

Package: d3r (via r-universe)

August 27, 2024

Type Package

Title 'd3.js' Utilities for R

Version 1.1.0

Date 2023-10-01

Maintainer Kent Russell <kent.russell@timelyportfolio.com>

URL <https://github.com/timelyportfolio/d3r>

BugReports <https://github.com/timelyportfolio/d3r/issues>

Description Provides a suite of functions to help ease the use of 'd3.js' in R. These helpers include 'htmltools::htmlDependency' functions, hierarchy builders, and conversion tools for 'partykit', 'igraph', 'table', and 'data.frame' R objects into the 'JSON' that 'd3.js' expects.

License BSD_3_clause + file LICENSE

Encoding UTF-8

Imports dplyr, htmltools, tidyr (>= 0.8.3)

Suggests httr, jsonlite, listviewer, purrr, testthat

Enhances igraph, partykit, rpart, treemap, V8

RoxygenNote 7.2.3

Repository <https://timelyportfolio.r-universe.dev>

RemoteUrl <https://github.com/timelyportfolio/d3r>

RemoteRef master

RemoteSha e824baa8a01fdc3492ee287d734ebb5c120f893c

Contents

change_to_name	2
d3_dep_jetpack	2
d3_dep_v3	4
d3_dep_v4	5

d3_dep_v5	5
d3_dep_v6	6
d3_dep_v7	7
d3_igraph	8
d3_json	9
d3_nest	9
d3_party	10
d3_table	12
d3_v8	12
promote_na	15
promote_na_one	15

Index 16

change_to_name	<i>Change Column Name in Children to "name"</i>
----------------	---

Description

Change Column Name in Children to "name"

Usage

```
change_to_name(x, column = 1)
```

Arguments

x	data.frame or data.frame derivative, such as tibble
column	column to convert

Value

data.frame

d3_dep_jetpack	<i>'d3.js' Dependency for Version 4 Jetpack</i>
----------------	---

Description

d3-jetpack is a set of nifty convenience wrappers that speed up your daily work with d3.js. Must be included after d3_dep_v4(). Learn more by reading [d3-jetpack](#) or by watching this [YouTube](#).

Usage

```
d3_dep_jetpack(offline = TRUE)
```

Arguments

offline logical to specify whether to use a local copy of d3.js (TRUE) or use cdn (FALSE)

Value

htmltools::htmlDependency

See Also

Other 'd3' dependency functions: [d3_dep_v3\(\)](#), [d3_dep_v4\(\)](#), [d3_dep_v5\(\)](#), [d3_dep_v6\(\)](#), [d3_dep_v7\(\)](#)

Examples

```
## Not run:

library(d3r)
library(htmltools)

t1 <- tagList(tags$script(HTML(sprintf(
"
var x = 5;

var svg = d3.select('body')
  .append('svg');

svg.append('rect')
  .at({
    x: 100,
    y: 100,
    width: 100,
    height: 100
  })
  .st({
    fill: 'blue',
    stroke: 'purple'
  });
"
))), d3_dep_v4(), d3_dep_jetpack())
browsable(t1)

t1 <- tagList(tags$script(HTML(sprintf(
"
var svg = d3.select('body')
  .append('svg');

test_data = [{x: 250, y: 250}, {x: 300, y: 300}, {x: 300, y: 100}];

svg.appendMany(test_data, 'circle')
  .at({
```

```

        cx: function(d){return d.x},
        cy: function(d){return d.y},
        r: 50
    })
    .st({
        fill: 'purple',
        stroke: 'blue'
    });
    "
  ))) , d3_dep_v4(), d3_dep_jetpack()

  browsable(t1)

  ## End(Not run)

```

d3_dep_v3

'd3.js' Dependency for Version 3

Description

'd3.js' Dependency for Version 3

Usage

```
d3_dep_v3(offline = TRUE)
```

Arguments

`offline` logical to specify whether to use a local copy of d3.js (TRUE) or use cdn (FALSE)

Value

htmltools::htmlDependency

See Also

[d3_dep_v7](#), [d3_dep_v6](#), [d3_dep_v5](#), [d3_dep_v4](#), and [d3_dep_jetpack](#).

