Package ‘modchart’

March 15, 2021

Title A 'shiny' Module for Creating Charts of Various Types

Version 0.5

License MIT + file LICENSE

Author Yadu Balehosur <yadu@vizualytics.com>

Maintainer Yadu Balehosur <yadu@vizualytics.com>

Description This is a 'shiny' module that encapsulates various charting options available in 'htmlwidgets', and provides options for each type of chart, a 'crosstalk' like interface for aggregate reports between 'DT' and other chart types.

NeedsCompilation no

Imports DT, RColorBrewer, collapsibleTree, dplyr, dygraphs,
highcharter, jsonlite, lazyeval, leaflet, networkD3, plotly,
rgdal, shiny, shinyBS, shinydashboard, shinydashboardPlus, sp,
sparkline, sunburstR, treemap, htmlwidgets, magrittr

Suggests modgetxl

RoxygenNote 7.1.1

Repository CRAN

Date/Publication 2021-03-15 16:00:02 UTC

R topics documented:

chart *chart*

Description

A 'shiny' module to display many types of charts available as 'htmlwidgets' with a dataframe as input

Usage

```
chart(input, output, session, g, setdrill = NULL, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`setdrill`	is the function chart will call upstream to set a drill value on a chart
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

The graph structure containing chart type and chart data is passed as input.

The graph/chart data is displayed in appropriate chart type with options to change to other chart types

See Also

See chartex for an example

chartex *modchart Examples*

Description

Use 'shiny' module 'modchart' for generating various types of charts

Details

The data for these examples are provided in the "extdata" directory of package

The location of these can be obtained via the call: system.file("extdata", "abcd.xlsx", "modchart")

Please select this location for accessing the files to run the example below

There are three data files supplied with this package to try out the charts.

They are mtcars.xlsx, airpass2.xlsx, and uspop.xlsx

mtcars.xlsx helps demonstrate multiple series in plotly

airpass2.xlsx helps to demonstrate dygraph time series chart as well as stack bar in plotly

uspop.xlsx helps to demonstrate map/leaflet chart

In addition, to demonstrate choropleth, associated shape files are provided as shapefile.xxx;

please copy these shape files into your www directory for example to work correctly

Examples

```
library(shiny)
library(modchart)
library(shinydashboard)
library(shinydashboardPlus)
app<- shinyApp(
ui= shinyUI(
 dashboardPage(skin='purple',
header=dashboardHeader(title = 'Charts Demo'),
sidebar=dashboardSidebar(sidebarMenuOutput('sidemenu')),
body=dashboardBody(uiOutput('mainbody'))
)
 ),
server=shinyServer(function(input, output, session) {
sink(file=stderr())

options(shiny.maxRequestSize=1*1024^2) # 1MB

output$xl<- renderUI({
getxlUI('server')
})
xl<- callModule(getxl, 'server')

output$charts<- renderUI({
if(length(xl$sheets) > 0) {
```

```
title<- xl$sheets[1]
if(title == 'mtcars' | title == 'airpass2')
ndim<- 2
else
ndim<- 1
nseries<- 1
g<- xl2g(xl, ndim=ndim, nseries=nseries)
callModule(chart, 'server', g)
chartUI('server', g)
}
})
output$sidemenu<- renderMenu({
m1<- menuItem( "Upload Excel", menuSubItem("Excel", tabName="xltab"))
m2<- menuItem( "Create Chart", menuSubItem("Chart", tabName="charttab"))
sidebarMenu(m1,m2)
})

output$mainbody<- renderUI({
t1<- list(); t1[[1]]<- tabItem(tabName="xltab", uiOutput("xl"))
t2<- list(); t2[[1]]<- tabItem(tabName="charttab", uiOutput("charts"))
do.call(tabItems, c(t1,t2))
})
})
)
if(interactive()) {
runApp(app)
}
```

chartUI *chartUI*

Description

User interface to display a chart

Usage

```
chartUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller's id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

`cr` *cr*

Description

cr is chart reactive that tracks user click on chart types

Usage

```
cr
```

Format

An object of class `reactivevalues` of length 3.

