Package 'NGCHM'

December 17, 2024

Type Package

Title Next Generation Clustered Heat Maps

Version 1.0.4

Description Next-Generation Clustered Heat Maps (NG-CHMs) allow for dynamic exploration of heat map data in a web browser. 'NGCHM' allows users to create both stand-alone HTML files containing a Next-Generation Clustered Heat Map, and .ngchm files to view in the NG-CHM viewer. See Ryan MC, Stucky M, et al (2020) <doi:10.12688/f1000research.20590.2> for more details.

License GPL-3

URL `https://md-anderson-bioinformatics.github.io/NGCHM-R/`,
`https://github.com/MD-Anderson-Bioinformatics/NGCHM-R`

Depends R (>= 3.5.0)

Imports digest, grDevices, htmltools, httr, jsonlite, logger (>= 0.2.0), methods, stats, tsvio, utils

Suggests knitr, rmarkdown, Rtsne, testthat, umap, uwot

Enhances NGCHMDemoData, NGCHMSupportFiles

Additional_repositories `https://md-anderson-bioinformatics.r-universe.dev`

Encoding UTF-8

RoxygenNote 7.2.3

Collate 'internalFunctions.R' 'allClasses.R' 'allGenerics.R' 'allMethods.R' 'functions.R' 'chmFromJSON.R' 'dendrogram-fixes.R' 'shaidy.R' 'ngchmshaidy.R' 'zzz.R' 'package.R'

SystemRequirements Java (>= 11) Git

NeedsCompilation no

Author Bradley M Broom [aut] (<`https://orcid.org/0000-0002-0915-3164`>),
Mary A Rohrdanz [ctb, cre],
Chris Wakefield [ctb],
James M Melott [ctb],
MD Anderson Cancer Center [cph]

Maintainer Mary A Rohrdanz <marohrdanz@mdanderson.org>

Repository CRAN

Date/Publication 2024-12-17 19:40:02 UTC

Contents

+ *Add an Axis to an NG-CHM Version 2*

Description

This function adds an ’ngchmAxis’ to an ’ngchmVersion2’ object.

Usage

```
## S4 method for signature 'ngchmVersion2,ngchmAxis'
e1 + e2
```

Arguments

`e1`	An object of class ’ngchmVersion2’ to which the axis is to be added.
`e2`	An object of class ’ngchmAxis’ representing the axis to be added.

Value

An updated ’ngchmVersion2’ object with the added axis.

`castAsInteger` *Helper function to cast variables as integers.*

Description

If variable value is far from integer, print error message and stop.

Usage

```
castAsInteger(variableToCast)
```

Arguments

`variableToCast`	Variable to cast as integer

Value

integer value of variableToCast

`castListAsInteger` *Helper function to cast list as integer*

Description

If variable value is far from integer, print error message and stop.

Usage

```
castListAsInteger(listToCast)
```

Arguments

`listToCast`	List to cast as integer

Value

list with members cast to integers

chmAdd *Add a list of objects to a NGCHM.*

Description

Each additional parameter is added to the NGCHM according to its type. Objects that require additional information (such as an axis) cannot be added using this function. Objects that can be added are layers (including numeric matrices), datasets, and colormaps.

Usage

```
chmAdd(chm, ...)

## S4 method for signature 'ngchm'
chmAdd(chm, ...)
```

Arguments

`chm`	The chm to add the object(s) to.
`...`	Zero or more objects to add to the NGCHM.

Value

The extended chm.

See Also

"chmAddAxisType"

"chmAddColormap"

"chmAddDataset"

"chmAddLayer"

"chmAddMetaData"

chmAddAxisType *Add an axis type to a NGCHM.*

Description

Adds an axis type to a Next Generation Clustered Heat Map (NGCHM) and returns the extended CHM. Multiple axis types may be added to either axis. When the NGCHM is made, any Axis functions matching the specified axis type will be automatically added to the appropriate axis menu, and any Matrix functions matching the types of the rows and columns will be automatically added to the matrix menu.

Usage

```
chmAddAxisType(chm, where, type, func)

## S4 method for signature 'ngchm,character,character,ngchmJS'
chmAddAxisType(chm, where, type, func)

## S4 method for signature 'ngchm,character,character,character'
chmAddAxisType(chm, where, type, func)

## S4 method for signature 'ngchm,character,character,missing'
chmAddAxisType(chm, where, type, func)
```

Arguments

`chm`	The chm to add the axis type to.
`where`	The axis to add the axis type to. Must be either "row" or "column".
`type`	The type to add to the specified axis.
`func`	A javascript function that gets values of that type from the current selection. If a string is provided, the function is obtained by calling chmGetFunction.

Value

The extended chm.

See Also

`chmListTypes()`

`chmRegisterAxisFunction()`

`chmRegisterMatrixFunction()`

`chmRegisterTypeMapper()`

ngchmAxisType

`chmAddColormap` *Add a colormap to a NGCHM.*

Description

Add a colormap to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM. Duplicate colormaps will be silently dropped.

Usage

```
chmAddColormap(chm, colormap)

## S4 method for signature 'ngchm,ngchmColormap'
chmAddColormap(chm, colormap)
```

Arguments

`chm` The chm to add the colormap to.

`colormap` The colormap to add to the chm.

Details

Note that it is not necessary to explicitly add colormaps included with data layers or classification bars. These will be included automatically. Explicitly using this function is only required in order to add additional predefined, but unused colormaps to the NGCHM.

Value

The extended chm.

See Also

`chmNewColorMap()`

ngchmColormap

`chmAddCovariate` *Add a covariate to an auxiliary dataset.*

Description

Add a covariate to an auxiliary dataset and return the extended dataset. Do not confuse this function with the one for adding a covariate bar to an NGCHM. For that, please refer to the function chmAddCovariateBar.

Usage

```
chmAddCovariate(dataset, where, covariate)

## S4 method for signature 'ngchmDataset,character,ngchmCovariate'
chmAddCovariate(dataset, where, covariate)
```

Arguments

`dataset` The dataset to add the covariate to.

`where` The dataset axis to add the covariate to. Must be one of "row", "column", or "both".

`covariate` The covariate to add to the dataset.

Value

The extended dataset.

See Also

`chmNewCovariate()`

ngchmCovariate

`chmAddCovariateBar` *Add a covariate bar to a NGCHM.*

Description

Add a covariate bar to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM. If passed a covariate, a covariate bar will be created (using any optional parameters supplied) and added.

Usage

```
chmAddCovariateBar(chm, where, covar, ...)

## S4 method for signature 'ngchm,character,ngchmBar'
chmAddCovariateBar(chm, where, covar)

## S4 method for signature 'ngchm,character,ngchmCovariate'
chmAddCovariateBar(chm, where, covar, ...)

## S4 method for signature 'ngchm,character,list'
chmAddCovariateBar(chm, where, covar, ...)
```

Arguments

`chm`	The chm to add the covariate bar to.
`where`	The chm axis(axes) to add the covariate bar to. Must be one of "row", "column", or "both".
`covar`	The covariate or covariate bar (or a list of them) to add to the chm.
`...`	Additional parameters passed to chmNewCovariateBar if covar is a covariate.

Details

If a covariate bar with the same name already exists on the specified axis or axes, the existing bar will be replaced by the new bar.

Value

The extended chm.

See Also

`chmNewCovariate()`

`chmNewCovariateBar()`

ngchmCovariate

chmAddCSS *Add custom CSS to a NGCHM.*

Description

Add custom Cascading Style Sheet (CSS) to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM.

Usage

```
chmAddCSS(chm, css)

## S4 method for signature 'ngchm,character'
chmAddCSS(chm, css)
```

Arguments

`chm`	The chm to add the CSS to.
`css`	The css selector and style information.

Value

The extended chm.

See Also

ngchmCSS

chmAddDataset *Add an auxiliary dataset to a NGCHM.*

Description

Add an auxiliary dataset to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM. The auxiliary dataset will be stored with the NGCHM and be available in whole or in part from the same server, for use, for example, in custom Javascript functions. Do not confuse this function with the one for adding an active data layer to the heatmap itself. For that, please refer to the function chmAddLayer.

Usage

```
chmAddDataset(chm, dataset)

## S4 method for signature 'ngchm,ngchmDataset'
chmAddDataset(chm, dataset)
```

Arguments

`chm`	The chm to add the dataset to.
`dataset`	The dataset to add to the chm.

Value

The extended chm.

See Also

`chmNewDataset()`

ngchmDataset

`chmAddDialog` *Add an extra dialog to a NGCHM.*

Description

Add an extra dialog to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM.

Usage

```
chmAddDialog(chm, dialog)

## S4 method for signature 'ngchm,ngchmDialog'
chmAddDialog(chm, dialog)
```

Arguments

`chm`	The chm to add the dialog to.
`dialog`	The dialog to add to the chm.

Value

The extended chm.

See Also

`chmNewDialog()`

ngchmDialog

chmAddLayer *Add a Layer to a NGCHM.*

Description

Add a Layer to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM. A CHM requires at least one Layer. The first layer added to a NGCHM becomes the primary layer. The second layer added to a NGCHM, if any, becomes the secondary (flicker) layer. Currently at most two layers can be added to a NGCHM.

Usage

```
chmAddLayer(chm, layer)

## S4 method for signature 'ngchm,ngchmLayer'
chmAddLayer(chm, layer)

## S4 method for signature 'ngchm,matrix'
chmAddLayer(chm, layer)
```

Arguments

`chm`	The chm to add the layer to.
`layer`	The layer to add to the chm.

Value

The extended chm.

See Also

`chmNewDataLayer()`

ngchmLayer

chmAddMenuItem *Add a menu entry to a NGCHM.*

Description

Add a popup menu entry to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM.

Usage

```
chmAddMenuItem(chm, where, label, func)

## S4 method for signature 'ngchm,character,character,ngchmJS'
chmAddMenuItem(chm, where, label, func)

## S4 method for signature 'ngchm,character,character,character'
chmAddMenuItem(chm, where, label, func)
```

Arguments

`chm`	The chm to add the menu entry to.
`where`	The chm menu(s) to add the menu entry to. Must be one of "row", "column", "both", or "element".
`label`	The label to display in the menu entry.
`func`	The javascript function to invoke when the menu entry is selected.

Value

The extended chm.

See Also

ngchmMenuItem

`chmAddMetaData` *Add MetaData to NG-CHM*

Description

This function adds metadata to a NG-CHM (Next-Generation Clustered Heat Map) object.

Usage

```
chmAddMetaData(chm, where, type, value)

## S4 method for signature 'ngchm,character,character,character'
chmAddMetaData(chm, where, type, value)
```

Arguments

`chm`	An object of class 'ngchm'.
`where`	A single character string specifying where to add the metadata. Can be "row", "column", or "both".
`type`	A single character string specifying the type of the metadata.
`value`	A character vector specifying the values of the metadata. If value is a character vector, elements of the vector will be attached as meta data to to NGCHM row of the same name.

Value

An updated 'ngchm' object with the new metadata added.

---

chmAddOverview *Generate an overview image of the NGCHM when making it.*

Description

Generate an overview image of the NGCHM when making it. By default, the system generates no default overview images. If only one of width or height is specified, the other is calculated based on the aspect ratio of the map.

Usage

```
chmAddOverview(chm, format, width, height)

## S4 method for signature 'ngchm,character,optNumeric,optNumeric'
chmAddOverview(chm, format, width, height)
```

Arguments

`chm`	The chm to add the overview to.
`format`	The format of the overview ('pdf', 'png', or 'svg').
`width`	The width of the overview.
`height`	The height of the overview.

Value

The extended chm.

---

chmAddPCA *Add PCA coordinates to an NG-CHM.*

Description

Add PCA coordinates as hidden covariate bars to an axis of an NG-CHM. One hidden covariate bar is added for each PCA coordinate (up to ndim coordinates). Coordinates are given names 'BASENAME.coordinate.N', where BASENAME is specified by the parameter basename (default "PC") and N ranges from 1 to the number of added covariate bars.

Usage

```
chmAddPCA(hm, axis, prc, basename = "PC", ndim = 2)
```

Arguments

`hm`	The NGCHM to add the coordinates to.
`axis`	The NGCHM axis ("row" or "column") to add the coordinates to.
`prc`	Principal component coordinates (output of `stats::prcomp()`) for the specified NGCHM axis.
`basename`	The prefix to use for the coordinate names.
`ndim`	The maximum number of coordinates to add.

Value

The NGCHM with added coordinates.

See Also

`chmAddTSNE()`

`chmAddUMAP()`

`chmAddUWOT()`

`chmAddReducedDim()`

Examples

```
# Examples using `chmNew()` require git to be installed.
## Not run:
  # If the NGCHMDemoData package is installed, use it to demo usage
  if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
   data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
    prc <- prcomp(TCGA.GBM.EXPR[1:50, 1:50])
    hm <- chmNew("gbm", TCGA.GBM.EXPR[1:50, 1:50])
    hm <- chmAddPCA(hm, "column", prc)
  }
  # Small example not requiring NGCHMDemoData
  matrix <- matrix(rnorm(100),
    nrow = 10, ncol = 10,
    dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
  )
  prc <- prcomp(matrix)
  hm <- chmNew("Demo PCA", matrix)
  hm <- chmAddPCA(hm, "column", prc)

## End(Not run)
```

chmAddProperty *Add custom property to a NGCHM.*

Description

Add custom property to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM.

Usage

```
chmAddProperty(chm, label, value)

## S4 method for signature 'ngchm,character,character'
chmAddProperty(chm, label, value)
```

Arguments

chm	The chm to add the property to.
label	The property label.
value	The property value.

Value

The extended chm.

See Also

ngchmProperty

chmAddReducedDim *Add reduced dimension coordinates to an NG-CHM.*

Description

Add (reduced) dimension coordinates from an object obj as hidden covariate bars to an axis of an NG-CHM. Depending on the object type, dimName and dimAxis can be used to specify the name of the dimension of interest in obj.

Usage

```
chmAddReducedDim(hm, axis, obj, dimName, maxDim, basename, dimAxis)
```

Arguments

hm	The NGCHM to add the coordinates to.
axis	The NGCHM axis ("row" or "column") to add the coordinates to.
obj	An object containing the (reduced) dimension.
dimName	The name of the (reduced) dimension to create covariate bars for.
maxDim	The maximum number of coordinates to add (default all).
basename	The prefix to use for the coordinate names (defaults to dimName).
dimAxis	The axis on obj containing the named dimension, if applicable.

Details

One hidden covariate bar is added for each coordinate obtained from obj. If specified, maxDim limits the maximum number of covariate bars added to the chm.

Coordinates have names 'BASENAME.coordinate.N', where BASENAME is specified by the parameter basename (defaults to dimName if omitted) and N ranges from 1 to the number of added covariate bars.

obj can be a numeric matrix, each column of which is a (reduced) dimension. In this case, dimName and dimAxis are not used for obtaining the reduced dimension. The number of rows of the matrix must equal the size of the specified NGCHM axis and each row of the matrix must be uniquely named using the names from that axis of the NG-CHM.

obj can also be an instance of class className if there exists an S3 method getDimensions.className. The method takes the object as its first parameter and up to two optional parameters, dimName and dimAxis, that can be used to specify the desired dimension. The method's return value is a matrix similar to the one described in the preceding paragraph. This package defines methods for classes prcomp and umap.

Value

The NGCHM with added coordinates.