Other 'd3' dependency functions: [d3_dep_jetpack\(\)](#), [d3_dep_v4\(\)](#), [d3_dep_v5\(\)](#), [d3_dep_v6\(\)](#), [d3_dep_v7\(\)](#)

Examples

```

library(d3r)
library(htmltools)

tagList(d3_dep_v3())

```

d3_dep_v4	<i>'d3.js' Dependency for Version 4</i>
-----------	---

Description

'd3.js' Dependency for Version 4

Usage

```
d3_dep_v4(offline = TRUE)
```

Arguments

offline	logical to specify whether to use a local copy of d3.js (TRUE) or use cdn (FALSE)
---------	---

Value

```
htmltools::htmlDependency
```

See Also

[d3_dep_v7](#), [d3_dep_v6](#), [d3_dep_v5](#), [d3_dep_v3](#), and [d3_dep_jetpack](#).

Other 'd3' dependency functions: [d3_dep_jetpack\(\)](#), [d3_dep_v3\(\)](#), [d3_dep_v5\(\)](#), [d3_dep_v6\(\)](#), [d3_dep_v7\(\)](#)

Examples

```
library(d3r)
library(htmltools)

tagList(d3_dep_v4())
```

d3_dep_v5	<i>'d3.js' Dependency for Version 5</i>
-----------	---

Description

'd3.js' Dependency for Version 5

Usage

```
d3_dep_v5(offline = TRUE)
```

Arguments

`offline` logical to specify whether to use a local copy of d3.js (TRUE) or use cdn (FALSE)

Value

`htmltools::htmlDependency`

See Also

[d3_dep_v7](#), [d3_dep_v6](#), [d3_dep_v4](#), [d3_dep_v3](#), and [d3_dep_jetpack](#).

Other 'd3' dependency functions: [d3_dep_jetpack\(\)](#), [d3_dep_v3\(\)](#), [d3_dep_v4\(\)](#), [d3_dep_v6\(\)](#), [d3_dep_v7\(\)](#)

Examples

```
library(d3r)
library(htmltools)

tagList(d3_dep_v5())
```

d3_dep_v6

'd3.js' Dependency for Version 6

Description

'd3.js' Dependency for Version 6

Usage

```
d3_dep_v6(offline = TRUE)
```

Arguments

`offline` logical to specify whether to use a local copy of d3.js (TRUE) or use cdn (FALSE)

Value

`htmltools::htmlDependency`

See Also

[d3_dep_v7](#), [d3_dep_v5](#), [d3_dep_v4](#), [d3_dep_v3](#), and [d3_dep_jetpack](#).

Other 'd3' dependency functions: [d3_dep_jetpack\(\)](#), [d3_dep_v3\(\)](#), [d3_dep_v4\(\)](#), [d3_dep_v5\(\)](#), [d3_dep_v7\(\)](#)

Examples

```
library(d3r)
library(htmltools)

tagList(d3_dep_v6())
```

`d3_dep_v7`*'d3.js' Dependency for Version 7*

Description

'd3.js' Dependency for Version 7

Usage

```
d3_dep_v7(offline = TRUE)
```

Arguments

`offline` logical to specify whether to use a local copy of d3.js (TRUE) or use cdn (FALSE)

Value

`htmltools::htmlDependency`

See Also

[d3_dep_v6](#), [d3_dep_v5](#), [d3_dep_v4](#), [d3_dep_v3](#), and [d3_dep_jetpack](#).

Other 'd3' dependency functions: [d3_dep_jetpack\(\)](#), [d3_dep_v3\(\)](#), [d3_dep_v4\(\)](#), [d3_dep_v5\(\)](#), [d3_dep_v6\(\)](#)

Examples

```
library(d3r)
library(htmltools)

tagList(d3_dep_v7())
```

`d3_igraph`*Convert 'igraph' to 'd3.js'*

Description

Convert 'igraph' to 'd3.js'

Usage

```
d3_igraph(igrf = NULL, json = TRUE)
```

Arguments

<code>igrf</code>	<code>igraph</code>
<code>json</code>	logical to return as JSON

