Package ‘debrowser’

December 8, 2023

Type Package

Title Interactive Differential Expresion Analysis Browser

Version 1.31.0

Date 2020-12-07

Author Alper Kucukural <`alper.kucukural@umassmed.edu`>,
Onur Yukselen <`onur.yukselen@umassmed.edu`>,
Manuel Garber <`manuel.garber@umassmed.edu`>

Maintainer Alper Kucukural <`alper.kucukural@umassmed.edu`>

Description Bioinformatics platform containing interactive plots and tables for differential gene and region expression studies. Allows visualizing expression data much more deeply in an interactive and faster way. By changing the parameters, users can easily discover different parts of the data that like never have been done before. Manually creating and looking these plots takes time. With DEBrowser users can prepare plots without writing any code. Differential expression, PCA and clustering analysis are made on site and the results are shown in various plots such as scatter, bar, box, volcano, ma plots and Heatmaps.

Depends R (>= 3.5.0),

License GPL-3 + file LICENSE

LazyData true

Imports shiny, jsonlite, shinyjs, shinydashboard, shinyBS, gplots, DT, ggplot2, RColorBrewer, annotate, AnnotationDbi, DESeq2, DOSE, igraph, grDevices, graphics, stats, utils, GenomicRanges, IRanges, S4Vectors, SummarizedExperiment, stringi, reshape2, org.Hs.eg.db, org.Mm.eg.db, limma, edgeR, clusterProfiler, methods, sva, RCurl, enrichplot, colourpicker, plotly, heatmaply, Harman, pathview, apeglm, ashr

RoxygenNote 7.2.3

Encoding UTF-8

Suggests testthat, rmarkdown, knitr

VignetteBuilder knitr, rmarkdown

URL https://github.com/UMMS-Biocore/debrowser

BugReports https://github.com/UMMS-Biocore/debrowser/issues/new

biocViews Sequencing, ChIPSeq, RNASeq, DifferentialExpression, GeneExpression, Clustering, ImmunoOncology

git_url https://git.bioconductor.org/packages/debrowser

git_branch devel

git_last_commit 58c2028

git_last_commit_date 2023-10-24

Repository Bioconductor 3.19

Date/Publication 2023-12-08

Table of contents:

`actionButtonDE` *Buttons including Action Buttons and Event Buttons*

Description

Creates an action button whose value is initially zero, and increments by one each time it is pressed.

Usage

```
actionButtonDE(
  inputId,
  label,
  styleclass = "",
  size = "",
  block = FALSE,
  icon = NULL,
  css.class = "",
  ...
)
```

Arguments

`inputId`	Specifies the input slot that will be used to access the value.
`label`	The contents of the button–usually a text label, but you could also use any other HTML, like an image.
`styleclass`	The Bootstrap styling class of the button–options are primary, info, success, warning, danger, inverse, link or blank
`size`	The size of the button–options are large, small, mini
`block`	Whehter the button should fill the block

`icon`	Display an icon for the button
`css.class`	Any additional CSS class one wishes to add to the action button
`...`	Other argument to feed into shiny::actionButton

Examples

```
actionButtonDE("goDE", "Go to DE Analysis")
```

`addDataCols` *addDataCols*

Description

add aditional data columns to de results

Usage

```
addDataCols(data = NULL, de_res = NULL, cols = NULL, conds = NULL)
```

Arguments

`data`	loaded dataset
`de_res`	de results
`cols`	columns
`conds`	inputconds

Value

data

Examples

```
x <- addDataCols()
```

`addID` *addID*

Description

Adds an id to the data frame being used.

Usage

```
addID(data = NULL)
```

Arguments

`data`	loaded dataset

Value

data

Examples

```
x <- addID()
```

`all2all` *all2all*

Description

Prepares all2all scatter plots for given datasets.

Usage

```
all2all(data, cex = 2)
```

Arguments

`data`	data that have the sample names in the header.
`cex`	text size

Value

all2all scatter plots

Examples

```
plot<-all2all(mtcars)
```

`all2allControlsUI` *all2allControlsUI*

Description

Generates the controls in the left menu for an all2all plot

Usage

```
all2allControlsUI(id)
```

Arguments

`id`	namespace id

Value

returns the controls for left menu

Note

`all2allControlsUI`

Examples

```
x <- all2allControlsUI("bar")
```

`applyFilters` *applyFilters*

Description

Applies filters based on user selected parameters to be displayed within the DEBrowser.

Usage

```
applyFilters(filt_data = NULL, cols = NULL, conds = NULL, input = NULL)
```

Arguments

`filt_data`	loaded dataset
`cols`	selected samples
`conds`	seleced conditions
`input`	input parameters

Value

data

Examples

```
x <- applyFilters()
```

`applyFiltersNew` *applyFiltersNew*

Description

Apply filters based on foldChange cutoff and padj value. This function adds a "Legend" column with "Up", "Down" or "NS" values for visualization.

Usage

```
applyFiltersNew(data = NULL, input = NULL)
```

Arguments

`data`	loaded dataset
`input`	input parameters

Value

data

Examples

```
x <- applyFiltersNew()
```

`applyFiltersToMergedComparison` *applyFiltersToMergedComparison*

Description

Gathers the merged comparison data to be used within the DEBrowser.

Usage

```
applyFiltersToMergedComparison(dc = NULL, nc = NULL, input = NULL)
```

Arguments

`dc`	all data
`nc`	the number of comparisons
`input`	input params

Value

data

Examples

```
x <- applyFiltersToMergedComparison()
```

`barMainPlotControlsUI` *barMainPlotControlsUI*

Description

Generates the controls in the left menu for a bar main plot

Usage

```
barMainPlotControlsUI(id)
```

Arguments

`id`	namespace id

Value

returns the controls for left menu

Note

`barMainPlotControlsUI`

Examples

```
x <- barMainPlotControlsUI("bar")
```

`batchEffectUI` *batchEffectUI Creates a panel to coorect batch effect*

Description

batchEffectUI Creates a panel to coorect batch effect

Usage

```
batchEffectUI(id)
```

Arguments

`id`	namespace id

Value

panel

Examples

```
x <- batchEffectUI("batcheffect")
```

`batchMethod` *batchMethod*

Description

select batch effect method

Usage

```
batchMethod(id)
```

Arguments

`id`	namespace id

Value

radio control

Note

`batchMethod`

Examples

```
x <- batchMethod("batch")
```

`BoxMainPlotControlsUI` *BoxMainPlotControlsUI*

Description

Generates the controls in the left menu for a Box main plot

Usage

```
BoxMainPlotControlsUI(id)
```

Arguments

`id`	namespace id

Value

returns the controls for left menu

Note

`BoxMainPlotControlsUI`

Examples

```
x <- BoxMainPlotControlsUI("box")
```

`changeClusterOrder` *changeClusterOrder*

Description

change order of K-means clusters

Usage

```
changeClusterOrder(order = NULL, cld = NULL)
```

Arguments

`order`	order
`cld`	data

Value

heatmap plot area

Note

`changeClusterOrder`

Examples

```
x <- changeClusterOrder()
```

`checkCountData` *checkCountData*

Description

Returns if there is a problem in the count data.

Usage

```
checkCountData(input = NULL)
```

Arguments

`input`	inputs

Value

error if there is a problem about the loaded data

Note

`checkCountData`

Examples

```
x <- checkCountData()
```

`checkMetaData` *checkMetaData*

Description

Returns if there is a problem in the count data.

Usage

```
checkMetaData(input = NULL, counttable = NULL)
```

Arguments

`input`	input
`counttable`	counttable

Value

error if there is a problem about the loaded data

Note

`checkMetaData`

Examples

```
    x <- checkMetaData()
```

`clusterData` *clusterData*

Description

Gathers the Cluster analysis data to be used within the GO Term plots.

Usage

```
clusterData(dat = NULL)
```

Arguments

`dat`	the data to cluster

Value

clustered data

Note

`clusterData`

Examples

```
mycluster <- clusterData()
```

`clustFunParamsUI` *clustFunParamsUI*

Description

get cluster function parameter control

Usage

```
clustFunParamsUI()
```

Value

cluster params

Note

`clustFunParamsUI`

Examples

```
x <- clustFunParamsUI()
```

`compareClust` *compareClust*

Description

Compares the clustered data to be displayed within the GO Term plots.

