

Package ‘mindr’

May 2, 2019

Version 1.2.2

Date 2019-05-01

Title Convert Files Between Markdown or Rmarkdown Files and Mindmaps

Author Peng Zhao

Maintainer Peng Zhao <pzhao@pzhao.net>

Depends R (>= 3.0.0)

Imports htmlwidgets, knitr, jsonlite, data.tree

Suggests

Description Convert Markdown (‘.md’) or Rmarkdown (‘.Rmd’) files into mindmap widgets or files (‘.mm’), and vice versa. FreeMind mindmap (‘.mm’) files can be opened by or imported to common mindmap software such as ‘FreeMind’ (<http://freemind.sourceforge.net/wiki/index.php/Main_Page>) and ‘XMind’ (<<http://www.xmind.net>>).

License MIT + file LICENSE

URL <https://github.com/pzhaonet/mindr>

BugReports <https://github.com/pzhaonet/mindr/issues>

RoxygenNote 6.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2019-05-02 11:40:11 UTC

R topics documented:

count_space	2
dir2	3
dir4	4
get_body	4
get_eqloc	5
get_filename_ext	5
get_foldername	6
get_heading	6

get_heading2	7
get_heading3	7
markmap	8
markmapOption	9
markmapOutput	10
md2mm	10
md2r	11
mdtxt2mmtxt	12
mm	13
mm2md	15
mm2r	15
outline	16
r2md	17
r2mm	18
r2rmd	18
rename2	19
renderMarkmap	19
rmd2r	20
rmvcode	20
tree	21
tree2mm	22
writeLines2	23

Index 24

count_space	<i>Count the spaces between two given strings</i>
-------------	---

Description

Count the spaces between two given strings

Usage

```
count_space(mychar, sep)
```

Arguments

mychar	The character to check.
sep	character for separation.

Value

character as title with '#' inserted.

dir2	<i>Convert a folder structure into a mindmap by using the 'tree' command.</i>
------	---

Description

Convert a folder structure into a mindmap by using the 'tree' command.

Usage

```
dir2(path = getwd(), savefile = TRUE, savefilename = "mindr.mm", output = c("mm",  
  "txt", "md", "Rmd"), backup = TRUE, dir_files = FALSE)
```

Arguments

path	character. the path of the folder.
savefile	logical. Whether to save the output as a file.
savefilename	character. Valid when savefile == TRUE.
output	a file with the folder structure.
backup	logical. Whether the existing target file, if any, should be saved as backup.
dir_files	logical. Whether to display files besides folders.

Details

For Linux OS and mac OS, the 'tree' command must be pre-installed.

- Linux: `sudo apt-get install tree`
- mac: install [Homebrew](#) first. Then in the terminal: `brew install tree`.

Value

a mindmap file, which can be viewed by common mindmap software, such as 'FreeMind' (http://freemind.sourceforge.net/wiki/index.php/Main_Page) and 'XMind' (<http://www.xmind.net>).

dir4 *Convert a folder structure into a mindmap.*

Description

Convert a folder structure into a mindmap.

Usage

```
dir4(path = getwd(), savefile = TRUE, savefilename = "mindr.mm", output = c("mm",
  "txt", "md", "Rmd"), backup = TRUE, dir_files = FALSE)
```

Arguments

path	character. the path of the folder.
savefile	logical. Whether to save the output as a file.
savefilename	character. Valid when savefile == TRUE.
output	a file with the folder structure.
backup	logical. Whether the existing target file, if any, should be saved as backup.
dir_files	logical. Whether to display files besides folders.

Value

a mindmap file, which can be viewed by common mindmap software, such as 'FreeMind' (http://freemind.sourceforge.net/wiki/index.php/Main_Page) and 'XMind' (<http://www.xmind.net>).

get_body *get the body out of given strings*

Description

get the body out of given strings

Usage

```
get_body(pattern = "^#[^ ]*", text)
```

Arguments

pattern	The definition of the body text
text	the given strings

Value

integer. the index of the body text in the given strings.

get_eqloc *Get the index of equations in a string vector*

Description

Get the index of equations in a string vector

Usage

get_eqloc(eq_begin, eq_end)

Arguments

eq_begin the beginning index of an equation
eq_end the end index of an equation

Value

a index vector

get_filename_ext *get the file name extension*

Description

get the file name extension

Usage

get_filename_ext(filename)

