

# Package: mindr (via r-universe)

May 19, 2026

**Version** 1.3.3

**Date** 2021-11-22

**Title** Generate Mind Maps with R

**Author** Peng Zhao

**Maintainer** Peng Zhao <pzhao@pzhao.net>

**Depends** R (>= 3.0.0)

**Imports** htmlwidgets, knitr, rmarkdown, pdftools

**SystemRequirements** pandoc (>= 1.14) - <http://pandoc.org>

**Description** Convert Markdown ('.md') or R Markdown ('.Rmd') texts, R scripts, directory structures, and other hierarchical structured documents into mind map widgets or files ('.mm'), and vice versa. ``FreeMind" mind map ('.mm') files can be opened by or imported to common mindmap software such as 'FreeMind' (<[http://freemind.sourceforge.net/wiki/index.php/Main\\_Page](http://freemind.sourceforge.net/wiki/index.php/Main_Page)>).

**License** GPL-3

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

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

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.1.2

**NeedsCompilation** no

**Config/pak/sysreqs** cmake make libjpeg-dev libuv1-dev libssl-dev libpoppler-cpp-dev poppler-data

**Repository** <https://pzhaonet.r-universe.dev>

**Date/Publication** 2021-11-22 13:47:02 UTC

**RemoteUrl** <https://github.com/pzhaonet/mindr>

**RemoteRef** HEAD

**RemoteSha** 92d60d4b23154506522336f11208cbcea4d4e164

## Contents

dir2md . . . . .	2
dir2mm . . . . .	3
dir2r . . . . .	4
filterNULL . . . . .	5
get_eqloc . . . . .	6
get_filename_ext . . . . .	6
guess_type . . . . .	7
list2heading . . . . .	7
markmap . . . . .	8
markmapOption . . . . .	10
md2dir . . . . .	11
md2mm . . . . .	12
md2r . . . . .	13
mdtxt2mmtxt . . . . .	14
mm . . . . .	15
mm2dir . . . . .	20
mm2md . . . . .	21
mm2r . . . . .	22
mmm . . . . .	23
outline . . . . .	24
outline_pdf . . . . .	25
r2dir . . . . .	26
r2md . . . . .	27
r2mm . . . . .	28
rmvcode . . . . .	29
<b>Index</b>	<b>30</b>

---

dir2md

*Display a directory hierarchical structure in Markdown syntax*

---

### Description

Display a directory hierarchical structure in Markdown syntax

### Usage

```
dir2md(from = ".", dir_files = TRUE, dir_all = TRUE, dir_excluded = NA)
```

### Arguments

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
dir_files	Logical. Whether to include files. If FALSE, only folders are included. If TRUE, folders and files are included.

dir_all	Logical. Whether to include all files in a directory. If FALSE, only the names of visible files are included (following Unix-style visibility, that is files whose name does not start with a dot). If TRUE, all file names will be included.
dir_excluded	Character. The directories which are not included in the output.

**Value**

Character, in Markdown syntax.

**Examples**

```
input <- system.file(package = "mindr")
dir2md(input)
dir2md(input, dir_files = FALSE, dir_all = TRUE, dir_excluded = "Meta")
output_txt <- dir2md(input)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".Rmd")
writeLines(output_txt, output, useBytes = TRUE)
message("Input: ", input, "\nOutput: ", output)
# file.show(output) # Open the output file file.remove(output) # remove the
# output file
```

---

dir2mm

---

*Display hierarchical structure of a directory in FreeMind mind map*


---

**Description**

Display hierarchical structure of a directory in FreeMind mind map

**Usage**

```
dir2mm(
  from = ".",
  root = NA,
  dir_files = TRUE,
  dir_all = TRUE,
  dir_excluded = NA,
  md_maxlevel = ""
)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
root	Character. The string displayed as the root (center) of the mind map.
dir_files	Logical. Whether to include files. If FALSE, only folders are included. If TRUE, folders and files are included.

dir_all	Logical. Whether to include all files in a directory. If FALSE, only the names of visible files are included (following Unix-style visibility, that is files whose name does not start with a dot). If TRUE, all file names will be included.
dir_excluded	Character. The directories which are not included in the output.
md_maxlevel	Integer or ". The maximum level of the markdown headings that are displayed in the mind map.

### Value

FreeMind mind map code.

### Examples

