MCnebula2

November 6, 2022

Title What the Package Does (One Line, Title Case)

Version 0.0.0.9000

Description What the package does (one paragraph).

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.0

Collate utils.R project.sirius.v4.R tools-colors.R tools-methods.R
tools-default_visualize.R tools-modify_ggset.R tools-export.R
tools-MSnbase-MODIFIED_compareSpectra.R tools-yaml.R tools-report.R
base-generic.R extra-generic.R main-generic.R
class-VIRTUAL_slots.R class-melody.R
class-project_conformation.R class-project_metadata.R
class-project_api.R class-project_dataset.R class-project.R
class-statistic_set.R class-mcn_dataset.R
class-msframe.R class-command.R class-ggset.R class-report.R
class-section.R class-nebula.R class-mcnebula.R
methods-initialize_mcnebula.R
extraMethods-collate_data.R methods-filter_structure.R
methods-filter_formula.R methods-filter_ppcp.R
methods-create_hierarchy.R methods-create_reference.R
methods-create_features_annotation.R
methods-create_stardust_classes.R methods-cross_filter_stardust.R
methods-backtrack_stardust.R methods-create_nebula_index.R
methods-compute_spectral_similarity.R
methods-create_parent_nebula.R methods-create_child_nebulae.R
methods-create_parent_layout.R methods-create_child_layouts.R
methods-activate_nebulae.R methods-visualize.R
extraMethods-draw_structures.R extraMethods-draw_nodes.R
methods-annotate_nebula.R extraMethods-binary_comparison.R
extraMethods-report.R

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Imports BiocStyle,
bookdown,
crayon,
data.table,
dplyr,
ggimage,
ggraph,
ggsci,
ggtext,
grid,
gridExtra,
grImport2,
igraph,
knitr,
pbapply,
rlang,
rmarkdown,
stringr,
styler,
svglite,
tidyr

Depends ggplot2

R topics documented:

ABSTRACT-MCnebula2 3
activate_nebulae-methods 6
annotate_nebula-methods 8
backtrack-class 9
backtrack_stardust-methods 9
binary_comparison-methods 10
code_block-class 12
collate_data-methods 14
command-class 16
compute_spectral_similarity-methods 18
create_child_layouts-methods 19
create_child_nebulae-methods 20
create_features_annotation-methods 21
create_hierarchy-methods 22
create_nebula_index,missing,missing-method 23
create_parent_layout-methods 24
create_parent_nebula-methods 25
create_reference-methods 26
create_stardust_classes-methods 28
cross_filter_stardust-methods 29
dataset-class 31
draw_nodes-methods 32

draw_structures-methods 34
export-class 35
filter_formula-methods 36
filter_ppcp-methods 37
filter_structure-methods 38
fun_modify 39
ggset-class 40
history_rblock-methods 41
include_figure-methods 42
include_table-methods 43
initialize_mcnebula-methods 44
layerSet-class 45
mcnebula-class 46
mcn_dataset-class 50
melody-class 51
msframe-class 53
nebula-class 56
project-class 60
project_api-class 63
project_conformation-class 64
project_dataset-class 66
project_metadata-class 67
reference-class 70
report-class 70
section-class 72
set_nodes_color-methods 74
set_ppcp_data-methods 75
set_ration_data-methods 76
set_tracer-methods 77
statistic_set-class 78
subscript-class 79
visualize-methods 80

Index **82**

`ABSTRACT-MCnebula2` *Overview of MCnebula2*

Description

MCnebula2 was used for metabonomics data analysis. It is written in the S4 system of object-oriented programming, and starts with a "class", namely "mcnebula". The whole process takes the "mcnebula" as the operating object to obtain visual results or operating objects.

Most methods of MCnebula2 are S4 methods and have the characteristics of parameterized polymorphism, that is, different functions will be used for processing according to different parameters passed to the same method.

MCnebula workflow is a complete metabolomics data analysis process, including initial data pre-processing (data format conversion, feature detection), compound identification based on MS/MS, statistical analysis, compound structure and chemical class focusing, multi-level data visualization, output report, etc.

It should be noted that the MCnebula2 R package currently cannot realize the entire analysis process of MCnebula workflow. If users want to complete the entire workflow, other software beyond the R console (for example, the MSconvert tool of proteowizard is used for data format conversion, which is a tool widely applicable to metabonomics and proteomics) should be used. This is a pity, but we will gradually integrate all parts of the workflow into this R package in the future to achieve one-stop analysis.

The analysis process in R is integrated into the following methods:

- `initialize_mcnebula()`
- `filter_structure()`
- `create_reference()`
- `filter_formula()`
- `create_stardust_classes()`
- `create_features_annotation()`
- `cross_filter_stardust()`
- `create_nebula_index()`
- `compute_spectral_similarity()`
- `create_parent_nebula()`
- `create_child_nebulae()`
- `create_parent_layout()`
- `create_child_layouts()`
- `activate_nebulae()`
- `visualize()`
- `binary_comparison()`
- ...

Details

Overall. We know that the analysis of untargeted LC-MS/MS dataset generally begin with feature detection. It detects 'peaks' as features in MS1 (MASS level 1) data. Each feature may represents a compound, and assigned with MS2 (MASS level 2) spectra. The MS2 spectra was used to find out the compound identity. The difficulty lies in annotating these features to discover their compound identity, mining out meaningful information, so as to serve further biological research. In addition, the un-targeted LC-MS/MS dataset is general a huge dataset, which leads to time-consuming analysis of the whole process. Herein, a classified visualization method, called MCnebula, was used for addressing these difficulty.

MCnebula utilizes the state-of-the-art computer prediction technology, SIRIUS workflow (SIRIUS, ZODIAC, CSI:fingerID, CANOPUS), for compound formula prediction, structure retrieve and classification prediction (`https://bio.informatik.uni-jena.de/software/sirius/`). MCnebula integrates an abundance-based classes (ABC) selection algorithm into features annotation: depending on the user, MCnebula focuses chemical classes with more or less features in the dataset (the abundance of classes), visualizes them, and displays the features they involved; these classes can be dominant structural classes or sub-structural classes. With MCnebula, we can switch from untargeted to targeted analysis, focusing precisely on the compound or chemical class of interest to the researcher.

MCnebula2. The MCnebula2 package itself does not contain any part of molecular formula prediction, structure prediction and chemical prediction of compounds, so the accuracy of these parts is not involved. MCnebula2 performs downstream analysis by extracting the prediction data from SIRIUS project. The core of MCnebula2 is its chemical filtering algorithm, called ABC selection algorithm.

Chemical structure and formula. To explain the ABC selection algorithm in detail, we need to start with MS/MS spectral analysis and identification of compounds: The analysis of MS/MS spectrum is a process of inference and prediction. For example, we speculate the composition of elements based on the molecular weight of MS1; combined with the possible fragmentation pattern of MS2 spectrum, we speculate the potential molecular formula of a compound; finally, we look for the exact compound from the compound structure database. Sometimes, this process is full of uncertainty, because there are too many factors that affect the reliability of MS/MS data and the correctness of inference. It can be assumed that there are complex candidates for the potential chemical molecular formula, chemical structure and chemical class behind MS/MS spectrum. Suppose we have these data of candidates now, MCnebula2 extracted these candidates and obtained the unique molecular formula and chemical structure for each MS/MS spectrum based on the highest score of chemical structure prediction; in this process, as most algorithms do, we make a choice based on the score, and only select the result of highest score.

The chemical formula and structure candidates can obtain by methods:

- `filter_formula()`
- `filter_structure()`

In order to obtain the best (maybe), corresponding and unique chemical formula and structure from complex candidates, an important intermediate link:

- `create_reference()`

Above, we talked about chemical molecular formula, chemical structural formula and chemical classes. We obtained the unique chemical molecular formula and chemical structure formula for reference by scoring and ranking. But for chemical classes, we can't adopt such a simple way to get things done.

Chemical classification. Chemical classification is a complex system. Here, we only discuss the structure based chemotaxonomy system, because the MS/MS spectrum is more indicative of the structure of compounds than biological activity and other information.

According to the division of the overall structure and local structure of compounds, we can call the structural characteristics as the dominant structure and substructure. (https://jcheminf.biomedcentral.com/articles/10.1186/s13321-016-0174-y). Correspondingly, in the chemical classification system, we can not only classify according to the dominant structure, but also classify according to the substructure. The chemical classification based on the dominant structure of compounds is easy to understand, because we generally define it in this way. For example, we will classify Taxifolin as "flavones", not "phenols", although its local structure has a substructure of "phenol".

We hope to classify a compound by its dominant structure rather than substructure, because such classify is more concise and contains more information. However, in the process of MS/MS spectral analysis, we sometimes can only make chemical classification based on the substructure of compounds, which may be due to: uncertainty in the process of structural analysis; it may be an unknown compound; MS/MS spectral fragment information is insufficient. In this case, it is necessary for us to classify the compounds with the aid of substructure information, otherwise we have no knowledge of the compounds for which we cannot obtain dominant structure information.

Above, we discussed the complex chemical classification for the substructure and dominant structure of compounds. We must also be clear about the complexity of another aspect of chemotaxonomy, i.e., the hierarchy of classification. This is easy to understand. For example, "Flavones" belongs to its superior, "Flavonoids"; its next higher level, "Phynylpropanoids and polyketides"; the further upward classification is "organic compounds".

ABC selection. The above section discusses the inferential prediction of individual MS/MS spectrum. In the un-targeted LC-MS/MS dataset, each feature has a corresponding MS/MS spectrum, and there are thousands of features in total. The ABC selection algorithm regards all features as a whole, examines the number and abundance of features of each chemical classification (classification at different levels, classification of substructure and dominant structure), and then selects representative classes (mainly screening the classes according to the number or abundance range of features) to serve the subsequent analysis. The core methods for ABC selection algorithm are:

- `create_stardust_classes()`
- `cross_filter_stardust()`
- `create_nebula_index()`

`activate_nebulae-methods` *generate 'ggset' for visualization*

Description

...

`activate_nebulae()`: get the default parameters for the method `activate_nebulae`.

`activate_nebulae(x, ...)`: use the default parameters whatever 'missing' while performing the method `activate_nebulae`.

...

FUNCTION_DESCRIPTION

...

Usage

```
## S4 method for signature 'missing,missing,missing'
activate_nebulae()

## S4 method for signature 'mcnebula,ANY,ANY'
activate_nebulae(x, fun_default_parent, fun_default_child)

## S4 method for signature 'mcnebula,`function`,`function`'
activate_nebulae(x, fun_default_parent, fun_default_child)

ggset_activate_parent_nebula(x)

ggset_activate_child_nebulae(x)
```

Arguments

x	...
fun_default_parent	...
fun_default_child	...

Details

...

...

DETAILS

...

Value

...

OUTPUT_DESCRIPTION

...

Examples

```
## Not run:
activate_nebulae(...)

## End(Not run)
## Not run:
if(interactive()){
 #EXAMPLE1
 }

## End(Not run)
## Not run:
```

```
if(interactive()){
 #EXAMPLE1
 }

## End(Not run)
```

---

annotate_nebula-methods

...

Description

...

...

Usage

```
## S4 method for signature 'ANY,character'
annotate_nebula(x, nebula_name)
```

Arguments

x	...
nebula_name	...

Details

...

...

Value

...

See Also

draw_nodes, draw_structures...

Examples

```
## Not run:
annotate_nebula(...)

## End(Not run)
```

backtrack-class *Share slots and methods for classes inherite from VIRTUAL_backtrack*

Description

This VIRTUAL class provides a slot for storing discarded data.

`backtrack`, `backtrack<-`: getter and setter for the `backtrack` slot of the object.

Usage

```
## S4 method for signature 'ANY'
backtrack(x)

## S4 replacement method for signature 'ANY'
backtrack(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`backtrack` list with names.

backtrack_stardust-methods ...

Description

...

...

...

...

Usage

```
## S4 method for signature 'mcnebula,missing,missing,missing'
backtrack_stardust(x)

## S4 method for signature 'mcnebula,character,missing,ANY'
backtrack_stardust(x, class.name, remove)

## S4 method for signature 'mcnebula,missing,numeric,ANY'
backtrack_stardust(x, rel.index, remove)
```

Arguments

`x`	...
`class.name`	...
`remove`	...
`rel.index`	...

Details

...

...

Value

...

See Also

`cross_filter_stardust`

Examples

```
## Not run:
backtrack_stardust(...)

## End(Not run)
## Not run:
backtrack_stardust(...)

## End(Not run)
## Not run:
backtrack_stardust(...)

## End(Not run)
```

`binary_comparison-methods`
...

Description

...

`binary_comparison()`: get the default parameters for the method `binary_comparison`.

`binary_comparison(x, ...)`: use the default parameters whatever 'missing' while performing the method `binary_comparison`.

...

Usage

```
## S4 method for signature 'missing,missing,missing,missing,missing'
binary_comparison()

## S4 method for signature 'ANY,ANY,ANY,ANY,ANY'
binary_comparison(x, ..., formula, fun_norm, top_coef, contrasts)

## S4 method for signature 'ANY,formula,`function`,ANY,character'
binary_comparison(x, formula, fun_norm, top_coef, contrasts)
```

Arguments

`x`	...
`...`	...
`formula`	...
`fun_norm`	...
`top_coef`	...
`contrasts`	...

Details

...

...

Value

...

See Also

`stats::model.matrix()`, `limma::makeContrasts()`, `limma::lmFit()`, `limma::eBayes()`, `limma::contrasts.fit()`, `limma::topTable()`...

Examples

```
## Not run:
binary_comparison(...)

## End(Not run)
```

code_block-class *Sequestrate code and setting run parameters.*

Description

Mainly desiged for R code block. The job of this class object is to record the codes and the running parameters of its source language or program; These information can then be output as formatted code block text (use `call_command()`).

`code_block_table`: class inherit from `code_block`, with default values for slot `command_args` facilitate showing table in document.

`code_block_figure`: class inherit from `code_block`, with default values for slot `command_args` facilitate showing figure in document.

`code_block`, `code_block<-`: getter and setter for the `code_block` slot of the object.

`codes`, `codes<-`: getter and setter for the `codes` slot of the object.

`new_code_block`: create a code_block object.

`new_code_block()`: get the default parameters for the method `new_code_block`.

`new_code_block(x, ...)`: use the default parameters whatever 'missing' while performing the method `new_code_block`.

`new_code_block_figure`: create code_block_figure object. This methods simplified parameter settings for displaying figures in documents.

`new_code_block_table`: create code_block_table object. This methods simplified parameter settings for displaying table in documents.

`call_command`: Format 'code_block' object as character.

Usage

```
## S4 method for signature 'code_block'
show(object)

## S4 method for signature 'code_block_table'
show(object)

## S4 method for signature 'code_block_figure'
show(object)

## S4 method for signature 'heading'
show(object)

## S4 method for signature 'section'
show(object)

## S4 method for signature 'ANY'
code_block(x)
```

```
## S4 replacement method for signature 'ANY'
code_block(x) <- value

## S4 method for signature 'code_block'
codes(x)

## S4 replacement method for signature 'code_block'
codes(x) <- value

## S4 method for signature 'character,character,list,logical,`function`'
new_code_block(language, codes, args, prettey, fun_prettey)

## S4 method for signature 'missing,missing,missing,missing,missing'
new_code_block()

## S4 method for signature 'ANY,ANY,ANY,ANY,ANY'
new_code_block(language, codes, args, prettey, fun_prettey)

## S4 method for signature 'character'
new_code_block_figure(name, caption, ...)

## S4 method for signature 'character'
new_code_block_table(name, ...)

## S4 method for signature 'code_block'
call_command(x)
```

Arguments

value	The value for the slot.
language	character(1). For slot command_name.
codes	character. For slot codes.
args	list. For slot command_args.
prettey	logical. If ture, use styler::style_text() to pretty the codes.
fun_prettey	function. Default is styler::style_text.
name	character(1). For cross-reference in document. See https://bookdown.org/yihui/rmarkdown-cookbook/cross-ref.html#cross-ref.
caption	character(1). Caption of figure display in document.
...	Other parameters passed to new_code_block().

Slots

codes character. Codes.

command_name character(1). Program or language. e.g., "r".

command_function function. Used for gather the codes and args as code block.

command_args list. Args passed to program.

See Also

command-class. https://bookdown.org/yihui/rmarkdown-cookbook/cross-ref.html#cross-ref. https://bookdown.org/yihui/rmarkdown/compile.html.

Other call_commands: command-class, ggset-class, report-class, section-class

Examples

```
## Not run: 
new('code_block', ...)