Arguments

filename character, the file name

Value

character, the file name extension

get_foldername	<i>Get the folder name of a given complete path</i>
----------------	---

Description

Get the folder name of a given complete path

Usage

```
get_foldername(path)
```

Arguments

path	The complete path
------	-------------------

Value

The folder name

get_heading	<i>get the headings out of given strings</i>
-------------	--

Description

get the headings out of given strings

Usage

```
get_heading(pattern = "^#+ ", text)
```

Arguments

pattern	The definition of the headings
text	the given strings

Value

integer. the index of the headings in the given strings.

get_heading2 *get the headings out of given strings*

Description

get the headings out of given strings

Usage

```
get_heading2(pattern = "^#= #+ ", text)
```

Arguments

pattern	The definition of the headings
text	the given strings

Value

integer. the index of the headings in the given strings.

get_heading3 *get the headings out of given strings*

Description

get the headings out of given strings

Usage

```
get_heading3(pattern = "^#' #+ ", text)
```

Arguments

pattern	The definition of the headings
text	the given strings

Value

integer. the index of the headings in the given strings.

 markmap

Create a markmap widget

Description

This function, modified from <https://github.com/seifer08ms/Rmarkmap>, creates a markmap widget using htmlwidgets. The widget can be rendered on HTML pages generated from R Markdown, Shiny, or other applications.

Usage

```
markmap(root = NA, input = c(".md", ".Rmd", ".mm"), path = ".",
  remove_curly_bracket = FALSE, width = NULL, height = NULL, elementId = NULL,
  options = markmapOption(preset = "colorful"), bookdown_style = TRUE,
  method = c("regexpr", "pandoc"))
```

Arguments

root	character. a string displayed as the root of the mind map
input	character, The format of the input files
path	character. The path of the folder which contains the input file(s).
remove_curly_bracket	logical. Whether to remove #ID in the headers of the markdown file (usually in a 'bookdown' https://github.com/rstudio/bookdown project).
width	the width of the markmap
height	the height of the markmap
elementId	character.
options	the markmap options
bookdown_style	logical. whether the markdown files are in bookdown style, i.e. index.Rmd at the beginning, # (PART), # (APPENDIX) and # References as an upper level of normal # title
method	"regexpr" uses regular expressions, 'pandoc' uses pandoc to find the headings.

Value

A HTML widget object rendered from a given document.

Examples

```
path <- system.file("examples/md", package = "mindr")
markmap(path = path)
markmap(path = path, remove_curly_bracket = TRUE)
```

markmapOption	<i>Options for markmap creation</i>
---------------	-------------------------------------

Description

This function is taken from <https://github.com/seifer08ms/Rmarkmap>.

Usage

```
markmapOption(preset = NULL, nodeHeight = 20, nodeWidth = 180, spacingVertical = 10,
  spacingHorizontal = 120, duration = 750, layout = "tree", color = "gray",
  linkShape = "diagonal", renderer = "boxed", ...)
```

Arguments

preset	the name of built-in theme for markmap. If present, any other parameters will be ignored.
nodeHeight	the height of nodes in the markmap.
nodeWidth	the width of nodes in the markmap.
spacingVertical	space of vertical.
spacingHorizontal	space of horizontal.
duration	duration time for animation.
layout	layout mode of makrmap. Currently, only 'tree' is accepted.
color	color of markmap. A character color value ,either 'gray' or a categorical colors including 'category10','category20','category20b' and 'category20c'.
linkShape	link shape of markmap. A character value, either 'diagonal' or 'bracket'.
renderer	rendered shaped of markmap. A character value ,either 'basic' or 'boxed'.
...	other options.

Details

Currently,markmap have 'default' and 'colorful' themes. 'colorful' themes have three different parameters from default themes: nodeHeight: 10, renderer: 'basic',color: 'category20'

Functions

- markmapOption: Options for markmap creation

See Also

<https://github.com/dundalek/markmap/blob/master/lib/view.mindmap.js> for details.

Examples

```
path <- system.file('examples/md', package = 'mindr')
markmap(path = path, remove_curly_bracket = TRUE,
  options = markmapOption(preset = 'colorful')) # 'colorful' theme
markmap(path = path, remove_curly_bracket = TRUE,
  options = markmapOption(color = 'category20b',
  linkShape = 'bracket')) # 'colorful' theme
markmap(path = path, remove_curly_bracket = TRUE,
  options = markmapOption(color = 'category20b',
  linkShape = 'diagonal',
  renderer = 'basic')) # 'colorful' theme
```

markmapOutput	<i>Shiny bindings for markmap</i>
---------------	-----------------------------------

Description

Output function for using markmap within Shiny applications and interactive Rmd documents. This function is taken from <https://github.com/seifer08ms/Rmarkmap>.