```
input <- system.file(package = "mindr")
dir2mm(input)
dir2mm(input, dir_files = FALSE, dir_all = TRUE, dir_excluded = "Meta")

output_txt <- dir2mm(input)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".mm")
writeLines(output_txt, output, useBytes = TRUE)
message("Input: ", input, "\nOutput: ", output)
# file.show(output) # Open the output file file.remove(output) # remove the
# output file
```

---

dir2r

*Convert a hierarchical directory into R code*

---

### Description

Convert a hierarchical directory into R code

### Usage

```
dir2r(
  from = ".",
  dir_files = TRUE,
  dir_all = TRUE,
  dir_excluded = NA,
  r_seclabel = " -----",
  r_chunkheading = FALSE
)
```

### Arguments

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
dir_files	Logical. Whether to include files. If FALSE, only folders are included. If TRUE, folders and files are included.

dir_all	Logical. Whether to include all files in a directory. If FALSE, only the names of visible files are included (following Unix-style visibility, that is files whose name does not start with a dot). If TRUE, all file names will be included.
dir_excluded	Character. The directories which are not included in the output.
r_seclabel	Character. The ending characters indicating sections in R Markdown.
r_chunkheading	Logical. Whether process the chunk label as headings.

**Value**

Character, R code

**Examples**

```
input <- system.file(package = "mindr")
dir2r(input)
dir2r(input, dir_files = FALSE, dir_all = TRUE, dir_excluded = "Meta")

output_txt <- dir2r(input)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".R")
writeLines(output_txt, output, useBytes = TRUE)
message("Input: ", input, "\nOutput: ", output)
# system2('open', input) # Open the input dir in explorer. file.show(output) #
# Open the output file file.remove(output) # remove the output file
```

---

filterNULL

*A function for markmap*

---

**Description**

A function for markmap

**Usage**

```
filterNULL(x)
```

**Arguments**

x something

**Value**

something else

---

`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

---

guess_type	<i>Guess the type of input or output</i>
------------	--

---

**Description**

Guess the type of input or output

**Usage**

```
guess_type(from)
```

**Arguments**

from	The source text
------	-----------------

**Value**

the type, including 'dir', 'mindmap', 'R', 'markdown'.

---

list2heading	<i>convert lists to headings in a text</i>
--------------	--

---

**Description**

convert lists to headings in a text

**Usage**

```
list2heading(text)
```

**Arguments**

text	the given strings
------	-------------------

**Value**

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

---

 markmap

*Create a mind map in HTML widget*


---

## Description

Create a mind map in HTML widget

## Usage

```
markmap(
  from = ".",
  root = NA,
  input_type = c("auto", "markdown", "mindmap", "R", "dir"),
  md_list = FALSE,
  md_eq = FALSE,
  md_braces = FALSE,
  md_bookdown = FALSE,
  md_maxlevel = "",
  dir_files = TRUE,
  dir_all = TRUE,
  dir_excluded = NA,
  widget_name = NA,
  widget_width = NULL,
  widget_height = NULL,
  widget_elementId = NULL,
  widget_options = markmapOption(preset = "colorful")
)
```

## Arguments

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
root	Character. The string displayed as the root (center) of the mind map.
input_type	Character. The type of the input text. It can be 'auto', 'markdown', 'mindmap', 'R', 'dir'. The default value is 'auto', which means the type will be automatically assigned according to the features of the input text.
md_list	Logical. whether to process lists like headings in the Markdown input.
md_eq	Logical. Whether to include LaTeX equations in the Markdown input when converted to other formats.
md_braces	Logical. Whether to remove <b>#ID</b> in the headings of the markdown file (usually in a <b>bookdown</b> > project.
md_bookdown	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and #References as an upper level of the Level 1 heading.

md_maxlevel	Integer or <code>"</code> . The maximum level of the markdown headings that are displayed in the mind map.
dir_files	Logical. Whether to include files. If FALSE, only folders are included. If TRUE, folders and files are included.
dir_all	Logical. Whether to include all files in a directory. If FALSE, only the names of visible files are included (following Unix-style visibility, that is files whose name does not start with a dot). If TRUE, all file names will be included.
dir_excluded	Character. The directories which are not included in the output.
widget_name	Character. The name of the html widget.
widget_width	Numeric. The width of the widget.
widget_height	Numeric. The height of the widget.
widget_elementId	Character. The ID of the Widget.
widget_options	List. Options for the markmap widget. It should be a list passed from the <code>markmapOption()</code> function.

## Details

This function, adapted from the [Rmarkup](#) package, creates a markmap widget using `htmlwidgets`. The widget can be rendered on HTML pages generated from R Markdown, Shiny, or other applications.

## Value

HTML widget object.

## Examples

