Package ‘trackViewer’

May 8, 2024

Type Package

Title A R/Bioconductor package with web interface for drawing elegant interactive tracks or lollipop plot to facilitate integrated analysis of multi-omics data

Version 1.40.0

Maintainer Jianhong Ou <jianhong.ou@duke.edu>

Description Visualize mapped reads along with annotation as track layers for NGS dataset such as ChIP-seq, RNA-seq, miRNA-seq, DNA-seq, SNPs and methylation data.

License GPL (>= 2)

Depends R (>= 3.5.0), grDevices, methods, GenomicRanges, grid

Imports GenomeInfoDb, GenomicAlignments, GenomicFeatures, Gviz, Rsamtools, S4Vectors, rtracklayer, BiocGenerics, scales, tools, IRanges, AnnotationDbi, grlImport, htmlwidgets, plotrix, InteractionSet, igraph, utils, rhdf5, strawr, txdbmaker

Suggests biomaRt, TxDb.Hsapiens.UCSC.hg19.knownGene, RUnit, org.Hs.eg.db, BiocStyle, knitr, VariantAnnotation, httr, htmltools, rmarkdown, motifStack

biocViews Visualization

VignetteBuilder knitr

RoxygenNote 7.3.1

Encoding UTF-8

git_url https://git.bioconductor.org/packages(trackViewer)

git_branch RELEASE_3_19

git_last_commit 14c514b

git_last_commit_date 2024-04-30

Repository Bioconductor 3.19

Date/Publication 2024-05-07

Author Jianhong Ou [aut, cre] (https://orcid.org/0000-0002-8652-2488), Julie Lihua Zhu [aut]
## Contents

<table>
<thead>
<tr>
<th>Function Name</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>trackViewer-package</td>
<td>3</td>
</tr>
<tr>
<td>addArrowMark</td>
<td>4</td>
</tr>
<tr>
<td>addGuideLine</td>
<td>5</td>
</tr>
<tr>
<td>addInteractionAnnotation</td>
<td>6</td>
</tr>
<tr>
<td>ARA</td>
<td>7</td>
</tr>
<tr>
<td>browseTracks</td>
<td>8</td>
</tr>
<tr>
<td>browseTracks-shiny</td>
<td>9</td>
</tr>
<tr>
<td>coverageGR</td>
<td>9</td>
</tr>
<tr>
<td>dandelion.plot</td>
<td>10</td>
</tr>
<tr>
<td>geneModelFromTxdb</td>
<td>11</td>
</tr>
<tr>
<td>geneTrack</td>
<td>13</td>
</tr>
<tr>
<td>getCurTrackViewport</td>
<td>13</td>
</tr>
<tr>
<td>getGeneIDsFromTxDb</td>
<td>14</td>
</tr>
<tr>
<td>getLocation</td>
<td>15</td>
</tr>
<tr>
<td>gi2track</td>
<td>15</td>
</tr>
<tr>
<td>gieStain</td>
<td>16</td>
</tr>
<tr>
<td>GIoperator</td>
<td>16</td>
</tr>
<tr>
<td>gridPlot</td>
<td>17</td>
</tr>
<tr>
<td>GRoperator</td>
<td>17</td>
</tr>
<tr>
<td>ideogramPlot</td>
<td>18</td>
</tr>
<tr>
<td>importBam</td>
<td>19</td>
</tr>
<tr>
<td>importData</td>
<td>20</td>
</tr>
<tr>
<td>importGInteractions</td>
<td>21</td>
</tr>
<tr>
<td>importScore</td>
<td>23</td>
</tr>
<tr>
<td>importScSeqScore</td>
<td>24</td>
</tr>
<tr>
<td>listChromosomes</td>
<td>25</td>
</tr>
<tr>
<td>listResolutions</td>
<td>26</td>
</tr>
<tr>
<td>loadIdeogram</td>
<td>26</td>
</tr>
<tr>
<td>lolliplot</td>
<td>27</td>
</tr>
<tr>
<td>loopBouquetPlot</td>
<td>29</td>
</tr>
<tr>
<td>optimizeStyle</td>
<td>31</td>
</tr>
<tr>
<td>parse2GRanges</td>
<td>32</td>
</tr>
<tr>
<td>parseWIG</td>
<td>32</td>
</tr>
<tr>
<td>plotGRanges</td>
<td>33</td>
</tr>
<tr>
<td>plotIdeo</td>
<td>34</td>
</tr>
<tr>
<td>plotOneIdeo</td>
<td>35</td>
</tr>
<tr>
<td>pos-class</td>
<td>36</td>
</tr>
<tr>
<td>reduce,GInteractions-method</td>
<td>36</td>
</tr>
<tr>
<td>trackList-class</td>
<td>37</td>
</tr>
<tr>
<td>trackStyle-class</td>
<td>38</td>
</tr>
<tr>
<td>trackViewerStyle-class</td>
<td>40</td>
</tr>
<tr>
<td>viewGene</td>
<td>41</td>
</tr>
<tr>
<td>viewTracks</td>
<td>42</td>
</tr>
<tr>
<td>xscale-class</td>
<td>43</td>
</tr>
<tr>
<td>yaxisStyle-class</td>
<td>44</td>
</tr>
</tbody>
</table>

## Index

<table>
<thead>
<tr>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>45</td>
</tr>
</tbody>
</table>
**trackViewer-package**

**Minimal designed plotting tool for genomic data**

**Description**

A package that plot data and annotation information along genomic coordinates in an elegance style. This tool is based on Gviz but want to draw figures in minimal style for publication.

**Author(s)**

**Maintainer**: Jianhong Ou <jianhong.ou@duke.edu> ([ORCID](https://orcid.org/))

Authors:

- Julie Lihua Zhu <Julie.Zhu@umassmed.edu>

**Examples**

```r
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
trs <- geneModelFromTxdb(TxDb.Hsapiens.UCSC.hg19.knownGene,
                         org.Hs.eg.db,
                         chrom="chr11",
                         start=122929275,
                         end=122930122)
extdata <- system.file("extdata", package="trackViewer",
                         mustWork=TRUE)
repA <- importScore(paste(extdata, "cpsf160.repA_+.wig", sep="/"),
paste(extdata, "cpsf160.repA_-_.wig", sep="/"),
format="WIG")
strand(repA@dat) <- "+
strand(repA@dat2) <- "-
fox2 <- importScore(paste(extdata, "fox2.bed", sep="/"), format="BED")
dat <- coverageGR(fox2@dat)
fox2@dat <- dat[strand(dat)=="+
fox2@dat2 <- dat[strand(dat)=="-
gr <- GRanges("chr11", IRanges(122929275, 122930122), strand="-"
vp <- viewTracks(trackList(repA, fox2, trs), gr=gr, autoOptimizeStyle=TRUE)
addGuideLine(c(122929767, 122929969), vp=vp)
addArrowMark(list(x=unit(.5, "npc"),
               y=unit(.39, "npc")),
col="blue")
```
addArrowMark  

Add arrow mark to the figure at a given position

Description

A function to add arrow mark for emphasizing peaks

Usage

addArrowMark(
  pos = grid.locator(),
  label = NULL,
  angle = 15,
  length = unit(0.25, "inches"),
  col = "red",
  cex = 1,
  quadrant = 4,
  type = "closed",
  vp = NULL
)