Usage

```
markmapOutput(outputId, width = "100%", height = "400px")
```

Arguments

outputId	output variable to read from
width	Must be a valid CSS unit (like '100%', '400px', 'auto') or a number, which will be coerced to a string and have 'px' appended.
height	See 'width'.

md2mm	<i>Convert markdown or rmarkdown files to mindmap files.</i>
-------	--

Description

Convert markdown or rmarkdown files to mindmap files.

Usage

```
md2mm(pattern = "*.[R]*md$", title = NA, path = ".", remove_curly_bracket = FALSE,
  savefile = TRUE, savefilename = NA, backup = TRUE, bookdown_style = TRUE,
  keep_eq = FALSE, method = c("regexpr", "pandoc"))
```

Arguments

pattern	an optional regular expression for filtering the input files. See <code>help(dir)</code> .
title	character. The title of the output file.
path	character. The path of the folder which contains the input file(s).
remove_curly_bracket	logical. Whether to remove #ID in the headers of the markdown file (usually in a 'bookdown' https://github.com/rstudio/bookdown project).
savefile	logical. Whether to save the output as a file.
savefilename	character. Valid when <code>savefile == TRUE</code> .
backup	logical. Whether the existing target file, if any, should be saved as backup.
bookdown_style	logical. whether the markdown files are in bookdown style, i.e. <code>index.Rmd</code> at the beginning, <code># (PART)</code> , <code># (APPENDIX)</code> and <code># References</code> as an upper level of normal <code># title</code>
keep_eq	logical. whether to keep LaTeX equations.
method	"regexpr" uses regular expressions, 'pandoc' uses pandoc to find the headings.

Value

a mindmap file, which can be viewed by common mindmap software, such as 'FreeMind' (http://freemind.sourceforge.net/wiki/index.php/Main_Page) and 'XMind' (<http://www.xmind.net>).

Examples

```
path <- system.file("examples/md", package = "mindr")
md2mm(path = path)
md2mm(path = path, remove_curly_bracket = TRUE)
```

 md2r

Convert .md or .Rmd files into a .R script

Description

Convert .md or .Rmd files into a .R script

Usage

```
md2r(filepattern = "*.R]*md$", path = ".", savefilename = NA, backup = TRUE,
      heading = " -----", body = "#")
```

Arguments

filepattern	the pattern of the file names
path	the path of the folder which contains the .Rmd or .md files
savefilename	the destined file name
backup	logical. whether backup the existent file
heading	the indicator of the headings
body	the indicator of the body text

Value

a .R script

Examples

```
path <- system.file("examples/md", package = "mindr")
md2r(path = path)
```

mdtxt2mmtxt

Convert markdown text to mindmap text.

Description

Convert markdown text to mindmap text.

Usage

```
mdtxt2mmtxt(title = "my title", mdtxt = "", keep_eq = FALSE)
```

Arguments

title	character. The title of the output file.
mdtxt	character. The markdown text to convert.
keep_eq	logical. whether to keep LaTeX equations.

Value

a mindmap text.

Examples

```
mdtxt2mmtxt(mdtxt = c("# Chapter 1", "## Section 1.1", "## Section 1.2"))
```

mm	<i>Convert between .R, .Rmd, .mm according to the given file names, and create a markmap widget</i>
----	---

Description

Convert between .R, .Rmd, .mm according to the given file names, and create a markmap widget

Usage

```
mm(from = NULL, to = NULL, type = c("file", "text", "dir"), root = NA,
  show_files = TRUE, remove_curly_bracket = TRUE, bookdown_style = TRUE,
  widget_name = NA, width = NULL, height = NULL, elementId = NULL,
  options = markmapOption(preset = "colorful"), method = c("regexpr",
    "pandoc"))
```

Arguments

from	character. The path of the input file, or the input markdown text, or the path to the directory. Dependent on 'type'.
to	character. The path of the output file.
type	character. The type of the input. If type == 'dir' and the OS is LinUx, the 'tree' command must be pre-installed: <code>sudo apt-get install tree</code> .
root	character. a string displayed as the root of the mind map
show_files	logical. Whether to show files in a directory. Only valid when type == 'dir'.
remove_curly_bracket	logical. Whether to remove #ID in the headers of the markdown file (usually in a 'bookdown' https://github.com/rstudio/bookdown project).
bookdown_style	logical. whether the markdown files are in bookdown style, i.e. index.Rmd at the beginning, # (PART), # (APPENDIX) and # References as an upper level of normal # title
widget_name	The file name of the html widget to save.
width	the width of the markmap
height	the height of the markmap
elementId	character.
options	the markmap options
method	"regexpr" uses regular expressions, 'pandoc' uses pandoc to find the headings.

