c("#00ff00", "#ff700", "#ff0000"))
  amSolidGauge(x = 90, color = c("#00ff00", "#ff700", "#ff0000"))

  # Add some text to the printed value
  amSolidGauge(x = 65, text = "Pct")
AmStockChart-class

# Modify textSize value
amSolidGauge(x = 65, text = "Pct", textSize = 50)
}

AmStockChart-class

AmStockChart

Description

Class to draw stock charts

Slots

balloon AmBalloon.

comparedDataSets list of DataSet. Properties of data sets selected for comparison.

dataSets list of DataSet. Each element must be a list of Dataset properties.

dataSetSelector list of DataSetSelector. You can add it if you have more than one data set and want users to be able to select/compare them.

mainDataSet DataSet. Data set selected as main.

panels list of StockPanel.

periodSelector PeriodSelector. You can add it if you want users to be able to enter date ranges or zoom chart with predefined period buttons.

theme character

type equals "stock"

group character for synchronization

is_ts_module logicalOrMissing. Don't use. For rAmChartsTimeSeriesUI

listeners list containing the listeners to add to the chart. The list must be named as in the official API. Each element must be a character string.

otherProperties list containing other available properties not yet implemented in the package.

value numeric.

Author(s)

datastorm-open

See Also

http://docs.amcharts.com/3/javascriptstockchart/AmStockChart
amStockMultiSet

Plotting multi data-sets

Description

amStockMultiSet compute a stock of multi data-sets

Usage

amStockMultiSet(
  data,
  panelColumn = NULL,
  ZoomButtonPosition = "bottom",
  ZoomButton = data.frame(Unit = "MAX", multiple = 1, label = "All"),
  color = c("#2E2EFE", "#31B404", "#FF4000"),
  precision = 1,
  export = FALSE,
  percentHeightPanel = NULL,
  creditsPosition = "top-right",
  ...
)

Arguments

data list, list of data.frame (same structure) first column is date, others are values
panelColumn numeric, numeric vector, controle panel adding for selected series
ZoomButtonPosition character, zoom button position. Possible values are:"left", "right", "bottom", "top"
ZoomButton data.frame, 3 columns : Unit, times unit multiple : multiple*unit label : button's label
color character, color of data-sets (in hexadecimal).
precision numeric, digits precision
export logical, default set to FALSE. TRUE to display export feature.
percentHeightPanel numeric, vector of size panel, same length than data
creditsPosition character, credits position. Possible values are:"top-right", "top-left", "bottom-right", "bottom-left"
...
other first level attributes

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI
See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

```r
## Not run:
data(data_stock_3)
amStockMultiSet(data = data_stock_3)
amStockMultiSet(data = data_stock_3, panelColumn = c(1,2,1,1))
amStockMultiSet(data = data_stock_3, panelColumn = c(1,2,3,4))
ZoomButton <- data.frame(Unit = c("DD", "DD", "MAX"), multiple = c(1, 10 ,1),
label = c("Day","10 days", "MAX"))
ZoomButtonPosition <- "bottom"
amStockMultiSet(data = data_stock_3, panelColumn = c(1,2,1,1), ZoomButton = ZoomButton, ZoomButtonPosition = "top")
amStockMultiSet(data = data_stock_3, precision = 2)
amStockMultiSet(data = data_stock_3, panelColumn = c(1,2,1,1), percentHeightPanel = c(3,1))
## End(Not run)
```

---

`amTimeSeries`  
*Plotting times series (with or without aggregation)*

**Description**

`amTimeSeries` computes a stock chart.

**Usage**

```r
amTimeSeries(
data,
col_date,
col_series,
main = "",
ylab = "",
color = c("#2E2EFE", "#31B404", "#FF4000", "#AEB404"),
type = c("line"),
bullet = NULL,
```
bulletSize = 2, 
linetype = c(0, 5, 10, 15, 20),
linewidth = c(1, 1, 1, 1, 1),
fillAlphas = 0,
precision = 1,
connect = FALSE,
export = FALSE,
legend = TRUE,
legendPosition = "bottom",
legendHidden = FALSE,
aggregation = c("Average", "Low", "High", "Sum"),
maxSeries = 300,
groupToPeriods = c("ss", "mm", "hh", "DD", "MM", "YYYY"),
checkGroupToPeriods = TRUE,
ZoomButton = data.frame_Unit = "MAX", multiple = 1, label = "All"),
ZoomButtonPosition = "bottom",
periodFieldsSelection = FALSE,
scrollbar = TRUE,
scrollbarPosition = "bottom",
scrollbarHeight = 40,
scrollbarGraph = NULL,
cursor = TRUE,
cursorValueBalloonsEnabled = TRUE,
creditsPosition = "top-right",
group = NULL,
is_ts_module = FALSE,

Arguments

data data.frame, data of graph.
col_date character name of date column
col_series character names of series columns
main character, title.
ylab character, value axis label.
color character, color of series (in hexadecimal).

type character, Type of graph. Possible values are: "line" (default), "column", "step", "smoothedLine"

bullet character, point shape. Possible values are: "diamond", "square", "bubble", "yError", "xError", "round", "triangleLeft", "triangleRight", "triangleUp", "triangleDown"

bulletSize numeric, size of bullet.

linetype numeric, line type, 0 : solid, number : dashed length

linewidth numeric, line width.

fillAlphas numeric, fill. Between 0 (no fill) to 1.

precision numeric, default set to 1.

connect logical, default set to FALSE. Specifies whether to connect data points if data is missing.

export logical, default set to FALSE. TRUE to display export feature.

legend logical, enabled or not legend ? Default to TRUE.

legendPosition character, legend position. Possible values are: "left", "right", "bottom", "top"

legendHidden logical, hide some series on rendering ? Default to FALSE

aggregation character, aggregation type. Possible values are: "Low", "High", "Average" and "Sum"

maxSeries numeric, Maximum series shown at a time. In case there are more data points in the selection than maxSeries, the chart will group data to longer periods, for example - you have 250 days in the selection, and maxSeries is 150 - the chart will group data to weeks.

groupToPeriods character, Periods to which data will be grouped in case there are more data items in the selected period than specified in maxSeries property. Possible value are: 'ss', 'mm', 'hh', 'DD', 'MM', 'YYYY'. It's also possible to add multiple like "30mm". Or NULL to disable.

checkGroupToPeriods logical. Check groupToPeriods ? Default to TRUE then check validity between data and groupToPeriods, else use directly groupToPeriods

ZoomButton data.frame, 3 or 4 columns:
  • "Unit" : Character. Times unit. 'ss', 'mm', 'hh', 'DD', 'MM', 'YYYY'
  • "multiple" : Numeric. multiple*unit
  • "label" : Character. button's label
  • "selected" : Boolean. Optional. To set initial selection. (One TRUE, others FALSE)

ZoomButtonPosition character, zoom button position. Possible values are: "left", "right", "bottom", "top"

periodFieldsSelection boolean, using zoom button, add also two fields to select period ?

scrollbar boolean, enabled or not scrollbar ? Default to TRUE.
scrollbarPosition
character, scrollbar position. Possible values are: "left", "right", "bottom", "top"

scrollbarHeight
numeric, height of scroll bar. Default: 40.

scrollbarGraph
character, name of serie (column) to print in scrollbar. Default to NULL.

cursor
boolean, enabled or not cursor? Default to TRUE.

cursorValueBalloonsEnabled
boolean, if cursor, enabled or not balloons on cursor? Default to TRUE.

creditsPosition
character, credits position. Possible values are: "top-right", "top-left", "bottom-right", "bottom-left"

group
character, like in dygraphs, for synchronization in shiny or rmarkdown.

is_ts_module
boolean. Don't use. For rAmChartsTimeSeriesUI

dataDateFormat
character Data date format. Default to 'YYYY-MM-DD JJ:NN:ss'. See details.

categoryBalloonDateFormats
list Date format objects for chart cursor. See details.

dateFormats
list Date format objects for x-axis. See details.

... other first level attributes

Details

For dateFormats & categoryBalloonDateFormats
Available periods are: fff - millisecond, ss - second, mm - minute, hh - hour, DD - date, WW - week, MM - month, YYYY - year

Available formats:

Year. The number of Y letters represents digits in the resulting number. I.e.: YY = 05 (always two digits), YYYY = 2005

Month of the year. The output depends on the number of M’s: M = 8 (one or two digits), MM = 08 (always two digits), MMM = Aug (3-letter month abbreviation), MMMM = August (full month name)

Week of the year W

Day of the month: D = 7 (one or two digits), DD = 07 (always two digits)

Day of week: E = 2 (single digit day of week), EE = 02 (two-digit day of week), EEE = Tue (3-letter abbreviation of the literal representation of the day of week), EEEE = Tuesday (full day of week name)

Hour: 0-23: J = 3 (one or two digits), JJ = 03 (always two digits)

Hour: 1-24: H = 3 (one or two digits), HH = 03 (always two digits) K Hour in am/pm: 0-11 L Hour in am/pm: 1-12 *

Minute in hour: N = 8 (one or two digits), NN = 08 (always two digits)

Second in minute: S = 5 (one or two digits), SS = 05 (always two digits)
amTimeSeries

Milliseconds: QQ = 51, QQQ = 051
Other characters Other characters will be displayed as they are without changing them. I.e.: YYYY-MM-DD = 2013-03-01

References
See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also
rAmChartsTimeSeriesUI for shiny module, amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

data("data_stock_2")
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"))

## Not run:
# upper /lower
data <- data_stock_2[1:50, ]
data$ts1low <- data$ts1-100
data$ts1up <- data$ts1+100

amTimeSeries(data, "date", list(c("ts1low", "ts1", "ts1up"), "ts2"))
amTimeSeries(data, "date", list(c("ts1low", "ts1", "ts1up"), "ts2"),
color = c("red", "blue"), bullet = c("round", "square"))

# column / step
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"),
type = c("column", "step"), fillAlphas = c(1, 0),
linewidth = c(0, 1))

# some parameters
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round")
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round",
groupToPeriods = c('hh', 'DD', '10DD'))
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round",
groupToPeriods = c('12hh', 'DD', '10DD'),
maxSeries = 50)
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round",
groupToPeriods = c('hh', 'DD', '10DD'),
linewidth = c(3, 1))
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), aggregation = "Sum")
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round",
```r
# groupToPeriods function
groupToPeriods = c('12hh', 'DD', '10DD'),
maxSeries = 50, precision = 5)

amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = c("diamond", "square"),
linetype = 0, bulletSize = c(5, 10),
groupToPeriods = c('12hh', 'DD', '10DD'),
maxSeries = 50, aggregation = "Sum")

ZoomButton <- data.frame(Unit = c("DD", "DD", "MAX"), multiple = c(1, 2, 1),
label = c("Day", "2 days", "MAX"))
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round",
ZoomButton = ZoomButton, main = "My title", ylab = "Interest")
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round",
ZoomButton = ZoomButton, main = "My title", ylab = "Interest",
extport = TRUE, ZoomButtonPosition = "right",
legendPosition = "bottom", scrollbarPosition = "top")
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"), bullet = "round",
ZoomButton = ZoomButton, main = "My title",
ylab = "Interest", export = TRUE,
creditsPosition = "bottom-left")

# date formats
amTimeSeries(data_stock_2, "date", c("ts1", "ts2"),
type = "column", fillAlphas = 1,
linwidth = 0, legendHidden = T,
categoryBalloonDateFormats = list(list(period = 'YYYY', format = 'YYYY'),
list(period = 'MM', format = 'MM'),
list(period = 'WW', format = 'MM-DD'),
list(period = 'DD', format = 'MM-DD'),
list(period = 'hh', format = 'MM-DD JJ:NN'),
list(period = 'mm', format = 'MM-DD JJ:NN'),
list(period = 'ss', format = 'MM-DD JJ:NN:SS'),
list(period = 'fff', format = 'MM-DD JJ:NN:SS')),
dateFormats = list(list(period = 'YYYY', format = 'YYYY'),
list(period = 'MM', format = 'MM:MM'),
list(period = 'WW', format = 'MM:MM-DD'),
list(period = 'DD', format = 'MM:MM-DD'),
list(period = 'hh', format = 'MM-DD JJ:NN'),
list(period = 'mm', format = 'MM-DD JJ:NN'),
list(period = 'ss', format = 'MM-DD JJ:NN:SS'),
list(period = 'fff', format = 'MM-DD JJ:NN:SS')))```

## End(Not run)
Description

amWaterfall computes a waterfall chart of the given value.

Usage

amWaterfall(data, start = 0, horiz = FALSE, show_values = FALSE, ...)

Arguments

data data.frame, dataframe with at least 3 columns: label (character), value (numeric), operation (character: "plus", "minus", "total"). You can add a third column "color" (character, colors in hexadecimal). You can also add a column "description" (character) containing the text you want to display when mouse is on the graphic ('<br>' for a new line). See data_waterfall.

start numeric, value from which to start.

horiz logical, TRUE for an horizontal chart, FALSE for a vertical one.

show_values logical, TRUE to display values on the chart.

... see amOptions for more options.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

data("data_waterfall")
amWaterfall(data = data_waterfall, show_values = TRUE)

# Other examples available which can be time consuming depending on your configuration.

# Change the orientation :
amWaterfall(data = data_waterfall, horiz = TRUE)
amWind

Description

amWind computes a windplot of the given data values.

Usage

amWind(data, col = NULL, backTransparency = 0.5, ...)

Arguments

data data.frame, a dataframe which columns are series of values, from weakest wind (first column) to stronger wind (last column). See data_wind.
col character, color(s) of serie(s) hexadecimal like "#00FF00".
backTransparency numeric, background transparency, between 0 and 1.
... see amOptions for more options.

References

See online documentation https://datastorm-open.github.io/introduction_ramcharts/ and amChartsAPI

See Also

amOptions, amBarplot, amBoxplot, amHist, amPie, amPlot, amTimeSeries, amStockMultiSet, amBullet, amRadar, amWind, amFunnel, amAngularGauge, amSolidGauge, amMekko, amCandlestick, amFloatingBar, amOHLC, amWaterfall

Examples

data("data_wind")
amWind(data_wind)

# Other examples available which can be time consuming depending on your configuration.

# Change color
amWind(data = data_wind, col = "#0404B4")
amWind(data = data_wind, col = c("#0404B4","#01DF01","#FFBF00"))

# Change backTransparency
amWind(data = data_wind, col = c("#0404B4","#01DF01","#FFBF00"), backTransparency = 0.1)
amWind(data = data_wind, col = c("#0404B4","#01DF01","#FFBF00"), backTransparency = 1)
amWind(data = data_wind, col = c("#0404B4","#01DF01","#FFBF00"), backTransparency = c(0.1, 0.1, 1))
AxisBase-class

api

See AmCharts API

Description

Open a window in your browser at the referenced documentation under http://docs.amcharts.com/3/javascriptstockchart/.

Usage

api(class = NULL)

Arguments

class Object of class character. Name of the class to see documentation. Please respect lower and upper case.

Examples

api()
api("AmChart")

AxisBase-class

AxisBase class

Description

Base class for ValueAxis and CategoryAxis. It can not be explicitly instantiated.

Slots

guides list

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.

otherProperties list containing other available properties not yet implemented in the package.

value numeric. Guides of this axis. Use addGuide method.

Author(s)

datastorm-open
CategoryAxis-class

Description

Children class of AxisBase. Automatically set.

Details

Run api("CategoryAxis") for more information and all available properties.

Slots

gridPosition character. Specifies if a grid line is placed on the center of a cell or on the beginning of a cell. Possible values are: "start" and "middle" This setting doesn’t work if parseDates is set to TRUE.

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.

otherProperties list containing other available properties not yet implemented in the package.

value numeric.

Author(s)

datastorm-open

ChartCursor-class

Description

Creates a cursor for the chart which follows the mouse movements. In case of AmSerialChart charts it shows the balloons of hovered data points.

Details

Run api("ChartCursor") for more information and all available properties.

Slots

oneBalloonOnly logical. If TRUE, border color will be changed when user rolls-over the slice, graph, etc, instead of background color.

valueLineAxis list. Properties of Axis of value line. If you set valueLineBalloonEnabled to true, but you have more than one axis, you can use this property to indicate which axis should display balloon.
listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.

otherProperties list containing other available properties not yet implemented in the package.

value numeric.

Author(s)

datastorm-open
controlShinyPlot  Tests the class of an expression.

Description

Only used in 'renderAmCharts'.

Usage

controlShinyPlot(x)

Arguments

x  expression passed to 'renderAmCharts'. Either an expression that generates an HTML widget, or an expression that generates an AmChart.

Details

This function has only an internal purpose. Never use it.

DataSet-class  DataSet class

Description

DataSet is an object which holds all information about data for AmStockChart

Details

Run api("DataSet") for more information.

Slots

dataProvider list, the data set data. Important: the data sets need to come pre-ordered in ascending order. Data with incorrect order might result in visual and functional glitches on the chart.

fieldMappings list, field mappings. Field mapping is an object with fromField and toField properties. fromField is the name of your value field in dataProvider. toField might be chosen freely, it will be used to set value/open/close/high/low fields for the StockChart. Example: fromField: "val1", toField: "value".

stockEvents list of StockEvent.

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.

otherProperties list containing other available properties not yet implemented in the package.

value numeric.
**DataSetSelector-class**  
*DataSetSelector class*

**Description**  
DataSetSelector is a tool used to select data sets as main and to compare to main data set.

**Details**  
Run `api("DataSetSelector")` for more information and all available properties.

**Slots**  
- **position** character. Possible values: "right", "left", "top", "bottom". "top" and "bottom" positions have a limitation - only one data set can be selected for comparing.
- **listeners** list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.
- **otherProperties** list containing other available properties not yet implemented in the package.
- **value** numeric.

**Author(s)**  
- datastorm-open

---

**data_AirPassengers**  
*Air passengers for example*

**Description**  
Based on the dataset 'AirPassengers' from the package 'datasets'.

**Usage**  
```r
data_AirPassengers
```

**Format**  
- 2 column, 144 rows:
  - **AirPassengers** numeric
  - **Period** character, MM/YYYY
**data_bar**

*Random data for plotting bar chart examples*

---

**Description**

This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

**Usage**

`data_bar`

**Format**

Dataset of 3 columns and 12 rows

- **country** character
- **visits** numeric
- **color** character

---

**data_candleStick1**

*Random data for plotting candlestick chart examples*

---

**Description**

This dataset is used in the tutorial, moreover if you notice a bug, use this dataset to give us an example.

**Usage**

`data_candleStick1`

**Format**

Dataset of 5 columns and 12 rows

- **category** character, can be parsed as a date
- **open** numeric
- **close** numeric
- **low** numeric
- **high** numeric
**data_candleStick2**

*Random data for plotting candlestick chart examples*

**Description**

This dataset is used in the tutorial, moreover if you notice a bug, use this dataset to give us an example.

**Usage**

data_candleStick2

**Format**

Dataset of 5 columns and 12 rows

- **category** character, can be parsed as a date
- **open** numeric
- **close** numeric
- **low** numeric
- **high** numeric

**data_fbar**

*Random data for plotting floating bar chart examples*

**Description**

This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

**Usage**

data_fbar

**Format**

Dataset of 2 columns and 1000 rows

- **country** character
- **visits_inf** numeric
- **visits_sup** numeric
- **color** character
data_funnel

Random data for plotting funnel chart examples

Description
This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

Usage
data_funnel

Format
Dataset of 2 columns and 7 rows
description character
value numeric

data_gantt
Random data for plotting gantt chart examples

Description
This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

Usage
data_gantt

Format
Dataset of 5 columns and 4 rows
category character
begin date
end date
color character
**data_gbar**

Random data for plotting bar chart examples

**Description**

This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

**Usage**

data_gbar

**Format**

Dataset of 5 columns and 12 rows

- **year** character
- **day** character
- **month** character
- **income** numeric
- **expenses** numeric

**data_gdp**

10 Richest Countries in the World by 2015 GDP

**Description**

Value in $ trillion

**Usage**

data_gdp

**Format**

Dataset of 2 columns and 10 rows

- **country** character
- **gdp** numeric

**Source**

**data_mekko**

| data_mekko | Random data for plotting mekko chart examples |

**Description**

This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

**Usage**

data_mekko

**Format**

Dataset of 2 columns and 1000 rows

- **var1** character
- **var2** numeric

**data_pie**

| data_pie | Random data for plotting pie chart examples |

**Description**

This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

**Usage**

data_pie

**Format**

Dataset of 2 columns and 5 rows

- **label** character
- **value** numeric
**data_radar**

*Random data for plotting radar chart examples*

**Description**

This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

**Usage**

