Package ‘trackViewer’

January 15, 2017

Type Package
Title A bioconductor package with minimalist design for drawing elegant tracks or lollipop plot
Version 1.10.2
Date 2016-12-01
Author Jianhong Ou, Yong-Xu Wang, Lihua Julie Zhu
Maintainer Jianhong Ou <jianhong.ou@umassmed.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 (>= 3.1.0), grDevices, methods, GenomicRanges, grid
Imports GenomicAlignments, GenomicFeatures, Gviz, phapply, Rsamtools, rtracklayer, S4Vectors, scales, tools, IRanges, AnnotationDbi, grImport
Suggests biomaRt, TxDb.Hsapiens.UCSC.hg19.knownGene, RUnit, org.Hs.eg.db, BiocGenerics, BiocStyle, knitr, VariantAnnotation
biocViews Visualization
VignetteBuilder knitr
NeedsCompilation no

R topics documented:

 trackViewer-package .................................................. 2
  addArrowMark ......................................................... 3
  addGuideLine ......................................................... 4
  coverageGR .......................................................... 5
  dandelion.plot ....................................................... 6
  geneModelFromTxdb ............................................... 7
  getCurTrackViewport .............................................. 8
  GRoperator .......................................................... 9
  importBam .......................................................... 9
  importData ........................................................ 10
  importScore ........................................................ 11
  lolliplot .......................................................... 12
  optimizeStyle ....................................................... 14
trackViewer-package

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.

Details

Package: trackViewer
Type: Package
Version: 1.0
Date: 2013-10-18
License: Artistic-2.0

This package is minimal designed to plot figure for publication.

Author(s)

Jianhong Ou, Julie Lihua Zhu
Maintainer: Jianhong Ou <jianhong.ou@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")
```
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(.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: bottomright, 3: topright, 4: 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

Author(s)
Jianhong Ou

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")
}
```

Description

A function to add lines for emphasizing the positions

Usage

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

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>guideLine</td>
<td>The genomic coordinates to draw the lines</td>
</tr>
<tr>
<td>col</td>
<td>A vector for the line color</td>
</tr>
<tr>
<td>lty</td>
<td>A vector for the line type</td>
</tr>
<tr>
<td>lwd</td>
<td>A vector for the line width</td>
</tr>
<tr>
<td>vp</td>
<td>A Grid viewport object. It must be output of viewTracks</td>
</tr>
</tbody>
</table>

Value

NULL

Author(s)
Jianhong Ou

See Also
See Also as getCurTrackViewport, addArrowMark, viewTracks
Examples

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

---

### Description

calculate coverage for `GRanges`, `GAlignments` or `GAlignmentPairs`

### Usage

```r
coverageGR(gr)
```

### Arguments

- `gr`: an object of `RGanges`, `GAlignments` or `GAlignmentPairs`

### Value

an object of `GRanges`

### Author(s)

Jianhong Ou

### 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)
```
dandelion.plot

dandelion.plot  dandelion.plots

Description
Plot variants and somatic mutations

Usage
dandelion.plot(SNP.gr, features=NULL, ranges=NULL,
type=c("fan", "circle", "pie", "pin"),
newpage=TRUE, ylab=TRUE,
xaxis=TRUE, legend=NULL,
cex=1, maxgaps=1/50, ...)

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.
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.
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.
maxgaps  maxgaps between the stem of dandelions. It is calculated by the width of plot
region devided by maxgaps.
...  not used.

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.

Value
NULL

Author(s)
Jianhong Ou
geneModelFromTxdb

Prepare gene model from an object of TxDb

Description

Generate an object of track for viewTracks by given parameters.

Usage

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

Arguments

taxdb          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

An object of track

Author(s)

Jianhong Ou

See Also

See Also as importScore, importBam, viewTracks
getCurTrackViewport  

Get current track viewport

Description

Get current track viewport for addGuideLine

Usage

getCurTrackViewport(curViewerStyle, start, end)

Arguments

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

Value

  an object of viewport

Author(s)

  Jianhong Ou

See Also

  See Also as addGuideLine

Examples

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

GRanges operator

Description
GRanges operations (add, subtract, multiply, divide)

Usage
GRoperator(A, B, col="score", operator = c("+", "-", "x", "/", "^", "%%"))

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.

Value
an object of GRanges

Author(s)
Jianhong Ou

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="x")
GRoperator(gr2, gr3, col="score", operator="/")

importBam
Reading data from a BAM file

Description
Read a track object from a BAM file

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

Author(s)

Jianhong Ou

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      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.

Author(s)

Jianhong Ou
importScore

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)))
}

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

Author(s)

Jianhong Ou
See Also

See Also as `importBam`, `track.viewTracks`

Examples