See Also

chmAddPCA()

chmAddTSNE()

chmAddUMAP()

chmAddUWOT()

getDimensions()

Examples

```
# Examples using `chmNew()` require git to be installed.
## Not run:
  if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
    data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
    mat <- TCGA.GBM.EXPR[1:10, 1:10]
    prc <- prcomp(mat)
    hm <- chmNew("Demo reduced dimension coordinates", mat)
    hm <- chmAddReducedDim(hm, "column", prc, "PCA", 3, "PC")
    umc <- umap::umap(t(mat), n_neighbors = 8)
    hm <- chmAddReducedDim(hm, "column", umc, "UMAP")
  }
  # Small example not requiring NGCHMDemoData
  matrix <- matrix(rnorm(100),
    nrow = 10, ncol = 10,
    dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
  )
  prc <- prcomp(matrix)
  hm <- chmNew("Demo reduced dimension coordinates", matrix)
  hm <- chmAddReducedDim(hm, "column", prc, "PCA", 3, "PC")
  umc <- umap::umap(t(matrix), n_neighbors = 8)
  hm <- chmAddReducedDim(hm, "column", umc, "UMAP")

## End(Not run)
```

chmAddRelated *Add a link to related information to the NGCHM.*

Description

Add a link to related information to the NGCHM.

Usage

```
chmAddRelated(chm, group, link, description)

## S4 method for signature 'ngchm,character,character,character'
chmAddRelated(chm, group, link, description)
```

Arguments

chm	The chm to add the related link to.
group	The name of the group this link belongs to.
link	The link to include. Should be either an absolute URL, or a NGCHM name on the same server.
description	A string describing the referenced link and its relationship to the current NGCHM.

Value

The extended chm.

chmAddRelatedGroup *Add a group of related links to the NGCHM.*

Description

Add a group of related links to the NGCHM.

Usage

```
chmAddRelatedGroup(chm, name, header, linktype, blurb)

## S4 method for signature 'ngchm,character,character,character,character'
chmAddRelatedGroup(chm, name, header, linktype, blurb)

## S4 method for signature 'ngchm,character,character,character,missing'
chmAddRelatedGroup(chm, name, header, linktype)
```

Arguments

`chm`	The chm to add the related link group to.
`name`	The name of the group of links.
`header`	The header that should be displayed for this group of links.
`linktype`	Type of link belonging to this group.
`blurb`	An optional descriptive paragraph to include between the group header and the group links.

Value

The extended chm.

chmAddSpecificAxisTypeFunction *Add a CHM-specific axis type function to a NGCHM.*

Description

Adds a CHM-specific axis type function to a Next Generation Clustered Heat Map (NGCHM) and returns the extended CHM. Multiple axis type functions may be added to either axis. When the NGCHM is made, any specific Axis functions matching the specified axis type will be automatically added to the appropriate axis menu.

Usage

```
chmAddSpecificAxisTypeFunction(chm, where, type, label, func)

## S4 method for signature 'ngchm,character,character,character,ngchmJS'
chmAddSpecificAxisTypeFunction(chm, where, type, label, func)

## S4 method for signature 'ngchm,character,character,character,character'
chmAddSpecificAxisTypeFunction(chm, where, type, label, func)
```

Arguments

`chm`	The chm to add the axis type to.
`where`	The axis to add the axis type to. Must be either "row", "column", or "both".
`type`	The type expected by the specified function.
`label`	The label to use if and when the function is added to the menu.
`func`	A javascript function that accepts a list of values of that type. If a string is provided, the function is obtained by calling chmGetFunction.

Value

The extended chm.

See Also

`chmListTypes()`

ngchmAxisType

`chmAddTag` *Add tags to a NGCHM.*

Description

Add one or more tags to a Next Generation Clustered Heat Map (NGCHM) and return the extended CHM.

Usage

```
chmAddTag(chm, tag)

## S4 method for signature 'ngchm,character'
chmAddTag(chm, tag)
```

Arguments

`chm`	The chm to add the dataset to.
`tag`	The tag(s) to add to the chm.

Value

The extended chm.

---

chmAddTemplate *Add a file template to the NGCHM.*

Description

Add a file template to the NGCHM.

Usage

```
chmAddTemplate(chm, source.path, dest.path, substitutions)

## S4 method for signature 'ngchm,charOrFunction,character,optList'
chmAddTemplate(chm, source.path, dest.path, substitutions)
```

Arguments

chm	The chm to add the file template to.
source.path	A string giving the path to the template, or a function that returns the template content as a string.
dest.path	A string giving the relative path where to store the template in the generated CHM.
substitutions	A list (may be empty) of substitutions to make in the template.

Value

The extended chm.

---

chmAddToolboxR *Add standard toolbox to an NG-CHM axis*

Description

This function adds a toolbox to a NG-CHM (Next-Generation Clustered Heat Map) axis.

Usage

```
chmAddToolboxR(CHM, axis, axistype, datasetname, idstr)

## S4 method for signature 'ngchm,character,character,character,character'
chmAddToolboxR(CHM, axis, axistype, datasetname, idstr)
```

Arguments

`CHM`	An object of class 'ngchm'.
`axis`	A single character string specifying the axis where the toolbox will be added. Can be "row", "column", or "both".
`axistype`	A single character string specifying the type of the axis.
`datasetname`	A single character string specifying the name of the dataset.
`idstr`	string to append to toolbox menu labels (default ")

Value

An updated 'ngchm' object with the new toolbox added.

`chmAddToolboxR2` *Add Toolbox R2 to NG-CHM*

Description

This function adds a toolbox of type R2 to a NG-CHM (Next-Generation Clustered Heat Map) object.

Usage

```
chmAddToolboxR2(CHM, axistype, datasetname, idstr)

## S4 method for signature 'ngchm,character,character,character'
chmAddToolboxR2(CHM, axistype, datasetname, idstr)
```

Arguments

`CHM`	An object of class 'ngchm'.
`axistype`	A single character string specifying the type of the axis.
`datasetname`	A single character string specifying the name of the dataset.
`idstr`	string to append to toolbox menu labels (default ")

Value

An updated 'ngchm' object with the new toolbox of type R2 added.

chmAddToolboxRC *Add Toolbox RC to NG-CHM*

Description

This function adds a toolbox of type RC to a NG-CHM (Next-Generation Clustered Heat Map) object.

Usage

```
chmAddToolboxRC(CHM, rowtype, coltype, datasetname, idstr)

## S4 method for signature 'ngchm,character,character,character,character'
chmAddToolboxRC(CHM, rowtype, coltype, datasetname, idstr)
```

Arguments

CHM	An object of class ’ngchm’.
rowtype	A single character string specifying the type of the row.
coltype	A single character string specifying the type of the column.
datasetname	A single character string specifying the name of the dataset.
idstr	string to append to toolbox menu labels (default ”)

Value

An updated ’ngchm’ object with the new toolbox of type RC added.

chmAddTSNE *Add TSNE coordinates to an NG-CHM.*

Description

Add TSNE coordinates as hidden covariate bars to an axis of an NG-CHM. One hidden covariate bar is added for each TSNE coordinate. Coordinates have names ’BASENAME.coordinate.N’, where BASENAME is specified by the parameter basename (default TSNE) and N ranges from 1 to the number of added covariate bars.

Usage

```
chmAddTSNE(hm, axis, tsne, pointIds, basename = "TSNE")
```

Arguments

hm	The NGCHM to add the coordinates to
axis	The NGCHM axis ("row" or "column") to add the coordinates to
tsne	TSNE coordinates (output of Rtsne::Rtsne()) for the specified NGCHM axis
pointIds	The NGCHM names for the data points in tsne
basename	The prefix to use for the coordinate names.

Details

pointIds is required because Rtsne::Rtsne() does not preserve the rownames of the data matrix it was applied to. Their values must match those on that axis of the NGCHM, but their order must match those in the data matrix passed to Rtsne::Rtsne().

Value

The NGCHM with added coordinates.

See Also

chmAddPCA()

chmAddUMAP()

chmAddUWOT()

chmAddReducedDim()

Examples

```
# Examples using  `chmNew()` require git to be installed.
## Not run:
  # If the NGCHMDemoData package is installed, use it to demo usage
  if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
    data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
    mat <- TCGA.GBM.EXPR[1:10, 1:10]
    rtc <- Rtsne::Rtsne(t(mat), check_duplicates = FALSE, perplexity = 3)
    hm <- chmNew("gbm", mat)
    hm <- chmAddTSNE(hm, "column", rtc, colnames(mat))
  }
  # Small example not requiring NGCHMDemoData
  matrix <- matrix(rnorm(100),
    nrow = 10, ncol = 10,
    dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
  )
  rtc <- Rtsne::Rtsne(t(matrix), check_duplicates = FALSE, perplexity = 3)
  hm <- chmNew("Demo TSNE", matrix)
  hm <- chmAddTSNE(hm, "column", rtc, colnames(matrix))

## End(Not run)
```

chmAddUMAP *Add UMAP coordinates to an NG-CHM.*

Description

Add UMAP coordinates as hidden covariate bars to an axis of an NG-CHM. One hidden covariate bar is added for each UMAP coordinate. Coordinates have names 'BASENAME.coordinate.N', where BASENAME is specified by the parameter basename (default UMAP) and N ranges from 1 to the number of added covariate bars.

Usage

```
chmAddUMAP(hm, axis, umap, basename = "UMAP")
```

Arguments

hm	The NGCHM to add the coordinates to.
axis	The NGCHM axis ("row" or "column") to add the coordinates to.
umap	UMAP coordinates (output of umap::umap()) for the specified NGCHM axis.
basename	The prefix to use for the coordinate names.

Value

The NGCHM with added coordinates.

See Also

chmAddPCA()

chmAddTSNE()

chmAddUWOT()

chmAddReducedDim()

Examples

```
# Examples using `chmNew()` require git to be installed.
## Not run:
  # If the NGCHMDemoData package is installed, use it to demo usage
  if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
    data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
    mat <- TCGA.GBM.EXPR[1:50, 1:50]
    umc <- umap::umap(t(mat))
    hm <- chmNew("gbm", mat)
    hm <- chmAddUMAP(hm, "column", umc)
  }
  # Small example not requiring NGCHMDemoData
  matrix <- matrix(rnorm(100),
    nrow = 10, ncol = 10,
    dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
  )
  umc <- umap::umap(t(matrix), n_neighbors = 8)
  hm <- chmNew("Demo UMAP", matrix)
  hm <- chmAddUMAP(hm, "column", umc)

## End(Not run)
```

chmAddUWOT *Add UWOT::UMAP coordinates to an NG-CHM.*

Description

Add UWOT::UMAP coordinates as hidden covariate bars to an axis of an NG-CHM. One hidden covariate bar is added for each UMAP coordinate. Coordinates have names 'BASENAME.coordinate.N', where BASENAME is specified by the parameter basename (default UMAP) and N ranges from 1 to the number of added covariate bars.

Usage

```
chmAddUWOT(hm, axis, uwot, pointIds, basename = "UMAP")
```

Arguments

hm	The NGCHM to add the coordinates to.
axis	The NGCHM axis ("row" or "column") to add the coordinates to.
uwot	UMAP coordinates (output of uwot::umap()) for the specified NGCHM axis.
pointIds	The NGCHM names for the data points in uwot
basename	The prefix to use for the coordinate names.

Details

pointIds is required because uwot::umap() does not preserve the rownames of the data matrix it was applied to. Their values must match those on that axis of the NGCHM, but their order must match those in the data matrix passed to uwot::umap().

Value

The NGCHM with added coordinates.

See Also

chmAddPCA()

chmAddTSNE()

chmAddUMAP()

chmAddReducedDim()

Examples

```
# Examples using `chmNew()` require git to be installed.
## Not run:
  # If the NGCHMDemoData package is installed, use it to demo usage
  if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
    data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
    umc <- uwot::umap(t(TCGA.GBM.EXPR[1:50, 1:50]))
    hm <- chmNew("gbm", TCGA.GBM.EXPR[1:50, 1:50])
    hm <- chmAddUWOT(hm, "column", umc, colnames(TCGA.GBM.EXPR[1:50, 1:50]))
  }
  # Small example not requiring NGCHMDemoData
  matrix <- matrix(rnorm(100),
    nrow = 10, ncol = 10,
    dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
  )
  umc <- uwot::umap(t(matrix), n_neighbors = 8)
  hm <- chmNew("Demo UMAP", matrix)
  hm <- chmAddUWOT(hm, "column", umc, colnames(matrix))

## End(Not run)
```

---

chmAxis *Create a new Axis for adding to an NG-CHM.*

Description

This function creates a new Axis for adding to a Next Generation Clustered Heat Map. You can specify any axis name here, but chmAdd only accepts row, column, and both.

Usage

```
chmAxis(axis, ...)
```

Arguments

`axis`	The name of the axis
`...`	Objects to add to the axis

Value

An object of class 'ngchmAxis' representing the newly created axis.

See Also

`chmAdd()`

Examples

```
x_axis <- chmAxis('row')
y_axis <- chmAxis('col')
```

---

chmAxisType *Create a new AxisType for adding to an ngchmAxis.*

Description

This function creates a new AxisType for adding to an ngchmAxis.

Usage

```
chmAxisType(typename, func)
```

Arguments

typename	The name of the axis type to be created. This should be a single character string.
func	The function to be used for getting label values. If not provided, the default 'getLabelValue' function is used. If a character string is provided, it is assumed to be the name of a function and is retrieved using 'chmGetFunction'. If a function is provided, it is checked to be of class 'ngchmJS'.

Value

An object of class 'ngchmAxisType' representing the newly created axis type.

See Also

chmAxis()

---

chmBindFunction *Bind values to an existing JS function.*

Description

Create a new JS function by binding values to extra parameters of an existing JS function.

Usage

```
chmBindFunction(name, fn, bindings)

## S4 method for signature 'character,ngchmJS,list'
chmBindFunction(name, fn, bindings)

## S4 method for signature 'character,character,list'
chmBindFunction(name, fn, bindings)
```

Arguments

`name`	A single character string specifying the name of the function.
`fn`	An object of class 'ngchmJS' representing the function to be bound.
`bindings`	A list containing at least one parameter binding. Each list element binds one parameter, starting from the first unbound parameter, and the name of each list element must match the name of the corresponding parameter.

Value

A new 'ngchmJS' object representing the bound function.

See Also

`chmNewFunction()`

`chmBrowse` *Browse the NGCHMs on the specified server in the viewer.*

Description

Opens the NG-CHM browser page in the viewer.

Usage

```
chmBrowse(server = NULL, viewer = NULL)
```

Arguments

`server`	The NG-CHM server to be browsed. If NULL, the function will use the first server in the list of available servers.
`viewer`	The function to be used to open the web browser. If NULL, the function will use the 'browseURL' function.

Value

None. This function is used for its side effects of opening a web browser to view the NG-CHM server.

See Also

`utils::browseURL()`

chmColOrder<- *Set the column order of data shown in a NGCHM.*

Description

This function sets the column order for a NG-CHM (Next-Generation Clustered Heat Map) object.

Usage

```
chmColOrder(chm) <- value

## S4 replacement method for signature 'ngchm,optDendrogram'
chmColOrder(chm) <- value
```

Arguments

chm	An object of class ’ngchm’.
value	An object of class ’optDendrogram’ or ’file’ specifying the new column order. If value is NULL, the labels will be displayed in the same order they are found in the first data layer. If value is a character vector, the labels will be displayed in that order. If value is a dendrogram, the labels displayed in the order they occur in a depth first traversal of the tree.

Value

An updated ’ngchm’ object with the new column order.

See Also

"chmRowOrder<-"

chmColorMap *Get the color map of an NG-CHM object.*

Description

Get the color map of an NG-CHM object.

