Package 'g3viz'

May 29, 2019

Type Package

Title Interactively Visualize Genetic Mutation Data using a Lollipop-Diagram

Version 1.1.2

Maintainer Xin Guo <g3viz.group@gmail.com>

Description R interface for 'g3-lollipop' JavaScript library. Visualize genetic mutation data using an interactive lollipop diagram in RStudio or your browser.

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Depends R (>= 3.0.0)

Imports jsonlite, cgdsr, stringr, htmlwidgets

Suggests shiny (>= 1.0.0), knitr, rmarkdown, kableExtra

URL https://github.com/G3viz/g3viz

BugReports https://github.com/G3viz/g3viz/issues

RoxygenNote 6.1.1

VignetteBuilder knitr

NeedsCompilation no

Author Xin Guo [aut, cre]

Repository CRAN

Date/Publication 2019-05-29 21:30:10 UTC

R topics documented:

`g3Lollipop` *Render g3lollipop diagram for the given mutation data*

Description

Render g3lollipop diagram for the given mutation data

Usage

```
g3Lollipop(mutation.dat, gene.symbol, uniprot.id = NA,
  gene.symbol.col = "Hugo_Symbol", aa.pos.col = "AA_Position",
  protein.change.col = c("Protein_Change", "HGVSp_Short"),
  factor.col = "Mutation_Class", plot.options = g3Lollipop.options(),
  save.png.btn = TRUE, save.svg.btn = TRUE, btn.style = NA,
  output.filename = "output")
```

Arguments

`mutation.dat`	Input genomic mutation data frame
`gene.symbol`	HGNC primary gene symbol
`uniprot.id`	UniProt ID, in case that the specified gene symbol links to multiple UniProt entries (isoforms). For example, *AKAP7* gene has two isoforms in UniProt, O43687 and Q9P0M2.
`gene.symbol.col`	Column name of Hugo gene symbols (e.g., TP53). Default *Hugo_Symbol*.
`aa.pos.col`	Column name of the parsed amino-acid change position. Default *AA_Position*.
`protein.change.col`	Column name of protein change information (e.g., p.K960R, G658S, L14Sfs*15). Default is a list of *Protein_Change*, *HGVSp_Short*.
`factor.col`	column of classes in the plot legend. IF `NA`, use parsed *Mutation_Class* column, otherwise, use specified. Default `NA`.
`plot.options`	g3lollipop diagram options in list format. Check `g3Lollipop.options`
`save.png.btn`	If add *save-as-png* button to the diagram. Default `TRUE`.
`save.svg.btn`	If add *save-as-svg* button to the diagram. Default `TRUE`.

`btn.style` button style, including browser default button style, and two built-in styles, *blue* or *gray*. Default `NA`, indicating browser default.

`output.filename`
Specify output file name.

Examples

```
# system mutation data
maf.file <- system.file("extdata", "TCGA.BRCA.varscan.somatic.maf.gz", package = "g3viz")
# read in MAF file
mutation.dat <- readMAF(maf.file)

# use built-in chart theme
chart.options <- g3Lollipop.theme(theme.name = "default",
                                  title.text = "PIK3CA gene (default theme)")
# generate chart
g3Lollipop(mutation.dat,
           gene.symbol = "PIK3CA",
           plot.options = chart.options,
           btn.style = "blue",
           output.filename = "default_theme")
```

---

g3Lollipop-shiny *Shiny bindings for g3Lollipop*

Description

Output and render functions for using g3viz lollipop diagram within Shiny applications and interactive Rmd documents.