Details

For LinUx OS and mac OS, the 'tree' command must be pre-installed before using 'show_files = FALSE'.

- Linux: `sudo apt-get install tree`
- mac: install [Homebrew](#) first. Then in the terminal: `brew install tree`.

Value

A HTML widget object rendered from a given document.

Examples

```
## Not run:
### text -> widget
input <- c("# Chapter 1", "## Section 1.1", "## Section 1.2", "# Chapter 2")
mm(from = input, type = "text", root = "mindr")

### directory -> widget
input <- paste0(.libPaths(), "/mindr")[1]
mm(from = input, type = "dir")
mm(from = input, type = "dir", widget_name = "mindrtest.html")
### directory -> mm
mm(from = input, type = "dir", to = "test.mm")
### directory -> md
mm(from = input, type = "dir", to = "test.md")
### directory -> txt
mm(from = input, type = "dir", to = "test.txt")

### Rmd -> widget
input <- system.file("examples/r/rmd2r.Rmd", package = "mindr")
mm(from = input, type = "file", root = "mindr")
### Rmd -> r
mm(from = input, type = "file", root = "mindr", to = "test.r")
### Rmd -> mm
mm(from = input, type = "file", root = "mindr", to = "test.mm")

### mm -> widget
input <- system.file("examples/mm/bookdownplus.mm", package = "mindr")
mm(from = input, type = "file", root = "mindr")
### mm -> Rmd
mm(from = input, type = "file", root = "mindr", to = "test.Rmd")
### mm -> r
mm(from = input, type = "file", root = "mindr", to = "test.r")

### r -> widget
input <- system.file("examples/r/r2rmd.R", package = "mindr")
mm(from = input, type = "file", root = "mindr")
### r -> Rmd
mm(from = input, type = "file", root = "mindr", to = "test.Rmd")
### r -> mm
mm(from = input, type = "file", root = "mindr", to = "test.mm")

### The outline of the book Learning R
input <- system.file("examples/xuer/xuer.md", package = "mindr")
mm(from = input, type = "file", root = "Learning R", to = "learningr.mm")

## End(Not run)
```

`mm2md`*Convert a mind map (.mm) into markdown headers.*

Description

Convert a mind map (.mm) into markdown headers.

Usage

```
mm2md(pattern = "*.mm$", path = ".", savefile = TRUE, savefilename = "mindr.md",
       backup = TRUE)
```

Arguments

<code>pattern</code>	an optional regular expression for filtering the input files. See <code>help(dir)</code> .
<code>path</code>	character. The path of the folder which contains the input file(s).
<code>savefile</code>	logical. Whether to save the output as a markdown file.
<code>savefilename</code>	character. Valid when <code>savefile == TRUE</code> .
<code>backup</code>	logical. Whether the existing target file, if any, should be saved as backups.

Value

a vector of strings showing outline of a markdown document or book.

Examples

```
path <- system.file("examples/mm", package = "mindr")
mm2md(path = path)
```

`mm2r`*Convert .mm into a .R script*

Description

Convert .mm into a .R script

Usage

```
mm2r(filepattern = "*.mm$", path = ".", savefile = TRUE, savefilename = NA,
     backup = TRUE, heading = " -----")
```

Arguments

filepattern	the pattern of the file names
path	the path of the folder which contains the .Rmd or .md files
savefile	logical. Whether to save the output as a file.
savefilename	the destined file name
backup	logical. whether backup the existent file
heading	the indicator of the headings

Value

a .R script

Examples

```
path <- system.file("examples/mm", package = "mindr")
mm2r(path = path)
```

outline	<i>Extract headers of markdown or rmarkdown files as an outline.</i>
---------	--

Description

Extract headers of markdown or rmarkdown files as an outline.