Value

list

Examples

```
## Not run:
library(igraph)
library(igraphdata)
library(d3r)

# with random graph
d3r::d3_igraph(igraph::sample_pa(100))

# check case where vertices 0 cols
d3_igraph(igraph::watts.strogatz.game(1, 50, 4, 0.05))

# with karate from igraphdata
# notice graph attributes are added
data("karate", package="igraphdata")
(karate_d3 <- d3r::d3_igraph(karate))

listviewer::jsonedit(karate_d3)

data("kite", package="igraphdata")
listviewer::jsonedit(d3_igraph(kite))

## End(Not run)
```

d3_json	<i>Create 'JSON' that 'd3.js' Expects</i>
---------	---

Description

Create 'JSON' that 'd3.js' Expects

Usage

```
d3_json(x = NULL, strip = TRUE)
```

Arguments

x	data, usually in the form of <code>data.frame</code> or <code>list</code>
strip	logical to remove outer array. Use <code>strip=TRUE</code> for hierarchies from <code>d3_nest</code>

Value

string of 'JSON' data

d3_nest	<i>Convert a <code>data.frame</code> to a 'd3.js' Hierarchy</i>
---------	---

Description

Convert a `data.frame` to a 'd3.js' Hierarchy

Usage

```
d3_nest(data = NULL, value_cols = character(), root = "root", json = TRUE)
```

Arguments

data	<code>data.frame</code> or <code>data.frame</code> derivative, such as <code>tibble</code>
value_cols	character vector with the names of the columns to use as data
root	character name of the root level of the hierarchy
json	logical to return as JSON

Value

nested `data.frame`

Examples

```

# convert Titanic to a nested d3 hierarchy

# devtools::install_github("timelyportfolio/d3r")
library(d3r)
library(dplyr)

titanic_df <- data.frame(Titanic)
tit_tb <- titanic_df %>%
  select(Class, Age, Survived, Sex, Freq) %>%
  d3_nest(value_cols="Freq", root="titanic")

tit_tb

# see as tibble
titanic_df %>%
  select(Class, Age, Survived, Sex, Freq) %>%
  d3_nest(value_cols="Freq", root="titanic", json=FALSE)

# see the structure with listviewer
tit_tb %>%
  listviewer::jsonedit()
## Not run:
library(treemap)
library(d3r)
library(dplyr)
library(tidyr)

treemap::random.hierarchical.data() %>%
  d3_nest(value_cols = "x")

# use example from ?treemap
data(GNI2014)
treemap(
  GNI2014,
  index=c("continent", "iso3"),
  vSize="population",
  vColor="GNI",
  type="value",
  draw=FALSE
) %>%
  {.$tm} %>%
  select(continent, iso3, color, vSize) %>%
  d3_nest(value_cols = c("color", "vSize"))

## End(Not run)

```

Description

This thing is not even close to being done, so please help with ideas and contributions.

Usage

```
d3_party(tree = NULL, json = TRUE)
```

Arguments

tree	partykit object to be converted
json	logical to return list or json

Value

list or json depending on json arg

Examples

```
## Not run:

library(d3r)
# from ?rpart
data("kyphosis", package="rpart")
d3_party(
  rpart::rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
)

# if you want the list instead of json
d3_party(
  rpart::rpart(Kyphosis ~ Age + Number + Start, data = kyphosis),
  json = FALSE
)

# with ctree instead of rpart
# using example from ?ctree
d3_party(partykit::ctree(Species ~ ., data = iris))

# devtools::install_github("timelyportfolio/d3treeR")

library(d3treeR)

d3tree2(
  d3_party(
    rpart::rpart(Kyphosis ~ Age + Number + Start, data = kyphosis)
  ),
  celltext = "rule",
  valueField = "n"
)

## End(Not run)
```

d3_table	<i>Converts Table to 'd3' Nodes and Links</i>
----------	---

Description

Converts Table to 'd3' Nodes and Links

Usage

```
d3_table(tB = NULL, vars = NULL, agg = "Freq")
```

Arguments

tB	table to convert
vars	character vector of column names
agg	character column name of aggregated value

Value

list of two data.frames - one for nodes and one for links of the network. This structure is helpful when working with networkD3 and visNetwork.