Usage

```
g3LollipopOutput(outputId, width = "100%", height = "520px")

renderG3Lollipop(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

`outputId` output variable to read from

`width, height` Must be a valid CSS unit (like `'100%'`, `'400px'`, `'auto'`) or a number, which will be coerced to a string and have `'px'` appended.

`expr` An expression that generates a g3-lollipop

`env` The environment in which to evaluate `expr`.

`quoted` Is `expr` a quoted expression (with `quote()`)? This is useful if you want to save an expression in a variable.

`g3Lollipop.options` *G3Lollipop plot options*

Description

G3Lollipop plot options

Usage

```
g3Lollipop.options(chart.width = 800, chart.type = "circle",
  chart.margin = list(left = 40, right = 20, top = 15, bottom = 25),
  chart.background = "transparent", transition.time = 600,
  y.axis.label = "# of mutations",
  axis.label.font = "normal 12px Arial", axis.label.color = "#4f4f4f",
  axis.label.alignment = "middle", axis.label.dy = "-2em",
  y.axis.line.color = "#c4c8ca", y.axis.line.style = "dash",
  y.axis.line.width = 1, y.max.range.ratio = 1.1,
  legend.margin = list(left = 10, right = 0, top = 5, bottom = 5),
  legend.interactive = TRUE, legend.title = NA,
  lollipop.track.height = 420,
  lollipop.track.background = "rgb(233,233,233)",
  lollipop.pop.min.size = 2, lollipop.pop.max.size = 12,
  lollipop.pop.info.limit = 8, lollipop.pop.info.color = "#EEE",
  lollipop.pop.info.dy = "0.35em",
  lollipop.line.color = "rgb(42,42,42)", lollipop.line.width = 0.5,
  lollipop.circle.color = "wheat", lollipop.circle.width = 0.5,
  lollipop.label.ratio = 1.4, lollipop.label.min.font.size = 10,
  lollipop.color.scheme = "accent", highlight.text.angle = "90",
  title.text = "", title.font = "normal 16px Arial",
  title.color = "#424242", title.alignment = "middle",
  title.dy = "0.35em", anno.height = 30, anno.margin = list(top = 4,
  bottom = 0), anno.background = "transparent",
  anno.bar.fill = "#e5e3e1", anno.bar.margin = list(top = 2, bottom =
  2), domain.color.scheme = "category10", domain.margin = list(top = 0,
  bottom = 0), domain.text.font = "normal 11px Arial",
  domain.text.color = "#f2f2f2", brush = TRUE,
  brush.selection.background = "#666", brush.selection.opacity = 0.2,
  brush.border.color = "#969696", brush.handler.color = "#333",
  brush.border.width = 1, legend = TRUE, tooltip = TRUE,
  zoom = TRUE)
```

Arguments

`chart.width`	chart width. Default 800.
`chart.type`	*pie* or *circle*. Default *circle*.
`chart.margin`	specify chart margin in _list_ format. Default `list(left = 40, right = 20, top = 15, bottom = 25)`.

`chart.background`
: chart background. Default *transparent*.

`transition.time`
: animation transition time when clicking lollipop pops to show labels (in millisecond). Default 600.

`y.axis.label`
: Y-axis label text. Default *"# of mutations"*.

`axis.label.font`
: css font style shorthand (*font-style font-variant font-weight font-size/line-height font-family*). Default *"normal 12px Arial"*.

`axis.label.color`
: axis label text color. Default *#4f4f4f*.

`axis.label.alignment`
: axis label text alignment (start/end/middle). Default *middle*.

`axis.label.dy`
: text adjustment of axis label text. Default *-2em*.

`y.axis.line.color`
: color of y-axis in-chart lines (ticks). Default *#c4c8ca*.

`y.axis.line.style`
: style of y-axis in-chart lines (ticks), "dash" or "line". Default *dash*.

`y.axis.line.width`