```
# Display Markdown:
input <- system.file("examples/mindr-md.Rmd", package = "mindr")
# file.show(input)
input_txt <- readLines(input, encoding = "UTF-8")
markmap(input_txt)
markmap(input_txt, root = basename(input), md_list = TRUE, md_eq = FALSE,
        md_braces = FALSE, md_bookdown = TRUE)

# Display Mind Map:
input <- system.file("examples/mindr-mm.mm", package = "mindr")
# file.show(input)
input_txt <- readLines(input, encoding = "UTF-8")
markmap(input_txt)
markmap(input_txt, root = basename(input))

# Display R script:
input <- system.file("examples/mindr-r.R", package = "mindr")
# file.show(input)
from <- input_txt <- readLines(input, encoding = "UTF-8")
markmap(input_txt)
markmap(input_txt, root = basename(input), md_list = TRUE, md_eq = FALSE,
```

```

md_braces = FALSE, md_bookdown = TRUE)

# Display directory:
input <- system.file(package = "mindr")
markmap(input)
markmap(input, root = "The mindr package", dir_files = FALSE, dir_excluded = c("Meta",
  "htmlwidgets/lib"), widget_elementId = "mindr-dir")

```

---

markmapOption

*Theme options for markmap creation*


---

## Description

Theme options for markmap creation

## 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 markmap. 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**

This function is adapted from the [Rmarkup](#) package.

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

**See Also**

<https://github.com/seifer08ms/Rmarkup> and <https://github.com/dundalek/markmap/blob/master/lib/view.mindmap.js> for details.

**Examples**

```
input <- system.file("examples/mindr-md.Rmd", package = "mindr")
# file.show(input)
input_txt <- readLines(input, encoding = "UTF-8")
markmap(input_txt)
markmap(input_txt, widget_options = markmapOption(preset = "default"))
markmap(input_txt, widget_options = markmapOption(color = "category20b",
  linkShape = "bracket"))
markmap(input_txt, widget_options = markmapOption(color = "category10",
  linkShape = "diagonal", renderer = "basic"))
markmap(input_txt, widget_options = markmapOption(nodeHeight = 30, nodeWidth = 100,
  spacingHorizontal = 60))
```

---

md2dir	<i>Create hierarchical directories according to (R) Markdown-syntax text</i>
--------	--

---

**Description**

Create hierarchical directories according to (R) Markdown-syntax text

**Usage**

```
md2dir(
  from = NA,
  dir_to = "mindr",
  md_list = FALSE,
  md_bookdown = TRUE,
  dir_quiet = FALSE
)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
dir_to	Character. The path of the output directory.

md_list	Logical. whether to process lists like headings in the Markdown input.
md_bookdown	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and # References as an upper level of the Level 1 heading.
dir_quiet	Logical. Whether to display the results of generated directories.

### Value

Directories generated.

### Examples

```
output <- file.path(tempdir(), "mindr")
md2dir(c("# a", "## a1", "## a2"), output)
message("Output: ", output)
# unlink(output, recursive = TRUE) # remove the output file

input <- system.file("examples/mindr-md.Rmd", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output <- file.path(tempdir(), "mindr")
md2dir(from = input_txt, output)
message("Input: ", input, "\nOutput: ", output)
# file.show(input) # Open the input file unlink(output, recursive = TRUE) #
# remove the output file

md2dir(from = input_txt, output, md_list = TRUE)
```

---

md2mm

*Convert (R) Markdown-syntax text to FreeMind mind map code*

---

### Description

Convert (R) Markdown-syntax text to FreeMind mind map code

### Usage

```
md2mm(
  from = NA,
  root = "mindr",
  md_list = FALSE,
  md_braces = FALSE,
  md_bookdown = FALSE,
  md_eq = FALSE,
  md_maxlevel = ""
)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
root	Character. The string displayed as the root (center) of the mind map.
md_list	Logical. whether to process lists like headings in the Markdown input.
md_braces	Logical. Whether to remove #ID in the headings of the markdown file (usually in a <code>bookdown</code> > project).
md_bookdown	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and #References as an upper level of the Level 1 heading.
md_eq	Logical. Whether to include LaTeX equations in the Markdown input when converted to other formats.
md_maxlevel	Integer or ". The maximum level of the markdown headings that are displayed in the mind map.

**Value**

FreeMind mind map code, which can be saved as a .mm file and viewed by common mind map software, such as `FreeMind` and `XMind`.

**Examples**