Usage

```
chmColorMap(x)
```

Arguments

`x`	The NG-CHM object to get the color map of. Can be: • An object of class ngchmLayer • An object of class ngchmBar • An object of class ngchmCovariate

Value

An ngchmColormap

See Also

chmNewColorMap

Examples

```
# If the NGCHMDemoData package is installed, use it to demo usage
if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
  data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
  colormap <- chmColorMap(chmNewDataLayer("New layer", TCGA.GBM.EXPR[1:3, 1:3]))
}
matrix <- matrix(rnorm(100),
  nrow = 10, ncol = 10,
  dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
)
colormap <- chmColorMap(chmNewDataLayer("New layer", matrix))
```

`chmColorMap<-` *Set the color map of an NG-CHM object*

Description

Set the color map of an NG-CHM object

Usage

```
chmColorMap(x) <- value
```

Arguments

`x`	The NG-CHM object on which to set the color map.
`value`	The ngchmColormap value to set.

Value

The modified NG-CHM object.

See Also

chmColorMap

Examples

```
# If the NGCHMDemoData package is installed, use it to demo usage
if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
  data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
  dataLayer <- chmNewDataLayer("GBM layer", TCGA.GBM.EXPR[1:30, 1:30])
  chmColorMap(dataLayer) <- chmNewColorMap(c(2, 14))
}
# Small example not requiring NGCHMDemoData
matrix <- matrix(rnorm(100),
  nrow = 10, ncol = 10,
  dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
)
dataLayer <- chmNewDataLayer("my layer", matrix)
chmColorMap(dataLayer) <- chmNewColorMap(c(2, 14))
```

chmColors — *Get the colors of an ngchmColormap, ngchmLayer, ngchmBar, or ngchmCovariate.*

Description

Get the colors of an ngchmColormap, ngchmLayer, ngchmBar, or ngchmCovariate.

Usage

```
chmColors(x)
```

Arguments

x	The object to get the colors of.

Value

A character string vector of the map colors.

See Also

ngchm

Examples

```
# If the NGCHMDemoData package is installed, use it to demo usage
if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
  data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
  colors <- chmColors(chmNewDataLayer("GBM Expression", TCGA.GBM.EXPR[1:50, 1:50]))
}
# Small example not requiring NGCHMDemoData
matrix <- matrix(rnorm(100),
  nrow = 10, ncol = 10,
  dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
)
colors <- chmColors(chmNewDataLayer("my layer", matrix))
```

---

chmColors<- *Set the colors of an ngchmColormap, ngchmLayer, ngchmBar, or ngchmCovariate.*

Description

Set the colors of an ngchmColormap, ngchmLayer, ngchmBar, or ngchmCovariate.

Usage

```
chmColors(x) <- value
```

Arguments

x	The NG-CHM object on which to set the colors.
value	A character string vector of colors. The vector length must equal the number of data points in the color map.

Value

The modified NG-CHM object.

See Also

chmColors

Examples

```
# If the NGCHMDemoData package is installed, use it to demo usage
if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
  data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
  layer <- chmNewDataLayer("GBM Layer", TCGA.GBM.EXPR[1:50, 1:50])
  chmColors(layer) <- c("blue", "white", "red")
}
```

```
# Small example not requiring NGCHMDemoData
matrix <- matrix(rnorm(100),
  nrow = 10, ncol = 10,
  dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
)
layer <- chmNewDataLayer("my layer", matrix)
chmColors(layer) <- c("blue", "white", "red")
```

chmCovariate *Get a covariate attached to an NG-CHM dataset.*

Description

Get a covariate attached to an NG-CHM dataset.

Usage

```
chmCovariate(dataset, fullname, where)
```

Arguments

dataset	The NG-CHM dataset to get the covariate from.
fullname	The full name of the covariate to get. If no covariate with that name exists, return NULL.
where	The axis or axes on which to look for the covariate Can be "row", "column", or "both" (default).

Value

A ngchmCovariate or NULL.

See Also

ngchmCovariate

chmNewCovariate

chmCovariateBar

Examples

```
# If the NGCHMDemoData package is installed, use it to create demo usage
if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
  data(TCGA.GBM.Demo, package = "NGCHMDemoData")
 dataset <- chmNewDataset("gbmexpr", "TCGA GBM Expression Data", TCGA.GBM.ExpressionData)
  dataset <- chmAddCovariate(
    dataset, "column",
    chmNewCovariate("TP53 Mutation", TCGA.GBM.TP53MutationData)
  )
  tp53_mutation <- chmCovariate(dataset, "TP53 Mutation")
}
# Small example not requiring NGCHMDemoData
matrix <- matrix(rnorm(100),
  nrow = 10, ncol = 10,
  dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
)
dataset <- chmNewDataset("Demo", "Random Demo Dataset", matrix)
covariate <- setNames(rnorm(10), colnames(matrix))
dataset <- chmAddCovariate(dataset, "column", chmNewCovariate("Random Covariate", covariate))
random_covariate <- chmCovariate(dataset, "Random Covariate")
```

chmCovariateBar *Get a covariate bar attached to an NG-CHM.*

Description

Get a covariate bar attached to an NG-CHM.

Usage

```
chmCovariateBar(hm, fullname, where)
```

Arguments

hm	The NG-CHM to get the covariate bar from.
fullname	The full name of the covariate bar to get. If no covariate bar with that name exists, return NULL.
where	The axis or axes on which to look for the covariate bar Can be "row", "column", or "both" (default).

Value

An ngchmBar or NULL.

See Also

ngchmBar

chmNewCovariateBar

chmCovariate

Examples

```
# Examples using `chmNew()` require git to be installed and available.
## Not run:
  # If the NGCHMDemoData package is installed, use it to demo usage
  if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
    # Create example NGCHM with covariate bar
    data(TCGA.GBM.Demo, package = "NGCHMDemoData")
    hm <- chmNew("gbmexpr", TCGA.GBM.ExpressionData[1:50, 1:50])
    hm <- chmAddCovariateBar(
      hm, "column",
      chmNewCovariate("TP53 Mutation", TCGA.GBM.TP53MutationData[1:50])
    )
    # Get covariate bar by name
    tp53_covariate_bar <- chmCovariateBar(hm, "TP53 Mutation")
  }
  # Small example not requiring NGCHMDemoData
  matrix <- matrix(rnorm(100),
    nrow = 10, ncol = 10,
    dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
  )
  hm <- chmNew("Demo", matrix)
  covariate <- setNames(rnorm(10), colnames(matrix))
  hm <- chmAddCovariateBar(hm, "column", chmNewCovariate("my covariate", covariate))
  my_covariate_bar <- chmCovariateBar(hm, "my covariate")

## End(Not run)
```

chmCreateCollection *Create a new NG-CHM Collection*

Description

This function creates a new NG-CHM (Next-Generation Clustered Heat Map) collection on the server.

Usage

```
chmCreateCollection(path, recursive = FALSE)
```

Arguments

path	The path where the collection should be created. This should be a single character string.
recursive	A logical value indicating whether to create parent collections if they do not exist. Default is FALSE.

Details

The path is a sequence of components separated by slashes (/). If the path begins with a double slash (//) the following component is interpreted as a server name. If the server name is omitted (i.e. empty) the default server will be used. If the path does not begin with a double slash, the current server will be used.

If the path begins with a slash, the components (following the server, if specified) are interpreted relative to the root collection of the server concerned. Otherwise, they are interpreted relative to the current collection.

The interpretation of each path component is server specific.

Value

None. This function is used for its side effects of creating a new collection on the server.

See Also

`chmCurrentCollection()`

---

chmCreateManagedServer

Create an ngchmServer object for a managed NG-CHM server

Description

Create an ngchmServer object called 'serverName' (see details). The new ngchmServer object is returned and registered so that it can be referenced by name, including retrieval using chmServer. This library will communicate with the NG-CHM using the private address. Returned URLs for viewing NG-CHMs will use the public address.

Usage

```
chmCreateManagedServer(
  serverName,
  privateAddr,
  publicAddr = NULL,
  chmPort = 80,
  managerPort = 18080,
  serviceName = "default",
  ...
)
```

Arguments

`serverName`	The name of the new server object.
`privateAddr`	Private IP name/address of the server.
`publicAddr`	Public IP name/address of the server.
`chmPort`	Port on which the chm viewer is listening.
`managerPort`	Port on which the chm manager is listening.
`serviceName`	Name of the chmManager service
`...`	Additional options passed to chmCreateServer

Value

The created (and registered) ngchmServer object.

See Also

`chmServer()`

`chmCreateServer()`

`chmCreateServer` *Create an ngchmServer object from a specification.*

Description

Create an ngchmServer object called 'serverName' from the specification 'serverSpec' (see details). serverOptions override those in the specification files option by option. The new ngchmServer object is returned and registered so that it can be referenced by name, including retrieval using chmServer.

Usage

```
chmCreateServer(serverName, serverSpec = NULL, serverOptions = NULL)
```

Arguments

`serverName`	The name of the new server object.
`serverSpec`	The specification for the server (defaults to servername).
`serverOptions`	A named list of server options.

Details

serverSpec can be any of:

- **A configuration directory path.** The specification will be read from a file 'config.txt' in that directory.
- **An NGCHM server URL (ending in '/chm' or '/Viewer' for instance).** A minimal specification will be inferred. Known methods for uploading NGCHMs to the server will be autoprobed unless specified manually.
- **A URL referencing a configuration file (must end in '/config.txt').** The specification will be read from the specified URL.

serverOptions can include both protocol-specific options and the following generic options:

- **'serverURL'.** The URL for the NGCHM server.
- **'serverProtocol'.** The protocol to be used for uploading etc. NGCHMs to the server.
- **'jarFile'.** The jarFile used to build NGCHMs.
- **'traceLevel'.** The amount of trace to output. Defaults to "PROGRESS".

Value

The created (and registered) ngchmServer object.

See Also

`chmServer()`

ngchmServer

`ngchmGetServerProtocol()`

ngchmServerProtocol

`chmCurrentCollection` *Get the user's current collection*

Description

Get the user's current collection

Usage

```
chmCurrentCollection()
```

Value

the identity of the current collection

See Also

`chmSetCollection()`

`chmCurrentServer` *Get the user's current server*

Description

Get the user's current server

Usage

```
chmCurrentServer()
```

Value

the identity of the current server

See Also

`chmListServers()`

`chmServer()`

`chmSetCollection()`

`chmDefaultColOrder` *Return default column order of an NGCHM*

Description

Return default column order of an NGCHM

Usage

```
chmDefaultColOrder(chm)
```

Arguments

`chm`	An NGCHM containing at least one layer

Value

Shaid of a dendrogram suitable for use as the chm's column order.

`chmDefaultRowOrder` *Return default row order of an NGCHM*

Description

Return default row order of an NGCHM

Usage

```
chmDefaultRowOrder(chm)
```

Arguments

`chm`	An NGCHM containing at least one layer

Value

Shaid of a dendrogram suitable for use as the chm's row order.

`chmDeployServer` *Get the name of a NGCHM server.*

Description

Return the name of a Next Generation Clustered Heat Map (NGCHM) server.

Usage

```
chmDeployServer(server)

## S4 method for signature 'ngchmServer'
chmDeployServer(server)
```

Arguments

`server`	The server whose name is required.

Value

The name of the server.

See Also

ngchmServer

chmExportToFile *Export a standalone NGCHM to a file.*

Description

Create a standalone viewer for the NGCHM in the specified file. This function requires **Java 11** and the **NGCHMSupportFiles** package.

Usage

```
chmExportToFile(
  chm,
  filename,
  overwrite = FALSE,
  shaidyMapGen,
  shaidyMapGenJava,
  shaidyMapGenArgs
)
```

Arguments

`chm`	The NGCHM to export
`filename`	The file in which to save the rendered NGCHM
`overwrite`	Overwrite file iff true (default false)
`shaidyMapGen`	Path to shaidyMapGen jar file (default to value of environment variable SHAIDYMAPGEN)
`shaidyMapGenJava`	Path to java executable with which to run shaidyMapGen (default to value of environment variable SHAIDYMAPGENJAVA or java)
`shaidyMapGenArgs`	Additional arguments to pass to java when running shaidyMapGen (default to value of environment variable SHAIDYMAPGENARGS)

Details

The NGCHMSupportFiles package can be installed from the R-universe repository:

```
install.packages('NGCHMDemoData',
repos = c('https://md-anderson-bioinformatics.r-universe.dev',
'https://cloud.r-project.org'))
```

Value

the rendered NGCHM

chmExportToHTML *Export a standalone HTML containing the NGCHM to a file.*

Description

Create a standalone HTML containing the NGCHM in the specified file. This function requires **Java 11** and the **NGCHMSupportFiles** package.

Usage

```
chmExportToHTML(
  chm,
  filename,
  overwrite = FALSE,
  shaidyMapGen,
  shaidyMapGenJava,
  shaidyMapGenArgs,
  ngchmWidgetPath
)
```

Arguments

`chm` The NGCHM to generate the HTML for

`filename` The file in which to save the HTML

`overwrite` Overwrite file iff true (default false)

`shaidyMapGen` Path to shaidyMapGen jar file (default to value of environment variable SHAIDYMAPGEN)

`shaidyMapGenJava` Path to java executable with which to run shaidyMapGen (default to value of environment variable SHAIDYMAPGENJAVA or java)

`shaidyMapGenArgs` Additional arguments to pass to java when running shaidyMapGen (default to value of environment variable SHAIDYMAPGENARGS)

`ngchmWidgetPath` Path to location of ngchm Widget (ngchmWidget-min.js). Defaults to environment variable NGCHMWIDGETPATH.

Details

The NGCHMSupportFiles package can be installed from the R-universe repository:

```
install.packages('NGCHMDemoData',
repos = c('https://md-anderson-bioinformatics.r-universe.dev',
'https://cloud.r-project.org'))
```

Value

filename

chmExportToPDF *Export a PDF of the NGCHM to a file.*

Description

Create a PDF of the NGCHM in the specified file. This function requires **Java 11** and the **NGCHMSupportFiles** package.

Usage

```
chmExportToPDF(
  chm,
  filename,
  overwrite = FALSE,
  shaidyMapGen,
  shaidyMapGenJava,
  shaidyMapGenArgs
)
```

Arguments

`chm`	The NGCHM to generate the PDF for
`filename`	The file in which to save the PDF
`overwrite`	Overwrite file iff true (default false)
`shaidyMapGen`	Path to shaidyMapGen jar file (default to value of environment variable SHAIDYMAPGEN)
`shaidyMapGenJava`	Path to java executable with which to run shaidyMapGen (default to value of environment variable SHAIDYMAPGENJAVA or java)
`shaidyMapGenArgs`	Additional arguments to pass to java when running shaidyMapGen (default to value of environment variable SHAIDYMAPGENARGS)

Details

The NGCHMSupportFiles package can be installed from the R-universe repository:

```
install.packages('NGCHMDemoData',
repos = c('https://md-anderson-bioinformatics.r-universe.dev',
'https://cloud.r-project.org'))
```

Value

filename

chmFieldAccessFunction

Get Javascript function name for accessing a specific string field in each element of string vector.

Description

This function returns the name of a Javascript function thats accepts a string vector as its parameter, and for each string in the vector splits the string into fields separated by fieldsep, and accesses field idx (zero origin). The function returns a vector of these fields.

Usage

```
chmFieldAccessFunction(fieldsep, idx)
```

Arguments

`fieldsep`	The separator to be used for splitting the input string. This should be a single character string.
`idx`	The index (zero origin) of the field to be returned after splitting the input string. This should be a single integer.

Details

The name of the function returned for a specific fieldsep and idx will be constant within an R session, but may differ between R sessions (or if this library is unloaded and reloaded).

Value

The name of the newly created field access function.

See Also

`chmGetFunction()`

`chmStringopFunction()`

Examples

```
# Create a new field access function that splits the input string at ',' and
# returns the first field.
chmFieldAccessFunction(',', 1)
# Create a new field access function that splits the input string at '-' and
# returns the second field.
chmFieldAccessFunction('-', 2)
```

chmGetDataset *Get the dataset from an NG-CHM object*

Description

This function retrieves the dataset associated with a specific NG-CHM (Next-Generation Clustered Heat Map).