## End(Not run)
## Not run: 
new('code_block_table', ...)

## End(Not run)
## Not run: 
new('code_block_figure', ...)

## End(Not run)
## Not run: 
new_code_block(...)

## End(Not run)
## Not run: 
new_code_block_figure(...)

## End(Not run)
## Not run: 
new_code_block_table(...)

## End(Not run)
## Not run: 
call_command(...)

## End(Not run)
```

---

collate_data-methods *Extract and format data from raw project directory*

Description

The primary method used to extract data from the raw project directory. By specifying subscript, this method reads all corresponding files, followed by gathering and formating the data, then stores these data in the slot (dataset(project_dataset(object))).

collate_data(): get the default parameters for the method collate_data.

collate_data(x, ...): use the default parameters whatever 'missing' while performing the method collate_data.

read_data: basic methods used to extract and format data from raw project directory.

Usage

```
## S4 method for signature 'missing,missing,missing'
collate_data()

## S4 method for signature 'ANY,ANY,ANY'
collate_data(x, subscript, fun_collate, ...)

## S4 method for signature 'ANY,character,`function`'
collate_data(x, subscript, fun_collate, ...)

## S4 method for signature
## 'ANY,
##   project_metadata,
##   character,
##   missing,
##   missing,
##   missing,
##   missing,
##   missing'
read_data(x, project_metadata, subscript)

## S4 method for signature
## 'missing,
##   missing,
##   character,
##   character,
##   character,
##   character,
##   `function`,
##   `function`'
read_data(subscript, path, .features_id, .candidates_id, fun_read, fun_format)
```

Arguments

`x`	project object or other class object inheriting it.
`subscript`	character(1). See subscript.
`fun_collate`	function. Used to extract and format the data from raw project directory. The default is `MCnebula2:::.collate_data.msframe`.
`...`	Other parameters passed to the fun_collate.
`project_metadata`	project_metadata object. Specifying the files to read.
`path`	character(1). The path of raw project directory.
`.features_id`	character. ID for signing files in sub-directory of each 'features'.
`.candidates_id`	character. ID for signing each candidates of 'features'.
`fun_read`	function. Used to read files from raw project directory.
`fun_format`	function. Used to format the data.

Details

This methods requires the name and path of the file in the raw project directory, as well as the reading function; These are recorded in project.

Note

Normally, users do not need to use this method for MCnebula2 analysis. `filter_formula()`, `filter_structure()`, `filter_ppcp()` provide more understandable usage.

Examples

```
## Not run:
collate_data(...)

## End(Not run)
## Not run:
read_data(...)

## End(Not run)
```

`command-class` *Preparation of an instruction to be executed*

Description

Packing the funciton and the args inside this class object, so that it can be performed easily at any time.

`command_name`, `command_name<-`: getter and setter for the `command_name` slot of the object.

`command_function`, `command_function<-`: getter and setter for the `command_function` slot of the object.

`command_args`, `command_args<-`: getter and setter for the `command_args` slot of the object.

`new_command`: create an object of command.

`call_command`: Execute the function (slot `command_function`) with the parameters (slot `command_args`).

Usage

```
## S4 method for signature 'command'
show(object)

## S4 method for signature 'command'
command_name(x)

## S4 replacement method for signature 'command'
command_name(x) <- value

## S4 method for signature 'command'
```

```
command_function(x)

## S4 replacement method for signature 'command'
command_function(x) <- value

## S4 method for signature 'command'
command_args(x)

## S4 replacement method for signature 'command'
command_args(x) <- value

## S4 method for signature '`function`,character'
new_command(fun, ..., name)

## S4 method for signature '`function`,missing'
new_command(fun, ..., name)

## S4 method for signature 'command'
call_command(x)
```

Arguments

`value`	The value for the slot.
`fun`	function.
`...`	parameters (with names or without names) passed to the function.
`name`	character(1). Name to slot `command_name`.

Slots

`command_name` character(1). Describe the command name.

`command_function` function.

`command_args` the parameters passed to the function.

See Also

Other call_commands: `code_block-class`, `ggset-class`, `report-class`, `section-class`

Other call_commands: `code_block-class`, `ggset-class`, `report-class`, `section-class`

Examples

```
## Not run:
new('command', ...)

## End(Not run)
## Not run:
new_command(...)

## End(Not run)
## Not run:
call_command(...)

## End(Not run)
```

compute_spectral_similarity-methods

...

Description

...

`compute_spectral_similarity()`: get the default parameters for the method `compute_spectral_similarity`.

`compute_spectral_similarity(x, ...)`: use the default parameters whatever 'missing' while performing the method `compute_spectral_similarity`.

...

Usage

```
## S4 method for signature 'missing,missing,missing,missing,missing'
compute_spectral_similarity()

## S4 method for signature 'mcnebula,ANY,ANY,ANY,ANY'
compute_spectral_similarity(x, within_nebula, recompute, sp1, sp2)

## S4 method for signature
## 'missing,missing,missing,lightSpectrum,lightSpectrum'
compute_spectral_similarity(sp1, sp2)

## S4 method for signature 'missing,missing,missing,data.frame,data.frame'
compute_spectral_similarity(sp1, sp2)

## S4 method for signature 'mcnebula,logical,logical,missing,missing'
compute_spectral_similarity(x, within_nebula, recompute)
```

Arguments

`x`	...
`within_nebula`	...
`recompute`	...
`sp1`	...
`sp2`	...

Details

...

...

Value

...

Examples

```
## Not run:
compute_spectral_similarity(...)

## End(Not run)
```

---

create_child_layouts-methods

...

Description

...

`create_child_layouts()`: get the function for generating default parameters for the method `create_child_layouts`.

...

`create_child_layouts(x, ...)`: use the default parameters whatever 'missing' while performing the method `create_child_layouts`.

Usage

```
## S4 method for signature
## 'missing,missing,missing,missing,missing,missing,missing'
create_child_layouts()

## S4 method for signature 'mcnebula,ANY,ANY,ANY,ANY,ANY,ANY'
create_child_layouts(
  x,
  ggraph_layouts,
  seeds,
  grid_layout,
  viewports,
  panel_viewport,
  legend_viewport
)
```

Arguments

x	...
ggraph_layouts	...
seeds	...

`grid_layout`	...
`viewports`	...
`panel_viewport`	...
`legend_viewport`	...

Details

...

...

Value

...

Examples

```
## Not run:
create_child_layouts(...)

## End(Not run)
```

---

create_child_nebulae-methods
...

Description

...

`create_child_nebulae()`: get the default parameters for the method `create_child_nebulae`.

`create_child_nebulae(x, ...)`: use the default parameters whatever 'missing' while performing the method `create_child_nebulae`.

...

Usage

```
## S4 method for signature 'missing,missing,missing,missing'
create_child_nebulae()

## S4 method for signature 'mcnebula,ANY,ANY,ANY'
create_child_nebulae(x, edge_cutoff, max_edge_number, use_tracer)

## S4 method for signature 'mcnebula,numeric,numeric,logical'
create_child_nebulae(x, edge_cutoff, max_edge_number, use_tracer)
```

Arguments

`x`	...
`edge_cutoff`	...
`max_edge_number`	...

Details

...

...

Value

...

Examples

```
## Not run:
create_child_nebulae(...)

## End(Not run)
```

`create_features_annotation-methods`

merge annotation for 'features'

Description

According to `specific_candidate(object)` data, merge the latest filtered chemical formulae annotation, structural annotation. The ion mass and retention time for each 'feature' would also be gathered. User can also pass custom annotation for each 'feature', as long as the 'data.frame' with column of '.features_id'.

Usage

```
## S4 method for signature 'mcnebula,data.frame,numeric'
create_features_annotation(x, extra_data, column)

## S4 method for signature 'mcnebula,data.frame,missing'
create_features_annotation(x, extra_data)

## S4 method for signature 'mcnebula,missing,missing'
create_features_annotation(x)
```

Arguments

`x`	mcnebula object.
`extra_data`	data.frame.
`column`	numeric(1). If name of columns not contain ".features_id", used to specify ID column for 'features'.

Details

The 'features_annotation' data created from:

- The 'specific_candidate' data: `specific_candidate(object)`
- The filtered chemical formula data: `latest(object, subscript = ".f2_formula")`
- The filtered structural data: `latest(object, subscript = ".f3_fingerid")`
- The ion mass and retention time (m/z and RT): latest(x, "project_dataset", ".f2_info")

The last would be collated via: `collate_data(object, subscript = ".f2_info")`

Examples

```
## Not run:
create_features_annotation(...)

## End(Not run)
```

---

create_hierarchy-methods
...

Description

...

...

`create_hierarchy()`: get the default parameters for the method `create_hierarchy`.

`create_hierarchy(x, ...)`: use the default parameters whatever 'missing' while performing the method `create_hierarchy`.

...

Usage

```
## S4 method for signature 'missing,missing'
create_hierarchy()

## S4 method for signature 'mcnebula,ANY'
create_hierarchy(x, fun_organize)

## S4 method for signature 'mcnebula,`function`'
create_hierarchy(x, fun_organize)
```

Arguments

`x`	...
`fun_organize`	...

Details

...

...

...

Value

...

Examples

```
## Not run:
create_hierarchy(...)

## End(Not run)
```

`create_nebula_index,missing,missing-method`

...

Description

`create_nebula_index()`: get the default parameters for the method `create_nebula_index`.

`create_nebula_index(x, ...)`: use the default parameters whatever 'missing' while performing the method `create_nebula_index`.

...

Usage

```
## S4 method for signature 'missing,missing'
create_nebula_index()

## S4 method for signature 'mcnebula,ANY'
create_nebula_index(x, force)

## S4 method for signature 'mcnebula,logical'
create_nebula_index(x, force)
```

Arguments

`x`	...
`force`	...

Details

...

Value

...

Examples

```
## Not run:
create_nebula_index(...)

## End(Not run)
```

---

create_parent_layout-methods
...

Description

...

`create_parent_layout()`: get the default parameters for the method `create_parent_layout`.

`create_parent_layout(x, ...)`: use the default parameters whatever 'missing' while performing the method `create_parent_layout`.

...

Usage

```
## S4 method for signature 'missing,missing,missing'
create_parent_layout()

## S4 method for signature 'mcnebula,ANY,ANY'
create_parent_layout(x, ggraph_layout, seed)

## S4 method for signature 'mcnebula,character,numeric'
create_parent_layout(x, ggraph_layout, seed)
```

Arguments

`x`	...
`ggraph_layout`	...
`seed`	...

Details

...

...

Value

...

Examples

```
## Not run:
create_parent_layout(...)

## End(Not run)
```

`create_parent_nebula-methods`
...

Description

...

`create_parent_nebula()`: get the default parameters for the method `create_parent_nebula`.

`create_parent_nebula(x, ...)`: use the default parameters whatever 'missing' while performing the method `create_parent_nebula`.

...

Usage

```
## S4 method for signature 'mcnebula,missing,missing'
create_parent_nebula(x)

## S4 method for signature 'mcnebula,numeric,missing'
create_parent_nebula(x, edge_cutoff)

## S4 method for signature 'mcnebula,numeric,logical'
create_parent_nebula(x, edge_cutoff, remove_isolate)
```

Arguments

`x`	...
`edge_cutoff`	...
`remove_isolate`	...

Details

...

...

Value

...

Examples

```
## Not run:
create_parent_nebula(...)

## End(Not run)
```

---

create_reference-methods

Establish 'specific candidate' for each 'feature'

Description

According to the filtered data, whether obtained by `filter_formula()`, `filter_structure()` or `filter_ppcp()`, establishing specific candidate of each 'feature' for subsequent data filtering. This step is an important intermediate link for the three part of data filtering, makes the final filtered results of chemical formula, structure and classification consistent.

Usage

```
## S4 method for signature 'mcnebula,ANY,ANY,ANY,ANY,logical,ANY'
create_reference(x, from, subscript, data, columns, fill, MoreArgs)

## S4 method for signature
## 'mcnebula,missing,missing,missing,missing,missing,missing'
create_reference(x)

## S4 method for signature
## 'mcnebula,character,missing,missing,missing,missing,missing'
create_reference(x, from)

## S4 method for signature
## 'mcnebula,missing,character,missing,missing,missing,missing'
create_reference(x, subscript)

## S4 method for signature
## 'mcnebula,missing,missing,data.frame,character,missing,missing'
create_reference(x, data, columns)

## S4 method for signature
## 'mcnebula,missing,missing,data.frame,integer,missing,missing'
create_reference(x, data, columns)

## S4 method for signature
## 'mcnebula,missing,missing,data.frame,missing,missing,missing'
create_reference(x, data)
```

Arguments

`x`	mcnebula object.
`from`	character(1). "structure", "formula" or "ppcp".
`subscript`	character(1). ".f3_fingerid", ".f2_formula" or ".f3_canopus". See subscript.
`data`	data.frame. An external channel for user to specify candidate customarily. Normally not used.
`columns`	character(2) or numeric(2). Specify the key columns in the parameter of data. Normally not used.
`fill`	logical. If `TRUE`, run post modification. Run filter_formula(object), and use its results to fill the data `specific_candidate` for 'features' without specified top candidate. Only useful when the data `specific_candidate` were based on scores of chemical structure or classes, as for some 'features' there may be no chemical structural or classified candidates but candidates for chemical formula.
`MoreArgs`	list. Used only `fill` = `T`. Parameters passed to `filter_formula()`.

Details

Establish reference upon top candidate Suppose we predicted a potential compound represented by LC-MS/MS spectrum, and obtained the candidates of chemical molecular formula, structure and chemical class. These candidates include both positive and negative results: for chemical molecular formula and chemical structure, the positive prediction was unique; for chemical class, multiple positive predictions that belong to various classification were involved. We did not know the exact negative and positive. Normally, we ranked and filtered these according to the scores. There were numerious scores, for isotopes, for mass error, for structural similarity, for chemical classes... Which score selected to rank candidates depends on the purpose of research. Such as:

- To find out the chemical structure mostly be positive, ranking the candidates by structural score.
- To determine whether the potential compound may be of a certain chemical classes, ranking the candidates by the classified score.

Ether by `filter_formula()`, `filter_structure()` or `filter_ppcp()`, the candidate with top score can be obtained. However, for the three module (formula, structure, classes), sometimes thier top score candidates were not in line with each other. That is, thier top score towards different chemical molecular formulas. To find out the corresponding data in other modules, `create_reference` should be performed to establish the 'specific_candidate' for subsequent filtering.

Examples

```
## Not run:
create_reference(...)

## End(Not run)
```

create_stardust_classes-methods

'Inner' filter for PPCP data for each 'feature'

Description

Perform 'inner' filter for PPCP (posterior probability of classification prediction) data of each 'feature'. Run after `create_reference()`. Standby for next step `cross_filter_stardust()`.

`create_stardust_classes()`: get the default parameters for the method `create_stardust_classes`.

`create_stardust_classes(x, ...)`: use the default parameters whatever 'missing' while performing the method `create_stardust_classes`.

Usage

```
## S4 method for signature 'missing,missing,missing,missing,missing'
create_stardust_classes()

## S4 method for signature 'mcnebula,ANY,ANY,ANY,ANY'
create_stardust_classes(
  x,
  pp.threshold,
  hierarchy_priority,
  position_isomerism,
  inherit_dataset
)

## S4 method for signature 'mcnebula,numeric,numeric,logical,logical'
create_stardust_classes(
  x,
  pp.threshold,
  hierarchy_priority,
  position_isomerism,
  inherit_dataset
)
```

Arguments

`x`	mcnebula object.
`pp.threshold`	numeric(1) Threshold for PPCP. `pp.threshold = 0.5` may work well.
`hierarchy_priority`	numeric. The specified hierarchy of classes to retain. The other hierarchy would be filtered out. The hierarchy: • n: ... • 5: Classes of Level 5. • 4: Classes of Subclass.

- 3: Classes of Class.
- 2: Classes of Super Class.
- ...

`position_isomerism`
: logical. If `TRUE`, use pattern match to filter out all classes names contains Arabic numerals. Generally, these classes describe about the position of chemical functional group, which were too subtle for machine to predict from LC-MS/MS spectrum.

`inherit_dataset`
: logical. If `TRUE`, use latest PPCP data formed by `filter_ppcp()`. i.e., data of:

  - latest(x, subscript = ".f3_canopus")

  Else, run `filter_ppcp()`.

Details

The PPCP data for each 'feature' contains the prediction of thousands of classes for the potential compound (even if the chemical structure was unknown). See `http://www.nature.com/articles/s41587-020-0740-8` for details about the prediction. The data contains attributes of:

- `class.name`: name of classes.
- `pp.value`: value of posterior probability.
- `hierarchy`: hierarchy of classes in the taxonomy. See `https://jcheminf.biomedcentral.com/articles/10.1186/s13321-016-0174-y` for details about hierarchy and taxonomy of chemical classification.
- ...

The method `create_stardust_classes()` use these inner attributes to filter classes candidates for each 'feature', which standby for next method `cross_filter_stardust()`.

Examples

```
## Not run:
create_stardust_classes(...)