```r
# 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
wigfile <- system.file("tests", "step.wig", package = "rtracklayer", mustWork=TRUE)
dat <- importScore(file=wigfile, format="WIG")

# 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 <- importScore(file=bwfile, format="BigWig")
}

# import 2 file
wigfile1 <- system.file("extdata", "cpsf160.repA_+.wig", package="trackViewer", mustWork=TRUE)
wigfile2 <- system.file("extdata", "cpsf160.repA_-.wig", package="trackViewer", mustWork=TRUE)
dat <- importScore(wigfile1, wigfile2, format="WIG", ranges=GRanges("chr11", IRanges(122817703, 122889073)))
```

---

**lolliplot**

*Lolliplots*

**Description**

Plot variants and somatic mutations

**Usage**

```r
lolliplot(SNP.gr, features = NULL, ranges = NULL,
  type = c("circle", "pie", "pin",
    "pie.stack"),
  newpage = TRUE,
  ylab=TRUE, yaxis=TRUE, xaxis=TRUE,
  legend=NULL, cex=1,
  dashline.col="gray80", ...)
```

**Arguments**

- `SNP.gr`: A 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 or pie.stack.

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.

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.

...      not used.

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. And also the controls for labels by name the metadata start as label.parameter.<properties> such as label.parameter.rot, label.parameter.gp. The parameter is used for grid.text. 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’.

Value

NULL

Author(s)

Jianhong Ou

Examples

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")
optimizeStyle

Optimize the style of plot

Description

Automatic optimize the style of trackViewer

Usage

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

Arguments

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

Value

A list of a trackList and a trackViewerStyle

Author(s)

Jianhong Ou

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")
re <- optimizeStyle(trackList(tracks))
trackList <- re$tracks
viewerStyle <- re$style
```
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

**Author(s)**

Jianhong Ou

**Examples**

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

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

**Value**

an object of track
plotGRanges

Author(s)
Jianhong Ou

See Also
track

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

Arguments

... 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

Author(s)
Jianhong Ou

See Also
See Also as addGuideLine, addArrowMark
Examples

```r
g1 <- GRanges("chr1", IRanges(1:50, 51:100))
g2 <- GRanges("chr1", IRanges(seq(from=10, to=80, by=5), seq(from=20, to=90, by=5)))
vp <- plotGRanges(g1, g2, 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)))
```

pos-class

Class "pos"

Description

An object of class "pos" represents a point location

Objects from the Class

Objects can be created by calls of the form `new("pos", x, y, unit)`

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`

track-class

Class "track"

Description

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

Usage

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

Arguments

- `ts` An object of track.
- `attr` the name of slot of `trackStyle` object to be changed.
- `value` values to be assigned.
Objects from the Class

Objects can be created by calls of the form `new("track", dat, dat2, type, format, style, name).

Slots

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' or 'gene'.
format The format of the input. It could be "BED", "bedGraph", "WIG", "BigWig" or "BAM"
styile Object of class trackStyle
name unused yet

Methods

setTrackStyleParam change the slot values of trackStyle object for an object of track
setTrackXscaleParam change the xscale slot values for an object of track
setTrackYaxisParam change the yaxisStyle values for an object of track
$.<- Get or set the slot of track
show show the details of track

See Also

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

Examples

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))

trackList-class  List of tracks

Description

An extension of List that holds only track objects.

constructor

trackList(..., heightDist=NA): 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. The heightDist is 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.

Objects from the Class

Objects can be created by calls of the form `new("trackStyle", tracktype, color, height, marginTop, marginBottom, xscale, yaxis, ylim, ylabpos, ylablas, ylabgp)`.

Slots

- **tracktype** "character" track type, could be peak or cluster. Default is "peak". "cluster" is not supported yet.
- **color** "character" track color. If the track has dat and dat2 slot, it should have two values.
- **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 position, ylabpos should be 'left', 'right', 'topleft', 'bottomleft', 'topright' or 'bottomright'.
- **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.

trackViewerStyle-class

Class "trackViewerStyle"

Description

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

Usage

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

Arguments

- **tvs** An object of `trackViewerStyle`.
- **attr** the name of slot to be changed.
- **value** values to be assigned.
Objects from the Class

Objects can be created by calls of the form `new("trackViewerStyle", margin, xlas, ...

constructor

trackViewerStyle(...): Each argument in ... becomes an slot in the new trackViewerStyle.