Arguments

pos  
A unit object representing the location of arrow mark to be placed at current viewport. Default is the value of grid.locator, which will get the location of the mouse click.

label  
A character or expression vector.

angle  
A parameter passed into grid::arrow function. The angle of arrow head in degrees (smaller numbers produce narrower, pointier arrows). Essentially describes the width of the arrow head.

length  
A parameter passed into grid::arrow function. A unit specifying the length of the arrow head.

col  
color of the arrow

cex  
Multiplier applied to fontsize

quadrant  
the direction of arrow, 1: to bottomleft, 2: to bottomright, 3: to topright, 4: to topleft

type  
A parameter passed into grid::arrow function. One of "open" or "closed" indicating whether the arrow head should be a closed triangle.

vp  
A Grid viewport object. It must be output of viewTracks

Value

invisible x, y position value.
addGuideLine

See Also

See Also as `addGuideLine, arrow`

Examples

```r
grid.newpage()
addArrowMark(list(x=unit(.5, "npc"),
       y=unit(.5, "npc"),
       label="label1",
       col="blue")
## how to get the position by mouse click
if(interactive()){  
pos <- addArrowMark(label="byClick")
   addArrowMark(pos, label="samePosAsAbove")
}
```

---

addGuideLine | Add guide lines to the tracks

Description

A function to add lines for emphasizing the positions

Usage

```r
addGuideLine(guideLine, col = "gray", lty = "dashed", lwd = 1, vp = NULL)
```

Arguments

- `guideLine`: The genomic coordinates to draw the lines
- `col`: A vector for the line color
- `lty`: A vector for the line type
- `lwd`: A vector for the line width
- `vp`: A Grid viewport object. It must be output of `viewTracks`

See Also

See Also as `getCurTrackViewport, addArrowMark, viewTracks`

Examples

```r
vp <- getCurTrackViewport(trackViewerStyle(), 10000, 10200)
addGuideLine(c(10010, 10025, 10150), vp=vp)
```
addInteractionAnnotation

Add annotation markers to the figure at a given position

Description

A function to add annotation markers for emphasizing interactions

Usage

addInteractionAnnotation(
  obj,
  idx,
  FUN = grid.polygon,
  panel = c("top", "bottom"),
  ...
)

Arguments

obj A GInteractions object, GRanges object or numeric vector. For numeric vector, the positive value will generate a line with slope 1 and negative value will generate a line at the position with slope -1.
idx The layer number of track.
FUN Function for plot. Available functions are grid.polygon, grid.lines, and grid.text for GInteractions object; grid.lines, and grid.text for GRanges object; FUN is not used for numeric vector.
panel Plot regions. Available values are "top", "bottom".
... Parameters will be passed to FUN.

Value

invisible viewport for plot region.

See Also

See Also as addGuideLine, addArrowMark

Examples

library(trackViewer)
library(InteractionSet)
gi <- readRDS(system.file("extdata", "nij.chr6.51120000.53200000.gi.rds", package="trackViewer"))
tads <- GInteractions(GRanges("chr6",
  IRanges(c(51130001, 51130001, 51450001, 52210001), width = 20000)),
  ...
ARA

Aggregate Region Analysis

Description

Extract the interaction signal means from given coordinates.

Usage

ARA(gr, upstream = 2e+05, downstream = upstream, resolution = 10000, ...)

Arguments

gr A ‘GRanges’ object. The center of the object will be used for alignment for all the given regions.
upstream, downstream numeric(1L). Upstream and downstream from the center of given ‘gr’ input will be used to extract the signals.
resolution numeric(1L). The resolution will be passed to importGInteractions function.
...
The parameters used by importGInteractions function. Please note that the ranges resolution and out parameter should not be involved.

Value

A GInteractions object with scores which represent the mean values of the interactions.

Examples

hic <- system.file("extdata", "test_chr22.hic", package = "trackViewer", mustWork=TRUE)
gr <- GRanges("22", c(seq(20000001, 50000001, by=10000000), width=1))

gi <- ARA(gr, file= hic, format=" hic")
rg <- GRanges("22", IRanges(1, 400000))
op <- optimizeStyle(trackList(gi2track(gi)))
heatmap <- op$tracks
sty <- op$style
setTrackViewerStyleParam(sty, "xat", c(1, 200000, 400000))
setTrackViewerStyleParam(sty, "xlabel", c("-20K", "center", "20K"))
viewTracks(heatmap, viewerStyle=sty, gr=rg)
browseTracks

**Description**

browse tracks by a web browser.

**Usage**

```r
browseTracks(
  trackList,
  gr = GRanges(),
  ignore.strand = TRUE,
  width = NULL,
  height = NULL,
  ...
)
```

**Arguments**

- **trackList**: an object of `trackList`
- **gr**: an object of `GRanges`
- **ignore.strand**: ignore the strand or not when do filter. default `TRUE`
- **width**: width of the figure
- **height**: height of the figure
- **...**: parameters not used

**Value**

An object of class `htmlwidget` that will intelligently print itself into HTML in a variety of contexts including the R console, within R Markdown documents, and within Shiny output bindings.

**Examples**

```r
extdata <- system.file("extdata", package="trackViewer", mustWork=TRUE)
files <- dir(extdata, ".-.wig")
tracks <- lapply(paste(extdata, files, sep="/"),
  importScore, format="WIG")
tracks <- lapply(tracks, function(.ele) {strand(.ele@dat) <- "+"; .ele})
names(tracks) <- c("trackA", "trackB")
fox2 <- importScore(paste(extdata, "fox2.bed", sep="/"), format="BED")
dat <- coverageGR(fox2@dat)
fox2@dat <- dat[strand(dat)=="+"]
fox2@dat2 <- dat[strand(dat)=="-"]
gr <- GRanges("chr11", IRanges(122929275, 122930122))
browseTracks(trackList(tracks, fox2), gr=gr)
```
Shiny bindings for browseTracks

Description
Output and render functions for using browseTracks within Shiny applications and interactive Rmd documents.

Usage

browseTracksOutput(outputId, width = "100%", height = "600px")

renderbrowseTracks(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 browseTracks
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.

calculate coverage

Description
calculate coverage for GRanges, GAlignments or GAlignmentPairs

Usage
coverageGR(gr)

Arguments

gr  an object of RGanges, GAlignments or GAlignmentPairs

Value
an object of GRanges
See Also

See Also as coverage, coverage-methods

Examples

```r
bed <- system.file("extdata", "fox2.bed", package="trackViewer",
              mustWork=TRUE)
fox2 <- importScore(bed)
fox2$dat <- coverageGR(fox2$dat)
```

Description

Plot variants and somatic mutations

Usage

```r
dandelion.plot(
    SNP.gr,
    features = NULL,
    ranges = NULL,
    type = c("fan", "circle", "pie", "pin"),
    newpage = TRUE,
    ylab = TRUE,
    ylab.gp = gpar(col = "black"),
    xaxis = TRUE,
    xaxis.gp = gpar(col = "black"),
    yaxis = FALSE,
    yaxis.gp = gpar(col = "black"),
    legend = NULL,
    cex = 1,
    maxgaps = 1/50,
    heightMethod = NULL,
    label_on_feature = FALSE,
    ...
)
```