`ctree` *ctree*

Description

A 'shiny' module to display 'collapsibleTree' chart with options

Usage

```
ctree(input, output, session, g, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Options for 'collapsibleTree' are color and size

`ctreeUI` *ctreeUI*

Description

User interface to display 'collapsibleTree' chart type

Usage

```
ctreeUI(id, g, noopt = 0)
```

Arguments

`id` is the caller's id

`g` is the graph/chart to be charted

`noopt` is a toggle that tells chart module not to display options to change chart defaults

`df2g` *df2g*

Description

This is a utility function to create a 'graph' data structure to pass to chart module

Usage

```
df2g(title, dxy, ndim = 1, nseries = 1)
```

Arguments

`title` is the title for the chart

`dxy` is the dataframe to draw the chart from

`ndim` is the number of dimensions in the xl file; it is assumed these are in the first ndim columns of the xl

`nseries` is the number of series in the xl file; it is assumed these are in the last nseries columns of the xl

dtbl *dtbl*

Description

A ’shiny’ module to display ’DT’ chart with options

Usage

```
dtbl(input, output, session, g, setdrill = NULL, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`setdrill`	is the function to chart will call upstream to set a drill value on a chart
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Options for ’DT’ are column and table heatmaps, and ’sparklines’ on the last dimension

dtblUI *dtblUI*

Description

User interface to display ’DT’ chart type

Usage

```
dtblUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller’s id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

dyg *dyg*

Description

A 'shiny' module to display 'dygraph' chart with options

Usage

```
dyg(input, output, session, g, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Options for 'dygraph' are range selector and line fill

dygUI *dygUI*

Description

User interface to display 'dygraph' chart type

Usage

```
dygUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller's id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

map *map*

Description

A 'shiny' module to display 'leaflet' chart with options

Usage

```
map(input, output, session, g, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Options for 'leaflet' are shapes or circles on map, basemap, function to apply, color palette, fill opacity, and circle scale

mapUI *mapUI*

Description

User interface to display 'leaflet' chart type

Usage

```
mapUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller's id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

plotly	*plotly*

Description

A ’shiny’ module to display ’plot_ly’ chart with options

Usage

```
plotly(input, output, session, g, setdrill = NULL, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`setdrill`	is the function to chart will call upstream to set a drill value on a chart
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Options for ’plotly’ are provided for bar, line, scatter and pie charts

Common options are tick angle for x axis, margin width/height, vertical or horizontal orientation, color/palette

Additional option for bar chart option is stacked bar chart

Additional options for line chart option are line type, line shape, and area fill

Additional options for pie chart option are donut size and clockwise/counter-clockwise drawing

plotlyUI	*plotlyUI*

Description

User interface to display ’plot_ly’ chart type

Usage

```
plotlyUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller’s id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

sky *sky*

Description

A 'shiny' module to display 'sankey' chart with options

Usage

```
sky(input, output, session, g, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Options for 'sankey' chart are font size and node width

skyUI *skyUI*

Description

User interface to display 'sankey' chart type

Usage

```
skyUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller's id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

sunb *sunb*

Description

A 'shiny' module to display 'sunburst' chart with options

Usage

```
sunb(input, output, session, g, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Option for 'sunburst' is color palette

sunbUI *sunbUI*

Description

User interface to display 'sunburst' chart type

Usage

```
sunbUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller's id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

tree *tree*

Description

A 'shiny' module to display 'treemap' chart with options

Usage

```
tree(input, output, session, g, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

Options for treemap are: interactive or static tree, and choice of color palette

treeUI *treeUI*

Description

User interface to display 'treemap' chart type

Usage

```
treeUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller's id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

vbox *vbox*

Description

A 'shiny' module to display 'valueBox' chart with options

Usage

```
vbox(input, output, session, g, noopt = 0)
```

Arguments

`input`	is shiny input variable
`output`	is shiny output variable
`session`	is shiny session variable
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

Details

This is drawn as one standard value box, with no further options

vboxUI *vboxUI*

Description

User interface to display 'valueBox' chart type

Usage

```
vboxUI(id, g, noopt = 0)
```

Arguments

`id`	is the caller's id
`g`	is the graph/chart to be charted
`noopt`	is a toggle that tells chart module not to display options to change chart defaults

xl2g *xl2g*

Description

This is a utility function to create a 'graph' data structure to pass to chart module from an 'Excel' sheet

Usage

```
xl2g(xl, ndim = 1, nseries = 1)
```

Arguments

`xl`	has the title and data of the 'Excel' file
`ndim`	is the number of dimensions in the 'Excel' file; it is assumed these are in the first ndim columns of the xl
`nseries`	is the number of series in the 'Excel' file; it is assumed these are in the last nseries columns of the xl

Index