Package ‘wilson’

April 19, 2021

Type Package

Title Web-Based Interactive Omics Visualization

Version 2.4.2

Description Tool-set of modules for creating web-based applications that use plot based strategies to visualize and analyze multi-omics data. This package utilizes the ‘shiny’ and ‘plotly’ frameworks to provide a user friendly dashboard for interactive plotting.

URL https://github.com/loosolab/wilson/

BugReports https://github.com/loosolab/wilson/issues/

License MIT + file LICENSE

Encoding UTF-8

Imports shiny, data.table, ggplot2, plotly (> 4.8.0), scales, shinydashboard, DT (>= 0.3), colourpicker, RColorBrewer, shinyjs, viridis, rje, grDevices, grid, plyr, circlize, ComplexHeatmap, stats, gplots, reshape, rintrojs, RJSONIO, ggrepel (>= 0.6.12), DESeq2, rjson, FactoMineR, factoextra, heatmaply (>= 0.14.1), shinyjsloaders, log4r, openssl, methods, R6, zip, shinyWidgets

RoxygenNote 7.1.1

biocViews

Suggests knitr, rmarkdown, testthat, stringi, utils

VignetteBuilder knitr

NeedsCompilation no

Author Hendrik Schultheis [aut, cre], Jens Preussner [aut], Looso Mario [aut]

Maintainer Hendrik Schultheis <hendrik.schultheis@mpi-bn.mpg.de>

Repository CRAN

Date/Publication 2021-04-19 09:40:02 UTC
**R topics documented:**

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>and</td>
<td>3</td>
</tr>
<tr>
<td>andUI</td>
<td>4</td>
</tr>
<tr>
<td>categoricalPalettes</td>
<td>5</td>
</tr>
<tr>
<td>Clarion</td>
<td>5</td>
</tr>
<tr>
<td>colorPicker</td>
<td>8</td>
</tr>
<tr>
<td>colorPickerUI</td>
<td>9</td>
</tr>
<tr>
<td>columnSelector</td>
<td>10</td>
</tr>
<tr>
<td>columnSelectorUI</td>
<td>11</td>
</tr>
<tr>
<td>create_geneview</td>
<td>12</td>
</tr>
<tr>
<td>create_heatmap</td>
<td>13</td>
</tr>
<tr>
<td>create_pca</td>
<td>14</td>
</tr>
<tr>
<td>create_scatterplot</td>
<td>16</td>
</tr>
<tr>
<td>divergingPalettes</td>
<td>17</td>
</tr>
<tr>
<td>download</td>
<td>18</td>
</tr>
<tr>
<td>equalize</td>
<td>19</td>
</tr>
<tr>
<td>featureSelector</td>
<td>19</td>
</tr>
<tr>
<td>featureSelectorGuide</td>
<td>20</td>
</tr>
<tr>
<td>featureSelectorUI</td>
<td>21</td>
</tr>
<tr>
<td>forceArgs</td>
<td>21</td>
</tr>
<tr>
<td>geneView</td>
<td>22</td>
</tr>
<tr>
<td>geneViewGuide</td>
<td>23</td>
</tr>
<tr>
<td>geneViewUI</td>
<td>23</td>
</tr>
<tr>
<td>global_cor_heatmap</td>
<td>24</td>
</tr>
<tr>
<td>global_cor_heatmapUI</td>
<td>25</td>
</tr>
<tr>
<td>global_cor_heatmap_guide</td>
<td>25</td>
</tr>
<tr>
<td>heatmap</td>
<td>26</td>
</tr>
<tr>
<td>heatmapGuide</td>
<td>27</td>
</tr>
<tr>
<td>heatmapUI</td>
<td>27</td>
</tr>
<tr>
<td>install_app</td>
<td>28</td>
</tr>
<tr>
<td>label</td>
<td>28</td>
</tr>
<tr>
<td>labelUI</td>
<td>29</td>
</tr>
<tr>
<td>limit</td>
<td>30</td>
</tr>
<tr>
<td>limitUI</td>
<td>30</td>
</tr>
<tr>
<td>log_message</td>
<td>31</td>
</tr>
<tr>
<td>marker</td>
<td>31</td>
</tr>
<tr>
<td>markerUI</td>
<td>32</td>
</tr>
<tr>
<td>orNumeric</td>
<td>32</td>
</tr>
<tr>
<td>orNumericUI</td>
<td>33</td>
</tr>
<tr>
<td>orTextual</td>
<td>34</td>
</tr>
<tr>
<td>orTextualUI</td>
<td>35</td>
</tr>
<tr>
<td>parser</td>
<td>35</td>
</tr>
<tr>
<td>parse_MaxQuant</td>
<td>36</td>
</tr>
<tr>
<td>pca</td>
<td>37</td>
</tr>
<tr>
<td>pcaGuide</td>
<td>38</td>
</tr>
<tr>
<td>pcaUI</td>
<td>38</td>
</tr>
<tr>
<td>release_questions</td>
<td>39</td>
</tr>
</tbody>
</table>
Description

This function evaluates output from multiple OR modules by combining with a logical and.

Usage

```r
and(
  input, 
  output, 
  session, 
  data, 
  show.elements = NULL, 
  element.grouping = NULL, 
  column.labels = NULL, 
  delimiter = NULL, 
  multiple = TRUE, 
  contains = FALSE, 
  ranged = FALSE, 
  step = 100, 
  reset = NULL
)
```

Arguments

- `input`: Shiny’s input object.
- `output`: Shiny’s output object.
- `session`: Shiny’s session object.
- `data`: The input data.frame for which selection should be provided. Evaluates an OR module for each column (Supports reactive).
- `show.elements`: A Vector of column names determining which OR modules are shown. Defaults to `names(data)`. (Supports reactive)
element.grouping
Group features in boxes. (Data.table: first column = columnnames, second column = groupnames) (Supports reactive)
column.labels
Additional labels for the columns, defaults to names(data).
delimiter
A single character, or a vector indicating how column values are delimited. (Fills vector sequentially if needed)(Supports reactive)
multiple
Whether or not textual ORs should allow multiple selections. (Fills vector sequentially if needed)(Supports reactive)
contains
Whether or not textual ORs are initialized as textInput checking entries for given string. (Fills vector sequentially if needed)(Supports reactive)
ranged
Whether or not numeric ORs are ranged. (Fills vector sequentially if needed)(Supports reactive)
step
Set numeric ORs slider steps. (Fills vector sequentially if needed)(Supports reactive)
reset
Reactive which will cause a UI reset on change.