```
data_radar
```

**Format**

- Dataset of 4 columns and 5 rows
- **label** character
- **Product1** numeric
- **Product2** numeric
- **Product3** numeric

**data_stock1**

*Random data for example*

**Description**

A list containing 4 datasets.

**Usage**

```
data_stock1
```

**Format**

- Each dataset is a data.table with 21 rows and 4 variables:
  - **date** vector of dates
  - **a** random vector of data
  - **b** random vector of data
data_stock_2

Random data for example

Description
Times Series on 2015-2016, one data by hour

Usage
data_stock_2

Format
Each dataset is a data.table with 21 rows and 4 variables:

date vector of dates

Times

ts1 random vector of data

Random vector of data
ts2 random vector of data

data_stock_3

Random data for example

Description
Times Series on 2017, by months

Usage
data_stock_3

Format
List of 4 datasets, 4 variables in each

date vector of dates

Times

value random vector of data

volume random vector of data

value2 random vector of data

value3 random vector of data
data_waterfall

Description
This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

Usage
data_waterfall

Format
Dataset of 3 columns and 15 rows

- **label** character, can be parsed as a date
- **value** numeric
- **operation** numeric

---

data_wind

Description
This dataset is used in the examples, moreover it can be used as a reference if you notice a bug.

Usage
data_wind

Format
Dataset of 3 columns and 8 rows

- **weak** numeric
- **middle** numeric
- **strong** numeric
GaugeArrow-class

Description
Creates an arrow for AmAngularGaugeChart, multiple can be assigned.

Details
Run api("GaugeArrow") for more information and all available properties.

Slots
axis list containing properties of GaugeAxis. Axis of the arrow. You can use reference to the axis or id of the axis. If you don’t set any axis, the first axis of the chart will be used.
listeners list containing the listeners to add to the chart. The list must be named as in the official API. Each element must be a character string.
otherProperties list containing other available properties not yet implemented in the package.
value numeric.

Author(s)
datastorm-open

GaugeAxis-class

Description
Creates an axis for AmAngularGaugeChart, multiple can be assigned.

Details
Run api("GaugeAxis") for more information and all available properties.

Slots
bands list containing properties of one or several GaugeBand objects. Bands are used to draw color fills between specified values.
listeners list containing the listeners to add to the chart. The list must be named as in the official API. Each element must be a character string.
otherProperties list containing other available properties not yet implemented in the package.
value numeric.

Author(s)
datastorm-open
GaugeBand-class  

Generics functions

Description

Creates a band for a specified value range on the GaugeAxis. Multiple bands can be assigned to a single GaugeAxis.

Details

Run api("GaugeBand") for more information and all available properties.

Slots

id character. Unique id of a band.

listeners list containing the listeners to add to the chart. The list must be named as in the official API. Each element must be a character string.

otherProperties list containing other available properties not yet implemented in the package.

value numeric.

Author(s)

datastorm-open

Generics functions  

General shared generic methods

Description

These methods are shared by at least two different classes

Usage

setBalloon(.Object, amBalloon = NULL, ...)

setDataProvider(.Object, dataProvider, keepNA = TRUE)

setTitle(.Object, title)

setType(.Object, type)

setGraph(.Object, graph = NULL, ...)

addGuide(.Object, guide = NULL, ...)
Generics functions

setText(.Object, text)

setValueAxis(.Object, valueAxis = NULL, ...)

Arguments

/Object/ AmObject.
/amBalloon/ AmBalloon.
/.../ Other properties.
/dataProvider/ data.frame.
/keepNA/ logical, default set to TRUE. Indicates if NULL values have to be kept or ignored.
/title/ character.
/type/ character.
/graph/ AmGraph.
/guide/ Guide.
/text/ character.
/valueAxis/ ValueAxis.

Details

Be cautious when using one of these functions since they have several signatures (S4).

setText is shared by Title and Label.

setValueAxis is shared by AmChart(type = "gantt"), TrendLine and Guide.

Value

An updated 'AmObject'.
getCurrentStockData  Get data in shiny module

Description

Get data in shiny module

Usage

generateCurrentStockData(
  data,
  col_date,
  col_series,
  zoom = NULL,
  maxPoints = 1000,
  tz = "UTC",
  ts = c("5 min", "10 min", "30 min", "hour", "12 hour", "day", "week", "month", "year"),
  fun_aggr = "mean",
  treat_missing = FALSE,
  maxgap = Inf,
  type_aggr = "first"
)

Arguments

data : data.frame to transform
col_date : Date column name, default to "date". Must be "POSIXct" or "CET24" column
col_series : Column name of quantitative variable(s) to be transformed. Default to setdiff(colnames(data), "date")
zoom : List for init subset. NULL to keep all
maxPoints : Maximal number of rows in results
tz : Timezone of result. Default to "UTC".
ts : Increment of the sequence. Default to "10 min". Can be a number, in seconds, or a character string containing one of "min", "hour", "day". This can optionally be preceded by a positive integer and a space
fun_aggr : Aggregation function to use ("min", "max", "sum", "mean", "first", "last"). Default to "mean".
treat_missing : Boolean. Default to FALSE Whether or not to interpolate missing values ? see na.approx
maxgap : When interpolate missing values with na.approx. Maximum number of consecutive NAs to fill. Default to Inf.
type_aggr : Character. Type of aggregation
getTransformTS

- "first": Date/Time result is equal to minimum of sequence, and this minimum is included in aggregation
- "last": Date/Time result is equal to maximum of sequence, and this maximum is included in aggregation

---

Transform quantitative variables.

Description

Transform quantitative variables. Aggregate or interpolate time series data.

Usage

getTransformTS(
  data,
  col_date = "date",
  col_series = setdiff(colnames(data), c(col_date, col_by)),
  col_by = NULL,
  ts = "10 min",
  tz = "UTC",
  fun_aggr = "mean",
  treat_missing = FALSE,
  control_date = TRUE,
  maxgap = Inf,
  keep_last = TRUE,
  type_aggr = "first",
  showwarn = FALSE
)

Arguments

data : data.frame to transform
col_date : Date column name, default to "date". Must be "POSIXct"
col_series : Column name of quantitative variable(s) to be transformed. Default to setdiff(colnames(data), "date")
col_by : Column name of an optional grouping variable. Default to NULL
ts : Increment of the sequence. Default to "10 min". Can be a number, in seconds, or a character string containing one of "min", "hour", "day". This can optionally be preceded by a positive integer and a space
tz : Timezone of result. Default to "UTC".
fun_aggr : Aggregation function to use ("min", "max", "sum", "mean", "first", "last"). Default to "mean".
treat_missing : Boolean. Default to FALSE Whether or not to interpolate missing values? see na.approx
control_date : Boolean. Control full data sequence? Default to TRUE and set to TRUE if present.
maxgap : When interpolate missing values with na.approx. Maximum number of consecutive NAs to fill. Default to Inf.
keep_last : Boolean. Keep last date/time value after interpolation?
type_aggr : Character. Type of aggregation
  • "first" : Date/Time result is equal to minimum of sequence, and this minimum is included in aggregation
  • "last" : Date/Time result is equal to maximum of sequence, and this maximum is included in aggregation
showwarn : Boolean. Show warnings?

Value
a data.frame

Description
Creates a horizontal/vertical guideline/area for amSerialChart, amXYChart and amRadarChart charts, automatically adapts it's settings from the axes if none has been specified.

Details
Run api("Guide") for more information and all available properties.

Slots
fillAlpha numeric. Specifies if a grid line is placed on the center of a cell or on the beginning of a cell. Possible values are: "start" and "middle" This setting doesn’t work if parseDates is set to TRUE.
valueAxis ValueAxis. As you can add guides directly to the chart, you might need to specify which value axis should be used.
listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.
otherProperties list containing other available properties not yet implemented in the package.
value numeric.

Author(s)
datastorm-open
**Description**

Initializes or updates an object **AmBalloon**.

**Usage**

```r
## S4 method for signature 'AmBalloon'
initialize(.Object, adjustBorderColor, color, cornerRadius, fillColor, ...)

amBalloon(adjustBorderColor, color, cornerRadius, fillColor, ...)

setAdjustBorderColor(.Object, adjustBorderColor)

## S4 method for signature 'AmBalloon,logical'
setAdjustBorderColor(.Object, adjustBorderColor)

setColor(.Object, color)

## S4 method for signature 'AmBalloon,character'
setColor(.Object, color)

setCornerRadius(.Object, cornerRadius)

## S4 method for signature 'AmBalloon,numeric'
setCornerRadius(.Object, cornerRadius)

setFillColor(.Object, fillColor)

## S4 method for signature 'AmBalloon,character'
setFillColor(.Object, fillColor)
```

**Arguments**

- `.Object` **AmBalloon**.
- `adjustBorderColor` logical, if TRUE, border color will be changed when user rolls-over the slice, graph, etc. instead of background color.
- `color` character, balloon text color.
- `cornerRadius` numeric, balloon corner radius.
- `fillColor` character, balloon background color. It is usually defined by the chart itself. If "adjustBorderColor" is set to TRUE, the balloon background color will be equal to "fillColor".
The `initialize.AmChart` method

Creating an `AmChart`

**Value**

An object, possibly updated, of class `AmBalloon`.

**Examples**

```r
new("AmBalloon", cornerRadius = 10)
```

```r
amBalloon(adjustBorderColor = TRUE, color = "#000000", other = TRUE)
```

```r
setAdjustBorderColor(.Object = amBalloon(), adjustBorderColor = TRUE)
```

```r
setColor(.Object = amBalloon(), color = "#000000")
```

```r
setCornerRadius(.Object = amBalloon(), cornerRadius = 5)
```

```r
setFillColor(.Object = amBalloon(), fillColor = "#FFFFFF")
```

---

**Description**

Method to initialize any S4 class provided by the package.

**Usage**

```r
## S4 method for signature 'AmChart'
initialize(
  .Object,
  allLabels,
  arrows,
  axes,
  balloon,
  categoryAxis,
  categoryField,
  chartCursor,
  chartScrollbar,
  creditsPosition,
  dataProvider,
  graphs,
  graph,
)```
initialize, AmChart-method

guides,
legend,
segmentsField,
theme,
titles,
trendLines,
type,
valueAxes,
valueAxis,
valueScrollbar,
...
)

amChart(
anAllLabels,
arrows,
axes,
balloon,
categoryAxis,
categoryField,
chartCursor,
chartScrollbar,
creditsPosition,
dataProvider,
graph,
graphs,
guides,
legend,
segmentsField,
theme,
titles,
trendLines,
type,
valueAxes,
valueAxis,
...
)

amAngularGaugeChart(arrows, titles, axes, ...)

amFunnelChart(dataProvider, marginLeft = 10, marginRight = 10, ...)

amRadarChart(
anAllLabels,
balloon,
categoryField,
creditsPosition,
dataProvider,
graphs,
guides,
legend,
titles,
valueAxes,
...)

amSerialChart(
  allLabels,
bubble,
categoryAxis,
categoryField,
chartCursor,
chartScrollbar,
creditsPosition,
dataProvider,
graphs,
guides,
legend,
titles,
trendLines,
valueAxes,
...)

amPieChart(
  allLabels,
bubble,
creditsPosition,
dataProvider,
legend,
titles,
...)

amGanttChart(categoryField, dataProvider, graph, segmentsField, valueAxis, ...)

amXYChart(creditsPosition, dataProvider, graphs, ...)

setAllLabels(.Object, allLabels)

## S4 method for signature 'AmChart,list'
setAllLabels(.Object, allLabels)

addLabel(.Object, label = NULL, ...)

## S4 method for signature 'AmChart,LabelOrMissing'
initialize,AmChart-method

```
addLabel(.Object, label = NULL, ...)
setArrows(.Object, arrows = NULL)

## S4 method for signature 'AmChart'
setArrows(.Object, arrows = NULL)
addArrow(.Object, arrow = NULL, ...)

## S4 method for signature 'AmChart,GaugeArrowOrMissing'
addArrow(.Object, arrow = NULL, ...)
setAxes(.Object, axes, ...)

## S4 method for signature 'AmChart,list'
setAxes(.Object, axes)
addAxe(.Object, axe = NULL, ...)

## S4 method for signature 'AmChart,GaugeAxisOrMissing'
addAxe(.Object, axe = NULL, ...)
addAxis(.Object, axis = NULL, ...)

## S4 method for signature 'AmChart,GaugeAxisOrMissing'
addAxis(.Object, axis = NULL, ...)

## S4 method for signature 'AmChart,AmBalloonOrMissing'
setBalloon(.Object, amBalloon = NULL, ...)
setCategoryAxis(.Object, categoryAxis = NULL, ...)

## S4 method for signature 'AmChart'
setCategoryAxis(.Object, categoryAxis = NULL, ...)
setCategoryField(.Object, categoryField)

## S4 method for signature 'AmChart,character'
setCategoryField(.Object, categoryField)
setChartCursor(.Object, chartCursor = NULL, ...)

## S4 method for signature 'AmChart,ChartCursorOrMissing'
setChartCursor(.Object, chartCursor = NULL, ...)
setChartScrollbar(.Object, chartScrollbar = NULL, ...)

## S4 method for signature 'AmChart,ChartScrollbarOrMissing'
```
initialize,AmChart-method

setChartScrollbar(.Object, chartScrollbar = NULL, ...)
setCreditsPosition(.Object, creditsPosition)

## S4 method for signature 'AmChart,character'
setCreditsPosition(.Object, creditsPosition)

setDataLoader(.Object, url, format, ...)

## S4 method for signature 'AmChart,character,character'
dataSetLoader(.Object, url, format, ...)

## S4 method for signature 'AmChart,ANY,logicalOrMissing'
dataSetProvider(.Object, dataProvider, keepNA = TRUE)

setGraphs(.Object, graphs)

## S4 method for signature 'AmChart,list'
setGraphs(.Object, graphs)

addGraph(.Object, amGraph = NULL, ...)

