RedeR

March 24, 2012

`PluginBuilder-class`
Class 'PluginBuilder'

Description

A class providing methods to build plugins to RedeR application.

Slots

`title`**:** A character string representing the plugin name.

`allMethods`**:** List of all plugin methods wrapped as R functions.

`allAddons`**:** List of all additional R expression wrapped as R functions.

Author(s)

Mauro Castro

Examples

```
#Initiates a new method, say mt1
mt1<-function(){"#Your code here!"}
#Initiates a plugin skeleton
plugin <- PluginBuilder ( title = "MyPlugin", allMethods = list(mt1=mt1) )
```

`PluginBuilder` *The constructor for the PluginBuilder class.*

Description

Constructor to build RedeR plugins.

Usage

```
PluginBuilder(title='plugin', allMethods, allAddons=NULL)
```

Arguments

`title`	A character string representing the plugin name.
`allMethods`	List of all plugin methods wrapped as R functions (does not accept arguments).
`allAddons`	List of all additional expressions wrapped as R functions (accept arguments).

Details

RedeR plug-ins have two main sections: methods and add-ons. The 'methods' section can be regarded as the plug-in trigger. When installed in the app, this trigger is used to start a given analysis by unfolding the R expressions wrapped in the methods. Add-ons use the same strategy, but remains hidden in the app – and it is optional.

Value

Build a new plugin.

Note

The 'allMethods' section does not accept arguments. Formal functions can be passed to add-ons as additional arguments.

Author(s)

Mauro Castro

See Also

`submitPlugin` `updatePlugins` `deletePlugin` `pluginParser` `dynwin`

Examples

```
#Wrap up a new method into a function
mt1 <- function()
{
  rdp   <- RedPort('MyPort')
  g     <- getGraph(rdp)
  dg    <- degree.distribution(g)
  dynwin(rdp)                              #creates a RedeR java graphics device
  plot(dg, xlab = "k", ylab = "P(k)")
```

```
}

#Initiate the plugin skeleton
plugin <- PluginBuilder(title="MyPlugin", allMethods=list(mt1=mt1))

#Invoke RedeR and submit the new plugin

rdp <- RedPort('MyPort')

## Not run:
  calld(rdp)
  submitPlugin(rdp, plugin)
  updatePlugins(rdp)

## End(Not run)
```

`RedPort-class` *Class "RedPort"*

Description

A class providing access to the RedeR application.

Slots

`title`: The name of the XML-RPC port.

`uri`: The uri to the XML-RPC server.

`port`: The port number to the XML-RPC server.

`jclass`: The RedeR Java class that should wrap up R graphics.

Methods

Get node attributes from a RedeR session:

getNodes
getNodeIDs
getNodeAliases
getNodeX
getNodeY
getNodeW
getNodeH
getNodeBend
getNodeSize
getNodeShape
getNodeColor
getNodeLineWidth
getNodeLineColor
getNodeFontName

getNodeFontStyle
getNodeFontSize
getNodeFontColor
getNodeFontX
getNodeFontY
getNodeWeight

Set node attributes from a RedeR session:

setNodeXY
setNodeBend
setNodeSize
setNodeShape
setNodeColor
setNodeLineWidth
setNodeLineColor
setNodeFontName
setNodeFontStyle
setNodeFontSize
setNodeFontColor
setNodeFontXY
setNodeWeight

Get edge attributes from a RedeR session:

getEdges
getSourceEdges
getTargetEdges
getEdgeIDs
getSourceEdgeIDs
getTargetEdgeIDs
getArrowDirection
getEdgeWidth
getEdgeColor
getEdgeType
getEdgeWeight

Set edge attributes from a RedeR session:

setArrowDirection
setEdgeWidth
setEdgeColor
setEdgeType
setEdgeWeight

Methods that change graph structure:

addGraph
getGraph
addNodes
deleteNodes

nestNodes
updateContainerSize
mergeOutEdges
getContainerComponets
mergeNodes
addEdges
addEdgeBetweenContainers
deleteEdges

Methods to wrap up attributes and add/get graphs to/from RedeR:

addGraph
getGraph
addSubgraph
addSeries
duplicateGraph

Other methods to manipulate RedeR graphs:

updateGraph
selectEdges
selectNodes
selectAllEdges
selectAllNodes
selectGraph
deSelectEdges
deSelectNodes
deSelectGraph
deleteSelectedEdges
deleteSelectedNodes
isDynamicsActive

Methods to establish RedeR server connection:

ping
version
calld
exitd
resetd

Methods to build RedeR plugins:

PluginBuilder
submitPlugin
deletePlugin
updatePlugins
pluginParser
dynwin

Details

RedPort methods invoke RedeR application via XML-RPC (remote procedure call) server. For each R method listed above there is a Java mirror that executes a callback procedure. Therefore, the Java callback engine must be initialized before any callback from RedeR (i.e. start the Java application).

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
# Creates a RedeR object by calling the constructor
rdp <- RedPort('MyPort')
```

RedPort *The constructor for the RedPort class.*

Description

Constructor to build RedeR interface via XML-RPC (remote procedure call) server.

Usage

```
RedPort(title = 'default', host = '127.0.0.1', port = 9091, jclass = 'reder/rj/C
```

Arguments

title	A character string representing the XML-RPC port.
host	The domain name of the machine that is running the RedeR XML-RPC server.
port	An integer specifying the port on which the XML-RPC server should listen.
jclass	A character string specifying the RedeR Java class that should wrap up R graphics.

Value

An object of the RedPort Class.

Author(s)

Mauro Castro

See Also

calld

Examples

```
rdp <- RedPort('MyPort')
```

RedeR-package *RedeR: bridging the gap between network analysis and visualization.*

Description

RedeR is an R-based package combined with a stand-alone Java application for dynamic network visualization and manipulation. It implements a callback engine by using a low-level R-to-Java interface to build and run common plugins. In this sense, RedeR takes advantage of R to run robust statistics, while the R-to-Java interface bridges the gap between network analysis and visualization: for R Developers, it allows the development of Java plug-ins exclusively using R codes; for Java Users, it runs R methods implemented in a stand-alone application, and for R Users RedeR interactively displays R graphs using a robust Java graphic engine embedded in this package.

Details

Package:	RedeR
Type:	Package
Version:	1.0.1
Date:	2011-05-01
License:	GPL
LazyLoad:	yes

Author(s)

Mauro Castro <mauro.a.castro@gmail.com>

References

Castro, M. A. A. et al. *RedeR: bridging the gap between network analysis and visualization*. Journal Paper (in preparation), 2011.

See Also

RedPort-class

RedeR.data *Pre-processed dataset for RedeR case study.*

Description

Preprocessed data from a time-course gene expression and ChIP-on-chip analysis of estrogen receptor (ER) binding sites in MCF7 breast cancer cell line (Carroll et al, 2006).

Usage

```
data(Carroll2006)
```

Format

`Carroll2006` List containing 'exp', 'tgs', 'ids', and 'bdsites' R objects.

Details

The gene expression dataset consists of 12 time-course Affymetrix U133Plus2.0 microarrays: 3 replicates at 0h, 3 replicates at 3h, 3 replicates at 6h and 3 replicates at 12h. The original dataset is available at GEO database (GSE11324). The gene ER binding site dataset consists of a Bed file of ER ChIP-on-chip experiment. The original dataset is available at http://research.dfci.harvard.edu/brownlab/datasets/index (ER sites from the Bed file '1E-5.bed').

Carroll2006$exp data.frame with log2 gene expression dataset.

Carroll2006$tgs data.frame with microarray details (e.g. targets for limma analysis).

Carroll2006$ids data.frame with gene ids used in RedeR case study.

Carroll2006$bdsites data.frame with ER binding sites mapped to genome build GRCh37.

hs.inter Human interactome extracted from the Human Protein Reference Database (HPRD) in April 2011 <igraph object> ('name' attribute is mapped to ENTREZ ID).

ER.limma data-frame containing pre-processed results from limma analysis and ER binding sites mapped to differentially expressed (DE) genes. Content: annotation (ENTREZ and Symbol), time-course fold change (logFC.t3, logFC.t6, logFC.t12), p values (p.value.t3, p.value.t6, p.value.t12), DE genes (degenes.t3, degenes.t6, degenes.t12) and distance of the closest ER bd site to the TSS – in kb (ERbdist).

ER.deg$dat Summary from ER.limma data object with extracted data for differentially expressed genes only.

ER.deg$exp Data matrix with log2 gene expression values of DE genes.

ER.deg$ceg Co-expression gene network of early ER-responsive genes computed by the function cea `cea`.

References

Carroll JS et al., Genome-wide analysis of estrogen receptor binding sites. Nat Genet. 38(11):1289-97, 2006.

Examples

```
data(Carroll2006)
data(hs.inter)
data(ER.limma)
data(ER.deg)
```

addEdgeBetweenContainers
Add edges between containers.

Description

Method to add edges between RedeR containers. This method adds non-nested assignments, in contrast to the default behavior that builds nested associations to-and-from containers.

Usage

```
addEdgeBetweenContainers(obj, containerA, containerB )
```

Arguments

obj	Object of RedPort Class.
containerA	<string>
containerB	<string>

Value

Add graph objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
el<-matrix(c('n1','n2','n3','n4'), ncol=2, byrow=TRUE)
g <- graph.edgelist(el)

## Not run:

  calld(rdp)
  addGraph( rdp, g, layout.kamada.kawai(g) )
  nestNodes( rdp, c('n1','n2') )
  nestNodes( rdp, c("n3","n4") )
  addEdgeBetweenContainers(rdp, "N0", "N1")
  updateGraph(rdp)

## End(Not run)
```

addEdges *Add edges to RedeR graphs.*

Description

Add edges to an active RedeR session.