```
input <- system.file("examples/mindr-md.Rmd", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output_txt <- md2mm(input_txt)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".mm")
writeLines(output_txt, output, useBytes = TRUE)
# file.show(input) # Open the input file file.show(output) # Open the output
# file
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file
```

---

md2r

---

*Convert (R) Markdown-syntax text into R code*


---

**Description**

Convert (R) Markdown-syntax text into R code

**Usage**

```
md2r(from = NA, r_seclabel = " -----", r_chunkheading = FALSE)
```

**Arguments**

`from` Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.

`r_seclabel` Character. The ending characters indicating sections in R Markdown.

`r_chunkheading` Logical. Whether process the chunk label as headings.

**Value**

Character, R code.

**Examples**

```
input <- system.file("examples/mindr-md.Rmd", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output_txt <- md2r(from = input_txt)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".R")
writeLines(output_txt, output, useBytes = TRUE)
# file.show(input) # Open the input file file.show(output) # Open the output
# file
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file
```

---

mdtxt2mmtxt

---

*Convert Markdown text to FreeMind mind map text.*


---

**Description**

Convert Markdown text to FreeMind mind map text.

**Usage**

```
mdtxt2mmtxt(from = "", root = "root", md_eq = FALSE)
```

**Arguments**

`from` Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.

`root` Character. The string displayed as the root (center) of the mind map.

`md_eq` Logical. Whether to include LaTeX equations in the Markdown input when converted to other formats.

**Value**

a mindmap text.

---

mm	<i>All-in-one wrapper for the conversion between (R) Markdown, FreeMind mind map, R code, directory structure, and HTML widget.</i>
----	---

---

### Description

All-in-one wrapper for the conversion between (R) Markdown, FreeMind mind map, R code, directory structure, and HTML widget.

### Usage

```
mm(
  from = NA,
  input_type = c("auto", "markdown", "mindmap", "R", "dir"),
  output_type = c("widget", "mindmap", "markdown", "R", "dir"),
  root = NA,
  md_list = FALSE,
  md_eq = FALSE,
  md_braces = FALSE,
  md_bookdown = FALSE,
  md_maxlevel = "",
  r_seclabel = " -----",
  r_chunkheading = FALSE,
  dir_files = TRUE,
  dir_all = TRUE,
  dir_excluded = NA,
  dir_to = NA,
  dir_quiet = FALSE,
  widget_name = NA,
  widget_width = NULL,
  widget_height = NULL,
  widget_elementId = NULL,
  widget_options = markmapOption(preset = "colorful")
)
```

### Arguments

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
input_type	Character. The type of the input text. It can be 'auto', 'markdown', 'mindmap', 'R', 'dir'. The default value is 'auto', which means the type will be automatically assigned according to the features of the input text.
output_type	Character. The type of the output. It can be 'widget', 'mindmap', 'markdown', 'R', 'dir'. The default value is 'widget'.
root	Character. The string displayed as the root (center) of the mind map.
md_list	Logical. whether to process lists like headings in the Markdown input.

md_eq	Logical. Whether to include LaTeX equations in the Markdown input when converted to other formats.
md_braces	Logical. Whether to remove #ID in the headings of the markdown file (usually in a <code>bookdown</code> project).
md_bookdown	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and #References as an upper level of the Level 1 heading.
md_maxlevel	Integer or ". The maximum level of the markdown headings that are displayed in the mind map.
r_seclabel	Character. The ending characters indicating sections in R Markdown.
r_chunkheading	Logical. Whether process the chunk label as headings.
dir_files	Logical. Whether to include files. If FALSE, only folders are included. If TRUE, folders and files are included.
dir_all	Logical. Whether to include all files in a directory. If FALSE, only the names of visible files are included (following Unix-style visibility, that is files whose name does not start with a dot). If TRUE, all file names will be included.
dir_excluded	Character. The directories which are not included in the output.
dir_to	Character. The path of the output directory.
dir_quiet	Logical. Whether to display the results of generated directories.
widget_name	Character. The name of the html widget.
widget_width	Numeric. The width of the widget.
widget_height	Numeric. The height of the widget.
widget_elementId	Character. The ID of teh Widget.
widget_options	List. Options for the markmap widget. It should be a list passed from the <code>markmapOption()</code> function.

## Details

`mm()` converts between (R) Markdown syntax text, R code, FreeMind mind map code, and directory, and display them in a HTML widget. It is a wrapper for other conversion functions in this package.

## Value

Desired output.

## Examples