Usage

```
chmGetDataset(object)

## S4 method for signature 'ngchmLayer'
chmGetDataset(object)
```

Arguments

`object`	An NG-CHM object containing an ngchmDataset

Value

The dataset associated with the specified object.

chmGetDeployServerConfig *Get per-user configuration for a specific deploy Server.*

Description

This function retrieves the configuration of a specified NG-CHM (Next-Generation Clustered Heat Map) deployment server.

Usage

```
chmGetDeployServerConfig(server)
```

Arguments

`server`	The server for which the configuration is to be retrieved. This can be either a character string representing the server name or an object of class 'ngchm-Server'.

Value

The configuration of the specified server if it exists, otherwise NULL.

chmGetFunction *Get a predefined Javascript function for use in NGCHM menus*

Description

This function returns a predefined Javascript function that can be used when building a Next Generation Clustered Heat Map.

Usage

```
chmGetFunction(name)
```

Arguments

name	The name of the predefined Javascript function desired.

Value

An object of class ngchmFunction if found, NULL otherwise.

See Also

chmAddMenuItem()

chmNewFunction()

ngchmAxisFunction

ngchmMatrixFunction

chmGetOverview *Get the file path to the specified overview file.*

Description

This function returns the file path to the specified overview image of the CHM. The CHM must be made before the file can be accessed. If idx is specified, format if given must equal that of the overview image, and the path to that overview image is returned. If idx is not specified, the file path to the first overview of the given format (default 'png') is returned.

Usage

```
chmGetOverview(chm, format = NULL, idx = NULL)
```

Arguments

`chm`	The CHM for which the overview is to be retrieved.
`format`	The format of overview image desired (defaults to ’png’ if idx is not specified).
`idx`	The index of the overview image desired (defaults to first image of the specified format).

Value

The path to the retrieved overview.

`chmGetProperty` *Get Property from NG-CHM*

Description

This function retrieves a specific property from a NG-CHM (Next-Generation Clustered Heat Map) object.

Usage

```
chmGetProperty(object, label)

## S4 method for signature 'ngchmVersion2,character'
chmGetProperty(object, label)
```

Arguments

`object`	An object of class ’ngchmVersion2’ representing the NG-CHM from which the property is to be retrieved.
`label`	A single character string specifying the label of the property to be retrieved.

Value

The property associated with the specified label in the ’ngchmVersion2’ object.

chmGetTypeInfo *Get information about a type name.*

Description

This function gets any registered information about a type name used for determining row and column linkouts. Registration of a typename is (currently) not required in order to use it, so it's possible for valid type name not to have any registered information.

Usage

```
chmGetTypeInfo(typename)
```

Arguments

typename	The name of the type.

Value

Object of class "ngchm.type.info" containing basic information about the type.

See Also

chmListTypes()

chmRegisterType()

chmGetURL *Get the URL for an installed NGCHM.*

Description

Return the URL for accessing the specified Next Generation Clustered Heat Map (NGCHM) on the specified server.

Usage

```
chmGetURL(chm, ...)

## S4 method for signature 'character'
chmGetURL(chm, server = NULL, ...)

## S4 method for signature 'ngchm'
chmGetURL(chm, server = NULL, ...)
```

Arguments

`chm`	A single character string specifying the name of the NG-CHM.
`...`	Ignored.
`server`	The server on which to view the NGCHM

Value

A character string representing the URL of the specified NG-CHM on the specified server.

See Also

ngchmServer

ngchm

chmHasProperty *Determine if the NG-CHM has the given property.*

Description

This function checks if a specific property exists in a NG-CHM (Next-Generation Clustered Heat Map) object.

Usage

```
chmHasProperty(object, label)

## S4 method for signature 'ngchmVersion2,character'
chmHasProperty(object, label)
```

Arguments

`object`	An object of class 'ngchmVersion2' representing the NG-CHM to be checked.
`label`	A single character string or a vector of character strings specifying the label(s) of the property(ies) to be checked.

Value

A logical value indicating whether the specified property(ies) exist in the 'ngchmVersion2' object. If 'label' is a vector, a logical vector is returned.

chmInstall *Add an NG-CHM to an NG-CHM collection.*

Description

Add the given Next-Generation Clustered Heat Map (NG-CHM) to the specified collection (default: current collection).

Usage

```
chmInstall(chm, ...)

## S4 method for signature 'ngchm'
chmInstall(chm, path, ...)
```

Arguments

`chm`	The NGCHM to install.
`...`	Additional server (protocol) specific parameters.
`path`	The path to the collection in which to install the NGCHM.

Value

The installed chm.

See Also

ngchmServer

ngchm

`chmUninstall()`

`chmMakePrivate()`

`chmMakePublic()`

chmLabel *Get the label/name of an NG-CHM object.*

Description

Get the label/name of an NG-CHM object.

Usage

```
chmLabel(x)
```

Arguments

`x`	The NG-CHM object to get the label/name of. Can be: • An object of class ngchm • An object of class ngchmLayer • An object of class ngchmDataset • An object of class ngchmBar • An object of class ngchmCovariate • An object of class ngchmColormap

Value

A character string (or a vector of strings for an ngchmColormap)

See Also

ngchm

Examples

```
chmLabel(chmNew("New CHM"))
```

`chmLabel<-` *Set the label/name of an NG-CHM object*

Description

Set the label/name of an NG-CHM object

Usage

```
chmLabel(x) <- value
```

Arguments

`x`	The NG-CHM object on which to set the label/name.
`value`	The new name (a single character string).

Value

The modified NG-CHM object.

See Also

chmLabel

Examples

```
hm <- chmNew("Old name")
chmLabel(hm) <- "A new name"
```

chmLayer *Get a specified Data Layer from an NG-CHM.*

Description

This function returns a Data Layer contained in a Next Generation Clustered Heat Map.

Usage

```
chmLayer(hm, label)
```

Arguments

hm	The NG-CHM object to get the data layer from.
label	The name or index of the data layer to get. If a name, return the layer with that name. If no layer with that name exists or if the index is out of range, return NULL.

Value

An object of class ngchmLayer or NULL.

See Also

ngchmLayer

Examples

```
# Examples using `chmNew()` require git to be installed and available.
## Not run:
 # If the NGCHMDemoData package is installed, use it to create an example usage
 if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
    # Create example NGCHM
    data(TCGA.GBM.Demo, package = "NGCHMDemoData")
    matrix <- TCGA.GBM.ExpressionData[1:50, 1:50]
    hm <- chmNew("New Heat Map") + chmNewDataLayer("my layer", matrix)
    layer <- chmLayer(hm, "my layer")
    same_layer <- chmLayer(hm, 1)
  }
  # Small example not requiring NGCHMDemoData
  matrix <- matrix(rnorm(100),
    nrow = 10, ncol = 10,
    dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
  )
  hm <- chmNew("New Heat Map") + chmNewDataLayer("my layer", matrix)
  layer <- chmLayer(hm, "my layer")
  same_layer <- chmLayer(hm, 1)

## End(Not run)
```

chmLayer<- *Set (or append) a specified Data Layer in an NG-CHM.*

Description

This function sets a Data Layer in a Next Generation Clustered Heat Map.

Arguments

x	The NG-CHM object to set the data layer of
label	The name or index of the data layer to set. If a name, replace the layer with that name. Append a new layer if no layer with that name exists. If an index, replace the specified layer. If zero (0), prepend the new layer. If minus one (-1) or N+1 (for an NG-CHM with N layers), appends a new layer.
colors	A colormap for the new layer. If missing, defaults to the color map of the layer being replaced, or to the default new layer color map for a new layer.
summarizationMethod	The summarization method for the new layer. If missing, defaults to the summarization method of the layer being replaced, or to the default new layer summarization method for a new layer.
cuts_color	The cuts color for the new layer. If missing, defaults to the cuts color of the layer being replaced, or to the default cuts color for a new layer.
value	Either a matrix or a data layer to set in the NG-CHM. If value is a matrix, the other data layer parameters (label, colors, summarizationMethod, and cuts_color) are set from the parameters if specified, from the old data layer (if any), or the defaults for a new data layer (see chmNewDataLayer). If value is a data layer, any other data layer parameters specified will override those in the replacement layer.

Value

An object of class ngchm.

See Also

ngchmLayer

chmNewDataLayer

Examples

```
# If the NGCHMDemoData package is installed, use demo usage
if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
  data(TCGA.GBM.Demo, package = "NGCHMDemoData")
  matrix <- TCGA.GBM.ExpressionData[1:50, 1:50]
  hm <- chmNew("New Heat Map")
  chmLayer(hm, "Layer 1") <- matrix
  chmLayer(hm, 1, cuts_color = "#fefefe") <- chmNewDataLayer("New data layer", matrix + 1)
}
# Small example not requiring NGCHMDemoData
matrix <- matrix(rnorm(100),
  nrow = 10, ncol = 10,
  dimnames = list(paste0("r", 1:10), paste0("c", 1:10))
)
hm <- chmNew("New Heat Map")
chmLayer(hm, "Layer 1") <- matrix
chmLayer(hm, 1, cuts_color = "#fefefe") <- chmNewDataLayer("New data layer", matrix + 1)
```

---

chmListFunctions *List the predefined Javascript functions available for use in NGCHM menus.*

Description

This function lists the predefined Javascript functions available for use in NGCHM menus.

Usage

```
chmListFunctions(re = ".*")
```

Arguments

`re` The regular expression to match. This should be a single character string. Default is ".*", which matches all functions.

Value

A string containing the names and descriptions of the matching functions.

See Also

`chmAddMenuItem()`

`chmGetFunction()`

`chmRegisterFunction()`

`grep()`

Examples

```
chmListFunctions() # List all functions.
chmListFunctions('^chm') # List all functions whose names start with 'chm'.
```

---

chmListServers *List NG-CHM Servers*

Description

This function lists all NG-CHM (Next-Generation Clustered Heat Map) servers that are currently available.

Usage

```
chmListServers()
```

Value

A character vector containing the names of all available servers.

Examples

```
servers <- chmListServers() # Get a list of all available servers.
```

---

chmListTypes *List known axis types.*

Description

This function returns a list of the axis types for which axis- or matrix- menu entries may be defined.

Usage

```
chmListTypes(re = ".*")
```

Arguments

re	Only types with names matching re are returned (default ".*")

Value

a character vector of axis type names

See Also

chmAddAxisType()

chmLoadCHM *Load CHM from NG-CHM server*

Description

Load an R CHM object from an NG-CHM server. The CHM concerned must have been built using this library, version 0.9.4 or later.

Usage

```
chmLoadCHM(serverOrURL, name)

## S4 method for signature 'ngchmServer,character'
chmLoadCHM(serverOrURL, name)

## S4 method for signature 'character,character'
chmLoadCHM(serverOrURL, name)

## S4 method for signature 'character,missing'
chmLoadCHM(serverOrURL, name)
```

Arguments

`serverOrURL`	An object of class ’ngchmServer’ representing the server from which the NG-CHM is to be loaded.
`name`	A single character string specifying the name of the NG-CHM to be loaded.

Value

An object of class ’ngchm’ representing the loaded NG-CHM.

chmLoadShaidyCHM *Load an NG-CHM from an NG-CHM server.*

Description

Load an NG-CHM from an NG-CHM server.

Usage

```
chmLoadShaidyCHM(mapid, debug = FALSE)
```

Arguments

`mapid`	An NG-CHM ShaidyID that identifies the NG-CHM to download.
`debug`	If TRUE, return a list containing additional information .

Value

An object of class ngchm.

See Also

`chmInstall()`

`ngchmPushSourceServer()`

`chmMake` *Compile a NGCHM.*

Description

Deprecated. Users should no longer call this method directly.

Usage

```
chmMake(chm, ...)

## S4 method for signature 'ngchm'
chmMake(chm, ...)
```

Arguments

`chm`	The NGCHM to compile.
`...`	Additional chmMake options that depend on the format of the NGCHM. For details of the additional parameters of format x see ngchmMakeFormat.x (e.g. ngchmMakeFormat.original).

Details

Compiles the specified Next Generation Clustered Heat Map (NGCHM) in preparation for installation.

Value

The chm

See Also

ngchmServer

ngchm

`chmNew()`

`chmInstall()`

`ngchmMakeFormat.original()`

chmMakePrivate *Make NG-CHM Private on Server*

Description

This function makes a specific NG-CHM (Next-Generation Clustered Heat Map) private on a specified server.

Usage

```
chmMakePrivate(server, chm)

## S4 method for signature 'ngchmServer,character'
chmMakePrivate(server, chm)

## S4 method for signature 'ngchmServer,ngchm'
chmMakePrivate(server, chm)

## S4 method for signature 'character,ngchm'
chmMakePrivate(server, chm)

## S4 method for signature 'character,character'
chmMakePrivate(server, chm)
```

Arguments

server	An object of class 'ngchmServer' representing the server where the NG-CHM is hosted.
chm	A single character string specifying the name of the NG-CHM to be made private.

Value

No return value. The function is called for its side effect of making the specified NG-CHM private on the specified server.

See Also

ngchmServer

ngchm

chmInstall()

chmUninstall()

chmMakePublic()

chmMakePublic *Make NG-CHM Public on Server*

Description

This function makes a specific NG-CHM (Next-Generation Clustered Heat Map) public on a specified server.

Usage

```
chmMakePublic(server, chm)

## S4 method for signature 'ngchmServer,character'
chmMakePublic(server, chm)

## S4 method for signature 'ngchmServer,ngchm'
chmMakePublic(server, chm)

## S4 method for signature 'character,ngchm'
chmMakePublic(server, chm)

## S4 method for signature 'character,character'
chmMakePublic(server, chm)
```

Arguments

`server`	An object of class 'ngchmServer' representing the server where the NG-CHM is hosted.
`chm`	A single character string specifying the name of the NG-CHM to be made public.

Value

No return value. The function is called for its side effect of making the specified NG-CHM public on the specified server.

See Also

ngchmServer

ngchm

`chmInstall()`

`chmUninstall()`

`chmMakePrivate()`

chmManager *Open the NG-CHM Manager*

Description

This function opens a web browser to view the NG-CHM (Next-Generation Clustered Heat Map) Manager on the specified server.

Usage

```
chmManager(server = NULL, viewer = NULL)
```

Arguments

server	The NG-CHM server to be browsed. If NULL, the function will use the first server in the list of available servers.
viewer	The function to be used to open the web browser. If NULL, the function will use the 'browseURL' function.

Value

None. This function is used for its side effects of opening a web browser to view the NG-CHM Manager.

See Also

`utils::browseURL()`

chmName *Get the name of a NGCHM.*

Description

This function returns the name of a Next Generation Clustered Heat Map (NGCHM) object.

Usage

```
chmName(chm)

## S4 method for signature 'ngchm'
chmName(chm)
```

Arguments

chm	The CHM for which the name is required.

Value

A string.

See Also

ngchm

---

chmNew *Create a new NGCHM.*

Description

This function creates a Next Generation Clustered Heat Map (NGCHM) object in memory. Additional parameters will be added to the new NGCHM (see chmAdd). The bare NGCHM needs at least one data layer added to it before it can be compiled. This function requires **git** to be installed.

Usage

```
chmNew(
  name,
  ...,
  rowOrder = chmDefaultRowOrder,
  rowDist = "correlation",
  rowAgglom = "ward.D2",
  colOrder = chmDefaultColOrder,
  colDist = "correlation",
  colAgglom = "ward.D2",
  rowAxisType = NULL,
  colAxisType = NULL,
  rowCovariates = NULL,
  colCovariates = NULL,
  format = "original",
  rowGapLocations = NULL,
  rowGapWidth = 5,
  colGapLocations = NULL,
  colGapWidth = 5,
  overview = c(),
  logLevel = "INFO",
  logFile = NULL
)
```

Arguments

name	The name under which the NGCHM will be saved to the NGCHM server.
...	Zero or more initial objects to include in the NGCHM (see chmAdd).
rowOrder	A vector, dendrogram, or function specifying the CHM row order.