Usage

```
addEdges(obj, edges)
```

Arguments

obj	Object of RedPort Class.
edges	Edge sequence as an array <array of strings>.

Value

Adds the specified edges to the graph.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)

## End(Not run)
```

`addGraph` *Add graphs to RedeR application.*

Description

Method to wrap R graphs into RedeR objects and send it to RedeR app.

Usage

```
addGraph(obj, g, ...)
```

Arguments

`obj`	Object of RedPort Class.
`g`	An igraph object.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

layout Vertex coordinates (graph layout). Accepts matrix with 2 cols (i.e. x and y coords) <matrix>.

gscale Expansion factor of the graph area related to the app panel area (default = 75) <numeric>.

zoom Sets the zoom scale for the app panel (range: 0.0 to 100.0; default = 100.0) <numerics>.

gcoord Sets the graph x,y center. Coords between 0 and 100 are set to the visible area of the app panel (default = c(50,50)) <numeric vector>.

isNest Logical value, whether to nest all nodes into a new container (default = FALSE). See additional args in `nestNodes`

isAnchor If isNest=TRUE, this logical value sets whether to anchor the container in dynamic layouts (default = TRUE).

isAssign If isNest=TRUE, this logical value sets whether to assign the container name to the nested nodes (default = FALSE).

loadEdges Logical value, whether to send edges to RedeR app (default = TRUE).

theme Some pre-defined nest attributes. Options: 'tm0','tm1','tm2','tm3','tm4','tm5','tm6' <string>. Alternatively, it can be a list with customized attributes.

ntransform Logical value, whether to transform nodes in containers (default = FALSE).

parent ID of a container already available in the app <string>. Nodes from g will be nested to this container.

Value

Submits R graphs to RedeR app.

Attributes passed by the igraph object

Graph attributes:

bgColor Sets the background color of the app panel <hexadecimal>.

zoom Sets the zoom scale for the app panel (range: 0.0 to 100.0) (Default=100) <numerics>.

gscale Expansion factor of the graph area related to the app panel (range: 0.0 to 100.0) (Default=100) <numerics> (PS. alternative entry to the 'gscale' argument above).

coordX Sets the graph x center; x between 0 and 100 sets to visible area <numeric> (PS. alternative entry to the 'gcoord' argument above).

coordY Sets the graph y center; y between 0 and 100 sets to visible area <numeric> (PS. alternative entry to the 'gcoord' argument above).

loadEdges Logical value, whether to send edges to RedeR app (Default=TRUE) (PS. alternative entry to the 'loadEdges' argument above).

isNest Logical value, whether to nest all nodes into a new container (Default=FALSE) (PS. alternative entry to the 'nest' argument above).

isAnchor If isNest=TRUE, this logical value sets whether to anchor the container in dynamic layouts (Default=FALSE).

isAssign If isNest=TRUE, this logical value sets whether to assign the container name to the nested nodes (Default=FALSE).

nestColor If isNest=TRUE, this attribute sets the 'color' of the new container <hexadecimal>.

nestAlias If isNest=TRUE, this attribute sets the label of the new container <string>.

nestFontSize If isNest=TRUE, this attribute sets the size of the container label (Default=12). <numerics>.

nestFontColor If isNest=TRUE, this attribute sets the 'color' of the container label <hexadecimal>.

nestFontX If isNest=TRUE, this attribute sets the x position of the label related to the container (Default=-8) <numerics>.

nestFontY If isNest=TRUE, this attribute sets the y position of the label related to the container (Default=-8) <numerics>.

nestShape If isNest=TRUE, this attribute sets the shape of the container, options: <'ELLIPSE'> and <'ROUNDED_RECTANGLE'> (Default= ELLIPSE).

nestSize If isNest=TRUE, this attribute sets the size of the container (Default=NULL) <numerics>.

nestLineWidth If isNest=TRUE, this attribute sets the line width of the container, options: >= 0 (Default=1.0) <numerics>.

nestLineColor If isNest=TRUE, this attribute sets the line color of the container <hexadecimal>.

nestImage If isNest=TRUE, sets the status of the container on the screen: <'plain'>, <'transparent'>, or <'hide'> (Default= plain).

nestLineType If isNest=TRUE, this attribute sets the line type of the container: <'SOLID'>, <'DOTTED'>, <'DOTTED_SHORT'>, <'LONG_DASH'> (Default='SOLID').

Vertex attributes:

name Node attribute 'name' <string>.

nodeAlias Node attribute 'alias' <string>.

nodeBend Node attribute 'bend', options: 0-100% (Default=50) <numeric>.

coordX Node attribute 'x coord' (Default=random coord) <numeric>.

coordY Node attribute 'y coord'(Default=random coord) <numeric>.

nodeSize Node attribute 'size', options: > 0 (Default=20) <numeric>.

nodeShape Node attribute 'shape', options: 'ELLIPSE', 'RECTANGLE', 'ROUNDED_RECTANGLE', 'TRIANGLE', 'DIAMOND' (Default= ELLIPSE) <string>.

nodeColor Node attribute 'color', e.g. "#ff0000" for red <hexadecimal>.

nodeWeight Node attribute 'weight', options: >= 0 (Default=0) <numeric>.

nodeLineWidth Node attribute 'line width', options: >= 0 (Default=1) <numeric>.

nodeLineColor Node attribute 'line color', e.g. "#ff0000" for red <hexadecimal>.

nodeFontSize Node attribute 'font size', options: >= 0 (Default=12) <integer>.

nodeFontColor Node attribute 'font color', e.g. "#ff0000" for red <hexadecimal>.

Edge attributes:

arrowDirection Edge attribute 'arrow direction', Options: 0 (A-B), 1 (A->B), 2 (A<-B) or 3 (A<->B) (Default=0) <integer>.

edgeWeight Edge attribute 'weight', options: >= 0 (Default=0.0) <numeric>.

edgeWidth Edge attribute 'width', options: >=0 (Default=1.0) <numeric>.

edgeColor Edge attribute 'color', e.g. "#ff0000" for red <hexadecimal>.

edgeType Edge attribute 'color', options: 'SOLID', 'DOTTED, 'DOTTED_SHORT', 'LONG_DASH' (Default='SOLID').

arrowLength Edge arrow attribute 'length', options: > 0 (Default=10) <numeric>.

arrowAngle Edge arrow attribute 'angle', options: 0-90 (Default=45) <numeric>.

linkType Set assignment type either between nodes and containers or containers and containers. Options: 'nested' and 'notnested' (Default='nested') <string>.

Note

In 'igraph' package, vertex and edge attributes can be assigned as arbitrary R objects. In order to pass these extensible features to RedeR the attributes must be provided in a valid syntax (see above). Only UNIQUE edges are accepted. If present, mutual/multiple edges will be collapsed to unique edges. In this cases, source-target information is transferred to 'arrowDirection' attribute; other attributes will be related to the first edge from the edge list.

Author(s)

Mauro Castro

See Also

`getGraph` `addLegend` `nesthc` `nestNodes` `mergeOutEdges` `relax` `selectNodes` `att`

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)

###

g1 <- graph.empty(n=10, directed=FALSE)
addGraph( rdp, g1, layout.random(g1) )
```

```
resetd(rdp)

###

g2 <- graph.lattice(c(5,5,5))
addGraph( rdp, g2, layout.kamada.kawai(g2) )

resetd(rdp)

###

g <- barabasi.game(10)
V(g)$name<-letters[1:10]
V(g)$nodeSize<-c(100,rep(30,9))
addGraph( rdp,  g, ntransform=TRUE )

sg <- barabasi.game(3)
addGraph( rdp, sg, parent="a" )

resetd(rdp)

###...to check loading of an interactome!

data(hs.inter)
system.time( addGraph(rdp, hs.inter, layout=NULL) )



## End(Not run)
```

`addLegend` *Add graph legends to RedeR application.*

Description

Methods to send legends to RedeR app.

Usage

```
addLegend.color(obj, colvec,  ...)
addLegend.size(obj, sizevec, ...)
addLegend.shape(obj, shapevec, ...)
```

Arguments

`obj`	Object of RedPort Class.
`colvec`	Vector with legend colors, either hexadecimal or valid R color names.
`sizevec`	Vector with legend node size, options: > 0 <numeric>.
`shapevec`	Vector with valid shape names: 'ELLIPSE', 'RECTANGLE', 'ROUNDED_RECTANGLE', 'TRIANGLE', 'DIAMOND'.
`...`	Additional arguments passed to RedeR application.