: width of y-axis in-chart lines (ticks). Default 1.

`y.max.range.ratio`
: ratio of y-axis range to data value range. Default 1.1.

`legend.margin`
: legend margin in *list*. Default `list(left = 10, right = 0, top = 5, bottom = 5)`.

`legend.interactive`
: legend interactive mode. Default `TRUE`.

`legend.title`
: legend title. If `NA`, *factor.col* in `g3Lollipop` is used. Default is `NA`.

`lollipop.track.height`
: height of lollipop track. Default 420.

`lollipop.track.background`
: background of lollipop track. Default *rgb(244,244,244)*

`lollipop.pop.min.size`
: lollipop pop minimal size. Default 2.

`lollipop.pop.max.size`
: lollipop pop maximal size. Default 12.

`lollipop.pop.info.limit`
: threshold of lollipop pop size to show count information in middle of pop. Default 8.

`lollipop.pop.info.color`
: lollipop pop information text color. Default *#EEE*.

`lollipop.pop.info.dy`
: y-axis direction text adjustment of lollipop pop information. Default *-0.35em*.

`lollipop.line.color`
: lollipop line color. Default *rgb(42,42,42)*.

`lollipop.line.width`
: lollipop line width. Default 0.5.

`lollipop.circle.color`
: lollipop circle border color. Default *wheat*.

`lollipop.circle.width`
: lollipop circle border width. Default 0.5.

`lollipop.label.ratio`
: lollipop click-out label font size to circle size ratio. Default 1.4.

`lollipop.label.min.font.size`
: lollipop click-out label minimal font size. Default 10.

`lollipop.color.scheme`
: color scheme to fill lollipop pops. Default *accent*.

`highlight.text.angle`
: pop-on-click highlight text angle. Default 90.

`title.text`
: title of chart. Default is empty.

`title.font`
: font of chart title. Default *normal 16px Arial*.

`title.color`
: color of chart title. Default *#424242*.

`title.alignment`
: text alignment of chart title (start/middle/end). Default *middle*.

`title.dy`
: text adjustment of chart title. Default *0.35em*.

`anno.height`
: height of protein structure annotation track. Default 30.

`anno.margin`
: margin of protein structure annotation track. Default `list(top = 4, bottom = 0)`.

`anno.background`
: background of protein structure annotation track. Default *transparent*.

`anno.bar.fill`
: background of protein bar in protein structure annotation track. Default *#e5e3e1*.

`anno.bar.margin`
: margin of protein bar in protein structure annotation track. Default `list(top = 2, bottom = 2)`.

`domain.color.scheme`
: color scheme of protein domains. Default *category10*.

`domain.margin`
: margin of protein domains. Default `list(top = 0, bottom = 0)`.

`domain.text.font`
: domain label text font in shorthand format. Default *normal 11px Arial*.

`domain.text.color`
: domain label text color. Default *#f2f2f2*.

`brush`
: if show brush. Default `TRUE`.

`brush.selection.background`
: background color of selection brush. Default *#666*.

`brush.selection.opacity`
: background opacity of selection brush. Default 0.2.

`brush.border.color`
: border color of selection brush. Default *#969696*.

`brush.handler.color`
: color of left and right handlers of selection brush. Default *#333*.

`brush.border.width`
: border width of selection brush. Default 1.

`legend`
: if show legend. Default `TRUE`.

`tooltip`
: if show tooltip. Default `TRUE`.

`zoom`
: if enable zoom feature. Default `TRUE`.

Value

a list with g3Lollipop plot options

`g3Lollipop.theme` *G3Lollipop chart options of built-in themes.*

Description

G3Lollipop chart options of built-in themes.

Usage