`rowDist` Distance method to use by default RowOrder

`rowAgglom` Agglomeration method to use by default RowOrder

`colOrder` A vector, dendrogram, or function specifying the CHM column order.

`colDist` Distance method to use by default ColOrder

`colAgglom` Agglomeration method to use by default ColOrder

`rowAxisType` The type(s) of the row labels (default: None).

`colAxisType` The type(s) of the column labels (default: None).

`rowCovariates` Covariate(Bar)(s) to add to the rows (default: None).

`colCovariates` Covariate(Bar)(s) to add to the columns (default: None).

`format` The format of NGCHM to produce (default: 'original').

`rowGapLocations` Locations for row gaps. Specify as a list of integers or `chmTreeGaps()` function.

`rowGapWidth` Width of row gaps (default: 5 rows)

`colGapLocations` Locations for col gaps. Specify as a list of integers or `chmTreeGaps()` function.

`colGapWidth` Width of col gaps (default: 5 cols)

`overview` The format(s) of overview image(s) to create (default: None).

`logLevel` The level of logs to output

`logFile` The file to which logs should be output

Value

An object of class ngchm

See Also

ngchm

ngchmServer

`chmAdd()`

`chmAddAxisType()`

`chmAddCovariateBar()`

`chmAddProperty()`

`chmAddOverview()`

`chmInstall()`

`chmExportToFile()`

`chmExportToPDF()`

`chmExportToHTML()`

Examples

```
mychm <- chmNew("test_chm")
mychm <- chmNew("test_chm", rowGapLocations = c(3, 5))
mychm <- chmNew("test_chm", rowGapLocations = chmTreeGaps(4))
mychm <- chmNew("test_chm", rowGapWidth = 3)
```

---

chmNewColorMap *Create a new Color Map for use in constructing a NGCHM*

Description

This function creates a new Color Map suitable for use in constructing Data Layers and Covariates in Next Generation Clustered Heat Maps. Color maps can be used in both discrete and continuous contents. In a discrete context, values specifies the properties of series. In a continuous context, values specifies the break points.

Usage

```
chmNewColorMap(
  values,
  colors = NULL,
  names = NULL,
  shapes = NULL,
  zs = NULL,
  type = "linear",
  missing.color = NULL,
  palette = NULL
)
```

Arguments

`values`	A vector specifying the series / break points for which the following colors are defined, or a data matrix.
`colors`	Either a string vector specifying the color to use for each series / break point, or a single integer.
`names`	A string vector specifying 'human-readable' names for each series / break point.
`shapes`	A string vector specifying the shape to use for each series.
`zs`	A numeric vector specifying the z order to use for each series.
`type`	The string "linear" (default) or "quantile" (or unique abbreviation thereof).
`missing.color`	A string specifying the color to use for missing data.
`palette`	A function(n) that returns a vector of n colors.

Details

If values is a matrix, the function will estimate a suitable sequence of color break points. For a quantile color map, the matrix data is ignored. For a linear color map, it will use equispaced values between a low value and a high value. The low value is the median of the minima of each row in the matrix, and the high value is the median of the row maxima. If the low and high values have different signs, the values will be symmetric about zero.

Value

An object of class ngchmColormap

See Also

ngchmColormap

chmNewDataLayer()

chmNewCovariateBar()

Examples

```
noise.colors <- chmNewColorMap(c(0, 1, 2),
  c("green", "black", "red"),
  missing.color = "yellow"
)
bar.colors <- chmNewColorMap(c("small", "big"),
  c("#00FFFF", "#FF00FF"),
  type = "quantile"
)
```

---

chmNewCovariate *Create a new Covariate for adding to an NGCHM auxilary dataset.*

Description

This function creates a new Covariate suitable for a covariate bar or attaching to an NGCHM auxilary dataset.

Usage

```
chmNewCovariate(
  fullname,
  values,
  value.properties = NULL,
  type = NULL,
  covabbv = NULL
)
```

Arguments

`fullname` The full (human readable) name of the covariate.

`values` A named vector of values (character, logical, or numeric).

`value.properties`
An ngchmColormap mapping values to properties.

`type` The string "discrete" or the string "continuous". (Defaults to continuous for numeric values, to discrete otherwise.)

`covabbv` The short R-compatible identifier used to identify the covariate (derived from fullname if not specified).

Value

An object of class ngchmCovariate.

See Also

ngchmCovariate

`chmAddCovariate()`

`chmNewColorMap()`

`chmNewCovariateBar` *Create a new covariate Bar for a NGCHM*

Description

This function creates a new covariate bar suitable for adding to a Next Generation Clustered Heat Map.

Usage

```
chmNewCovariateBar(
  covar,
  display = "visible",
  thickness = as.integer(10),
  merge,
  barType,
  loBound,
  hiBound,
  fgColor,
  bgColor
)
```

Arguments

`covar`	The covariate to be displayed in the bar.
`display`	Whether the covariate bar will be "hidden" or "visible" (default).
`thickness`	The thickness of the covariate bar in pixels. (Default 10).
`merge`	Algorithm for merging covariates when necessary ("average", "peakColor", "specialColor", or "mostCommon").
`barType`	Type of covariate bar ("color_plot", "scatter_plot", "bar_plot"). Default "color_plot".
`loBound`	Low bound for bar and scatter plots. Default minimum data value.
`hiBound`	High bound for bar and scatter plots. Default maximum data value.
`fgColor`	Foreground color for bar and scatter plots. Default black.
`bgColor`	Background color for bar and scatter plots. Default white.

Value

An object of class ngchmBar

See Also

ngchmBar

`chmNewColorMap()`

`chmAddCovariateBar()`

Examples

```
bar.data <- ifelse(rnorm(1000) < 0, "negative", "non-negative")
names(bar.data) <- sprintf("Sample%d", 1:length(bar.data))
bar.colors <- chmNewColorMap(c("negative", "non-negative"),
  c("white", "black"),
  missing.color = "red"
)
covar <- chmNewCovariate("Group", bar.data, bar.colors, "discrete")
bar <- chmNewCovariateBar(covar)
```

`chmNewDataLayer` *Create a new Data Layer for a NGCHM.*

Description

This function creates a new Data Layer suitable for adding to a Next Generation Clustered Heat Map.

Usage

```
chmNewDataLayer(label, data, colors, summarizationMethod, cuts_color)
```

Arguments

`label` The name under which the data layer will be displayed to the user.

`data` A matrix containing the data to display. Must have rownames and colnames.

`colors` A color map specifying how the data should be rendered. If omitted or NULL, a default green-black-red color map will be estimated from the data.

`summarizationMethod` The method to use when summarizing multiple data points per pixel. Possible values are average (default), sample, and mode.

`cuts_color` color of cuts

Value

An object of class ngchmLayer

See Also

ngchmLayer

`chmNewColorMap()`

`chmAddLayer()`

Examples

```
noise <- matrix(runif(1000) + runif(1000 * 1000), nrow = 1000)
rownames(noise) <- sprintf("Row%d", 1:nrow(noise))
colnames(noise) <- sprintf("Col%d", 1:ncol(noise))
noise.colors <- chmNewColorMap(c(0, 1, 2),
  c("green", "black", "red"),
  missing.color = "yellow"
)
layer <- chmNewDataLayer("Noisy Data", noise, noise.colors)
```

chmNewDataset *Create a new Dataset for a NGCHM.*

Description

This function creates a new Dataset suitable for attaching to a Next Generation Clustered Heat Map.

Usage

```
chmNewDataset(
  name,
  description,
  data,
  row.type = NULL,
  column.type = NULL,
  row.covariates = NULL,
  column.covariates = NULL
)
```

Arguments

`name`	The filename prefix under which the dataset will be saved to the ngchm.
`description`	A description of the dataset.
`data`	A matrix containing the data in the dataset. Must have rownames and colnames.
`row.type`	The type, if any, of the dataset rows.
`column.type`	The type, if any, of the dataset columns.
`row.covariates`	An optional list of row covariates.
`column.covariates`	An optional list of column covariates.

Value

An object of class ngchmDataset

See Also

ngchmDataset

ngchmCovariate

`chmAddDataset()`

`chmNewDialog` *Create a new Dialog for a NGCHM.*

Description

This function creates a new Dialog suitable for adding to a Next Generation Clustered Heat Map.

Usage

```
chmNewDialog(id, title, fn)
```

Arguments

`id`	The html id for the dialog.
`title`	The dialog title / menu entry name.
`fn`	The javascript function for customizing the dialog's contents.

Value

An object of class ngchmDialog

See Also

chmAdd()

chmAddDialog()

---

chmNewFunction *Create a new Javascript function for adding to a NGCHM menu.*

Description

This function creates a new Javascript function object for adding to a Next Generation Clustered Heat Map menu.

Usage

```
chmNewFunction(
  name,
  description,
  implementation,
  extraParams = NULL,
  requires = NULL,
  global = FALSE
)
```

Arguments

name	The name of the Javascript function
description	A short description of the Javascript function
implementation	A string containing the javascript code required to define the function. When called the function is passed a list of selected values (e.g. labels). Additional parameters can be declared before the values parameter and must be resolved through currying (binding) before the function is used in menus.
extraParams	An optional list of extra parameters. (Default NULL.)
requires	An optional vector of (custom) Javascript function names that this function requires.
global	A logical: TRUE if should be defined globally, not within a customization section. (Default FALSE.)

Value

An object of class ngchmJS

See Also

ngchmJS

chmAddMenuItem()

chmBindFunction()

chmRegisterFunction()

Examples

```
alertFn <- chmNewFunction("showAlert", "Display the parameter in an alert box",
  "function showAlert(label) { alert(label); }",
  global = TRUE
)
dbLookup <- chmNewFunction(
  "dbLookup", "Lookup the parameter in a database",
  "function showAlert(database, label) { alert(database[label]); }",
  c("database")
)
```

---

chmNewProperty *Create a new Property for adding to a NGCHM.*

Description

This function creates a new Property object for adding to a Next Generation Clustered Heat Map.

Usage

```
chmNewProperty(label, value)
```

Arguments

label	The property label
value	The property value

Value

An object of class ngchmProperty

See Also

ngchm

chmAddProperty()

Examples

```
prop <- chmNewProperty(
  "chm.info.caption",
  "This is a nifty new CHM."
)
```

---

chmNewServer *Create a new object representing a NGCHM server.*

Description

This function creates a new object that represents a NGCHM server.

Usage

```
chmNewServer(
  serverName,
  serverPort = 8080,
  deployServer = NULL,
  protoOpts = NULL,
  jarFile = NULL,
  serverURL = NULL
)
```

Arguments

serverName	The DNS name of the NGCHM server.
serverPort	The port on which the server is listening.
deployServer	The DNS name to use when deploying a NGCHM (defaults to serverName).
protoOpts	A list of protocol-specific parameters
jarFile	The location of the heatmap build jar file to use when making a NGCHM (defaults to jar file on serverURL WS).
serverURL	The URL used to access the NGCHM server (defaults to serverName:serverPort/chm).

Value

An object of class ngchmServer

See Also

ngchmServer

chmInstall()

chmUninstall()

Examples

```
cloudServ <- chmNewServer("dnsname.domain")
```

`chmOriginalColOrder` *Return original column order of an NGCHM*

Description

Return original column order of an NGCHM

Usage

```
chmOriginalColOrder(chm)
```

Arguments

`chm`	An NGCHM containing at least one layer

Value

Shaid of a label order suitable for use as the chm's column order.

`chmOriginalRowOrder` *Return original row order of an NGCHM*

Description

Return original row order of an NGCHM

Usage

```
chmOriginalRowOrder(chm)
```

Arguments

`chm`	An NGCHM containing at least one layer

Value

Shaid of a label order suitable for use as the chm's row order.

chmProperties *Create NG-CHM Properties*

Description

This function creates one or more NG-CHM (Next-Generation Clustered Heat Map) properties.

Usage

```
chmProperties(...)
```

Arguments

`...`	Named arguments representing the properties to be created. Each argument should be a single value of type character, double, integer, or logical.

Value

A list of properties. Each property is represented as a list with two elements: ’label’ and ’value’.

See Also

`chmAdd()`

Examples

```
# Create three properties: 'prop1', 'prop2', and 'prop3'.
props <- chmProperties(prop1 = "value1", prop2 = 2, prop3 = TRUE)
```

chmProperty *Get the value of an NG-CHM property.*

Description

Get the value of an NG-CHM property.

Usage

```
chmProperty(hm, label)
```

Arguments

`hm`	The NG-CHM object to get the property value from.
`label`	The name of the property to get. If no property with that name exists, return NULL.

```
   Well-known property labels used by the NG-CHM system include:

  * "chm.info.caption"  A paragraph describing the NG-CHM's contents (set by user).
 * "chm.info.built.time"  The date and time the NG-CHM was saved (set by system).
```

Value

A property value or NULL.

See Also

ngchm

Examples

```
hm <- chmNew("Empty")
chmProperty(hm, "chm.info.caption")
```

chmProperty<-   *Set the value of an NG-CHM property.*

Description

Set the value of an NG-CHM property.

Usage

```
chmProperty(x, label) <- value
```

Arguments

`x`	The NG-CHM object on which to set the property.
`label`	The name of the property to set. If no property with that name exists, a new property with that name is appended.
`value`	A non-empty vector of character, logical, or numeric values.

Value

The modified NG-CHM object.

See Also

ngchm

Examples

```
hm <- chmNew("Empty")
chmProperty(hm, "chm.info.caption") <- "Nothing to see here"
```

chmRandomColOrder *Return random column order of an NGCHM*

Description

Return random column order of an NGCHM

Usage

```
chmRandomColOrder(chm)
```

Arguments

chm	An NGCHM containing at least one layer

Value

Shaid of a label order suitable for use as the chm's column order.

chmRandomRowOrder *Return random row order of an NGCHM*

Description

Return random row order of an NGCHM

Usage

```
chmRandomRowOrder(chm)
```

Arguments

chm	An NGCHM containing at least one layer

Value

Shaid of a label order suitable for use as the chm's row order.

chmRegisterAxisFunction *Register a predefined Javascript function for use in NGCHM Axis menus.*

Description

This function registers a Javascript function that will be automatically added to the appropriate axis menu(s) when building a Next Generation Clustered Heat Map for axes that match the function's axis type. This function is intended for use by NGCHM system developers.

Usage

```
chmRegisterAxisFunction(type, label, fn)
```

Arguments

`type`	The axis type required by this function.
`label`	The name of the axis menu entry to be used for this function.
`fn`	The Javascript function to register.

Value

None. This function is used for its side effects of registering a new axis function.

See Also

`chmAddAxisType()`

`chmRegisterMatrixFunction()`

`chmRegisterTypeMapper()`

`chmNewFunction()`

chmRegisterFunction *Register a predefined Javascript function for use in NGCHM menus.*

Description

This function registers a Javascript function that can be used when building a Next Generation Clustered Heat Map. This function is intended for use by NGCHM system developers.

Usage

```
chmRegisterFunction(fn)
```

Arguments

`fn` The function to be registered. This should be an object of class 'ngchmJS'.

Value

The registered function.

See Also

chmAddMenuItem()

chmNewFunction()

ngchmAxisFunction

ngchmMatrixFunction

---

chmRegisterGetMetadataFunction

Define and register a Javascript function for obtaining a specific metadata value.

Description

This function defines and registers a Javascript function for obtaining a specific metadata value and returning it as a javascript list. The function is suitable for use as an axis type accessor function.

Usage

```
chmRegisterGetMetadataFunction(functionName, metadataColumnName)
```

Arguments

`functionName` A single character string specifying the name of the function to be registered.