Usage

```
compareClust(
  dat = NULL,
  ont = "CC",
  org = "org.Hs.eg.db",
  fun = "enrichGO",
  title = "Ontology Distribution Comparison",
  pvalueCutoff = 0.01
)
```

Arguments

`dat`	data to compare clusters
`ont`	the ontology to use
`org`	the organism used
`fun`	fun
`title`	title of the comparison
`pvalueCutoff`	pvalueCutoff

Value

compared cluster

Note

`compareClust`

Examples

```
x <- compareClust()
```

`condSelectUI` *condSelectUI Creates a panel to select samples for each condition*

Description

condSelectUI Creates a panel to select samples for each condition

Usage

```
condSelectUI()
```

Value

panel

Examples

```
x <- condSelectUI()
```

`correctCombat` *Correct Batch Effect using Combat in sva package*

Description

Batch effect correction

Usage

```
correctCombat(input = NULL, idata = NULL, metadata = NULL, method = NULL)
```

Arguments

`input`	input values
`idata`	data
`metadata`	metadata
`method`	method: either Combat or CombatSeq

Value

data

Examples

```
x<-correctCombat ()
```

`correctHarman` *Correct Batch Effect using Harman*

Description

Batch effect correction

Usage

```
correctHarman(input = NULL, idata = NULL, metadata = NULL)
```

Arguments

`input`	input values
`idata`	data
`metadata`	metadata

Value

data

Examples

```
x<-correctHarman ()
```

`customColorsUI` *customColorsUI*

Description

get Custom Color controls

Usage

```
customColorsUI(id)
```

Arguments

`id`	namespace ID

Value

color range

Note

`getColRng`

Examples

```
x <- customColorsUI("heatmap")
```

`cutOffSelectionUI` *cutOffSelectionUI*

Description

Gathers the cut off selection for DE analysis

Usage

```
cutOffSelectionUI(id)
```

Arguments

`id`	namespace id

Value

returns the left menu according to the selected tab;

Note

`cutOffSelectionUI`

Examples

```
x <- cutOffSelectionUI("cutoff")
```

`dataLCFUI` *dataLCFUI Creates a panel to filter low count genes and regions*

Description

dataLCFUI Creates a panel to filter low count genes and regions

Usage

```
dataLCFUI(id)
```

Arguments

`id`	namespace id

Value

panel

Examples

```
x <- dataLCFUI("lcf")
```

`dataLoadUI` *dataLoadUI*

Description

Creates a panel to upload the data

Usage

```
dataLoadUI(id)
```

Arguments

`id`	namespace id

Value

panel

Examples

```
x <- dataLoadUI("load")
```

`debrowserall2all` *debrowserall2all*

Description

Module for a bar plot that can be used in data prep, main plots low count removal modules or any desired module

Usage

```
debrowserall2all(input, output, session, data = NULL, cex = 2)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values
`cex`	the size of the dots

Value

all2all plot

Examples

```
x <- debrowserall2all()
```

`debrowserbarmainplot` *debrowserbarmainplot*

Description

Module for a bar plot that can be used in data prep, main plots low count removal modules or any desired module

Usage

```
debrowserbarmainplot(
  input,
  output,
  session,
  data = NULL,
  cols = NULL,
  conds = NULL,
  cond_names = NULL,
  key = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values
`cols`	columns
`conds`	conditions
`cond_names`	condition names
`key`	the gene or region name

Value

density plot

Examples

```
x <- debrowserbarmainplot()
```

`debrowserbatcheffect` *debrowserbatcheffect*

Description

Module to correct batch effect

Usage

```
debrowserbatcheffect(input, output, session, ldata = NULL)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`ldata`	loaded data

Value

main plot

panel

Examples

```
x <- debrowserbatcheffect()
```

`debrowserboxmainplot` *debrowserboxmainplot*

Description

Module for a box plot that can be used in DEanalysis main part and used heatmaps

Usage

```
debrowserboxmainplot(
  input = NULL,
  output = NULL,
  session = NULL,
  data = NULL,
  cols = NULL,
  conds = NULL,
  cond_names = NULL,
  key = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values
`cols`	columns
`conds`	conditions
`cond_names`	condition names
`key`	the gene or region name

Value

density plot

Examples

```
x <- debrowserboxmainplot()
```

`debrowsercondselect` *debrowsercondselect*

Description

Condition selection This is not a module. Module construction didn't used here, just use it as functions not in a module.

Usage

```
debrowsercondselect(
  input = NULL,
  output = NULL,
  session = NULL,
  data = NULL,
  metadata = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	count data
`metadata`	metadata

Value

main plot

panel

Examples

```
x <- debrowsercondselect()
```

`debrowserdataload` *debrowserdataload*

Description

Module to load count data and metadata

Usage

```
debrowserdataload(
  input = NULL,
  output = NULL,
  session = NULL,
  nextpagebutton = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`nextpagebutton`	the name of the next page button after loading the data

Value

main plot

panel

Examples

```
x <- debrowserdataload()
```

`debrowserdeanalysis` *debrowserdeanalysis*

Description

Module to perform and visualize DE results.

Usage

```
debrowserdeanalysis(
  input = NULL,
  output = NULL,
  session = NULL,
  data = NULL,
  metadata = NULL,
  columns = NULL,
  conds = NULL,
  params = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values
`metadata`	metadata
`columns`	columns
`conds`	conditions
`params`	de parameters

Value

DE panel

Examples

```
x <- debrowserdeanalysis()
```

`debrowserdensityplot` *debrowserdensityplot*

Description

Module for a density plot that can be used in data prep and low count removal modules

Usage

```
debrowserdensityplot(input = NULL, output = NULL, session = NULL, data = NULL)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values

Value

density plot

Examples

```
x <- debrowserdensityplot()
```

`debrowserheatmap` *debrowserheatmap*

Description

Heatmap module to create interactive heatmaps and get selected list from a heatmap

Usage

```
debrowserheatmap(input, output, session, expdata = NULL)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`expdata`	a matrix that includes expression values

Value

heatmapply plot

Examples

```
x <- debrowserheatmap()
```

`debrowserhistogram` *debrowserhistogram*

Description

Module for a histogram that can be used in data prep and low count removal modules

Usage

```
debrowserhistogram(input = NULL, output = NULL, session = NULL, data = NULL)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values

Value

histogram

Examples

```
x <- debrowserhistogram()
```

`debrowserIQRplot` *debrowserIQRplot*

Description

Module for an IQR plot that can be used in data prep and low count removal modules

Usage

```
debrowserIQRplot(input = NULL, output = NULL, session = NULL, data = NULL)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values

Value

IQR

Examples

```
x <- debrowserIQRplot()
```

`debrowserlowcountfilter` *debrowserlowcountfilter*

Description

Module to filter low count genes/regions

Usage

```
debrowserlowcountfilter(
  input = NULL,
  output = NULL,
  session = NULL,
  ldata = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`ldata`	loaded data

Value

main plot

panel

Examples

```
x <- debrowserlowcountfilter()
```

`debrowsermainplot` *debrowsermainplot*

Description

Module for a scatter, volcano and ma plots that are going to be used as a mainplot in debrowser

Usage

```
debrowsermainplot(
  input = NULL,
  output = NULL,
  session = NULL,
  data = NULL,
  cond_names = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`data`	a matrix that includes expression values
`cond_names`	condition names

Value

main plot

panel

Examples

```
x <- debrowsermainplot()
```

`debrowserpcaplot` *debrowserpcaplot*

Description

Module for a pca plot with its loadings as a mainplot in debrowser

Usage

```
debrowserpcaplot(
  input = NULL,
  output = NULL,
  session = NULL,
  pcadata = NULL,
  metadata = NULL
)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session
`pcadata`	a matrix that includes expression values
`metadata`	metadata to color the plots

Value

main plot

panel

Examples

```
x <- debrowserpcaplot()
```

`dendControlsUI` *dendControlsUI*

Description

get distance metric parameters

Usage

```
dendControlsUI(id, dendtype = "Row")
```

Arguments

`id`	module ID
`dendtype`	Row or Col

Value

controls

Note

`dendControlsUI`

Examples

```
x <- dendControlsUI("heatmap")
```

`densityPlotControlsUI` *densityPlotControlsUI*

Description

Generates the controls in the left menu for a densityPlot

Usage

```
densityPlotControlsUI(id)
```

Arguments

`id`	namespace id

Value

returns the left menu

Note

`densityPlotControlsUI`

Examples

```
x <- densityPlotControlsUI("density")
```

`deServer` *deServer*

Description

Sets up shinyServer to be able to run DEBrowser interactively.

Usage

```
deServer(input, output, session)
```

Arguments

`input`	input params from UI
`output`	output params to UI
`session`	session variable

Value

the panel for main plots;

Note

`deServer`

Examples

```
deServer
```

`deUI` *deUI*

Description

Creates a shinyUI to be able to run DEBrowser interactively.

Usage

```
deUI()
```

Value

the panel for main plots;

Note

`deUI`

Examples

```
x<-deUI()
```

`distFunParamsUI` *distFunParamsUI*

Description

get distance metric parameters

Usage

```
distFunParamsUI()
```

Value

funParams

Note

`distFunParamsUI`

Examples

```
x <- distFunParamsUI()
```

`drawKEGG` *drawKEGG*

Description

draw KEGG patwhay with expression values

Usage

```
drawKEGG(input = NULL, dat = NULL, pid = NULL)
```

Arguments

`input`	input
`dat`	expression matrix
`pid`	pathway id

Value

enriched DO

Note

`drawKEGG`

Examples

```
x <- drawKEGG()
```

`drawPCAExplained` *Creates a more detailed plot using the PCA results from the selected dataset.*

Description

Creates a more detailed plot using the PCA results from the selected dataset.