Examples

```
library(d3r)
d3_table(Titanic, c("Class", "Sex"))
```

d3_v8	<i>Create V8 Context with D3</i>
-------	----------------------------------

Description

Create V8 Context with D3

Usage

```
d3_v8(...)
```

Arguments

...	arguments passed to v8()
-----	--------------------------

Value

v8 context with d3.js loaded and available as d3

Examples

```

## Not run:

# to do this all in R, please see ggraph
# https://github.com/thomasp85/ggraph
# by Thomas Lin Pedersen
library(d3r)

# make a simple data.frame of US state data
states <- data.frame(
  region = as.character(state.region),
  state = as.character(state.abb),
  population = state.x77[,"Population"],
  stringsAsFactors = FALSE
)

# use d3_nest to get the data.frame in a d3 hierarchy
state_hier <- d3_nest(
  states,
  value_cols = "population"
)

# use d3_v8 to do something useful with d3 and, our state data
ctx <- d3_v8()
ctx$eval(sprintf(
  " var states = %s",
  state_hier
))
ctx$eval(
  "
  // we assigned the variable states above
  // so now make it a real d3 hierarchy
  var root = d3.hierarchy(states);

  // sum on population
  root.sum(function(d) {return d.population ? d.population : 0});

  // use d3 to circle pack or state hierarchy
  d3.pack()(root);

  // get something we can convert into a data.frame in R
  var states_packed = [];
  root.each(function(d) {
    states_packed.push({
      name: d.data.name,
      radius: d.r,
      x: d.x,
      y: d.y
    });
  });
  });
  "

```

```

)

# now get states_packed from our context
# to plot in R
states_packed <- ctx$get("states_packed")
opar <- par(no.readonly=TRUE)
# make it square
par(pty="s")
symbols(
  states_packed$x,
  states_packed$y,
  states_packed$radius,
  inches=FALSE,
  asp=1
)
text(y~x, data=states_packed, labels=states_packed$name)
# return to original par before we made it square
par(opar)

# d3.quadtree example

library(d3r)

x = runif(100)
y = runif(100)

ctx <- d3_v8()
# assign pts as array of pts in V8
ctx$assign("pts", matrix(c(x,y),ncol=2,byrow=TRUE))
# use d3.quadtree() to plot rects
ctx$eval(
  "
  var d3q = d3.quadtree()
  .addAll(pts);
  // nodes function from https://bl.ocks.org/mbostock/4343214
  function nodes(quadtree) {
  var nodes = [];
  quadtree.visit(function(node, x0, y0, x1, y1) {
  nodes.push({x0:x0, y0:y0, x1: x1, y1: y1})
  });
  return nodes;
  }
  "
)

nodes <- ctx$get("nodes(d3q)", simplifyVector = FALSE)
# draw points
opar <- par(no.readonly=TRUE)
# make it square
par(pty="s")
plot(y~x)
# draw quadtree rects
rect(

```

```
lapply(nodes, function(x){x$x0}),
lapply(nodes, function(x){x$y0}),
lapply(nodes, function(x){x$x1}),
lapply(nodes, function(x){x$y1})
)
par(opar)

## End(Not run)
```

promote_na *Apply 'promote_na' to All Rows*

Description

Apply 'promote_na' to All Rows

Usage

```
promote_na(x)
```

Arguments

x data.frame

Value

data.frame

promote_na_one *Promote NA to Top Level*

Description

Promote NA to Top Level

Usage

```
promote_na_one(x)
```

Arguments

x data.frame

Value

data.frame

Index

* 'd3' dependency functions

d3_dep_jetpack, [2](#)

d3_dep_v3, [4](#)

d3_dep_v4, [5](#)

d3_dep_v5, [5](#)

d3_dep_v6, [6](#)

d3_dep_v7, [7](#)

change_to_name, [2](#)

d3_dep_jetpack, [2](#), [4–7](#)

d3_dep_v3, [3](#), [4](#), [5–7](#)

d3_dep_v4, [3](#), [4](#), [5](#), [6](#), [7](#)

d3_dep_v5, [3–5](#), [5](#), [6](#), [7](#)

d3_dep_v6, [3–6](#), [6](#), [7](#)

d3_dep_v7, [3–6](#), [7](#)

d3_igraph, [8](#)

d3_json, [9](#)

d3_nest, [9](#)

d3_party, [10](#)

d3_table, [12](#)

d3_v8, [12](#)

promote_na, [15](#)

promote_na_one, [15](#)