## S4 method for signature 'AmChart,AmGraphOrMissing'
addGraph(.Object, amGraph = NULL, ...)

## S4 method for signature 'AmChart,AmGraphOrMissing'
setGraph(.Object, graph = NULL, ...)

setGuides(.Object, guides)

## S4 method for signature 'AmChart,list'
setGuides(.Object, guides)

## S4 method for signature 'AmChart,GuideOrMissing'
addGuide(.Object, guide = NULL, ...)

setLegend(.Object, amLegend = NULL, ...)

## S4 method for signature 'AmChart,AmLegendOrMissing'
setLegend(.Object, amLegend = NULL, ...)

addSegment(.Object, categoryIDs, sgts)

## S4 method for signature 'AmChart,numeric'
addSegment(.Object, categoryIDs, sgts)

addSubData(.Object, categoryIDs, data)
## S4 method for signature 'AmChart,numeric'
addSubData(.Object, categoryIDs, data)

setSubChartProperties(.Object, .subObject = NULL, ...)

## S4 method for signature 'AmChart'
setSubChartProperties(.Object, .subObject = NULL, ...)

setTheme(.Object, theme)

## S4 method for signature 'AmChart,character'
setTheme(.Object, theme)

setTitles(.Object, titles)

## S4 method for signature 'AmChart,list'
setTitles(.Object, titles)

addTitle(.Object, title = NULL, ...)

## S4 method for signature 'AmChart,TitleOrMissing'
addTitle(.Object, title = NULL, ...)

setTrendLines(.Object, trendLines)

## S4 method for signature 'AmChart,list'
setTrendLines(.Object, trendLines)

addTrendLine(.Object, trendLine = NULL, ...)

## S4 method for signature 'AmChart,TrendLineOrMissing'
addTrendLine(.Object, trendLine = NULL, ...)

## S4 method for signature 'AmChart,character'
setType(.Object, type)

setValueAxes(.Object, valueAxes)

## S4 method for signature 'AmChart,list'
setValueAxes(.Object, valueAxes)

addValueAxes(.Object, valueAxis = NULL, ...)

## S4 method for signature 'AmChart,ValueAxisOrMissing'
addValueAxes(.Object, valueAxis = NULL, ...)

addValueAxis(.Object, valueAxis = NULL, ...)
## S4 method for signature 'AmChart,ValueAxisOrMissing'
addValueAxis(.Object, valueAxis = NULL, ...)

## S4 method for signature 'AmChart,ValueAxisOrMissing'
setValueAxis(.Object, valueAxis = NULL, ...)

setValueScrollbar(.Object, valueScrollbar, ...)

## S4 method for signature 'AmChart,ChartScrollbarOrMissing'
setValueScrollbar(.Object, valueScrollbar, ...)

### Arguments

- **.Object** : *AmChart*
- **allLabels** : list of *Label*. Example of a label object, with all possible properties: label(x = 20, y = 20, text = "this is a label", align = "left", size = 12, color = "#CC0000", alpha = 1, rotation = 0, bold = TRUE, url = "http://www.amcharts.com"). Run api("Label") for more informations.
- **arrows** : list of *GaugeArrow*. Only valid for gauge charts. Run api("GaugeArrow") for more informations.
- **axes** : list of *GaugeAxis* properties. Only valid for gauge charts. Run api("GaugeAxis") for more informations.
- **balloon** : *AmBalloon*. Creates the balloons (tooltips) of the chart. It follows the mouse cursor when you roll-over the data items. The framework automatically generates the instances you just have to adjust the appearance to your needs. Run api("AmBalloon") for more informations.
- **categoryAxis** : *CategoryAxis*. Read-only. Chart creates category axis itself. If you want to change some properties, you should get this axis from the chart and set properties to this object. Run api("CategoryAxis") for more informations.
- **categoryField** : character, category field name indicates the name of the field in your dataProvider object which will be used for category axis values.
- **chartCursor** : *ChartCursor*. Chart's cursor. Run api("ChartCursor") for more informations.
- **chartScrollbar** : *ChartScrollbar*. Chart's scrollbar. Run api("ChartScrollbar") for more informations.
- **creditsPosition** : character, specifies position of the amCharts’ website link. Allowed values are: "top-left", "top-right", "bottom-left" and "bottom-right".
- **dataProvider** : data.frame, containing the data.
- **graphs** : list of *AmGraph*. Creates the visualization of the data in following types: line, column, step line, smoothed line, olhc and candlestick. Run api("AmGraph") for more informations.
- **graph** : *AmGraph*. Only valid for Gantt charts. Gantt chart actually creates multiple graphs (separate for each segment). Properties of this graph are passed to each of the created graphs - this allows you to control the look of segments. Run api("AmGraph") for more informations.
initialize, AmChart-method

- **guides** - list of Guide. Instead of adding guides to the axes, you can push all of them to this array. In case guide has category or date defined, it will automatically be assigned to the category axis, otherwise to the first value axis, unless you specify a different valueAxes for the guide. Run api("Guide") for more informations.

- **legend** - AmLegend. Legend of a chart. Run api("AmLegend") for more informations.

- **segmentsField** - character, segments field in your data provider. Only valid for Gantt Charts.


- **titles** - list of Title. Run api("Title") for more informations.

- **trendLines** - list of TrendLine objects added to the chart. You can add trend lines to a chart using this list or access already existing trend lines. Run api("TrendLine") for more informations.

- **type** - character, possible types are: "serial", "pie", "radar", "xy", "radar", "funnel", "gauge", "stock". See details about using argument type. (type map is in development).

- **valueAxes** - list of ValueAxis. Chart creates one value axis automatically, so if you need only one value axis, you don’t need to create it. Run api("ValueAxis") for more informations.

- **valueAxis** - ValueAxis. Only valid for Gantt Charts. Set it's type to "date" if your data is date or time based. Run api("ValueAxis") for more informations.

- **valueScrollbar** - ChartScrollbar. Value scrollbar, enables scrolling value axes.

- **marginLeft** - character, left margin of the chart.

- **marginRight** - character, right margin of the chart.

- **label** - (optional) Label. Argument of method addLabel.

- **arrow** - (optional) GaugeArrow. Argument of method addArrow.


- **axis** - (optional) GaugeAxis. same as axe.

- **amBalloon** - AmBalloon, argument of method 'setBalloon'.

- **url** - character.

- **format** - character.

- **keepNA** - object of class logical, default TRUE. Indicates if NULL values have to be kept or ignored.

- **amGraph** - (optional) AmGraph.


- **amLegend** - (optional) AmLegend.

- **categoryIDs** - numeric, see details.

- **sgts** - data.frame (or list of data.frame for multiple add ).
initialize,AmChart-method

data
.data.frame. Data to draw at the second level, after clicking on the column.

.subObject AmChart.

title (optional) Title, argument of method addTitle.

trendLine (optional) TrendLine. Argument of method addTrendLine.

Details

amAngularGaugeChart is a shortcut for instantiating AmChart of type gauge.
amFunnelChart is a shortcut for instantiating AmChart of type funnel.
amRadarChart is a shortcut for instantiating AmChart of type radar.
amSerialChart is a shortcut constructor for instantiating AmChart of type serial.
amPieChart is a shortcut constructor for instantiating AmChart of type pie.
amGanttChart is a constructor for instantiating AmChart of type gantt.
amXYChart is a shortcut constructor for instantiating AmChart of type xy.

Method 'addAxe' is deprecated, use 'addAxis'.
Method setGraph is only valid for Gantt Charts.

'addSubData' allows to add subdata for a chart with drilldown. In this case, categoryIDs indicates corresponding indice(s) of the dataProvider where to add the data.
For method addValueAxis: valueAxis is optional. Method addValueAxes is deprecated.
Method setValueAxis is only valid for Gantt charts.

Value

(updated) AmChart with given properties.

See Also

Refer to http://docs.amcharts.com/3/javascriptcharts/.

Examples

new("AmChart", valueField = "value", theme = "patterns")

amChart(type = "pie")

amAngularGaugeChart()

amFunnelChart(marginLeft = 15)

amRadarChart()}
initialize, AmChart-method

```r
amSerialChart(creditsPosition = "top-right")

amPieChart()

amGanttChart(segmentsField = "segments")

amXYChart()

allLabels <- list(label(text = "balloonText"), label(text = "column"))
amSerialChart(allLabels = allLabels)

# ---
addLabel(.Object = amSerialChart(), text = "balloonText")
# equivalent to:
label_obj <- label(text = "balloonText")
addLabel(.Object = amSerialChart(), label = label_obj)

# ---
arrows_ls <- list(gaugeArrow(value = 130), gaugeArrow(value = 150))
amAngularGaugeChart(arrows = arrows_ls)

# ---
chart <- addArrow(.Object = amAngularGaugeChart(), value = 10); print(chart)
# equivalent to:
gaugeArrow_obj <- gaugeArrow(value = 10)
addArrow(.Object = amAngularGaugeChart(), arrow = gaugeArrow_obj)

# ---
axes_ls <- list(gaugeAxis(value = 130), gaugeAxis(value = 150))
setAxes(.Object = amAngularGaugeChart(), axes = axes_ls)
# If possible, simplify your code by using the constructor:
amAngularGaugeChart(axes = axes_ls)

# ---
addAxis(.Object = amAngularGaugeChart(), startValue = 0, endValue = 100, valueInterval = 10)
# equivalent to:
gaugeAxis_obj <- gaugeAxis(startValue = 0, enValue = 100, valueInterval = 10)
addAxis(.Object = amAngularGaugeChart(), axis = gaugeAxis_obj)

# ---
setBalloon(.Object = amSerialChart(), adjustBorderColor = TRUE, fillColor = "#FFFFFF",
             color = "#000000", cornerRadius = 5)
# equivalent to:
```

initialize-AmChart-method

amBalloon_obj <- amBalloon(adjustBorderColor = TRUE, fillColor = "#FFFFFF",
  color = "#000000", cornerRadius = 5)
setBalloon(.Object = amSerialChart(), amBalloon = amBalloon_obj)

# ---
setCategoryAxis(.Object = amSerialChart(), gridPosition = "start")
# equivalent to:
categoryAxis_obj <- categoryAxis(gridPosition = "start")
setCategoryAxis(.Object = amSerialChart(), categoryAxis = categoryAxis_obj)

# ---
setCategoryField(.Object = amSerialChart(), categoryField = "country")
# ---
# with default value, no argument needed
setChartCursor(.Object = amSerialChart())
# other example
setChartCursor(.Object = amSerialChart(), oneBallOnly = TRUE)
# equivalent to
chartCursor_obj <- chartCursor(oneBallOnly = TRUE)
setChartCursor(.Object = amSerialChart(), chartCursor = chartCursor_obj)

# ---
# Add the default scrollbar
setChartScrollbar(.Object = amSerialChart())
# equivalent to:
chartScrollbar_obj <- chartScrollbar(updateOnReleaseOnly = FALSE)
setChartScrollbar(.Object = amSerialChart(), chartScrollbar = chartScrollbar_obj)

# ---
setCreditsPosition(.Object = amPieChart(), creditsPosition = "top-right")

# ---
set DataLoader(.Object = amSerialChart(), url = "data.json", format = "json")

# ---
dataProvider_obj <- data.frame(key = c("FR", "US", "GER", "ENG", "IT" ),
  value = round(runif(5, max = 100)))
setDataProvider(.Object = amPieChart(), dataProvider = dataProvider_obj)

# ---
graphs_ls <- list(graph(balloonText = "balloonText"), graph(type = "column"))
setGraphs(.Object = amSerialChart(), graphs = graphs_ls)

# ---
addGraph(.Object = amSerialChart(), balloonText = "balloonText", type = "column")
# equivalent to
amGraph_obj <- amGraph(balloonText = "balloonText", type = "column")
addGraph(.Object = amSerialChart(), amGraph = amGraph_obj)

# ---

print(setGraph(.Object = amGanttChart(), id = "amGraph-1"))
# equivalent to:
amGraph_obj <- amGraph(id = "amGraph-1")
setGraph(.Object = amGanttChart(), amGraph = amGraph_obj)

# ---
guides_ls <- list(guide(fillAlpha = .1), guide(fillAlpha = .5))
amSerialChart(guides = guides_ls)

# ---
chart <- addGuide(.Object = amSerialChart(), fillAlpha = .1, value = 0, toValue = 10)
print(chart)
# equivalent to
guide_obj <- guide(fillAlpha = .1, value = 0, toValue = 10, valueAxis = "1")
addGuide(.Object = amSerialChart(), guide = guide_obj)

setLegend(.Object = amChart(), amLegend = amLegend(useGraphSettings = TRUE))
# equivalent to:
setLegend(.Object = amChart(), useGraphSettings = TRUE)

# ---

pipeR::pipeline(
  amGanttChart(segmentsField = "segments"),
  setDataProvider(data.frame(category = c("John", "Julia"))),
  addSegment(1, data.frame(start = 7, duration = 2:3, task = c("Task #1", "Task #2"))),
  addSegment(2, data.frame(start = 10, duration = 2:3, task = c("Task #1", "Task #2")))
)

# ---
ls <- list(data.frame(start = 7, duration = 2:3, task = c("Task #1", "Task #2")),
            data.frame(start = 10, duration = 2:3, task = c("Task #1", "Task #2")))
pipeR::pipeline(
  amGanttChart(segmentsField = "segments"),
  setDataProvider(data.frame(category = c("John", "Julia"))),
  addSegment(1:2, ls)
)

# ---
amChart_obj <- amChart(provider = data.frame(a = 1:5, b = 6:10))
addSubData(.Object = amChart_obj, categoryIDs = 3, data = data.frame(a = 1:10, b = 11:20))
# ---
setSubChartProperties(.Object = amSerialChart(), type = "serial")

# ---

setTheme(.Object = amPieChart(), theme = "dark")

# ---

titles_ls <- list(amTitle(text = "balloonText"), amTitle(text = "column"))
setTitles(.Object = amXYChart(), titles = titles_ls)
# or...
amXYChart(titles = titles_ls)

# ---

addTitle(.Object = amPieChart(), text = "balloonText", size = 15)
# equivalent to
title_obj <- amTitle(text = "balloonText", size = 15)
addTitle(.Object = amPieChart(), title = title_obj)

# ---

trendLines <- list(trendLine(initialValue = 1, finalValue = 5),
                   trendLine(initialValue = 7, finalValue = 19))
setTrendLines(.Object = amSerialChart(), trendLines = trendLines)
# or...
amSerialChart(trendLines = trendLines) # Equivalent

# ---

addTrendLine(.Object = amSerialChart(), initialValue = 1, initialXValue = 1,
              finalValue = 11, finalXValue = 12)
# equivalent to:
trendLine_obj <- trendLine(initialValue = 1, initialXValue = 1, finalValue = 11, finalXValue = 12)
chart <- addTrendLine(.Object = amSerialChart(), trendLine = trendLine_obj); print(chart)

# ---

setType(.Object = amChart(), type = "pie")
# equivalent to:
amPieChart()

valueAxes <- list(valueAxis(axisTitleOffset = 12, tickLength = 10),
                   valueAxis(axisTitleOffset = 10, tickLength = 10))
setValueAxes(.Object = amSerialChart(), valueAxes = valueAxes)
# or...
amSerialChart(valueAxes = valueAxes)

# ---

print(addValueAxis(.Object = amSerialChart(), axisTitleOffset = 12, tickLength = 10, title = "foo"))
# equivalent to:
```r
valueAxis_obj <- valueAxis(axisTitleOffset = 12, tickLength = 10, title = "foo")
addValueAxis(.Object = amSerialChart(), valueAxis = valueAxis_obj)

# ---

setValueAxis(.Object = amGanttChart())
setValueAxis(.Object = amGanttChart(), type = "date")

valueScrollbar_obj <- chartScrollbar(updateOnReleaseOnly = FALSE)
chart <- setValueScrollbar(.Object = amSerialChart(), valueScrollbar = valueScrollbar_obj)
print(chart)
# or...
amSerialChart(updateOnReleaseOnly = FALSE)

# ---
```

---

### initialize, AmGraph-method

*Initializes an AmGraph*

---

**Description**

To create an AmGraph, you can use the usual method `Initialize` or the constructor. You can update properties with setters.

**Usage**

```
## S4 method for signature 'AmGraph'
initialize(
  .Object,
  animationPlayed = FALSE,
  balloonText,
  title,
  type,
  valueField,
  ...
)

amGraph(animationPlayed = FALSE, balloonText, title, type, valueField, ...)
graph(animationPlayed = FALSE, balloonText, title, type, valueField, ...)
setBalloonText(.Object, balloonText)

## S4 method for signature 'AmGraph,character'
setBalloonText(.Object, balloonText)
```
## S4 method for signature 'AmGraph,character'
setTitle(.Object, title)

## S4 method for signature 'AmGraph,character'
setType(.Object, type)

setValueField(.Object, valueField)

## S4 method for signature 'AmGraph,character'
setValueField(.Object, valueField)

### Arguments

- **.Object**: `AmGraph`
- **animationPlayed**: logical
- **balloonText**: character, balloon text. You can use tags like `[[value]]`, `[[description]]`, `[[percents]]`, `[[open]]`, `[[category]]` or any other field name from your data provider. HTML tags can also be used.
- **title**: character, graph title.
- **type**: character, type of the graph. Possible values are: "line", "column", "step", "smoothedLine", "candlestick", "ohlc". XY and Radar charts can only display "line" otherArguments graphs.
- **valueField**: character, name of the value field in your dataProvider.
- **...**: other properties of AmGraph. See [http://docs.amcharts.com/3/javascriptcharts/AmGraph](http://docs.amcharts.com/3/javascriptcharts/AmGraph).