```
##### Example 1: From Markdown to
##### other outputs #####

## Source document ####
input <- system.file("examples/mindr-md.Rmd", package = "mindr")

## file.show(input) # Open the input file with the default program, if any
```

```

input_txt <- readLines(input, encoding = "UTF-8")

## Convert to mind map text, markdown outline, R script, and HTML widget ####
mm_output <- mm(input_txt, output_type = c("mindmap", "markdown", "R", "widget"))
mm_output

## Save the output texts as files #####

### mind map ####
output <- tempfile(pattern = "file", tmpdir = tmpdir(), fileext = ".mm")
writeLines(mm_output$mindmap, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program, if any
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### markdown outline ####
output <- tempfile(pattern = "file", tmpdir = tmpdir(), fileext = ".md")
writeLines(mm_output$markdown, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program, if any
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### R script ####
output <- tempfile(pattern = "file", tmpdir = tmpdir(), fileext = ".R")
writeLines(mm_output$r, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program, if any
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### Widget ##### output <- tempfile(pattern = 'file', tmpdir = tmpdir(),
### fileext = '.html') htmlwidgets::saveWidget(mm_output$widget, file =
### output) file.show(output) # Open the output file with the default program,
### if any message('Input: ', input, '\nOutput: ', output) file.remove(output)
### # remove the output file

## Generate directory according to the source document #####
temp_dir <- file.path(tmpdir(), "mindr")
mm_output <- mm(input_txt, output_type = "dir", root = "mindr", md_list = TRUE,
  md_braces = TRUE, md_bookdown = TRUE, dir_to = temp_dir)
# system2('open', temp_dir) # Open the generated directory unlink(temp_dir,
# recursive = TRUE) # remove the generated directory

## More arguments #####
mm_output <- mm(input_txt, output_type = c("mindmap", "markdown", "R", "widget"),
  root = "mindr", md_list = TRUE, md_braces = TRUE, md_bookdown = TRUE)
mm_output

##### Example 2: From mind map to
##### other outputs #####

## Source document ####
input <- system.file("examples/mindr-mm.mm", package = "mindr")

```

```

## file.show(input) # Open the input file with the default program, if any
input_txt <- readLines(input, encoding = "UTF-8")

## Convert markdown outline, R script, and HTML widget ####
mm_output <- mm(input_txt, output_type = c("markdown", "R", "widget"))
mm_output

## Save the output texts as files ####

### markdown outline ####
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".md")
writeLines(mm_output$markdown, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### R script ####
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".R")
writeLines(mm_output$r, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### Widget ####
output <- tempfile(pattern = 'file', tmpdir = tempdir(),
### fileext = '.html') htmlwidgets::saveWidget(mm_output$widget, file =
### output) file.show(output) # Open the output file with the default program,
### if any message('Input: ', input, '\nOutput: ', output) file.remove(output)
### # remove the output file

## Generate directory according to the source document ####
temp_dir <- file.path(tempdir(), "mindr")
mm_output <- mm(input_txt, output_type = "dir", root = "mindr", dir_to = temp_dir)
# system2('open', temp_dir) # Open the generated directory unlink(temp_dir,
# recursive = TRUE) # remove the generated directory

##### Example 3: From R script to
##### other outputs #####

## Source document ####
input <- system.file("examples/mindr-r.R", package = "mindr")

## file.show(input) # Open the input file with the default program, if any
input_txt <- readLines(input, encoding = "UTF-8")

## Convert to mind map text, markdown text, and HTML widget ####
mm_output <- mm(input_txt, output_type = c("mindmap", "markdown", "widget"))
mm_output

## Save the output texts as files ####

### mind map ####
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".mm")
writeLines(mm_output$mindmap, output, useBytes = TRUE)

```

```

# file.show(output) # Open the output file with the default program, if any
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### R markdown ####
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".Rmd")
writeLines(mm_output$markdown, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program, if any
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### Widget #### output <- tempfile(pattern = 'file', tmpdir = tempdir(),
### fileext = '.html') htmlwidgets::saveWidget(mm_output$widget, file =
### output) file.show(output) # Open the output file with the default program,
### if any message('Input: ', input, '\nOutput: ', output) file.remove(output)
### # remove the output file