Usage

```
drawPCAExplained(explainedData = NULL)
```

Arguments

`explainedData`	selected data

Value

explained plot

Examples

```
x <- drawPCAExplained()
```

`fileTypes` *fileTypes*

Description

Returns fileTypes that are going to be used in creating fileUpload UI

Usage

```
fileTypes()
```

Value

file types

Note

`fileTypes`

Examples

```
x <- fileTypes()
```

`fileUploadBox` *fileUploadBox*

Description

File upload module

Usage

```
fileUploadBox(id = NULL, inputId = NULL, label = NULL)
```

Arguments

`id`	namespace id
`inputId`	input file ID
`label`	label

Value

radio control

Note

`fileUploadBox`

Examples

```
x <- fileUploadBox("meta", "metadata", "Metadata")
```

`generateTestData` *generateTestData*

Description

This generates a test data that is suitable to main plots in debrowser

Usage

```
generateTestData(dat = NULL)
```

Arguments

`dat`	DESeq results will be generated for loaded data

Value

testData

Examples

```
x <- generateTestData()
```

`getAfterLoadMsg` *getAfterLoadMsg*

Description

Generates and displays the message to be shown after loading data within the DEBrowser.

Usage

```
getAfterLoadMsg()
```

Value

return After Load Msg

Note

`getAfterLoadMsg`

Examples

```
    x <- getAfterLoadMsg()
```

`getAll2AllPlotUI` *getAll2AllPlotUI*

Description

all2all plots UI.

Usage

```
getAll2AllPlotUI(id)
```

Arguments

`id`	namespace id

Value

the panel for all2all plots;

Note

`getAll2AllPlotUI`

Examples

```
x <- getAll2AllPlotUI("bar")
```

`getBarMainPlot` *getBarMainPlot*

Description

Makes Density plots

Usage

```
getBarMainPlot(
  data = NULL,
  cols = NULL,
  conds = NULL,
  cond_names = NULL,
  key = NULL,
  title = "",
  input = NULL
)
```

Arguments

`data`	count or normalized data
`cols`	cols
`conds`	conds
`cond_names`	condition names
`key`	key
`title`	title
`input`	input

Examples

```
getBarMainPlot()
```

`getBarMainPlotUI` *getBarMainPlotUI*

Description

main bar plots UI.

Usage

```
getBarMainPlotUI(id)
```

Arguments

`id`	namespace id

Value

the panel for Density plots;

Note

`getBarMainPlotUI`

Examples

```
x <- getBarMainPlotUI("bar")
```

`getBoxMainPlot` *getBoxMainPlot*

Description

Makes Density plots

Usage

```
getBoxMainPlot(
  data = NULL,
  cols = NULL,
  conds = NULL,
  cond_names = NULL,
  key = NULL,
  title = "",
  input = NULL
)
```

Arguments

`data`	count or normalized data
`cols`	cols
`conds`	conds
`cond_names`	condition names
`key`	key
`title`	title
`input`	input

Examples

```
getBoxMainPlot()
```

`getBoxMainPlotUI` *getBoxMainPlotUI*

Description

main Box plots UI.

Usage

```
getBoxMainPlotUI(id)
```

Arguments

`id`	namespace id

Value

the panel for Density plots;

Note

`getBoxMainPlotUI`

Examples

```
x <- getBoxMainPlotUI("box")
```

`getBSTableUI` *getBSTableUI prepares a Modal to put a table*

Description

getBSTableUI prepares a Modal to put a table

Usage

```
getBSTableUI(
  name = NULL,
  label = NULL,
  trigger = NULL,
  size = "large",
  modal = NULL
)
```

Arguments

`name`	name
`label`	label
`trigger`	trigger button for the modal
`size`	size of the modal
`modal`	modal yes/no

Value

the modal

Examples

```
x<- getBSTableUI()
```

`getColors` *getColors*

Description

get colors for the domains

Usage

```
getColors(domains = NULL)
```

Arguments

`domains`	domains to be colored

Value

colors

Examples

```
x<-getColors()
```

`getColorShapeSelection` *getColorShapeSelection*

Description

Generates the fill and shape selection boxes for PCA plots. metadata file has to be loaded in this case

Usage

```
getColorShapeSelection(metadata = NULL, input = NULL, session = NULL)
```

Arguments

`metadata`	metadata table
`input`	input
`session`	session

Value

Color and shape selection boxes

Examples

```
x <- getColorShapeSelection()
```

`getCompSelection` *getCompSelection*

Description

Gathers the user selected comparison set to be used within the DEBrowser.

Usage

```
getCompSelection(name = NULL, count = NULL)
```

Arguments

`name`	the name of the selectInput
`count`	comparison count

Note

`getCompSelection`

Examples

```
x <- getCompSelection(name="comp", count = 2)
```

`getConditionSelector` *getConditionSelector*

Description

Selects user input conditions to run in DESeq.

Usage

```
getConditionSelector(num = NULL, choices = NULL, selected = NULL)
```

Arguments

`num`	panel that is going to be shown
`choices`	sample list
`selected`	selected smaple list

Examples

```
x <- getConditionSelector()
```

`getConditionSelectorFromMeta` *getConditionSelectorFromMeta*

Description

Selects user input conditions to run in DESeq from metadata

Usage

```
getConditionSelectorFromMeta(
  metadata = NULL,
  input = NULL,
  index = 1,
  num = 0,
  choices = NULL,
  selected = NULL
)
```

Arguments

`metadata`	meta data table
`input`	input
`index`	index
`num`	num
`choices`	choices
`selected`	selected

Examples

```
x <- getConditionSelectorFromMeta()
```

`getCondMsg` *getCondMsg*

Description

Generates and displays the current conditions and their samples within the DEBrowser.

Usage

```
getCondMsg(dc = NULL, input = NULL, cols = NULL, conds = NULL)
```

Arguments

`dc`	columns
`input`	selected comparison
`cols`	columns
`conds`	selected conditions

Value

return conditions

Note

`getCondMsg`

Examples

```
x <- getCondMsg()
```

`getCovariateDetails` *getCovariateDetails*

Description

get the covariate detail box after DE method selected

Usage

```
getCovariateDetails(num = NULL, input = NULL, metadata = NULL)
```

Arguments

`num`	panel that is going to be shown
`input`	user input
`metadata`	metadata

Examples

```
x <- getCovariateDetails()
```

`getCutOffSelection` *getCutOffSelection*

Description

Gathers the cut off selection for DE analysis

Usage

```
getCutOffSelection(nc = 1)
```

Arguments

`nc`	total number of comparisons

Value

returns the left menu according to the selected tab;

Note

`getCutOffSelection`

Examples

```
x <- getCutOffSelection()
```

`getDataAssesmentText` *getDataAssesmentText DataAssesment text*

Description

getDataAssesmentText DataAssesment text

Usage

```
getDataAssesmentText()
```

Value

help text for data assesment

Examples

```
x<- getDataAssesmentText()
```

`getDataForTables` *getDataForTables get data to fill up tables tab*

Description

getDataForTables get data to fill up tables tab

Usage

```
getDataForTables(
  input = NULL,
  init_data = NULL,
  filt_data = NULL,
  selected = NULL,
  getMostVaried = NULL,
  mergedComp = NULL,
  explainedData = NULL
)
```

Arguments

`input`	input parameters
`init_data`	initial dataset
`filt_data`	filt_data
`selected`	selected genes
`getMostVaried`	most varied genes
`mergedComp`	merged comparison set
`explainedData`	pca gene set

Value

data

Examples

```
    x <- getDataForTables()
```

`getDataPreparationText` *getDataPreparationText DataPreparation text*

Description

getDataPreparationText DataPreparation text

Usage

```
getDataPreparationText()
```

Value

help text for data preparation

Examples

```
x<- getDataPreparationText()
```

`getDEAnalysisText` *getDEAnalysisText DEAnalysis text*

Description

getDEAnalysisText DEAnalysis text

Usage

```
getDEAnalysisText()
```

Value

help text for DE Analysis

Examples

```
x<- getDEAnalysisText()
```

`getDensityPlot` *getDensityPlot*

Description

Makes Density plots

Usage

```
getDensityPlot(data = NULL, input = NULL, title = "")
```

Arguments

`data`	count or normalized data
`input`	input
`title`	title

Examples

```
getDensityPlot()
```

`getDensityPlotUI` *getDensityPlotUI*

Description

Density plot UI.

Usage

```
getDensityPlotUI(id)
```

Arguments

`id`	namespace id

Value

the panel for Density plots;

Note

`getDensityPlotUI`

Examples

```
x <- getDensityPlotUI("density")
```

`getDEResultsUI` *getDEResultsUI Creates a panel to visualize DE results*

Description

getDEResultsUI Creates a panel to visualize DE results

Usage

```
getDEResultsUI(id)
```

Arguments

`id`	namespace id

Value

panel

Examples

```
x <- getDEResultsUI("batcheffect")
```

`getDomains` *getDomains*

Description

Get domains for the main plots.