### Value

An object of class `AmGraph` with the given properties.

### Examples

```r
# --- method 'initialize'
new("AmGraph", valueField = "value")

# constructor
amGraph(balloonText = "My text")

## Not run:
amGraph(balloonText = "balloonText", "type" = "column", title = "myGraph!",
valueField = "value", animationPlayed = TRUE, other = TRUE)

## End(Not run)
amGraph(balloonText = "some text")
# --- shortcut constructor
graph(balloonText = "balloonText", "type" = "column",
valueField = "value", animationPlayed = TRUE)

# --- update 'balloonText'
```
setBalloonText(.Object = amGraph(), balloonText = "performance")

# --- update 'title'
setTitle(.Object = amGraph(), title = "Power")

# --- update 'type'
setType(.Object = amGraph(), type = "type")

# --- update valueField
setValueField(.Object = amGraph(), valueField = "score")

---

initialize,AmLegend-method

Initializes legend of the chart

---

Description

Constructor for an AmLegend.

Usage

```r
## S4 method for signature 'AmLegend'
initialize(.Object, useGraphSettings, ...)

amLegend(useGraphSettings, ...)

legend(useGraphSettings, ...)

setUseGraphSettings(.Object, useGraphSettings)

## S4 method for signature 'AmLegend,logical'
setUseGraphSettings(.Object, useGraphSettings)
```

Arguments

- `.Object` *AmLegend*
- `useGraphSettings` logical, if TRUE, border color will be changed when user rolls-over the slice, graph, etc, instead of background color.
- `...` Other properties of *AmLegend*. See [http://docs.amcharts.com/3/javascriptstockchart/AmLegend](http://docs.amcharts.com/3/javascriptstockchart/AmLegend)

Examples

```r
new("AmLegend", useGraphSettings = TRUE)
amLegend(useGraphSettings = FALSE)
rAmCharts:::legend(useGraphSettings = FALSE)
setUseGraphSettings(.Object = amLegend(), useGraphSettings = TRUE)
```
initialize,AmStockChart-method

Initializes an AmStockChart

Description

Method to initialize any S4 class provided by the package.

Usage

```r
## S4 method for signature 'AmStockChart'
initialize(
  .Object,
  balloon,
  comparedDataSets,
  dataSets,
  dataSetSelector,
  mainDataSet,
  panels,
  periodSelector,
  theme,
  group,
  is_ts_module,
  ...
)

amStockChart(
  balloon,
  comparedDataSets,
  dataSets,
  dataSetSelector,
  mainDataSet,
  panels,
  periodSelector,
  theme,
  group,
  is_ts_module,
  ...
)

## S4 method for signature 'AmStockChart,AmBalloonOrMissing'
setBalloon(.Object, amBalloon = NULL, ...)

setCategoryAxesSettings(.Object, ...)
```

```r
## S4 method for signature 'AmStockChart'
setCategoryAxesSettings(.Object, ...)
```
setChartCursorSettings(.Object, ...)

## S4 method for signature 'AmStockChart'
setChartCursorSettings(.Object, ...)

setChartScrollbarSettings(.Object, chartScrollbarSettings = NULL, ...)

## S4 method for signature 'AmStockChart,ChartScrollbarOrMissing'
setChartScrollbarSettings(.Object, chartScrollbarSettings = NULL, ...)

setComparedDataSets(.Object, comparedDataSets)

## S4 method for signature 'AmStockChart'
setComparedDataSets(.Object, comparedDataSets)

addComparedDataSet(.Object, dataSet = NULL, ...)

## S4 method for signature 'AmStockChart,DataSetOrMissing'
addComparedDataSet(.Object, dataSet = NULL, ...)

setDataSets(.Object, dataSets)

## S4 method for signature 'AmStockChart'
setDataSets(.Object, dataSets)

addDataSet(.Object, dataSet = NULL, ...)

## S4 method for signature 'AmStockChart,DataSetOrMissing'
addDataSet(.Object, dataSet = NULL, ...)

setDataSetSelector(.Object, dataSetSelector = NULL, ...)

## S4 method for signature 'AmStockChart'
setDataSetSelector(.Object, dataSetSelector = NULL, ...)

setLegendSettings(.Object, ...)

## S4 method for signature 'AmStockChart'
setLegendSettings(.Object, ...)

setMainDataSet(.Object, dataSet = NULL, ...)

## S4 method for signature 'AmStockChart,DataSetOrMissing'
setMainDataSet(.Object, dataSet = NULL, ...)

setPanels(.Object, panels)
## S4 method for signature 'AmStockChart,list'
setPanels(.Object, panels)

addPanel(.Object, panel = NULL, ...)

## S4 method for signature 'AmStockChart,StockPanelOrMissing'
addPanel(.Object, panel = NULL, ...)

setPanelsSettings(.Object, ...)

## S4 method for signature 'AmStockChart'
setPanelsSettings(.Object, ...)

setPeriodSelector(.Object, periodSelector = NULL, ...)

## S4 method for signature 'AmStockChart,PeriodSelectorOrMissing'
setPeriodSelector(.Object, periodSelector = NULL, ...)

setStockEventsSettings(.Object, ...)

## S4 method for signature 'AmStockChart'
setStockEventsSettings(.Object, ...)

setValueAxesSettings(.Object, ...)

## S4 method for signature 'AmStockChart'
setValueAxesSettings(.Object, ...)

Arguments

- **.Object** `AmStockChart`
- **balloon** `AmBalloon`
- **comparedDataSets** list of `DataSet`. Properties of data sets selected for comparing.
- **dataSets** list of `DataSet`. Each element must be a list of `DataSet` properties.
- **dataSetSelector** list of `DataSetSelector`. You can add it if you have more than one data set and want users to be able to select/compare them.
- **mainDataSet** `DataSet`. Data set selected as main.
- **panels** list of `StockPanel`
- **periodSelector** `PeriodSelector`. You can add it if you want user’s to be able to enter date ranges or zoom chart with predefined period buttons.
- **theme** character.
- **group** character for synchronization
- **is_ts_module** boolean. Don’t use. For `rAmChartsTimeSeriesUI`
- **...** other properties of `AmStockChart`. 
amBalloon

argument for method setBalloon.

cartScrollbarSettings

Argument for method setScrollbar. If you change a property after the chart is initialized, you should call stockChart.validateNow() method in order for it to work. If there is no default value specified, default value of ChartScrollbar class will be used.

dataSet

Argument for method dataSet.

panel

Argument for method setPanel.

Details

CategoryAxesSettings sets common settings for all CategoryAxes of StockPanels. If you change a property after the chart is initialized, you should call stockChart.validateNow() method. If there is no specified value, default value of CategoryAxis class will be used. You should get this axis from the chart and set properties to this object.

ChartCursorSettings sets settings for chart cursor. If you change a property after the chart is initialized, you should call stockChart.validateNow() method. If there is no specified value, default value of ChartCursor class will be used.

You can add it if you have more than one data set and want users to be able to select/compare them.

Value

An object of class AmStockChart.

Examples

# --- method 'initialize'
new("AmStockChart", theme = "dark")

# --- constructor
amStockChart()

library(pipeR)

# Dummy example
amStockChart()

# Dummy example
setCategoryAxesSettings(.Object = amStockChart(), gridPosition = "start")

# Dummy example
setChartCursorSettings(.Object = amStockChart(), oneBallOnly = TRUE)

# Dummy example
initialize, AmStockChart-method

amchart <- setChartScrollbarSettings(.Object = amStockChart(), enabled = TRUE)
print(amchart)

# equivalent to:
chartScrollbarSettings_obj <- chartScrollbarSettings()
setChartScrollbarSettings(.Object = amStockChart(),
chartScrollbarSettings = chartScrollbarSettings_obj)

# Dummy example
comparedDataSets_ls <- list(dataSet(compared = TRUE), dataSet(compared = TRUE))
setComparedDataSets(.Object = amStockChart(), comparedDataSets = comparedDataSets_ls)

# Dummy example
addComparedDataSet(.Object = amStockChart(), compared = TRUE)

# Dummy example
dataSets_ls <- list(dataSet(compared = FALSE), dataSet(compared = FALSE))
setDataSets(.Object = amStockChart(), dataSets = dataSets_ls)

# Dummy example
addDataSet(.Object = amStockChart(), compared = FALSE)
# equivalent to:
dataSet_obj <- dataSet(compared = FALSE)
addDataSet(.Object = amStockChart(), dataSet = dataSet_obj)

# Dummy example
print(setDataSetSelector(.Object = amStockChart(), width = 180))
# equivalent to:
dataSetSelector_obj <- dataSetSelector(width = 180)
print(setDataSetSelector(.Object = amStockChart(),
dataSetSelector = dataSetSelector_obj))

# Dummy example
setLegendSettings(.Object = amStockChart(), equalWidths = TRUE)

# Dummy example
setMainDataSet(.Object = amStockChart(), showInCompare = TRUE)

# Dummy example
panels_ls <- list(stockPanel(compared = TRUE), stockPanel(compared = TRUE))
setPanels(.Object = amStockChart(), panels = panels_ls)

# Dummy example
initialize,CategoryAxis-method

**initialize, CategoryAxis-method**

*Initializes a CategoryAxis*

Description

Initializes or update a CategoryAxis.

Usage

```r
## S4 method for signature 'CategoryAxis'
initialize(.Object, gridPosition, guides, ...)

categoryAxis(gridPosition, ...)

setGridPosition(.Object, gridPosition)
```

Arguments

- `.Object` *CategoryAxis.*
- `gridPosition` character, specifies if a grid line is placed on the center of a cell or on the beginning of a cell. Possible values are: "start" and "middle" This setting doesn’t work if parseDates is set to TRUE.
- `guides` list of Guide.
- `...` Other properties.
Examples

guides <- list(guide(fillAlpha = .4, adjustBorderColor = TRUE),
guide(fillAlpha = .4, adjustBorderColor = TRUE))
new("CategoryAxis", gridPosition = "start", gridThickness = 1, guides = guides)

new("CategoryAxis")
new("CategoryAxis", gridPosition = "start", 1) # 1 is not take into account

categoryAxis(gridPosition = "start", adjustBorderColor = TRUE)
setGridPosition(.Object = categoryAxis(), gridPosition = "start")

Description

Initializes a ChartCursor.

Usage

## S4 method for signature 'ChartCursor'
initialize(.Object, oneBalloonOnly, valueLineAxis, ...)

chartCursor(animationDuration = 0.3, oneBalloonOnly, valueLineAxis, ...)

setOneBalloonOnly(.Object, oneBalloonOnly)

## S4 method for signature 'ChartCursor,logical'
setOneBalloonOnly(.Object, oneBalloonOnly)

setValueLineAxis(.Object, valueLineAxis = NULL, ...)

## S4 method for signature 'ChartCursor,ValueAxisOrCharacterOrMissing'
setValueLineAxis(.Object, valueLineAxis = NULL, ...)

Arguments

/Object/ ChartCursor.

oneBalloonOnly logical, if TRUE, border color will be changed when user rolls-over the slice, graph, etc, instead of background color.

valueLineAxis ValueAxis. If you set valueLineBalloonEnabled to TRUE, but you have more than one axis, you can use this property to indicate which axis should display balloon.
... other properties of ChartCursor. Run : api("ChartCursor") for more information.

animationDuration

numeric, duration of animation of a line, in seconds.

Value

(updated) .Object of class ChartCursor.

Examples

new("ChartCursor", oneBalloonOnly = TRUE)

chartCursor()

chartCursor(oneBalloonOnly = TRUE)

setOneBalloonOnly(.Object = chartCursor(), oneBalloonOnly = TRUE)

setValueLineAxis(.Object = chartCursor(), id = "valueAxis1",
                    title = "Hello !", axisTitleOffset = 12)

# equivalent to:
valueLineAxis_obj <- valueAxis(id = "valueAxis1", title = "Hello !", axisTitleOffset = 12)

setValueLineAxis(.Object = chartCursor(), valueLineAxis = valueLineAxis_obj)

# or iff 'valueLineAxis_obj' has already been added to the chart:
setValueLineAxis(.Object = chartCursor(), valueLineAxis = "valueAxis1")

Description

ChartScrollbarSettings sets settings for chart scrollbar. If you change a property after the chart is initialized, you should call stockChart.validateNow() method. If there is no default value specified, default value of ChartScrollbar class will be used. Run api("ChartScrollbarSettings") for more informations.

Usage

## S4 method for signature 'ChartScrollbar'
initialize(.Object, graph, enabled, ...)

chartScrollbar(graph, enabled = TRUE, ...)

chartScrollbarSettings(graph, enabled = TRUE, ...)

## S4 method for signature 'ChartScrollbar,AmGraphOrCharacterOrMissing'
setGraph(.Object, graph = NULL, ...)

setEnabled(.Object, enabled)
initialize,DataSet-method

## S4 method for signature 'ChartScrollbar,logical'
setEnabled(.Object, enabled)

### Arguments
-.Object: `ChartScrollbar`.
.graph: `AmGraph`. Specifies which graph will be displayed in the scrollbar.
.enabled: logical, specifies if the chart should be updated while dragging/resizing the scrollbar or only when user releases mouse button.

... other properties of ChartScrollbar. Run `api("ChartScrollbar")` for more information.

### Examples

```r
ew("ChartScrollbar", graph = "g1")
new("ChartScrollbar", graph = amGraph(test = 1))
chartScrollbar()
chartScrollbar(enabled = TRUE)
chartScrollbar()
chartScrollbar(enabled = TRUE)
# chartScrollbar with default graph
setGraph(.Object = chartScrollbar())

# example with arguments
setGraph(.Object = chartScrollbar(), id = "graph1", balloonText = "performance")
# equivalent to:
graph_obj <- amGraph(id = "graph1", balloonText = "performance")
(chartScrollbar_obj <- setGraph(.Object = chartScrollbar(), graph = graph_obj))
# or, iff graph_obj has already been added to the chart:
setGraph(.Object = chartScrollbar(), graph = "graph1")

# ---
setEnabled(.Object = chartScrollbar(), enabled = TRUE)
```

---

initialize,DataSet-method

*Creates or updates a DataSet*

### Description

Uses the constructors to create the object with its properties or updates an existing one with the setters.
initialize, dataSet-method

Usage

```r
## S4 method for signature 'DataSet'
initialize(
  .Object,
  compared = FALSE,
  dataProvider,
  fieldMappings,
  stockEvents,
  ...
)
```

dataSet(compared = FALSE, dataProvider, fieldMappings, stockEvents, ...)

```r
## S4 method for signature 'DataSet,ANY,ANY'
setDataProvider(.Object, dataProvider, keepNA = TRUE)
```

```r
setFieldMappings(.Object, fieldMappings)
```

```r
## S4 method for signature 'DataSet,list'
setFieldMappings(.Object, fieldMappings)
```

```r
addFieldMapping(.Object, ...)
```

```r
## S4 method for signature 'DataSet'
addFieldMapping(.Object, ...)
```

```r
setStockEvents(.Object, stockEvents)
```

```r
## S4 method for signature 'DataSet,list'
setStockEvents(.Object, stockEvents)
```

```r
addStockEvent(.Object, stockEvent = NULL, ...)
```

```r
## S4 method for signature 'DataSet,StockEventOrMissing'
addStockEvent(.Object, stockEvent = NULL, ...)
```

Arguments

- `.Object` **DataSet**.
- `compared` logical.
- `dataProvider` data.frame, the data set data. Important: the data sets need to come pre-ordered in ascending order. Data with incorrect order might result in visual and functional glitches on the chart.
- `fieldMappings` list of field mappings. Field mapping is an object with fromField and toField properties. fromField is a name of your value field in dataProvider. toField might be chosen freely, it will be used to set value/open/close/high/low fields for the StockGraph. Example: list(fromField = "val1", toField = "value").
initialize,DataSetSelector-method

Create or updates a DataSetSelector

Description

Use the constructors to create the object with its properties or update an existing one with the setters.

Usage

## S4 method for signature 'DataSetSelector'
initialize(.Object, position, ...)

dataSetSelector(position, ...)

setPosition(.Object, position)

## S4 method for signature 'DataSetSelector,character'
setPosition(.Object, position)

stockEvents  StockEvent.
... other properties of DataSet.
keepNA logical, TRUE to keep NA values.