Value
A reactive containing named list with a boolean vector of length nrow(data) (bool), indicating whether an observation is selected or not and a vector of Strings showing the used filter (text).

andUI
AND module UI representation

Description
The AND module connects filtering and selection across multiple columns of a data.frame. Columns of class boolean, character or factor will be represented as textual ORs, numeric columns as numerical OR.

Usage
andUI(id)

Arguments
id The ID of the modules namespace.

Value
A list with HTML tags from tag.
categoricalPalettes  

*Function to generate categorical (qualitative) color palettes*

**Description**

Function to generate categorical (qualitative) color palettes

**Usage**

`categoricalPalettes(n)`

**Arguments**

- `n` Number of colors to generate

**Value**

A data.table with (named) color palettes of length n

---

**Clarion**

*Clarion R6-class definition*

**Description**

Use this to create a clarion object. This object is used by all top-level wilson modules.

**Constructor**

`Clarion$new(header = NULL, metadata, data, validate = TRUE)`

**Constructor Arguments**

<table>
<thead>
<tr>
<th>Variable</th>
<th>Return</th>
</tr>
</thead>
<tbody>
<tr>
<td>header</td>
<td>A named list. Defaults to NULL.</td>
</tr>
<tr>
<td>metadata</td>
<td>Clarion metadata in form of a data.table.</td>
</tr>
<tr>
<td>data</td>
<td>Data.table according to metadata.</td>
</tr>
<tr>
<td>validate</td>
<td>Logical value to validate on initialization. Defaults to TRUE.</td>
</tr>
</tbody>
</table>

**Public fields**

- header  List of global information regarding the whole experiment.
- metadata Data.table with additional information for each column.
data Data.table containing experiment result data.

Methods

Public methods:

- `Clarion$get_id()`
- `Clarion$get_name()`
- `Clarion$get_delimiter()`
- `Clarion$is_delimited()`
- `Clarion$get_factors()`
- `Clarion$get_level()`
- `Clarion$get_label()`
- `Clarion$validate()`
- `Clarion$new()`
- `Clarion$write()`
- `Clarion$clone()`

**Method `get_id()`**: Returns name of unique identifier column. Assumes first feature to be unique if not specified.

*Usage:*

```
Clarion$get_id()
```

*Returns:* Name of the id column.

**Method `get_name()`**: Returns name of name column. If not specified return unique Id.

*Usage:*

```
Clarion$get_name()
```

*Returns:* Name of the name column.

**Method `get_delimiter()`**: Return delimiter used within multi-value fields (no delimiter = NULL).

*Usage:*

```
Clarion$get_delimiter()
```

**Method `is_delimited()`**: Logical whether the given column name is delimited.

*Usage:*

```
Clarion$is_delimited(x)
```

*Arguments:*

- `x` Name of the column.

*Returns:* boolean

**Method `get_factors()`**: Get factors to all columns.

*Usage:*

```
Clarion$get_factors()
```
Details: Named factors (e.g. factor1="name") will be cropped to their name.

Returns: Returns a data.table columns: key and factor(s) if any.

Method get_level(): Get level(s) to given column name(s).

Usage:
Clarion$get_level(column)

Arguments:
column One or more column name(s).

Returns: Provide a vector of levels to the given column names in column. Returns NA for missing columns and character(0) if column = NULL.

Method get_label(): Get label(s) to given column name(s).

Usage:
Clarion$get_label(column = NULL, sub_label = TRUE, sep = " ")

Arguments:
column One or more column name(s).
sub_label Whether the sub_label should be included.
sep Separator between label and sub_label.

Details: If a column does not have a label the key is returned.

Returns: Provides a vector of labels (+ sub_label) to the given column names in column. Returns NA for missing columns and all labels if column = NULL.

Method validate(): Check the object for inconsistencies.

Usage:
Clarion$validate(solve = TRUE)

Arguments:
solve For solve = TRUE try to resolve some warnings.

Method new(): Initialize a new clarion object.

Usage:
Clarion$new(header = NULL, metadata, data, validate = TRUE)

Arguments:
header A named list. Defaults to NULL.
metadata Clarion metadata in form of a data.table.
data Data.table according to metadata.
validate Logical value to validate on initialization. Defaults to TRUE.

Returns: Clarion object.

Method write(): Save the object as a clarion file.

Usage:
Clarion$write(file)

Arguments:
Method `clone()`: The objects of this class are cloneable with this method.

Usage:
```
Clarion$clone(deep = FALSE)
```

Arguments:
- `deep` Whether to make a deep clone.

Examples
```
## Not run:
# initializing a new object
object <- Clarion$new(header, metadata, data, validate = TRUE)

# create a deep copy
object_copy <- object$clone(deep = TRUE)

## End(Not run)
```

---

colorPicker module server logic

Description

Provides server logic for the colorPicker2 module.

Usage
```
colorPicker(
  input,  # Shiny's input object
  output,  # Shiny's output object
  session,  # Shiny's session object
  num.colors = 256,  # Define length of colorpalette vector (Default = 256).
  distribution = "all",  # Define distribution
  winsorize = NULL,  # Winsorize values
  selected = NULL   # Selected color
)
```

Arguments
- `input` Shiny’s input object
- `output` Shiny’s output object
- `session` Shiny’s session object
- `num.colors` Define length of colorpalette vector (Default = 256).
colorPickerUI

distribution  Decide which palettes are selectable. One or more of list("sequential", "diverging", "categorical"). Defaults to "all" (Supports reactive).

winsorize  Numeric vector of two. Dynamically change lower and upper limit (supports reactive). Defaults to NULL.

selected  Set the default selected palette.

Details

A custom colorpalette’s return will be NULL if there is something wrong with it. equalize will be returned as FALSE if not selected.

Value

Reactive containing list(palette = c(colors), name = palette_name, transparency = Integer, reverse = Boolean, winsorize = NULL or a two-component vector containing lower and upper limits).

colorPickerUI  

Description

The functions creates HTML tag definitions of its representation based on the parameters supplied. Currently, two UI can be created for the user to choose either (a) colors from a given color scheme, or (b) choose one or more single colors.