```
g3Lollipop.theme(theme.name = "default", title.text = "",
  y.axis.label = "# of mutations", legend.title = NA)
```

Arguments

`theme.name`
: theme name, including *default*, *cbioportal*, *nature*, *nature2*, *dark*, *blue*, *ggplot2*, and *simple*. Default *default*.

`title.text`
: title of chart. Default is empty.

`y.axis.label`
: Y-axis label text. Default *"# of mutations"*.

`legend.title`
: legend title. If `NA`, *factor.col* in `g3Lollipop` is used. Default is `NA`.

Value

a list with g3Lollipop plot options

`getMutationsFromCbioportal` *Query cancer genomic mutation data from cBioPortal*

Description

Retrieve and parse mutation data from cBioPortal by the given cBioPortal cancer study ID and the gene symbol.

Usage

```
getMutationsFromCbioportal(study.id, gene.symbol, output.file = NA,
  mutation.type.to.class.df = NA,
  cgds.url = "http://www.cbioportal.org/", test.cgds = FALSE)
```

Arguments

`study.id`	cbioprotal study ID
`gene.symbol`	HGNC gene symbol.
`output.file`	if specified, output to a file in *CSV* format. Default is `NA`.
`mutation.type.to.class.df`	mapping table from mutation type to class. See `mapMutationTypeToMutationClass` for details. Default `NA`, which indicates to use default mappings.
`cgds.url`	the URL for the public CGDS server (Cancer Genomic Data Server). Default is `http://www.cbioportal.org/`. Check *cgdsr* R-package for details.
`test.cgds`	if test CGDS connection. Default is `FALSE`

Value

a data frame with columns

Hugo_Symbol Hugo gene symbol

Protein_Change Protein change information (cBioprotal uses *HGVSp* format)

Sample_ID Sample ID

Mutation_Type mutation type, aka, variant classification.

Chromosome chromosome

Start_Position start position

End_Position end position

Reference_Allele reference allele

Variant_Allele variant allele

Mutation_Class mutation class (e.g., Truncating/Missense/Inframe/Other)

AA_Position amino-acid position of the variant; if the variant is not in protein-coding region, `NA`

Examples

```
## Not run:
# Usage:
# Connection to CGDS (Cange Genomic Data Server). Internet access required.
# Note: this may need more than 10 seconds, and sometimes it may fail.
library(cgdsr)
cgds <- CGDS("http://www.cbioportal.org/")

# test if connection is OK (warning: sometimes it may fail)
test(cgds)

# list all studies of cBioPortal
all.studies <- getCancerStudies(cgds)

# First, select a cancer study that contains mutation data set ("caner_study_id")
# then, query genomic mutation data using a HGNC gene symbol,
# for example
mutation.dat <- getMutationsFromCbioportal("msk_impact_2017", "TP53")
mutation.dat <- getMutationsFromCbioportal("all_stjude_2016", "TP53")

## End(Not run)
```

`guessMAFColumnName` *Guess column name for MAF file*

Description

Guess column name for MAF file

Usage

```
guessMAFColumnName(maf.df, alt.column.names)
```

Arguments

`maf.df`	MAF data frame
`alt.column.names`	a vector of alternative column names

Value

if hit one alternative column name, return the name; otherwise, return `NA`

hgnc2pfam *Map from Hugo symbol to Pfam domains*

Description

Mapping from Hugo symbol to Pfam-A domain composition. If the given Hugo symbol has multiple UniProt ID mappings, and `guess == TRUE`, the longest UniProt protein is selected. Return is either a list of a JSON.

Usage

```
hgnc2pfam(hgnc.symbol, guess = TRUE, uniprot.id = NA,
  output.format = "json")
```

Arguments

`hgnc.symbol`	primary Hugo symbol
`guess`	if the given Hugo symbol links to multiple UniProt IDs, choose the longest one (`guess == TRUE`); otherwise `NA` (`guess == FALSE`). Default `TRUE`.
`uniprot.id`	UniProt ID, in case that gene symbol maps to multiple UniProt entries.
`output.format`	output format: JSON or list

Value

A list or a JSON with attributes: *symbol*, *uniprot*, *length*, and a list of *Pfam* entries, including *hmm.acc*, *hmm.name*, *start*, *end*, and *type*.

Examples

```
# general usage
hgnc2pfam("TP53")
hgnc2pfam("TP53", output.format = "json")
hgnc2pfam("TP53", output.format = "list")
hgnc2pfam("TP53", output.format = "json", uniprot.id = "P04637") # OK

# for gene mapping to multiple UniProt enties
hgnc2pfam("GNAS", guess = TRUE)
hgnc2pfam("GNAS", guess = FALSE)
hgnc2pfam("GNAS", output.format = "list")
hgnc2pfam("GNAS", output.format = "list", uniprot.id = "P84996")
## Not run:
hgnc2pfam("GNAS", output.format = "list", uniprot.id = "P84997") # not exists, returns FALSE

## End(Not run)

hgnc2pfam("PRAMEF9", output.format = "list") # no Pfam mappings
```

hgnc2pfam.df *Mapping table between gene.symbol, uniprot.id, and pfam*

Description

A dataset containing the mapping table between Hugo symbol, UniProt ID, and Pfam ACC.

Usage

```
hgnc2pfam.df
```

Format

A data frame with columns:

symbol Gene symbol

uniprot UniProt ID

length protein length

start starting position of Pfam domain

end ending position of Pfam domain

hmm.acc Pfam accession number

hmm.name Pfam name

type Pfam type, i.e., domain/family/motif/repeat/disordered/coiled-coil

Source

Pfam (v31.0) and UniProt

Examples