Details

Alternatively, colvec, sizevec and shapevec can be igraph objects with legend information previously set by the functions att.setv and att.sete.

Additional arguments:

type Legend type. Options: "node" or "edge" (default: "node") <character>.

labvec Vector with legend labels <character>.

position Position of the legend in RedeR panel. Options: 'topleft','topright','bottomleft','bottomright' (default: addLegend.color "topright", addLegend.size "bottomleft", and addLegend.shape "bottomright")<character>.

dxborder Distance (in pixel) from panel border (default: 5) <numeric>.

dyborder Distance (in pixel) from panel border (default: 5) <numeric>.

vertical Logical value, set vertical/horizontal position of the legend in the app panel (default: TRUE for addLegend.color and addLegend.size and FALSE for addLegend.shape.

ftsize Font size (in pixel) (default: 8) <numeric>.

title Legend title <string>.

dxtitle Distance (in pixel) from legend title to the main axis (default: 35) <numeric>.

size Legend size; only for addLegend.color and addLegend.shape methods (default: 30) <numeric>.

bend Legend width/height ratio; only for addLegend.color method (default: 0.85) <numeric>.

col Legend color; only for addLegend.size and addLegend.shape methods (default: "#000000") <either hexadecimal or valid color name>.

intersp Legend inter space (only for addLegend.size and addLegend.shape methods) (default: 0) <numeric>.

edgelen Length of the edges in addLegend.size method (default: 50) <numeric>.

Value

Send legend objects to RedeR app.

Author(s)

Mauro Castro

See Also

addGraph att.setv att.sete

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)

cols<-colorRampPalette(colors=c('red','blue'))(14)
addLegend.color(rdp,cols)
addLegend.color(rdp,cols,type="edge")
```

```
size<-c(10,20,30,40,50)
addLegend.size(rdp,size)

size<-c(1:10)
addLegend.size(rdp,size,type="edge")

shape<-c('ELLIPSE', 'RECTANGLE', 'ROUNDED_RECTANGLE', 'TRIANGLE', 'DIAMOND')
addLegend.shape(rdp,shape)

shape<-c('SOLID','DOTTED','DOTTED_SHORT','LONG_DASH')
addLegend.shape(rdp,shape,type="edge")



## End(Not run)
```

`addNodes` *Add nodes to RedeR graphs.*

Description

Method to add nodes to an active RedeR session.

Usage

```
addNodes(obj, nodes)
```

Arguments

`obj`	Object of RedPort Class.
`nodes`	Node sequence as an array <array of strings>

Value

Add graph objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`

Examples

```
rdp <- RedPort('MyPort')
nodes<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addNodes(rdp, nodes)
  updateGraph(rdp)

## End(Not run)
```

`addSeries` *Add series to RedeR application.*

Description

Method to send series of graphs to RedeR app.

Usage

```
addSeries(obj, g, ...)
```

Arguments

`obj`	Object of RedPort Class.
`g`	An igraph object.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

setnodes Logical value, whether to update node attributes in the new item of the series (default = TRUE).

setedges Logical value, whether to add edges and update attributes in the new item of the series (default = TRUE).

Value

Submits series of R graphs to RedeR app.

Author(s)

Mauro Castro

See Also

`addGraph`

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)

###

g1 <- graph.lattice(c(3,3,3))
addGraph( rdp, g1, layout.kamada.kawai(g1) )
V(g1)$nodeColor<-heat.colors(vcount(g1))
addSeries( rdp, g1)

## End(Not run)
```

`addSubgraph` *Add subgraphs to RedeR application.*

Description

Method to send subgraph to RedeR app.

Usage

```
addSubgraph(obj, g, nodes, ...)
```

Arguments

`obj`	Object of RedPort Class.
`g`	An igraph object.
`nodes`	Nodes of the subgraph <array of strings>
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

gatt A list of graph attributes. See attribute syntax in `addGraph`

gscale Expansion factor of the graph area related to the app panel (default = 75) <numerics>.

gcoord Sets the graph x,y center. Coords between 0 and 100 are set to the visible area of the app panel (default = c(75,75)) <numeric vector>.

theme Some pre-defined nest attributes. Options: 'tm0','tm1','tm2','tm3','tm4','tm5'

Value

Extracts subgraphs from 'igraph' objects and sends the result to the RedeR app.

Author(s)

Mauro Castro

See Also

addGraph addSubgraph.list

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)

g <- graph.lattice(c(5,5,5))

#..extracts a subgraph from g and sends to RedeR:
addSubgraph( rdp, g, nodes=c(1:10) )

#..sets some attributes on g prior to extraction!
g$isNest<-TRUE
g$nestColor="#ff0000"
g$scale=50
addSubgraph( rdp, g, nodes=c(1:10) )

#..alternatively, sets an independent list of attributes:
att <-list()
att$isNest<-TRUE
att$nestColor="#0000ff"
att$scale=50
att$coordX=25
att$coordY=25
addSubgraph( rdp, g, nodes=c(20:30), gatt=att )

#..for further attributes see 'addGraph' function!

## End(Not run)
```

---

addSubgraph.list *Add a list of subgraphs to RedeR application.*

Description

Method to send subgraphs to RedeR app.

Usage

```
addSubgraph.list(obj, g, nodeList, ...)
```

Arguments

obj	Object of RedPort Class.
g	An igraph object.
nodeList	List of nodes. Will be used to extra subgraphs from g.
...	Additional arguments passed to RedeR application.

Details

Additional arguments:

gridRows Number of lines to layout the subgraph panel (default = 2) <integer>

gridScale Expansion factor of the grid area in the app panel. Options: 0.0 to 100 (default = 50) <numeric>.

gscale Expansion factor each subgraph related to the app panel (default = 20) <numeric>.

gatt Either a list or data frame with graph attributes (for data frames, attribute names on cols). See attribute syntax in addGraph

update String argument: if 'all' it forces to update node/edge attributes of a graph already available in the app panel; if 'partial', only node attributes are updated (default = NULL).

theme Some pre-defined nest attributes. Options: 'tm0','tm1','tm2','tm3','tm4','tm5','tm6'.

Value

Extracts subgraphs from 'igraph' objects and sends the result to the RedeR app.

Author(s)

Mauro Castro

See Also

addSubgraph addGraph

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)

g <- graph.lattice(c(5,5,5))

#..extract subgraphs from g and send to RedeR:
nl<-list(c(1:10),c(15:20))
att<-data.frame(isNest=c(TRUE,TRUE), nestColor=c("#0000ff","#ff0000"))
addSubgraph.list( rdp, g, nodeList=nl, gridRows=1, gatt=att, gridScale=80)

#..for further attributes see 'addGraph' function!

## End(Not run)
```

`att` *Map and set edge and vertex attributes to RedeR application.*

Description

These functions map data frames containing edge/vertex attributes to an igraph object and set attributes to RedeR.

Usage

```
att.setv(g, from, to='nodeColor', pal=1, cols=NULL, na.col=grey(0.7), xlim=c(20,
att.sete(g, from, to='edgeColor', pal=1, cols=NULL, na.col=grey(0.7), xlim=c(20,
att.mapv(g, dat, refcol=1)
att.mape(g, dat, refcol=c(1,2))
```

Arguments

`g`	An igraph object.
`from`	An attribute name available in 'g' <string>.
`to`	A valid RedeR attribute name (see `addGraph` or type 'att.setv()' or 'att.sete()').
`breaks`	A numeric vector of two or more breakpoints to be applied to the attribute values.
`pal`	Default color palette. Options: 1 or 2.
`xlim`	A numeric vector with three boundaries: c(<lower boundary>, <upper boundary>, <NA>). It corresponds to boundary values to be apply to numeric attributes (e.g. nodeSize). Default: c(20,100,1).
`cols`	Vector of colors (either hexadecimals or valid R color names).
`na.col`	A color representing eventual NAs. Default: grey(0.7)
`shapes`	A string vector with valid RedeR shapes (see `addGraph` or type 'att.setv()' or 'att.sete()').
`categvec`	Optional: levels to encode attributes as a factor <vector>.
`nquant`	Optional: number of breakpoints to split attribute values by quantiles <integer>.
`isrev`	Optional: reversed version of attribute values <logical>.
`getleg`	Optional: return legend values <logical>.
`dat`	A data frame with the attributes to be mapped to 'g'.
`refcol`	The reference columns in the 'data' object with either node ids (one column <integer>) or edge ids (two columns <vector of two integers>).
`roundleg`	Integer indicating the number of decimal places (round) in the legend of numerical attributes.

Value

Map/set RedeR attributes to igraph objects.