Usage

```
getDomains(filt_data = NULL)
```

Arguments

`filt_data`	data to get the domains

Value

domains

Examples

```
x<-getDomains()
```

`getDown` *getDown get down regulated data*

Description

getDown get down regulated data

Usage

```
getDown(filt_data = NULL)
```

Arguments

`filt_data`	filt_data

Value

data

Examples

```
x <- getDown()
```

`getDownloadSection` *getDownloadSection*

Description

download section button and dataset selection box in the menu for user to download selected data.

Usage

```
getDownloadSection(choices = NULL)
```

Arguments

`choices`	main vs. QC section

Value

the panel for download section in the menu;

Note

`getDownloadSection`

Examples

```
x<- getDownloadSection()
```

`getEnrichDO` *getEnrichDO*

Description

Gathers the Enriched DO Term analysis data to be used within the GO Term plots.

Usage

```
getEnrichDO(genelist = NULL, pvalueCutoff = 0.01)
```

Arguments

`genelist`	gene list
`pvalueCutoff`	the p value cutoff

Value

enriched DO

Note

`getEnrichDO`

Examples

```
x <- getEnrichDO()
```

`getEnrichGO` *getEnrichGO*

Description

Gathers the Enriched GO Term analysis data to be used within the GO Term plots.

Usage

```
getEnrichGO(
  genelist = NULL,
  pvalueCutoff = 0.01,
  org = "org.Hs.eg.db",
  ont = "CC"
)
```

Arguments

`genelist`	gene list
`pvalueCutoff`	p value cutoff
`org`	the organism used
`ont`	the ontology used

Value

Enriched GO

Note

`getEnrichGO`

Examples

```
x <- getEnrichGO()
```

`getEnrichKEGG` *getEnrichKEGG*

Description

Gathers the Enriched KEGG analysis data to be used within the GO Term plots.

Usage

```
getEnrichKEGG(genelist = NULL, pvalueCutoff = 0.01, org = "org.Hs.eg.db")
```

Arguments

`genelist`	gene list
`pvalueCutoff`	the p value cutoff
`org`	the organism used

Value

Enriched KEGG

Note

`getEnrichKEGG`

Examples

```
x <- getEnrichKEGG()
```

`getEntrezIds` *getEntrezIds*

Description

Gathers the gene list to use for GOTerm analysis.

Usage

```
getEntrezIds(genes = NULL, org = "org.Hs.eg.db")
```

Arguments

`genes`	gene list with fold changes
`org`	orgranism for gene symbol entrez ID conversion

Value

ENTREZ ID list

Note

`GOTerm`

`getEntrezIds` symobol to ENTREZ ID conversion

Examples

```
x <- getEntrezIds()
```

`getEntrezTable` *getEntrezTable*

Description

Gathers the entrezIds of the genes in given list and their data

Usage

```
getEntrezTable(genes = NULL, dat = NULL, org = "org.Hs.eg.db")
```

Arguments

`genes`	gene list
`dat`	data matrix
`org`	orgranism for gene symbol entrez ID conversion

Value

table with the entrez IDs in the rownames

Note

`GOTerm`

`getEntrezTable` symobol to ENTREZ ID conversion

Examples

```
x <- getEntrezTable()
```

`getGeneList` *getGeneList*

Description

Gathers the gene list to use for GOTerm analysis.

Usage

```
getGeneList(
  genes = NULL,
  org = "org.Hs.eg.db",
  fromType = "SYMBOL",
  toType = c("ENTREZID")
)
```

Arguments

`genes`	gene list
`org`	orgranism for gene symbol entrez ID conversion
`fromType`	from Type
`toType`	to Type

Value

ENTREZ ID list

Note

`GOTerm`

`getGeneList` symobol to ENTREZ ID conversion

Examples

```
x <- getGeneList(c('OCLN', 'ABCC2'))
```

`getGeneSetData` *getGeneSetData*

Description

Gathers the specified gene set list to be used within the DEBrowser.

Usage

```
getGeneSetData(data = NULL, geneset = NULL)
```

Arguments

`data`	loaded dataset
`geneset`	given gene set

Value

data

Examples

```
x <- getGeneSetData()
```

`getGOLeftMenu` *getGOLeftMenu*

Description

Generates the GO Left menu to be displayed within the DEBrowser.

Usage

```
getGOLeftMenu()
```

Value

returns the left menu according to the selected tab;

Note

`getGOLeftMenu`

Examples

```
x <- getGOLeftMenu()
```

`getGoPanel` *getGoPanel*

Description

Creates go term analysis panel within the shiny display.

Usage

```
getGoPanel()
```

Value

the panel for go term analysis;

Note

`getGoPanel`

Examples

```
x <- getGoPanel()
```

`getGOPlots` *getGOPlots*

Description

Go term analysis panel. Generates appropriate GO plot based on user selection.

Usage

```
getGOPlots(dataset = NULL, GSEARes = NULL, input = NULL)
```

Arguments

`dataset`	the dataset used
`GSEARes`	GSEA results
`input`	input params

Value

the panel for go plots;

Note

`getGOPlots`

Examples

```
x<- getGOPlots()
```

`getGroupSelector` *getGroupSelector Return the groups*

Description

getGroupSelector Return the groups

Usage

```
getGroupSelector(metadata = NULL, input = NULL, index = 1, num = 0)
```

Arguments

`metadata`	meta data table
`input`	input params
`index`	index
`num`	num

Value

meta select box

Examples

```
x<-getGroupSelector()
```

`getGSEA` *getGSEA*

Description

Gathers the Enriched KEGG analysis data to be used within the GO Term plots.

Usage

```
getGSEA(
  dataset = NULL,
  pvalueCutoff = 0.01,
  org = "org.Hs.eg.db",
  sortfield = "log2FoldChange"
)
```

Arguments

`dataset`	dataset
`pvalueCutoff`	the p value cutoff
`org`	the organism used
`sortfield`	sort field for GSEA

Value

GSEA

Note

`getGSEA`

Examples

```
    x <- getGSEA()
```

`getHeatmapUI` *getHeatmapUI*

Description

Generates the left menu to be used for heatmap plots

Usage

```
getHeatmapUI(id)
```

Arguments

`id`	module ID

Value

heatmap plot area

Note

`getHeatmapUI`

Examples

```
x <- getHeatmapUI("heatmap")
```

`getHelpButton` *getHelpButton prepares a helpbutton for to go to a specific site in the documentation*

Description

getHelpButton prepares a helpbutton for to go to a specific site in the documentation

Usage

```
getHelpButton(name = NULL, link = NULL)
```

Arguments

`name`	name that are going to come after info
`link`	link of the help

Value

the info button

Examples

```
x<- getHelpButton()
```

`getHideLegendOnOff` *getHideLegendOnOff*

Description

hide legend

Usage

```
getHideLegendOnOff(id = "pca")
```

Arguments

`id`	namespace id

Examples

```
x <- getHideLegendOnOff("pca")
```

`getHistogramUI` *getHistogramUI*

Description

Histogram plots UI.

Usage

```
getHistogramUI(id)
```

Arguments

`id`	namespace id

Value

the panel for PCA plots;

Note

`getHistogramUI`

Examples

```
x <- getHistogramUI("histogram")
```

`getIntroText` *getIntroText Intro text*

Description

getIntroText Intro text

Usage

```
getIntroText()
```

Value

the JS for tab updates

Examples

```
x<- getIntroText()
```

`getIQRPlot` *getIQRPlot*

Description

Makes IQR boxplot plot

Usage

```
getIQRPlot(data = NULL, input = NULL, title = "")
```

Arguments

`data`	count or normalized data
`input`	input
`title`	title

Examples

```
getIQRPlot()
```

`getIQRPlotUI` *getIQRPlotUI*

Description

IQR plot UI.

Usage

```
getIQRPlotUI(id)
```

Arguments

`id`	namespace id

Value

the panel for IQR plots;

Note

`getIQRPlotUI`

Examples

```
x <- getIQRPlotUI("IQR")
```

`getJSLine` *getJSLine*

Description

heatmap JS code for selection functionality

Usage

```
getJSLine()
```

Value

JS Code

Examples

```
x <- getJSLine()
```

`getKEGGModal` *getKEGGModal prepares a modal for KEGG plots*

Description

getKEGGModal prepares a modal for KEGG plots

getKEGGModal prepares a helpbutton for to go to a specific site in the documentation

Usage

```
getKEGGModal()

getKEGGModal()
```

Value

the info button

the info button

Examples

```
x<- getKEGGModal()

x<- getKEGGModal()
```

`getLeftMenu` *getLeftMenu*

Description

Generates the left menu for for plots within the DEBrowser.