Arguments

- **SNP.gr**: A object of GRanges or GRangesList. All the width of GRanges must be 1.
- **features**: A object of GRanges or GRangesList.
- **ranges**: A object of GRanges or GRangesList.
- **type**: Character. Could be fan, circle, pie or pin.
geneModelFromTxdb

Prepare gene model from an object of TxDb

Description

Generate an object of track for viewTracks by given parameters.

ewpage

plot in the new page or not.

ylab
plot ylab or not. If it is a character vector, the vector will be used as ylab.

ylab.gp, xaxis.gp, yaxis.gp
An object of class gpar for ylab, xaxis or yaxis.

xaxis, yaxis
plot xaxis/yaxis or not. If it is a numeric vector with length greater than 1, the vector will be used as the points at which tick-marks are to be drawn. And the names of the vector will be used to as labels to be placed at the tick points if it has names.

legend
If it is a list with named color vectors, a legend will be added.

cex
cex will control the size of circle.

maxgaps
maxgaps between the stem of dandelions. It is calculated by the width of plot region divided by maxgaps. If a GRanges object is set, the dandelions stem will be clustered in each genomic range.

heightMethod
A function used to determine the height of stem of dandelion. eg. Mean. Default is length.

label_on_feature
Labels of the feature directly on them. Default FALSE.

Details

In SNP.gr and features, metadata of the GRanges object will be used to control the color, fill, border, height, data source of pie if the type is pie.

Examples

SNP <- c(10, 100, 105, 108, 400, 410, 420, 600, 700, 805, 840, 1400, 1402)
SNP.gr <- GRanges("chr1", IRanges(SNP, width=1, names=paste0("snp", SNP)), score=sample.int(100, length(SNP))/100)
features <- GRanges("chr1", IRanges(c(1, 501, 1001), width=c(120, 500, 405), names=paste0("block", 1:3)), color="black", fill=c("#FF8833", "#51C6E6", "#DFA32D"), height=c(0.1, 0.05, 0.08))
dandelion.plot(SNP.gr, features, type="fan")
Usage

geneModelFromTxdb(
  txdb,
  orgDb,
  gr,
  chrom,
  start,
  end,
  strand = c("*", "+", "-"),
  txdump = NULL
)

Arguments

txdb An object of TxDb
orgDb An object of "OrgDb"
gr An object of GRanges.
chrom chromosome name, must be a seqname of txdb
start start position
end end position
strand strand
txdump output of as.list(txdb), a list of data frames that can be used to make the db again with no loss of information.

Value

Generate a list of track from a TxDb object.

See Also

See Also as importScore, importBam, viewTracks

Examples

library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
trs <- geneModelFromTxdb(TxDb.Hsapiens.UCSC.hg19.knownGene,
                        org.Hs.eg.db,
                        chrom="chr20",
                        start=22560000,
                        end=22565000,
                        strand="-"
**geneTrack**

*track from TxDb*

**Description**

Generate a track object from TxDb by given gene ids

**Usage**

```r
geneTrack(ids, txdb, symbols, type = c("gene", "transcript"), asList = TRUE)
```

**Arguments**

- `ids`: Gene IDs. A vector of character. It should be keys in txdb.
- `txdb`: An object of `TxDb`.
- `symbols`: symbol of genes.
- `type`: Output type of track, "gene" or "transcript".
- `asList`: Output a list of tracks or not. Default TRUE.

**Value**

An object of `track`

**Examples**

```r
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
ids <- c("3312", "390259", "341056", "79827")
symbols <- mget(ids, org.Hs.egSYMBOL)
geneTrack(ids, TxDb.Hsapiens.UCSC.hg19.knownGene, symbols)
```

---

**getCurTrackViewport**

*Get current track viewport*

**Description**

Get current track viewport for `addGuideLine`

**Usage**

```r
getCurTrackViewport(curViewerStyle, start, end)
```
getGeneIDsFromTxDb

Arguments

curViewerStyle  an object of trackViewerStyle
start            start position of current track
end              end position of current track

Value

an object of viewport

See Also

See Also as addGuideLine

Examples

vp <- getCurTrackViewport(trackViewerStyle(), 10000, 10200)
addGuideLine(c(10010, 10025, 10150), vp=vp)

getGeneIDsFromTxDb

get gene ids by genomic location

Description

retrieve gene ids from txdb object by genomic location.

Usage

geneIDsFromTxDb(gr, txdb)

Arguments

gr            GRanges object.
txdb           An object of TxDb.

Value

A character vector of gene ids

Examples

library(TxDb.Hsapiens.UCSC.hg19.knownGene)
gr <- parse2GRanges("chr11:122,830,799-123,116,707")
ids <- geneIDsFromTxDb(gr, TxDb.Hsapiens.UCSC.hg19.knownGene)
得到位置

**getDescription**

given a gene name, get the genomic coordinates.

**Usage**

得到位置(symbol, txdb, org)

**Arguments**

- symbol: Gene symbol
- txdb: txdb will be used to extract the genes
- org: org package name

**Examples**

```r
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
geolocation("HSPA8", TxDb.Hsapiens.UCSC.hg19.knownGene, "org.Hs.eg.db")
```

**gi2track**

**convert GInteractions to track object**

**Description**

Convert GInteractions object to track object

**Usage**

gi2track(gi, gi2)

**Arguments**

- gi: an object of GInteractions
- gi2: an object of GInteractions

**Value**

an track object

**Examples**

```r
gi <- readRDS(system.file("extdata", "nij.chr6.51120000.53200000.gi.rds", package="trackViewer"))
gi2track(gi)
```
gieStain

**Description**

Describe the colors of giemsa stain results

**Usage**

gieStain()

**Value**

A character vector of colors

**Examples**

gieStain()

---

GIoperator  

**Description**

GInteractions operations (add, subtract, multiply, divide)

**Usage**

GIoperator(gi_list, col = "score", operator = c("+", "-", "\*", "/"))

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>gi_list</td>
<td>a list of GInteractions objects</td>
</tr>
<tr>
<td>col</td>
<td>colname of metadata to be calculated</td>
</tr>
<tr>
<td>operator</td>
<td>operator, &quot;+&quot; means A + B, and so on. User-defined function also could be used.</td>
</tr>
</tbody>
</table>

**Value**

an object of GInteractions
Examples

```r
library(InteractionSet)
gr2 <- GRanges(seqnames=c("chr1", "chr1"),
ranges=IRanges(c(7,13), width=3))
gr3 <- GRanges(seqnames=c("chr1", "chr1"),
ranges=IRanges(c(1, 4), c(3, 9)))
gi <- GInteractions(gr2, gr3, score=c(1, 2))
gi2 <- GInteractions(gr2, gr3, score=c(3, 4))
GIoperator(list(gi, gi2), col="score", operator="+")
GIoperator(list(gi, gi2), col="score", operator="-")
```

gridPlot

**plot GRanges metadata**

Description

plot GRanges metadata for different types

Usage

```r
gridPlot(gr, gp, type, xscale)
```

Arguments

- `gr` an object of `GRanges` with metadata. All metadata must be numeric.
- `gp` an object of `gpar`
- `type` type of the figure, could be barplot, line, point and heatmap
- `xscale` x scale of the viewport

GRoperator

**GRanges operator**

Description

GRanges operations (add, subtract, multiply, divide)

Usage

```r
GRoperator(  
  A,
  B,
  col = "score",
  operator = c("+", ","", ", divides", ",", ", divides", ",", ", divides"),
  ignore.strand = TRUE
)
```
Arguments

A  an object of GRanges
B  an object of GRanges
col  colname of A and B to be calculated
operator  operator, "+" means A + B, and so on. User-defined function also could be used.
ignore.strand  When set to TRUE, the strand information is ignored in the overlap calculations.