Usage

colorPickerUI(
  id,
  label = "Color scheme",
  custom = FALSE,
  multiple = FALSE,
  show.reverse = TRUE,
  show.scaleoptions = TRUE,
  show.transparency = TRUE
)

Arguments

id  The ID of the modules namespace.
label  Either a character vector of length one with the label for the color scheme dropdown, or a character vector containing labels of the single colors.
custom  Boolean if TRUE custom colors can be selected (Default = FALSE).
multiple  Boolean value, if set to TRUE custom colorpalettes can be made. Only if custom = TRUE (Default = FALSE).
show.reverse Logical, whether or not to show the reverse switch (Default = TRUE).
show.scaleoptions Logical, whether or not to show color scaling option winsorize (Default = TRUE).
show.transparency Logical, whether or not to show the transparency slider (Default = TRUE).

Value
A list with HTML tags from tag.

columnSelector module server logic

description
columnSelector module server logic

Usage
columnSelector(
  input,
  output,
  session,
  type.columns,
  type = NULL,
  column.type.label = "Type of Column",
  label.label = "Label",
  multiple = TRUE,
  none = FALSE,
  sep = ",",
  suffix = NULL
)

Arguments
input Shiny’s input object
output Shiny’s output object
session Shiny’s session object
type.columns data.table: (Supports reactive) key = columnnames (id) level = datalevel/ type of column label = optional, used instead of id sub_label = optional, added to id/ label
type The type (contrast/group/sample of the type dropdown menu, selected in step 1 (upper dropdown). Defaults to unique(type.columns[,2]) (Supports reactive)
column.type.label Changes the label of the first UI element
columnSelectorUI

<table>
<thead>
<tr>
<th>label.label</th>
<th>Change label above label text input.</th>
</tr>
</thead>
<tbody>
<tr>
<td>multiple</td>
<td>Boolean value whether multiple values can be selected in second selector. (Default = TRUE)</td>
</tr>
<tr>
<td>none</td>
<td>If TRUE adds &quot;None to secondSelector and select is. (Default = FALSE)</td>
</tr>
<tr>
<td>sep</td>
<td>Used to separate labels on multi value selection.</td>
</tr>
<tr>
<td>suffix</td>
<td>Added to label only on multiple = FALSE (supports reactive). Also uses sep as separator.</td>
</tr>
</tbody>
</table>

**Value**

Returns the input. As named list: names("type", "selected_columns", "label")

---

**Description**

columnSelector module UI representation

**Usage**

columnSelectorUI(id, label = FALSE, title = NULL)

**Arguments**

<table>
<thead>
<tr>
<th>id</th>
<th>The ID of the modules namespace.</th>
</tr>
</thead>
<tbody>
<tr>
<td>label</td>
<td>Boolean value; if true include a text input field with the desired axis label (this should be preset with the headline of the column)</td>
</tr>
<tr>
<td>title</td>
<td>String which is displayed as module title. (Default = NULL)</td>
</tr>
</tbody>
</table>

**Value**

A list from `tag` with the UI elements.
create_geneview  Method for geneView creation

Description

Method for geneView creation

Usage

create_geneview(
data,
  grouping,
  plot.type = "line",
  facet.target = "gene",
  facet.cols = 2,
  colors = NULL,
  ylabel = NULL,
  ylimits = NULL,
  gene.label = NULL,
  plot.method = "static",
  width = "auto",
  height = "auto",
  ppi = 72,
  scale = 1
)

Arguments

data       data.table containing plot data
grouping   data.table metadata containing: column1 : key column2 : factor1
plot.type  String specifying which plot type is used c("box", "line", "violin", "bar").
facet.target Target to plot on x-Axis c("gene", "condition").
facet.cols  Number of plots per row.
colors      Vector of colors used for color palette
ylabel      Label of the y-axis (default = NULL).
ylimits     Vector defining scale of y-axis (default = NULL).
gene.label  Vector of labels used instead of gene names (default = NULL).
plot.method Choose which method used for plotting. Either "static" or "interactive" (Default = "static").
width       Set the width of the plot in cm (default = "auto").
height      Set the height of the plot in cm (default = "auto").
ppi         Pixel per inch (default = 72).
scale       Modify plot size while preserving aspect ratio (Default = 1).
create_heatmap

Details

Width/ height limit = 500. If exceeded default to 500 and issue exceed_size = TRUE.

Value

Returns depending on plot.method list(\texttt{plot} = ggplot/ plotly object, \texttt{width} = width in cm, \texttt{height} = height in cm, \texttt{ppi} = pixel per inch, \texttt{exceed_size} = Boolean).

description

Method for heatmap creation

Usage

\begin{verbatim}
create_heatmap(
  data,
  unitlabel = "auto",
  row.label = TRUE,
  row.custom.label = NULL,
  column.label = TRUE,
  column.custom.label = NULL,
  clustering = "none",
  clustdist = "auto",
  clustmethod = "auto",
  colors = NULL,
  winsorize.colors = NULL,
  plot.method = "static",
  width = "auto",
  height = "auto",
  ppi = 72,
  scale = 1
)
\end{verbatim}

Arguments

data \hspace{1cm} data.table containing plot data. First column contains row labels.
unitlabel \hspace{1cm} label of the colorbar
row.label \hspace{1cm} Logical whether or not to show row labels.
row.custom.label \hspace{1cm} Vector of custom row labels.
column.label \hspace{1cm} Logical whether or not to show column labels.
column.custom.label \hspace{1cm} Vector of custom column labels.
clustering: How to apply clustering on data. c("none", "both", "column", "row")
clustdist: Which cluster distance to use. See heatmap.
clustmethod: Which cluster method to use. See heatmap.
colors: Vector of colors used for color palette.
winsorize.colors: NULL or a vector of length two, giving the values of colorbar ends (default = NULL).
plot.method: Choose which method is used for plotting. Either "plotly" or "complexHeatmap" (Default = "complexHeatmap").
width: Set width of plot in cm (Default = "auto").
height: Set height of plot in cm (Default = "auto").
ppi: Pixel per inch (default = 72).
scale: Modify plot size while preserving aspect ratio (Default = 1).