Usage

```
outline(pattern = "*.[R]*md", path = ".", remove_curly_bracket = FALSE, savefile = TRUE,
  savefilename = "outline.md", backup = TRUE, bookdown_style = TRUE, keep_eq = FALSE,
  method = c("regexpr", "pandoc"))
```

Arguments

pattern	an optional regular expression for filtering the input files. See <code>help(dir)</code> .
path	character. The path of the folder which contains the input file(s).
remove_curly_bracket	logical. Whether to remove #ID in the headers of the markdown file (usually in a 'bookdown' https://github.com/rstudio/bookdown project).
savefile	logical. Whether to save the output as a markdown file.
savefilename	character. Valid when <code>savefile == TRUE</code> .
backup	logical. Whether the existing target file, if any, should be saved as backups.
bookdown_style	logical. whether the markdown files are in bookdown style, i.e. index.Rmd at the beginning, # (PART), # (APPENDIX) and # References as an upper level of normal # title
keep_eq	logical. whether to keep LaTeX equations.
method	"regexpr" uses regular expressions, 'pandoc' uses pandoc to find the headings.

Value

a vector of strings showing outline of a markdown document or book.

Examples

```
path <- system.file("examples/md", package = "mindr")
outline(path = path)
outline(path = path, remove_curly_bracket = TRUE)
```

r2md

Convert .R scripts into a .md/.Rmd file

Description

Convert .R scripts into a .md/.Rmd file

Usage

```
r2md(filepattern = "*.R$", path = ".", savefilename = NA, backup = TRUE, body = "#' ")
```

Arguments

filepattern	the pattern of the script file names
path	the path of the folder which contains the .R scripts
savefilename	the destined file name
backup	logical. whether backup the existent file
body	the indicator of the body text

Value

a markdown file

Examples

```
r2md()
```

r2mm *Convert .R scripts into a .mm file*

Description

Convert .R scripts into a .mm file

Usage

```
r2mm(filepattern = "*.R$", path = ".", title = NA, savefile = TRUE, savefilename = NA)
```

Arguments

filepattern	the pattern of the script file names
path	the path of the folder which contains the .R scripts
title	title of the mindmap
savefile	logical. Whether to save the output as a file.
savefilename	the destined file name

Value

an mindmap file

Examples

```
path <- system.file("examples/r", package = "mindr")
r2mm(path = path)
```

r2rmd *Convert .R scripts into a .Rmd file*

Description

Convert .R scripts into a .Rmd file

Usage

```
r2rmd(filepattern = "*.R$", savefile = TRUE, path = ".", savefilename = NA)
```

Arguments

filepattern	the pattern of the script file names
savefile	logical. Whether to save the output as a file.
path	the path of the folder which contains the .R scripts
savefilename	the destined file name

Value

an R markdown file

Examples

```
path <- system.file("examples/r", package = "mindr")
r2rmd(path = path)
```

rename2	<i>Rename a file automatically with a time stamp</i>
---------	--

Description

Rename a file automatically with a time stamp

Usage

```
rename2(filename, connect = "-")
```

Arguments

filename	character.
connect	the connecting character in the time stamp

Value

a new file name

renderMarkmap	<i>Shiny bindings for markmap</i>
---------------	-----------------------------------

Description

Render function for using markmap within Shiny applications and interactive Rmd documents. This function is taken from <https://github.com/seifer08ms/Rmarkmap>.

Usage

```
renderMarkmap(expr, env = parent.frame(), quoted = FALSE)
```

Arguments

expr	An expression that generates a markmap
env	The environment in which to evaluate expr.
quoted	Is expr a quoted expression (with quote())? This is useful if you want to save an expression in a variable.

rmd2r

Convert .md or .Rmd files into a .R script

Description

Convert .md or .Rmd files into a .R script

Usage

```
rmd2r(filepattern = "*.[R]*md$", path = ".", savefile = TRUE, savefilename = NA,  
      backup = TRUE, heading = " -----", chunkheading = FALSE)
```

Arguments

filepattern	the pattern of the file names
path	the path of the folder which contains the .Rmd or .md files
savefile	logical. Whether to save the output as a file.
savefilename	the destined file name
backup	logical. whether backup the existent file
heading	the indicator of the headings
chunkheading	logical. whether treat chunk options as headings (ending with ---)