## End(Not run)
```

`cross_filter_stardust-methods`
...

Description

...

`cross_filter_stardust()`: get the default parameters for the method `cross_filter_stardust`.

`cross_filter_stardust(x, ...)`: use the default parameters whatever 'missing' while performing the method `cross_filter_stardust`.

`cross_filter_stardust` include 3 parts: `cross_filter_quantity`,

...

...

...

Usage

```
## S4 method for signature 'missing'
cross_filter_stardust()

## S4 method for signature 'mcnebula'
cross_filter_stardust(
  x,
  min_number,
  max_ratio,
  types,
  cutoff,
  tolerance,
  hierarchy_range,
  identical_factor
)

## S4 method for signature 'mcnebula,numeric,numeric'
cross_filter_quantity(x, min_number, max_ratio)

## S4 method for signature 'mcnebula,character,numeric,numeric'
cross_filter_score(x, types, cutoff, tolerance)

## S4 method for signature 'mcnebula,numeric,numeric'
cross_filter_identical(x, hierarchy_range, identical_factor)
```

Arguments

`x`	...
`min_number`	...
`max_ratio`	...
`types`	...
`cutoff`	...
`tolerance`	...

`hierarchy_range`
: ...

`identical_factor`
: ...

`cross_filter_score`, `cross_filter_identical`.

Details

...

...

...

...

Value

...

...

...

Examples

```
## Not run:
cross_filter_quantity(...)

## End(Not run)
## Not run:
cross_filter_score(...)

## End(Not run)
## Not run:
cross_filter_identical(...)

## End(Not run)
```

`dataset-class` *Share slots and methods for classes inherite from VIRTUAL_dataset*

Description

This VIRTUAL class provides a slot for storing data and methods for accessing data in slot.

`dataset`, `dataset<-`: getter and setter for the `dataset` slot of the object.

Usage

```
## S4 method for signature 'ANY'
dataset(x)

## S4 replacement method for signature 'ANY'
dataset(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`dataset` list with names (subscript, imply file names).

See Also

Other datasets: `mcn_dataset-class`, `project_dataset-class`

`draw_nodes-methods` ...

Description

...

`draw_nodes()`: get the function for generating default parameters for the method `draw_nodes`.

`draw_nodes(x, ...)`: use the default parameters whatever 'missing' while performing the method `draw_nodes`.

...

`show_node()`: get the default parameters for the method `show_node`.

...

`show_node(x, ...)`: use the default parameters whatever 'missing' while performing the method `show_node`.

FUNCTION_DESCRIPTION

Usage

```
## S4 method for signature
## 'missing,missing,missing,missing,missing,missing,missing'
draw_nodes()

## S4 method for signature 'mcnebula,character,ANY,ANY,ANY,ANY,ANY'
draw_nodes(
  x,
  nebula_name,
  nodes_color,
  add_id_text,
  add_structure,
  add_ppcp,
  add_ration
)

## S4 method for signature
## 'mcnebula,character,character,logical,logical,logical,logical'
draw_nodes(
  x,
  nebula_name,
  nodes_color,
  add_id_text,
  add_structure,
  add_ppcp,
  add_ration
)

## S4 method for signature 'missing,missing,missing,missing'
show_node()

## S4 method for signature 'ANY,character,ANY,ANY'
show_node(x, .features_id, panel_viewport, legend_viewport)

ggset_activate_nodes(
  x,
  .features_id,
  nodes_color = "#FFF9F2",
  add_ppcp = T,
  add_ration = T
)
```

Arguments

`x`	...
`nebula_name`	...
`nodes_color`	...
`add_id_text`	...
`add_structure`	...
`add_ppcp`	...
`add_ration`	...
`.features_id`	...
`panel_viewport`	...
`legend_viewport`	...

Details

...

...

...

DETAILS

Value

...

...

OUTPUT_DESCRIPTION

Examples

```
## Not run:
draw_nodes(...)

## End(Not run)
## Not run:
show_node(...)

## End(Not run)
## Not run:
if(interactive()){
 #EXAMPLE1
 }

## End(Not run)
```

---

draw_structures-methods

...

Description

...

...

...

Usage

```
## S4 method for signature 'mcnebula,character'
draw_structures(x, nebula_name)

## S4 method for signature 'ANY,character'
show_structure(x, .features_id)
```

Arguments

x	...
nebula_name	...
.features_id	...

Details

...

...

Value

...

Examples

```
## Not run:
draw_structures(...)

## End(Not run)
## Not run:
show_structure(...)

## End(Not run)
```

`export-class` *Share slots and methods for classes inherite from VIRTUAL_export*

Description

This VIRTUAL class provides slots for recording export path and export name of attributes.

`export_name`, `export_name<-`: getter and setter for the `export_name` slot of the object.

`export_path`, `export_path<-`: getter and setter for the `export_path` slot of the object.

Usage

```
## S4 method for signature 'ANY'
export_name(x)

## S4 replacement method for signature 'ANY'
export_name(x) <- value

## S4 method for signature 'ANY'
export_path(x)

## S4 replacement method for signature 'ANY'
export_path(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`export_path`	character(1). The export directory path.
`export_name`	character with names. While export, the attribute name will be converted to the value.

`filter_formula-methods` *Collate and filter candidates of chemical formula for each 'feature'*

Description

This methods provide an approach to collate and filter chemical formula candidates data in baches for each 'feature'.

`filter_formula()`: get the default parameters for the method `filter_formula`.

`filter_formula(x, ...)`: use the default parameters whatever 'missing' while performing the method `filter_formula`.

Usage

```
## S4 method for signature 'missing,missing,missing'
filter_formula()

## S4 method for signature 'mcnebula,ANY,ANY'
filter_formula(x, fun_filter, ..., by_reference)

## S4 method for signature 'mcnebula,`function`,logical'
filter_formula(x, fun_filter, ..., by_reference)
```

Arguments

`x`	mcnebula object.
`fun_filter`	function. Used to filter data.frame. The function would run for candidates data (data.frame) for each 'features'. Such as: • `lapply(split(all_data, ~.features_id), fun_filter, ...)`. This parameter provides an elegant and flexible way to filter data. Users can pass function `dplyr::filter()` to specify any attributes condition to filter the data.
`...`	Other parameters passed to the function `fun_filter`.
`by_reference`	logical. Use `specific_candidate(object)` data to filter candidates data. See `create_reference()`.

Details

In SIRIUS project directory, if the computation job has done, each 'feature' has multiple prediction candidates whether for chemical formula, structure, or classification. This method provides an approach to collate and filter these data in baches. See MCnebula2 for details of chemical formula, structure and classification.

Examples

```
## Not run:
filter_formula(...)

## End(Not run)
```

`filter_ppcp-methods` *Collate and filter candidates of chemical classification for each 'feature'*

Description

This methods provide an approach to collate and filter chemical classification candidates data in baches for each 'feature'.

`filter_ppcp()`: get the default parameters for the method `filter_ppcp`.

`filter_ppcp(x, ...)`: use the default parameters whatever 'missing' while performing the method `filter_ppcp`.

Usage

```
## S4 method for signature 'missing,missing,missing'
filter_ppcp()

## S4 method for signature 'mcnebula,ANY,ANY'
filter_ppcp(x, fun_filter, ..., by_reference)

## S4 method for signature 'mcnebula,`function`,logical'
filter_ppcp(x, fun_filter, ..., by_reference)
```

Arguments

`x`	mcnebula object.
`fun_filter`	function. Used to filter data.frame. The function would run for candidates data (data.frame) for each 'features'. Such as: • `lapply(split(all_data, ~.features_id), fun_filter, ...)`. This parameter provides an elegant and flexible way to filter data. Users can pass function `dplyr::filter()` to specify any attributes condition to filter the data.

`...`	Other parameters passed to the function `fun_filter`.
`by_reference`	logical. Use `specific_candidate(object)` data to filter candidates data. See `create_reference()`.

Details

Filter for PPCP (posterior probability of classification prediction) data. See details about classification prediction for compounds: `http://www.nature.com/articles/s41587-020-0740-8`. See other details in `filter_formula()`.

Examples

```
## Not run:
filter_ppcp(...)

## End(Not run)
```

filter_structure-methods

Collate and filter candidates of chemical structure for each 'feature'

Description

This methods provide an approach to collate and filter chemical structure candidates data in baches for each 'feature'.

`filter_structure()`: get the default parameters for the method `filter_structure`.

`filter_structure(x, ...)`: use the default parameters whatever 'missing' while performing the method `filter_structure`.

Usage

```
## S4 method for signature 'missing,missing,missing'
filter_structure()

## S4 method for signature 'mcnebula,ANY,ANY'
filter_structure(x, fun_filter, ..., by_reference)

## S4 method for signature 'mcnebula,`function`,logical'
filter_structure(x, fun_filter, ..., by_reference)
```

Arguments

`x`	mcnebula object.
`fun_filter`	function. Used to filter data.frame. The function would run for candidates data (data.frame) for each 'features'. Such as: • `lapply(split(all_data, ~.features_id), fun_filter, ...)`.

	This parameter provides an elegant and flexible way to filter data. Users can pass function `dplyr::filter()` to specify any attributes condition to filter the data.
`...`	Other parameters passed to the function `fun_filter`.
`by_reference`	logical. Use `specific_candidate(object)` data to filter candidates data. See `create_reference()`.

Details

See details in `filter_formula()`.

Examples

```
## Not run:
filter_structure(...)

## End(Not run)
```

`fun_modify` ...

Description

...

`modify_default_child`: ...

`modify_set_labs_and_unify_scale_limits`: ...

`modify_annotate_child`: ...

`modify_rm_legend`: ...

`modify_set_margin`: ...

`modify_unify_scale_limits`: ...

`modify_set_labs`: ...

Usage

```
modify_default_child(ggset, x)

modify_set_labs_and_unify_scale_limits(ggset, x)

modify_annotate_child(ggset, x)

modify_rm_legend(ggset)

modify_set_margin(ggset, margin = grid::unit(rep(-8, 4), "lines"))

modify_unify_scale_limits(ggset, x)

modify_set_labs(ggset, x)
```

Arguments

`ggset`	...
`x`	...

Details

...

`ggset-class` *Management for 'ggplot' visualzation*

Description

Let each packed "ggplot2" function (packed as command object) into layers in sequence, allowing post modifications programmatically and visualizing as "ggplot2" plot at any time.

`show_layers`: show functions and parameters in layers with a pretty and readable form.

`new_ggset`: Simplified creation of ggset object.

`mutate_layer`: Pass new parameters or modify pre-existing parameters to the packed function.

`call_command`: plot as 'ggplot' object.

Usage

```
## S4 method for signature 'ggset'
show_layers(x)

## S4 method for signature 'ANY'
new_ggset(...)

## S4 method for signature 'ggset,numeric'
mutate_layer(x, layer, ...)

## S4 method for signature 'ggset,character'
mutate_layer(x, layer, ...)

## S4 method for signature 'ggset'
call_command(x)
```

Arguments

`x`	ggset object
`...`	parameters passed to the layer.
`layer`	numeric(1) or character(1). If "character", the name must be unique in slot `layers`.

Slots

`layers` list with names. Each element of list must be a command object packed 'ggplot2' function and its args.

See Also

Other layerSets: `layerSet-class`, `report-class`

Other call_commands: `code_block-class`, `command-class`, `report-class`, `section-class`

Examples

```
## Not run:
show_layers(...)

## End(Not run)
## Not run:
new_ggset(...)

## End(Not run)
## Not run:
mutate_layer(...)

## End(Not run)
```

`history_rblock-methods`

...

Description

...

`history_rblock()`: get the default parameters for the method `history_rblock`.

`history_rblock(x, ...)`: use the default parameters whatever 'missing' while performing the method `history_rblock`.

...

Usage

```
## S4 method for signature 'missing,missing,missing,missing'
history_rblock()

## S4 method for signature 'numeric,ANY,ANY,ANY'
history_rblock(nrow, pattern_start, pattern_end, exclude)

## S4 method for signature 'numeric,missing,missing,numeric'
history_rblock(nrow, exclude)
```

```
## S4 method for signature 'missing,character,character,ANY'
history_rblock(pattern_start, pattern_end, exclude)
```

Arguments

`nrow`	...
`pattern_start`	...
`pattern_end`	...
`exclude`	...

Details

...

...

Value

...

See Also

`report-class`, `code_block-class`...

Examples

```
## Not run:
history_rblock(...)

## End(Not run)
```

include_figure-methods

...

Description

...

...

Usage

```
## S4 method for signature 'character,character,character'
include_figure(file, name, caption)
```

Arguments

`file`	...
`name`	...
`caption`	...

Details

...

...

Value

...

See Also

`report-class`, `code_block-class`...

Examples

```
## Not run:
include_figure(...)

## End(Not run)
```

`include_table-methods` ...

Description

...

...

Usage

```
## S4 method for signature 'data.frame,character,character'
include_table(data, name, caption)
```

Arguments

`data`	...
`name`	...
`caption`	...

Details

...

...

Value

...

Examples

```
## Not run:
include_table(...)

## End(Not run)
```

---

initialize_mcnebula-methods
...

Description

...

...

...

Usage

```
## S4 method for signature 'mcnebula,ANY'
initialize_mcnebula(x, sirius_version, sirius_project, output_directory)

## S4 method for signature 'melody,ANY'
initialize_mcnebula(x)
```

Arguments

`x`	melody object.
`sirius_version`	character. e.g., "sirius.v4".
`sirius_project`	character. The path of SIRIUS project space.
`output_directory`	character. The path for output.

Details

...

...

...

Value

mcnebula object.

melody object.

See Also

`ggsci::pal_simpsons()`, `ggsci::pal_igv()`, `ggsci::pal_ucscgb()`, `ggsci::pal_d3()`...

Examples

```
## Not run:
initialize_mcnebula(...)

## End(Not run)
## Not run:
initialize_mcnebula(...)

## End(Not run)
```

`layerSet-class` *Share slots and methods for classes inherite from VIRTUAL_layerSet*

Description

This VIRTUAL class provides: slot `layers` for storing hierarchical data; and methods for modify slot `layers`.

`layers`, `layers<-`: getter and setter for the `layers` slot of the object.

`add_layers`: add extra "layer" into slot `layers`.

`delete_layers`: delete "layer" in slot `layers`.

`move_layers`: change the order of "layer" in slot `layers`.

Usage

```
## S4 method for signature 'layerSet'
layers(x)

## S4 replacement method for signature 'layerSet'
layers(x) <- value

## S4 method for signature 'layerSet'
show(object)

## S4 method for signature 'layerSet'
add_layers(x, ...)

## S4 method for signature 'layerSet,numeric'
```

```
delete_layers(x, layers)

## S4 method for signature 'layerSet,numeric,numeric'
move_layers(x, from, to)
```

Arguments

`x`	object contains slot `layers`.
`value`	The value for the slot.
`...`	extra "layer".
`layers`	numeric. The specified "layer" in slot `layers`.
`from`	sequence (sequence in list) of "layer" move from.
`to`	sequence (sequence in list) of "layer" move to.

Slots

`layers` list with names.

See Also

Other layerSets: `ggset-class`, `report-class`

Examples

```
## Not run:
add_layers(...)