Details
Width/ height limit = 500. If exceeded default to 500 and issue exceed_size = TRUE.

Value
Returns list(plot = complexHeatmap/ plotly object, width = width in cm, height = height in cm, ppi = pixel per inch, exceed_size = Boolean whether width/ height exceeded max) depending on plot.method.

Description
Method for pca creation.

Usage
create_pca(
data,
color.group = NULL,
color.title = NULL,
palette = NULL,
shape.group = NULL,
shape.title = NULL,
shapes = c(15:25),
dimension.a = 1,
dimension.b = 2,
dimensions = 6,
on.columns = TRUE,
create_pca

```r
labels = FALSE,
custom.labels = NULL,
pointsize = 2,
labelsize = 3,
width = 28,
height = 28,
ppi = 72,
scale = 1
)
```

**Arguments**

- **data**
  - data.table from which the plot is created (First column will be handled as row-names if not numeric).
- **color.group**
  - Vector of groups according to samples (= column names).
- **color.title**
  - Title of the color legend.
- **palette**
  - Vector of colors used for color palette.
- **shape.group**
  - Vector of groups according to samples (= column names).
- **shape.title**
  - Title of the shape legend.
- **shapes**
  - Vector of shapes see `points`. Will recycle/ cut off shapes if needed. Default = `c(15:25)`
- **dimension.a**
  - Number of dimension displayed on X-Axis.
- **dimension.b**
  - Number of dimension displayed on Y-Axis.
- **dimensions**
  - Number of dimensions to create.
- **on.columns**
  - Boolean perform pca on columns or rows.
- **labels**
  - Boolean show labels.
- **custom.labels**
  - Vector of custom labels. Will replace columnnames.
- **pointsize**
  - Size of the data points.
- **labelsize**
  - Size of texts inside plot (default = 3).
- **width**
  - Set the width of the plot in cm (default = 28).
- **height**
  - Set the height of the plot in cm (default = 28).
- **ppi**
  - Pixel per inch (default = 72).
- **scale**
  - Modify plot size while preserving aspect ratio (Default = 1).

**Details**

If width and height are the same axis ratio will be set to one (quadratic plot).
Width/ height limit = 500. If exceeded default to 500 and issue exceed_size = TRUE.

**Value**

A named list(plot = ggplot object, data = pca.data, width = width of plot (cm), height = height of plot (cm), ppi = pixel per inch, exceed_size = Boolean whether width/ height exceeded max).
create_scatterplot

Method for scatter plot creation

Description

Method for scatter plot creation

Usage

```r
create_scatterplot(
  data,
  data.labels = NULL,
  data.hovertext = NULL,
  transparency = 1,
  pointsize = 1,
  labelsize = 3,
  color = NULL,
  x_label = "",
  y_label = "",
  z_label = "",
  density = TRUE,
  line = TRUE,
  categorized = FALSE,
  highlight.data = NULL,
  highlight.labels = NULL,
  highlight.hovertext = NULL,
  highlight.color = "#FF0000",
  xlim = NULL,
  ylim = NULL,
  colorbar.limits = NULL,
  width = "auto",
  height = "auto",
  ppi = 72,
  plot.method = "static",
  scale = 1
)
```

Arguments

data: data.table containing plot data column 1: id column 2, 3(, 4): x, y(, z)
data.labels: Vector of labels used for data. Length has to be equal to nrow(data).
data.hovertext: Character vector with additional hovertext. Length has to be equal to nrow(data).
transparency: Set point transparency. See `geom_point`.
pointsize: Set point size. See `geom_point`.
labelsize: Set label size. See `geom_text`.
divergingPalettes

Function to generate diverging (two-sided) color palettes (e.g. for log2fc, zscore)

Description

Function to generate diverging (two-sided) color palettes (e.g. for log2fc, zscore)

Usage

divergingPalettes(n)
download

**Arguments**

- `n` Number of colors to generate

**Value**

A data.table with (named) color palettes of length n

---

**download**

Function used for downloading. Creates a zip container containing plot in png, pdf and user input in json format. Use inside `downloadHandler` content function.

**Description**

Function used for downloading. Creates a zip container containing plot in png, pdf and user input in json format. Use inside `downloadHandler` content function.

**Usage**

```r
download(
  file,
  filename,
  plot,
  width,
  height,
  ppi = 72,
  save_plot = TRUE,
  ui = NULL
)
```

**Arguments**

- `file` See `downloadHandler` content parameter.
- `filename` See `downloadHandler`.
- `plot` Plot to download.
- `width` in centimeter.
- `height` in centimeter.
- `ppi` pixel per inch. Defaults to 72.
- `save_plot` Logical if plot object should be saved as .RData.
- `ui` List of user inputs. Will be converted to JavaScript Object Notation. See `toJSON`.

**Value**

Path to zip archive invisibly. See `zipr`. 
equalize

Method to get equalized min/max values from vector

Description

Method to get equalized min/max values from vector

Usage

equalize(values)

Arguments

values Numeric vector or table

Value

Vector with c(min, max).

featureSelector

featureSelector module server logic

Description

featureSelector module server logic

Usage

featureSelector(
  input,
  output,
  session,
  clarion,
  multiple = TRUE,
  contains = FALSE,
  ranged = TRUE,
  step = 100,
  truncate = 30,
  selection.default = "all"
)
Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Shiny’s input object.</td>
</tr>
<tr>
<td>output</td>
<td>Shiny’s output object.</td>
</tr>
<tr>
<td>session</td>
<td>Shiny’s session object.</td>
</tr>
<tr>
<td>clarion</td>
<td>A clarion object. See Clarion. (Supports reactive)</td>
</tr>
<tr>
<td>multiple</td>
<td>Whether or not textual ORs should allow multiple selections. (Fills vector sequentially if needed)(Supports reactive)</td>
</tr>
<tr>
<td>contains</td>
<td>Whether or not textual ORs are initialized as textInput checking entries for given string. (Fills vector sequentially if needed)(Supports reactive)</td>
</tr>
<tr>
<td>ranged</td>
<td>Whether or not numeric ORs are ranged. (Fills vector sequentially if needed)(Supports reactive)</td>
</tr>
<tr>
<td>step</td>
<td>Set numeric ORs number of slider steps. (Fills vector sequentially if needed)(Supports reactive)</td>
</tr>
<tr>
<td>truncate</td>
<td>Truncate datatable entries at x characters (Default = 30).</td>
</tr>
<tr>
<td>selection.default</td>
<td>Decide whether everything or nothing is selected on default (no filters applied). Either &quot;all&quot; or &quot;none&quot; (Default = &quot;all&quot;).</td>
</tr>
</tbody>
</table>

Details

Keep in mind that the order of features (columns in clarion$data) is the order in which multiple, contains, ranged and step are evaluated.