Value

a .R script

Examples

```
path <- system.file("examples/r", package = "mindr")  
rmd2r(path = path)
```

rmvcode*check whether a digital number is within a given range*

Description

check whether a digital number is within a given range

Usage

```
rmvcode(index, loc)
```

Arguments

index	integer. a row number in a markdown file
loc	integer vector. the row numbers of the code block indicator, e.g. triple backticks

Value

logical.

tree	<i>Draw a mindmap of a directory</i>
------	--------------------------------------

Description

Draw a mindmap of a directory

Usage

```
tree(from = ".", to = NULL, root = NA, show_files = FALSE,
     widget_name = NA, width = NULL, height = NULL, elementId = NULL,
     options = markmapOption(preset = "colorful"))
```

Arguments

from	character. The path to the directory.
to	character. The path of the output file.
root	character. a string displayed as the root of the mind map
show_files	logical. Whether to show files in a directory.
widget_name	The file name of the html widget to save.
width	the width of the markmap
height	the height of the markmap
elementId	character.
options	the markmap options

Value

A HTML widget object rendered from a given document.

Examples

```
## Not run:
tree()
input <- system.file(package = "mindr")
tree(input)
tree(input, root = "mindr", show_files = TRUE)
tree(input, root = "mindr", show_files = TRUE, to = "mindr.mm")
tree(input, root = "mindr", show_files = TRUE, to = "mindr.md")
tree(input, root = "mindr", show_files = TRUE, to = "mindr.txt")

## End(Not run)
```

tree2mm

Convert a directory tree to a mindmap file.

Description

Convert a directory tree to a mindmap file.

Usage

```
tree2mm(tree, savefile = TRUE, savefilename = "mindr", backup = TRUE, n_root = 1)
```

Arguments

tree	character. The directory tree.
savefile	logical. Whether to save the output as a file.
savefilename	character. Valid when savefile == TRUE.
backup	logical. Whether the existing target file, if any, should be saved as backup.
n_root	numeric. Which element is the root of the tree.

Value

a mindmap file, which can be viewed by common mindmap software, such as 'FreeMind' (http://freemind.sourceforge.net/wiki/index.php/Main_Page) and 'XMind' (<http://www.xmind.net>).

Examples

```
et2 <- c("/Root name", "/Path A", "/Path A/Product A", "/Path A/Product A/Process A",
"/Path A/Product A/Process A/Step A", "/Path A/Product A/Process A/Step A/Record 1",
"/Path A/Product A/Process A/Step A/Record 1/Analyses",
"/Path A/Product A/Process A/Step A/Record 1/Analyses/Object 1",
"/Path A/Product A/Process A/Step A/Record 1/Analyses/Object 1/Type: data source",
"/Path A/Product A/Process A/Step A/Record 1/Analyses/Object 1/Version: 3",
"/Path A/Product A/Process A/Step A/Record 1/Analyses/Object 2",
"/Path A/Product A/Process A/Step A/Record 1/Analyses/Object 3",
```

```
"/Path A/Product A/Process A/Step A/Record 1/Setup Parts",
"/Path A/Product A/Process A/Step A/Record 1/Setup Parts/Par 1",
"/Path A/Product A/Process A/Step A/Record 1/Setup Parts/Par 2",
"/Path A/Product A/Process A/Step A/Record 1/Setup Parts/Par 3",
"/Path B", "/Path C")
tree2mm(et2)
```

`writelines2`*Write txt files avoiding overwriting existent files.*

Description

Write txt files avoiding overwriting existent files.

Usage

```
writelines2(text, filename, backup = TRUE)
```

Arguments

<code>text</code>	The text to write.
<code>filename</code>	The destined file name
<code>backup</code>	Logical.

Value

a txt file

Index

[count_space](#), 2

[dir2](#), 3

[dir4](#), 4

[get_body](#), 4

[get_eqloc](#), 5

[get_filename_ext](#), 5

[get_foldername](#), 6

[get_heading](#), 6

[get_heading2](#), 7

[get_heading3](#), 7

[markmap](#), 8

[markmapOption](#), 9

[markmapOutput](#), 10

[md2mm](#), 10

[md2r](#), 11

[mdtxt2mmtxt](#), 12

[mm](#), 13

[mm2md](#), 15

[mm2r](#), 15

[outline](#), 16

[r2md](#), 17

[r2mm](#), 18

[r2rmd](#), 18

[rename2](#), 19

[renderMarkmap](#), 19

[rmd2r](#), 20

[rmvcode](#), 20

[tree](#), 21

[tree2mm](#), 22

[writeLines2](#), 23