## Generate directory according to the source document ####
temp_dir <- file.path(tempdir(), "mindr")
mm_output <- mm(input_txt, output_type = "dir", root = "mindr", dir_to = temp_dir)
# system2('open', temp_dir) # Open the generated directory unlink(temp_dir,
# recursive = TRUE) # remove the generated directory

##### Example 4: From directory to
##### other outputs #####

## Source directory ####
input <- system.file(package = "mindr")

## Convert to mind map text, markdown outline, R script, and HTML widget ####
mm_output <- mm(input, output_type = c("mindmap", "markdown", "R", "widget"))
mm_output

## Save the output texts as files ####

### mind map ####
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".mm")
writeLines(mm_output$mindmap, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program, if any
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### markdown outline ####
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".md")
writeLines(mm_output$markdown, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program, if any
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### R script ####
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".R")
writeLines(mm_output$r, output, useBytes = TRUE)
# file.show(output) # Open the output file with the default program, if any

```

```

message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file

### Widget #### output <- tempfile(pattern = 'file', tmpdir = tmpdir(),
### fileext = '.html') htmlwidgets::saveWidget(mm_output$widget, file =
### output) file.show(output) # Open the output file with the default program,
### if any message('Input: ', input, '\nOutput: ', output) file.remove(output)
### # remove the output file

## Clone the source directory ####
temp_dir <- file.path(tmpdir(), "mindr")
mm_output <- mm(input, output_type = "dir", dir_to = temp_dir)
# system2('open', temp_dir) # Open the generated directory unlink(temp_dir,
# recursive = TRUE) # remove the generated directory

##### Example 5: From any format to
##### mind map #####

# With the help of pandoc, you can display the outline of any documents that
# pandoc can convert to Markdown.

# # HTML: here we use the R-FQA webpage myurl <-
# 'https://cran.r-project.org/doc/FAQ/R-FAQ.html' input <- tempfile(pattern =
# 'file', tmpdir = tmpdir()) markdown_temp <- tempfile(pattern = 'file',
# tmpdir = tmpdir(), fileext = '.md') download.file(myurl, destfile = input,
# method = 'curl') rmarkdown::pandoc_convert(input, to = 'markdown', output =
# markdown_temp) input_txt <- readLines(markdown_temp, encoding = 'UTF-8')
# mindr::mm(input_txt)

# # MS Word: here we use a .docx document shipped by the 'officer' package
# input <- system.file('doc_examples/example.docx', package = 'officer')
# markdown_temp <- tempfile(pattern = 'file', tmpdir = tmpdir(), fileext =
# '.md') rmarkdown::pandoc_convert(input, to = 'markdown', output =
# markdown_temp) input_txt <- readLines(markdown_temp, encoding = 'UTF-8')
# mindr::mm(input_txt, md_list = TRUE)

```

---

mm2dir

---

*Generate hierarchical directories according to a FreeMind mind map*


---

## Description

Generate hierarchical directories according to a FreeMind mind map

## Usage

```
mm2dir(from = NA, dir_to = NA, dir_quiet = FALSE)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
dir_to	Character. The path of the output directory.
dir_quiet	Logical. Whether to display the results of generated directories.

**Value**

Directory generated.

**Examples**

```
input <- system.file("examples/mindr-mm.mm", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output <- file.path(tempdir(), "mindr")
mm2dir(input_txt, output)
message("Input: ", input, "\nOutput: ", output)
# file.show(input) # Open the input file system2('open', output) # Open the
# output dir in explorer. unlink(output, recursive = TRUE) # remove the output
# file
```

---

mm2md

---

*Convert FreeMind mind map code into Markdown headings*


---

**Description**

Convert FreeMind mind map code into Markdown headings

**Usage**

```
mm2md(from = NA)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
------	---

**Value**

Character, showing outline in Markdown syntax.

**Examples**

```
input <- system.file("examples/mindr-mm.mm", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output_txt <- mm2md(input_txt)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".md")
writeLines(output_txt, output, useBytes = TRUE)
# file.show(input) # Open the input file file.show(output) # Open the output
# file
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file
```

mm2r

*Convert FreeMind mind map code into .R code***Description**

Convert FreeMind mind map code into .R code

**Usage**

```
mm2r(from = NA, r_seclabel = " -----", r_chunkheading = FALSE)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
r_seclabel	Character. The ending characters indicating sections in R Markdown.
r_chunkheading	Logical. Whether process the chunk label as headings.

**Value**

Character, R code.