Author(s)

Mauro Castro

See Also

addGraph

Examples

```
data(ER.deg)

sg <- ER.deg$ceg  # an igraph object
dt <- ER.deg$dat  # a data frame object

# maps the data frame to the igraph object
sg <- att.mapv(g=sg, dat=dt, refcol=1)

# Sets graph attributes to RedeR!

# sets gene symbol do nodeAlias attribute
sg <- att.setv(sg, from="Symbol", to="nodeAlias")

# sets numerical value to nodeColor attribute
sg <- att.setv(sg, from="logFC.t3...t0", to="nodeColor", breaks=seq(-1,1,0.2), pal=2)

# sets numerical value to nodeSize attribute
sg <- att.setv(sg, from="ERbdist", to="nodeSize", nquant=10, isrev=TRUE, xlim=c(5,40,1))
```

---

calld *Call RedeR app from R.*

Description

Method to invoke RedeR application from R.

Usage

```
calld(obj, ...)
```

Arguments

obj	Object of RedPort Class.
...	Additional arguments passed to RedeR application.

Details

Other arguments can be passed to the system in order to open the application or set additional environment variables. This maybe required when using the callback functions available in RedeR main panel (i.e. 'RCall'). If this is the case, try to pass the location of R home and the path to the dynamic library directory.

filepath Path to 'reder.jar' file <string>

ADDPATH Add any additional path to RedeR shell <string>

checks Paths' security checks. Option: 'lock' (default) or 'unlock' <string>

Value

Systems call to open RedeR application and XML-RPC server.

Author(s)

Mauro Castro

See Also

RedPort addGraph

Examples

```
rdp <- RedPort('MyPort')

## Not run:

    calld(rdp)

## End(Not run)
```

---

cea *Co-expression analysis.*

Description

Simple function for correlation analysis. This function computes a null distribution via permutation and returns the significant correlation values.

Usage

```
cea(x, sig=0.01, p.adj.method="fdr", cor.method="spearman", nper=100, plotcea=TR
```

Arguments

x	A matrix or data frame.
sig	Significance threshold.
p.adj.method	Correction method passed to "p.adjust" function.
cor.method	Correlation method passed to "cor" function.
nper	Number of permutations.
plotcea	Logical value, option to plot density and the null distributions.
...	Additional arguments passed to plotcea option.

Details

Additional arguments:

ptype If plotcea=TRUE, ptype provides 5 pre-defined plotting options: 1, 2, 3, 4, 5 (Default=1) <integer>.

bk If plotcea=TRUE, bk removes non-significant values from the density distribution (0 <= bk <=1) <numerics>.

n.breaks If plotcea=TRUE, n.breaks sets the number of histogram breaks <integer>.

plotnull Logical value, whether to plot the null distribution (Default=TRUE).

avnull If plotnull=TRUE, avnull takes the average null distribution (Default=TRUE).

nullcol If plotnull=TRUE, nullcol sets the color of the null distribution (Default="black").

Value

Matrix with significant correlation values.

Author(s)

Mauro Castro

See Also

cor p.adjust

Examples

```
data(ER.deg)
#--- a gene expression matrix
exp <- ER.deg$exp
#--- a sample from gx!!
idx <- sample(1:nrow(exp))[1:100]
exp <- exp[idx,]

res <- cea(x=exp, nper=100, plot=FALSE, ptype=4) #ps set 'nper' for at least 1000
```

---

deSelectEdges *Unmark selected edges.*

Description

Unmark all selected edges in an active RedeR session.

Usage

```
deSelectEdges(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Unmark edges.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectEdges(rdp,"n1","n3")
  deSelectEdges(rdp)
  updateGraph(rdp)

## End(Not run)
```

---

deSelectGraph *Unmark selected graph objects.*

Description

Unmark all selected objects in an active RedeR session.

Usage

```
deSelectGraph(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Unmark graph.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, selectNodes, selectEdges, selectGraph

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectGraph(rdp)
  deSelectGraph(rdp)
  updateGraph(rdp)

## End(Not run)
```

---

deSelectNodes *Unmark selected nodes.*

Description

Unmark all selected nodes in an active RedeR session.

Usage

```
deSelectNodes(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Unmark nodes.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectNodes(rdp,c("n3","n4","n5"))
  deSelectNodes(rdp)
  updateGraph(rdp)

## End(Not run)
```

---

deleteEdges *Remove edges from RedeR graphs.*

Description

Method to remove edges between nodes in an active RedeR session.

Usage

```
deleteEdges(obj, edges)
```

Arguments

obj	Object of RedPort Class.
edges	Edge sequence as an array <array of strings>

Value

Removes the specified edges from the graph.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  deleteEdges(rdp, c("n1","n3","n1","n7") )
  updateGraph(rdp)

## End(Not run)
```

---

deleteNodes *Remove nodes from RedeR graphs.*

Description

Method to remove nodes from an active RedeR session.

Usage

```
deleteNodes(obj, nodes)
```

Arguments

obj	Object of RedPort Class.
nodes	Node sequence as an array <array of strings>

Value

Remove graph objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  deleteNodes(rdp, c("n1","n3") )
  updateGraph(rdp)

## End(Not run)
```

`deletePlugin` *Remove plugins from RedeR application.*

Description

Method to remove plugins from the RedeR application, including the main menu.

Usage

```
deletePlugin(obj, pluginName)
```

Arguments

`obj`	Object of RedPort Class.
`pluginName`	Plugin name that is available at RedeR application <string>

Details

Need description!

Value

Delete a plugin from RedeR application.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`PluginBuilder` `submitPlugin` `updatePlugins` `pluginParser`

Examples

```
#Prior calling this method build "MyPlugin" using the PluginBuilder!

rdp <- RedPort('MyPort')

## Not run:

  calld(rdp)
  deletePlugin(rdp, "MyPlugin")
  updatePlugins(rdp)
  #Remove the plugin "MyPlugin" from RedeR

## End(Not run)
```

`deleteSelectedEdges` *Delete selected edges in RedeR graphs.*

Description

Remove all edges selected in an active RedeR session.

Usage

```
deleteSelectedEdges(obj)
```

Arguments

`obj`	Object of RedPort Class.

Value

Remove graph objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `selectAllEdges`, `selectEdges`, `deSelectEdges`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectEdges(rdp,"n1","n3")
  deleteSelectedEdges(rdp)
  updateGraph(rdp)

## End(Not run)
```

`deleteSelectedNodes`

Delete selected nodes in RedeR graphs.

Description

Remove all selected nodes from an active RedeR session.

Usage

```
deleteSelectedNodes(obj)
```

Arguments

`obj`	Object of RedPort Class.

Value

Remove graph objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `selectAllNodes`, `selectNodes`, `deSelectNodes`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectNodes(rdp,c("n3","n4"))
  deleteSelectedNodes(rdp)
  updateGraph(rdp)

## End(Not run)
```

`duplicateGraph` *Duplicate graphs in RedeR application.*

Description

Method to duplicate graphs and subgraphs of a network.

Usage

```
duplicateGraph(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

isToCopyEdges Logical value, whether to include edges to the copy (default = TRUE).

isDefaultCopy Logical value, whether to duplicate the complete network or to copy only the original graph (default = TRUE).

nodes Optional: nodes to be duplicated <array of strings> (p.s. in this case, isDefaultCopy=TRUE).

Value

Duplicates graphs in RedeR app.

Author(s)

Mauro Castro

See Also

`addGraph`

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)

###

g1 <- graph.lattice(c(3,3,3))
addGraph( rdp, g1, layout.kamada.kawai(g1) )
duplicateGraph(rdp)

## End(Not run)
```

---

dynwin *Wrap R graphics.*

Description

Method to wrap R graphics in RedeR Java classes.

Usage

```
dynwin(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Other arguments:

width Initial width of the window <integer>

height Initial height of the window <integer>

ps Initial point size <integer>

Value

Returns R graphics as canvas.

Note

This method should be called only from RedeR plugins!

Author(s)

Mauro Castro

See Also

PluginBuilder RedPort

Examples

```
#To be used just in Rcalls from RedeR plugins!

## Not run:

dynwin( rdp, width=400, height=300, ps=10 )

## End(Not run)
```

---

exitd *Exit RedeR R-to-Java interface.*

Description

Exit R interface and close the active RedeR session.

Usage

```
exitd(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Exit software.