## End(Not run)
## Not run:
delete_layers(...)

## End(Not run)
## Not run:
move_layers(...)

## End(Not run)
```

`mcnebula-class` *Overall object class of MCnebula2*

Description

For analysis of MCnebula2, all data stored in this class object, all main methods performed with this object.

`latest(x, slot, subscript)`: get the data in slot (`mcn_dataset(object)` or `prject_dataset(object)`) and format as ’tbl’.

`latest()`: get the default parameters for the method `latest`.

`latest(x, ...)`: use the default parameters whatever 'missing' while performing the method `latest`.

`creation_time`, `creation_time<-`: getter and setter for the `creation_time` slot of the object.

`ion_mode`, `ion_mode<-`: getter and setter for the `ion_mode` slot of the object.

`palette_set`, `palette_gradient`, `palette_stat`, `palette_col`: fast channel to obtain the downstream slot. For `palette_set`, e.g., getter for the `palette_set` slot in sub-object of `melody` slot of the object. Equals:

- `palette_set(melody(object))`
- `palette_set(object)`.

`reference`: fast channel to obtain the downstream slot, getter for the `reference` slot in sub-object of `mcn_dataset` slot of the object. Equals:

- `reference(mcn_dataset(object))`
- `reference(object)`

`specific_candidate`, `hierarchy`, `stardust_classes`, `nebula_index`, `spectral_similarity`, `features_annotation`, `features_quantification`, `sample_metadata`: fast channel to obtain data (mostly 'tbl' or 'data.frame') inside the downstream slot ('list'). e.g., getter for the data named `specific_candidate` in `reference` slot (a 'list') in sub-object of `mcn_dataset` slot of the object. Equals:

- `reference(mcn_dataset(object))$specific_candidate`
- `specific_candidate(object)`.

`spectral_similarity<-`, `features_quantification<-`, `sample_metadata<-`: fast channel to replace data (mostly 'tbl' or 'data.frame') inside the downstream slot ('list'). e.g., setter for the data named `spectral_similarity` in `reference` slot (a 'list') in sub-object of `mcn_dataset` slot of the object. Similar:

- `reference(mcn_dataset(object))$spectral_similarity<-`
- `spectral_similarity(object)<-`.

But the latter not only replace and also validate.

`classification`: fast channel to obtain data deeply inside the downstream slot ('list'), getter for the data named `".canopus"` in `dataset` slot (a 'list') in sub-object of `project_dataset` slot of the object. Equals:

- `tibble::as_tibble(entity(dataset(project_dataset(object))$.canopus))`
- `classification(object)`.

Usage

```
## S4 method for signature 'mcnebula'
show(object)

## S4 method for signature 'mcnebula,character,ANY'
latest(x, slot, subscript)

## S4 method for signature 'missing,missing,missing'
latest()

## S4 method for signature 'mcnebula,ANY,ANY'
latest(x, slot, subscript)

## S4 method for signature 'mcnebula'
creation_time(x)

## S4 replacement method for signature 'mcnebula'
creation_time(x) <- value

## S4 method for signature 'mcnebula'
ion_mode(x)

## S4 replacement method for signature 'mcnebula'
ion_mode(x) <- value

## S4 method for signature 'mcnebula'
palette_set(x)

## S4 method for signature 'mcnebula'
palette_gradient(x)

## S4 method for signature 'mcnebula'
palette_stat(x)

## S4 method for signature 'mcnebula'
palette_col(x)

## S4 method for signature 'mcnebula'
palette_label(x)

## S4 method for signature 'mcnebula'
reference(x)

## S4 method for signature 'mcnebula'
specific_candidate(x)

## S4 method for signature 'mcnebula'
hierarchy(x)
```

```
## S4 method for signature 'mcnebula'
stardust_classes(x)

## S4 method for signature 'mcnebula'
nebula_index(x)

## S4 method for signature 'mcnebula'
spectral_similarity(x)

## S4 replacement method for signature 'mcnebula'
spectral_similarity(x) <- value

## S4 method for signature 'mcnebula'
features_annotation(x)

## S4 method for signature 'mcnebula'
features_quantification(x)

## S4 replacement method for signature 'mcnebula'
features_quantification(x) <- value

## S4 method for signature 'mcnebula'
sample_metadata(x)

## S4 replacement method for signature 'mcnebula'
sample_metadata(x) <- value

## S4 method for signature 'mcnebula'
classification(x)
```

Arguments

`x`	mcnebula object
`slot`	Character. Slot name.
`subscript`	numeric or character. The sequence or name for dataset in the ’list’.
`value`	The value for the slot.

Slots

`creation_time` character(1).

`ion_mode` character(1).

`melody` melody object.

`mcn_dataset` mcn_dataset object.

`statistic_set` statistic_set object.

`...` Slots inherit from project, nebula, export.

See Also

tibble::as_tibble()

Other nebulae: nebula-class

Other latests: mcn_dataset-class, msframe-class, project_dataset-class, project_metadata-class

Other subscripts: msframe-class, project_conformation-class, subscript-class

Examples

```
## Not run:
new('mcnebula', ...)
mcnebula()

## End(Not run)
## Not run:
latest(x)
latest(x, "project_dataset")
latest(x, "mcn_dataset")

## End(Not run)
```

---

mcn_dataset-class *Store processed data*

Description

This is a class object used to store filtered data and formated data. These data would be used for further analysis or visualization.

`mcn_dataset`, `mcn_dataset<-`: getter and setter for the `mcn_dataset` slot of the object.

`latest`: get the first data in `dataset` slot and format as "tbl". Equals:

- `latest(object)`
- `tibble::as_tibble(entity(dataset(x)[[1]]))`.

`extract_mcnset`: For fast extract data in object which containing `mcn_dataset` slot. Normally not used.

Usage

```
## S4 method for signature 'ANY'
mcn_dataset(x)

## S4 replacement method for signature 'ANY'
mcn_dataset(x) <- value

## S4 method for signature 'mcn_dataset,ANY,ANY'
latest(x)
```

```
## S4 method for signature 'ANY,character'
extract_mcnset(x, subscript)
```

Arguments

`value`	The value for the slot.
`subscript`	See subscript

Slots

`dataset` list with names of subscript. Store preliminary filtered data.

`reference` list with names of standard names. Store formated data, which is useful reference for further analysis or visualization.

`backtrack` list with names. Recovery stations halfway through data processing.

See Also

dataset

subscript

Other datasets: `dataset-class`, `project_dataset-class`

Other latests: `mcnebula-class`, `msframe-class`, `project_dataset-class`, `project_metadata-class`

Examples

```
## Not run:
new('mcn_dataset', ...)

## End(Not run)
## Not run:
latest(...)

## End(Not run)
## Not run:
extract_mcnset(...)

## End(Not run)
```

`melody-class` *Mutiple color palette in hexadecimal code*

Description

This is a class object store Hex color used for visualization. In default (use `initialize_mcnebula()` to initialize the object), these these Hex color in each palette were get from package `ggsci`. Most of these palette in this package would passed to ggplot2::scale_fill_manual for filling color. So, let these Hex color with names may work well to specify target.

`melody`, `melody<-`: getter and setter for the `melody` slot of the object.

`palette_set`, `palette_set<-`: getter and setter for the `palette_set` slot of the object.

`palette_gradient`, `palette_gradient<-`: getter and setter for the `palette_gradient` slot of the object.

`palette_stat`, `palette_stat<-`: getter and setter for the `palette_stat` slot of the object.

`palette_col`, `palette_col<-`: getter and setter for the `palette_col` slot of the object.

`palette_label`, `palette_label<-`: getter and setter for the `palette_label` slot of the object.

Usage

```
## S4 method for signature 'melody'
show(object)

## S4 method for signature 'ANY'
melody(x)

## S4 replacement method for signature 'ANY'
melody(x) <- value

## S4 method for signature 'melody'
palette_set(x)

## S4 replacement method for signature 'melody'
palette_set(x) <- value

## S4 method for signature 'melody'
palette_gradient(x)

## S4 replacement method for signature 'melody'
palette_gradient(x) <- value

## S4 method for signature 'melody'
palette_stat(x)

## S4 replacement method for signature 'melody'
palette_stat(x) <- value

## S4 method for signature 'melody'
palette_col(x)

## S4 replacement method for signature 'melody'
```

```
palette_col(x) <- value

## S4 method for signature 'melody'
palette_label(x)

## S4 replacement method for signature 'melody'
palette_label(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`palette_set` character with names or not. Hex color.

`palette_gradient` character with names or not. Hex color.

`palette_stat` character with names or not. Hex color.

`palette_col` character with names or not. Hex color.

`palette_label` character with names or not. Hex color.

See Also

`ggsci::pal_simpsons()`, `ggsci::pal_igv()`, `ggsci::pal_ucscgb()`, `ggsci::pal_d3()`...

Examples

```
## Not run:
new('melody', ...)

## End(Not run)
```

`msframe-class` *format and filter table data*

Description

Class for table data manipulation inside this package.

`msframe`, `msframe<-`: getter and setter for the `msframe` slot of the object.

`latest`: get data inside `entity(object)` and format as 'tbl'.

`entity`, `entity<-`: getter and setter for the `entity` slot of the object.

`format_msframe`:

`filter_msframe`: filter data in slot `entity` (data.frame).

Usage

```
## S4 method for signature 'msframe'
show(object)

## S4 method for signature 'ANY'
msframe(x)

## S4 replacement method for signature 'ANY'
msframe(x) <- value

## S4 method for signature 'msframe,ANY,ANY'
latest(x)

## S4 method for signature 'msframe'
entity(x)

## S4 replacement method for signature 'msframe'
entity(x) <- value

## S4 method for signature 'msframe,missing,missing,missing,missing,`function`'
format_msframe(x, fun_format)

## S4 method for signature
## 'data.frame,missing,missing,missing,missing,`function`'
format_msframe(x, fun_format)

## S4 method for signature
## 'msframe,character,missing,character,missing,missing'
format_msframe(x, names, types)

## S4 method for signature 'msframe,missing,missing,missing,missing,missing'
format_msframe(x)

## S4 method for signature
## 'msframe,missing,`function`,missing,`function`,missing'
format_msframe(x, fun_names, fun_types)

## S4 method for signature 'msframe,`function`,missing'
filter_msframe(x, fun_filter, f, ...)

## S4 method for signature 'msframe,`function`,formula'
filter_msframe(x, fun_filter, f, ...)
```

Arguments

`x`	msframe object.
`value`	The value for the slot.
`fun_format`	function to format slot `entity`. e.g., `MCnebula2:::.format_msframe()`

`names`	character with names. e.g., c(tani.score = "tanimotoSimilarity", mol.formula = "molecularFormula").
`types`	character with names. e.g., c(tani.score = "numeric", mol.formula = "character").
`fun_names`	function to get names. e.g., `MCnebula2:::.get_attribute_name_sirius.v4()`
`fun_types`	function to get types. e.g., `MCnebula2:::.get_attribute_type_sirius.v4()`
`fun_filter`	function used to filter the slot `entity` (data.frame). e.g., `dplyr::filter()`, `head()`.
`f`	formula passed to `split()`.
`...`	extra parameter passed to fun_filter.

Slots

`entity` data.frame.

`subscript` character(1). See subscript.

Note

The class is not for normal use of the package.

See Also

`tibble::as_tibble()`

Other subscripts: `mcnebula-class`, `project_conformation-class`, `subscript-class`

Other latests: `mcn_dataset-class`, `mcnebula-class`, `project_dataset-class`, `project_metadata-class`

Examples

```
## Not run:
new('msframe', ...)

## End(Not run)
## Not run:
latest(...)

## End(Not run)
## Not run:
format_msframe(...)

## End(Not run)
## Not run:
filter_msframe(...)

## End(Not run)
```

`nebula-class` *Visualization component of chemical nebulae/nebula*

Description

This class store multiple components for visualization.

`parent_nebula`: Store data for visualization of parent-nebula.

`child_nebulae`: store data for visualization of child-nebulae.

`parent_nebula`, `parent_nebula<-`: getter and setter for the `parent_nebula` slot of the object.

`child_nebulae`, `child_nebulae<-`: getter and setter for the `child_nebulae` slot of the object.

`igraph`, `igraph<-`: getter and setter for the `igraph` slot of the object.

`tbl_graph`, `tbl_graph<-`: getter and setter for the `tbl_graph` slot of the object.

`layout_ggraph`, `layout_ggraph<-`: getter and setter for the `layout_ggraph` slot of the object.

`grid_layout`, `grid_layout<-`: getter and setter for the `grid_layout` slot of the object.

`viewports`, `viewports<-`: getter and setter for the `viewports` slot of the object.

`ggset`, `ggset<-`: getter and setter for the `ggset` slot of the object.

`panel_viewport`, `panel_viewport<-`: getter and setter for the `panel_viewport` slot of the object.

`legend_viewport`, `legend_viewport<-`: getter and setter for the `legend_viewport` slot of the object.

`structures_grob`, `structures_grob<-`: getter and setter for the `structures_grob` slot of the object.

`nodes_ggset`, `nodes_ggset<-`: getter and setter for the `nodes_ggset` slot of the object.

`nodes_grob`, `nodes_grob<-`: getter and setter for the `nodes_grob` slot of the object.

`ppcp_data`, `ppcp_data<-`: getter and setter for the `ppcp_data` slot of the object.

`ration_data`, `ration_data<-`: getter and setter for the `ration_data` slot of the object.

`ggset_annotate`, `ggset_annotate<-`: getter and setter for the `ggset_annotate` slot of the object.

Usage

```
## S4 method for signature 'parent_nebula'
show(object)

## S4 method for signature 'child_nebulae'
show(object)

## S4 method for signature 'ANY'
parent_nebula(x)

## S4 replacement method for signature 'ANY'
parent_nebula(x) <- value
```

```
## S4 method for signature 'ANY'
child_nebulae(x)

## S4 replacement method for signature 'ANY'
child_nebulae(x) <- value

## S4 method for signature 'ANY'
igraph(x)

## S4 replacement method for signature 'ANY'
igraph(x) <- value

## S4 method for signature 'ANY'
tbl_graph(x)

## S4 replacement method for signature 'ANY'
tbl_graph(x) <- value

## S4 method for signature 'ANY'
layout_ggraph(x)

## S4 replacement method for signature 'ANY'
layout_ggraph(x) <- value

## S4 method for signature 'ANY'
grid_layout(x)

## S4 replacement method for signature 'ANY'
grid_layout(x) <- value

## S4 method for signature 'ANY'
viewports(x)

## S4 replacement method for signature 'ANY'
viewports(x) <- value

## S4 method for signature 'ANY'
ggset(x)

## S4 replacement method for signature 'ANY'
ggset(x) <- value

## S4 method for signature 'ANY'
panel_viewport(x)

## S4 replacement method for signature 'ANY'
panel_viewport(x) <- value
```

```
## S4 method for signature 'ANY'
legend_viewport(x)

## S4 replacement method for signature 'ANY'
legend_viewport(x) <- value

## S4 method for signature 'ANY'
structures_grob(x)

## S4 replacement method for signature 'ANY'
structures_grob(x) <- value

## S4 method for signature 'ANY'
nodes_ggset(x)

## S4 replacement method for signature 'ANY'
nodes_ggset(x) <- value

## S4 method for signature 'ANY'
nodes_grob(x)

## S4 replacement method for signature 'ANY'
nodes_grob(x) <- value

## S4 method for signature 'ANY'
ppcp_data(x)

## S4 replacement method for signature 'ANY'
ppcp_data(x) <- value

## S4 method for signature 'ANY'
ration_data(x)

## S4 replacement method for signature 'ANY'
ration_data(x) <- value

## S4 method for signature 'ANY'
ggset_annotate(x)

## S4 replacement method for signature 'ANY'
ggset_annotate(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`parent_nebula`	parent_nebula object.

`child_nebulae` child_nebulae object.

`igraph` "igraph" object or its list. See `igraph::graph_from_data_frame()`. The slot contains edges and nodes data of child-nebulae or parent-nebula. The "igraph" object can be output use `igraph::write_graph()` as ".graphml" file, which belong to a network data format that can be operated by other software such as Cytoscape (`https://cytoscape.org/`).

`tbl_graph` "tbl_graph" object or its list. See `tidygraph::as_tbl_graph()`. Converted from slot `igraph`.

`layout_ggraph` "layout_ggraph" object or its list. See `ggraph::create_layout()`. Create from slot `tbl_graph`, passed to `ggraph::ggraph()` for visualization.

`grid_layout` "layout" object. See `grid::grid.layout()`. Grid layout for position of each child-nebula to visualize.

`viewports` list with names. Each element must be "viewport" object. See `grid::viewport()`. Position for each child-nebula to visualize.

`panel_viewport` "viewport" object. See `grid::viewport()`. For visualization, the position to place overall child-nebulae.

`legend_viewport` "viewport" object. See `grid::viewport()`. For visualization, the position to place legend.

`ggset` ggset object or its list with names. Each ggset object can be visualized directly use `call_command()`.

`structures_grob` list with names. Each element is a "grob" object. See `grid::grob()`. Use `grid::grid.draw()` to visualize the chemical structure.

`nodes_ggset` list of ggset object. For drawing each node of 'features' ('features' means the detected peaks while processing LC-MS data) with annotation. Use `call_command()` to visualize the ggset.

`nodes_grob` list of "grob" object. Converted from slot `nodes_ggset` with slot `structures_grob`. Use `grid::grid.draw()` to visualize the "grob".

`ppcp_data` list with names. Each element is a data.frame. This is an annotation data of 'features' which would be visualize in nodes border as a radial bar plot. `ppcp_data`, i.e., posterior probability of classification prediction. See `filter_ppcp()`.

`ration_data` list with names. Each element is a data.frame. This is an annotation data of 'features' which would be visualize in nodes nucleus as ring plot. Generally, `ration_data` is the statistic data for samples.

`ggset_annotate` a list of ggset object. The annotated child-nebulae gathered from slot `ggset` and slot `nodes_grob`. Use `call_command()` to visualize the ggset. Be care, the object sometimes is too large that need lot of time to loading for visualization.

See Also

Other nebulae: `mcnebula-class`

Examples

```
## Not run:
new('nebula', ...)