Value

(updated) DataSet object

Examples

dataset <- new("DataSet")
dataset <- dataSet(categoryField = "categoryField")
setDataProvider(.Object = dataSet(), data.frame(key = c("FR", "US"), value = c(20,10)))
dataset <- addFieldMapping(.Object = dataSet(), fieldMappings = list(fromField ="val1", toField ="value"))
print(dataset)
dataset <- addFieldMapping(.Object = dataSet(), fromField ="val1", toField ="value")
print(dataset)
addStockEvent(.Object = dataSet(), backgroundAlpha = 1, backgroundColor = "#DADADA")
# equivalent to:
stockEvent_obj <- stockEvent(backgroundAlpha = 1, backgroundColor = "#DADADA")
chart <- addStockEvent(.Object = dataSet(), stockEvent = stockEvent_obj); print(chart)
Arguments

/Object\hfill \texttt{DataSetSelector}.

\textbf{position} \hfill character. Possible values: "right", "left", "top", "bottom". "top" and "bottom" positions has a limitation - only one data set can be selected for comparison.

... \hfill other properties of \texttt{DataSetSelector}.

Value

(updated) \texttt{DataSetSelector}.

Examples

\begin{verbatim}
new("DataSetSelector", size = 10)

DataSetSelector(position = "left")
setPosition(.Object = DataSetSelector(), position = "left")
\end{verbatim}

\section*{Description}

Uses the constructor to create the object with its properties or update an existing one with the setters.

Usage

\begin{verbatim}
## S4 method for signature 'GaugeArrow'
initialize(.Object, alpha = 1, axis, ...)

gaugeArrow(alpha = 1, axis, ...)

setAxis(.Object, axis = NULL, ...)

## S4 method for signature 'GaugeArrow,GaugeAxisOrCharacterOrMissing'
setAxis(.Object, axis = NULL, ...)
\end{verbatim}

Arguments

/Object\hfill \texttt{GaugeArrow}.

\textbf{alpha} \hfill numeric.

\textbf{axis} \hfill \texttt{GaugeAxis}. Axis of the arrow. You can use reference to the axis or id of the axis. If you don’t set any axis, the first axis of the chart will be used.

... \hfill other properties of \texttt{GaugeArrow}.
Value

(updated) Object of class `GaugeArrow`.

Examples

```r
# --- method initialize
define("initialize", alpha = 2)

# --- constructor
new("GaugeArrow", alpha = 10)

# -- update 'axis' property
setAxis(.Object = gaugeArrow(), id = "axis1", startValue = 0, endValue = 100, valueInterval = 10)
# equivalent to:
axis_obj <- gaugeAxis(id = "axis1", startValue = 0, endValue = 100, valueInterval = 10)
setAxis(.Object = gaugeArrow(), axis = axis_obj)
# or, if 'axis_obj' has already been added to the chart
setAxis(.Object = gaugeArrow(), axis = "axis1")
```

---

**initialize.GaugeAxis-method**

*Initializes a GaugeAxis*

Description

Uses the constructor to create the object or update an existing one with the setters.

Usage

```r
## S4 method for signature 'GaugeAxis'
initialize(.Object, axisAlpha = 1, bands, ...)

gaugeAxis(axisAlpha = 1, bands, ...)

setBands(.Object, bands)

## S4 method for signature 'GaugeAxis,list'
setBands(.Object, bands)

addBand(.Object, band = NULL, ...)

## S4 method for signature 'GaugeAxis,GaugeBandOrMissing'
addBand(.Object, band = NULL, ...)
```
**initialize,GaugeBand-method**

**Arguments**

- `.Object` **GaugeAxis**.
- `axisAlpha` numeric.
- `bands` list of **GaugeBand**. Bands are used to draw color fills between specified values.
- `...` other properties of GaugeAxis.
- `band` **GaugeBand**. Argument for method `addBand`.

**Examples**

```r
# --- method initialize
new("GaugeAxis", alpha = 1)

# -- constructor
gaugeAxis()

# -- update 'bands' at once
bands <- list(gaugeBand(startValue = 70, endValue = 90),
               gaugeBand(startValue = 40, endValue = 60))
gaugeAxis(bands = bands)

# --- add 'band' one by one
addBand(.Object = gaugeAxis(), startValue = 0, endValue = 100)
# equivalent to
gaugeBand_obj <- gaugeBand(startValue = 0, endValue = 100)
addBand(.Object = gaugeAxis(), band = gaugeBand_obj)
```

---

**initialize,GaugeBand-method**

*Initializes a GaugeBand*

**Description**

Uses the constructor to create the object or update an existing one with the setters.

**Usage**

```r
## S4 method for signature 'GaugeBand'
initialize(.Object, alpha = 1, id, ...)

gaugeBand(alpha = 1, id, ...)

setID(.Object, id)

## S4 method for signature 'GaugeBand'
setID(.Object, id)
```
Arguments

/Object  \texttt{GaugeBand} (or "GaugeBand" for initialize).

\begin{itemize}
  \item \texttt{alpha}  numeric.
  \item \texttt{id}  character.
  \item \ldots  other properties of \texttt{GaugeBand}.
\end{itemize}

Value

(\textit{updated}) \texttt{.Object} of class \texttt{GaugeBand}.

Examples

\begin{verbatim}
# --- method 'initialize'
new("GaugeBand")

# --- constructor
gaugeBand(alpha = 2, id = "band2")

# --- set the 'id'
setID(.Object = gaugeBand(), id = "1")
\end{verbatim}

Description

Uses the constructor to create the object or update an existing one with the setters.

Usage

\begin{verbatim}
$$\text{# S4 method for signature 'Guide'}$$
initialize(.Object, fillAlpha, valueAxis, value, ...)

guide(fillAlpha, valueAxis, value, ...)

setFillAlpha(.Object, fillAlpha)

$$\text{# S4 method for signature 'Guide,numeric'}$$
setFillAlpha(.Object, fillAlpha)

$$\text{# S4 method for signature 'Guide,ValueAxisOrCharacterOrMissing'}$$
setValueAxis(.Object, valueAxis = NULL, ...)
\end{verbatim}
Arguments

,Object Guide
fillAlpha numeric. specifies if a grid line is placed on the center of a cell or on the be-
"middle" This setting doesn’t work if parseDates is set to TRUE.
valueAxis ValueAxis class. As you can add guides directly to the chart, you might need to
value numeric.
... other properties of Guide.

Examples

# --- method initialize
ew("Guide", fillAlpha = 0.1, gridThickness = 1, value = 1)

# --- constructor
guide(fillAlpha = .4, value = 1)
guide(fillAlpha = .4, adjustBorderColor = TRUE, gridThickness = 1)

setValueAxis(.Object = guide(), valueAxis = valueAxis_obj)

---

initialize,Label-method

Initializes Label

Description

Uses the constructor to create the object or update an existing one with the setters.

Usage

## S4 method for signature 'Label'
initialize(.Object, text, bold, x, y, ...)

label(text, bold, x, y, ...)

setBold(.Object, bold)

## S4 method for signature 'Label,logical'
setBold(.Object, bold)

## S4 method for signature 'Label,character'
setText(.Object, text)
setX(.Object, x)

## S4 method for signature 'Label,numericOrCharacter'
setX(.Object, x)

setY(.Object, y)

## S4 method for signature 'Label,numericOrCharacter'
setY(.Object, y)

Arguments

/Object      Label.
text        character, text of a title.
bold        character, specifies if label is bold or not.
x           numeric, label’s x position.
y           numeric, label’s y position.
...          other properties of Label.

Value

(updated) .Object of class Label.

Examples

# --- method initialize
new("Label", x = 10)

# --- constructor
label(text = "bonjour")
label(text = "Male", x = "28", y = "97")

# --- update property 'bold'
setBold(.Object = label(), bold = TRUE)

# --- update 'text'
setText(.Object = label(), text = "Bonjour")

# --- update 'x'
setX(.Object = label(), x = 16)

# --- update 'y'
setY(.Object = label(), y = 16)
**Description**

Uses the constructors to create the object with its properties or update an existing one with the setters.

**Usage**

```r
## S4 method for signature 'PeriodSelector'
initialize(.Object, periods, ...)
periodSelector(periods, ...)
addPeriod(.Object, ...)
```

**Arguments**

- `.Object` PeriodSelector.
- `periods` list. Period object has 4 properties - period, count, label and selected. Possible period values are: "ss" - seconds, "mm" - minutes, "hh" - hours, "DD" - days, "MM" - months and "YYYY" - years. property "count" specifies how many periods this button will select. "label" will be displayed on a button and "selected" is a logical. which specifies if this button is selected when chart is initialized or not.
- `...` other properties of PeriodSelector.

**Value**

(updated) .Object of class PeriodSelector.

**Examples**

```r
new("PeriodSelector")
periodSelector(fillAlpha = .4, value = 1)
periodSelector(fillAlpha = .4, adjustBorderColor = TRUE, gridThickness = 1)
addPeriod(.Object = periodSelector(), period = "MM", selected = TRUE, count = 1, label= "1 month")
```
**initialize,StockEvent-method**

*Initialize a StockEvent*

**Description**

Use the constructor to create the object or update an existing one with the setters.

**Usage**

```r
## S4 method for signature 'StockEvent'
initialize(.Object, backgroundAlpha = 1, stockGraph, ...)
```

```r
stockEvent(backgroundAlpha = 1, stockGraph, ...)
```

```r
setStockGraph(.Object, stockGraph = NULL, ...)
```

```r
## S4 method for signature 'StockEvent,AmGraphOrCharacterOrMissing'
setStockGraph(.Object, stockGraph = NULL, ...)
```

**Arguments**

- `backgroundAlpha` | numeric.
- `stockGraph` | AmGraph created with stockGraph(*). This is the graph on which event will be displayed. You can use a reference to the stock graph object or id of the graph.
- `...` | other properties of StockEvent.

**Value**

(updated) argument `.Object` of class StockEvent.

**Examples**

```r
new("StockEvent")
stockEvent()
setStockGraph(.Object = stockEvent(), id = "stockGraph1", balloonText = "balloonText")
# equivalent to:
stockGraph_obj <- stockGraph(id = "stockGraph1", balloonText = "balloonText")
setStockGraph(.Object = stockEvent(), stockGraph = stockGraph_obj)
# if stockGraph_obj has already been added to the chart:
setStockGraph(.Object = stockEvent(), stockGraph = "stockGraph1")
```
**initialize,StockPanel-method**

*Initialize a StockPanel*

---

**Description**

Use the constructor to create the object or update an existing one with the setters.

**Usage**

```r
## S4 method for signature 'StockPanel'
initialize(
  .Object,
  allLabels,
  axes,
  balloon,
  categoryAxis,
  categoryField,
  chartCursor,
  chartScrollbar,
  creditsPosition,
 dataProvider,
  graphs,
  graph,
  guides,
  legend,
  theme,
  title,
  titles,
  trendLines,
  type,
  valueAxes,
  valueScrollbar,
  drawOnAxis,
  stockGraphs,
  stockLegend,
  ...
)
stockPanel(...)
panel(...)
setDrawOnAxis(.Object, valueAxis = NULL, ...)
```

## S4 method for signature 'StockPanel,ValueAxisOrCharacterOrMissing'
setDrawOnAxis(.Object, valueAxis = NULL, ...)
setStockGraphs(.Object, stockGraphs)

## S4 method for signature 'StockPanel,list'
setStockGraphs(.Object, stockGraphs)

addStockGraph(.Object, stockGraph = NULL, ...)

## S4 method for signature 'StockPanel,AmGraphOrMissing'
addStockGraph(.Object, stockGraph = NULL, ...)

setStockLegend(.Object, stockLegend = NULL, ...)

## S4 method for signature 'StockPanel,AmLegendOrMissing'
setStockLegend(.Object, stockLegend = NULL, ...)

**Arguments**

- `Object` : `StockPanel`
- `allLabels` : list of `Label`. Example of a label object, with all possible properties: `label(x = 20, y = 20, text = "this is a label", align = "left", size = 12, color = "#CC0000", alpha = 1, rotation = 0, bold = TRUE, url = "http://www.amcharts.com")`. Run `api("Label")` for more informations.
- `balloon` : `AmBalloon`. Creates the balloons (tooltips) of the chart. It follows the mouse cursor when you roll-over the data items. The framework automatically generates the instances you just have to adjust the appearance to your needs. Run `api("AmBalloon")` for more informations.
- `categoryAxis` : `CategoryAxis`. Read-only. Chart creates category axis itself. If you want to change some properties, you should get this axis from the chart and set properties to this object. Run `api("CategoryAxis")` for more informations.
- `categoryField` : character, category field name indicates the name of the field in your dataProvider object which will be used for category axis values.
- `chartCursor` : `ChartCursor`. Chart’s cursor. Run `api("ChartCursor")` for more informations.
- `chartScrollbar` : `ChartScrollbar`. Chart’s scrollbar. Run `api("ChartScrollbar")` for more informations.
- `creditsPosition` : character, specifies position of the amCharts’ website link. Allowed values are: "top-left", "top-right", "bottom-left" and "bottom-right".
- `dataProvider` : `data.frame`, containing the data.
- `graphs` : list of `AmGraph`. Creates the visualization of the data in following types: line, column, step line, smoothed line, ohlc and candlestick. Run `api("AmGraph")` for more informations.
- `graph` : `AmGraph`. Only valid for Gantt charts. Gant chart actually creates multiple graphs (separate for each segment). Properties of this graph are passed to each
of the created graphs - this allows you to control the look of segments. Run api("AmGraph") for more informations.

**guides** list of **Guide**. Instead of adding guides to the axes, you can push all of them to this array. In case guide has category or date defined, it will automatically be assigned to the category axis, otherwise to the first value axis, unless you specify a different valueAxes for the guide. Run api("Guide") for more informations.

**legend** **AmLegend**. Legend of a chart. Run api("AmLegend") for more informations.


**title** A title of a panel. Note, StockLegend should be added in order title to be displayed.

**titles** list of **Title**. Run api("Title") for more informations.

**trendLines** list of **TrendLine** objects added to the chart. You can add trend lines to a chart using this list or access already existing trend lines. Run api("TrendLine") for more informations.

**type** character, possible types are: "serial", "pie", "radar", "xy", "radar", "funnel", "gauge", "stock". See details about using argument type. (type map is in development).

**valueAxes** list of **ValueAxis**. Chart creates one value axis automatically, so if you need only one value axis, you don’t need to create it. Run api("ValueAxis") for more informations.

**valueScrollbar** **ChartScrollbar**. Value scrollbar, enables scrolling value axes.

**drawOnAxis** **ValueAxis**. Specifies on which value axis user can draw trend lines. Set drawingIconsEnabled to true if you want drawing icons to be visible. First value axis will be used if not set here. You can use a reference to the value axis object or id of value axis.

**stockGraphs** list of **AmGraph**. Each element must be have been created with stockGraph(*)

**stockLegend** list of **AmLegend**. Each element must be have been created with stockLegend(*)

... other properties of StockPanel.

**valueAxis** A **ValueAxis** for the property ‘drawOnAxis’.

**stockGraph** **AmGraph**, created with stockGraph(...). Argument for method addStockGraph.

**Value**

(value updated) **StockPanel** with given properties.

**Examples**

```
new("StockPanel", title = "Volume")

stockPanel(stockLegend = amLegend(useGraphSettings = TRUE))
panel(creditsPosition = "top-right")
panel(title = "top-right")
valueAxis_obj <- valueAxis(id = "valueAxis1")
```
setDrawOnAxis(.Object = stockPanel(), valueAxis = valueAxis_obj)
setDrawOnAxis(.Object = stockPanel(), valueAxis = "valueAxis1")
# ---
stockGraphs <- list(stockGraph(comparable = TRUE), stockGraph(comparable = FALSE))
setStockGraphs(.Object = stockPanel(), stockGraphs = stockGraphs)
stockPanel(stockGraphs = stockGraphs)
# ---
stock_panel <- addStockGraph(.Object = stockPanel(), comparable = FALSE); print(stock_panel)
# or...
stock_panel <- addStockGraph(.Object = stockPanel(), stockGraph = stockGraph(comparable = FALSE))
# ---
setStockLegend(.Object = stockPanel(), valueTextRegular = "[[value]]")
# equivalent to:
stockLegend_obj <- stockLegend(valueTextRegular = "[[value]]")
setStockLegend(.Object = stockPanel(), stockLegend = stockLegend_obj)
# ---

initialize,Title-method

Initializes A Title

Description

Uses the constructor to create the object or update an existing one with the setters.

Usage

## S4 method for signature 'Title'
initialize(.Object, text, size, ...)

title(text, size, ...)

amTitle(text, size, ...)

## S4 method for signature 'Title,character'
setText(.Object, text)

setSize(.Object, size)

## S4 method for signature 'Title,numeric'
setText(.Object, size)

Arguments

,Object Title

text character, title text.

size numeric, title size.

... other properties of Title.
Value

(updated) Title

Examples

new("Title", size = 10)
rmAmCharts::title(text = "Main", size = 10)
mTitle(text = "Main", size = 10)
mTitle(text = "Main", bold = TRUE)
setText(.Object = mTitle(), text = "Bonjour")
setSize(mTitle(), 16)

initialize, TrendLine-method

Initializes a TrendLine

Description

Uses the constructor to create the object or update an existing one with the setters.