`metadataColumnName`
A single character string specifying the name of the metadata column to be retrieved by the function.

Value

The registered function.

See Also

chmAddAxisType()

chmGetFunction()

chmListFunctions()

chmRegisterMatrixFunction

Register a predefined Javascript function for use in NGCHM Matrix menus.

Description

This function registers a Javascript function that will be automatically added to the matrix menu when building a Next Generation Clustered Heat Map for matrices whose rows and columns match then function's axes types. This function is intended for use by NGCHM system developers.

Usage

```
chmRegisterMatrixFunction(rowtype, columntype, label, fn)
```

Arguments

rowtype	A character vector specifying the row type(s) of the matrix function.
columntype	A character vector specifying the column type(s) of the matrix function.
label	A single character string specifying the label of the matrix function.
fn	The function to be registered. This can be either a function or a character string representing the name of a function.

Value

None. This function is used for its side effects of registering a new function in the NGCHM matrix menues.

See Also

chmAddAxisType()

chmRegisterAxisFunction()

chmRegisterTypeMapper()

chmNewFunction()

chmRegisterToolboxFunction

Register a Javascript function for use in the NGCHM toolbox.

Description

This function registers a Javascript function that can included in the toolbox of an NGCHM. This function is intended for use by NGCHM system developers.

Usage

```
chmRegisterToolboxFunction(tbtype, menulabel, jsfn)
```

Arguments

`tbtype`	A single character string specifying the type of the toolbox function.
`menulabel`	A single character string specifying the menu label of the toolbox function.
`jsfn`	The function to be registered. This should be an object of class 'ngchmJS'.

Value

None. This function is used for its side effects of registering a new toolbox function.

See Also

`chmNewFunction()`

ngchmAxisFunction

ngchmMatrixFunction

chmRegisterType

Register a type name.

Description

This function registers a type name used for determining row and column linkouts. This function is intended to be used by NGCHM system developers to record basic information about the semantic interpretation of a type name. Registration of a typename is (currently) not required in order to use it.

Usage

```
chmRegisterType(typename, description)
```

Arguments

`typename` A character vector specifying the name(s) of the type(s) to be registered.

`description` A single character string specifying the description of the type(s).

Value

None. This function is used for its side effects of registering a new type.

See Also

`chmListTypes()`

`chmGetTypeInfo()`

`chmRegisterTypeMapper()`

`chmRegisterTypeMapper` *Register a predefined Javascript function for converting values from one type to another.*

Description

This function registers a Javascript function that will be automatically added to a Next Generation Clustered Heat Map as required for converting values from one type into another more basic type. This function is intended for use by NGCHM system developers.

Usage

```
chmRegisterTypeMapper(fromtype, totype, op, ...)
```

Arguments

`fromtype` The type of values the function expects as input.

`totype` The type of values the function will produce. The length of totype must be shorter than fromtype.

`op` The operation code for performing the conversion

`...` Additional parameters required for specifying the conversion (op specific)

Value

NULL

op can have the following values:

- 'field' Split source into fields separated by 'separator' and select field 'num' (0 origin)
- 'expr' Compute string expression 'expr'. 'return' value is a vector or a scalar (default)
- 'javascript' Evaluate javascript function 'fn' (deprecated)

See Also

chmAddAxisType()

chmRegisterAxisFunction()

chmRegisterMatrixFunction()

chmNewFunction()

---

chmRegisterTypeSplitter

Define and register a Javascript function for converting a lists of type values into single values.

Description

This function defines and registers a Javascript function for converting a list of type values separated by the specified separator into the single values, and registers it as a type mapper.

Usage

```
chmRegisterTypeSplitter(functionName, listtype, itemtype, separator)
```

Arguments

functionName	A single character string specifying the name of the function to be registered.
listtype	A single character string specifying the type of the list to be split.
itemtype	A single character string specifying the type of the items in the list after splitting.
separator	A single character string specifying the separator to be used for splitting.

Value

None. This function is used for its side effects of registering a new type splitter.

See Also

chmGetFunction()

chmListFunctions()

chmRegisterTypeMapper()

chmRowOrder<- *Set the row order of data shown in a NGCHM.*

Description

This function sets the row order for a NG-CHM (Next-Generation Clustered Heat Map) object.

Usage

```
chmRowOrder(chm) <- value

## S4 replacement method for signature 'ngchm,optDendrogram'
chmRowOrder(chm) <- value
```

Arguments

`chm`	An object of class 'ngchm'.
`value`	An object of class 'optDendrogram' or 'file' specifying the new row order. If value is NULL, the labels will be displayed in the same order they are found in the first data layer. If value is a character vector, the labels will be displayed in that order. If value is a dendrogram, the labels displayed in the order they occur in a depth first traversal of the tree.

Value

An updated 'ngchm' object with the new row order.

See Also

"chmColOrder<-"

chmServer *Get a registered ngchmServer object for use in making and installing NGCHMs*

Description

This function returns a ngchmServer object that can be used when making and installing a Next Generation Clustered Heat Map.

Usage

```
chmServer(name)
```

Arguments

`name`	The name of the ngchmServer desired.

Value

An object of class ngchmServer if found, NULL otherwise. If multiple servers of the same name have been defined (in different namespaces), the most recently defined is returned.

See Also

`chmInstall()`

`chmUninstall()`

ngchmServer

`chmSetCollection` *Set the user's current server and/or collection*

Description

The path is a sequence of components separated by slashes (/). If the path begins with a double slash (//) the following component is interpreted as a server name. If the server name is omitted (i.e. empty) the default server will be used. If the path does not begin with a double slash, the current server will be used.

Usage

```
chmSetCollection(path)
```

Arguments

`path`	A single character string specifying the path of the collection to be set. The path should be in the format '//server/collection'.

Details

If the path begins with a slash, the components (following the server, if specified) are interpreted relative to the root collection of the server concerned. Otherwise, they are interpreted relative to the current collection.

The interpretation of each path component is server specific.

Value

None. This function is used for its side effects of setting the current server and collection.

See Also

`chmCurrentCollection()`

`chmServer()`

`chmListServers()`

chmSetCredentials *Set Access Credentials for NG-CHM Server*

Description

This function sets the credentials for a specific NG-CHM (Next-Generation Clustered Heat Map) server.

Usage

```
chmSetCredentials(resource, credentials)

## S4 method for signature 'ngchmServer,character'
chmSetCredentials(resource, credentials)

## S4 method for signature 'character,character'
chmSetCredentials(resource, credentials)
```

Arguments

`resource`	An object of class ’ngchmServer’ or a character string representing the server for which the credentials are to be set.
`credentials`	A single character string specifying the credentials to be set for the server.

Value

No return value. The function is called for its side effect of setting the credentials for the specified server.

chmSetDeployServerConfig *Specify per-user configuration for a specific deploy Server.*

Description

Specify per-user configuration for a specific deploy Server.

Usage

```
chmSetDeployServerConfig(server, config)
```

Arguments

`server`	An object of class ’chmServer’ or a character string specifying the name of the server.
`config`	A list specifying the configuration to be set for the server.

Value

None. This function is used for its side effects of setting the deployment server configuration.

---

chmStringopFunction — *Get Javascript function name for performing a specific string operation on each element of a string vector.*

Description

This function returns the name of a Javascript function thats accepts a string vector as its parameter, and for each string in the vector performs the operation stringop on the string. Stringop must be valid Javascript code that can be appended to a string value. The function returns a vector of the resulting strings.

Usage

```
chmStringopFunction(stringop)
```

Arguments

`stringop`	A javascript code fragment that can be applied to a string to yield another string.

Details

The name of the function returned for a specific stringop will be constant within an R session, but may differ between R sessions (or if this library is unloaded and reloaded).

Value

A character string specifying the name of the new function.

See Also

`chmGetFunction()`

`chmFieldAccessFunction()`

chmTreeGaps *Creates new treeCuts object*

Description

This function was designed to facilitate setting rowGapLocations and colGapLocations in the `chmNew()` function. See examples section.

Usage

```
chmTreeGaps(numberOfCuts)
```

Arguments

`numberOfCuts`	Number of tree cuts

Value

treeCuts object with specified number of tree cuts

Examples

```
mychm <- chmNew("test_chm", rowGapLocations = chmTreeGaps(5))
```

chmUninstall *Remove an NG-CHM from Server*

Description

This function removes a specific NG-CHM (Next-Generation Clustered Heat Map) from a specified server.

Usage

```
chmUninstall(chm, ...)

## S4 method for signature 'character'
chmUninstall(chm, server = NULL, ...)

## S4 method for signature 'ngchm'
chmUninstall(chm, ...)
```

Arguments

`chm`	A single character string specifying the NG-CHM's name, or an object of class "ngchm" representing the NG-CHM to be uninstalled.
`...`	Additional server (protocol) specific parameters.
`server`	An object of class 'ngchmServer' or a character string representing the server from which the NG-CHM is to be uninstalled. If not provided, the current server is used.

Value

No return value. The function is called for its side effect of uninstalling the specified NG-CHM from the specified server.

See Also

ngchmServer

ngchm

`chmInstall()`

`chmUrlBase` *Get the base URL for a NGCHM installed on a NGCHM server.*

Description

Return the base URL of a Next Generation Clustered Heat Map (NGCHM) that has been installed on a NGCHM server.

Usage

```
chmUrlBase(server)

## S4 method for signature 'ngchmServer'
chmUrlBase(server)
```

Arguments

`server`	The server whose base URL is required.

Value

The base URL for accessing NGCHMs installed on the server.

See Also

ngchmServer

chmWriteCustomJS *Output Javascript code required to customize an NGCHM.*

Description

This function outputs the Javascript required to customize an NGCHM.

Usage

```
chmWriteCustomJS(chm, filename)
```

Arguments

`chm`	An object of class 'chm' representing the heat map.
`filename`	A single character string specifying the name of the file where the JavaScript will be written.

Value

None. This function is used for its side effects of writing the JavaScript to a file.

getDimensions *Generic method to get a dimensions matrix from obj.*

Description

The return value must be NULL or a numeric matrix, each column of which is a (reduced) dimension. The rows of the returned matrix must be named.

Usage

```
getDimensions(obj, ...)

## Default S3 method:
getDimensions(obj, ...)

## S3 method for class 'prcomp'
getDimensions(obj, ...)

## S3 method for class 'umap'
getDimensions(obj, ...)

## S3 method for class 'Seurat'
getDimensions(obj, dimName, ...)
```

Arguments

`obj`	The object from which to obtain the dimension(s).
`...`	Additional class-specific parameters for specifying the desired dimension.
`dimName`	The name of the dimension matrix to obtain.

Value

A matrix with one dimension per column and one named row per observation in obj.

See Also

`chmAddReducedDim()`

`gitHashObject` *Obtain the git hash of an existing file.*

Description

Obtain the git hash of an existing file.

Usage

```
gitHashObject(path)
```

Arguments

`path`	filename of file to hash

Value

a string containing the file hash

`initLogging` *Initialize Logging*

Description

This function initializes logging for the application.

Usage

```
initLogging(log_level, log_file = NULL)
```

Arguments

`log_level` A single character string specifying the log level. This should be one of 'TRACE', 'DEBUG', 'INFO', 'WARN', 'ERROR', or 'FATAL'.

`log_file` An optional character string specifying the name of the file where the log will be written. If this is NULL, the log will be written to the console.

Value

None. This function is used for its side effects of initializing the logging.

NGCHM *Next Generation Clustered Heat Map (NGCHM) Construction Library*

Description

NGCHM provides tools for defining the contents of a new NGCHM, and for compiling and installing it on a NGCHM server.

Details

Typical usage (see example) is to create a base NGCHM using chmNew; extend it with at least one ngchmLayer; typically extend it further with an additional ngchmLayer, row and column dendrograms, classification bars, and popup menu entries; compile and install it on an available ngchmServer.

Note:

- `chmNew()` requires **git** to be installed.
- `chmExportToFile()`, `chmExportToHTML()`, and `chmExportToPDF()` require **Java 11** and the **NGCHMSupportFiles** package. The NGCHMSupportFiles package can be installed with:

```
install.packages('NGCHMDemoData',
repos = c('https://md-anderson-bioinformatics.r-universe.dev',
'https://cloud.r-project.org'))
```

Initialization

When first loaded the NGCHM library reads configuration files in the directories specified by the NGCHMCONFIGPATH environment variable. This is a colon (:) separated list of directory names. If not set it defaults to /etc/ngchm:/usr/local/ngchm:/opt/ngchm:$HOME/.ngchm. See NGCHM-initialization for details.

Author(s)

Maintainer: Mary A Rohrdanz <marohrdanz@mdanderson.org> [contributor]

Authors:

- Bradley M Broom <bmbroom@mdanderson.org> (ORCID)

Other contributors:

- Chris Wakefield <cwakefield@mdanderson.org> [contributor]
- James M Melott <jmmelott@mdanderson.org> [contributor]
- MD Anderson Cancer Center [copyright holder]

See Also

chmNew()

chmAdd()

chmExportToFile()

chmExportToPDF()

chmSetCollection()

chmInstall()

ngchm

Examples

```
# Examples using `chmNew()` require git to be installed.
# The NGCHMSupportFiles package is required by chmExportToFile and chmExportToPDF
# The NGCHMDemoData package is used to create a demo NGCHM
## Not run:
  if (requireNamespace("NGCHMSupportFiles", quietly = TRUE)) {
    if (requireNamespace("NGCHMDemoData", quietly = TRUE)) {
      library(NGCHMSupportFiles)
      library(NGCHMDemoData)
      data(TCGA.GBM.EXPR, package = "NGCHMDemoData")
      chm1 <- chmNew("gbm", TCGA.GBM.EXPR[1:50, 1:50],
        rowAxisType = "bio.gene.hugo",
        colAxisType = "bio.tcga.barcode.sample.vial.portion.analyte.aliquot"
      )
      chmExportToFile(chm1, tempfile("gbm", fileext = ".ngchm"))
      chmExportToPDF(chm1, tempfile("gbm", fileext = ".pdf"))
    }
  }
  mat <- matrix(rnorm(100), nrow = 10)
  rownames(mat) <- sprintf("ABCA%d", 1:10)
  colnames(mat) <- sprintf("Sample%d", 1:10)
  chm <- chmNew("my-chm", mat)

## End(Not run)
```

`ngchm-class` *Class representing a Next Generation Clustered Heat Map (NGCHM) under construction.*

Description

An NG-CHM is produced by creating a heat map object with `chmNew()`, possibly modifying or augmenting it using additional functions, such as `chmAddLayer()`, `chmAddCovariateBar()`, etc., and then either saving it to a server using `chmInstall()` or saving it to a standalone file using `chmExportToFile()`.

See Also

`chmNew()`

`chmRowOrder<-()`

`chmColOrder<-()`

`chmAdd()`

`chmAddLayer()`

`chmAddCovariateBar()`

`chmAddDataset()`

`chmAddAxisType()`

`chmInstall()`

`chmExportToFile()`

`NGCHM-functions` *Javascript extensions for the Next Generation Clustered Heat Map (NGCHM) Construction Library*

Description

Currently:

- Axis function View Ideogram is added for the appropriate axis types.

See Also

`chmGetFunction()`

`chmListFunctions()`

NGCHM-initialization *Initialization of the NGCHM library.*

Description

When first loaded the NGCHM library reads configuration files in the configuration path specified by the NGCHMCONFIGPATH environment variable. The configuration path is a colon (:) separated list of directory names. If not set it defaults to /etc/ngchm:/usr/local/ngchm:/opt/ngchm:$HOME/.ngchm.

Details

For each configuration directory in the configuration path, the NGCHM package reads the contents of the configuration files in the conf.d subdirectory in order (as determined by the R sort function). Other subdirectories are not scanned unless instructed to by an entry in a configuration file.