## End(Not run)
## Not run:
new('parent_nebula', ...)

## End(Not run)
## Not run:
new('child_nebulae', ...)

## End(Not run)
```

`project-class` *Collection of Interface for extracting data from raw directory*

Description

This is a class object designed to extract files in the project directory. Its responsibility is to describe the name, path and reading method of the file under the project directory; Use these information to extract and store data.

`project_version`, `project_version<-`: getter and setter for the `project_version` slot of the object.

`project_path`, `project_path<-`: getter and setter for the `project_path` slot of the object.

`file_name`, `file_api`, `attribute_name`: fast channel to obtain the downstream slot. e.g., getter for the `file_name` slot in sub-object of `project_conformation` slot of the object. Equals:

- `file_name(project_conformation(object))`
- `file_name(object)`.

`metadata`: fast channel to obtain the downstream slot, getter for the `metadata` slot in sub-object of `project_metadata` slot of the object. Equals:

- `metadata(project_metadata(object))`
- `metadata(object)`.

`methods_read`, `methods_format`, `methods_match`: fast channel to obtain the downstream slot. e.g., getter for the `methods_read` slot in sub-object of `project_api` slot of the object. Equals:

- `methods_read(project_api(object))`
- `methods_read(object)`.

`match.candidates_id`, `match.features_id`: fast channel to obtain data (mostly 'tbl' or 'data.frame') inside the downstream slot ('list'), getter for the data named `match.candidates_id` in `methods_match` slot (a 'list') in sub-object of `project_api` slot of the object. Equals:

- `methods_match(project_api(object))$match.candidates_id`
- `match.candidates_id(object)`.

`get_upper_dir_subscript`: Get the "subscript" name of the folder.

Usage

```
## S4 method for signature 'ANY'
project_version(x)

## S4 replacement method for signature 'ANY'
project_version(x) <- value

## S4 method for signature 'ANY'
project_path(x)

## S4 replacement method for signature 'ANY'
project_path(x) <- value

## S4 method for signature 'ANY'
file_name(x)

## S4 method for signature 'ANY'
file_api(x)

## S4 method for signature 'ANY'
attribute_name(x)

## S4 method for signature 'ANY'
metadata(x)

## S4 method for signature 'ANY'
methods_read(x)

## S4 method for signature 'ANY'
methods_format(x)

## S4 method for signature 'ANY'
methods_match(x)

## S4 method for signature 'ANY'
match.candidates_id(x)

## S4 method for signature 'ANY'
match.features_id(x)

## S4 method for signature 'ANY,character,missing'
get_upper_dir_subscript(x, subscript)
```

Arguments

`x`	Maybe object of class inherit project.
`value`	The value for the slot.
`subscript`	the "subscript" name of file. See subscript.

Details

It is a collection of classes whose names start with "project_":

- project_conformation: The name, path and attribute name of the file are described.
- project_api: Functions for reading and formatting data are provided.
- project_metadata: Metadata, which records the files stored in the project directory.
- project_dataset: The extracted data is stored here.

The above class objects are coordinated into a whole through the "subscript" name (see subscript). For example, when a command (`collate_data(x, ".f3_fingerid")`) requests to extract the files of subscript of ".f3_fingerid", the data extraction module:

- from slot of `project_conformation`, get the file name (pattern string) and path of subscript of ".f3_fingerid";
- match the files under the path with the pattern string (i.e., get the metadata of the files), then stored the metadata into slot of `project_metadata`;
- from slot of `project_api`, get the functions of subscript of ".f3_fingerid";
- use these functions to read and format the data in batches;
- store the extracted data into slot of `project_dataset`.

This class is mainly designed for extracting files under the SIRIUS project directory. These files are: mainly "tables" that can be read through functions such as `read.table`; numerous and have multiple directories; need to be processed in batches. SIRIUS project may alter the name and path of internal files during version changes, which is in fact deadly for MCnebula2. To make the data extraction module of MCnebula2 free from version issues, this class object is designed to flexibly handle the extraction of internal files. Most contents need to be considered by MCnebula2 developers. The only thing users need to know: slot of project_dataset object stores the extracted data.

Slots

`project_version` character(1). The target project version. e.g., "sirius.v4".

`project_path` character(1). The target project path.

`project_conformation` project_conformation object.

`project_metadata` project_metadata object.

`project_api` project_api object.

`project_dataset` project_dataset object.

See Also

Other projects: `project_api-class`, `project_conformation-class`, `project_dataset-class`, `project_metadata-class`

Examples

```
## Not run:
new('project', ...)

## End(Not run)
## Not run:
object <- initialize_mcnebula(mcnebula())
get_upper_dir_subscript(object, ".f3_fingerid")

## End(Not run)
```

---

project_api-class *Function set for extracting data*

Description

This is a class object used to store various functions for extracting and formatting data. See project for joint application with other related classes.

`project_api`, `project_api<-`: getter and setter for the `project_api` slot of the object.

`methods_read`, `methods_read<-`: getter and setter for the `methods_read` slot of the object.

`methods_format`, `methods_format<-`: getter and setter for the `methods_format` slot of the object.

`methods_match`, `methods_match<-`: getter and setter for the `methods_match` slot of the object.

Usage

```
## S4 method for signature 'project_api'
show(object)

## S4 method for signature 'ANY'
project_api(x)

## S4 replacement method for signature 'ANY'
project_api(x) <- value

## S4 method for signature 'project_api'
methods_read(x)

## S4 replacement method for signature 'project_api'
methods_read(x) <- value

## S4 method for signature 'project_api'
methods_format(x)

## S4 replacement method for signature 'project_api'
methods_format(x) <- value
```

```
## S4 method for signature 'project_api'
methods_match(x)

## S4 replacement method for signature 'project_api'
methods_match(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

- `methods_read` list. Store a list of functions for reading data. The list with the names: "read" + "subscript". e.g., "read.f3_fingerid".
- `methods_format` function. The function is used to format the data (e.g., rename the column names; convert the columns of character type into numeric).
- `methods_match` list. Store a list of functions for matching and extracting string.

Note

The class is not for normal use of the package.

See Also

Other projects: `project-class`, `project_conformation-class`, `project_dataset-class`, `project_metadata-class`

Examples

```
## Not run:
new('project_api', ...)

## End(Not run)
```

`project_conformation-class` *Clarify the name, path and attribute name of files in the project (directory)*

Description

This is a class object used to record the name, path and attribute name of the file. These records can be retrieved by "subscript" (see subscript). See project for joint application with other related classes.

`project_conformation`, `project_conformation<-`: getter and setter for the `project_conformation` slot of the object.

`file_name`, `file_name<-`: getter and setter for the `file_name` slot of the object.

`file_api`, `file_api<-`: getter and setter for the `file_api` slot of the object.

`attribute_name`, `attribute_name<-`: getter and setter for the `attribute_name` slot of the object.

Usage

```
## S4 method for signature 'project_conformation'
show(object)

## S4 method for signature 'ANY'
project_conformation(x)

## S4 replacement method for signature 'ANY'
project_conformation(x) <- value

## S4 method for signature 'project_conformation'
file_name(x)

## S4 replacement method for signature 'project_conformation'
file_name(x) <- value

## S4 method for signature 'project_conformation'
file_api(x)

## S4 replacement method for signature 'project_conformation'
file_api(x) <- value

## S4 method for signature 'project_conformation'
attribute_name(x)

## S4 replacement method for signature 'project_conformation'
attribute_name(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`file_name` character with names. Record the filenames or pattern string or function name (begin with "FUN_") for each "subscript" (imply file names).

`file_api` character with names. Record the file path for each "subscript" (imply file names). The path is descriped by "subscript" with "/".

`attribute_name` character with names. Record the attribute name for each "subscript" (imply column names).

Note

The class is not for normal use of the package.

See Also

Other projects: `project-class`, `project_api-class`, `project_dataset-class`, `project_metadata-class`

Other subscripts: mcnebula-class, msframe-class, subscript-class

Examples

```
## Not run:
new('project_conformation', ...)

## End(Not run)
```

---

project_dataset-class *Store extracted data*

Description

This is a class object used to store extracted data (raw data). See project for joint application with other related classes.

`project_dataset`, `project_dataset<-`: getter and setter for the `project_dataset` slot of the object.

`latest`: get the first data in `dataset` slot ('list') and format as 'tbl'. Equals:

- `latest(object)`
- `tibble::as_tibble(entity(dataset(object)[[1]]))`

`extract_rawset`: For fast extract data in object which containing `project_dataset` slot. Normally not used.

Usage

```
## S4 method for signature 'ANY'
project_dataset(x)

## S4 replacement method for signature 'ANY'
project_dataset(x) <- value

## S4 method for signature 'project_dataset,ANY,ANY'
latest(x)

## S4 method for signature 'ANY,character,ANY'
extract_rawset(x, subscript)

## S4 method for signature 'ANY,character,`function`'
extract_rawset(x, subscript, fun_collate, ...)
```

Arguments

`x`	an object contain `project_dataset` slot.
`value`	The value for the slot.
`subscript`	character. Specified the data in `dataset` slot in `project_dataset` slot. See `VIRTUAL_subscript-class`.
`fun_collate`	function. If the specified data not exists in `dataset` slot, it will be used to collate data. This parameter is not for normal use.
`...`	parameters passed to 'fun_collate'.

Slots

`dataset` list. See dataset.

See Also

`tibble::as_tibble()`

Other projects: `project-class`, `project_api-class`, `project_conformation-class`, `project_metadata-class`

Other datasets: `dataset-class`, `mcn_dataset-class`

Other latests: `mcn_dataset-class`, `mcnebula-class`, `msframe-class`, `project_metadata-class`

Examples

```
## Not run:
new('project_dataset', ...)

## End(Not run)
## Not run:
latest(object)

## End(Not run)
## Not run:
extract_rawset(object, ".f3_fingerid")

## End(Not run)
```

`project_metadata-class` *Metadata of files*

Description

This is a class object used to store metadata of files. See project for joint application with other related classes.

`project_metadata`, `project_metadata<-`: getter and setter for the `project_metadata` slot of the object.

`latest`: get the first data in `metadata` slot and format as "tbl".

`metadata`, `metadata<-`: getter and setter for the `metadata` slot of the object.

`add_dataset`: add the list into slot `metadata`.

`extract_metadata`: use "subscript" to extract metadata from an object with slot `project_metadata`, and then return it as a new `project_metadata`.

`get_metadata`: for an object with slot of `project_metadata`, get the metadata of files of specified "subscript", then return the object.

Usage

```
## S4 method for signature 'project_metadata'
show(object)

## S4 method for signature 'ANY'
project_metadata(x)

## S4 replacement method for signature 'ANY'
project_metadata(x) <- value

## S4 method for signature 'project_metadata,ANY,ANY'
latest(x)

## S4 method for signature 'project_metadata'
metadata(x)

## S4 replacement method for signature 'project_metadata'
metadata(x) <- value

## S4 method for signature 'project_metadata,list'
add_dataset(x, list)

## S4 method for signature 'ANY,character'
extract_metadata(x, subscript)

## S4 method for signature 'ANY,character,ANY,ANY,ANY'
get_metadata(x, subscript)

## S4 method for signature
## 'missing,character,project_metadata,project_conformation,character'
get_metadata(subscript, project_metadata, project_conformation, path)
```

Arguments

`value`	The value for the slot.
`list`	a list (with names) of metadata (data.frame) with names.
`subscript`	see subscript.
`project_metadata`	project_metadata object. Used by `get_metadata()`. If 'missing', the slot `project_metadata` inside the object will be used.
`project_conformation`	project_conformation object. Used by `get_metadata()`. If 'missing', the slot `project_conformation` inside the object will be used.
`path`	character. The path of the project directory (generally, SIRIUS project). If 'missing', the slot `project_path` inside the object will be used.

Slots

`metadata` a list with names of subscript. Each element of the list is a data.frame.

Note

The class is not for normal use of the package.

See Also

Other projects: `project-class`, `project_api-class`, `project_conformation-class`, `project_dataset-class`

Other latests: `mcn_dataset-class`, `mcnebula-class`, `msframe-class`, `project_dataset-class`

Examples

```
## Not run:
new('project_metadata', ...)

## End(Not run)
## Not run:
latest(...)

## End(Not run)
## Not run:
add_dataset(...)

## End(Not run)
## Not run:
extract_metadata(...)

## End(Not run)
## Not run:
get_metadata(...)

## End(Not run)
```

`reference-class` *Share slots and methods for classes inherite from VIRTUAL_reference*

Description

This VIRTUAL class provides a slot for storing processed data.

`reference`, `reference<-`: getter and setter for the `reference` slot of the object.

Usage

```
## S4 method for signature 'ANY'
reference(x)

## S4 replacement method for signature 'ANY'
reference(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`reference` list with names (formal name).

`report-class` *Creating a formatted report*

Description

The report module can create output report quickly for and not just for the mcnebula2 workflow. The report system is primarily a class object that manages text and code blocks. Heading or paragraphs or code blocks were treated as individual report units and deposited sequentially in "layers". The report system provides methods to exhibit, modify these layers. Reports can be exported as ".Rmd" text files, and subsequently, users can call `rmarkdown::render()` for output formatted documents.

`show_layers`: show `layers` slots in a pretty and readable style.

`yaml`, `yaml<-`: getter and setter for the `yaml` slot of the object.

`new_report`: Create a report object.

`new_report()`: get the default parameters for the method `new_report`.

`new_report(x, ...)`: use the default parameters whatever 'missing' while performing the method `new_report`.

`call_command`: Format 'report' object as character, which can be output by `writeLines()` function as '.Rmd' file and than use `rmarkdown::render` output as pdf, html, or other format files.

Usage

```
## S4 method for signature 'report'
show_layers(x)

## S4 method for signature 'ANY'
yaml(x)

## S4 replacement method for signature 'ANY'
yaml(x) <- value

## S4 method for signature 'character'
new_report(..., yaml)

## S4 method for signature 'missing'
new_report(..., yaml)

## S4 method for signature 'report'
call_command(x)
```

Arguments

`value`	The value for the slot.
`...`	An arbitrary number of heading, section or code_block in sequence.
`yaml`	character. Passed to .Rmd for setting format of documentation.

Slots

`yaml` character. Metadata passed to .Rmd for setting format of documentation. See `https://bookdown.org/yihui/rmarkdown/compile.html` for details.