Value

an object of GRanges

Examples

gr2 <- GRanges(seqnames=c("chr1", "chr1"),
ranges=IRanges(c(7,13), width=3),
strand=c("-", "-"), score=3:4)
gr3 <- GRanges(seqnames=c("chr1", "chr1"),
ranges=IRanges(c(1, 4), c(3, 9)),
strand=c("-", "-"), score=c(6L, 2L))
GRoperator(gr2, gr3, col="score", operator="+")
GRoperator(gr2, gr3, col="score", operator="-")
GRoperator(gr2, gr3, col="score", operator="*")
GRoperator(gr2, gr3, col="score", operator="/"
GRoperator(gr2, gr3, col="score", operator=mean)

ideogramPlot

plot ideogram with data

Description

plot ideogram with data for multiple chromosomes

Usage

ideogramPlot(
  ideo,
dataList,
  layout = NULL,
horiz = TRUE,
  parameterList = list(vp = plotViewport(margins = c(0.1, 4.1, 0.3, 0.1)), ideoHeight = unit(1/(1 + length(dataList)), "npc"), vgap = unit(0.3, "lines"), ylabs = "auto",
ylabsRot = ifelse(horiz, 0, 90), ylabsPos = unit(2.5, "lines"), xaxis = FALSE, yaxis = FALSE, xlab = "", types = "barplot", heights = NULL, dataColumn = "score", gps = gpar(col = "black", fill = "gray")),
colorSheme = gieStain(),
gp = gpar(fill = NA, lwd = 2),
  ...)
}
**importBam**

**Reading data from a BAM file**

**Description**

Read a `track` object from a BAM file

**Usage**

`importBam(file, file2, ranges = GRanges(), pairs = FALSE)`

**Arguments**

- **ideo** output of `loadIdeogram`
- **dataList** a `GRangesList` of data to plot
- **layout** The layout of chromosomes. Could be a list with chromosome names as its elements.
- **horiz** a logical value. If FALSE, the ideograms are drawn vertically to the left. If TRUE, the ideograms are drawn horizontally at the bottom.
- **parameterList** a list of parameters for each dataset in the `dataList`. The elements of the parameters could be `xlabs`, `ylabs`, etc. type could be `barplot`, `line`, `point`, `heatmap`.
- **colorScheme** A character vector of giemsa stain colors.
- **gp** parameters used for `grid.roundrect`.
- **...** parameters not used.

**Examples**

```r
## Not run:
ideo <- loadIdeogram("hg38")
library(rtracklayer)
library(grid)
dataList <- ideo
dataList$score <- as.numeric(dataList$gieStain)
dataList <- dataList[dataList$gieStain!="gneg"]
dataList <- GRangesList(dataList)
grid.newpage()
ideogramPlot(ideo, dataList,
layout=list("chr1", "chr2", c("chr3", "chr22"), c("chr4", "chr21"), c("chr5", "chr20"), c("chr6", "chr19"), c("chr7", "chr18"), c("chr8", "chr17"), c("chr9", "chr16"), c("chr10", "chr15"), c("chr11", "chr14"), c("chr12", "chr13"), c("chrX", "chrY")),
parameterList = list(types="heatmap", colorKeyTitle="sample1"))
## End(Not run)
```
importData

Arguments

file The path to the BAM file to read.
file2 The path to the second BAM file to read.
ranges An object of GRanges to indicate the range to be imported
pairs logical object to indicate the BAM is paired or not. See readGAlignments

Value

a track object

See Also

See Also as importScore, track, viewTracks

Examples

bamfile <- system.file("extdata", "ex1.bam", package="Rsamtools",
mustWork=TRUE)
dat <- importBam(file=bamfile, ranges=GRanges("seq1", IRanges(1, 50), strand="+"))

importData file2

Reading data from a BED or WIG file to RleList

Description

Read a track object from a BED, bedGraph, WIG or BigWig file to RleList

Usage

importData(files, format = NA, ranges = GRanges())

Arguments

files The path to the files to read.
format The format of import file. Could be BAM, BED, bedGraph, WIG or BigWig
ranges An object of GRanges to indicate the range to be imported

Value

a list of RleList.
importGInteractions

Examples

# import a BED file
bedfile <- system.file("tests", "test.bed", package="rtracklayer", mustWork=TRUE)
dat <- importData(files=bedfile, format="BED", ranges=GRanges("chr7", IRanges(127471197, 127474697)))

# import a WIG file
wigfile <- system.file("tests", "step.wig", package = "rtracklayer", mustWork=TRUE)
dat <- importData(files=wigfile, format="WIG", ranges=GRanges("chr19", IRanges(59104701, 59110920)))

# import a BigWig file
if(.Platform$OS.type!="windows"){
  # this is because we are using rtracklayer::import
  bwfile <- system.file("tests", "test.bw", package = "rtracklayer", mustWork=TRUE)
dat <- importData(files=bwfile, format="BigWig", ranges=GRanges("chr19", IRanges(1500, 2700)))
}

importGInteractions

Reading data from a ginteractions, hic, cool, or validPairs file

Description

Read a track object from a ginteractions, hic, mcool, or validPairs file

Usage

importGInteractions

Arguments

- **file**: The path to the file to read.
- **format**: The format of import file. Could be ginteractions, hic, cool or validPairs.
- **ranges**: An object of GRanges to indicate the range to be imported. For .hic file, if the length of ranges is 2, the first range will be used as anchor 1 and the second range will be used as anchor 2.
- **ignore.strand**: Ignore the strand or not when do filter. Default TRUE.
- **out**: Output format. Default is track. Possible values: track, GInteractions.
- **resolution**: Resolutions for the interaction data.
- **unit**: BP (base pair) or FRAG (fragment) (.hic file only).
- **normalization**: Type of normalization, NONE, VC, VC_SORT or KR for .hic and NONE, balanced for .cool.
- **matrixType**: Type of matrix for .hic file. Available choices are "observed", "oe", and "expected". Default is "observed".
- ... NOT used.