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')

## Not run:

  calld(rdp)
  exitd(rdp)


## End(Not run)
```

`getArrowDirection` *Get arrow direction.*

Description

Get edge attributes 'arrow direction' from an active RedeR session.

Usage

```
getArrowDirection(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns edge attributes <array of integers>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setArrowDirection` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getArrowDirection(rdp)

## End(Not run)
```

`getContainerComponets` *Get container components.*

Description

Method to get components (nested objects) of a specific container from an active RedeR session.

Usage

```
getContainerComponets(obj, container)
```

Arguments

`obj`	Object of RedPort Class.
`container`	Name of the container in the graph <string>

Value

Returns all nested objects assigned to a container <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`

Examples

```
library(igraph)

rdp <- RedPort('MyPort')
el<-matrix(c('n1','n2','n3','n4'), ncol=2, byrow=TRUE)
g <- graph.edgelist(el)

## Not run:

  calld(rdp)
  addGraph( rdp, g, layout.kamada.kawai(g) )
  nestNodes( rdp, c('n1','n2') )
  nestNodes( rdp, c("n3","n4") )
  updateGraph(rdp)
  getContainerComponets(rdp, "N0")

## End(Not run)
```

`getEdgeColor` *Get edge color.*

Description

Method to get edge attributes 'color' from an active RedeR session.

Usage

```
getEdgeColor(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns edge attributes <array of hexadecimal color codes>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setEdgeColor` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getEdgeColor(rdp)

## End(Not run)
```

`getEdgeIDs` *Get edge IDs.*

Description

Method to get ids of all edges from an active RedeR application.

Usage

```
getEdgeIDs(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns edges<array of integers>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getEdgeIDs(rdp)

## End(Not run)
```

`getEdgeType` *Get edge type.*

Description

Method to get edge attributes 'type' from an active RedeR session.

Usage

```
getEdgeType(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns edge attributes <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setEdgeType` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getEdgeType(rdp)

## End(Not run)
```

`getEdgeWeight` *Get edge weight.*

Description

Method to get edge attributes 'weight' from an active RedeR session.

Usage

```
getEdgeWeight(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns edge attributes <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setEdgeWeight` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getEdgeWeight(rdp)

## End(Not run)
```

`getEdgeWidth` *Get edge width.*

Description

Method to get edge attributes 'width' from an active RedeR session

Usage

```
getEdgeWidth(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns edge attributes <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setEdgeWidth` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getEdgeWidth(rdp)

## End(Not run)
```

`getEdges` *Get edges.*

Description

Method to get all edges from an active RedeR application.

Usage

```
getEdges(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='selected'

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

Value

Returns edges <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getEdges(rdp)

## End(Not run)
```

`getGraph` *Get RedeR graph.*

Description

Method to get and wrap up RedeR graphs into R objects.

Usage

```
getGraph(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

status Filter options for RedeR graph status. Valid arguments: <'selected'>, <'nonselected'> or <'all'> (default='all').

type Filter options for RedeR graph objects. Valid arguments: <'node'>, <'container'> or <'all'> (default='node').

attribs Filter options for RedeR graph attributes. Valid arguments: <'plain'>, <'minimal'> or <'all'> (default='plain').

Value

Returns igraph objects.

Author(s)

Mauro Castro

See Also

`addGraph` `RedPort`

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)
#ps. first add a graph (e.g. see samples in RedeR or 'addGraph' method)!
g <- getGraph(rdp)

## End(Not run)
```

`getNodeAliases` *Get node aliases.*

Description

Method to get node attributes 'aliases' from an active RedeR session.

Usage

```
getNodeAliases(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeAliases(rdp)

## End(Not run)
```

`getNodeBend` *Get node bend.*

Description

Method to get node attributes 'bend' from an active RedeR session.

Usage

```
getNodeBend(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeBend` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeBend(rdp)

## End(Not run)
```

`getNodeColor` *Get node color.*

Description

Method to get node attributes 'node color' from an active RedeR session.

Usage

```
getNodeColor(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of hexadecimal color codes>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeColor` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeColor(rdp)

## End(Not run)
```

`getNodeFontColor` *Get font color.*

Description

Method to get node attributes 'font color' from an active RedeR session.

Usage

```
getNodeFontColor(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of hexadecimal color codes>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeFontColor` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeFontColor(rdp)

## End(Not run)
```

`getNodeFontName` *Get font name.*

Description

Method to get node attributes 'font name' from an active RedeR session.

Usage

```
getNodeFontName(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeFontName` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeFontName(rdp)

## End(Not run)
```

`getNodeFontSize` *Get font size.*

Description

Method to get node attributes 'font size' from an active RedeR session.

Usage

```
getNodeFontSize(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeFontSize` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeFontSize(rdp)

## End(Not run)
```

`getNodeFontStyle` *Get font style.*

Description

Method to get node attributes 'font style' from an active RedeR session.

Usage

```
getNodeFontStyle(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of integers>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeFontStyle` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeFontStyle(rdp)

## End(Not run)
```

`getNodeFontX` *Get font x coordinate.*

Description

Method to get node attributes 'font X position' from an active RedeR session.

Usage

```
getNodeFontX(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `getNodeFontY`, `setNodeFontXY` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeFontX(rdp)

## End(Not run)
```

`getNodeFontY` *Get font y coordinate.*

Description

Method to get node attributes 'font Y position' from an active RedeR session.

Usage

```
getNodeFontY(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`getGraph RedPort`, `getNodeFontX`, `setNodeFontXY`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeFontY(rdp)

## End(Not run)
```

getNodeH *Get node height.*

Description

Method to get node attributes 'node height' from an active RedeR session.

Usage

```
getNodeH(obj, ...)
```

Arguments

obj	Object of RedPort Class.
...	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort getGraph

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeH(rdp)

## End(Not run)
```

`getNodeIDs` *Get node IDs.*

Description

Method to get node attributes 'node IDs' from an active RedeR session.

Usage

```
getNodeIDs(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
library(RedeR)
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeIDs(rdp)

## End(Not run)
```

`getNodeLineColor` *Get node-line color.*

Description

Method to get node attributes 'line color' from an active RedeR session.

Usage

```
getNodeLineColor(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of hexadecimal color codes>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeLineColor` `getGraph`

Examples

```
library(RedeR)
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeLineColor(rdp)

## End(Not run)
```

`getNodeLineWidth` *Get node-line width.*

Description

Method to get node attributes 'line width' from an active RedeR session.

Usage

```
getNodeLineWidth(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeLineWidth` `getGraph`

Examples

```
library(RedeR)
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeLineWidth(rdp)

## End(Not run)
```

`getNodeShape` *Get node shape.*

Description

Method to get node attributes 'node shape' from an active RedeR session.

Usage

```
getNodeShape(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of strings>.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeShape` `setNodeBend` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeShape(rdp)

## End(Not run)
```

`getNodeSize` *Get node size.*

Description

Method to get node attributes 'node size' from an active RedeR session.

Usage

```
getNodeSize(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of integers>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeSize` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeSize(rdp)

## End(Not run)
```

getNodeW *Get node with.*

Description

Method to get node attributes 'node with' from an active RedeR session.

Usage

```
getNodeW(obj, ...)
```

Arguments

obj	Object of RedPort Class.
...	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort getGraph

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeW(rdp)

## End(Not run)
```

`getNodeWeight` *Get node weight.*

Description

Method to get node attributes 'weight' from an active RedeR session.

Usage

```
getNodeWeight(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `setNodeWeight` `getGraph`

Examples

```
library(RedeR)
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeWeight(rdp)

## End(Not run)
```

getNodeX *Get node x coordinate.*

Description

Method to get node attributes 'node X position' from an active RedeR session.

Usage

```
getNodeX(obj, ...)
```

Arguments

obj	Object of RedPort Class.
...	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

getGraph RedPort, getNodeY, setNodeXY

Examples

```
library(RedeR)
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeX(rdp)

## End(Not run)
```

getNodeY *Get node y coordinate.*

Description

Method to get node attributes 'node Y position' from an active RedeR session.

Usage

```
getNodeY(obj, ...)
```

Arguments

obj	Object of RedPort Class.
...	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns node attributes <array of numerics>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

getGraph RedPort, getNodeX, setNodeXY

Examples

```
library(RedeR)
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodeY(rdp)

## End(Not run)
```

`getNodes` *Get nodes.*

Description

Method to get node list from an active RedeR session.

Usage

```
getNodes(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='selected'

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

Value

Returns nodes <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getNodes(rdp)

## End(Not run)
```

`getSourceEdgeIDs` *Get source-edge IDs.*

Description

Method to get IDs of all 'source' edges from an active RedeR session.

Usage

```
getSourceEdgeIDs(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns 'source' edges <array of integers>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getSourceEdgeIDs(rdp)

## End(Not run)
```

`getSourceEdges` *Get source edges.*

Description

Method to get all 'source' edges from an active RedeR session.

Usage

```
getSourceEdges(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns 'source' edges <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getSourceEdges(rdp)

## End(Not run)
```

`getTargetEdgeIDs` *Get target-edge IDs.*

Description

Method to get IDs of all 'target' edges from an active RedeR session.

Usage

```
getTargetEdgeIDs(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns 'target' edges <array of integers>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getTargetEdgeIDs(rdp)

## End(Not run)
```

`getTargetEdges` *getTargetEdges*

Description

Get IDs of all 'target' edges from an active RedeR session.

Usage

```
getTargetEdges(obj, ...)
```

Arguments

`obj`	Object of RedPort Class.
`...`	Additional arguments passed to RedeR application.