`layers` list. Element in list must be section, heading or code_block.

See Also

`writeLines()`, `rmarkdown::render()`...

Other layerSets: `ggset-class`, `layerSet-class`

Other call_commands: `code_block-class`, `command-class`, `ggset-class`, `section-class`

Examples

```
## Not run:
new('report', ...)

## End(Not run)
## Not run:
show_layers(...)

## End(Not run)
## Not run:
```

```
new_report(...)

## End(Not run)
## Not run:
call_command(...)

## End(Not run)
```

`section-class` *Basic cells in the report*

Description

A class object consist of heading, paragraph (character), and code_block. These section belong to basic cells of report.

This is a class object used to clarify the heading and its hierarchy.

`heading`, `heading<-`: getter and setter for the `heading` slot of the object.

`level`, `level<-`: getter and setter for the `level` slot of the object.

`new_heading`: create heading object.

`call_command`: Format 'heading' object as character.

`paragraph`, `paragraph<-`: getter and setter for the `paragraph` slot of the object.

`new_section()`: get the default parameters for the method `new_section`.

`new_section(x, ...)`: use the default parameters whatever 'missing' while performing the method `new_section`.

`new_section`: create section object.

`call_command`: Format 'section' object as character.

Usage

```
## S4 method for signature 'ANY'
heading(x)

## S4 replacement method for signature 'ANY'
heading(x) <- value

## S4 method for signature 'heading'
level(x)

## S4 replacement method for signature 'heading'
level(x) <- value

## S4 method for signature 'character,numeric'
new_heading(heading, level)
```

```
## S4 method for signature 'heading'
call_command(x)

## S4 method for signature 'section'
paragraph(x)

## S4 replacement method for signature 'section'
paragraph(x) <- value

## S4 method for signature 'missing,missing,missing,missing'
new_section(heading, level, paragraph, code_block)

## S4 method for signature 'ANY,ANY,ANY,ANY'
new_section(heading, level, paragraph, code_block)

## S4 method for signature 'character,numeric,character,maybe_code_block'
new_section(heading, level, paragraph, code_block)

## S4 method for signature '`NULL`,numeric,character,maybe_code_block'
new_section(heading, level, paragraph, code_block)

## S4 method for signature 'section'
call_command(x)

## S4 method for signature '`NULL`'
call_command(x)
```

Arguments

`value`	The value for the slot.
`heading`	character(1). For slot `.Data`.
`level`	numeric(1). For slot `level`.
`paragraph`	character. Text for description.
`code_block`	code_block object.

Slots

`heading` heading object.

`paragraph` character. Text for description.

`code_block` code_block object.

`.Data` character(1). Text of heading.

`level` numeric. Level of heading.

See Also

Other call_commands: `code_block-class`, `command-class`, `ggset-class`, `report-class`

Examples

```
## Not run:
new('section', ...)

## End(Not run)
## Not run:
new('heading', ...)

## End(Not run)
## Not run:
new_heading(...)

## End(Not run)
## Not run:
call_command(...)

## End(Not run)
## Not run:
new_section(...)

## End(Not run)
## Not run:
call_command(...)

## End(Not run)
```

`set_nodes_color-methods` ...

Description

...

...

Usage

```
## S4 method for signature 'mcnebula,character,data.frame,missing'
set_nodes_color(x, attribute, extra_data)

## S4 method for signature 'mcnebula,character,missing,missing'
set_nodes_color(x, attribute)

## S4 method for signature 'mcnebula,missing,missing,logical'
set_nodes_color(x, use_tracer)
```

Arguments

`x`	...
`attribute`	...
`extra_data`	...
`use_tracer`	...

Details

...

...

Value

...

See Also

`activate_nebulae()`, `visualize()`...

Examples

```
## Not run:
set_nodes_color(...)

## End(Not run)
```

`set_ppcp_data-methods` ...

Description

...

`set_ppcp_data()`: get the function for generating default parameters for the method `set_ppcp_data`.

`set_ppcp_data(x, ...)`: use the default parameters whatever 'missing' while performing the method `set_ppcp_data`.

...

Usage

```
## S4 method for signature 'missing,missing'
set_ppcp_data()

## S4 method for signature 'mcnebula,ANY'
set_ppcp_data(x, classes)

## S4 method for signature 'mcnebula,character'
set_ppcp_data(x, classes)
```

Arguments

`x`	...
`classes`	...

Details

...

Examples

```
## Not run:
set_ppcp_data(...)

## End(Not run)
```

`set_ration_data-methods`
...

Description

...

`set_ration_data()`: get the default parameters for the method `set_ration_data`.

`set_ration_data(x, ...)`: use the default parameters whatever 'missing' while performing the method `set_ration_data`.

...

Usage

```
## S4 method for signature 'missing,missing'
set_ration_data()

## S4 method for signature 'mcnebula,ANY'
set_ration_data(x, mean)

## S4 method for signature 'mcnebula,logical'
set_ration_data(x, mean)
```

Arguments

`x`	...
`mean`	...

Details

...

Examples

```
## Not run:
set_ration_data(...)

## End(Not run)
```

`set_tracer-methods` ...

Description

...

`set_tracer()`: get the function for generating default parameters for the method `set_tracer`.

`set_tracer(x, ...)`: use the default parameters whatever 'missing' while performing the method `set_tracer`.

...

Usage

```
## S4 method for signature 'missing,missing,missing,missing'
set_tracer()

## S4 method for signature 'mcnebula,ANY,ANY,ANY'
set_tracer(x, .features_id, colors, rest)

## S4 method for signature 'mcnebula,character,character,character'
set_tracer(x, .features_id, colors, rest)
```

Arguments

`x`	...
`.features_id`	...
`colors`	...
`rest`	...

Details

...

...

Value

...

See Also

create_nebula_index()

Examples

```
## Not run:
set_tracer(...)

## End(Not run)
```

---

statistic_set-class *Data used for statistic analysis*

Description

A class object for statistic analysis, associate with package of "limma" for binary comparison.

statistic_set, statistic_set<-: getter and setter for the statistic_set slot of the object.

design_matrix, design_matrix<-: getter and setter for the design_matrix slot of the object.

contrast_matrix, contrast_matrix<-: getter and setter for the contrast_matrix slot of the object.

top_table, top_table<-: getter and setter for the top_table slot of the object.

Usage

```
## S4 method for signature 'ANY'
statistic_set(x)

## S4 replacement method for signature 'ANY'
statistic_set(x) <- value

## S4 method for signature 'ANY'
design_matrix(x)

## S4 replacement method for signature 'ANY'
design_matrix(x) <- value

## S4 method for signature 'ANY'
contrast_matrix(x)

## S4 replacement method for signature 'ANY'
contrast_matrix(x) <- value

## S4 method for signature 'ANY'
top_table(x)

## S4 replacement method for signature 'ANY'
top_table(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`design_matrix` matrix. Create by `stats::model.matrix()`.

`contrast_matrix` matrix. Create by `limma::makeContrasts()`.

`dataset` ANY. Dataset used for `limma::lmFit()`, `limma::eBayes()` and other functions.

`top_table` list with names. Each element of list should be "data.frame" or "tbl".

Examples

```
## Not run:
new('statistic_set', ...)

## End(Not run)
```

`subscript-class` *Share slots and methods for classes inherite from VIRTUAL_subscript*

Description

This VIRTUAL class provides a slot for signing the data. The "subscript" like the signature for data, used to distinguish different data or file and retrieve it accurately. The "subscript" is mostly used for project (as well as its related classes):

- imply file names. e.g., for "sirius.v4", ".f3_fingerid" indicate all files in directory of "fingerid" for each features.
- imply attribute names. e.g., for "sirius.v4", "tani.score" indicate attribute name of "tanimoto-Similarity".

In essence, "subscript" is the alias of a file or data or attribute. In this package, using the "subscript" system means that all external data names are given an alias. In fact, this makes things more complicated. Why did we do this? Because the naming system of external data is not constant, these names may change with the version of the data source. In order to enable this R package to accurately extract and call these data, it is necessary to establish a set of aliases within the package. "Subscript" names are used internally by this package. They correspond to external data and are equivalent to providing an interface to interface with external data.

`subscript`, `subscript<-`: getter and setter for the `subscript` slot of the object.

Usage

```
## S4 method for signature 'ANY'
subscript(x)

## S4 replacement method for signature 'ANY'
subscript(x) <- value
```

Arguments

`value`	The value for the slot.

Slots

`subscript` character(1).

See Also

Other subscripts: `mcnebula-class`, `msframe-class`, `project_conformation-class`

`visualize-methods` ...

Description

...

`visualize(x)`: get a 'tbl' about child-nebulae candidates for `visualize` methods to visualize.

`visualize()`: get the default parameters for the method `visualize`.

`visualize(x, ...)`: use the default parameters whatever 'missing' while performing the method `visualize`.

...

`visualize_all()`: get the default parameters for the method `visualize_all`.

`visualize_all(x, ...)`: use the default parameters whatever 'missing' while performing the method `visualize_all`.

...

Usage

```
## S4 method for signature 'mcnebula,missing,ANY,missing'
visualize(x, fun_modify)

## S4 method for signature 'missing,missing,missing,missing'
visualize()

## S4 method for signature 'mcnebula,ANY,ANY,ANY'
visualize(x, item, fun_modify, annotate)

## S4 method for signature 'mcnebula,character,`function`,missing'
visualize(x, item, fun_modify)

## S4 method for signature 'mcnebula,numeric,`function`,missing'
visualize(x, item, fun_modify)

## S4 method for signature 'mcnebula,numeric_or_character,`function`,logical'
```

```
visualize(x, item, fun_modify, annotate)

## S4 method for signature 'missing,missing,missing,missing'
visualize_all()

## S4 method for signature 'mcnebula,ANY,ANY,ANY'
visualize_all(x, newpage, fun_modify, legend_hierarchy)

## S4 method for signature 'mcnebula,logical,`function`,logical'
visualize_all(x, newpage, fun_modify, legend_hierarchy)
```

Arguments

x	...
fun_modify	...
item	...
annotate	...
newpage	...
legend_hierarchy	...

Details

...

...

Examples

```
## Not run:
visualize(...)