Usage

## S4 method for signature 'TrendLine'
initialize(
  .Object,
  initialValue,
  initialXValue,
  finalValue,
  finalXValue,
  valueAxis,
  valueAxisX,
  ...
)

trendLine(
  .Object,
  initialValue,
  initialXValue,
  finalValue,
  finalXValue,
  valueAxis,
  valueAxisX,
  ...
)

setInitialValue(.Object, initialValue)
## S4 method for signature 'TrendLine,numeric'
setInitialValue(.Object, initialValue)

setInitialXValue(.Object, initialXValue)

## S4 method for signature 'TrendLine,numeric'
setFinalValue(.Object, finalValue)

setFinalXValue(.Object, finalXValue)

## S4 method for signature 'TrendLine,numeric'
setFinalValue(.Object, finalValue)

setFinalXValue(.Object, finalXValue)

## S4 method for signature 'TrendLine,ValueAxisOrCharacterOrMissing'
setValueAxis(.Object, valueAxis = NULL, ...)

setValueAxisX(.Object, valueAxisX = NULL, ...)

## S4 method for signature 'TrendLine,ValueAxisOrCharacterOrMissing'
setValueAxisX(.Object, valueAxisX = NULL, ...)

### Arguments

**.Object**  
*TrendLine.*

**initialValue**  
numeric, value from which trend line should start.

**initialXValue**  
numeric, used by XY chart only. X value from which trend line should start.

**finalValue**  
numeric, value at which trend line should end.

**finalXValue**  
numeric, used by XY chart only. X value at which trend line should end.

**valueAxis**  
*ValueAxis.* Value axis of the trend line. Will use first value axis of the chart if not set any. You can use a reference to the value axis object or id of value axis.

**valueAxisX**  
*ValueAxis.* Used by XY chart only. X axis of trend line. Will use first X axis of the chart if not set any. You can use a reference to the value axis object or id of value axis.

**...**  
other properties of TrendLine.

### Value

(possibly updated) .Object of class *TrendLine.*
initialize, ValueAxis-method

Examples

```r
new("TrendLine", initialValue = 1, finalValue = 11)

# Other example
valueAxis <- valueAxis(title = "Hello!", axisTitleOffset = 12)
new("TrendLine", valueAxis = valueAxis)

trendLine(initialValue = 1, finalValue = 11)
setInitialValue(.Object = trendLine(), initialValue = 16)
setInitialXValue(.Object = trendLine(), initialXValue = 16)
setFinalValue(.Object = trendLine(), finalValue = 16)
setFinalXValue(.Object = trendLine(), finalXValue = 16)
setValueAxis(.Object = trendLine(), id = "valueAxis-1",
              title = "Hello!", axisTitleOffset = 12)
# equivalent to:
valueAxis_obj <- valueAxis(id = "valueAxis-1", title = "Hello!", axisTitleOffset = 12)
trendLine(valueAxis = valueAxis_obj)
# or...
trendLine(valueAxis = "valueAxis-1")
# valid if and only if 'valueAxis_obj' has already been added to the chart

setValueAxisX(.Object = trendLine(), id = "valueAxisX-1",
              title = "Hello!", axisTitleOffset = 12)
# equivalent to:
valueAxisX_obj <- valueAxis(id = "valueAxisX-1", title = "Hello!", axisTitleOffset = 12)
trendLine(valueAxisX = valueAxisX_obj)
# or...
trendLine(valueAxisX = "valueAxisX-1")
# valid if and only if 'valueAxisX_obj' has already been added to the chart
```

initialize, ValueAxis-method

Initializes ValueAxis

Description

Creates a ValuesAxis or updates its properties.

Usage

```r
## S4 method for signature 'ValueAxis'
initialize(.Object, title, guides, ...)

valueAxis(...)

## S4 method for signature 'ValueAxis, character'
setTitle(.Object, title)
```
Arguments

/Object: ValueAxis.
title: character.
guides: list of Guide.
... Other properties (depend of call function)

Examples

guides <- list(guide(fillAlpha = .4), guide(fillAlpha = .5))
new("ValueAxis", title = "Hello !", gridThickness = 1, guides = guides)

valueAxis(title = "Hello !", axisTitleOffset = 12)
setTitle(.Object = valueAxis(), title = "Hello !")

Label-class Label class

Description

Creates a label on the chart which can be placed anywhere, multiple can be assigned.

Details

Run api("Label") for more information and all available properties.

Slots

bold character. Specifies if label is bold or not.
text character. Text of a title.
x numeric. X position of a label.
y numeric. Y position of a label.
listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.
otherProperties list containing other available properties not yet implemented in the package.
value numeric.

Author(s)

datastorm-open
listProperties

Description

Each S4 class implements the method to list its properties (usefull to update complex properties).

Usage

listProperties(.Object)

 Arguments

,Object any class object of the package

Value

A list containing all the chart's properties.

Examples

amChart(type = "serial")

PeriodSelector-class

Description

Defines the PeriodSelector properties.

Slots

periods list. Period object has 4 properties - period, count, label and selected. Possible period values are: "ss" - seconds, "mm" - minutes, "hh" - hours, "DD" - days, "MM" - months and "YYYY" - years. property "count" specifies how many periods this button will select. "label" will be displayed on a button and "selected" is logical. which specifies if this button is selected when chart is initialized or not.

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string. See examples for details.

otherProperties list containing other avalaible properties not yet implemented in the package.

value Object of class numeric.
Author(s)

datastorm-open

plot,AmCharts-method  PLOTTING METHOD

Description

Basic method to plot an AmChart

Usage

## S4 method for signature 'AmCharts'
plot(x, y, width = "100\%", height = NULL, background = "#ffffff", ...)

Arguments

x  AmChart
y  unused.
width  character.
height  character.B
background  character.
...  Other properties.

Details

Plots an object of class AmChart

print,AmObject-method  Visualize with print

Description

Display the object in the console.

Usage

## S4 method for signature 'AmObject'
print(x, withDetail = TRUE, ...)

Arguments

x  AmChart.
withDetail  logical, TRUE to display details.
...  Other properties.
Details
If the object possess a 'dataProvider' property, it will be hidden in the console. To see if it's correctly registered use '@dataProvider'.

Examples
```
print(new("AmChart", categoryField = "variables", type = "serial"))
print(new("AmChart", categoryField = "variables", type = "serial"), withDetail = FALSE)
```

Description
This function need the base64enc package to save image.

Usage
```
# Shiny module to export rAmCharts graphics on server-side

rAmChartsExportServerUI(id)

rAmChartsExportServer(
  input, output, session,
  list_am_graph,
  path = shiny::reactive(tempdir()),
  mode = "single",
  progress = T,
  message = "Calculation in progress",
  detail = "This may take a while..."
)
```

Arguments
```
id character, used to specify namesapce, see shiny::NS
input standard, shiny input
output standard, shiny output
session standard, shiny session
list_am_graph named list, reactive expression with all amCharts to export
  - "graph"rAmCharts object to export
  - "name"character, name of file, with ".jpg" extension
  - "width"Optionnal, character. Linked to amChartsOutput
  - "height"Optionnal, character. Linked to amChartsOutput
  - "type"Optionnal, character. Linked to amChartsOutput
```
path character, directory. tempdir() by Default
mode character, 'single' : graphics are rendered and saved one by one. 'multiple' all at same time
progress boolean, set a progress bar or not?
message character, if progress, message. Default to "Calculation in progress"
detail character, if progress, detail. Default to "This may take a while..."

Value

a reactive expression

Examples

## Not run:

# ui
rAmChartsExportServerUI("export_server_graphs")

# server
mult_amgraph <- reactive({
  if(input$goSave > 0){
    isolate({
      list(
        list(graph = amPie(data = data_pie), name = "pie.jpg", height = "200px", width = "300px"),
        list(graph = amBarplot(x = "country", y = "visits", data = data_bar, main = "example")
          setExport(), name = "bar.jpg", height = "600px"")
      )
    })
  } else {
    NULL
  }
})

callModule(rAmChartsExportServer, "export_server_graphs", mult_amgraph,
  reactive("/home/benoit/amchart_export")

## End(Not run)

---

rAmCharts-shinymodules-ts

*Shiny module to render large time-series data with live server-client aggregation*
Description

Shiny module to render large time-series data with live server-client aggregation

Usage

rAmChartsTimeSeriesUI(id, width = "100\%", height = "400px")

rAmChartsTimeSeriesServer(
  input,
  output,
  session,
  data,
  col_date,
  col_series,
  maxPoints = shiny::reactive(600),
  tz = shiny::reactive("UTC"),
  ts = shiny::reactive(c("5 min", "10 min", "30 min", "hour", "3 hour", "12 hour",
                         "day", "week", "month", "year")),
  fun_aggr = shiny::reactive("mean"),
  treat_missing = shiny::reactive(FALSE),
  maxgap = shiny::reactive(Inf),
  type_aggr = shiny::reactive("first"),
  main = shiny::reactive(""),
  ylab = shiny::reactive(""),
  color = shiny::reactive(c("#2E2EFE", "#31B404", "#FF4000", "#AEB404")),
  type = shiny::reactive(c("line")),
  bullet = shiny::reactive(NULL),
  bulletSize = shiny::reactive(2),
  linetype = shiny::reactive(c(0, 5, 10, 15, 20)),
  linewidth = shiny::reactive(c(1, 1, 1, 1, 1)),
  fillAlphas = shiny::reactive(0),
  precision = shiny::reactive(1),
  connect = shiny::reactive(FALSE),
  export = shiny::reactive(FALSE),
  legend = shiny::reactive(TRUE),
  legendPosition = shiny::reactive("bottom"),
  legendHidden = shiny::reactive(FALSE),
  ZoomButton = shiny::reactive(data.frame(Unit = "MAX", multiple = 1, label = "All")),
  ZoomButtonPosition = shiny::reactive("bottom"),
  periodFieldsSelection = shiny::reactive(FALSE),
  scrollbar = shiny::reactive(TRUE),
  scrollbarPosition = shiny::reactive("bottom"),
  scrollbarHeight = shiny::reactive(40),
  scrollbarGraph = shiny::reactive(NULL),
  cursor = shiny::reactive(TRUE),
  cursorValueBalloonsEnabled = shiny::reactive(TRUE),
  creditsPosition = shiny::reactive("top-right"),
  group = shiny::reactive(NULL),
  ...)
dataDateFormat = shiny::reactive("YYYY-MM-DD JJ:NN:ss"),
categoryBalloonDateFormats = shiny::reactive(list(list(period = "YYYY", format = "YYYY"),
  list(period = "MM", format = "YYYY-MM"),
  list(period = "WW", format = "YYYY-MM-DD"),
  list(period = "DD", format = "YYYY-MM-DD"),
  list(period = "hh", format = "YYYY-MM-DD JJ:NN"),
  list(period = "mm", format = "YYYY-MM-DD JJ:NN"),
  list(period = "ss", format = "YYYY-MM-DD JJ:NN:ss"),
  list(period = "fff", format = "YYYY-MM-DD JJ:NN:ss"))),
dateFormats = shiny::reactive(list(list(period = "YYYY", format = "YYYY"),
  list(period = "MM", format = "MMM"),
  list(period = "WW", format = "MMM DD"),
  list(period = "DD", format = "MMM DD"),
  list(period = "hh", format = "JJ:NN"),
  list(period = "mm", format = "JJ:NN"),
  list(period = "ss", format = "JJ:NN:ss"),
  list(period = "fff", format = "JJ:NN:ss")))
)

Arguments

id character, used to specify namespace, see shiny::NS
width character, the width of the chart container. For amChartsOutput.
height character, the height of the chart container. For amChartsOutput.
input standard, shiny input
output standard, shiny output
session standard, shiny session
data : data.frame to transform.
col_date Date column name, default to "date". Must be "POSIXct"
col_series Column name of quantitative variable(s) to be transformed. Default to setdiff(colnames(data), "date")
maxPoints : Maximal number of rows in results
tz : Timezone of result. Default to "UTC".
ts All enabled aggregation. Default to c("5 min", "10 min", "30 min", "hour",
  "3 hour", "12 hour", "day", "week", "month", "year"). Can be a number, in
  seconds, or a character string containing one of "min", "hour", "day".... This
  can optionally be preceded by a positive integer and a space
fun_aggr : Aggregation function to use ("min", "max", "sum", "mean", "first", "last").
  Default to "mean".
treat_missing : Boolean. Default to FALSE Whether or not to interpolate missing values ? see
  na.approx
maxgap When interpolate missing values with na.approx. Maximum number of con-
  secutive NAs to fill. Default to Inf.
type_aggr character Type of aggregation
  • "first" : Date/Time result is equal to minimum of sequence, and this mini-
    mum is included in aggregation
  • "last" : Date/Time result is equal to maximum of sequence, and this maxi-
    mum is included in aggregation
main character, title.
ylab character, value axis label.
color character, color of series (in hexadecimal).
type character, Type of graph. Possible values are: "line" (default), "column", "step", "smoothedLine"
bullet character, point shape. Possible values are: "diamond", "square", "bubble", "yError", "xError", "round", "triangleLeft", "triangleRight", "triangleUp"
bulletSize numeric, size of bullet.
linetype numeric, line type, 0: solid, number: dashed length
linewidth numeric, line width.
fillAlphas numeric, fill. Between 0 (no fill) to 1.
precision numeric, default set to 1.
connect logical, default set to FALSE. Specifies whether to connect data points if data is missing.
export logical, default set to FALSE. TRUE to display export feature.
legend logical, enabled or not legend? Default to TRUE.
legendPosition character, legend position. Possible values are: "left", "right", "bottom", "top"
legendHidden logical, hide some series on rendering? Default to FALSE
ZoomButton data.frame, 3 or 4 columns:
  • "Unit": Character. Times unit. 'ss', 'mm', 'hh', 'DD', 'MM', 'YYYY'
  • "multiple": Numeric. multiple*unit
  • "label": Character. button's label
  • "selected": Boolean. Optional. To set initial selection. (One TRUE, others FALSE)
ZoomButtonPosition character, zoom button position. Possible values are: "left", "right", "bottom", "top"
periodFieldsSelection boolean, using zoom button, add also two fields to select period?
scrollbar boolean, enabled or not scrollbar? Default to TRUE.
scrollbarPosition character, scrollbar position. Possible values are: "left", "right", "bottom", "top"
scrollbarHeight numeric, height of scroll bar. Default: 40.
scrollbarGraph character, name of serie (column) to print in scrollbar. Default to NULL.
cursor boolean, enabled or not cursor? Default to TRUE.
cursorValueBalloonsEnabled boolean, if cursor, enabled or not balloons on cursor? Default to TRUE.
creditsPosition character, credits position. Possible values are: "top-right", "top-left", "bottom-right", "bottom-left"
group character, like in dygraphs, for synchronization in shiny or rmarkdown.
dataDateFormat character Data date format. Default to 'YYYY-MM-DD JJ:NN:ss'. See `amTimeSeries`.
categoryBalloonDateFormats list Date format objects for chart cursor. See `amTimeSeries`.
dateFormats list Date format objects for x-axis. See `amTimeSeries`.

Value

a reactive expression with aggregate data and ts

Examples

```r
## Not run:

library(shiny)
library(rAmCharts)

# number of points
n <- 1000000
data <- data.frame(date = seq(c(ISOdate(1999,12,31)), by = "5 min", length.out = n),
                  value = rnorm(n, 100, 50))

# maximum of points in javascript
max_points <- 1000

# Call module in UI
ui <- fluidPage(
  rAmChartsTimeSeriesUI("ts_1", height = "600px"),
  h4(textOutput("ts"))
)

# Define server
server <- function(input, output) {
  # Call module in server
  res <- callModule(rAmChartsTimeSeriesServer, "ts_1", reactive(data), reactive("date"),
                   reactive("value"), maxPoints = shiny::reactive(max_points),
                   main = reactive("Example of rAmChartsTimeSeries module"),
                   color = reactive("red"), periodFieldsSelection = reactive(TRUE))

  # show module return and print ts
  output$ts <- renderText({
    print(res())
    paste0("Current ts : ", res()$ts)
  })
}

# Run the application
shinyApp(ui = ui, server = server)
```
### renderAmCharts

**Description**

Widget output function to use in Shiny.

**Usage**

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

**Arguments**

- `expr`: an expression that generates an HTML widget.
- `env`: the environment in which `expr` must be evaluated.
- `quoted`: is `expr` a quoted expression (with `quote()`). This is useful if you want to save an expression into variable.

---

### runExamples

**Description**

See some examples in a shiny web application. Both 'am' functions and basic functions are illustrated.

**Usage**

```r
runExamples()
```

**Examples**

```r
## Not run:
if (interactive()) runExamples()

## End(Not run)
```
**setExport**

*Setters for AmChart and AmStockChart.*

**Description**

These methods can be used both for AmChart and AmStockChart. There are general for some first-level properties.

**Usage**

```r
setExport(.Object, enabled = TRUE, ...)
## S4 method for signature 'AmCharts,logicalOrMissing'
setExport(.Object, enabled = TRUE, ...)

setResponsive(.Object, enabled = TRUE, ...)
## S4 method for signature 'AmCharts,logicalOrMissing'
setResponsive(.Object, enabled = TRUE, ...)
```

**Arguments**

- `.Object` *AmChart or AmStockChart.*
- `enabled` logical, TRUE to display the export button.
- `...` Other properties that can be used depending on the setter.