Configuration files may be either text files (.txt extension), R scripts (.R extension), or javascript files (.js extension).

Here is an example directory structruce for a server named ’my_server’:

```
.
|-- conf.d
|   \-- 00-servers.txt
\-- my_server
    \-- config.txt
```

Here are the contents of an example 00-servers.txt file:

```
[servers]
my-server = /usr/local/ngchm/my_server
```

Here are the contents of an example config.txt file:

```
serverProtocol = shaidy
accessMethod = api
basePath = <URL to server. e.g. "https://mydomain.edu/server/api">
serverURL = <URL to server. e.g. "https://mydomain.edu/server">
```

Value

None. This function is used for its side effects of loading configuration files.

Text files

A text configuration file consists of one or more sections. Each section begins with a single line containing the section type enclosed in square brackets. Subsequent lines in the section are either blank or contain a definition of the form "name separator value". The default separator is the equals sign (=).

The ’servers’ section defines available servers. The name field defines the name by which the server is known to the library. The value field specifies a directory containing a specification of the server’s properties. The server specification directory must contain a config.txt that contains lines of the form "name separator value". The config.txt file must define the value of ’serverProtocol’ to be the name of a ngchmServerProtocol. It must also define the values of any mandatory parameters required by ngchmServerProtocol, and may optionally define any optional parameters.

R scripts

R scripts are sourced. They can be used to define local NGCHM related functions.

Javascript scripts

Javascript files define context specific menu entries.

---

ngchmAddDatasetBlob *Add a data file to a local shaidy repository*

Description

Add a data file to a local shaidy repository

Usage

```
ngchmAddDatasetBlob(shaidyRepo, format, filename, properties = NULL)
```

Arguments

shaidyRepo	The shaidy repository
format	The format of the data file
filename	The filesystem path to the data file
properties	A list of additional properties to save with file

Value

The file’s shaid

ngchmAddMatrixToCollection

Add a matrix reference to a collection

Description

Add a matrix reference to a collection

Usage

```
ngchmAddMatrixToCollection(collection, name, shaid)
```

Arguments

`collection`	A list containing details of a collection
`name`	The name to associate with the matrix reference
`shaid`	The shaid of the matrix to add to the collection

Value

An updated list contaiing details of the collection

ngchmAddObjectToCollection

Add an object reference to a collection

Description

Add an object reference to a collection

Usage

```
ngchmAddObjectToCollection(repo, uuid, shaid)
```

Arguments

`repo`	The repository containing the collection
`uuid`	A collection uuid
`shaid`	The shaid of the object to add to the collection

Value

An updated list containing details of the collection

ngchmAxis-class *Class representing an axis of a Next Generation Clustered Heat Map (NG-CHM).*

Description

Class representing an axis of a Next Generation Clustered Heat Map (NG-CHM).

See Also

`chmAxis()`

ngchmAxisFunction-class *Class representing an axis function for Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing an axis function for Next Generation Clustered Heat Map (NGCHM).

ngchmAxisType-class *Class representing a type attached to an axis in a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a type attached to an axis in a Next Generation Clustered Heat Map (NGCHM).

ngchmBar-class *Class representing a Covariate Bar on a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a Covariate Bar on a Next Generation Clustered Heat Map (NGCHM).

`ngchmCollectionInCollection` *Recursively determine if collection uuid is contained in collection A collecton always contains itself.*

Description

Recursively determine if collection uuid is contained in collection A collecton always contains itself.

Usage

```
ngchmCollectionInCollection(collection, uuid)
```

Arguments

`collection`	A list containing details of a collection
`uuid`	A string containing the UUID to check

Value

TRUE iff collection contains uuid, otherwise FALSE

`ngchmCollectionTree` *Create a recursive description of a collection*

Description

Create a recursive description of a collection

Usage

```
ngchmCollectionTree(collection, depth = 0)
```

Arguments

`collection`	A list containing details of a collection
`depth`	The indentation depth to use

Value

a string vector describing the contents of the collection

ngchmColormap-class *Class representing a Color Map on a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a Color Map on a Next Generation Clustered Heat Map (NGCHM).

ngchmCovariate-class *Class representing a Covariate attached to a Dataset*

Description

Class representing a Covariate attached to a Dataset

ngchmCreateServerProtocol *Create and register an NGCHM server protocol implementation.*

Description

This function creates and registers a protocol implementation for manipulating an NGCHM server.

Usage

```
ngchmCreateServerProtocol(
  protocolName,
  chmFormat,
  requiredParams,
  optionalParams,
  paramValidator,
  findCollection,
  createCollection,
  installMethod,
  uninstallMethod,
  makePrivate,
  makePublic,
  setCredentials
)
```

Arguments

`protocolName` A single character string specifying the name of the protocol.

`chmFormat` A single character string specifying the format of the heat map. Defaults to "original".

`requiredParams` A character vector specifying the required parameters for the protocol, if any.

`optionalParams` A character vector specifying the optional parameters for the protocol, if any.

`paramValidator` A function(list) for validating the parameters specified for a new server.

`findCollection` A function(server,collection,path) that finds a collection on the server.

`createCollection` A function(server,collection,name) that creates a collection on the server.

`installMethod` A function(server,chm) that installs a heat map on the server.

`uninstallMethod` A function(server,chmname) that uninstalls a heat map from the server.

`makePrivate` A function(server,chmname) that makes a heat map private on the server.

`makePublic` A function(server,chmname) that makes a heat map public on the server.

`setCredentials` A function(server,credentialstring) that sets the credentials for the server.

Value

An object of class 'ngchmServerProtocol' representing the new server protocol.

`ngchmCSS-class` *Class representing custom CSS for a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing custom CSS for a Next Generation Clustered Heat Map (NGCHM).

`ngchmDataset-class` *Class representing a Dataset attached to a NGCHM*

Description

Class representing a Dataset attached to a NGCHM

`ngchmDialog-class` *Class representing an addon dialog*

Description

Class representing an addon dialog

`ngchmFindRepo` *Find a repository, if any, that contains the requested shaid*

Description

Find a repository, if any, that contains the requested shaid

Usage

```
ngchmFindRepo(shaid, required = TRUE)
```

Arguments

`shaid`	The shaid to search for
`required`	Abort if requireed and shaid not found in a known repo

Value

The first repository containing the shaid, otherwise NULL. The temporary repositories are searched before source repositories.

`ngchmGetDataFileShaid` *Compute shaid for a data file*

Description

Compute shaid for a data file

Usage

```
ngchmGetDataFileShaid(format, filename)
```

Arguments

`format`	The format of the data file
`filename`	The filesystem path to the data file

Value

The shaid of the data file

`ngchmGetEnv` *Get the ngchm environment (for debugging only).*

Description

Get the library's internal ngchm environment to help debugging.

Usage

```
ngchmGetEnv()
```

Value

A list representing the current environment for NG-CHM.

`ngchmGetHandleHTTR` *Get a HTTR handle for the server's view/WS URL*

Description

This function returns a 'handle' suitable for use with the server's view/WS URL

Usage

```
ngchmGetHandleHTTR(server)
```

Arguments

`server`	An object of class ngchmServer

Value

An HTTR handle

`ngchmGetLabels` *Get the axis labels of a shaidy dataset or dendrogram*

Description

Get the axis labels of a shaidy dataset or dendrogram

Usage

```
ngchmGetLabels(shaid, axis = NULL)
```

Arguments

`shaid`	The shaid of the dataset or dendrogram to get the labels of
`axis`	For datasets, the axis of the labels to get

Value

a list of shaids containing the labels

`ngchmGetLabelsStr` *Get the axis labels of a shaidy dataset or dendrogram*

Description

Get the axis labels of a shaidy dataset or dendrogram

Usage

```
ngchmGetLabelsStr(shaid, axis = NULL)
```

Arguments

`shaid`	The shaid of the dataset or dendrogram to get the labels of
`axis`	For datasets, the axis of the labels to get

Value

A string vector containing the axis labels of the dataset or dendrogram

ngchmGetProtoParam *Get Protocol Parameter for NG-CHM Server*

Description

This function gets a protocol parameter for a specified NG-CHM (Next-Generation Clustered Heat Map) server.

Usage

```
ngchmGetProtoParam(server, option, default = NULL)
```

Arguments

`server`	An object of class 'ngchmServer' representing the server.
`option`	A single character string specifying the name of the protocol parameter.
`default`	An optional default value to return if the protocol parameter is not found. Defaults to NULL.

Value

The value of the protocol parameter if it is found, otherwise the specified default value.

ngchmGetServerProtocol *Get Server Protocol for NG-CHM*

Description

This function gets a server protocol for NG-CHM (Next-Generation Clustered Heat Map) by its name.

Usage

```
ngchmGetServerProtocol(protocolName)
```

Arguments

`protocolName`	A single character string specifying the name of the protocol.

Value

An object of class 'ngchmServerProtocol' representing the server protocol.

ngchmInitShaidyRepository *Initialize Shaidy Repository for NG-CHM*

Description

This function initializes a Shaidy repository for NG-CHM (Next-Generation Clustered Heat Map).

Usage

```
ngchmInitShaidyRepository(shaidyDir)
```

Arguments

`shaidyDir`	A single character string specifying the directory where the Shaidy repository will be initialized.

Value

None. This function is used for its side effects of initializing the Shaidy repository.

ngchmJS-class *Class representing a custom Javascript function for a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a custom Javascript function for a Next Generation Clustered Heat Map (NGCHM).

ngchmLayer-class *Class representing a Layer on a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a Layer on a Next Generation Clustered Heat Map (NGCHM).

ngchmListServerProtocols *List defined server protocols*

Description

List defined server protocols

Usage

```
ngchmListServerProtocols()
```

Value

A character vector

ngchmLoadDatasetBlob *Load a data matrix from a local shaidy repository*

Description

Load a data matrix from a local shaidy repository

Usage

```
ngchmLoadDatasetBlob(shaidyRepo, shaid, datatype)
```

Arguments

shaidyRepo	The shaidy repository
shaid	The shaid of the dataset blob to load
datatype	Prototype of matrix data elements (defaults to 0.0)

Value

a list containing details of the loaded dataset

ngchmMakeFormat.original

Make an original format NGCHM.

Description

Make an original format NGCHM.

Usage

```
ngchmMakeFormat.original(
  chm,
  server,
  deleteOld = TRUE,
  useJAR = NULL,
  javaTraceLevel = NULL,
  javaOptions = NULL,
  buildArchive = NULL
)
```

Arguments

`chm` The original format CHM to compile.

`server` The server for which to compile the NGCHM. Default `getOption("NGCHM.Server",chmListServers()` Required iff useJar is not defined.

`deleteOld` If TRUE, delete any old CHM of this name before beginning build. (Default is TRUE.)

`useJAR` If defined, the location (filename) of the chmbuilder jar file. The package will not download a current jar file from the server. It is the caller’s responsibility to ensure the builder jar file is compatible with the server on which the NGCHM will be installed. (Default is not defined.)

`javaTraceLevel` Trace level option passed to the Java process. Default is getOption("NGCHM.Java.Trace’,’PROGRESS’).

`javaOptions` Additional options to pass to the Java process. Default is getOption(’NGCHM.Java.Options’,’-Xmx2G’).

`buildArchive` If TRUE, build a tar archive of the generated NGCHM. Default is getOption(’NGCHM.Build.Archive’,TR

Value

The CHM

`ngchmMakeFormat.shaidy` *Make a shaidy format NGCHM.*

Description

Make a shaidy format NGCHM.

Usage

```
ngchmMakeFormat.shaidy(chm)
```

Arguments

`chm`	The shaidy format CHM to compile.

Value

The CHM

`ngchmMatrixFunction-class` *Class representing a matrix function for Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a matrix function for Next Generation Clustered Heat Map (NGCHM).

`ngchmMenuItem-class` *Class representing a Menu Item for a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a Menu Item for a Next Generation Clustered Heat Map (NGCHM).

`ngchmMetaData-class` *Class representing meta data attached to an NG-CHM*

Description

Class representing meta data attached to an NG-CHM

ngchmNewBar — *Create a new Classification Bar for a NGCHM*

Description

This function is deprecated and will be removed in a future version. Please use chmNewCovariateBar. This function creates a new Classification Bar suitable for adding to a Next Generation Clustered Heat Map.

Usage

```
ngchmNewBar(
  label,
  type,
  data,
  colors = NULL,
  display = "visible",
  thickness = as.integer(10),
  merge,
  barType,
  loBound,
  hiBound,
  fgColor,
  bgColor
)
```

Arguments

`label`	The name by which the classification bar will be known.
`type`	The string "discrete" or the string "continuous".
`data`	A vector of the data to be displayed in the classification bar. names(data) must be defined.
`colors`	A color map specifying how the data should be rendered.
`display`	Whether the classification bar will be "hidden" or "visible" (default).
`thickness`	The thickness of the classification bar in pixels. (Default 10).
`merge`	Algorithm for merging classifications when necessary ("average", "peakColor", "specialColor", or "mostCommon").
`barType`	Type of covariate bar ("color_plot", "scatter_plot", "bar_plot"). Default "color_plot".
`loBound`	Low bound for bar and scatter plots. Default minimum data value.
`hiBound`	High bound for bar and scatter plots. Default maximum data value.
`fgColor`	Foreground color for bar and scatter plots. Default black.
`bgColor`	Background color for bar and scatter plots. Default white.

Value

An object of class ngchmBar

See Also

ngchmBar

chmNewColorMap()

chmNewCovariateBar()

chmAddCovariateBar()

---

ngchmNewCollection *Create a new collection in a local shaidy repository*

Description

Create a new collection in a local shaidy repository

Usage

```
ngchmNewCollection(shaidyRepo, labels = data.frame())
```

Arguments

shaidyRepo	The shaidy repository
labels	Initial labels for collection (a data.frame of (Name,Value) tuples)

Value

a string containing the UUID of the newly created repository

---

ngchmOverview-class *Class representing an overview of a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing an overview of a Next Generation Clustered Heat Map (NGCHM).

ngchmProperty-class *Class representing a Generic Property for a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a Generic Property for a Next Generation Clustered Heat Map (NGCHM).

ngchmProtoParamCheck *Check Protocol Parameters for NG-CHM*

Description

Check that all required parameters are specified, and all specified parameters are either required or optional.

Usage

```
ngchmProtoParamCheck(params, required, optional)
```

Arguments

`params`	A list of parameters to be checked.
`required`	A character vector specifying the required parameters.
`optional`	A character vector specifying the optional parameters.

Value

None. This function is used for its side effects of checking the parameters and potentially stopping execution with an error message.

ngchmPushSourceRepository *Push a local shaidy repository onto the stack of source repositories*

Description

This function pushes a source repository for NG-CHM (Next-Generation Clustered Heat Map) onto the Shaidy stack.

Usage

```
ngchmPushSourceRepository(shaidyDir, accessMethod = "file")
```

Arguments

shaidyDir
: A single character string specifying the directory of the source repository.

accessMethod
: A single character string specifying the access method for the source repository. Defaults to "file".

Value

None. This function is used for its side effects of pushing the source repository onto the Shaidy stack.

---

ngchmPushSourceServer *Push a shaidy server onto the stack of source repositories*

Description

This function pushes a source server for NG-CHM (Next-Generation Clustered Heat Map) onto the Shaidy stack.

Usage

```
ngchmPushSourceServer(server)
```

Arguments

server
: An object of class 'ngchmServer' or a single character string specifying the name of the server.

Value

None. This function is used for its side effects of pushing the source server onto the Shaidy stack.

See Also

`chmLoadShaidyCHM()`

`chmCreateServer()`

ngchmPushTempRepository *Push Temporary Repository for NG-CHM*

Description

This function pushes a temporary repository for NG-CHM (Next-Generation Clustered Heat Map) onto the Shaidy stack.