## End(Not run)
## Not run:
visualize_all(...)

## End(Not run)
```

Index

∗ **backtracks**
backtrack-class, 9
∗ **call_commands**
code_block-class, 12
command-class, 16
ggset-class, 40
report-class, 70
section-class, 72
∗ **datasets**
dataset-class, 31
mcn_dataset-class, 50
project_dataset-class, 66
∗ **exports**
export-class, 35
∗ **latests**
mcn_dataset-class, 50
mcnebula-class, 46
msframe-class, 53
project_dataset-class, 66
project_metadata-class, 67
∗ **layerSets**
ggset-class, 40
layerSet-class, 45
report-class, 70
∗ **nebulae**
mcnebula-class, 46
nebula-class, 56
∗ **projects**
project-class, 60
project_api-class, 63
project_conformation-class, 64
project_dataset-class, 66
project_metadata-class, 67
∗ **references**
reference-class, 70
∗ **subscripts**
mcnebula-class, 46
msframe-class, 53
project_conformation-class, 64
subscript-class, 79
.child_nebulae *(nebula-class)*, 56
.code_block *(code_block-class)*, 12
.code_block_figure *(code_block-class)*, 12
.code_block_table *(code_block-class)*, 12
.command *(command-class)*, 16
.ggset *(ggset-class)*, 40
.heading *(section-class)*, 72
.mcn_dataset *(mcn_dataset-class)*, 50
.melody *(melody-class)*, 51
.msframe *(msframe-class)*, 53
.nebula *(nebula-class)*, 56
.parent_nebula *(nebula-class)*, 56
.project *(project-class)*, 60
.project_api *(project_api-class)*, 63
.project_conformation *(project_conformation-class)*, 64
.project_dataset *(project_dataset-class)*, 66
.project_metadata *(project_metadata-class)*, 67
.report *(report-class)*, 70
.section *(section-class)*, 72
.statistic_set *(statistic_set-class)*, 78

ABSTRACT-MCnebula2, 3
activate_nebulae *(activate_nebulae-methods)*, 6
activate_nebulae(), *4*, *75*
activate_nebulae,mcnebula,ANY,ANY-method *(activate_nebulae-methods)*, 6
activate_nebulae,mcnebula,function,function-method *(activate_nebulae-methods)*, 6
activate_nebulae,missing,missing,missing-method *(activate_nebulae-methods)*, 6
activate_nebulae-methods, 6
add_dataset *(project_metadata-class)*, 67

add_dataset,project_metadata,list-method
(project_metadata-class), 67
add_layers (layerSet-class), 45
add_layers,layerSet-method
(layerSet-class), 45
annotate_nebula
(annotate_nebula-methods), 8
annotate_nebula,ANY,character-method
(annotate_nebula-methods), 8
annotate_nebula-methods, 8
attribute_name (project-class), 60
attribute_name
(project_conformation-class),
64
attribute_name,ANY-method
(project-class), 60
attribute_name,project_conformation-method
(project_conformation-class),
64
attribute_name<-
(project_conformation-class),
64
attribute_name<-,project_conformation-method
(project_conformation-class),
64

backtrack (backtrack-class), 9
backtrack,ANY-method (backtrack-class),
9
backtrack-class, 9
backtrack<- (backtrack-class), 9
backtrack<-,ANY-method
(backtrack-class), 9
backtrack_stardust
(backtrack_stardust-methods), 9
backtrack_stardust,mcnebula,character,missing,ANY-method
(backtrack_stardust-methods), 9
backtrack_stardust,mcnebula,missing,missing,missing,ANY-method
(backtrack_stardust-methods), 9
backtrack_stardust,mcnebula,missing,numeric,ANY-method
(backtrack_stardust-methods), 9
backtrack_stardust-methods, 9
binary_comparison
(binary_comparison-methods), 10
binary_comparison(), *4*
binary_comparison,ANY,ANY,ANY,ANY,ANY-method
(binary_comparison-methods), 10
binary_comparison,ANY,formula,function,ANY,character-method
(binary_comparison-methods), 10
binary_comparison,missing,missing,missing,missing,missing-
(binary_comparison-methods), 10
binary_comparison-methods, 10

call_command (code_block-class), 12
call_command (command-class), 16
call_command (ggset-class), 40
call_command (report-class), 70
call_command (section-class), 72
call_command(), *12*, *59*
call_command,code_block-method
(code_block-class), 12
call_command,command-method
(command-class), 16
call_command,ggset-method
(ggset-class), 40
call_command,heading-method
(section-class), 72
call_command,NULL-method
(section-class), 72
call_command,report-method
(report-class), 70
call_command,section-method
(section-class), 72
child_nebulae, *59*
child_nebulae (nebula-class), 56
child_nebulae,ANY-method
(nebula-class), 56
child_nebulae-class (nebula-class), 56
child_nebulae<- (nebula-class), 56
child_nebulae<-,ANY-method
(nebula-class), 56
classification (mcnebula-class), 46
classification,mcnebula-method
(mcnebula-class), 46
code_block, *12*, *71–73*
code_block (code_block-class), 12
code_block,ANY-method
(code_block-class), 12
code_block-class, 12
code_block<- (code_block-class), 12
code_block<-,ANY-method
(code_block-class), 12
code_block_figure, *12*
code_block_figure (code_block-class), 12
code_block_figure-class
(code_block-class), 12
code_block_table, *12*
code_block_table (code_block-class), 12

code_block_table-class
(code_block-class), 12
codes (code_block-class), 12
codes,code_block-method
(code_block-class), 12
codes<- (code_block-class), 12
codes<-,code_block-method
(code_block-class), 12
collate_data (collate_data-methods), 14
collate_data,ANY,ANY,ANY-method
(collate_data-methods), 14
collate_data,ANY,character,function-method
(collate_data-methods), 14
collate_data,missing,missing,missing-method
(collate_data-methods), 14
collate_data-methods, 14
command, *16*, *40*, *41*
command (command-class), 16
command-class, 16
command_args (command-class), 16
command_args,command-method
(command-class), 16
command_args<- (command-class), 16
command_args<-,command-method
(command-class), 16
command_function (command-class), 16
command_function,command-method
(command-class), 16
command_function<- (command-class), 16
command_function<-,command-method
(command-class), 16
command_name (command-class), 16
command_name,command-method
(command-class), 16
command_name<- (command-class), 16
command_name<-,command-method
(command-class), 16
compute_spectral_similarity
(compute_spectral_similarity-methods),
18
compute_spectral_similarity(), *4*
compute_spectral_similarity,mcnebula,ANY,ANY,ANY,ANY-method
(compute_spectral_similarity-methods),
18
compute_spectral_similarity,mcnebula,logical,logical,missing,missing-method
(compute_spectral_similarity-methods),
18
compute_spectral_similarity,missing,missing,missing,data.frame-method
(compute_spectral_similarity-methods),
18
compute_spectral_similarity,missing,missing,missing,lightS
(compute_spectral_similarity-methods),
18
compute_spectral_similarity,missing,missing,missing,missin
(compute_spectral_similarity-methods),
18
compute_spectral_similarity-methods,
18
contrast_matrix (statistic_set-class),
78
contrast_matrix,ANY-method
(statistic_set-class), 78
contrast_matrix<-
(statistic_set-class), 78
contrast_matrix<-,ANY-method
(statistic_set-class), 78
create_child_layouts
(create_child_layouts-methods),
19
create_child_layouts(), *4*
create_child_layouts,mcnebula,ANY,ANY,ANY,ANY,ANY,ANY-meth
(create_child_layouts-methods),
19
create_child_layouts,missing,missing,missing,missing,missi
(create_child_layouts-methods),
19
create_child_layouts-methods, 19
create_child_nebulae
(create_child_nebulae-methods),
20
create_child_nebulae(), *4*
create_child_nebulae,mcnebula,ANY,ANY,ANY-method
(create_child_nebulae-methods),
20
create_child_nebulae,mcnebula,numeric,numeric,logical-meth
(create_child_nebulae-methods),
20
create_child_nebulae,missing,missing,missing,missing-metho
(create_child_nebulae-methods),
20
create_child_nebulae-methods, 20
create_features_annotation
(create_features_annotation-methods),
21
create_features_annotation(), *4*
create_features_annotation,mcnebula,data.frame,missing-met

(create_features_annotation-methods), 21
create_features_annotation,mcnebula,data.frame,numeric-method (create_features_annotation-methods), 21
create_features_annotation,mcnebula,missing,missing-method (create_features_annotation-methods), 21
create_features_annotation-methods, 21
create_hierarchy (create_hierarchy-methods), 22
create_hierarchy,mcnebula,ANY-method (create_hierarchy-methods), 22
create_hierarchy,mcnebula,function-method (create_hierarchy-methods), 22
create_hierarchy,missing,missing-method (create_hierarchy-methods), 22
create_hierarchy-methods, 22
create_nebula_index (create_hierarchy-methods), 22
create_nebula_index (create_nebula_index,missing,missing-method), 23
create_nebula_index(), *4*, *6*, *78*
create_nebula_index,mcnebula,ANY-method (create_nebula_index,missing,missing-method), 23
create_nebula_index,mcnebula,logical-method (create_nebula_index,missing,missing-method), 23
create_nebula_index,missing,missing-method, 23
create_parent_layout (create_parent_layout-methods), 24
create_parent_layout(), *4*
create_parent_layout,mcnebula,ANY,ANY-method (create_parent_layout-methods), 24
create_parent_layout,mcnebula,character,numeric-method (create_parent_layout-methods), 24
create_parent_layout,missing,missing,missing-method (create_parent_layout-methods), 24
create_parent_layout-methods, 24
create_parent_nebula (create_parent_nebula-methods), 25
create_parent_nebula(), *4*
create_parent_nebula,mcnebula,missing,missing-method (create_parent_nebula-methods), 25
create_parent_nebula,mcnebula,numeric,logical-method (create_parent_nebula-methods), 25
create_parent_nebula,mcnebula,numeric,missing-method (create_parent_nebula-methods), 25
create_parent_nebula-methods, 25
create_reference (create_reference-methods), 26
create_reference(), *4*, *5*, *28*, *36*, *38*, *39*
create_reference,mcnebula,ANY,ANY,ANY,ANY,logical,ANY-meth (create_reference-methods), 26
create_reference,mcnebula,character,missing,missing,missin (create_reference-methods), 26
create_reference,mcnebula,missing,character,missing,missin (create_reference-methods), 26
create_reference,mcnebula,missing,missing,data.frame,chara (create_reference-methods), 26
create_reference,mcnebula,missing,missing,data.frame,integ (create_reference-methods), 26
create_reference,mcnebula,missing,missing,data.frame,missi (create_reference-methods), 26
create_reference,mcnebula,missing,missing,missing,missing, (create_reference-methods), 26
create_reference-methods, 26
create_stardust_classes (create_stardust_classes-methods), 28
create_stardust_classes(), *4*, *6*, *29*
create_stardust_classes,mcnebula,ANY,ANY,ANY,ANY-method (create_stardust_classes-methods), 28
create_stardust_classes,mcnebula,numeric,numeric,logical,l (create_stardust_classes-methods), 28
create_stardust_classes,missing,missing,missing,missing,mi (create_stardust_classes-methods), 28
create_stardust_classes-methods, 28
creation_time (mcnebula-class), 46
creation_time,mcnebula-method (mcnebula-class), 46
creation_time<- (mcnebula-class), 46

creation_time<-,mcnebula-method (mcnebula-class), 46
cross_filter_identical (cross_filter_stardust-methods), 29
cross_filter_identical,mcnebula,numeric,numeric-method (cross_filter_stardust-methods), 29
cross_filter_quantity (cross_filter_stardust-methods), 29
cross_filter_quantity,mcnebula,numeric,numeric-method (cross_filter_stardust-methods), 29
cross_filter_score (cross_filter_stardust-methods), 29
cross_filter_score,mcnebula,character,numeric,numeric-method (cross_filter_stardust-methods), 29
cross_filter_stardust, *10*
cross_filter_stardust (cross_filter_stardust-methods), 29
cross_filter_stardust(), *4*, *6*, *28*, *29*
cross_filter_stardust,mcnebula-method (cross_filter_stardust-methods), 29
cross_filter_stardust,missing-method (cross_filter_stardust-methods), 29
cross_filter_stardust-methods, 29

dataset, *51*, *67*
dataset (dataset-class), 31
dataset,ANY-method (dataset-class), 31
dataset-class, 31
dataset<- (dataset-class), 31
dataset<-,ANY-method (dataset-class), 31
delete_layers (layerSet-class), 45
delete_layers,layerSet,numeric-method (layerSet-class), 45
design_matrix (statistic_set-class), 78
design_matrix,ANY-method (statistic_set-class), 78
design_matrix<- (statistic_set-class), 78
design_matrix<-,ANY-method (statistic_set-class), 78
dplyr::filter(), *36*, *37*, *39*
draw_nodes, *8*
draw_nodes (draw_nodes-methods), 32
draw_nodes,mcnebula,character,ANY,ANY,ANY,ANY,ANY-method (draw_nodes-methods), 32
draw_nodes,mcnebula,character,character,logical,logical,lo (draw_nodes-methods), 32
draw_nodes,missing,missing,missing,missing,missing,missing (draw_nodes-methods), 32
draw_nodes-methods, 32
draw_structures, *8*
draw_structures (draw_structures-methods), 34
draw_structures,mcnebula,character-method (draw_structures-methods), 34
draw_structures-methods, 34

entity (msframe-class), 53
entity,msframe-method (msframe-class), 53
entity<- (msframe-class), 53
entity<-,msframe-method (msframe-class), 53
export, *49*
export (export-class), 35
export-class, 35
export_name (export-class), 35
export_name,ANY-method (export-class), 35
export_name<- (export-class), 35
export_name<-,ANY-method (export-class), 35
export_path (export-class), 35
export_path,ANY-method (export-class), 35
export_path<- (export-class), 35
export_path<-,ANY-method (export-class), 35
extract_mcnset (mcn_dataset-class), 50
extract_mcnset,ANY,character-method (mcn_dataset-class), 50
extract_metadata (project_metadata-class), 67
extract_metadata,ANY,character-method (project_metadata-class), 67
extract_rawset (project_dataset-class), 66
extract_rawset,ANY,character,ANY-method (project_dataset-class), 66

extract_rawset,ANY,character,function-method
(project_dataset-class), 66

features_annotation (mcnebula-class), 46
features_annotation,mcnebula-method
(mcnebula-class), 46
features_quantification
(mcnebula-class), 46
features_quantification,mcnebula-method
(mcnebula-class), 46
features_quantification<-,mcnebula-method
(mcnebula-class), 46
file_api (project-class), 60
file_api (project_conformation-class), 64
file_api,ANY-method (project-class), 60
file_api,project_conformation-method
(project_conformation-class), 64
file_api<-
(project_conformation-class), 64
file_api<-,project_conformation-method
(project_conformation-class), 64
file_name (project-class), 60
file_name (project_conformation-class), 64
file_name,ANY-method (project-class), 60
file_name,project_conformation-method
(project_conformation-class), 64
file_name<-
(project_conformation-class), 64
file_name<-,project_conformation-method
(project_conformation-class), 64
filter_formula
(filter_formula-methods), 36
filter_formula(), *4, 5, 16, 26, 27, 38, 39*
filter_formula(object), *27*
filter_formula,mcnebula,ANY,ANY-method
(filter_formula-methods), 36
filter_formula,mcnebula,function,logical-method
(filter_formula-methods), 36
filter_formula,missing,missing,missing-method
(filter_formula-methods), 36
filter_formula-methods, 36
filter_msframe (msframe-class), 53
filter_msframe,msframe,function,formula-method
(msframe-class), 53
filter_msframe,msframe,function,missing-method
(msframe-class), 53
filter_ppcp (filter_ppcp-methods), 37
filter_ppcp(), *16, 26, 27, 29, 59*
filter_ppcp,mcnebula,ANY,ANY-method
(filter_ppcp-methods), 37
filter_ppcp,mcnebula,function,logical-method
(filter_ppcp-methods), 37
filter_ppcp,missing,missing,missing-method
(filter_ppcp-methods), 37
filter_ppcp-methods, 37
filter_structure
(filter_structure-methods), 38
filter_structure(), *4, 5, 16, 26, 27*
filter_structure,mcnebula,ANY,ANY-method
(filter_structure-methods), 38
filter_structure,mcnebula,function,logical-method
(filter_structure-methods), 38
filter_structure,missing,missing,missing-method
(filter_structure-methods), 38
filter_structure-methods, 38
format_msframe (msframe-class), 53
format_msframe,data.frame,missing,missing,missing,missing,
(msframe-class), 53
format_msframe,msframe,character,missing,character,missing
(msframe-class), 53
format_msframe,msframe,missing,function,missing,function,m
(msframe-class), 53
format_msframe,msframe,missing,missing,missing,missing,fun
(msframe-class), 53
format_msframe,msframe,missing,missing,missing,missing,mis
(msframe-class), 53
fun_modify, 39

get_metadata (project_metadata-class), 67
get_metadata,ANY,character,ANY,ANY,ANY-method
(project_metadata-class), 67
get_metadata,missing,character,project_metadata,project_co
(project_metadata-class), 67
get_upper_dir_subscript
(project-class), 60
get_upper_dir_subscript,ANY,character,missing-method
(project-class), 60
ggplot2::scale_fill_manual, *52*
ggraph::create_layout(), *59*

ggraph::ggraph(), *59*
ggsci::pal_d3(), *45*, *53*
ggsci::pal_igv(), *45*, *53*
ggsci::pal_simpsons(), *45*, *53*
ggsci::pal_ucscgb(), *45*, *53*
ggset, *40*, *59*
ggset *(ggset-class)*, 40
ggset *(nebula-class)*, 56
ggset,ANY-method *(nebula-class)*, 56
ggset-class, 40
ggset<- *(nebula-class)*, 56
ggset<-,ANY-method *(nebula-class)*, 56
ggset_activate_child_nebulae *(activate_nebulae-methods)*, 6
ggset_activate_nodes *(draw_nodes-methods)*, 32
ggset_activate_parent_nebula *(activate_nebulae-methods)*, 6
ggset_annotate *(nebula-class)*, 56
ggset_annotate,ANY-method *(nebula-class)*, 56
ggset_annotate<- *(nebula-class)*, 56
ggset_annotate<-,ANY-method *(nebula-class)*, 56
grid::grid.draw(), *59*
grid::grid.layout(), *59*
grid::grob(), *59*
grid::viewport(), *59*
grid_layout *(nebula-class)*, 56
grid_layout,ANY-method *(nebula-class)*, 56
grid_layout<- *(nebula-class)*, 56
grid_layout<-,ANY-method *(nebula-class)*, 56

heading, *71–73*
heading *(section-class)*, 72
heading,ANY-method *(section-class)*, 72