Details

Additional arguments:

type Filter options. Valid arguments: <'node'>, <'container'> or <'all'>. Default='node'.

status Filter options. Valid arguments: <'selected'>, <'nonselected'> or <'all'>. Default='all'

Value

Returns 'target' edges <array of strings>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort` `getGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  getTargetEdges(rdp)

## End(Not run)
```

gtoy.rm — *Random graphs and modules.*

Description

A very simple function to generate random graphs with modular structures.

Usage

```
gtoy.rm(m=3, nmax=30, nmin=3, p1=0.5, p2=0.05, p3=0.9)
```

Arguments

m	Number of modules.
nmax	The maximum number of vertices in each module.
nmin	The minimum number of vertices in each module.
p1	Probability for adding new vertices to a module.
p2	Probability for drawing an edge between modules.
p3	Probability for drawing an edge within modules.

Value

Returns a igraph object.

Author(s)

Mauro Castro

Examples

```
g<-gtoy.rm()
```

isDynamicsActive — *Inquires about RedeR current state.*

Description

Inquires whether 'dynamics' algorithm is active in RedeR application.

Usage

```
isDynamicsActive(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Returns 1<integer> if true, 0<integer> otherwise.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')

## Not run:

  calld(rdp)
  isDynamicsActive (rdp)
    # 1 or 0

## End(Not run)
```

---

mergeNodes *Merge nodes.*

Description

Merge nodes in an active RedeR session and build a new group.

Usage

```
mergeNodes(obj, nodes)
```

Arguments

obj	Object of RedPort Class.
nodes	Node sequence <array of strings>

Value

Add/change graph objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
nodes<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addNodes(rdp, nodes)
  mergeNodes(rdp,c("n2","n3","n4"))
  updateGraph(rdp)

## End(Not run)
```

---

mergeOutEdges *Merge out-edges between connected containers and transfers edges from nodes to containers.*

Description

Method to assign out-edges to containers in an active RedeR session. This method transfers edges from nodes to the respective containers.

Usage

```
mergeOutEdges(obj,...)
```

Arguments

obj	Object of RedPort Class.
...	Additional arguments passed to RedeR application.

Details

Additional arguments:

rescale Logical value. Whether to rescale the out-edge width to fit container size limits; if false, it will run a simple sum (default=TRUE).

lb Custom lower bound to rescale edge width (default=NULL) <numerics>.

ub Custom upper bound to rescale edge width between containers (default=NULL) <numerics>.

nlev Number of levels to be merged in the hierarchy (default=1) <integer>.

Value

Add/change edge assigments.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
el<-matrix(c("n1","n2","n1","n3","n1","n4","n2","n5","n2","n6","n2","n7"), ncol=2, byrow=
g <- graph.edgelist(el)

## Not run:

  calld(rdp)
  addGraph( rdp, g, layout.kamada.kawai(g) )
  nestNodes( rdp, c("n1","n2") )
  mergeOutEdges(rdp)
  updateGraph(rdp)

## End(Not run)
```

---

nestNodes *Nest nodes to containers.*

Description

Method to nest nodes in an active RedeR session.

Usage

```
nestNodes(obj, nodes, ...)
```

Arguments

obj	Object of RedPort Class.
nodes	<array of strings>
...	Additional arguments passed to RedeR application.

Details

Additional arguments:

nestImage Status of the container on the screen: <'plain'>, <'transparent'>, or <'hide'> (default = 'plain').

isAssign Logical value, whether to assign the container name to the nested nodes (default = TRUE).

isAnchor Logical value, whether is to anchor the container in dynamic layouts (default = FALSE).

gscale Expansion factor of the nest area related to a parent nest – or related to the app panel (default = 40) <numerics>.

gcoord Sets the nest c(x,y) center related to the parent center. Coords between 0 and 100 are set to the inner area (default = NULL) <numeric vector>.

parent Nest ID of a parent nest. Must be used with 'isAssign=TRUE' (default = NULL).

gatt A list with graph attributes. See nest attribute syntax in addGraph

theme Some pre-defined nest attributes. Options: 'tm0','tm1','tm2','tm3','tm4','tm5', 'tm6' <string>. Alternatively, it can be a list with customized attributes.

Value

Add/change graph objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
el<-matrix(c('n1','n2','n3','n4'), ncol=2, byrow=TRUE)
g <- graph.edgelist(el)

## Not run:

  calld(rdp)
  addGraph( rdp, g, layout.kamada.kawai(g) )
  nestNodes( rdp, c('n1','n2') )
  nestNodes( rdp, c("n3","n4") )

## End(Not run)
```

`nesthc` *Nest hclust objects to containers.*

Description

Method to nest nodes in an active RedeR session.

Usage

```
nesthc(obj, hc, ...)
```

Arguments

`obj`	Object of RedPort Class.
`hc`	Either an object of hclust of pvclust class.
`...`	Additional arguments passed to RedeR application; if a "pvclust" object, it also passes arguments for "pvpick" function (e.g. to set the p-value threshold).

Details

Additional arguments:

cutlevel Numeric value indicating the point where the hclust object should be cut (default = 2). The distance is related to the option 'metric'. For "rootdist" and "leafdist", the cut level is related to the n steps required to get to the root's level or to the leaf's level, respectively (n>=1). For 'height', the cut is related to the corresponding dendrogram height <numeric>.

metric Metric used to cut the hclust object at the top level (Options: "rootdist", "leafdist" or "height"; default="rootdist") <string>.

nmemb Minimum number of members for a nest (>=2) <numeric>.

nlev Maximum number of levels of a nested sequence (default=2) <numeric>.

grid Number of rows and cols to lay out graphs in the panel (default = c(2,3)) <numeric>.

gridScale Expansion factor of the grid area in the app panel. Options: 0.0 to 100 (default = 75) <numeric>.

gscale Expansion factor to set the nest area related to the parents – or related to the app panel. Provided as a vector with three numbers, c(n1,n2,n3): n1 is related to nests at the first level of the hierarchy (i.e. nests rooted to the panel); n2 is related to nests from single branches, and n3 nests from double branches (default = c(30,75,45)) <numeric>.

isAnchor Logical value; it sets whether to anchor containers in dynamic layouts.

isAssign Logical value; it sets whether to assign container names to nested nodes.

theme Some pre-defined nest attributes. Options: 'tm0','tm1','tm2','tm3','tm4','tm5', 'tm6' (default: 'tm6') <string>. Alternatively, it can be a list with customized attributes.

nlinewidth Line width of a nested series containers.

nfontsz Label font size a nested series containers.

plothc Logical value; whether to plot the corresponding hclust object (i.e. dendrogram).

col A color vector; it is used to color labels in both containers and corresponding hclust object (i.e. dendrogram nodes).

cex Numeric character expansion factor of dendrogram text and labels.

xlab A label for the dendrogram x axis.

ylab A label for the dendrogram y axis.

Value

Add/change graph objects and plot corresponding hclust object.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort nestNodes gtoy.rm

Examples

```
g <- gtoy.rm()
hc<-hclust(dist(get.adjacency(g)))
#plot(hc)

rdp <- RedPort('MyPort')

## Not run:

  calld(rdp)
  addGraph(rdp,g)
  nesthc(rdp, hc)


## End(Not run)
```

---

ping *Test RedeR R-to-Java interface.*

Description

Test R interface and the connection to an active RedeR session.

Usage

```
ping(obj)
```

Arguments

obj	Object of RedPort Class.

Value

"R interface is ready to use!"

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')

## Not run:

calld(rdp)

ping (rdp)

    # "R interface is ready to use!"


## End(Not run)
```

---

pluginParser *RedeR plugin parser.*

Description

Function to parse R expressions to RedeR plugins.

Usage

```
pluginParser(dyname, dyncode, args=FALSE)
```

Arguments

dyname	Name of the expression to display in RedeR application.
dyncode	R code (e.g. function) to be parsed.
args	Option to parse formal arguments.

Value

Returns parsed but unevaluated expressions.

Note

This function is used internally by the 'PluginBuilder' to parse plug-in methods and add-ons.

Author(s)

Mauro Castro

See Also

PluginBuilder

Examples

```
x<-function(){...}
pluginParser('MyMethod', x, args=TRUE)
```

---

rederpost *Remote procedure calls optimized for RedeR.*

Description

Internal function.

Author(s)

Mauro Castro

---

relax *relax*

Description

This function starts the dynamic layout and sets the force-directed options available in RedeR app.

Usage