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

autolas "logical" automatic determine y label direction

flip "logical" flip the x-axis or not, default FALSE

Methods

setTrackViewerStyleParam change the slot values of an object of trackViewerStyle

Examples

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

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)

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 +, -, *, /, ^, %%. "-" means dat - dat2, and so on.

Value
An object of viewport for addGuideLine

Author(s)
Jianhong Ou

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")
tracks <- lapply(tracks, function(.ele) {strand(.ele@dat) <- "-"; .ele})
fox2 <- importScore(paste(extdata, "fox2.bed", sep="/"), format="BED")
dat <- coverageGR(fox2@dat)
dat2 <- dat[strand(dat)=="-"]
gr <- GRanges("chr11", IRanges(122929275, 122930122), strand="-")
viewTracks(trackList(tracks, fox2), gr=gr, autoOptimizeStyle=TRUE)
```

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

Objects from the Class
Objects can be created by calls of the form new("xscale", from, to, label, gp, draw).

Slots

- from: A pos class, indicates the start point position of x-scale.
- to: A 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 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.
Class "yaxisStyle"

Description

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

Objects from the Class

Objects can be created by calls of the form `new("yaxisStyle", at, label, gp, draw, main)`.

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 `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

*Topic `textscitildemisc`
  addArrowMark, 3

*Topic `classes`
  pos-class, 17
  track-class, 17
  trackList-class, 18
  trackStyle-class, 19
  trackViewerStyle-class, 19
  xscale-class, 21
  yaxisStyle-class, 22

*Topic `importData`
  coverageGR, 5
  geneModelFromTxdb, 7
  importBam, 9
  importData, 10
  importScore, 11

*Topic `misc`
  addGuideLine, 4
  dandelion.plot, 6
  getCurTrackViewport, 8
  GRoperator, 9
  lollyplot, 12
  parse2GRanges, 15
  parseWIG, 15

*Topic `package`
  trackViewer-package, 2

*Topic `plot`
  optimizeStyle, 14
  plotGRanges, 16
  viewTracks, 20
  $,track-method (track-class), 17
  $<-,track-method (track-class), 17

addArrowMark, 3, 4, 16, 21
addGuideLine, 4, 4, 8, 16, 21
arrow, 4

coverage, 5
coverageGR, 5
dandelion.plot, 6
gAlignmentPairs, 5
gAlignments, 5

geneModelFromTxdb, 7
gCurTrackViewport, 4, 8
gpar, 19–22
GRanges, 5, 6, 10–13, 15, 16, 18, 20
GRangesList, 6, 12, 13
grid.text, 13
GRoperator, 9
importBam, 7, 9, 12, 18
importData, 10
importScore, 7, 10, 11, 18
lollyplot, 12
numeric, 17
optimizeStyle, 14, 16, 21
par, 19, 20
parse2GRanges, 15
parseWIG, 15
plotGRanges, 16
pos, 21
pos(pos-class), 17
pos-class, 17
readGAlignments, 10
RleList, 10

setTrackStyleParam (track-class), 17
setTrackStyleParam, track, character, ANY-method (track-class), 17
setTrackStyleParam, track, character-method (track-class), 17
setTrackViewerStyleParam (trackViewerStyle-class), 19
setTrackViewerStyleParam, trackViewerStyle, character, ANY-method (trackViewerStyle-class), 19
setTrackViewerStyleParam, trackViewerStyle, character-method (trackViewerStyle-class), 19
setTrackXscaleParam (track-class), 17
setTrackXscaleParam, track, character, ANY-method (track-class), 17
setTrackXscaleParam, track, character-method (track-class), 17
INDEX

setTrackYaxisParam(track-class), 17
setTrackYaxisParam(track, character, ANY-method
(track-class), 17
setTrackYaxisParam(track, character-method
(track-class), 17
show(track-method (track-class), 17

track, 7, 9–12, 16, 18
track (track-class), 17
track-class, 17
trackList, 14, 20
trackList(trackList-class), 18
trackList-class, 18
trackStyle, 17, 18
trackStyle (trackStyle-class), 19
trackStyle-class, 19
trackViewer (trackViewer-package), 2
trackViewer-package, 2
trackViewerStyle, 8, 14, 16, 21
trackViewerStyle
(trackViewerStyle-class), 19
trackViewerStyle-class, 19
TxDb, 7

unit, 17

viewport, 8, 16, 21
viewTracks, 3, 4, 7, 10, 12, 14, 20

xscale, 18, 19
xscale (xscale-class), 21
xscale-class, 21

yaxisStyle, 18–20
yaxisStyle (yaxisStyle-class), 22
yaxisStyle-class, 22