Value

Reactive containing names list: Selected data as reactive containing clarion object (object). Used filter to select data (filter).

---

**featureSelectorGuide**  
***featureSelector module guide***

Description

featureSelector module guide

Usage

featureSelectorGuide(session)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>session</td>
<td>The shiny session</td>
</tr>
</tbody>
</table>

Value

A shiny reactive that contains the texts for the guide steps.
**featureSelectorUI**  
*featureSelector module UI representation*

**Description**  
featureSelector module UI representation

**Usage**  
`featureSelectorUI(id)`

**Arguments**  
- id  
  - The ID of the modules namespace

**Value**  
A list with HTML tags from `tag`

---

**forceArgs**  
*Force evaluation of the parent function’s arguments.*

**Description**  
Force evaluation of the parent function’s arguments.

**Usage**  
`forceArgs(args)`

**Arguments**  
- args  
  - List of Argument names to force evaluation. Defaults to all named arguments  
    see `match.call`.

**Details**  
Similar to `forceAndCall` but used from within the respective function.  
This method is not using `force` as it is restricted to it’s calling environment. Instead `get` is used.
Description

Provides server logic for the geneView module.

Usage

geneView(
  input,
  output,
  session,
  clarion,
  plot.method = "static",
  label.sep = ", "
)

Arguments

input Shiny’s input object.
output Shiny’s output object.
session Shiny’s session object.
clarion A clarion object. See Clarion. (Supports reactive)
plot.method Choose which method is used for plotting. Either "static" or "interactive" (Default = "static").
label.sep Separator used for label merging (Default = ", ").
width Width of the plot in cm. Defaults to minimal size for readable labels and supports reactive.
height Height of the plot in cm. Defaults to minimal size for readable labels and supports reactive.
ppi Pixel per inch. Defaults to 72 and supports reactive.
scale Scale plot size. Defaults to 1, supports reactive.

Details

Width/ height/ ppi less or equal to default will use default value.
Ppi less or equal to zero will use default.

Value

Reactive containing data.table used for plotting.
geneViewGuide

geneView module guide

Description

geneView module guide

Usage

geneViewGuide(session)

Arguments

session The shiny session

Value

A shiny reactive that contains the texts for the Guide steps.

geneViewUI

geneView’s module UI representation

Description

geneView’s module UI representation

Usage

geneViewUI(id, plot.columns = 3)

Arguments

id The ID of the modules namespace.
plot.columns Initial value of plot column slider. Integer value between 1 and 7 (Default = 3).

Value

A list with HTML tags from tag.


**global_cor_heatmap**

global correlation heatmap module server logic

**Description**

global correlation heatmap module server logic

**Usage**

```r
global_cor_heatmap(
  input, output, session, clarion,
  plot.method = "static",
  width = "auto",
  height = "auto",
  ppi = 72,
  scale = 1
)
```

**Arguments**

- **input** Shiny’s input object
- **output** Shiny’s output object
- **session** Shiny’s session object
- **clarion** A clarion object. See Clarion. (Supports reactive)
- **plot.method** Choose which method is used for plotting. Either "static" or "interactive" (Default = "static").
- **width** Width of the plot in cm. Defaults to minimal size for readable labels and supports reactive.
- **height** Height of the plot in cm. Defaults to minimal size for readable labels and supports reactive.
- **ppi** Pixel per inch. Defaults to 72 and supports reactive.
- **scale** Scale plot size. Defaults to 1, supports reactive.

**Value**

Reactive containing data used for plotting.
global_cor_heatmapUI

description

global correlation heatmap module UI representation

usage

global_cor_heatmapUI(id)

arguments

id

the ID of the modules namespace.

value

A list with HTML tags from tag

global_cor_heatmap_guide

global correlation heatmap module guide

description

global correlation heatmap module guide

usage

global_cor_heatmap_guide(session)

arguments

session

the shiny session

value

A shiny reactive that contains the texts for the Guide steps.
Description

heatmap module server logic

Usage

heatmap(
  input, output, session, clarion,
  plot.method = "static", label.sep = ", "
  width = "auto", height = "auto",
  ppi = 72, scale = 1
)

Arguments

input  Shiny's input object
output Shiny's output object
session Shiny's session object
clarion A clarion object. See Clarion. (Supports reactive)
plot.method Choose which method is used for plotting. Either "static" or "interactive" (Default = "static").
label.sep Separator used for label merging (Default = ", ",
width  Width of the plot in cm. Defaults to minimal size for readable labels and supports reactive.
height Height of the plot in cm. Defaults to minimal size for readable labels and supports reactive.
ppi    Pixel per inch. Defaults to 72 and supports reactive.
scale  Scale plot size. Defaults to 1, supports reactive.

Value

Reactive containing data used for plotting.
### heatmapGuide

**heatmap module guide**

#### Description

heatmap module guide

#### Usage

`heatmapGuide(session)`

#### Arguments

- `session` — The shiny session

#### Value

A shiny reactive that contains the texts for the Guide steps.

---

### heatmapUI

**heatmap module UI representation**

#### Description

heatmap module UI representation

#### Usage

`heatmapUI(id, row.label = TRUE)`

#### Arguments

- `id` — The ID of the modules namespace.
- `row.label` — Boolean Value set initial Value for rowlabel checkbox (Default = TRUE).

#### Value

A list with HTML tags from `tag`.
install_app | Download and install Wilson Apps

Description

Download and install Wilson Apps

Usage

```r
install_app(
    location = ".",
    remove_data = FALSE,
    start_after_install = FALSE,
    app_name = "wilson-basic",
    repository = "https://github.molgen.mpg.de/loosolab/wilson-apps"
)
```

Arguments

- **location**: Where the app should be installed. Default is current location.
- **remove_data**: If TRUE demo data will be deleted.
- **start_after_install**: Start the app when done installing.
- **app_name**: Select app to install.
- **repository**: Link to the repository that holds the apps.