**Examples**

```
input <- system.file("examples/mindr-mm.mm", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output_txt <- mm2r(input_txt)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".R")
writeLines(output_txt, output, useBytes = TRUE)
# file.show(input) # Open the input file file.show(output) # Open the output
# file
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file
```

---

 mmm

 Convert almost any file into mind map.
 

---

## Description

Convert almost any file into mind map.

## Usage

```
mmm(
  input_file = NA,
  output_type = c("widget", "mindmap", "markdown"),
  root = NA,
  md_list = FALSE,
  md_eq = FALSE,
  md_braces = FALSE,
  md_bookdown = FALSE,
  md_maxlevel = "",
  r_seclabel = " -----",
  r_chunkheading = FALSE,
  widget_name = NA,
  widget_width = NULL,
  widget_height = NULL,
  widget_elementId = NULL,
  widget_options = markmapOption(preset = "colorful")
)
```

## Arguments

<code>input_file</code>	Character. The path to the file for input.
<code>output_type</code>	Character. The type of the output. It can be 'widget', 'mindmap', 'markdown', 'R', 'dir'. The default value is 'widget'.
<code>root</code>	Character. The string displayed as the root (center) of the mind map.
<code>md_list</code>	Logical. whether to process lists like headings in the Markdown input.
<code>md_eq</code>	Logical. Whether to include LaTeX equations in the Markdown input when converted to other formats.
<code>md_braces</code>	Logical. Whether to remove <b>#ID</b> in the headings of the markdown file (usually in a <b>bookdown</b> > project).
<code>md_bookdown</code>	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and #References as an upper level of the Level 1 heading.
<code>md_maxlevel</code>	Integer or "". The maximum level of the markdown headings that are displayed in the mind map.
<code>r_seclabel</code>	Character. The ending characters indicating sections in R Markdown.

`r_chunkheading` Logical. Whether process the chunk label as headings.  
`widget_name` Character. The name of the html widget.  
`widget_width` Numeric. The width of the widget.  
`widget_height` Numeric. The height of the widget.  
`widget_elementId`  
 Character. The ID of teh Widget.  
`widget_options` List. Options for the markmap widget. It should be a list passed from the `markmapOption()` function.

### Details

The input file type could be `.md`, `.Rmd`, `.R`, `.mm`, `.pdf`, `.docx`, `.html`, `.rtf`, `.odt`, `.epub`, `.tex`, and any other types which pandoc can convert from. See [pandoc](#) for more details.

### Value

Desired output.

### Examples

```

# pdf: here we use the pdf files in the R installation
input <- file.path(R.home(), "doc/manual/R-intro.pdf")
mmm(input)
input <- file.path(R.home(), "doc/manual/R-data.pdf")
mmm(input, root = "R-data", md_maxlevel = 2)

# MS Word: here we use a .docx document shipped by the 'officer' package
input <- system.file("doc_examples/example.docx", package = 'officer')
mmm(input)

# HTML: here we use the R-FQA webpage
input <- file.path(R.home(), "doc/html/rw-FAQ.html")
mmm(input)

```

---

outline

*Extract headings of (R) Markdown-syntax text as an outline*

---

### Description

Extract headings of (R) Markdown-syntax text as an outline

### Usage

```

outline(
  from,
  md_list = FALSE,
  md_eq = FALSE,
  md_braces = FALSE,

```

```

    md_bookdown = FALSE,
    md_maxlevel = ""
  )

```

### Arguments

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
md_list	Logical. whether to process lists like headings in the Markdown input.
md_eq	Logical. Whether to include LaTeX equations in the Markdown input when converted to other formats.
md_braces	Logical. Whether to remove <b>#ID</b> in the headings of the markdown file (usually in a <b>bookdown</b> > project).
md_bookdown	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and # References as an upper level of the Level 1 heading.
md_maxlevel	Integer or ". The maximum level of the markdown headings that are displayed in the mind map.

### Value

Character, showing the outline.

### Examples

```

input <- system.file("examples/mindr-md.Rmd", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
outline(input_txt)
outline(input_txt, md_list = TRUE, md_bookdown = TRUE)
outline(input_txt, md_list = TRUE, md_bookdown = TRUE, md_maxlevel = 2)

```

---

outline\_pdf

*Extract the outline from pdf toc, and output as Markdown*

---

### Description

Extract the outline from pdf toc, and output as Markdown

### Usage

```
outline_pdf(input_toc)
```

### Arguments

input_toc	Character. The table of contents (TOC) of a pdf file, extracted by pdf tools: :pdf_toc().
-----------	---

**Value**

Character, showing the TOC in Markdown.