Value

- A track object

See Also

See Also as `listResolutions`, `listChromosomes`, `readHicNormTypes`

Examples

```r
# import a ginteractions file
# gi <- system.file("extdata", "test.ginteractions.tsv", package="trackViewer", # mustWork=TRUE)
# dat <- importGInteractions(file=gi, format="ginteractions", # ranges=GRanges("chr7", IRanges(127471197, 127474697)))

# import a hic file
if(.Platform$OS.type!="windows"){
  hic <- system.file("extdata", "test_chr22.hic", package = "trackViewer", mustWork=TRUE)
  dat <- importGInteractions(file=hic, format="hic", # ranges=GRanges("22", IRanges(1500000, 100000000)))
}

# import a cool file
cool <- system.file("extdata", "test.mcool", package = "trackViewer", mustWork=TRUE)
dat <- importGInteractions(file=cool, format="cool", resolution = 2, # ranges=GRanges("chr1", IRanges(10, 28)))

# import a validPairs file
```
importScore

Reading data from a BED or WIG file

Description

Read a track object from a BED, bedGraph, WIG or BigWig file

Usage

importScore(
  file,
  file2, 
  format = c("BED", "bedGraph", "WIG", "BigWig"),
  ranges = GRanges(),
  ignore.strand = TRUE
)

Arguments

file  
The path to the file to read.

file2  
The path to the second file to read.

format  
The format of import file. Could be BED, bedGraph, WIG or BigWig

ranges  
An object of GRanges to indicate the range to be imported

ignore.strand  
ignore the strand or not when do filter. default TRUE

Value

a track object

See Also

See Also as importBam, track, viewTracks

Examples

#import a BED file
bedfile <- system.file("tests", "test.bed", package="rtracklayer",
  mustWork=TRUE)
dat <- importScore(file=bedfile, format="BED",
  ranges=GRanges("chr7", IRanges(127471197, 127474697)))

# import a WIG file
importScSeqScore

Plot single cell RNAseq data as heatmap track for Seurat object.

Usage

importScSeqScore(
  object,
  files,
  samplenames,
  ...
  txdb,
  gene,
  id,
  idents,
  gr,
  color,
  withCoverageTrack = TRUE,
  flag = scanBamFlag(isSecondaryAlignment = FALSE, isUnmappedQuery = FALSE,
                     isNotPassingQualityControls = FALSE, isSupplementaryAlignment = FALSE)
)

Arguments

object Seurat object.
files bam file to be scanned.

Description

Plot single cell RNAseq data as heatmap track for Seurat object.
### Description

List the chromosomes available in the file.

### Usage

```r
listChromosomes(file, format = c("hic", "cool"))
```

### Arguments

- `file` character(1). File name of .hic or .cool/.mcool/.scool
- `format` character(1). File format, "hic" or "cool".

### Examples

```r
hicfile <- system.file("extdata", "test_chr22.hic", package="trackViewer")
listChromosomes(hicfile)
coolfile <- system.file("extdata", "test.mcool", package="trackViewer")
listChromosomes(coolfile, format="cool")
```
listResolutions  
*List the available resolutions*

Description
List the resolutions available in the file.

Usage
```r
dlisterResolutions(file, format = c("hic", "cool"))
```

Arguments
- **file**: character(1). File name of .hic or .cool/.mcool/.scool.
- **format**: character(1). File format, "hic" or "cool".

Examples
```r
hicfile <- system.file("extdata", "test_chr22.hic", package="trackViewer")
dlisterResolutions(hicfile)
coolfile <- system.file("extdata", "test.mcool", package="trackViewer")
dlisterResolutions(coolfile, format="cool")
```

loadIdeogram  
*load ideogram from UCSC*

Description
Download ideogram table from UCSC.

Usage
```r
loadIdeogram(genome, chrom = NULL, ranges = NULL, ...)
```

Arguments
- **genome**: Assembly name assigned by UCSC, such as hg38, mm10.
- **chrom**: A character vector of chromosome names, or NULL.
- **ranges**: A `GRanges` object with the intervals.
- `...`: Additional arguments to pass to the `GRanges` constructor.

Value
A `GRanges` object.
lolliplot

See Also

See Also as ideogramPlot

Examples

```r
## Not run:
head(loadIdeogram("hg38", chrom = "chr1"))

## End(Not run)
```

---

### lolliplot

#### Lolliplots

Description

Plot variants and somatic mutations

Usage

```r
lolliplot(
  SNP.gr,
  features = NULL,
  ranges = NULL,
  type = "circle",
  newpage = TRUE,
  ylab = TRUE,
  ylab.gp = gpar(col = "black"),
  yaxis = TRUE,
  yaxis.gp = gpar(col = "black"),
  xaxis = TRUE,
  xaxis.gp = gpar(col = "black"),
  legend = NULL,
  cex = 1,
  dashline.col = "gray80",
  jitter = c("node", "label"),
  rescale = FALSE,
  label_on_feature = FALSE,
  lollipop_style_switch_limit = 10,
  ...
)
```

Arguments

- **SNP.gr**: An object of GRanges, GRangesList or a list of GRanges. All the width of GRanges must be 1.
features  A object of GRanges, GRangesList or a list of GRanges. The metadata 'featureLayerID' are used for drawing features in different layers. See details in vignette.

ranges  A object of GRanges or GRangesList.

type  character. Could be circle, pie, pin, pie.stack or flag.

newpage  Plot in the new page or not.

ylab  Plot ylab or not. If it is a character vector, the vector will be used as ylab.

ylab.gp, xaxis.gp, yaxis.gp  An object of class gpar for ylab, xaxis or yaxis.

yaxis  Plot yaxis or not.

xaxis  Plot xaxis or not. If it is a numeric vector with length greater than 1, the vector will be used as the points at which tick-marks are to be drawn. And the names of the vector will be used to as labels to be placed at the tick points if it has names.

legend  If it is a list with named color vectors, a legend will be added.

cex  cex will control the size of circle.

dashline.col  color for the dashed line.

jitter  jitter the position of nodes or labels.

rescale  logical(1), character(1), numeric vector, or a dataframe with rescale from and to. Rescale the x-axis or not. if dataframe is used, colnames must be from.start, from.end, to.start, to.end. And the from scale must cover the whole plot region. The rescale parameter can be set as "exon" or "intron" to emphasize "exon" or "intron" region. The "exon" or "intron" can be followed with an integer e.g. "exon_80", or "intron_99". The integer indicates the total percentage of "exon" or "intron" region. Here "exon" indicates all regions in features. And "intron" indicates all flank regions of the features.

label_on_feature  Labels of the feature directly on them. Default FALSE.

lollipop_style_switch_limit  The cutoff value for lollipop style for the 'circle' type. If the max score is greater than this cutoff value, trackViewer will only plot one shape at the highest score. Otherwise trackViewer will draw the shapes like 'Tanghulu'.

Details

In SNP.gr and features, metadata of the GRanges object will be used to control the color, fill, border, alpha, shape, height, cex, dashline.col, data source of pie if the type is pie. And also the controls for labels by name the metadata start as label.parameter.<properties>, and for node labels by name the metadata start as node.label.<properties>, such as label.parameter.rot, label.parameter.gp. The parameter is used for grid.text or plotMotifLogoA. The metadata 'featureLayerID' for features are used for drawing features in different layers. The metadata 'SNPsideID' for SNP.gr are used for determining the side of lollipops. And the 'SNPsideID' could only be 'top' or 'bottom'.