Usage

```
getLeftMenu(input = NULL)
```

Arguments

`input`	input values

Value

returns the left menu according to the selected tab;

Note

`getLeftMenu`

Examples

```
x <- getLeftMenu()
```

`getLegendColors` *getLegendColors*

Description

Generates colors according to the data

Usage

```
getLegendColors(Legend = c("up", "down", "NS"))
```

Arguments

`Legend`	unique Legends

Value

mainPlotControls

Note

`getLegendColors`

Examples

```
x <- getLegendColors(c("up", "down", "GS", "NS"))
```

`getLegendRadio` *getLegendRadio*

Description

Radio buttons for the types in the legend

Usage

```
getLegendRadio(id)
```

Arguments

`id`	namespace id

Value

radio control

Note

`getLegendRadio`

Examples

```
x <- getLegendRadio("deprog")
```

`getLegendSelect` *getLegendSelect*

Description

select legend

Usage

```
getLegendSelect(id = "pca")
```

Arguments

`id`	namespace id

Note

`getLegendSelect`

Examples

```
x <- getLegendSelect("pca")
```

`getLevelOrder` *getLevelOrder*

Description

Generates the order of the overlapping points

Usage

```
getLevelOrder(Level = c("up", "down", "NS"))
```

Arguments

`Level`	factor levels shown in the legend

Value

order

Note

`getLevelOrder`

Examples

```
x <- getLevelOrder(c("up", "down", "GS", "NS"))
```

`getLoadingMsg` *getLoadingMsg*

Description

Creates and displays the loading message/gif to be displayed within the DEBrowser.

Usage

```
getLoadingMsg(output = NULL)
```

Arguments

`output`	output message

Value

loading msg

Note

`getLoadingMsg`

Examples

```
x <- getLoadingMsg()
```

`getLogo` *getLogo*

Description

Generates and displays the logo to be shown within DEBrowser.

Usage

```
getLogo()
```

Value

return logo

Note

`getLogo`

Examples

```
x <- getLogo()
```

`getMainPanel` *getMainPanel*

Description

main panel for volcano, scatter and maplot. Barplot and box plots are in this page as well.

Usage

```
getMainPanel()
```

Value

the panel for main plots;

Note

`getMainPanel`

Examples

```
x <- getMainPanel()
```

`getMainPlotsLeftMenu` *getMainPlotsLeftMenu*

Description

Generates the Main PLots Left menu to be displayed within the DEBrowser.

Usage

```
getMainPlotsLeftMenu()
```

Value

returns the left menu according to the selected tab;

Note

`getMainPlotsLeftMenu`

Examples

```
x <- getMainPlotsLeftMenu()
```

`getMainPlotUI` *getMainPlotUI*

Description

main plot for volcano, scatter and maplot.

Usage

```
getMainPlotUI(id)
```

Arguments

`id`	namespace id

Value

the panel for main plots;

Note

`getMainPlotUI`

Examples

```
x <- getMainPlotUI("main")
```

`getMean` *getMean*

Description

Gathers the mean for selected condition.

Usage

```
getMean(data = NULL, selcols = NULL)
```

Arguments

`data`	dataset
`selcols`	input cols

Value

data

Examples

```
x <- getMean()
```

`getMergedComparison` *getMergedComparison*

Description

Gathers the merged comparison data to be used within the DEBrowser.

Usage

```
getMergedComparison(dc = NULL, nc = NULL, input = NULL)
```

Arguments

`dc`	data container
`nc`	the number of comparisons
`input`	input params

Value

data

Examples

```
x <- getMergedComparison()
```

`getMetaSelector` *getMetaSelector*

Description

Return the sample selection box using meta data table

Usage

```
getMetaSelector(metadata = NULL, input = NULL, n = 0)
```

Arguments

`metadata`	meta data table
`input`	input params
`n`	the box number

Value

meta select box

Examples

```
x<-getMetaSelector()
```

`getMethodDetails` *get the detail boxes after DE method selected*

Description

get the detail boxes after DE method selected

Usage

```
getMethodDetails(num = NULL, input = NULL)
```

Arguments

`num`	panel that is going to be shown
`input`	user input

Examples

```
x <- getMethodDetails()
```

`getMostVariedList` *getMostVariedList*

Description

Calculates the most varied genes to be used for specific plots within the DEBrowser.

Usage

```
getMostVariedList(datavar = NULL, cols = NULL, input = NULL)
```

Arguments

`datavar`	loaded dataset
`cols`	selected columns
`input`	input

Value

data

Examples

```
x <- getMostVariedList()
```

`getNormalizedMatrix` *getNormalizedMatrix*

Description

Normalizes the matrix passed to be used within various methods within DEBrowser. Requires edgeR package

Usage

```
getNormalizedMatrix(M = NULL, method = "TMM")
```

Arguments

`M`	numeric matrix
`method`	normalization method for edgeR. default is TMM

Value

normalized matrix

Note

`getNormalizedMatrix`

Examples

```
x <- getNormalizedMatrix(mtcars)
```

`getOrganism` *getOrganism*

Description

getOrganism

Usage

```
getOrganism(org)
```

Arguments

`org`	organism

Value

organism name for keg

Note

`getOrganism`

Examples

```
x <- getOrganism()
```

`getOrganismBox` *getOrganismBox*

Description

Get the organism Box.

Usage

```
getOrganismBox()
```

Value

selectInput

Note

`getOrganismBox`

`getOrganismBox` makes the organism box

Examples

```
x <- getOrganismBox()
```

`getOrganismPathway` *getOrganismPathway*

Description

getOrganismPathway

Usage

```
getOrganismPathway(org)
```

Arguments

`org`	organism

Value

organism name for pathway

Note

`getOrganismPathway`

Examples

```
x <- getOrganismPathway()
```

`getPCAcontolUpdatesJS` *getPCAcontolUpdatesJS in the prep menu we have two PCA plots to show how batch effect correction worked. One set of PCA input controls updates two PCA plots with this JS.*

Description

getPCAcontolUpdatesJS in the prep menu we have two PCA plots to show how batch effect correction worked. One set of PCA input controls updates two PCA plots with this JS.

Usage

```
getPCAcontolUpdatesJS()
```

Value

the JS for tab updates

Examples

```
x<- getTabUpdateJS()
```

`getPCAexplained` *getPCAexplained*

Description

Creates a more detailed plot using the PCA results from the selected dataset.

Usage

```
getPCAexplained(datasetInput = NULL, pca_data = NULL, input = NULL)
```

Arguments

`datasetInput`	selected data
`pca_data`	from user
`input`	input params

Value

explained plot

Examples

```
load(system.file("extdata", "demo", "demodata.Rda", package="debrowser"))
input<-c()
input$qcplot<-"pca"
input$col_list<-colnames(demodata[,1:6])
dat <- getNormalizedMatrix(demodata[,1:6])
pca_data <- run_pca(dat)
x <- getPCAexplained(dat, pca_data, input)
```

`getPCAPlotUI` *getPCAPlotUI*

Description

PCA plots UI.

Usage

```
getPCAPlotUI(id)
```

Arguments

`id`	namespace id

Value

the panel for PCA plots;

Note

`getPCAPlotUI`

Examples

```
x <- getPCAPlotUI("pca")
```

`getPCselection` *getPCselection*

Description

Generates the PC selection number to be used within DEBrowser.

Usage

```
getPCselection(id, num = 1, xy = "x")
```

Arguments

`id`	namespace id
`num`	PC selection number
`xy`	x or y coordinate

Value

PC selection for PCA analysis

Note

`getPCselection`

Examples

```
x <- getPCselection("pca")
```

`getPlotArea` *getPlotArea*

Description

returns plot area either for heatmaply or heatmap.2

Usage

```
getPlotArea(input = NULL, session = NULL)
```

Arguments

`input`	input variables
`session`	session

Value

heatmapply/heatmap.2 plot area

Examples

```
x <- getPlotArea()
```

`getProgramTitle` *getProgramTitle*

Description

Generates the title of the program to be displayed within DEBrowser. If it is called in a program, the program title will be hidden

Usage

```
getProgramTitle(session = NULL)
```

Arguments

`session`	session var

Value

program title

Note

`getProgramTitle`

Examples

```
title<-getProgramTitle()
```

`getQAText` *getQAText Some questions and answers*

Description

getQAText Some questions and answers

Usage

```
getQAText()
```

Value

help text for QA

Examples

```
x<- getQAText()
```

`getQCLeftMenu` *getQCLeftMenu*

Description

Generates the left menu to be used for QC plots within the DEBrowser.

Usage

```
getQCLeftMenu(input = NULL)
```

Arguments

`input`	input values

Value

QC left menu

Note

`getQCLeftMenu`

Examples

```
x <- getQCLeftMenu()
```

`getQCPanel` *getQCPanel*

Description

Gathers the conditional panel for QC plots

Usage

```
getQCPanel(input = NULL)
```

Arguments

`input`	user input

Value

the panel for QC plots

Note

`getQCSection`

Examples

```
x <- getQCPanel()
```

`getSampleDetails` *getSampleDetails*

Description

get sample details

Usage

```
getSampleDetails(output = NULL, summary = NULL, details = NULL, data = NULL)
```

Arguments

`output`	output
`summary`	summary output name
`details`	details ouput name
`data`	data

Value

panel

Examples

```
x <- getSampleDetails()
```

`getSampleNames` *getSampleNames*

Description

Prepares initial samples to fill condition boxes. it reads the sample names from the data and splits into two.