```
hgnc2pfam.df
```

hgnc2uniprot *Mapping from Hugo symbol to UniProt IDs*

Description

Mapping from Hugo Symbol to UniProt ID using internal mapping table. Return a data frame with columns *symbol* (Hugo symbol), *uniprot* (UniProt ID), and *length* (protein length).

Usage

```
hgnc2uniprot(hgnc.symbol)
```

Arguments

`hgnc.symbol`	primary HUGO symbol

Value

a data frame with columns *symbol* (Hugo symbol), *uniprot* (UniProt ID), and *length* (protein length).

Examples

```
# maps to single UniProt entry
hgnc2uniprot("TP53")

# maps to multiple UniProt entries
hgnc2uniprot("GNAS")
hgnc2uniprot("AKAP7")
```

`mapMutationTypeToMutationClass` *Map from mutation type (aka, variant classification) to mutation class*

Description

Map from mutation type (aka, variant classification) to mutation class. Default mappings are as follows,

- Missense
  - *Missense_Mutation* — a point mutation in which a single nucleotide change results in a codon that codes for a different amino acid See https://en.wikipedia.org/wiki/Missense_mutation.
- Inframe
  - *In_Frame_Del* — a deletion that keeps the sequence in frame
  - *In_Frame_Ins* — an insertion that keeps the sequence in frame
  - *Silent* — variant is in coding region of the chosen transcript, but protein structure is identical (i.e., a synonymous mutation)
  - *Targeted_Region* — targeted region
- Truncating
  - *Frame_Shift* — a variant caused by indels of a number of nucleotides in a DNA sequence that is not divisible by three. See https://en.wikipedia.org/wiki/Frameshift_mutation.
  - *Frame_Shift_Ins* — a variant caused by insertion that moves the coding sequence out of frame. See https://en.wikipedia.org/wiki/Frameshift_mutation.
  - *Frame_Shift_Del* — a variant caused by deletion that moves the coding sequence out of frame. See https://en.wikipedia.org/wiki/Frameshift_mutation.

  - *Nonsense_Mutation* — a premature stop codon that is created by the variant. See https://en.wikipedia.org/wiki/Nonsense_mutation.
  - *Nonstop_Mutation* — a variant that removes stop codon.
  - *Splice_Site* — a variant that is within two bases of a splice site.
  - *Splice_Region* — a variant that is within splice region.
- Other
  - *5'UTR* — a variant that is on the 5'UTR for the chosen transcript.
  - *3'UTR* — a variant that is on the 3'UTR for the chosen transcript.
  - *5'Flank* — a variant that is upstream of the chosen transcript (generally within 3kb).
  - *3'Flank* — a variant that is downstream of the chosen transcript (generally within 3kb).
  - *Fusion* — a gene fusion.
  - *IGR* — an intergenic region. Does not overlap any transcript.
  - *Intron* — a variant that lies between exons within the bounds of the chosen transcript.
  - *Translation_Start_Site* — a variant that is in translation start site.
  - *De_novo_Start_InFrame* — a novel start codon that is created by the given variant using the chosen transcript. However, it is in frame relative to the coded protein.
  - *De_novo_Start_OutOfFrame* — a novel start codon that is created by the given variant using the chosen transcript. However, it is out of frame relative to the coded protein.
  - *Start_Codon_SNP* — a point mutation that overlaps the start codon.
  - *Start_Codon_Ins* — an insertion that overlaps the start codon.
  - *Start_Codon_Del* — a deletion that overlaps the start codon.
  - *RNA* — a variant that lies on one of the RNA transcripts.
  - *lincRNA* — a variant that lies on one of the lincRNAs.
  - *Unknown* — Unknown

Usage