...  not used.
### Examples

```r
SNP <- c(10, 100, 105, 108, 400, 410, 420, 600, 700, 805, 840, 1400, 1402)
x <- sample.int(100, length(SNP))
SNP.gr <- GRanges("chr1", IRanges(SNP, width=1, names=paste0("snp", SNP)),
    value1=x, value2=100-x)
SNP.gr$color <- rep(list(c("red", "blue")), length(SNP))
SNP.gr$border <- sample.int(7, length(SNP), replace=TRUE)
features <- GRanges("chr1", IRanges(c(1, 501, 1001),
    width=c(120, 500, 405),
    names=paste0("block", 1:3)),
    color="black",
    fill=c("#FF8833", "#51C6E6", "#DFA32D"),
    height=c(0.1, 0.05, 0.08),
    label.parameter.rot=45)
lolliplot(SNP.gr, features, type="pie")
```

---

### Description

plot graph for GInteractions

### Usage

```r
loopBouquetPlot(
    gi,
    range,
    feature.gr,
    atacSig,
    label_region = FALSE,
    show_edges = TRUE,
    show_cluster = TRUE,
    lwd.backbone = 2,
    col.backbone = "gray",
    lwd.maxAtacSig = 8,
    reverseATACSig = TRUE,
    col.backbone_background = "gray70",
    lwd.gene = 2,
    lwd.nodeCircle = 1,
    col.nodeCircle = "#DDDDDD25",
    lwd.edge = 2,
    col.edge = "gray80",
    coor_mark_interval = 1e+05,
    col.coor = "black",
    show_coor = TRUE,
    coor_tick_unit = 1000,
)```
label_gene = TRUE,
col.tension_line = "black",
lwd.tension_line = 1,
length.arrow = NULL,
safe_text_force = 3,
method = 1,
doReduce = FALSE,
...
)

Arguments

- **gi** An object of `GInteractions`
- **range** The region to plot. An object of `GRanges`
- **feature.gr** The annotation features to be added. An object of `GRanges`.
- **atacSig** The ATAC-seq signals. An object of `GRanges` with scores or an object of `track`.
- **label_region** Label the region node or not.
- **show_edges** Plot the interaction edges or not.
- **show_cluster** Plot the cluster background or not.
- **lwd.backbone**, **lwd.gene**, **lwd.nodeCircle**, **lwd.edge**, **lwd.tension_line**, **lwd.maxAtacSig** Line width for the linker, gene, interaction node circle, the dashed line of interaction edges, the tension line and the maximal reversed ATAC signal.
- **col.backbone**, **col.backbone_background**, **col.nodeCircle**, **col.edge**, **col.tension_line**, **col.coor** Color for the DNA chain, the compact DNA chain, the node circle, the linker, the tension line and the coordinates marker.
- **reverseATACSig** Plot the ATAC-seq signals in reverse values.
- **coor_mark_interval** The coordinates marker interval. Numeric(1). Set to 0 to turn it off. The default value 1e5 means show coordinates every 0.1M bp.
- **show_coor** Plot ticks in the line to show the DNA compact tension.
- **coor_tick_unit** The bps for every ticks. Default is 1K.
- **label_gene** Show gene symbol or not.
- **length.arrow** Length of the edges of the arrow head (in inches).
- **safe_text_force** The loops to avoid the text overlapping.
- **method** Plot method. Could be 1 or 2.
- **doReduce** Reduce the GInteractions or not.
- **...** Parameter will be passed to `layout_with_fr`. 
Examples

```r
library(InteractionSet)
gi <- readRDS(system.file("extdata", "gi.rds", package="trackViewer"))
range <- GRanges("chr2", IRanges(234500000, 235000000))
library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
feature.gr <- genes(TxDB.Hsapiens.UCSC.hg19.knownGene)
feature.gr <- subsetByOverlaps(feature.gr, range(regions(gi)))
symbols <- mget(feature.gr$gene_id, org.Hs.egSYMBOL, ifnotfound=NA)
feature.gr$label[lengths(symbols)==1] <- unlist(symbols[lengths(symbols)==1])
feature.gr$col <= sample(1:7, length(feature.gr), replace=TRUE)
feature.gr$type <- sample(c("cRE", "gene"), length(feature.gr), replace=TRUE,
                        prob=c(0.1, 0.9))
loopBouquetPlot(gi, range, feature.gr)
```

---

**optimizeStyle**

*Optimize the style of plot*

**Description**

Automatic optimize the style of trackViewer

**Usage**

```r
optimizeStyle(trackList, viewerStyle = trackViewerStyle(), theme = NULL)
```

**Arguments**

- `trackList`: An object of `trackList`
- `viewerStyle`: An object of `trackViewerStyle`
- `theme`: A character string. Could be "bw", "col" or "safe".

**Value**

A list of a `trackList` and a `trackViewerStyle`

**See Also**

See Also as `viewTracks`

**Examples**

```r
extdata <- system.file("extdata", package="trackViewer",
                      mustWork=TRUE)
files <- dir(extdata, ".wig")
tracks <- lapply(paste(extdata, files, sep="/"),
                 importScore, format="WIG")
```
parseWIG

convert WIG format track to BED format track

Description
convert WIG format track to BED format track for a given range

Usage
parseWIG(trackScore, chrom, from, to)

Arguments
trackScore an object of track with WIG format
chrom sequence name of the chromosome
from start coordinate
to end coordinate

parse2GRanges
parse text into GRanges

Description
parse text like "chr13:99,443,451-99,848,821:-" into GRanges

Usage
parse2GRanges(text)

Arguments

Value
an object of GRanges

Examples
parse2GRanges("chr13:99,443,451-99,848,821:-")
plotGRanges

Value
an object of track

Examples

```r
extdata <- system.file("extdata", package="trackViewer", mustWork=TRUE)
repA <- importScore(file.path(extdata, "cpsf160.repA_.wig"),
                    file.path(extdata, "cpsf160.repA_+.wig"),
                    format="WIG")
strand(repA$dat) <- "-
strand(repA$dat2) <- "+
parseWIG(repA, chrom="chr11", from=122929275, to=122930122)
```

plotGRanges  plot GRanges data

Description
A function to plot GRanges data for given range

Usage

```r
plotGRanges(
  ..., 
  range = GRanges(), 
  viewerStyle = trackViewerStyle(), 
  autoOptimizeStyle = FALSE, 
  newpage = TRUE 
)
```

Arguments