Usage

```
getSampleNames(cnames = NULL, part = 1)
```

Arguments

`cnames`	sample names in the header of a dataset
`part`	c(1,2). 1=first half and 2= second half

Value

sample names.

Examples

```
x<-getSampleNames()
```

`getSearchData` *getSearchData*

Description

search the geneset in the tables and return it

Usage

```
getSearchData(dat = NULL, input = NULL)
```

Arguments

`dat`	table data
`input`	input params

Value

data

Examples

```
x <- getSearchData()
```

`getSelectedCols` *getSelectedCols*

Description

gets selected columns

Usage

```
getSelectedCols(data = NULL, datasetInput = NULL, input = NULL)
```

Arguments

`data`	all loaded data
`datasetInput`	selected dataset
`input`	user input params

Examples

```
getSelectedCols()
```

`getSelectedDatasetInput` *getSelectedDatasetInput*

Description

Gathers the user selected dataset output to be displayed.

Usage

```
getSelectedDatasetInput(
  rdata = NULL,
  getSelected = NULL,
  getMostVaried = NULL,
  mergedComparison = NULL,
  input = NULL
)
```

Arguments

`rdata`	filtered dataset
`getSelected`	selected data
`getMostVaried`	most varied data
`mergedComparison`	merged comparison data
`input`	input parameters

Value

data

Examples

```
x <- getSelectedDatasetInput()
```

`getSelectInputBox` *getSelectInputBox*

Description

Selects user input conditions to run in DESeq.

Usage

```
getSelectInputBox(
  id = NULL,
  name = NULL,
  num = 0,
  choices = NULL,
  selected = NULL,
  cw = 2,
  multiple = FALSE
)
```

Arguments

`id`	input id
`name`	label of the box
`num`	panel that is going to be shown
`choices`	sample list
`selected`	selected smaple list
`cw`	column width
`multiple`	if multiple choices are available

Examples

```
x <- getSelectInputBox()
```

`getSelHeat` *getSelHeat*

Description

heatmap selection functionality

Usage

```
getSelHeat(expdata = NULL, input = NULL)
```

Arguments

`expdata`	selected genes
`input`	input params

Value

plot

Examples

```
x <- getSelHeat()
```

`getShapeColor` *getShapeColor*

Description

Generates the fill and shape selection boxes for PCA plots. metadata file has to be loaded in this case

Usage

```
getShapeColor(input = NULL)
```

Arguments

`input`	input values

Value

Color and shape from selection boxes or defaults

Examples

```
x <- getShapeColor()
```

`getStartPlotsMsg` *getStartPlotsMsg*

Description

Generates and displays the starting messgae to be shown once the user has first seen the main plots page within DEBrowser.

Usage

```
getStartPlotsMsg()
```

Value

return start plot msg

Note

`getStartPlotsMsg`

Examples

```
x <- getStartPlotsMsg()
```

`getStartupMsg` *getStartupMsg*

Description

Generates and displays the starting message within DEBrowser.

Usage

```
getStartupMsg()
```

Value

return startup msg

Note

`getStartupMsg`

Examples

```
x <- getStartupMsg()
```

`getTableDetails` *getTableDetails*

Description

get table details To be able to put a table into two lines are necessary; into the server part; getTableDetails(output, session, "dataname", data, modal=TRUE) into the ui part; uiOutput(ns("dataname"))

Usage

```
getTableDetails(
  output = NULL,
  session = NULL,
  tablename = NULL,
  data = NULL,
  modal = NULL
)
```

Arguments

`output`	output
`session`	session
`tablename`	table name
`data`	matrix data
`modal`	if it is true, the matrix is going to be in a modal

Value

panel

Examples

```
x <- getTableDetails()
```

`getTableModal` *getTableModal prepares table modal for KEGG*

Description

getTableModal prepares table modal for KEGG

Usage

```
getTableModal()
```

Value

the info button

Examples

```
x<- getTableModal()
```

`getTableStyle` *getTableStyle*

Description

User defined selection that selects the style of table to display within the DEBrowser.

Usage

```
getTableStyle(
  dat = NULL,
  input = NULL,
  padj = c("padj"),
  foldChange = c("foldChange"),
  DEsection = TRUE
)
```

Arguments

`dat`	dataset
`input`	input params
`padj`	the name of the padj value column in the dataset
`foldChange`	the name of the foldChange column in the dataset
`DEsection`	if it is in DESection or not

Note

`getTableStyle`

Examples

```
x <- getTableStyle()
```

`getTabUpdateJS` *getTabUpdateJS prepmenu tab and discovery menu tab updates*

Description

getTabUpdateJS prepmenu tab and discovery menu tab updates

Usage

```
getTabUpdateJS()
```

Value

the JS for tab updates

Examples

```
x<- getTabUpdateJS()
```

`getUp` *getUp get up regulated data*

Description

getUp get up regulated data

Usage

```
getUp(filt_data = NULL)
```

Arguments

`filt_data`	filt_data

Value

data

Examples

```
x <- getUp()
```

`getUpDown` *getUpDown get up+down regulated data*

Description

getUpDown get up+down regulated data

Usage

```
getUpDown(filt_data = NULL)
```

Arguments

`filt_data`	filt_data

Value

data

Examples

```
x <- getUpDown()
```

`getVariationData` *getVariationData*

Description

Adds an id to the data frame being used.

Usage

```
getVariationData(inputdata = NULL, cols = NULL, conds = NULL, key = NULL)
```

Arguments

`inputdata`	dataset
`cols`	columns
`conds`	conditions
`key`	gene or region name

Value

plotdata

Examples

```
x <- getVariationData()
```

`get_conditions_given_selection` *get_conditions_given_selection*

Description

Return the two set of conditions given the selection of meta select box

Usage

```
get_conditions_given_selection(metadata = NULL, selection = NULL)
```

Arguments

`metadata`	meta data table
`selection`	selection

Value

meta select box

Examples

```
x<-get_conditions_given_selection()
```

`heatmapControlsUI` *heatmapControlsUI*

Description

Generates the left menu to be used for heatmap plots

Usage

```
heatmapControlsUI(id)
```

Arguments

`id`	module ID

Value

HeatmapControls

Note

`heatmapControlsUI`

Examples

```
x <- heatmapControlsUI("heatmap")
```

`heatmapJScode` *heatmapJScode*

Description

heatmap JS code for selection functionality

Usage

```
heatmapJScode()
```

Value

JS Code

Examples

```
x <- heatmapJScode()
```

`heatmapServer` *heatmapServer*

Description

Sets up shinyServer to be able to run heatmapServer interactively.

Usage

```
heatmapServer(input, output, session)
```

Arguments

`input`	input params from UI
`output`	output params to UI
`session`	session variable

Value

the panel for main plots;

Note

`heatmapServer`

Examples

```
    heatmapServer
```

`heatmapUI` *heatmapUI*

Description

Creates a shinyUI to be able to run DEBrowser interactively.

Usage

```
heatmapUI(input, output, session)
```

Arguments

`input`	input variables
`output`	output objects
`session`	session

Value

the panel for heatmapUI;

Note

`heatmapUI`

Examples

```
x<-heatmapUI()
```

`hideObj` *hideObj*

Description

Hides a shiny object.

Usage

```
hideObj(btns = NULL)
```

Arguments

`btns`	hide group of objects with shinyjs

Examples

```
x <- hideObj()
```

`histogramControlsUI` *histogramControlsUI*

Description

Generates the controls in the left menu for a histogram

Usage

```
histogramControlsUI(id)
```

Arguments

`id`	namespace id

Value

returns the left menu

Note

`histogramControlsUI`

Examples

```
x <- histogramControlsUI("histogram")
```

`installpack` *installpack*

Description

install packages if they don’t exist display.

Usage

```
installpack(package_name = NULL)
```

Arguments

`package_name`	package name to be installed

Note

`installpack`

Examples

```
x <- installpack()
```

`IQRPlotControlsUI` *IQRPlotControlsUI*

Description

Generates the controls in the left menu for an IQR plot#'

Usage

```
IQRPlotControlsUI(id)
```

Arguments

`id`	namespace id

Value

returns the left menu

Note

`IQRPlotControlsUI`

Examples

```
x <- IQRPlotControlsUI("IQR")
```

`kmeansControlsUI` *kmeansControlsUI*

Description

get kmeans controls

Usage

```
kmeansControlsUI(id)
```

Arguments

`id`	module ID

Value

controls

Note

`kmeansControlsUI`

Examples

```
x <- kmeansControlsUI("heatmap")
```

`lcfMetRadio` *lcfMetRadio*

Description

Radio buttons for low count removal methods

Usage

```
lcfMetRadio(id)
```

Arguments

`id`	namespace id

Value

radio control

Note

`lcfMetRadio`

Examples

```
x <- lcfMetRadio("lcf")
```

`loadpack` *loadpack*

Description

load packages

Usage

```
loadpack(package_name = NULL)
```

Arguments

`package_name`	package name to be loaded

Note

`loadpack`

Examples

```
    x <- loadpack()
```

`mainPlotControlsUI` *mainPlotControlsUI*

Description

Generates the left menu to be used for main plots

Usage

```
mainPlotControlsUI(id)
```

Arguments

`id`	module ID

Value

mainPlotControls

Note

`mainPlotControlsUI`

Examples

```
x <- mainPlotControlsUI("main")
```

`mainScatterNew` *mainScatterNew*

Description

Creates the main scatter, volcano or MA plot to be displayed within the main panel.