```
mapMutationTypeToMutationClass(mutation.type.vec,
  mutation.type.to.class.df = NA)
```

Arguments

`mutation.type.vec`
: a vector of mutation type information

`mutation.type.to.class.df`
: A mapping table from mutation type (header *Mutation_Type*) to mutation class (header *Mutation_Class*). Default `NA`, which indicates to use default mappings.

Value

a vector of mapped mutation class information

mutation.table.df *Default mapping table between mutation type (aka, variant classification) to mutation class*

Description

A dataset containing the mapping table between genomic mutation type (aka, variant classification) to mutation class. See `mapMutationTypeToMutationClass` for details.

Usage

```
mutation.table.df
```

Format

A data frame with three columns:

Mutation_Type Mutation type, aka, variant classification

Mutation_Class mutation class

Short_Name short name of mutation type

Examples

```
mutation.table.df
```

parseProteinChange *Extract* amino_acid_position *from* Protein_Change

Description

Parse *amino_acid_position* according to HGVSp_short format.
For example, *p.Q16Rfs*28*, amino-acid position is 16. See http://varnomen.hgvs.org/recommendations/protein/ or https://www.hgvs.org/mutnomen/recs-prot.html.

Usage

```
parseProteinChange(protein.change.vec, mutation.class.vec)
```

Arguments

`protein.change.vec`
: a vector of strings with protein change information, usually in HGVSp_short format.

`mutation.class.vec`
: a vector of strings with mutation class (or so-called variant classification) information.

Value

a vector of parsed amino-acid position

`readMAF` *Read MAF file*

Description

Read mutation information from MAF file. For MAF format specification, see https://docs.gdc.cancer.gov/Data/File_Formats/MAF_Format/.

Usage

```
readMAF(maf.file, gene.symbol.col = "Hugo_Symbol",
  variant.class.col = c("Variant_Classification", "Mutation_Type"),
  protein.change.col = c("Protein_Change", "HGVSp_Short"),
  if.parse.aa.pos = TRUE, if.parse.mutation.class = TRUE,
  mutation.class.col = "Mutation_Class", aa.pos.col = "AA_Position",
  mutation.type.to.class.df = NA, sep = "\t", quote = "", ...)
```

Arguments

`maf.file`	MAF file name. Gnuzipped input file allowed, with ".gz" file extension.
`gene.symbol.col`	Column name of Hugo gene symbols (e.g., TP53). Default *Hugo_Symbol*.
`variant.class.col`	Column name for variant class information (e.g., *Missense_Mutation*, *Nonsense_Mutation*). Default is the first match of *Variant_Classification* or *Mutation_Type*.
`protein.change.col`	Column name for protein change information (e.g., p.K960R, G658S, L14Sfs*15). Default is the first match of *Protein_Change* or *HGVSp_Short*.
`if.parse.aa.pos`	if parse amino-acid position of mutations. Default is `TRUE`.
`if.parse.mutation.class`	if parse mutation class from mutation type (variant classification) information. Default is `TRUE`.
`mutation.class.col`	Column name of the parsed mutation class. Default *Mutation_Class*.
`aa.pos.col`	Column name of the parsed amino-acid change position. Default *AA_Position*.
`mutation.type.to.class.df`	mapping table from mutation type to class. `mapMutationTypeToMutationClass` for details. Default `NA`, which indicates to use default mappoings.
`sep`	separator of columns. Default `sep = "\t"`.
`quote`	the set of quoting characters. To disable quoting altogether, use `quote = ""`. Default `quote = ""`.
`...`	additional parameters pass to `read.table`.

Value

a data frame containing MAF information, plus optional columns of the parsed *Mutation_Class* and *Protein_Position*.

`uniprot2pfam` *From UniProt ID to Pfam-A domain composition*

Description

Map from UniProt ID to Pfam-A domain compostion.

Usage

```
uniprot2pfam(uniprot.id)
```

Arguments

`uniprot.id`	UniProt ID

Value

a data frame with columns

- *uniprot* — UniProt ID
- *length* — protein length
- *hmm.acc* — accession number of Pfam HMM model, e.g., PF08563
- *hmm.name* — Pfam name, e.g., P53_TAD
- *start* — Pfam domain start position
- *end* — Pfam domain end position
- *type* — Pfam type, including domain/motif/family

Examples

```
uniprot2pfam("Q5VWM5")  # PRAMEF9; PRAMEF15
uniprot2pfam("P04637")
```

Index