```r
... one or more objects of GRanges
range an object of GRanges
viewerStyle an object of trackViewerStyle
autoOptimizeStyle should use optimizeStyle to optimize style
newpage should be draw on a new page?
```

Value
An object of viewport for addGuideLine

See Also
See Also as addGuideLine, addArrowMark
plotIdeo

Examples

gr1 <- GRanges("chr1", IRanges(1:50, 51:100))
gr2 <- GRanges("chr1", IRanges(seq(from=10, to=80, by=5),
        seq(from=20, to=90, by=5)))
vp <- plotGRanges(gr1, gr2, range=GRanges("chr1", IRanges(1, 100)))
addGuideLine(guideLine=c(5, 10, 50, 90), col=2:5, vp=vp)

gr <- GRanges("chr1", IRanges(c(1, 11, 21, 31), width=9),
        score=c(5, 10, 5, 1))
plotGRanges(gr, range=GRanges("chr1", IRanges(1, 50)))

Description

plot ideogram for one chromosome

Usage

plotIdeo(
  ideo,
  chrom = seqlevels(ideo)[[1]],
  colorSheme = gieStain(),
  gp = gpar(fill = NA),
  ...
)

Arguments

ideo output of loadIdeogram.
chrom A length 1 character vector of chromosome name.
colorSheme A character vector of giemsa stain colors.
gp parameters used for grid.roundrect.
... parameters not used.

Examples

# Not run:
ideo <- loadIdeogram("hg38")
library(grid)
grid.newpage()
plotIdeo(ideo)

# End(Not run)
plotOneIdeo

plot ideogram with data for one chromosome

Description

plot ideogram with data for one chromosome

Usage

plotOneIdeo(
    ideo,
    dataList,
    parameterList = list(wp = plotViewport(margins = c(0.1, 4.1, 1.1, 0.1)), ideoHeight = 
        unit(1/(1 + length(dataList)), "npc"), vgap = unit(1, "lines"), ylabs = 
        seqlevels(ideo)[1], ylabsRot = 90, ylabsPos = unit(2.5, "lines"), xaxis = FALSE, 
        yaxis = FALSE, xlab = "", types = "barplot", heights = NULL, dataColumn = "score", 
        gpl = gpar(col = "black", fill = "gray"),
        chrom = seqlevels(ideo)[1],
        colorSheme = gieStain(),
        gp = gpar(fill = NA, lwd = 2),
        ...
)

Arguments

ideo output of loadIdeogram.
dataList a GRangesList of data to plot.
parameterList a list of parameters for each dataset in the dataList. The elements of the parameters could be xlabs, ylabs, etc. type could be barplot, line, point, heatmap.
chrom A length 1 character vector of chromosome name.
colorSheme A character vector of giesma stain colors.
gp parameters used for grid.roundrect.

Examples

## Not run:
ideo <- loadIdeogram("hg38")
library(rtracklayer)
library(grid)
dataList <- ideo[seqnames(ideo) %in% "chr1"]
dataList$score <- as.numeric(dataList$gieStain)
dataList <- dataList[dataList$gieStain!="gneg"]
dataList <- GRangesList(dataList, dataList)
grid.newpage()
plotOneIdeo(ideo, dataList, chrom="chr1")
## End(Not run)

---

### pos-class

**Class**: "pos"

### Description

An object of class "pos" represents a point location

### Slots

- **x**: A numeric value, indicates the x position
- **y**: A numeric value, indicates the y position
- **unit**: "character" specifying the units for the corresponding numeric values. See **unit**

---

### reduce,GInteractions-method

*Reduce method for 'GInteractions'*

### Description

Reduce returns an object of the same type as x containing reduced ranges for each distinct (seqname, strand) pairing.

### Usage

```r
## S4 method for signature 'GInteractions'
reduce(x, min.gapwidth = 1L, ignore.strand = TRUE, ...)
```

### Arguments

- **x**: GInteractions object.
- **min.gapwidth**: Ranges separated by a gap of at least min.gapwidth positions are not merged.
- **ignore.strand**: TRUE or FALSE. Whether the strand of the input ranges should be ignored or not.
- **...**: Not used.
Examples

```r
## Not run:
library(InteractionSet)
gi <- readRDS(system.file("extdata", "gi.rds", package="trackViewer"))
reduce(head(gi, n=20))

## End(Not run)
```

---

**trackList-class**  
*List of tracks*

Description

An extension of List that holds only `track` objects.

Usage

```r
## S4 replacement method for signature 'trackList'
seqlevelsStyle(x) <- value

trackList(..., heightDist = NA)
```

Arguments

- `x`  
  *trackList* object.

- `value`  
  values to be assigned.

- `...`  
  Each tracks in ... becomes an element in the new trackList, in the same order.  
  This is analogous to the list constructor, except every argument in ... must be derived from `track`.

- `heightDist`  
  A vector or NA to define the height of each track.

See Also

- `track`.
**trackStyle-class**  
*Class* "trackStyle"

**Description**

An object of class "trackStyle" represents track style.

An object of class "track" represents scores of a given track.

**Usage**

```r
## S4 method for signature 'track'
seqlevels(x)

## S4 method for signature 'track'
seqlevelsStyle(x)

## S4 replacement method for signature 'track'
seqlevelsStyle(x) <- value

## S4 method for signature 'track'
show(object)

## S4 method for signature 'track'
x$name

## S4 replacement method for signature 'track'
x$name <- value

setTrackStyleParam(ts, attr, value)

## S4 method for signature 'track,character'
setTrackStyleParam(ts, attr, value)

setTrackXscaleParam(ts, attr, value)

## S4 method for signature 'track,character'
setTrackXscaleParam(ts, attr, value)

setTrackYaxisParam(ts, attr, value)

## S4 method for signature 'track,character'
setTrackYaxisParam(ts, attr, value)
```

**Arguments**

- **x**  
  An object of trackStyle
value values to be assigned.
object an object of trackStyle.
name slot name of trackStyle
ts An object of track.
attr the name of slot of trackStyle object to be changed.

Details

The attr of setTrackXscaleParam could not only be a slot of xscale, but also be position. If the attr is set to position, value must be a list of x, y and label. For example setTrackXscaleParam(track, attr="position", value=list(x=122929675, y=4, label=500))

Slots

tracktype "character" track type, could be peak or cluster. Default is "peak". "cluster" is not supported yet. For interaction data, it could be "heatmap" or "link".
color "character" track color. If the track has dat and dat2 slot, it should have two values.
NAcolor "character" NA color for interactionData.
breaks "numeric" breaks for color keys of interactionData.
height "numeric" track height. It should be a value between 0 and 1
marginTop "numeric" track top margin
marginBottom "numeric" track bottom margin
xscale object of xscale, describe the details of x-scale
yaxis object of yaxisStyle, describe the details of y-axis
ylim "numeric" y-axis range
ylabpos "character", ylab postion, ylabpos should be 'left', 'right', 'topleft', 'bottomleft', 'topright', 'bottomright', 'abovebaseline' or 'underbaseline'. For gene type track, it also could be 'upstream' or 'downstream'
ylablas "numeric" y lable direction. It should be a integer 0-3. See par:las
ylabgp A "list" object, It will convert to an object of class gpar. This is basically a list of graphical parameter settings of y-label.
dat Object of class GRanges the scores of a given track. It should contain score metadata.
dat2 Object of class GRanges the scores of a given track. It should contain score metadata. When dat2 and dat is paired, dat will be drawn as positive value where dat2 will be drawn as negative value (-1 * score)
type The type of track. It could be 'data', 'gene', 'transcript', 'scSeq', 'lollipopData' or 'interactionData'.
format The format of the input. It could be "BED", "bedGraph", "WIG", "BigWig" or "BAM"
style Object of class trackStyle
name unused yet
See Also

Please try to use `importScore` and `importBam` to generate the object.

Examples

```r
extdata <- system.file("extdata", package="trackViewer",
mustWork=TRUE)
fox2 <- importScore(file.path(extdata, "fox2.bed"), format="BED")
setTrackStyleParam(fox2, "color", c("red","green"))
setTrackXscaleParam(fox2, "gp", list(cex=.5))
setTrackYaxisParam(fox2, "gp", list(col="blue"))
fox2$dat <- GRanges(score=numeric(0))
```

trackViewerStyle-class

`Class "trackViewerStyle"`

Description

An object of class "trackViewerStyle" represents track viewer style.

Usage

```r
trackViewerStyle(...)
setTrackViewerStyleParam(tvs, attr, value)
```

## S4 method for signature 'trackViewerStyle,character'
setTrackViewerStyleParam(tvs, attr, value)

Arguments

... Each argument in ... becomes an slot in the new trackViewerStyle.
tvs An object of trackViewerStyle.
attr the name of slot to be changed.
value values to be assigned.

Slots