Details

Will create a folder named after parameter app_name.

---

label | label module server logic

Description

label module server logic
Usage

```r
label(
  input,
  output,
  session,
  data,
  label = "Select label columns",
  multiple = TRUE,
  sep = ", ",
  unique = TRUE,
  unique_sep = "_",
  disable = NULL
)
```

Arguments

- **input**: Shiny's input object.
- **output**: Shiny's output object.
- **session**: Shiny’s session object.
- **data**: Data.table used for label creation. Column names will be used for selection. (supports reactive)
- **label**: Set label of selectizeInput.
- **multiple**: Allow multiple selection which will be merged with sep (default = TRUE).
- **sep**: Separator used to collapse selection (default = ", ").
- **unique**: Make labels unique. Defaults to TRUE. See `make.unique`.
- **unique_sep**: Separator used for unique (default = "_"). Should differ from sep.
- **disable**: Reactive containing boolean. To disable/ enable module.

Value

Reactive containing list(label = vector of strings or NULL on empty selection, selected = user input).

---

**labelUI**  
*label module UI representation*

Description

label module UI representation

Usage

```r
labelUI(id)
```
Arguments

id The ID of the modules namespace

Value

A list with HTML tags from tag

limit

limit module server logic

Description

limit module server logic

Usage

limit(input, output, session, lower = NULL, upper = NULL)

Arguments

input Shiny’s input object.
output Shiny’s output object.
session Shiny’s session object.
lower Set lower limit (supports reactive).
upper Set upper limit (supports reactive).

Value

Reactive containing: list(lower, upper).

limitUI

limit module UI representation

Description

limit module UI representation

Usage

limitUI(id, label = "Limit")

Arguments

id The ID of the modules namespace
label Set the modules label.
log_message

Value

A list with HTML tags from tag

---

log_message  

**logger message convenience function**

---

**Description**

logger message convenience function

**Usage**

```r
log_message(
  message,
  level = c("DEBUG", "INFO", "WARN", "ERROR", "FATAL"),
  token = NULL
)
```

**Arguments**

- **message**  
  String of message to be written in log. See `levellog`.

- **level**  
  Set priority level of the message (number or character). See `levellog`.

- **token**  
  Use token bound to this identifier.

**Details**

Does nothing if logger doesn’t exist.

---

marker  

**marker module server logic**

---

**Description**

marker module server logic

**Usage**

```r
marker(input, output, session, clarion)
```

**Arguments**

- **input**  
  Shiny’s input object.

- **output**  
  Shiny’s output object.

- **session**  
  Shiny’s session object.

- **clarion**  
  A clarion object. See Clarion. (Supports reactive)
markerUI

marker module UI representation

Description

marker module UI representation

Usage

markerUI(id, label = "Highlight/ Label Selected Features")

Arguments

id 
The ID of the modules namespace

label 
Set label of first element.

Value

A list with HTML tags from tag

orNumeric

orNumeric module server logic

Description

Provides server logic for the orNumeric module.

Usage

orNumeric(
  input,
  output,
  session,
  choices,
  value,
  label = "Column",
  step = 100,
  stepsize = NULL,
  min. = shiny::reactive(min(choices_r(), na.rm = TRUE)),
  max. = shiny::reactive(max(choices_r(), na.rm = TRUE)),
  label.slider = NULL,
  zoomable = TRUE,
  reset = NULL
)

Value

A named list containing reactives (highlight, color, labelColumn, label, clarion).
**Arguments**

- **input**: Shiny’s input object.
- **output**: Shiny’s output object.
- **session**: Shiny’s session object.
- **choices**: A list or a numeric vector with the possible choices offered in the UI. See `sliderInput` (Supports reactive).
- **value**: Initial value of the slider. Creates a ranged slider if numeric vector of two given (Supports reactive).
- **label**: Label of the entire module.
- **step**: Number of steps on interval (Default = 100).
- **stepsize**: Value defining interval size of the slider. Will be used instead of step (Default = NULL).
- **min.**: Minimum value that can be selected on slider (defaults to min(choices)) (Supports reactive).
- **max.**: Maximum value that can be selected on slider (defaults to max(choices)) (Supports reactive).
- **label.slider**: A character vector of length one with the label for the `sliderInput`.
- **zoomable**: Boolean to enable zooming. Redefine the sliders range. Defaults to TRUE.
- **reset**: A reactive which will trigger a module reset on change.

**Value**

Returns a reactive containing a named list with the label, the selected choices as a character vector (text), a boolean vector of length length(choices) (bool), and a vector of the selected value(s) (value), indicating whether a item has been chosen. If no item has been chosen, the return is TRUE for items.

---

**orNumericUI**

 **orNumeric** module **UI representation**

**Description**

This module allows to select value/range inputs from a `sliderInput` element. The functions creates HTML tag definitions of its representation based on the parameters supplied.

**Usage**

```r
orNumericUI(id)
```

**Arguments**

- **id**: The ID of the modules namespace.

**Value**

A list with HTML tags from `tag`.
orTextual  
orTextual module server logic

Description

Provides server logic for the orTextual module.

Usage

orTextual(
  input, 
  output, 
  session, 
  choices, 
  selected = NULL, 
  label = "Column", 
  delimiter = NULL, 
  multiple = TRUE, 
  contains = FALSE, 
  reset = NULL, 
  parse_mode = TRUE
)

Arguments

input  Shiny’s input object.
output Shiny’s output object.
session Shiny’s session object.
choices A list or a character vector with the possible choices offered in the UI. See selectInput.
selected The initially selected value. See selectInput.
label A character vector of length one with the label for the selectInput.
delimiter A single character indicating if and how items are delimited (default: NULL indicates no delimitation). Only if contains = FALSE.
multiple Whether or not selection of multiple items is allowed.
contains Logical variable. If TRUE shows module as a textsearch input.
reset A reactive which will trigger a module reset on change.
parse_mode Boolean to enable text to selection parsing. Ignored if multiple = FALSE or contains = TRUE.