**Examples**

```r
# Dummy examples
setExport(amPlot(1:10))
setExport(amStockChart())

# Dummy examples
setResponsive(amSerialChart())
setResponsive(amStockChart())
```
show,AmChart-method

Visualize AmStockChart with show

Description
Display the object in the console.

Usage
## S4 method for signature 'AmChart'
show(object)

Arguments
object AmChart.

Value
If the object has a valid type, it will plot the chart. If not the method will trigger the method 'print'.

show,AmObject-method

Visualize with show

Description
Display the object in the console.

Usage
## S4 method for signature 'AmObject'
show(object)

Arguments
object AmObject.

Examples
library(pipeR)
amPieChart(valueField = "value", titleField = "key", backgroundColor = "#7870E8") %>>% setDataProvider(data.frame(key = c("FR", "US"), value = c(20,10))) %>>% setExport(position = "bottom-left")
show,AmStockChart-method

Visualize AmStockChart with show

Description
Display the object in the console.

Usage

```r
## S4 method for signature 'AmStockChart'
show(object)
```

Arguments

- `object` AmStockChart.

Value

If the object has a valid type, it will plot the chart. If not the method will trigger the method 'print'.

StockEvent-class

StockEvent class

Description

StockEvent is an object which holds information about event (bullet). Values from StockEventsSettings will be used if not set. Stock event bullet’s size depends on it’s graphs fontSize. When user rolls-over, clicks or rolls-out of the event bullet, AmStockChart dispatches events.

Details

Run `api("StockEvent")` for more informations.

Slots

- `stockGraph` AmGraph containing properties of stockGraph. This is the graph on which event will be displayed. You can use a reference to the stock graph object or id of the graph.
- `listeners` list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string.
- `otherProperties` list containing other avalaible properties not yet implemented in the package.
- `value` numeric.

Author(s)

datastorm-open
stockGraph

Constructor for a stockGraph (class AmGraph)

Description

Constructor used for AmStockChart

Usage

stockGraph(animationPlayed = FALSE, balloonText, title, type, valueField, ...)

Arguments

- animationPlayed: logical.
- balloonText: character. Balloon text. You can use tags like [[value]], [[description]], [[percents]], [[open]], [[category]] or any other field name from your data provider. HTML tags can also be used.
- title: character. Graph title.
- type: character. Type of the graph. Possible values are: "line", "column", "step", "smoothedLine", "candlestick", "ohlc". XY and Radar charts can only display "line" otherArguments graphs.
- valueField: character. Name of the value field in your dataProvider.
- ...: Other properties

Value

An object of class AmGraph.

Examples

```r
# --- constructor
stockGraph(balloonText = "balloonText", valueField = "value", animationPlayed = TRUE)
```

stockLegend

Constructor for StockLegend.

Description

This method is used for AmStockChart.

Usage

stockLegend(useGraphSettings, valueTextComparing = "[[percents.value]]\%", ...)
StockPanel-class

Arguments

useGraphSettings

logical Legend markers can mirror graph’s settings, displaying a line and a
real bullet as in the graph itself. Set this property to TRUE if you want to enable
this feature.

valueTextComparing

character

... Properties of AmLegend. See http://docs.amcharts.com/3/javascriptstockchart/
StockLegend

Value

An AmLegend object

Examples

stockLegend(useGraphSettings = TRUE)

Description

StockPanel class creates stock panels (charts). AmStockChart can have multiple Stock panels.

Details

Run api("StockPanel") for more information and all available properties.

Fields

drawOnAxis ValueAxis. Specifies on which value axis user can draw trend lines. Set drawingIcon-
sEnabled to TRUE if you want icons to be visible. First value axis will be used if not set here.
You can use a reference to the value axis object or id of value axis.

stockGraphs list. Each element must be have been created with stockGraph(*)
stockLegend list. Each element must be have been created with stockLegend(*)

allLabels list of Label. Example of label object, with all possible properties: label(x = 20, y =
20, text = "this is label", align = "left", size = 12, color = 
#CC0000", alpha = 1, rotation = 0,
bold = TRUE, url = "http://www.amcharts.com"). Run api("Label") for more informations.

arrows list of GaugeArrow. Only valid for gauge charts. Run api("GaugeArrow") for more
informations.

axes list of GaugeAxis properties. Only valid for gauge charts. Run api("GaugeAxis") for
more informations.
balloon *AmBalloon* Creates the balloons ( tooltips) of the chart. It follows the mouse cursor when you roll-over the data items. The framework automatically generates the instances you just have to adjust the appearance to your needs. Run `api("AmBalloon")` for more informations.

categoryAxis *CategoryAxis*. Read-only. Chart creates category axis itself. If you want to change some properties, you should get this axis from the chart and set properties to this object.

categoryField character. Category field name indicates the chart the name of the field in your dataProvider object which will be used for category axis values.

ChartCursor *ChartCursor*. Cursor of a chart. Run `api("ChartCursor")` for more informations.

ChartScrollbar *ChartScrollbar*. Chart's scrollbar. Run `api("ChartScrollbar")` for more informations.

creditsPosition character, specifies position of the amCharts' website link. Allowed values are: "top-left", "top-right", "bottom-left" and "bottom-right".

dataProvider data.frame, containing the data.

graphs list of *AmGraph*. Creates the visualization of the data in following types: line, column, step line, smoothed line, olhc and candlestick.

graph *AmGraph*. Only valid for Gantt charts. Gant chart actually creates multiple graphs (separate for each segment). Properties of this graph are passed to each of the created graphs - this allows you to control the look of segments. Run `api("AmGraph")` for more informations.

guides list of *Guide*. Instead of adding guides to the axes, you can push all of them to this array. In case guide has category or date defined, it automatically will be assigned to the category axis. Otherwise to first value axis, unless you specify a different valueAxes for the guide. Run `api("Guide")` for more informations.

legend *AmLegend*. Chart's legend. Run `api("AmLegend")` for more informations.

segmentsField character. Segments field in your data provider. Only valid for Gantt Charts.

subChartProperties list. Only valid for Drilldown charts.


titles list of *Title*. Run `api("Title")` for more informations.

trendLines list of *TrendLine* objects added to a chart. You can add trend lines to a chart using this list or access already existing trend lines.

type character. Possible types are: "serial", "pie", "radar", "xy", "radar", "funnel", "gauge", "stock". See details about using argument type. (type map is in development).

valueAxes list of *ValueAxis*. Chart creates one value axis automatically, so if you need only one value axis, you don’t need to create it. Run `api("ValueAxis")` for more informations.

valueAxis *ValueAxis*. Only valid for Gantt Charts. Set it’s type to "date" if your data is date or time based.

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string. Run `runShinyExamples()` for examples.

otherProperties list containing other available properties not yet implemented in the package.

value numeric.

**Author(s)**

datastorm-open
Title-class

Title class

Description

Creates a title on above the chart, multiple can be assigned.

Details

Run api("Title") for more informations and all available properties.

Slots

text character, title's text.
size numeric, title's size.
listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string. See examples for details.
otherProperties list containing other available properties not yet implemented in the package.
value numeric.

Author(s)

datastorm-open

TrendLine-class

TrendLine class

Description

Creates a trendline for amSerialChart and amXYChart charts which indicates the trend of your data or covers some different purposes. Multiple can be assigned.

Details

Run api("TrendLine") for more information and all available properties.

Slots

finalValue numeric. Value at which trend line should end.
finalXValue numeric. Used by XY chart only. X value at which trend line should end.
initialValue numeric. Value from which trend line should start.
initialXValue numeric. Used by XY chart only. X value from which trend line should start.
valueAxis ValueAxis. Value axis of the trend line. Will use first value axis of the chart if not set any. You can use a reference to the value axis object or id of value axis.
valueAxisX `ValueAxis`. Used by XY chart only. X axis of trend line. Will use first X axis of the chart if not set any. You can use a reference to the value axis object or id of value axis.

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string. See examples for details.

otherProperties list, containing other available properties.

value numeric.

Author(s)

datastorm-open

---

ValueAxis-class  `ValueAxis` class

Description

Extension for ValueAxis to create an axis for `amSerialChart`, `amRadarChart`, `amXYChart` charts, multiple can be assigned. Gets automatically populated, one for `amSerialChart` and two for `amXYChart` charts, if none has been specified.

Details

Run `api("ValueAxis")` for more information and all available properties.

Slots

title character. Title of the axis.

guides list.

listeners list containing the listeners to add to the object. The list must be named as in the official API. Each element must be a character string. See examples for details.

otherProperties list containing other available properties not yet implemented in the package.

value numeric. Guides belonging to this axis. Use addGuide method

Author(s)

datastorm-open
Index

*Topic datasets
   data_AirPassengers, 62
data_bar, 63
data_candleStick1, 63
data_candleStick2, 64
data_fbar, 64
data_funnel, 65
data_gantt, 65
data_gbar, 66
data_gdp, 66
data_mekko, 67
data_pie, 67
data_radar, 68
data_stock1, 68
data_stock_2, 69
data_stock_3, 69
data_waterfall, 70
data_wind, 70
add_animate_dependency, 7
add_dataLoader_dependency, 8
add_export_dependency, 8
add_responsive_dependency, 9
add_theme_dependency, 9
addArrow (initialize, AmChart-method), 78
addArrow, AmChart, GaugeArrowOrMissing-method (initialize, AmChart-method), 78
addAxe (initialize, AmChart-method), 78
addAxe, AmChart, GaugeAxisOrMissing-method (initialize, AmChart-method), 78
addAxis (initialize, AmChart-method), 78
addAxis, AmChart, GaugeAxisOrMissing-method (initialize, AmChart-method), 78
addBand (initialize, GaugeAxis-method), 106
addBand, GaugeAxis, GaugeBandOrMissing-method (initialize, GaugeAxis-method), 106
addComparedDataSet (initialize, AmStockChart-method), 94
addComparedDataSet, AmStockChart, DataSetOrMissing-method (initialize, AmStockChart-method), 94
addDataSet (initialize, AmStockChart-method), 94
addDataSet, AmStockChart, DataSetOrMissing-method (initialize, AmStockChart-method), 94
addFieldMapping (initialize, DataSet-method), 102
addFieldMapping, DataSet-method (initialize, DataSet-method), 102
addGraph (initialize, AmChart-method), 78
addGraph, AmChart, AmGraphOrMissing-method (initialize, AmChart-method), 78
addGuide (Generics functions), 72
addLabel (initialize, AmChart-method), 78
addLabel, AmChart, LabelOrMissing-method (initialize, AmChart-method), 78
addListener, 5
addListener, AmObject, character, character-method (addListener), 5
addPanel (initialize, AmStockChart-method), 94
addPanel, AmStockChart, StockPanelOrMissing-method (initialize, AmStockChart-method), 94
addPeriod (initialize, PeriodSelector-method), 111
addPeriod, PeriodSelector-method
  (initialize, PeriodSelector-method), 111
addSegment (initialize, AmChart-method), 78
addSegment, AmChart, numeric-method
  (initialize, AmChart-method), 78
addStockEvent
  (initialize, DataSet-method), 102
addStockEvent, DataSet, StockEventOrMissing-method
  (initialize, DataSet-method), 102
addStockGraph
  (initialize, StockPanel-method), 113
addStockGraph, StockPanel, AmGraphOrMissing-method
  (initialize, StockPanel-method), 113
addSubData (initialize, AmChart-method), 78
addSubData, AmChart, numeric-method
  (initialize, AmChart-method), 78
addTitle (initialize, AmChart-method), 78
addTitle, AmChart, TitleOrMissing-method
  (initialize, AmChart-method), 78
addTrendLine
  (initialize, AmChart-method), 78
addTrendLine, AmChart, TrendLineOrMissing-method
  (initialize, AmChart-method), 78
addValueAxes
  (initialize, AmChart-method), 78
addValueAxes, AmChart, ValueAxisOrMissing-method
  (initialize, AmChart-method), 78
addValueAxis
  (initialize, AmChart-method), 78
addValueAxis, AmChart, ValueAxisOrMissing-method
  (initialize, AmChart-method), 78
amAngularGauge, 10, 11, 14, 17, 19, 21, 26,
  28, 31, 34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57
amAngularGaugeChart
  (initialize, AmChart-method), 78
AmBalloon, 22, 48, 73, 77, 78, 84, 85, 96, 97,
  114, 135
amBalloon
  (initialize, AmBalloon-method), 77
AmBalloon-class, 11
amBarplot, 11, 12, 14, 17, 19, 21, 26, 28, 31,
  34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57
amBoxplot, 11, 14, 15, 17, 19, 21, 26, 28, 31,
  34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57
amBullet, 11, 14, 17, 18, 19, 21, 26, 28, 31,
  34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57
amChartsAPI, 11, 14, 17, 19, 21, 24, 26, 28,
  31, 34, 36, 38, 40, 43, 45, 47, 49, 54, 56, 57
amChartsOutput, 24, 123
amFloatingBar, 11, 14, 17, 19, 21, 25, 26, 28,
  31, 34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57
amFunnel, 11, 14, 17, 19, 21, 26, 27, 28, 31,
  34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57
amGanttChart
  (initialize, AmChart-method), 78
AmGraph, 23, 73, 84, 85, 92, 102, 112, 114,
  115, 132, 133, 135
amGraph (initialize, AmGraph-method), 91
AmGraph-class, 29
amHist, 11, 14, 17, 19, 21, 26, 28, 30, 31, 34,
  36, 38, 40, 43, 45, 47, 50, 54, 56, 57
amLegend, 23, 85, 93, 115, 134, 135
amLegend (initialize, AmLegend-method), 93
AmLegend-class, 32
amLines, 32
amMekko, 11, 14, 17, 19, 21, 26, 28, 31, 33, 34,
  36, 38, 40, 43, 45, 47, 50, 54, 56, 57
AmObject, 6, 12, 73, 131
AmObject-class, 35
amOHLC, 11, 14, 17, 19, 21, 26, 28, 31, 34, 35,
  36, 38, 40, 43, 45, 47, 50, 54, 56, 57
<table>
<thead>
<tr>
<th>Class/Method</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>amOptions</td>
<td>11, 13, 14, 16, 17, 19, 21, 26, 28, 30, 31, 34, 36, 37, 38, 40, 42, 43, 45, 47, 50, 54, 56, 57</td>
</tr>
<tr>
<td>amPie</td>
<td>11, 14, 17, 19, 21, 26, 28, 31, 34, 36, 38, 39, 40, 43, 45, 47, 50, 54, 56, 57</td>
</tr>
<tr>
<td>amPieChart (initialize, AmChart-method)</td>
<td>78</td>
</tr>
<tr>
<td>amPlot</td>
<td>11, 14, 17, 19, 21, 26, 28, 31, 34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57</td>
</tr>
<tr>
<td>amRadarChart (initialize, AmChart-method)</td>
<td>78</td>
</tr>
<tr>
<td>amSerialChart (initialize, AmChart-method)</td>
<td>78</td>
</tr>
<tr>
<td>amSolidGauge</td>
<td>11, 14, 17, 19, 21, 26, 28, 31, 34, 36, 38, 40, 43, 45, 47, 49, 50, 54, 56, 57</td>
</tr>
<tr>
<td>AmStockChart.</td>
<td>61, 96, 97, 130, 132, 133</td>
</tr>
<tr>
<td>AmStockChart-class</td>
<td>48</td>
</tr>
<tr>
<td>amStockMultiSet</td>
<td>11, 14, 17, 19, 21, 26, 28, 31, 34, 36, 38, 40, 43, 45, 47, 49, 50, 54, 56, 57</td>
</tr>
<tr>
<td>amTimeSeries</td>
<td>11, 14, 17, 19, 21, 26, 28, 31, 34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57, 128</td>
</tr>
<tr>
<td>amTitle (initialize, Title-method)</td>
<td>116</td>
</tr>
<tr>
<td>amWaterfall (initialize, Title-method)</td>
<td>116</td>
</tr>
<tr>
<td>amWind</td>
<td>11, 14, 17, 19, 21, 26, 28, 31, 34, 36, 38, 40, 43, 45, 47, 50, 54, 56, 57, 128</td>
</tr>
<tr>
<td>amXYChart (initialize, AmChart-method)</td>
<td>78</td>
</tr>
<tr>
<td>api</td>
<td>58</td>
</tr>
<tr>
<td>AxisBase.</td>
<td>58</td>
</tr>
<tr>
<td>CategoryAxis.</td>
<td>22, 84, 99, 114, 135</td>
</tr>
<tr>
<td>CategoryAxis (initialize, CategoryAxis-method)</td>
<td>99</td>
</tr>
<tr>
<td>CategoryAxis-class</td>
<td>59</td>
</tr>
<tr>
<td>ChartCursor.</td>
<td>22, 84, 100, 101, 114, 135</td>
</tr>
<tr>
<td>chartCursor (initialize, ChartCursor-method)</td>
<td>100</td>
</tr>
<tr>
<td>chartScrollbar</td>
<td>22, 23, 84, 85, 97, 102, 114, 115, 135</td>
</tr>
<tr>
<td>chartScrollbar (initialize, ChartScrollbar-method)</td>
<td>101</td>
</tr>
<tr>
<td>chartScrollbar-class</td>
<td>60</td>
</tr>
<tr>
<td>chartScrollbarSettings</td>
<td>60</td>
</tr>
<tr>
<td>controlShinyPlot</td>
<td>61</td>
</tr>
<tr>
<td>data_AirPassengers</td>
<td>62</td>
</tr>
<tr>
<td>data_bar</td>
<td>13, 63</td>
</tr>
<tr>
<td>data_candleStick1</td>
<td>20, 63</td>
</tr>
<tr>
<td>data_candleStick2</td>
<td>20, 64</td>
</tr>
<tr>
<td>data_fbar</td>
<td>25, 64</td>
</tr>
<tr>
<td>data_funnel</td>
<td>28, 65</td>
</tr>
<tr>
<td>data_gantt</td>
<td>65</td>
</tr>
<tr>
<td>data_gbar</td>
<td>13, 66</td>
</tr>
<tr>
<td>data_gdp</td>
<td>66</td>
</tr>
<tr>
<td>data_mekko</td>
<td>34, 67</td>
</tr>
<tr>
<td>data_pie</td>
<td>40, 67</td>
</tr>
<tr>
<td>data_radar</td>
<td>45, 68</td>
</tr>
<tr>
<td>data_stock1</td>
<td>68</td>
</tr>
<tr>
<td>data_stock_2</td>
<td>69</td>
</tr>
<tr>
<td>data_stock_3</td>
<td>69</td>
</tr>
<tr>
<td>data_waterfall</td>
<td>56, 70</td>
</tr>
<tr>
<td>data_wind</td>
<td>57, 70</td>
</tr>
<tr>
<td>DataSet.</td>
<td>48, 96, 97, 103, 104</td>
</tr>
<tr>
<td>dataSet (initialize, DataSet-method)</td>
<td>102</td>
</tr>
<tr>
<td>DataSet-class</td>
<td>61</td>
</tr>
<tr>
<td>DataSetSelector</td>
<td>48, 96, 105</td>
</tr>
<tr>
<td>dataSetSelector (initialize, DataSetSelector-method)</td>
<td>104</td>
</tr>
<tr>
<td>DataSetSelector-class</td>
<td>62</td>
</tr>
<tr>
<td>GaugeArrow.</td>
<td>22, 84, 85, 105, 106, 134</td>
</tr>
<tr>
<td>gaugeArrow (initialize, GaugeArrow-method)</td>
<td>105</td>
</tr>
<tr>
<td>GaugeArrow-class</td>
<td>71</td>
</tr>
<tr>
<td>GaugeAxis.</td>
<td>22, 71, 84, 85, 105, 107, 114, 134</td>
</tr>
<tr>
<td>gaugeAxis (initialize, GaugeAxis-method)</td>
<td>106</td>
</tr>
<tr>
<td>GaugeAxis-class</td>
<td>71</td>
</tr>
</tbody>
</table>
GaugeBand, 71, 107, 108