```
margin "numeric", specify the bottom, left, top and right margin.
xlas "numeric", label direction of x-axis mark. It should be a integer 0-3. See `par:las`
xgp A "list", object. It will convert to an object of class `gpar`. This is basically a list of graphical parameter settings of x-axis. For y-axis, see `yaxisStyle`
xaxis "logical", draw x-axis or not
xat "numeric", the values will be passed to grid.xaxis as 'at' parameter.
```
viewGene

xlabel "character", the values will be passed to grid.xaxis as 'label' parameter.
autolas "logical" automatic determine y label direction
flip "logical" flip the x-axis or not, default FALSE

Examples

tvs <- trackViewerStyle()
setTrackViewerStyleParam(tvs, "xaxis", TRUE)

viewGene  plot tracks based on gene name

Description
given a gene name, plot the tracks.

Usage

viewGene(
  symbol,
  filenames,
  format,
  txdb,
  org,
  upstream = 1000,
  downstream = 1000,
  anchor = c("gene", "TSS"),
  plot = FALSE
)

Arguments

  symbol    Gene symbol
  filenames files used to generate tracks
  format    file format used to generate tracks
  txdb      txdb will be used to extract the genes
  org       org package name
  upstream  upstream from anchor
  downstream downstream from anchor
  anchor    TSS, or gene
  plot      plot the tracks or not.

Value

an invisible list of a trackList, a trackViewerStyle and a GRanges
Examples

library(TxDb.Hsapiens.UCSC.hg19.knownGene)
library(org.Hs.eg.db)
extdata <- system.file("extdata", package="trackViewer", mustWork=TRUE)
filename = file.path(extdata, "fox2.bed")
optSty <- viewGene("HSPA8", filenames=filename, format="BED",
                    txdb=TxDB.Hsapiens.UCSC.hg19.knownGene,
                    org="org.Hs.eg.db")

viewTracks
plot the tracks

Description

A function to plot the data for given range

Usage

viewTracks(
  trackList, chromosome, start, end, strand,
  gr = GRanges(), ignore.strand = TRUE,
  viewerStyle = trackViewerStyle(),
  autoOptimizeStyle = FALSE, newpage = TRUE,
  operator = NULL, smooth = FALSE,
  lollipop_style_switch_limit = 10
)

Arguments

trackList an object of trackList
chromosome chromosome
start start position
end end position
strand strand
gr an object of GRanges
ignore.strand ignore the strand or not when do filter. default TRUE
viewerStyle an object of trackViewerStyle
autoOptimizeStyle

should use optimizeStyle to optimize style

newpage

should be draw on a new page?

operator

operator, could be +, -, *, /, %, ^, %%, and NA. "-" means dat - dat2, and so on. NA means do not apply any operator. Note: if multiple operator is supplied, please make sure the length of operator keep same as the length of trackList.

smooth

logical(1) or numeric(). Plot smooth curve or not. If it is numeric, eg n, mean of nearby n points will be used for plot. If it is numeric, the second number will be the color. Default color is 2 (red).

lollipop_style_switch_limit

The cutoff value for lollipop style for the 'circle' type. If the max score is greater than this cutoff value, trackViewer will only plot one shape at the highest score. Otherwise trackViewer will draw the shapes like 'Tanghulu'.

Value

An object of viewport for addGuideLine

See Also

See Also as addGuideLine, addArrowMark

Examples

```r
extdata <- system.file("extdata", package="trackViewer",
mustWork=TRUE)
files <- dir(extdata, ".wig")
tracks <- lapply(paste(extdata, files, sep="/"),
importScore, format="WIG")
fox2 <- importScore(paste(extdata, "fox2.bed", sep="/"), format="BED")
dat <- coverageGR(fox2@dat)
fox2@dat <- dat[strand(dat)=="+"]
fox2@dat2 <- dat[strand(dat)=="-"]
gr <- GRanges("chr11", IRanges(122929275, 122930122), strand="-"
viewTracks(trackList(track=tracks, fox2=fox2), gr=gr, autoOptimizeStyle=TRUE)
```

---

**Description**

An object of class "xscale" represents x-scale style.
Slots

from  A \texttt{pos} class, indicates the start point position of x-scale.
to   A \texttt{pos} class, indicates the end point position of x-scale.
label "character" the label of x-scale
gp   A "list" object, It will convert to an object of class \texttt{gpar}. This is basically a list of graphical parameter settings of x-scale.
draw A "logical" value indicating whether the x-scale should be draw.

---

\texttt{yaxisStyle-class} Class "yaxisStyle"

Description

An object of class "yaxisStyle" represents y-axis style.

Slots

at  "numeric" vector of y-value locations for the tick marks
label "logical" value indicating whether to draw the labels on the tick marks.
gp  A "list" object, It will convert to an object of class \texttt{gpar}. This is basically a list of graphical parameter settings of y-axis.
draw A "logical" value indicating whether the y-axis should be draw.
main A "logical" value indicating whether the y-axis should be draw in left (TRUE) or right (FALSE).
Index

$, track-method (trackStyle-class), 38
$<-, track-method (trackStyle-class), 38
addArrowMark, 4, 5, 6, 33, 43
addGuideline, 5, 5, 6, 14, 33, 43
addInteractionAnnotation, 6
ARA, 7
arrow, 5
browseTracks, 8
browseTracks-shiny, 9
browseTracksOutput
(browseTracks-shiny), 9
coverage, 10
coverageGR, 9
dandelion.plot, 10
gAlignmentPairs, 9
gAlignments, 9
geneModelFromTxdb, 11
geneTrack, 13
getCurTrackViewport, 5, 13
getGeneIDsFromTxDb, 14
getLocation, 15
gi2track, 15
gieStain, 16
GInteractions, 6, 7, 30
GIoperator, 16
gpar, 17, 39, 40, 44
GRanges, 6, 8–10, 17, 20, 22, 23, 26–28, 30, 32, 33, 39, 41, 42
GRangesList, 10, 19, 27, 28, 35
grid.lines, 6
grid.polygon, 6
grid.roundrect, 19, 34, 35
grid.text, 6, 28
gridPlot, 17
GRIoperator, 17
ideogramPlot, 18, 27
importBam, 12, 19, 23, 40
importData, 20
importGInteractions, 7, 21
importScore, 12, 20, 23, 40
importScSeqScore, 24
layout_with_fr, 30
listChromosomes, 22, 25
listResolutions, 22, 26
loadIdeogram, 19, 26, 34, 35
lollipop, 27
loopBouquetPlot, 29
numeric, 36
optimizeStyle, 31, 33, 43
par, 39, 40
parse2GRanges, 32
parseWIG, 32
plotGRanges, 33
plotIdeo, 34
plotMotifLogoA, 28
plotOneIdeo, 35
pos, 44
pos (pos-class), 36
pos-class, 36
Ranges, 26
readGAlignments, 20, 25
readGAlignmentsList, 25
readHicNormTypes, 22
reduce,GInteractions
(reduce,GInteractions-method), 36
reduce,GInteractions-method, 36
renderbrowseTracks
(browseTracks-shiny), 9
RleList, 20