Value

Returns a reactive containing a named list with the label, the selected choices as a character vector (text) and a boolean vector of length length(choices) (bool), indicating whether a item has been chosen. If no item has been chosen, the return is TRUE for items.
**orTextualUI**  

**orTextual module UI representation**

---

**Description**

This module allows to select (multiple) inputs from a `selectInput` element. The functions creates HTML tag definitions of its representation based on the parameters supplied.

**Usage**

`orTextualUI(id)`

**Arguments**

- **id**  
The ID of the modules namespace.

**Value**

A list with HTML tags from `tag`.

---

**parser**  

**Method to parse input file.**

---

**Description**

Method to parse input file.

**Usage**

`parser(file, dec = ".")`

**Arguments**

- **file**  
  Path to file that needs parsing.

- **dec**  
  The decimal separator. See `fread`.

**Value**

Clarion object. See `Clarion`
parse_MaxQuant  Converting MaxQuant Output file proteinGroups.txt to CLARION format by creating a headline of metadata for each column

Description
List with columns of reduced version (see config.json file) If you only want the samples of a specific keyword write: column;exp For example: You got: Intensity Intensity 'experiment_name' Do you want both add "Intensity" to the list. Do you only want the sample add "Intensity;exp" to the list Anything else like 'Intensity;ex' or 'Intensity;' results in writing both. Only works if there are samples of that type. If not, column does not show up in file

Usage
parse_MaxQuant(
  proteinGroups_in,
  summary_in,
  outfile,
  outfile_reduced,
  config = system.file("extdata", "parser_MaxQuant_config.json", package = "wilson"),
  delimiter = ";",
  format = NULL,
  version = NULL,
  experiment_id = NULL
)

Arguments
proteingroups_in  path of proteinGroup.txt file
summary_in  path of belonging summary.txt file
outfile  path of full CLARION output file
outfile_reduced  path of reduced CLARION output file
config  path of config file (containing information about metadata)
delimiter  delimiter (Default = ;)
format  pre-header information about format (optional)
version  pre-header information about version (optional)
experiment_id  pre-header information about experiment id (optional)

Value
TRUE on success
Author(s)

Rene Wiegandt

Description

pca module server logic

Usage

```r
pca(
  input,
  output,
  session,
  clarion,
  width = 28,
  height = 28,
  ppi = 72,
  scale = 1
)
```

Arguments

- **input**: Shiny’s input object
- **output**: Shiny’s output object
- **session**: Shiny’s session object
- **clarion**: A clarion object. See Clarion. (Supports reactive)
- **width**: Width of the plot in cm. Defaults to 28 and supports reactive.
- **height**: Height of the plot in cm. Defaults to 28 and supports reactive.
- **ppi**: Pixel per inch. Defaults to 72 and supports reactive.
- **scale**: Scale plot size. Defaults to 1, supports reactive.

Details

Width/ height/ ppi less or equal to zero will use default value.

Value

A reactive containing list with dimensions.
### pcaGuide

**Description**

pca module guide

**Usage**

```r
pcaGuide(session)
```

**Arguments**

- `session` The shiny session

**Value**

A shiny reactive that contains the texts for the Guide steps.

### pcaUI

**Description**

pca module UI representation

**Usage**

```r
cpaUI(id, show.label = TRUE)
```

**Arguments**

- `id` The ID of the modules namespace.
- `show.label` Set initial value of show label checkbox (Default = TRUE).

**Value**

A list with HTML tags from `tag`.
**release_questions**

*Defines additional questions asked before CRAN submission. DO NOT EXPORT!*

**Description**

Defines additional questions asked before CRAN submission. DO NOT EXPORT!

**Usage**

```r
release_questions()
```

**scatterPlot**

*scatterPlot module server logic*

**Description**

scatterPlot module server logic

**Usage**

```r
scatterPlot(
  input,
  output,
  session,
  clarion,
  marker.output = NULL,
  plot.method = "static",
  width = "auto",
  height = "auto",
  ppi = 72,
  scale = 1
)
```

**Arguments**

- `input`: Shiny’s input object
- `output`: Shiny’s output object
- `session`: Shiny’s session object
- `clarion`: A clarion object. See `Clarion`. (Supports reactive)
- `marker.output`: Marker module output. See `marker`
- `plot.method`: Choose to rather render a 'interactive' or 'static’ plot. Defaults to 'static'.
- `width`: Width of the plot in cm. Defaults to minimal size for readable labels and supports reactive.
height: Height of the plot in cm. Defaults to minimal size for readable labels and supports reactive.

ppi: Pixel per inch. Defaults to 72 and supports reactive.

scale: Scale plot size. Defaults to 1, supports reactive.

Details

As markerOutput provides a second dataset used for highlighting it is crucial for it to have the same columnnames as the dataset provided by clarion.

Intersections between marker and clarion will be removed from clarion in favor of highlighting them.

Value

Returns reactive containing data used for plot.

---

scatterPlotGuide

scatterPlot module guide

Description

scatterPlot module guide

Usage

scatterPlotGuide(session, marker = FALSE)

Arguments

- session: The shiny session
- marker: Logical if marker step should be enabled (Default = FALSE).

Value

A shiny reactive that contains the texts for the Guide steps.
scatterPlotUI  

**Description**

scatterPlot module UI representation

**Usage**

scatterPlotUI(id)

**Arguments**

- **id**
  
  The ID of the modules namespace.

**Value**

A list with HTML tags from `tag`.

---

searchData 

*Function to search data for selection*

**Description**

Function to search data for selection

**Usage**

searchData(  
  input,  
  choices,  
  options = c("=" ,"<", ">"),  
  min. = min(choices, na.rm = TRUE),  
  max. = max(choices, na.rm = TRUE)  
)

**Arguments**

- **input**
  
  Vector length one (single) or two (ranged) containing numeric values for selection.

- **choices**
  
  Vector on which input values are applied.

- **options**
  
  Vector on how the input and choices should be compared. It can contain: single = `c("=", ",<", ",>")` or ranged = `c("inner", ",outer")`.

- **min.**
  
  Minimum value that can be selected on slider (defaults to `min(choices)`).