Usage

```
ngchmPushTempRepository(shaidyDir)
```

Arguments

shaidyDir	A single character string specifying the directory of the temporary repository.

Value

None. This function is used for its side effects of pushing the temporary repository onto the Shaidy stack.

ngchmRegisterServer *Register an ngchmServer.*

Description

This function registers an ngchmServer that can be used when making and installing a Next Generation Clustered Heat Map.

Usage

```
ngchmRegisterServer(uuid, server)
```

Arguments

uuid	A string that identifies the server namespace.
server	The ngchmServer to register.

Value

the server that was registered

See Also

- `chmInstall()`
- `chmUninstall()`
- `ngchmUnregisterServer()`
- ngchmServer

`ngchmRelated-class` *Class representing a link related to a NGCHM*

Description

Class representing a link related to a NGCHM

`ngchmRelatedGroup-class` *Class representing a group of related links to a NGCHM*

Description

Class representing a group of related links to a NGCHM

`ngchmRenderChm` *Render a shaidy NGCHM*

Description

Render a shaidy NGCHM

Usage

```
ngchmRenderChm(repo, shaid)
```

Arguments

`repo`	The repository containing the chm
`shaid`	The shaid of the chm to render

Value

Nothing

ngchmResponseJSON *Return response content interpreted as JSON*

Description

Return response content interpreted as JSON

Usage

```
ngchmResponseJSON(httrResponse)
```

Arguments

httrResponse	The httr response object

Value

The response parsed as JSON and returned as an R object

ngchmRowCenter *Row center a shaidy dataset*

Description

Row center a shaidy dataset

Usage

```
ngchmRowCenter(shaidyRepo, shaid)
```

Arguments

shaidyRepo	The shaidy repository
shaid	The shaid of the dataset to row center

Value

A list of shaids for the row centered dataset

ngchmSaveAsDatasetBlob

Save a numeric matrix as a blob in a shaidy repository

Description

Save a numeric matrix as a blob in a shaidy repository

Usage

```
ngchmSaveAsDatasetBlob(shaidyRepo, format, mat)
```

Arguments

shaidyRepo	The shaidy repository
format	The format in which to save the matrix
mat	The data matrix

Value

The shaid of the saved blob

ngchmSaveAsDendrogramBlob

Save a dendrogram as a blob in a shaidy repository

Description

Save a dendrogram as a blob in a shaidy repository

Usage

```
ngchmSaveAsDendrogramBlob(shaidyRepo, ddg)
```

Arguments

shaidyRepo	The shaidy repository
ddg	The dendrogram

Value

The shaid of the saved blob

ngchmSaveChmAsBlob *Save an NGCHM as a shaidy blob*

Description

Save an NGCHM as a shaidy blob

Usage

```
ngchmSaveChmAsBlob(shaidyRepo, chm)
```

Arguments

shaidyRepo	The shaidy repository to write to
chm	The NGCHM to write

Value

The shaid of the saved NGCHM

ngchmServer-class *Class representing a Next Generation Clustered Heat Map (NGCHM) server.*

Description

Class representing a Next Generation Clustered Heat Map (NGCHM) server.

ngchmServerProtocol-class *Class representing a deployment method for a Next Generation Clustered Heat Map (NGCHM) server.*

Description

Class representing a deployment method for a Next Generation Clustered Heat Map (NGCHM) server.

`ngchmTemplate-class` *Class representing a Template attached to a NGCHM*

Description

Class representing a Template attached to a NGCHM

`ngchmTileDataset` *Get the tiles for a shaidy dataset*

Description

Get the tiles for a shaidy dataset

Usage

```
ngchmTileDataset(repo, dataset, rowOrder, colOrder)
```

Arguments

`repo`	The shaidy repository in which to create the tile
`dataset`	The shaid of the dataset to tile
`rowOrder`	The row order of the tiles
`colOrder`	The column order of the tiles

Value

a list of shaids containing the tiles

`ngchmTypeMapper-class` *Class representing a type mapper function for Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing a type mapper function for Next Generation Clustered Heat Map (NGCHM).

`ngchmUnregisterServer` *Unregister NG-CHM Server*

Description

This function unregisters a server for NG-CHM (Next-Generation Clustered Heat Map) by its UUID and optionally by its name.

Usage

```
ngchmUnregisterServer(uuid, name = NULL)
```

Arguments

`uuid`	A single character string specifying the UUID of the server.
`name`	The names(s) of the ngchmServer(s) to unregister. If not specified, all ngchmServers in the namespace are unregistered. Defaults to NULL.

Value

None. This function is used for its side effects of unregistering the server.

See Also

`ngchmRegisterServer()`

ngchmServer

`ngchmValueProp-class` *Class representing the properties of a data point in a Next Generation Clustered Heat Map (NGCHM).*

Description

Class representing the properties of a data point in a Next Generation Clustered Heat Map (NGCHM).

ngchmVersion2-class *Class representing ngchmVersion2 object*

Description

Class representing ngchmVersion2 object

Slots

`name` The name under which the NGCHM will be saved to the NGCHM server.

`version` Integer version number (default: 2)

`format` (default: "original")

`uuid` character

`baggage` optCharacter

`inpDir` character

`outDir` character

`saveDir` (default: tempdir())

`propFile` (default: "chm.properties")

`layers` List of data layers

`colormaps` Color map

`rowMenu` optList

`colMenu` optList

`datasets` optList

`dialogs` optList

`tags` optCharacter

`elementMenu` optList

`rowTypeFunctions` optList

`colTypeFunctions` optList

`elementTypeFunctions` optList

`axisTypes` optList

`css` optList

`extrafiles` optCharacter

`extrascripts` optCharacter

`properties` optList

`overviews` optList

`javascript` optList

`rowOrder` A vector, dendrogram, or function specifying the CHM row order

- `rowDist` Distance method to use by default RowOrder. (default: "correlation", which is 1 minus the Pearson correlation among the rows.)
- `rowAgglom` Agglomeration method to use by default RowOrder. Choices are those from stats::hclust. (default: "ward.D2")
- `colOrder` A vector, dendrogram, or function specifying the CHM column order.
- `colDist` Distance method to use by default ColOrder. (default: "correlation", which is 1 minus the Pearson correlation among the cols.)
- `colAgglom` Agglomeration method to use by default ColOrder. Choices are those from stats::hclust. (default: "ward.D2")
- `rowOrderMethod` character (default: "User")
- `colOrderMethod` character (default: "User")
- `rowCutLocations` Explicit list of row cut locations. If specified, rowTreeCuts is set to NULL.
- `rowTreeCuts` Number of tree cuts for row. If specified, rowCutLocations is set to NULL.
- `rowCutWidth` Width of row cuts (default: 5 rows)
- `rowTopItems` optCharacter
- `rowDisplayLength` optInteger
- `rowDisplayAbbreviation` optCharacter
- `colCutLocations` Explicit list of col cut locations. If specified, colTreeCuts is set to NULL.
- `colTreeCuts` Number of tree cuts for col. If specified, colCutLocations is set to NULL.
- `colCutWidth` Width of col cuts (defautl: 5 columns)
- `colTopItems` optCharacter
- `colDisplayLength` optInteger
- `colDisplayAbbreviation` optCharacter
- `rowMeta` optList
- `colMeta` optList
- `rowCovariateBars` optList
- `colCovariateBars` optList
- `relatedLinks` optList
- `relatedGroups` optList
- `templates` optList
- `width` default: 500
- `height` default: 500

`optDendrogram-class` *Optional Dendrogram*

Description

Optional Dendrogram

plot.ngchmVersion2 *Open the NG-CHM on the specified server in the viewer.*

Description

Open the NG-CHM on the specified server in the viewer.

Usage

```
## S3 method for class 'ngchmVersion2'
plot(x, server = NULL, viewer = NULL, ...)
```

Arguments

x	The NGCHM to view.
server	The server containing the NG-CHM. Defaults to option "NGCHM.Server" or the first server.
viewer	The viewer to use. Defaults to option "viewer" or browseURL.
...	Ignored.

Value

No return value. The function is called for its side effect of plotting the specified NG-CHM.

print.ngchm.type.info *Pretty Print NGCHM Type Information*

Description

This function takes an object of class ’ngchm.type.info’ and returns a formatted string that provides a detailed description of the NGCHM type.

Usage

```
## S3 method for class 'ngchm.type.info'
print(x, ...)
```

Arguments

x	An object of class ’ngchm.type.info’ as returned by chmGetTypeInfo.
...	Additional arguments (not used).

Value

A string that provides a detailed description of the NGCHM type.

See Also

chmGetTypeInfo()

print.shaidyRepo *Print a shaidy repository*

Description

Print a shaidy repository

Usage

```
## S3 method for class 'shaidyRepo'
print(x, ...)
```

Arguments

x	The shaidy repository to print
...	Unused extra parameters

Value

The shaidy repository

shaid-class *Class representing the shaid of an object*

Description

Class representing the shaid of an object

shaidyAddFileBlob *Add data file(s) and properties to a local shaidy repository*

Description

Add data file(s) and properties to a local shaidy repository

Usage

```
shaidyAddFileBlob(
  shaidyRepo,
  blob.type,
  blob.file,
  filename,
  properties = NULL,
  shaid = NULL
)
```

Arguments

shaidyRepo	The shaidy repository
blob.type	The blob.type of the data file
blob.file	Name of the file(s) within the blob
filename	The filesystem path(s) to the file(s) to insert
properties	A list of additional properties to save with the file(s)
shaid	Shaid to store the blob as.

Value

The file's shaid

shaidyBlobExists *Determine if one more blobs exist in a shaidy repository*

Description

Determine if one more blobs exist in a shaidy repository

Usage

```
shaidyBlobExists(repo, shaids)
```

Arguments

`repo`	The shaidy repository
`shaids`	A shaid or list of shaids

Value

a boolean vector

`shaidyCopyBlob` *Copy a blob from one repository to another*

Description

Copy a blob from one repository to another

Usage

```
shaidyCopyBlob(src, shaid, dst)
```

Arguments

`src`	The source repository
`shaid`	The shaid of the blob to copy
`dst`	The destination repository

Value

the shaid

`shaidyCreateProtoBlob` *Create a prototype blob in a shaidy repository*

Description

Create a prototype blob in a shaidy repository

Usage

```
shaidyCreateProtoBlob(shaidyRepo, blob.type)
```

Arguments

`shaidyRepo`	The shaidy repository
`blob.type`	The blob.type of the prototype blob

Value

The file path of the prototype blob

`shaidyFinalizeProtoBlob` *Finalize a prototype blob*

Description

Finalize a prototype blob

Usage

```
shaidyFinalizeProtoBlob(shaidyRepo, shaid, protoblob)
```

Arguments

`shaidyRepo`	The shaidy repository
`shaid`	The shaid to assign the protoblob
`protoblob`	The prototype blob to finalize

Value

The shaid (invisibly)

The protoblob must have been created in the specified shaidy repository and with the same blob type as the shaid. When this function returns the protoblob will no longer be accessible . If a blob with the same shaid already exists in this repository, the protoblob is quitely removed without affecting the existing blob.

`shaidyFindRepo` *Find the first repository, if any, that contains the requested shaid*

Description

Find the first repository, if any, that contains the requested shaid

Usage

```
shaidyFindRepo(repos, shaid)
```

Arguments

`repos`	The list of repositories to search
`shaid`	The shaid to search for

Value

The first repository containing the shaid, otherwise NULL

shaidyGetComponents *Get an object's component shaids*

Description

Get an object's component shaids

Usage

```
shaidyGetComponents(object)

## S4 method for signature 'ngchm'
shaidyGetComponents(object)

## S4 method for signature 'ngchmDataset'
shaidyGetComponents(object)

## S4 method for signature 'ngchmCovariate'
shaidyGetComponents(object)
```

Arguments

`object`	The object, such as a chm, dataset, etc., for which to get the component shaids

Value

A list of shaids.

shaidyGetShaid *Get shaid for an object*

Description

Get shaid for an object

Usage

```
shaidyGetShaid(object)

## S4 method for signature 'ngchm'
shaidyGetShaid(object)
```

Arguments

`object`	The object, such as a chm, dataset, etc., for which to get the shaid

Value

The shaid of the object.

`shaidyHashProtoBlob` *Compute the shaid to assign a protoblob*

Description

Compute the shaid to assign a protoblob

Usage

```
shaidyHashProtoBlob(blob.type, protoblob)
```

Arguments

`blob.type`	The blob.type of the prototype blob
`protoblob`	The prototype blob

Value

The shaid to assign the protoblob

`shaidyInitRepository` *Create and initialize Shaidy Repository*

Description

This function initializes a Shaidy repository in a specified directory with specified blob types.

Usage

```
shaidyInitRepository(shaidyDir, blob.types)
```

Arguments

`shaidyDir`	A single character string specifying the directory where the Shaidy repository will be initialized.
`blob.types`	A character vector specifying the blob types for the Shaidy repository.

Value

None. This function is used for its side effects of initializing the Shaidy repository.

`shaidyLoadProvenanceDB` *Load the provid -> shaid DB for a local shaidy repository*

Description

Load the provid -> shaid DB for a local shaidy repository

Usage

```
shaidyLoadProvenanceDB(shaidyDir)
```

Arguments

`shaidyDir`	Basepath to a local shaidy repository.

Value

A shaidyProvenanceDB

`shaidyLoadProvidDB` *Load the provid -> labels DB for a local shaidy repository.*

Description

Load the provid -> labels DB for a local shaidy repository.

Usage

```
shaidyLoadProvidDB(shaidyDir)
```

Arguments

`shaidyDir`	Basepath to a local shaidy repository.

Value

A shaidyProvidDB

`shaidyLoadRepository` *Load a shaidy repository*

Description

Load a shaidy repository

Usage

```
shaidyLoadRepository(accessMethod, shaidyDir)
```

Arguments

`accessMethod`	Method for accessing repository.
`shaidyDir`	Basepath to shaidy repository.

Value

A shaidyRepo

`shaidyNewCache` *Create in memory shaid cache*

Description

Create in memory shaid cache

Usage

```
shaidyNewCache(shaidyDir)
```

Arguments

`shaidyDir`	Basepath to a local shaidy repository.

Value

An in memory shaid cache

`shaidyProvenance` *Create a provid from a list of label values*

Description

Create a provid from a list of label values

Usage

```
shaidyProvenance(...)
```

Arguments

`...`	shaidyRepo followed by a list of name=value labels to store in the provid

Value

A string containing the provid for the list of label values.

`shaidyRepoAPI` *Get the methods for the repository API called api*

Description

Get the methods for the repository API called api

Usage

```
shaidyRepoAPI(api)
```

Arguments

`api`	The name of a repository API

Value

A list of repository methods

`treeCuts-class` *Helper class for setting row/col gap locations as tree cuts*

Description

This class is to facilitate specification of row/col gaps in `chmNew()`. Note: user-facing function use the term 'gap', while internal functions that interact with java programs in the NGCHM viewer project use the term 'cut'.

Slots

`numberOfCuts` Integer number of cuts

See Also

`chmNew()`

`chmTreeGaps()`

`verifyNumeric` *Helper function to verify if variable is numeric.*

Description

If not numeric, print error message and stop.

Usage

```
verifyNumeric(variableToCheck)
```

Arguments

`variableToCheck`
The variable to check for being numeric.

Value

TRUE

`$.shaidyRepo` *Provide a simpler method for accessing repo methods*

Description

Provide a simpler method for accessing repo methods

Usage

```
## S3 method for class 'shaidyRepo'
repo$method
```

Arguments

`repo`	The repository to obtain the method for
`method`	The name of the method to obtain

Value

A function that calls the method with the repository as its first parameter

Index