**Examples**

```
input <- file.path(R.home(), "doc/manual/R-intro.pdf")
input_toc <- pdftools::pdf_toc(input)
outline_pdf(input_toc)
```

---

r2dir

*Generate hierarchical directories according to the outline of R code*

---

**Description**

Generate hierarchical directories according to the outline of R code

**Usage**

```
r2dir(
  from = NA,
  dir_to = NA,
  md_list = FALSE,
  md_bookdown = TRUE,
  dir_quiet = FALSE
)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
dir_to	Character. The path of the output directory.
md_list	Logical. whether to process lists like headings in the Markdown input.
md_bookdown	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and #References as an upper level of the Level 1 heading.
dir_quiet	Logical. Whether to display the results of generated directories.

**Value**

Directory generated.

## Examples

```
input <- system.file("examples/mindr-r.R", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output <- file.path(tempdir(), "mindr")
r2dir(input_txt, output)
message("Input: ", input, "\nOutput: ", output)
# file.show(input) # Open the input file system2('open', output) # Open the
# output dir in explorer. unlink(output, recursive = TRUE) # remove the output
# file
```

---

r2md

*Convert R code into (R) Markdown-syntax text*

---

## Description

Convert R code into (R) Markdown-syntax text

## Usage

```
r2md(from = NA)
```

## Arguments

**from** Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.

## Value

R markdown-syntax text.

## Examples

```
input <- system.file("examples/mindr-r.R", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output_txt <- r2md(from = input_txt)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".Rmd")
writeLines(output_txt, output, useBytes = TRUE)
# file.show(input) # Open the input file file.show(output) # Open the output
# file
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file
```

r2mm

*Convert R code into FreeMind mind map code***Description**

Convert R code into FreeMind mind map code

**Usage**

```
r2mm(
  from = NA,
  root = NA,
  md_list = FALSE,
  md_braces = FALSE,
  md_bookdown = FALSE,
  md_eq = FALSE,
  md_maxlevel = ""
)
```

**Arguments**

from	Character. The source text of the (R) Markdown syntax text, the R code, the FreeMind mind map code, or the path to the directory.
root	Character. The string displayed as the root (center) of the mind map.
md_list	Logical. whether to process lists like headings in the Markdown input.
md_braces	Logical. Whether to remove <b>#ID</b> in the headings of the markdown file (usually in a <b>bookdown</b> > project).
md_bookdown	Logical. Whether the R Markdown syntax text is in bookdown style, i.e. # (PART), # (APPENDIX), and # References as an upper level of the Level 1 heading.
md_eq	Logical. Whether to include LaTeX equations in the Markdown input when converted to other formats.
md_maxlevel	Integer or ". The maximum level of the markdown headings that are displayed in the mind map.

**Value**

Character, FreeMind mind map code.

**Examples**

```
input <- system.file("examples/mindr-r.R", package = "mindr")
input_txt <- readLines(input, encoding = "UTF-8")
output_txt <- r2mm(from = input_txt)
output <- tempfile(pattern = "file", tmpdir = tempdir(), fileext = ".mm")
writeLines(output_txt, output, useBytes = TRUE)
```

```
# file.show(input) # Open the input file file.show(output) # Open the output
# file
message("Input: ", input, "\nOutput: ", output)
# file.remove(output) # remove the output file
```

---

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 backsticks

### **Value**

logical.

# Index

[dir2md](#), [2](#)  
[dir2mm](#), [3](#)  
[dir2r](#), [4](#)

[filterNULL](#), [5](#)

[get\\_eqloc](#), [6](#)  
[get\\_filename\\_ext](#), [6](#)  
[guess\\_type](#), [7](#)

[list2heading](#), [7](#)

[markmap](#), [8](#)  
[markmapOption](#), [10](#)  
[md2dir](#), [11](#)  
[md2mm](#), [12](#)  
[md2r](#), [13](#)  
[mdtxt2mmtxt](#), [14](#)  
[mm](#), [15](#)  
[mm2dir](#), [20](#)  
[mm2md](#), [21](#)  
[mm2r](#), [22](#)  
[mmm](#), [23](#)

[outline](#), [24](#)  
[outline\\_pdf](#), [25](#)

[r2dir](#), [26](#)  
[r2md](#), [27](#)  
[r2mm](#), [28](#)  
[rmvcode](#), [29](#)