- **max.**
  
  Maximum value that can be selected on slider (defaults to `max(choices)`).
Value
Returns a logical vector with the length of choices, where every matched position is TRUE.

sequentialPalettes
Function to generate sequential (one-sided) color palettes (e.g. for expression, enrichment)

Description
Function to generate sequential (one-sided) color palettes (e.g. for expression, enrichment)

Usage
sequentialPalettes(n)

Arguments
n Number of colors to generate

Value
A data.table with (named) color palettes of length n

set_logger
set a log4r logger used within the package

Description
set a log4r logger used within the package

Usage
set_logger(logger, token = NULL)

Arguments
logger A logger object see create.logger. NULL to disable logging.
token Set a unique identifier for this logger.

Details
This function will save each logger in the wilson.globals environment. Each logger is stored by the name 'logger'[token] (e.g. 'logger6b821824b0b53b1a3e8f531a34d0d6e6'). Use onSessionEnded to clean up after logging. See onFlush.
Description

Click here for more information about TOBIAS.

Usage

tobiasParser(
  input,
  output,
  filter_columns = NULL,
  filter_pattern = NULL,
  config = system.file("extdata", "tobias_config.json", package = "wilson"),
  omit_NA = FALSE,
  condition_names = NULL,
  condition_pattern = "_bound$",
  in_field_delimiter = ",",
  dec = ".",
  ...
)

Arguments

input Path to input table
output Output path.
filter_columns Either a vector of column names or a file containing one column name per row.
filter_pattern Keep columns matching the given pattern. Uses parameter filter_columns for matching if set. In the case of no matches a warning will be issued and all columns will be used.
config Json file containing metadata information for all columns. Will use first occurrence for duplicate column names.
omit_NA Logical whether all rows containing NA should be removed.
condition_names Vector of condition names. Default = NULL. Used to classify columns not provided in config.
condition_pattern Used to identify condition names by matching and removing given pattern with grep. Ignored when condition_names is set.
in_field_delimiter Delimiter for multi-value fields. Default = ",".
dec Decimal separator. Used in file reading and writing.
... Used as header information.
transformation

Details
During conversion the parser will try to use the given config (if provided) to create the Clarion metadata. In the case of insufficient config information it will try to approximate by referencing condition names issuing warnings in the process. As the format requires an unique id the parser will create one if necessary. Factor grouping (metadata factor columns) is currently not implemented!

transformation module server logic

Description
The module provides several transformations on a numeric data matrix for the user.

Usage
transformation(
  input, output, session, data,
  transpose = FALSE, pseudocount = 1, replaceInf = TRUE, replaceNA = TRUE
)

Arguments
input Shiny’s input object.
output Shiny’s output object.
session Shiny’s session object.
data Numeric matrix on which transformation is performed (column-wise). (Supports reactive)
transpose Whether the matrix should be transposed to enable row-wise transformation. (Supports reactive)
pseudocount Numeric Variable to add a pseudocount to log-based transformations. (Supports reactive)
replaceInf Change Infinite to NA, applied after transformation. (Supports reactive)
replaceNA Change NA to 0, applied after transformation. (Supports reactive)

Value
Namedlist of two containing data and name of the used method. data: Reactive containing the transformed matrix. Infinite values are replaced by NA and NA values are replaced by 0. method: Reactive containing String. transpose: Reactive containing String.
transformationUI

---

transformation module UI representation

---

**Description**

This function provides an input to select a transformation method.

**Usage**

```r
transformationUI(
  id,
  label = "Transformation",
  selected = "raw",
  choices = list(  
    None = "raw",  
    log2 = "log2",  
    `-log2` = "-log2",  
    log10 = "log10",  
    `-log10` = "-log10",  
    `Z score` = "zscore",  
    `regularized log` = "rlog"),
  transposeOptions = FALSE
)
```

**Arguments**

- **id**
  - The ID of the modules namespace.

- **label**
  - A character vector of length one with the label for the `selectInput`.

- **selected**
  - The initially selected value. See `selectInput`.

- **choices**
  - Named list of available transformations. Possible transformations are list(‘None’ = "raw", ‘log2’ = "log2", ‘-log2’ = "-log2", ‘log10’ = "log10", ‘-log10’ = "-log10", ‘Z score’ = "zscore", ‘regularized log’ = "rlog") which is also the default.

- **transposeOptions**
  - Boolean value if transpose radioButtons are shown (Default = FALSE).

**Value**

A list with HTML tags from tag.
Index

and, 3
andUI, 4
categoricalPalettes, 5
Clarion, 5, 20, 22, 24, 26, 31, 35, 37, 39
colorPicker, 8
colorPickerUI, 9
columnSelector, 10
columnSelectorUI, 11
create.logger, 42
create_geneview, 12
create_heatmap, 13
create_pca, 14
create_scatterplot, 16
divergingPalettes, 17
download, 18
downloadHandler, 18
equalize, 19
featureSelector, 19
featureSelectorGuide, 20
featureSelectorUI, 21
force, 21
forceAndCall, 21
forceArgs, 21
fread, 35
geneView, 22
geneViewGuide, 23
geneViewUI, 23
geom_point, 16
geom_text, 16
get, 21
global_cor_heatmap, 24
global_cor_heatmap_guide, 25
global_cor_heatmapUI, 25
grep, 43
heatmap, 26
heatmapGuide, 27
heatmapr, 14
heatmapUI, 27
install_app, 28
label, 28
labelUI, 29
levellog, 31
limit, 30
limitUI, 30
lims, 17
log_message, 31
make.unique, 29
marker, 31, 39
markerUI, 32
match.call, 21
onFlush, 42
orNumeric, 32
orNumericUI, 33
orTextual, 34
orTextualUI, 35
parse_MaxQuant, 36
parser, 35
pca, 37
pcaGuide, 38
pcaUI, 38
points, 15
release_questions, 39
scatterPlot, 39
scatterPlotGuide, 40
scatterPlotUI, 41
searchData, 41
selectInput, 34, 35, 45
sequentialPalettes, 42
set_logger, 42
INDEX

sliderInput, 33

tag, 4, 10, 11, 21, 23, 25, 27, 30–33, 35, 38, 41, 45
tobias_parser, 43
toJSON, 18
transformation, 44
transformationUI, 45

zipr, 18