```
relax(obj,p1=100,p2=100,p3=100,p4=100,p5=100,p6=100,p7=10,p8=10,ps=FALSE)
```

Arguments

obj	Object of RedPort Class.
p1	Edge target length (in pixels; >= 1) <numeric>.
p2	Edge stiffness (arbitrary unities; >= 1) <numeric>.
p3	Node repel factor (arbitrary unities; >= 1) <numeric>.
p4	Node perimeter effect (in pixels; >= 1) <numeric>.
p5	Node speed limit (arbitrary unities; >= 1) <numeric>.
p6	Nest-nest edge target length, i.e., edge target between linked containers (in pixels; >= 1) <numeric>.
p7	Nest-node repel factor, i.e., repulsion among containers and out-nodes (arbitrary unities; >= 1) <numeric>.
p8	Repulsion radius, i.e., this parameter limits the repel factor range (given in p1 unites; >= 1) <numeric>.
ps	Panel settings: logical value, whether to start interactive panel.

Details

One of the most versatile features of RedeR is the ability to deal with nested network objects using dynamic simulation, which makes it possible to represent, for example, subnetworks and time-series onto the same graph in a user-friendly routine. The simulation uses force-directed algorithms as described elsewhere (Brandes 2001; Fruchterman and Reingold 1991). Here we adapted the method to deal with nested networks. In force-directed graphs, each edge can be regarded as a spring - with a given target length - and can either exert a repulsive or attractive force on the connected nodes, while nodes are analogous to mutually repulsive charged particles that move according to the applied forces. In RedeR, the simulation is additionally constrained by the hierarchical structure. For example, a nested node is constrained to its parent-node by opposing forces applied by the nest, which is regarded as a special node whose nested objects can reach a local equilibrium independently from other network levels. The simulation is adjusted by global options and evolves iteratively (and interactively) until the system reaches the equilibrium state. The parameters controlling the dynamics are arbitrarily set to layout sparse networks with a few nodes (e.g. 10-100 nodes). For large and dense networks better results can be achieved interactively by tuning one or more parameters.

Author(s)

Mauro Castro

References

Brandes U. Drawing graphs: methods and models. In: Lecture notes in computer science. Kaufmann M. and Wagner D. (Ed), vol. 2025. Heidelberg: Springer; 2001: 71-86.

Fruchterman TMJ, Reingold EM. Graph drawing by force-directed placement. Software: Practice and Experience 1991, 21(11):1129-1164.

Examples

```
rdp <- RedPort('MyPort')

g <- graph.lattice(c(5,5,5))

## Not run:

  calld(rdp)
  addGraph( rdp, g, layout.random(g) )
  relax(rdp)


## End(Not run)
```

`resetd` *Reset RedeR app.*

Description

Reset the active RedeR session.

Usage

```
resetd(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Reset the software panel.

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')

## Not run:

  calld(rdp)
  resetd(rdp)


## End(Not run)
```

---

selectAllEdges *Select all edges.*

Description

Method to mark all edges in an active RedeR application. Selected objects are put available for other methods. It can be done interactively as well.

Usage

```
selectAllEdges(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Mark edges.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, deleteSelectedEdges

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectAllEdges(rdp)
  updateGraph(rdp)

## End(Not run)
```

---

selectAllNodes *selectAllNodes*

Description

Mark all nodes in an active RedeR application.

Usage

```
selectAllNodes(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Mark nodes.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, deleteSelectedNodes

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectAllNodes(rdp)
  updateGraph(rdp)

## End(Not run)
```

---

selectEdges *selectEdges*

Description

Select edges in an active RedeR application.

Usage

```
selectEdges(obj, nodeA, nodeB)
```

Arguments

obj	Object of RedPort Class.
nodeA	<string>
nodeB	<string>

Value

Mark edges – which can be handled by other methods.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, deleteSelectedEdges

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectEdges(rdp,"n1","n3")
  updateGraph(rdp)

## End(Not run)
```

`selectGraph` *Select graph.*

Description

Method to mark all objects in an active RedeR application. Selected objects are put available for other methods. It can be done interactively as well.

Usage

```
selectGraph(obj)
```

Arguments

`obj`	Object of RedPort Class.

Value

Mark graph.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `deleteSelectedNodes`, `deleteSelectedEdges`, `deSelectGraph`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectGraph(rdp)
  updateGraph(rdp)

## End(Not run)
```

---

selectNodes *Select nodes.*

Description

Method to select nodes in an active RedeR application. Selected objects are put available for other methods. It can be done interactively as well.

Usage

```
selectNodes(obj, nodes, nt=NULL)
```

Arguments

obj	Object of RedPort Class.
nodes	Names of nodes (or containers) <string or array of strings>
nt	Optional for nested nodes: to restrict searching to a specific container <string>

Value

Mark nodes – which can be handled by other methods.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, deleteSelectedNodes

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  selectNodes(rdp,c("n3","n4","n5"))
  updateGraph(rdp)

## End(Not run)
```

---

setArrowDirection *Set arrow direction.*

Description

Method to set edge attribute 'arrow direction' in active RedeR sessions.

Usage

```
setArrowDirection(obj, nodeA, nodeB, direction)
```

Arguments

obj	Object of RedPort Class.
nodeA	Name <string>
nodeB	Name <string>
direction	Options: 0 (A-B), 1 (A->B), 2 (A<-B) or 3 (A<->B) <integer>

Value

Sets edge attribute <integer>

Note

The direction is set according to the edge order in the app (i.e. the edge list available inside RedeR). So, if a request for direction "1" places nodeA='B' and nodeB='A', then the direction will appear as A->B in the app.

Author(s)

Mauro Castro

See Also

RedPort, getArrowDirection

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setArrowDirection(rdp, "n1", "n2", 2)
  updateGraph(rdp)

## End(Not run)
```

---

setEdgeColor *Set edge color.*

Description

Method to set edge attribute 'color' in active RedeR sessions.

Usage

```
setEdgeColor(obj, nodeA, nodeB, color)
```

Arguments

obj	Object of RedPort Class.
nodeA	Name <string>
nodeB	Name <string>
color	e.g. "#ff0000" for red <hexadecimal>

Value

Sets edge attribute <hexadecimal>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getEdgeColor

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setEdgeColor(rdp, "n1", "n2", "#ff0000")
  updateGraph(rdp)

## End(Not run)
```

---

setEdgeType *Set edge type.*

Description

Method to set edge attribute 'type' in active RedeR sessions.

Usage

```
setEdgeType(obj, nodeA, nodeB, type)
```

Arguments

obj	Object of RedPort Class.
nodeA	Name <string>
nodeB	Name <string>
type	Options: 'SOLID, 'DOTTED, 'DOTTED_SHORT','LONG_DASH' <string>

Value

Sets edge attribute <string>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getEdgeType

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setEdgeType(rdp, "n1", "n2", "DOTTED")
  updateGraph(rdp)

## End(Not run)
```

`setEdgeWeight` *Set edge weight.*

Description

Method to set edge attribute 'weight' in active RedeR sessions.

Usage

```
setEdgeWeight(obj, nodeA, nodeB, weight)
```

Arguments

`obj`	Object of RedPort Class.
`nodeA`	Node name <string>
`nodeB`	Node name <string>
`weight`	Edge weight <numeric>

Value

Sets edge attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `getEdgeWeight`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  setEdgeWeight(rdp, "n1", "n2", 0.5)
  #Check attribs in RedeR app and run 'relax'!

## End(Not run)
```

---

setEdgeWidth *Set edge width.*

Description

Method to set edge attribute 'width' in active RedeR sessions.

Usage

```
setEdgeWidth(obj, nodeA, nodeB, width)
```

Arguments

obj	Object of RedPort Class.
nodeA	Node name <string>
nodeB	Node name <string>
width	Options: > 0.0 <numeric>

Value

Sets edge attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getEdgeWidth

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setEdgeWidth(rdp, "n1", "n2", 4.0)
  updateGraph(rdp)

## End(Not run)
```

---

setNodeAlias *Set node alias.*

Description

Method to set node attributes 'alias' in active RedeR sessions.

Usage

```
setNodeAlias(obj, node, alias)
```

Arguments

obj	Object of RedPort Class.
node	Node name <string>
alias	Node alias <string>

Value

Sets node attribute <string>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort getNodeAliases

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeAlias(rdp, "n1", "Node1")
  updateGraph(rdp)

## End(Not run)
```

`setNodeBend` *Set node bend.*

Description

Method to set node attribute 'bend' in active RedeR sessions.

Usage

```
setNodeBend(obj, node, bend)
```

Arguments

`obj`	Object of RedPort Class.
`node`	Name <string>
`bend`	Options: 0-100% <numeric>

Details

This is an arbitrary argument that produces different forms using the same basic node shape (e.g. to change the eccentricity of an ellipse)

Value

Sets node attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `getNodeBend`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeBend(rdp, "n1", 2.5)
  updateGraph(rdp)

## End(Not run)
```

---

setNodeColor *Set node color.*

Description

Method to set node attribute 'color' in active RedeR sessions.

Usage

```
setNodeColor(obj, node, color)
```

Arguments

obj	Object of RedPort Class.
node	Name <string>
color	e.g. "#ff0000" for red <hexadecimal>

Value

Sets node attribute <hexadecimal>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeColor

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeColor(rdp, "n1", "#ff0000")
  updateGraph(rdp)

## End(Not run)
```

`setNodeFontColor` *Set font color.*

Description

Method to set node attribute 'font color' in active RedeR sessions.