Usage

```
mainScatterNew(input = NULL, data = NULL, cond_names = NULL, source = NULL)
```

Arguments

`input`	input params
`data`	dataframe that has log2FoldChange and log10padj values
`cond_names`	condition names
`source`	for event triggering to select genes

Value

scatter, volcano or MA plot

Examples

```
x <- mainScatterNew()
```

`niceKmeans` *niceKmeans*

Description

Generates hierarchially clustered K-means clusters

Usage

```
niceKmeans(df = NULL, input = NULL, iter.max = 1000, nstart = 100)
```

Arguments

`df`	data
`input`	user inputs
`iter.max`	max iteration for kmeans clustering
`nstart`	n for kmeans clustering

Value

heatmap plot area

Note

`niceKmeans`

Examples

```
x <- niceKmeans()
```

`normalizationMethods` *normalizationMethods*

Description

Select box to select normalization method prior to batch effect correction

Usage

```
normalizationMethods(id)
```

Arguments

`id`	namespace id

Value

radio control

Note

`normalizationMethods`

Examples

```
x <- normalizationMethods("batch")
```

`palUI` *palUI*

Description

get pallete

Usage

```
palUI(id)
```

Arguments

`id`	namespace ID

Value

pals

Note

`palUI`

Examples

```
x <- palUI("heatmap")
```

`panel.cor` *panel.cor*

Description

Prepares the correlations for the all2all plot.

Usage

```
panel.cor(x, y, prefix = "rho=", cex.cor = 2, ...)
```

Arguments

`x`	numeric vector x
`y`	numeric vector y
`prefix`	prefix for the text
`cex.cor`	correlation font size
`...`	additional parameters

Value

all2all correlation plots

Examples

```
panel.cor(c(1,2,3), c(4,5,6))
```

`panel.hist` *panel.hist*

Description

Prepares the historgram for the all2all plot.

Usage

```
panel.hist(x, ...)
```

Arguments

`x`	a vector of values for which the histogram is desired
`...`	any additional params

Value

all2all histogram plots

Examples

```
panel.hist(1)
```

`pcaPlotControlsUI` *pcaPlotControlsUI*

Description

Generates the PCA PLots Left menu to be displayed within the DEBrowser.

Usage

```
pcaPlotControlsUI(id = "pca")
```

Arguments

`id`	namespace id

Value

returns the left menu according to the selected tab;

Note

`pcaPlotControlsUI`

Examples

```
x <- pcaPlotControlsUI("pca")
```

`plotData` *plotData*

Description

prepare plot data for mainplots

Usage

```
plotData(pdata = NULL, input = NULL)
```

Arguments

`pdata`	data
`input`	input

Value

prepdata

Note

`plotData`

Examples

```
x <- plotData()
```

`plotMarginsUI` *plotMarginsUI*

Description

Margins module for plotly plots

Usage

```
plotMarginsUI(id, t = 20, b = 100, l = 100, r = 20)
```

Arguments

`id`	id
`t`	top margin
`b`	bottom margin
`l`	left margin
`r`	right margin

Value

size and margins controls

Note

`plotMarginsUI`

Examples

```
x <- plotMarginsUI("heatmap")
```

`plotSizeMarginsUI` *plotSizeMarginsUI*

Description

Size and margins module for plotly plots

Usage

```
plotSizeMarginsUI(id, w = 800, h = 640, t = 20, b = 100, l = 100, r = 20)
```

Arguments

`id`	id
`w`	width
`h`	height
`t`	top margin
`b`	bottom margin
`l`	left margin
`r`	right margin

Value

size and margins controls

Note

`plotSizeMarginsUI`

Examples

```
x <- plotSizeMarginsUI("heatmap")
```

`plotSizeUI` *plotSizeUI*

Description

Size module for plotly plots

Usage

```
plotSizeUI(id, w = 800, h = 600)
```

Arguments

`id`	id
`w`	width
`h`	height

Value

size and margins controls

Note

`plotSizeUI`

Examples

```
x <- plotSizeUI("heatmap")
```

`plotTypeUI` *plotTypeUI*

Description

Plot download type

Usage

```
plotTypeUI(id)
```

Arguments

`id`	id

Value

size and margins controls

Note

`plotTypeUI`

Examples

```
x <- plotTypeUI("heatmap")
```

`plot_pca` *plot_pca*

Description

Plots the PCA results for the selected dataset.

Usage

```
plot_pca(
  dat = NULL,
  pcx = 1,
  pcy = 2,
  metadata = NULL,
  color = NULL,
  shape = NULL,
  size = NULL,
  textonoff = "On",
  legendSelect = "samples",
  input = NULL
)
```

Arguments

`dat`	data
`pcx`	x axis label
`pcy`	y axis label
`metadata`	additional data
`color`	color for plot
`shape`	shape for plot
`size`	size of the plot
`textonoff`	text on off
`legendSelect`	select legend
`input`	input param

Value

pca list

Examples

```
load(system.file("extdata", "demo", "demodata.Rda",
        package="debrowser"))
metadata<-cbind(colnames(demodata[,1:6]),
        colnames(demodata[,1:6]),
        c(rep("Cond1",3), rep("Cond2",3)))
colnames(metadata)<-c("samples", "color", "shape")

a <- plot_pca(getNormalizedMatrix(
        demodata[rowSums(demodata[,1:6])>10,1:6]),
        metadata = metadata, color = "samples",
        size = 5, shape = "shape")
```

`prepDataContainer` *prepDataContainer*

Description

Prepares the data container that stores values used within DESeq.

Usage

```
prepDataContainer(data = NULL, counter = NULL, input = NULL, meta = NULL)
```

Arguments

`data`	loaded dataset
`counter`	the number of comparisons
`input`	input parameters
`meta`	loaded metadata

Value

data

Examples

```
x <- prepDataContainer()
```

`prepGroup` *prepGroup*

Description

prepare group table

Usage

```
prepGroup(conds = NULL, cols = NULL, metadata = NULL, covariates = NULL)
```

Arguments

`conds`	inputconds
`cols`	columns
`metadata`	metadata
`covariates`	covariates

Value

data

Examples

```
x <- prepGroup()
```

`prepHeatData` *prepHeatData*

Description

scales the data

Usage

```
prepHeatData(expdata = NULL, input = NULL)
```

Arguments

`expdata`	a matrixthat includes expression values
`input`	input variables

Value

heatdata

Examples

```
x <- prepHeatData()
```

`prepPCADat` *prepPCADat*

Description

prepares pca data with metadata. If metadata doesn't exists it puts all the sampels into a signlge group; "Conds".

Usage

```
prepPCADat(pca_data = NULL, metadata = NULL, input = NULL, pcx = 1, pcy = 2)
```

Arguments

`pca_data`	pca run results
`metadata`	additional meta data
`input`	input
`pcx`	x axis label
`pcy`	y axis label

Value

Color and shape from selection boxes or defaults

Examples

```
x <- prepPCADat()
```

`push` *push*

Description

Push an object to the list.

Usage

```
push(l, ...)
```

Arguments

`l`	that are going to push to the list
`...`	list object

Value

combined list

Examples

```
mylist <- list()
newlist <- push ( 1, mylist )
```

`removeCols` *removeCols*

Description

remove unnecessary columns

Usage

```
removeCols(cols = NULL, dat = NULL)
```

Arguments

`cols`	columns that are going to be removed from data frame
`dat`	data

Value

data

Examples

```
x <- removeCols()
```

`removeExtraCols` *removeExtraCols*

Description

remove extra columns for QC plots

Usage

```
removeExtraCols(dat = NULL)
```

Arguments

`dat`	selected data

Examples

```
removeExtraCols()
```

`round_vals` *round_vals*

Description

Plot PCA results.

Usage

```
round_vals(l)
```

Arguments

`l`	the value

Value

round value

Examples

```
x<-round_vals(5.1323223)
```

`runDE` *runDE*

Description

Run DE algorithms on the selected parameters. Output is to be used for the interactive display.