heading-class *(section-class)*, 72
heading<- *(section-class)*, 72
heading<-,ANY-method *(section-class)*, 72
hierarchy *(mcnebula-class)*, 46
hierarchy,mcnebula-method *(mcnebula-class)*, 46
history_rblock *(history_rblock-methods)*, 41
history_rblock,missing,character,character,ANY-method *(history_rblock-methods)*, 41
history_rblock,missing,missing,missing,missing-method *(history_rblock-methods)*, 41
history_rblock,numeric,ANY,ANY,ANY-method *(history_rblock-methods)*, 41
history_rblock,numeric,missing,missing,numeric-method *(history_rblock-methods)*, 41
history_rblock-methods, 41

igraph *(nebula-class)*, 56
igraph,ANY-method *(nebula-class)*, 56
igraph::graph_from_data_frame(), *59*
igraph::write_graph(), *59*
igraph<- *(nebula-class)*, 56
igraph<-,ANY-method *(nebula-class)*, 56
include_figure *(include_figure-methods)*, 42
include_figure,character,character,character-method *(include_figure-methods)*, 42
include_figure-methods, 42
include_table *(include_table-methods)*, 43
include_table,data.frame,character,character-method *(include_table-methods)*, 43
include_table-methods, 43
initialize_mcnebula *(initialize_mcnebula-methods)*, 44
initialize_mcnebula(), *4*, *52*
initialize_mcnebula,mcnebula,ANY-method *(initialize_mcnebula-methods)*, 44
initialize_mcnebula,melody,ANY-method *(initialize_mcnebula-methods)*, 44
initialize_mcnebula-methods, 44
ion_mode *(mcnebula-class)*, 46
ion_mode,mcnebula-method *(mcnebula-class)*, 46
ion_mode<- *(mcnebula-class)*, 46
ion_mode<-,mcnebula-method *(mcnebula-class)*, 46

latest *(mcn_dataset-class)*, 50
latest *(mcnebula-class)*, 46
latest *(msframe-class)*, 53
latest *(project_dataset-class)*, 66
latest *(project_metadata-class)*, 67
latest,mcn_dataset,ANY,ANY-method *(mcn_dataset-class)*, 50

latest,mcnebula,ANY,ANY-method
  *(*mcnebula-class*)*, 46
latest,mcnebula,character,ANY-method
  *(*mcnebula-class*)*, 46
latest,missing,missing,missing-method
  *(*mcnebula-class*)*, 46
latest,msframe,ANY,ANY-method
  *(*msframe-class*)*, 53
latest,project_dataset,ANY,ANY-method
  *(*project_dataset-class*)*, 66
latest,project_metadata,ANY,ANY-method
  *(*project_metadata-class*)*, 67
layers *(*layerSet-class*)*, 45
layers,layerSet-method
  *(*layerSet-class*)*, 45
layers<- *(*layerSet-class*)*, 45
layers<-,layerSet-method
  *(*layerSet-class*)*, 45
layerSet *(*layerSet-class*)*, 45
layerSet-class, 45
layout_ggraph *(*nebula-class*)*, 56
layout_ggraph,ANY-method
  *(*nebula-class*)*, 56
layout_ggraph<- *(*nebula-class*)*, 56
layout_ggraph<-,ANY-method
  *(*nebula-class*)*, 56
legend_viewport *(*nebula-class*)*, 56
legend_viewport,ANY-method
  *(*nebula-class*)*, 56
legend_viewport<- *(*nebula-class*)*, 56
legend_viewport<-,ANY-method
  *(*nebula-class*)*, 56
level *(*section-class*)*, 72
level,heading-method *(*section-class*)*, 72
level<- *(*section-class*)*, 72
level<-,heading-method *(*section-class*)*,
  72
limma::contrasts.fit(), *11*
limma::eBayes(), *11*, *79*
limma::lmFit(), *11*, *79*
limma::makeContrasts(), *11*, *79*
limma::topTable(), *11*

match.candidates_id *(*project-class*)*, 60
match.candidates_id,ANY-method
  *(*project-class*)*, 60
match.features_id *(*project-class*)*, 60
match.features_id,ANY-method
  *(*project-class*)*, 60
mcn_dataset, *49*
mcn_dataset *(*mcn_dataset-class*)*, 50
mcn_dataset,ANY-method
  *(*mcn_dataset-class*)*, 50
mcn_dataset-class, 50
mcn_dataset<- *(*mcn_dataset-class*)*, 50
mcn_dataset<-,ANY-method
  *(*mcn_dataset-class*)*, 50
mcnebula, *22*, *27*, *28*, *36–38*, *45*, *49*
mcnebula *(*mcnebula-class*)*, 46
mcnebula-class, 46
MCnebula2, *37*
MCnebula2 *(*ABSTRACT-MCnebula2*)*, 3
melody, *44*, *45*, *49*
melody *(*melody-class*)*, 51
melody,ANY-method *(*melody-class*)*, 51
melody-class, 51
melody<- *(*melody-class*)*, 51
melody<-,ANY-method *(*melody-class*)*, 51
metadata *(*project_metadata-class*)*, 67
metadata,ANY-method *(*project-class*)*, 60
metadata,project_metadata-method
  *(*project_metadata-class*)*, 67
metadata<- *(*project_metadata-class*)*, 67
metadata<-,project_metadata-method
  *(*project_metadata-class*)*, 67
methods_format *(*project-class*)*, 60
methods_format *(*project_api-class*)*, 63
methods_format,ANY-method
  *(*project-class*)*, 60
methods_format,project_api-method
  *(*project_api-class*)*, 63
methods_format<- *(*project_api-class*)*, 63
methods_format<-,project_api-method
  *(*project_api-class*)*, 63
methods_match *(*project-class*)*, 60
methods_match *(*project_api-class*)*, 63
methods_match,ANY-method
  *(*project-class*)*, 60
methods_match,project_api-method
  *(*project_api-class*)*, 63
methods_match<- *(*project_api-class*)*, 63
methods_match<-,project_api-method
  *(*project_api-class*)*, 63
methods_read *(*project-class*)*, 60
methods_read *(*project_api-class*)*, 63
methods_read,ANY-method
  *(*project-class*)*, 60

methods_read,project_api-method
(project_api-class), 63
methods_read<- (project_api-class), 63
methods_read<-,project_api-method
(project_api-class), 63
modify_annotate_child (fun_modify), 39
modify_default_child (fun_modify), 39
modify_rm_legend (fun_modify), 39
modify_set_labs (fun_modify), 39
modify_set_labs_and_unify_scale_limits
(fun_modify), 39
modify_set_margin (fun_modify), 39
modify_unify_scale_limits (fun_modify),
39
move_layers (layerSet-class), 45
move_layers,layerSet,numeric,numeric-method
(layerSet-class), 45
msframe, *54*
msframe (msframe-class), 53
msframe,ANY-method (msframe-class), 53
msframe-class, 53
msframe<- (msframe-class), 53
msframe<-,ANY-method (msframe-class), 53
mutate_layer (ggset-class), 40
mutate_layer,ggset,character-method
(ggset-class), 40
mutate_layer,ggset,numeric-method
(ggset-class), 40

nebula, *49*
nebula (nebula-class), 56
nebula-class, 56
nebula_index (mcnebula-class), 46
nebula_index,mcnebula-method
(mcnebula-class), 46
new_code_block (code_block-class), 12
new_code_block(), *13*
new_code_block,ANY,ANY,ANY,ANY,ANY-method
(code_block-class), 12
new_code_block,character,character,list,logical,function-method
(code_block-class), 12
new_code_block,missing,missing,missing,missing,missing-method
(code_block-class), 12
new_code_block_figure
(code_block-class), 12
new_code_block_figure,character-method
(code_block-class), 12
new_code_block_table
(code_block-class), 12
new_code_block_table,character-method
(code_block-class), 12
new_command (command-class), 16
new_command,function,character-method
(command-class), 16
new_command,function,missing-method
(command-class), 16
new_ggset (ggset-class), 40
new_ggset,ANY-method (ggset-class), 40
new_heading (section-class), 72
new_heading,character,numeric-method
(section-class), 72
new_report (report-class), 70
new_report,character-method
(report-class), 70
new_report,missing-method
(report-class), 70
new_section (section-class), 72
new_section,ANY,ANY,ANY,ANY-method
(section-class), 72
new_section,character,numeric,character,maybe_code_block-m
(section-class), 72
new_section,missing,missing,missing,missing-method
(section-class), 72
new_section,NULL,numeric,character,maybe_code_block-method
(section-class), 72
nodes_ggset (nebula-class), 56
nodes_ggset,ANY-method (nebula-class),
56
nodes_ggset<- (nebula-class), 56
nodes_ggset<-,ANY-method
(nebula-class), 56
nodes_grob (nebula-class), 56
nodes_grob,ANY-method (nebula-class), 56
nodes_grob<- (nebula-class), 56
nodes_grob<-,ANY-method (nebula-class),
56

palette_col (mcnebula-class), 46
palette_col (melody-class), 51
palette_col,mcnebula-method
(mcnebula-class), 46
palette_col,melody-method
(melody-class), 51
palette_col<- (melody-class), 51
palette_col<-,melody-method
(melody-class), 51
palette_gradient (mcnebula-class), 46
palette_gradient (melody-class), 51

palette_gradient,mcnebula-method (*mcnebula-class*), 46
palette_gradient,melody-method (*melody-class*), 51
palette_gradient<- (*melody-class*), 51
palette_gradient<-,melody-method (*melody-class*), 51
palette_label (*mcnebula-class*), 46
palette_label (*melody-class*), 51
palette_label,mcnebula-method (*mcnebula-class*), 46
palette_label,melody-method (*melody-class*), 51
palette_label<- (*melody-class*), 51
palette_label<-,melody-method (*melody-class*), 51
palette_set (*mcnebula-class*), 46
palette_set (*melody-class*), 51
palette_set,mcnebula-method (*mcnebula-class*), 46
palette_set,melody-method (*melody-class*), 51
palette_set<- (*melody-class*), 51
palette_set<-,melody-method (*melody-class*), 51
palette_stat (*mcnebula-class*), 46
palette_stat (*melody-class*), 51
palette_stat,mcnebula-method (*mcnebula-class*), 46
palette_stat,melody-method (*melody-class*), 51
palette_stat<- (*melody-class*), 51
palette_stat<-,melody-method (*melody-class*), 51
panel_viewport (*nebula-class*), 56
panel_viewport,ANY-method (*nebula-class*), 56
panel_viewport<- (*nebula-class*), 56
panel_viewport<-,ANY-method (*nebula-class*), 56
paragraph (*section-class*), 72
paragraph,section-method (*section-class*), 72
paragraph<- (*section-class*), 72
paragraph<-,section-method (*section-class*), 72
parent_nebula, *58*
parent_nebula (*nebula-class*), 56
parent_nebula,ANY-method (*nebula-class*), 56
parent_nebula-class (*nebula-class*), 56
parent_nebula<- (*nebula-class*), 56
parent_nebula<-,ANY-method (*nebula-class*), 56
ppcp_data (*nebula-class*), 56
ppcp_data,ANY-method (*nebula-class*), 56
ppcp_data<- (*nebula-class*), 56
ppcp_data<-,ANY-method (*nebula-class*), 56
project, *15, 16*, *49*, *61*, *63, 64*, *66*, *68*, *79*
project (*project-class*), 60
project-class, 60
project_api, *62*
project_api (*project_api-class*), 63
project_api,ANY-method (*project_api-class*), 63
project_api-class, 63
project_api<- (*project_api-class*), 63
project_api<-,ANY-method (*project_api-class*), 63
project_conformation, *62*, *69*
project_conformation (*project_conformation-class*), 64
project_conformation,ANY-method (*project_conformation-class*), 64
project_conformation-class, 64
project_conformation<- (*project_conformation-class*), 64
project_conformation<-,ANY-method (*project_conformation-class*), 64
project_dataset, *62*
project_dataset (*project_dataset-class*), 66
project_dataset,ANY-method (*project_dataset-class*), 66
project_dataset-class, 66
project_dataset<- (*project_dataset-class*), 66
project_dataset<-,ANY-method (*project_dataset-class*), 66
project_metadata, *15*, *62*, *69*
project_metadata (*project-class*), 60

project_metadata
(project_metadata-class), 67
project_metadata,ANY-method
(project_metadata-class), 67
project_metadata-class, 67
project_metadata<-
(project_metadata-class), 67
project_metadata<-,ANY-method
(project_metadata-class), 67
project_path (project-class), 60
project_path,ANY-method
(project-class), 60
project_path<- (project-class), 60
project_path<-,ANY-method
(project-class), 60
project_version (project-class), 60
project_version,ANY-method
(project-class), 60
project_version<- (project-class), 60
project_version<-,ANY-method
(project-class), 60

ration_data (nebula-class), 56
ration_data,ANY-method (nebula-class),
56
ration_data<- (nebula-class), 56
ration_data<-,ANY-method
(nebula-class), 56
read_data,ANY,project_metadata,character,missing,missing,missing-method
(collate_data-methods), 14
read_data,missing,missing,character,character,character,function,function-method
(collate_data-methods), 14
reference (mcnebula-class), 46
reference (reference-class), 70
reference,ANY-method (reference-class),
70
reference,mcnebula-method
(mcnebula-class), 46
reference-class, 70
reference<- (reference-class), 70
reference<-,ANY-method
(reference-class), 70
report, *70*
report (report-class), 70
report-class, 70
rmarkdown::render(), *70*, *71*

sample_metadata (mcnebula-class), 46
sample_metadata,mcnebula-method
(mcnebula-class), 46
sample_metadata<- (mcnebula-class), 46
sample_metadata<-,mcnebula-method
(mcnebula-class), 46
section, *71*, *72*
section (section-class), 72
section-class, 72
set_nodes_color
(set_nodes_color-methods), 74
set_nodes_color,mcnebula,character,data.frame,missing-meth
(set_nodes_color-methods), 74
set_nodes_color,mcnebula,character,missing,missing-method
(set_nodes_color-methods), 74
set_nodes_color,mcnebula,missing,missing,logical-method
(set_nodes_color-methods), 74
set_nodes_color-methods, 74
set_ppcp_data (set_ppcp_data-methods),
75
set_ppcp_data,mcnebula,ANY-method
(set_ppcp_data-methods), 75
set_ppcp_data,mcnebula,character-method
(set_ppcp_data-methods), 75
set_ppcp_data,missing,missing-method
(set_ppcp_data-methods), 75
set_ppcp_data-methods, 75
set_ration_data
(set_ration_data-methods), 76
set_ration_data,mcnebula,ANY-method
(set_ration_data-methods), 76
set_ration_data,mcnebula,logical-method
(set_ration_data-methods), 76
set_ration_data,missing,missing-method
(set_ration_data-methods), 76
set_ration_data-methods, 76
set_tracer (set_tracer-methods), 77
set_tracer,mcnebula,ANY,ANY,ANY-method
(set_tracer-methods), 77
set_tracer,mcnebula,character,character,character-method
(set_tracer-methods), 77
set_tracer,missing,missing,missing,missing-method
(set_tracer-methods), 77
set_tracer-methods, 77
show (code_block-class), 12
show (command-class), 16
show (layerSet-class), 45
show (mcnebula-class), 46
show (melody-class), 51

show (msframe-class), 53
show (project_api-class), 63
show (project_conformation-class), 64
show (project_metadata-class), 67
show,child_nebulae-method (nebula-class), 56
show,code_block-method (code_block-class), 12
show,code_block_figure-method (code_block-class), 12
show,code_block_table-method (code_block-class), 12
show,command-method (command-class), 16
show,heading-method (code_block-class), 12
show,layerSet-method (layerSet-class), 45
show,mcnebula-method (mcnebula-class), 46
show,melody-method (melody-class), 51
show,msframe-method (msframe-class), 53
show,parent_nebula-method (nebula-class), 56
show,project_api-method (project_api-class), 63
show,project_conformation-method (project_conformation-class), 64
show,project_metadata-method (project_metadata-class), 67
show,section-method (code_block-class), 12
show_layers (ggset-class), 40
show_layers (report-class), 70
show_layers,ggset-method (ggset-class), 40
show_layers,report-method (report-class), 70
show_node (draw_nodes-methods), 32
show_node,ANY,character,ANY,ANY-method (draw_nodes-methods), 32
show_node,missing,missing,missing,missing-method (draw_nodes-methods), 32
show_structure,ANY,character-method (draw_structures-methods), 34
specific_candidate (mcnebula-class), 46
specific_candidate,mcnebula-method (mcnebula-class), 46
spectral_similarity (mcnebula-class), 46
spectral_similarity,mcnebula-method (mcnebula-class), 46
spectral_similarity<- (mcnebula-class), 46
spectral_similarity<-,mcnebula-method (mcnebula-class), 46
stardust_classes (mcnebula-class), 46
stardust_classes,mcnebula-method (mcnebula-class), 46
statistic_set, *49*
statistic_set (statistic_set-class), 78
statistic_set,ANY-method (statistic_set-class), 78
statistic_set-class, 78
statistic_set<- (statistic_set-class), 78
statistic_set<-,ANY-method (statistic_set-class), 78
stats::model.matrix(), *11*, *79*
structures_grob (nebula-class), 56
structures_grob,ANY-method (nebula-class), 56
structures_grob<- (nebula-class), 56
structures_grob<-,ANY-method (nebula-class), 56
styler::style_text(), *13*
subscript, *14, 15*, *27*, *51*, *55*, *61*, *62*, *64*, *69*
subscript (subscript-class), 79
subscript,ANY-method (subscript-class), 79
subscript-class, 79
subscript<- (subscript-class), 79
subscript<-,ANY-method (subscript-class), 79

tbl_graph (nebula-class), 56
tbl_graph,ANY-method (nebula-class), 56
tbl_graph<- (nebula-class), 56
tbl_graph<-,ANY-method (nebula-class), 56
tibble::as_tibble(), *50*, *55*, *67*
tidygraph::as_tbl_graph(), *59*
top_table (statistic_set-class), 78
top_table,ANY-method (statistic_set-class), 78
top_table<- (statistic_set-class), 78
top_table<-,ANY-method (statistic_set-class), 78

viewports *(nebula-class)*, 56
viewports,ANY-method *(nebula-class)*, 56
viewports<- *(nebula-class)*, 56
viewports<-,ANY-method *(nebula-class)*, 56
VIRTUAL_backtrack *(backtrack-class)*, 9
VIRTUAL_dataset *(dataset-class)*, 31
VIRTUAL_export *(export-class)*, 35
VIRTUAL_layerSet *(layerSet-class)*, 45
VIRTUAL_reference *(reference-class)*, 70
VIRTUAL_subscript *(subscript-class)*, 79
visualize *(visualize-methods)*, 80
visualize(), *4*, *75*
visualize,mcnebula,ANY,ANY,ANY-method *(visualize-methods)*, 80
visualize,mcnebula,character,function,missing-method *(visualize-methods)*, 80
visualize,mcnebula,missing,ANY,missing-method *(visualize-methods)*, 80
visualize,mcnebula,numeric,function,missing-method *(visualize-methods)*, 80
visualize,mcnebula,numeric_or_character,function,logical-method *(visualize-methods)*, 80
visualize,missing,missing,missing,missing-method *(visualize-methods)*, 80
visualize-methods, 80
visualize_all,mcnebula,ANY,ANY,ANY-method *(visualize-methods)*, 80
visualize_all,mcnebula,logical,function,logical-method *(visualize-methods)*, 80
visualize_all,missing,missing,missing,missing-method *(visualize-methods)*, 80

writeLines(), *71*

yaml *(report-class)*, 70
yaml,ANY-method *(report-class)*, 70
yaml<- *(report-class)*, 70
yaml<-,ANY-method *(report-class)*, 70