Usage

```
setNodeFontColor(obj, node, color)
```

Arguments

`obj`	Object of RedPort Class.
`node`	Name <string>
`color`	e.g. "#ff0000" for red <hexadecimal>

Value

Sets node attribute <hexadecimal>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `getNodeFontColor`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeFontColor(rdp, "n1", "#ff0000")
  updateGraph(rdp)

## End(Not run)
```

---

setNodeFontName *Set font name.*

Description

Method to set node attribute 'font name' in active RedeR sessions.

Usage

```
setNodeFontName(obj, node, name)
```

Arguments

obj	Object of RedPort Class.
node	Node name <string>
name	Font name (options are JRE or system dependent) <string>

Value

Sets node attribute <string>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeFontName

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
#Note: font names are system dependent!
  setNodeFontName(rdp, "n1", "Arial")
  updateGraph(rdp)

## End(Not run)
```

---

setNodeFontSize *Set font size.*

Description

Method to set node attribute 'font size' in active RedeR sessions.

Usage

```
setNodeFontSize(obj, node, size)
```

Arguments

obj	Object of RedPort Class.
node	Name <string>
size	Options: >= 0 <integer>

Value

Sets node attribute <integer>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeFontSize

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeFontSize(rdp, "n1", 50)
  updateGraph(rdp)

## End(Not run)
```

---

setNodeFontStyle *Set font style.*

Description

Method to set node attribute 'font style' in active RedeR sessions.

Usage

```
setNodeFontStyle(obj, node, style)
```

Arguments

obj	Object of RedPort Class.
node	Node name <string>
style	Options: 0 (PLAIN), 1 (BOLD), 2 (ITALIC), or 3 (BOLD+ITALIC) <integer>

Value

Sets node attribute <integer>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeFontStyle

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeFontStyle(rdp, "n1", 1)
  updateGraph(rdp)

## End(Not run)
```

`setNodeFontXY` *Set font x,y coordinates.*

Description

Method to set node attribute 'font XY position' in active RedeR sessions.

Usage

```
setNodeFontXY(obj, node, x, y)
```

Arguments

`obj`	Object of RedPort Class.
`node`	Node name<string>
`x`	relative x coord. (related to x node position) <numeric>
`y`	relative y coord. (related to y node position)<numeric>

Value

Sets node attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `getNodeFontX`, `getNodeFontY`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeFontXY(rdp, "n1", 50.0, 50.0)
  updateGraph(rdp)

## End(Not run)
```

---

setNodeLineColor *Set node-line color.*

Description

Method to set node attribute 'line color' in active RedeR sessions.

Usage

```
setNodeLineColor(obj, node, color)
```

Arguments

obj	Object of RedPort Class.
node	Name <string>
color	e.g. "#ff0000" for red <hexadecimal>

Value

Sets node attribute <hexadecimal>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeLineColor

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeLineColor(rdp, "n1", "#3333ff")
  updateGraph(rdp)

## End(Not run)
```

---

setNodeLineWidth *Set node-line width.*

Description

Method to set node attribute 'line width' in active RedeR sessions.

Usage

```
setNodeLineWidth(obj, node, width)
```

Arguments

obj	Object of RedPort Class.
node	Name <string>
width	Options: >= 0 <numeric>

Value

Sets node attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeLineWidth

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeLineWidth(rdp, "n1", 5.5)
  updateGraph(rdp)

## End(Not run)
```

---

setNodeShape *Set node shape.*

Description

Method to set node attribute 'shape' in active RedeR sessions.

Usage

```
setNodeShape(obj, node, shape)
```

Arguments

obj	Object of RedPort Class.
node	Name <string>
shape	Options: 'ELLIPSE', 'RECTANGLE','ROUNDED_RECTANGLE','TRIANGLE','DIAMOND' <string>

Details

See 'setNodeBend' to produce different shapes using the same basic node form (e.g. to change the eccentricity of an ellipse).

Value

Sets node attribute <string>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeShape setNodeBend

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeShape(rdp, "n1", "RECTANGLE")
  updateGraph(rdp)

## End(Not run)
```

`setNodeSize` *Set node size.*

Description

Method to set node attribute 'size' in active RedeR sessions.

Usage

```
setNodeSize(obj, node, size)
```

Arguments

`obj`	Object of RedPort Class.
`node`	Name <string>
`size`	Options: >= 0 <numeric>

Value

Sets node attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`, `getNodeSize`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeSize(rdp, "n1", 40.0)
  updateGraph(rdp)

## End(Not run)
```

---

setNodeWeight *Set node weight.*

Description

Method to set node attribute 'weight' in active RedeR sessions.

Usage

```
setNodeWeight(obj, node, weight)
```

Arguments

obj	Object of RedPort Class.
node	Node name <string>
weight	Node weight <numeric>

Value

Sets node attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeWeight

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)
  setNodeWeight(rdp, "n1", 2.0)
  getNodeWeight(rdp)
  #Available to internal RedeR methods (e.g. transfer att to node color or size)

## End(Not run)
```

---

setNodeXY *Set node x,y coordinates.*

Description

Method to set node attribute 'XY position' in active RedeR sessions.

Usage

```
setNodeXY(obj, node, x, y)
```

Arguments

obj	Object of RedPort Class.
node	Node name<string>
x	x coord. <numeric>
y	y coord. <numeric>

Value

Sets node attribute <numeric>

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

RedPort, getNodeX, getNodeY

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  setNodeXY(rdp, "n1", 200.0, 200.0)
  setNodeXY(rdp, "n2", 100.0, 100.0)
  setNodeXY(rdp, "n3", 300.0, 300.0)
  updateGraph(rdp)

## End(Not run)
```

---

subg *Subgraph of a graph.*

Description

Creates a subgraph containing only nodes specified from a data frame, including all edges among neighbors.

Usage

```
subg(g, dat, refcol=1, maincomp=TRUE, connected=TRUE, transdat=TRUE)
```

Arguments

g	An igraph object.
dat	A data frame with node ids and attributes to be mapped to 'g'.
refcol	The reference column (node ids) in the 'dat' object.
maincomp	Logical value, whether to return only the main component of the subgraph.
connected	Logical value, whether to return only connected nodes.
transdat	Logical value, whether to transfer node attributes from the 'dat' object to the subgraph.

Value

Returns a igraph object.

Author(s)

Mauro Castro

Examples

```
data(hs.inter)
data(ER.deg)
subnet <- subg(g=hs.inter, dat=ER.deg$dat, refcol=1)
```

submitPlugin *Submit plugins to RedeR application.*

Description

Method to send R-code to RedeR app in order to upload a new plugin.

Usage

```
submitPlugin(obj, plugin)
```

Arguments

obj	Object of RedPort Class.
plugin	Object of PluginBuilder Class.

Value

Send a new plugin to RedeR application.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

PluginBuilder updatePlugins deletePlugin pluginParser

Examples

```
#Set a simple example to initiate a plugin skeleton

#Initiates a new method, say mt1
mt1<-function(){"#Your code here!"}

#Initiates a plugin skeleton
plugin <- PluginBuilder ( title = "MyPlugin", allMethods = list(mt1=mt1) )

## Not run:
  #Invoke RedeR application and load the new plugin
  rdp <- RedPort('MyPort')
  calld(rdp)
  submitPlugin(rdp, plugin)
  updatePlugins(rdp)

## End(Not run)
```

`updateContainerSize` *Update container size.*

Description

Updates the size of all containers in an active RedeR session.

Usage

```
updateContainerSize(obj)
```

Arguments

`obj`	Object of RedPort Class.

Value

Updates RedeR's container objects.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  nestNodes( rdp, c("n2","n3") )
  updateContainerSize(rdp)
  updateGraph(rdp)

## End(Not run)
```

updateGraph *Update RedeR graphs.*

Description

Updates an active RedeR application session.

Usage

```
updateGraph(obj)
```

Arguments

obj	Object of RedPort Class.

Value

Updates RedeR graph.

Note

Prior calling this method make sure that there is an active RedeR session.

Author(s)

Mauro Castro

See Also

RedPort

Examples

```
rdp <- RedPort('MyPort')
edges<-c("n1","n2","n1","n3","n1","n4","n1","n5","n1","n6","n1","n7")

## Not run:

  calld(rdp)
  addEdges(rdp, edges)
  updateGraph(rdp)

## End(Not run)
```

`updatePlugins` *Update RedeR plugins.*

Description

Method to call RedeR application in order to update plugins sent by R command.

Usage

```
updatePlugins(obj)
```

Arguments

`obj`	Object of RedPort Class.

Value

Updates new plugins.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`PluginBuilder` `submitPlugin` `deletePlugin` `pluginParser`

Examples

```
rdp <- RedPort('MyPort')

## Not run:
  calld(rdp)
  updatePlugins(rdp)

## End(Not run)
```

`version` *Version*

Description

Check RedeR application version.

Usage

```
version(obj)
```

Arguments

`obj`	Object of RedPort Class.

Value

Returns the version of the current RedeR application that is listening a specified XML-RPC port.

Note

Prior calling this method invoke RedeR application via XML-RPC server (i.e. 'calld').

Author(s)

Mauro Castro

See Also

`RedPort`

Examples

```
rdp <- RedPort('MyPort')

## Not run:

  calld(rdp)
  version(rdp)
    # "RedeR v1.0"

## End(Not run)
```

Index