gaugeBand
  (initialize,GaugeBand-method), 107
GaugeBand-class, 72

Generics functions, 72

getCurrentStockData, 74

getTransformTS, 75

graph (initialize,AmGraph-method), 91

Guide, 23, 73, 85, 99, 109, 115, 120, 135

guide (initialize,Guide-method), 108

Guide-class, 76

initialize,AmBalloon-method, 77
initialize,AmChart-method, 78
initialize,AmGraph-method, 91
initialize,AmLegend-method, 93
initialize,AmStockChart-method, 94
initialize,CategoryAxis-method, 99
initialize,ChartCursor-method, 100
initialize,ChartScrollbar-method, 101
initialize,DataSet-method, 102
initialize,DataSetSelector-method, 104
initialize,GaugeArrow-method, 105
initialize,GaugeAxis-method, 106
initialize,GaugeBand-method, 107
initialize,Guide-method, 108
initialize,Label-method, 109
initialize,PeriodSelector-method, 111
initialize,StockEvent-method, 112
initialize,StockPanel-method, 113
initialize,Title-method, 116
initialize,TrendLine-method, 117
initialize,ValueAxis-method, 119

Label, 22, 84, 85, 110, 114, 134

label (initialize,Label-method), 109
Label-class, 120

legend (initialize,AmLegend-method), 93
listProperties, 121
listProperties,AmObject-method (listProperties), 121

NS, 123, 126

panel (initialize,StockPanel-method), 113

PeriodSelector, 48, 96, 111

periodSelector
  (initialize,PeriodSelector-method), 111

PeriodSelector-class, 121
plot,AmCharts-method, 122
print,AmObject-method, 122

rAmCharts-shinymodules, 123
rAmCharts-shinymodules-ts, 124
rAmChartsExportServer
  (rAmCharts-shinymodules), 123
rAmChartsExportServerUI
  (rAmCharts-shinymodules), 123
rAmChartsTimeSeriesServer
  (rAmCharts-shinymodules-ts), 124
rAmChartsTimeSeriesUI
  (rAmCharts-shinymodules-ts), 124
renderAmCharts, 129
resetProperties (addListener), 5
resetProperties,AmObject-method (addListener), 5
runExamples, 129

setAdjustBorderColor
  (initialize,AmBalloon-method), 77

setAdjustBorderColor,AmBalloon,logical-method
  (initialize,AmBalloon-method), 77

setAllLabels
  (initialize,AmChart-method), 78
setAllLabels,AmChart,list-method
  (initialize,AmChart-method), 78

setArrows (initialize,AmChart-method), 78

setArrows,AmChart-method
  (initialize,AmChart-method), 78
setAxes (initialize,AmChart-method), 78

setAxes,AmChart,list-method
  (initialize,AmChart-method), 78

setAxis (initialize,GaugeArrow-method), 105

setAxis,GaugeArrow,GaugeAxisOrCharacterOrMissing-method
  (initialize,GaugeArrow-method), 105

setBalloon (Generics functions), 72
setBalloon, AmChart, AmBalloonOrMissing-method
(initialize, AmChart-method), 78

setBalloon, AmStockChart, AmBalloonOrMissing-method
(initialize, AmStockChart-method), 94

setColor (initialize, AmBalloon-method), 77

setColor, AmBalloon, character-method
(initialize, AmBalloon-method), 77

setComparedDataSets
(initialize, AmStockChart-method), 94

setDrawOnAxis
(initialize, StockPanel-method),

setLabelColor
(initialize, Label-method), 109

setLabelColor, Label, logical-method
(initialize, Label-method), 109

setCornerRadius
(initialize, AmBalloon-method), 77

setCornerRadius, AmBalloon, numeric-method
(initialize, AmBalloon-method), 77

setCreditsPosition
(initialize, AmChart-method), 78

setCreditsPosition, AmChart, character-method
(initialize, AmChart-method), 78

setChartScrollbar
(initialize, AmChart-method), 78

setChartScrollbar, AmChart, ChartScrollbarOrMissing-method
(initialize, AmChart-method), 78

setDataLoader
(initialize, AmChart-method), 78

setDataLoader, AmChart, character, character-method
(initialize, AmChart-method), 78

setDataProvider (Generics functions), 72

setDataProvider, AmChart, ANY, logicalOrMissing-method
(initialize, AmChart-method), 78

setDataProvider, DataSet, ANY, ANY-method
(initialize, DataSet-method),

setCornerRadius (initialize, AmBalloon-method), 77

setCornerRadius, AmBalloon, numeric-method
(initialize, AmBalloon-method), 77

setDataLoader
(initialize, AmChart-method), 78

setDataLoader, AmChart, character, character-method
(initialize, AmChart-method), 78

setDataProvider (Generics functions), 72

setDataProvider, AmChart, ANY, logicalOrMissing-method
(initialize, AmChart-method), 78

setDataProvider, DataSet, ANY, ANY-method
(initialize, DataSet-method),

setDataSets
(initialize, AmChart-method), 78

setDataSets, AmChart, character-method
(initialize, AmChart-method), 78

setDataSets
(initialize, AmChart-method), 78

setDataSets, AmStockChart-method
(initialize, AmStockChart-method), 94

setDataSets, AmStockChart-method
(initialize, AmStockChart-method), 94

setDataSetSelector
(initialize, AmChart-method), 78

setDataSetSelector, AmChart, character-method
(initialize, AmChart-method), 78

setDataSetSelector
(initialize, AmChart-method), 78

setDataSetSelector, AmChart, character-method
(initialize, AmChart-method), 78

setDrawOnAxis
(initialize, StockPanel-method),
setDrawOnAxis, StockPanel, ValueAxisOrCharacterOrMissing-method
  (initialize, StockPanel-method), 113
setEnabled
  (initialize, ChartScrollbar-method), 101
setEnabled, ChartScrollbar, logical-method
  (initialize, ChartScrollbar-method), 101
setExport, AmCharts, logicalOrMissing-method
  (setExport), 130
setExport, AmChart, AmGraphOrMissing-method
  (initialize, AmChart-method), 78
setExport
  (initialize, CategoryAxis-method), 99
setGridPosition, CategoryAxis, character-method
  (initialize, CategoryAxis-method), 99
setGridPosition
  (initialize, CategoryAxis-method), 99
setGuides
  (initialize, AmChart-method), 78
setGuides, AmChart, list-method
  (initialize, AmChart-method), 78
setID
  (initialize, GaugeBand-method), 107
setID
  (initialize, GaugeBand-method), 107
setInitialValue
  (initialize, TrendLine-method), 117
setInitialValue
  (initialize, TrendLine-method), 117
setLegend
  (initialize, AmChart-method), 78
setLegend, AmChart, AmLegendOrMissing-method
  (initialize, AmChart-method), 78
setLegendSettings
  (initialize, AmStockChart-method), 94
setLegendSettings
  (initialize, AmStockChart-method), 94
setMainDataSet
  (initialize, AmStockChart-method), 94
setMainDataSet, AmStockChart, DataSetOrMissing-method
  (initialize, AmStockChart-method), 94
setOneBalloonOnly
  (initialize, ChartCursor-method), 100
setGraphs
  (initialize, AmChart-method), 72
setGraph, AmChart, AmGraphOrMissing-method
  (initialize, AmChart-method), 78
setGraph, ChartScrollbar, AmGraphOrCharacterOrMissing-method
  (initialize, ChartScrollbar-method), 101
setGraphs
  (initialize, AmChart-method), 72
setFillAlpha
  (initialize, Guide-method), 108
setFillAlpha, Guide, numeric-method
  (initialize, Guide-method), 108
setFillColor
  (initialize, AmBalloon-method), 77
setFillColor
  (initialize, AmBalloon-method), 77
setFinalValue
  (initialize, TrendLine-method), 117
setFinalValue
  (initialize, TrendLine-method), 117
setFinalXValue
  (initialize, TrendLine-method), 117
setFinalXValue
  (initialize, TrendLine-method), 117
setLegend
  (initialize, AmChart-method), 78
setLegend, AmChart, AmLegendOrMissing-method
  (initialize, AmChart-method), 78
setLegendSettings
  (initialize, AmStockChart-method), 94
setLegendSettings
  (initialize, AmStockChart-method), 94
setMainDataSet
  (initialize, AmStockChart-method), 94
setMainDataSet, AmStockChart, DataSetOrMissing-method
  (initialize, AmStockChart-method), 94
setOneBalloonOnly
  (initialize, ChartCursor-method), 100
setOneBalloonOnly, ChartCursor, logical-method
  (initialize, ChartCursor-method), 100
setPanels
  (initialize, AmStockChart-method), 94
setPanels, AmStockChart, list-method
  (initialize, AmStockChart-method), 94
setPanelsSettings
  (initialize, AmStockChart-method), 94
setPeriodSelector
  (initialize, AmStockChart-method), 94
setPeriodSelector, AmStockChart, PeriodSelectorOrMissing-method
  (initialize, AmStockChart-method), 94
setPosition
  (initialize, DataSetSelector-method), 104
setPosition, DataSetSelector, character-method
  (initialize, DataSetSelector-method), 104
setProperties
  (addListener), 5
setProperties, AmObject-method
  (addListener), 5
setResponsive (setExport), 130
setResponsive, AmCharts, logicalOrMissing-method
  (setExport), 130
setSize (initialize, Title-method), 116
setSize, Title, numeric-method
  (initialize, Title-method), 116
setStockEvents
  (initialize, DataSet-method), 102
setStockEvents, DataSet, list-method
  (initialize, DataSet-method), 102
setStockEventsSettings
  (initialize, AmStockChart-method), 94
setStockEventsSettings, AmStockChart-method
  (initialize, AmStockChart-method),
setStockGraph
  (initialize, StockEvent-method),
setStockGraph, StockEvent, AmGraphOrCharacterOrMissing-method
  (initialize, StockEvent-method),
setStockGraphs
  (initialize, StockPanel-method), 113
setStockGraphs, StockPanel, list-method
  (initialize, StockPanel-method), 113
setStockLegend
  (initialize, StockPanel-method), 113
setStockLegend, StockPanel, AmLegendOrMissing-method
  (initialize, StockPanel-method),
setStockLegend, StockPanel, AmLegendOrMissing-method
  (initialize, StockPanel-method),
setStockLegend, StockPanel, AmLegendOrMissing-method
  (initialize, StockPanel-method),
setStockLegend, StockPanel, AmLegendOrMissing-method
  (initialize, StockPanel-method),
setStockLegend, StockPanel, AmLegendOrMissing-method
  (initialize, StockPanel-method),
setStockLegend, StockPanel, AmLegendOrMissing-method
  (initialize, StockPanel-method),
setSubChartProperties
  (initialize, AmChart-method), 78
setSubChartProperties, AmChart-method
  (initialize, AmChart-method), 78
setTheme
  (initialize, AmChart-method), 78
setTheme, AmChart, character-method
  (initialize, AmChart-method), 78
setTitle
  (initialize, AmChart-method), 78
setTitle, AmGraph, character-method
  (initialize, AmGraph-method), 91
setTitle, ValueAxis, character-method
  (initialize, ValueAxis-method), 119
setTitle, AmChart, character-method
  (initialize, AmChart-method), 78
setTitle, AmChart, list-method
  (initialize, AmChart-method), 78
testText (Generics functions), 72
testText, Label, character-method
  (initialize, Label-method), 109
testText, Title, character-method
  (initialize, Title-method), 116
testText, Title, character-method
  (initialize, Title-method), 116
setTitles (initialize, AmChart-method), 78
setTitles, AmChart, list-method
  (initialize, AmChart-method), 78
setTrendLines
  (initialize, AmChart-method), 78
setTrendLines, AmChart, list-method
  (initialize, AmChart-method), 78
setType (Generics functions), 72
setType, AmChart, character-method
  (initialize, AmChart-method), 78
setType, AmGraph, character-method
(initialize, AmGraph-method), 91

setUseGraphSettings
(initialize, AmLegend-method), 93

setUseGraphSettings, AmLegend, logical-method
(initialize, AmLegend-method), 93

setValueAxes
(initialize, AmChart-method), 78

setValueAxes, AmChart, list-method
(initialize, AmChart-method), 78

setValueAxesSettings
(initialize, AmStockChart-method), 94

setValueAxesSettings, AmStockChart-method
(initialize, AmStockChart-method), 94

setValueAxis (Generics functions), 72

setValueAxis, AmChart, ValueAxisOrMissing-method
(initialize, AmChart-method), 78

setValueAxis, Guide, ValueAxisOrCharacterOrMissing-method
(initialize, Guide-method), 108

setValueAxis, TrendLine, ValueAxisOrCharacterOrMissing-method
(initialize, TrendLine-method), 117

setValueAxisX
(initialize, TrendLine-method), 117

setValueAxisX, TrendLine, ValueAxisOrCharacterOrMissing-method
(initialize, TrendLine-method), 117

setValueField
(initialize, AmGraph-method), 91

setValueField, AmGraph, character-method
(initialize, AmGraph-method), 91

setValueLineAxis
(initialize, ChartCursor-method), 100

setValueLineAxis, ChartCursor, ValueAxisOrCharacterOrMissing-method
(initialize, ChartCursor-method), 100

setValueScrollbar
(initialize, AmChart-method), 78

setValueScrollbar, AmChart, ChartScrollbarOrMissing-method
(initialize, AmChart-method), 78

setX
(initialize, Label-method), 109

setX, Label, numericOrCharacter-method
(initialize, Label-method), 109

show, AmChart-method, 131

show, AmObject-method, 131

show, AmStockChart-method, 132

StockEvent, 61, 104, 112

stockEvent
(initialize, StockEvent-method), 112

StockEvent-class, 132

stockGraph, 133

stockLegend, 133

StockPanel, 48, 96, 97, 114, 115

stockPanel
(initialize, StockPanel-method), 113

StockPanel-class, 134

Title, 23, 85, 86, 115–117, 135

title (initialize, Title-method), 116

Title-class, 136

TrendLine, 23, 85, 86, 115, 118, 135

trendLine
(initialize, TrendLine-method), 117

TrendLine-class, 136

ValueAxis, 23, 73, 76, 85, 100, 109, 115, 118,

title (initialize, ValueAxis-method), 119

ValueAxis-class, 137