Usage

```
runDE(
  data = NULL,
  metadata = NULL,
  columns = NULL,
  conds = NULL,
  params = NULL
)
```

Arguments

`data`	A matrix that includes all the expression raw counts, rownames has to be the gene, isoform or region names/IDs
`metadata`	metadata of the matrix of expression raw counts
`columns`	is a vector that includes the columns that are going to be analyzed. These columns has to match with the given data.
`conds`	experimental conditions. The order has to match with the column order
`params`	all params for the DE methods

Value

de results

Examples

```
x <- runDE()
```

`runDESeq2` *runDESeq2*

Description

Run DESeq2 algorithm on the selected conditions. Output is to be used for the interactive display.

Usage

```
runDESeq2(
  data = NULL,
  metadata = NULL,
  columns = NULL,
  conds = NULL,
  params = NULL
)
```

Arguments

`data`	A matrix that includes all the expression raw counts, rownames has to be the gene, isoform or region names/IDs
`metadata`	metadata of the matrix of expression raw counts
`columns`	is a vector that includes the columns that are going to be analyzed. These columns has to match with the given data.
`conds`	experimental conditions. The order has to match with the column order

`params`	fitType: either "parametric", "local", or "mean" for the type of fitting of dispersions to the mean intensity. See estimateDispersions for description. betaPrior: whether or not to put a zero-mean normal prior on the non-intercept coefficients See nbinomWaldTest for description of the calculation of the beta prior. By default, the beta prior is used only for the Wald test, but can also be specified for the likelihood ratio test. testType: either "Wald" or "LRT", which will then use either Wald significance tests (defined by nbinomWaldTest), or the likelihood ratio test on the difference in deviance between a full and reduced model formula (defined by nbinomLRT) shrinkage: Adds shrunken log2 fold changes (LFC) and SE to a results table from DESeq run without LFC shrinkage. For consistency with results, the column name lfcSE is used here although what is returned is a posterior SD. Three shrinkage estimators for LFC are available via type (see the vignette for more details on the estimators). The apeglm publication demonstrates that 'apeglm' and 'ashr' outperform the original 'normal' shrinkage estimator.

Value

deseq2 results

Examples

```
x <- runDESeq2()
```

`runEdgeR` *runEdgeR*

Description

Run EdgeR algorithm on the selected conditions. Output is to be used for the interactive display.

Usage

```
runEdgeR(
  data = NULL,
  metadata = NULL,
  columns = NULL,
  conds = NULL,
  params = NULL
)
```

Arguments

`data`	A matrix that includes all the expression raw counts, rownames has to be the gene, isoform or region names/IDs
`metadata`	metadata of the matrix of expression raw counts

`columns`	is a vector that includes the columns that are going to be analyzed. These columns has to match with the given data.
`conds`	experimental conditions. The order has to match with the column order
`params`	normfact: Calculate normalization factors to scale the raw library sizes. Values can be "TMM","RLE","upperquartile","none". dispersion: either a numeric vector of dispersions or a character string indicating that dispersions should be taken from the data object. If a numeric vector, then can be either of length one or of length equal to the number of genes. Allowable character values are "common", "trended", "tagwise" or "auto". Default behavior ("auto" is to use most complex dispersions found in data object. testType: exactTest or glmLRT. exactTest: Computes p-values for differential abundance for each gene between two digital libraries, conditioning on the total count for each gene. The counts in each group as a proportion of the whole are assumed to follow a binomial distribution. glmLRT: Fit a negative binomial generalized log-linear model to the read counts for each gene. Conduct genewise statistical tests for a given coefficient or coefficient contrast.

Value

edgeR results

Examples

```
x <- runEdgeR()
```

`runHeatmap` *runHeatmap*

Description

Creates a heatmap based on the user selected parameters within shiny

Usage

```
runHeatmap(input = NULL, session = NULL, expdata = NULL)
```

Arguments

`input`	input variables
`session`	session
`expdata`	a matrix that includes expression values

Value

heatmapply plot

Examples

```
x <- runHeatmap()
```

`runHeatmap2` *runHeatmap2*

Description

Creates a heatmap based on the user selected parameters within shiny

Usage

```
runHeatmap2(input = NULL, session = NULL, expdata = NULL)
```

Arguments

`input`	input variables
`session`	session
`expdata`	a matrix that includes expression values

Value

heatmap.2

Examples

```
x <- runHeatmap2()
```

`runLimma` *runLimma*

Description

Run Limma algorithm on the selected conditions. Output is to be used for the interactive display.

Usage

```
runLimma(
  data = NULL,
  metadata = NULL,
  columns = NULL,
  conds = NULL,
  params = NULL
)
```

Arguments

`data`	A matrix that includes all the expression raw counts, rownames has to be the gene, isoform or region names/IDs
`metadata`	metadata of the matrix of expression raw counts
`columns`	is a vector that includes the columns that are going to be analyzed. These columns has to match with the given data.
`conds`	experimental conditions. The order has to match with the column order
`params`	normfact: Calculate normalization factors to scale the raw library sizes. Values can be "TMM","RLE","upperquartile","none". fitType, fitting method; "ls" for least squares or "robust" for robust regression normBet: Normalizes expression intensities so that the intensities or log-ratios have similar distributions across a set of arrays.

Value

Limma results

Examples

```
x <- runLimma()
```

`run_pca` *run_pca*

Description

Runs PCA on the selected dataset.

Usage

```
run_pca(x = NULL, retx = TRUE, center = TRUE, scale = TRUE)
```

Arguments

`x`	dataframe with experiment data
`retx`	specifies if the data should be returned
`center`	center the PCA (Boolean)
`scale`	scale the PCA (Boolean)

Value

pca list

Examples

```
load(system.file("extdata", "demo", "demodata.Rda",
    package="debrowser"))
pca_data<-run_pca(getNormalizedMatrix(
    demodata[rowSums(demodata[,1:6])>10,1:6]))
```

`selectConditions` *selectConditions*

Description

Selects user input conditions, multiple if present, to be used in DESeq.

Usage

```
selectConditions(
  Dataset = NULL,
  metadata = NULL,
  choicecounter = NULL,
  session = NULL,
  input = NULL
)
```

Arguments

`Dataset`	used dataset
`metadata`	metadatatable to select from metadata
`choicecounter`	choicecounter to add multiple comparisons
`session`	session
`input`	input params

Value

the panel for go plots;

Note

`selectConditions`

Examples

```
x<- selectConditions()
```

`selectedInput` *selectedInput*

Description

Selects user input conditions to run in DESeq.

Usage

```
selectedInput(id = NULL, num = 0, default = NULL, input = NULL)
```

Arguments

`id`	input id
`num`	panel that is going to be shown
`default`	default text
`input`	input params

Examples

```
x <- selectedInput()
```

`selectGroupInfo` *selectGroupInfo*

Description

Group info column selection. This can be used in batch effect or coloring the groups in the plots.

Usage

```
selectGroupInfo(
  metadata = NULL,
  input = NULL,
  selectname = "groupselect",
  label = "Group info"
)
```

Arguments

`metadata`	metadata
`input`	input values
`selectname`	name of the select box
`label`	label of the select box

Note

`selectGroupInfo`

Examples

```
x <- selectGroupInfo()
```

`sepRadio` *sepRadio*

Description

Radio button for separators

Usage

```
sepRadio(id, name)
```

Arguments

`id`	module id
`name`	name

Value

radio control

Note

`sepRadio`

Examples

```
x <- sepRadio("meta", "metadata")
```

`setBatch` *setBatch to skip batch effect correction batch variable set with the filter results*

Description

setBatch to skip batch effect correction batch variable set with the filter results

Usage

```
setBatch(fd = NULL)
```

Arguments

`fd`	filtered data

Value

fd data

Examples

```
x <- setBatch()
```

`showObj` *showObj*

Description

Displays a shiny object.

Usage

```
showObj(btns = NULL)
```

Arguments

`btns`	show group of objects with shinyjs

Examples

```
x <- showObj()
```

`startDEBrowser` *startDEBrowser*

Description

Starts the DEBrowser to be able to run interactively.

Usage

```
startDEBrowser()
```

Value

the app

Note

`startDEBrowser`

Examples

```
startDEBrowser()
```

`startHeatmap` *startHeatmap*

Description

Starts the DEBrowser heatmap

Usage

```
startHeatmap()
```

Value

the app

Note

`startHeatmap`

Examples

```
startHeatmap()
```

`textareaInput` *textareaInput*

Description

Generates a text area input to be used for gene selection within the DEBrowser.

Usage

```
textareaInput(id, label, value, rows = 20, cols = 35, class = "form-control")
```

Arguments

`id`	id of the control
`label`	label of the control
`value`	initial value
`rows`	the # of rows
`cols`	the # of cols
`class`	css class

Examples

```
x <- textareaInput("genesetarea", "Gene Set",
    "Fgf21", rows = 5, cols = 35)
```

`togglePanels` *togglePanels*

Description

User defined toggle to display which panels are to be shown within DEBrowser.

Usage

```
togglePanels(num = NULL, nums = NULL, session = NULL)
```

Arguments

`num`	selected panel
`nums`	all panels
`session`	session info

Note

`togglePanels`

Examples

```
x <- togglePanels()
```

Index