Package: simplermarkdown (via r-universe)

September 10, 2024

Title Simple Engine for Generating Reports using R

Version 0.0.6

Description Runs R-code present in a pandoc markdown file and includes the resulting output in the resulting markdown file. This file can then be converted into any of the output formats supported by pandoc. The package can also be used as an engine for writing package vignettes.

BugReports https://github.com/djvanderlaan/simplermarkdown/issues

URL https://github.com/djvanderlaan/simplermarkdown

License GPL (>= 3)

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.2.3

Imports rjson, tools

Suggests MASS

VignetteBuilder simplermarkdown

SystemRequirements Pandoc (http://pandoc.org) needs to installed and available in the search path.

Repository https://djvanderlaan.r-universe.dev

RemoteUrl https://github.com/djvanderlaan/simplermarkdown

RemoteRef HEAD

RemoteSha 7150dd5507dae611df14963093009e2bb548af53

Contents

`file_subs_ext` *Replace file extension by another file extension*

Description

Replace file extension by another file extension

Usage

```
file_subs_ext(fn, new_ext, check = TRUE)
```

Arguments

`fn`	character vector with file names
`new_ext`	character vector of length one with the new extension (when it does not start with a period a period is added).
`check`	check if the new file name is not equal to the original filename. If so, generate an error.

Value

Returns a character vector of the same length of `fn` with the extension of the file names in `fn` replaced by `new_ext`.

`format_traditional` *Format the result of running a block of code*

Description

Format the result of running a block of code

Usage

```
format_traditional(x)

format_copypaste(x)
```

Arguments

x	the result of running the code. See 'details' for the format.

Details

The input vector `x` should be a list. Each element of this list should be a list with two elements: `input` and `output`. `input` should contain the evaluated expression; this can be more than one line. `output` should contain the output of the evaluation. When there is no output this is character vector of length 0.

Value

A character vector of length 1 with the formatted code.

`get_extensions` *Get the extensions that are needed when converting from json to markdown by pandoc.*

Description

Get the extensions that are needed when converting from json to markdown by pandoc.

Usage

```
get_extensions()
```

Details

Checks which extensions are available in pandoc and returns a character string disabling extensions that interfere with the conversion of the processed document (in json format) back to markdown. At the moment only one extension is disabled.

`raw_attribute` This extension is disabled. Some of the output functions write raw markdown blocks. This extension only recognises html and tex raw blocks. Raw markdown blocks are ignored when converting to other formats.

Value

Returns a character vector of the form `"+extension1-extension2"` with the extensions that are to be enabled (`+`) or disables (`-`). The function calls `pandoc` to check which extensions are available.

markdown_block — *Return a code block object that can be included in the pandoc parse tree*

Description

Return a code block object that can be included in the pandoc parse tree

Usage

```
markdown_block(content, language, id = "", ...)
```

Arguments

content	a character vector containing the code
language	language of the code in the code block
id	optional id of the code block
...	additional arguments should be named. These are added to the markdown block as additional arguments.

Value

Returns a `list` with the correct structure for a code block in the pandoc parse tree.

mdtangle — *Extract code from the code blocks in a markdown file*

Description

Extract code from the code blocks in a markdown file

Usage

```
mdtangle(
  fn,
  ofn = file_subs_ext(basename(fn), ".R"),
  extra_arguments = "",
  cmd = "pandoc %3$s -s \"%1$s\" -t json -o \"%2$s\"",
  ...
)
```

Arguments

fn	filename of the markdown file (should use pandoc markdown).
ofn	name of the resulting R-script
extra_arguments	extra arguments passed on to pandoc. Should be a length 1 character vector.
cmd	command used to run pandoc. See details.
...	ignored

Details

mdtangle calls pandoc. Pandoc will parse the markdown document and write the parsed file to temporary file. This file is read by mdtangle and the code is extracted from it and written to ofn.

Using the cmd argument the exact command used to run pandoc can be modified. It is passed on to sprintf and uses positional arguments: (1) name of the input file, (2) location of the temporary file to which the parsed document is written, (3) the value of extra_arguments.

Value

Returns the filename of the generated file.

---

mdweave *Run the code in a markdown file and generate a new markdown file*

Description

Run the code in a markdown file and generate a new markdown file

Usage

```
mdweave(
  fn,
  ofn = file_subs_ext(basename(fn), ".md", FALSE),
  cmd1 = "pandoc -s \"%1$s\" -t json -o \"%2$s\"",
  cmd2 = "pandoc -s \"%1$s\" -t markdown%3$s -o \"%2$s\"",
  ...
)
```

Arguments

fn	filename of the markdown file (should use pandoc markdown).
ofn	name of the resulting markdown file.
cmd1	command used to run pandoc. See details.
cmd2	command used to run pandoc. See details.
...	ignored

Details

mdweave calls pandoc twice. In the first call the markdown file is parsed by pandoc and the parse tree is written to a temporary file. This parse tree is the read by mdweave and any R-code in the tree is executed resulting in a modified parse tree. This file is then stored to a new temporary file. Pandoc is the called a second time to convert the new parse tree to a markdown file.

The arguments cmd1 and cmd2 contain the calls used to run pandoc. The arguments can be used to, for example pas additional arguments to pandoc. They use positional arguments. In cmd1, the first argument (%1$s) is the input file name and the second (%2$s) the temporary file containing the parsed tree. In cmd2, the first argument is the temporary file with the modified parse tree and the second argument the output file.

cmd2 also has a third argument (%3$s) that contains a list of extensions that are enabled or disabled. This is because some extensions interfere with the conversion of the parsed tree to markdown. See get_extensions to see which extensions are disabled.

Value

Returns the file name of the file generated (ofn). Called mainly for the side effect of parsing and generating a markdown file (and possibly secondary files such as figures).

---

mdweave_to_pdf *Run the code in a markdown file and generate a new document*

Description

Run the code in a markdown file and generate a new document

Usage

```
mdweave_to_pdf(
  fn,
  ofn = file_subs_ext(basename(fn), ".pdf", FALSE),
  extra_arguments2 = "--self-contained",
  run_in_temp = TRUE,
  cmd2 = "pandoc %3$s -s \"%1$s\" -t latex -o \"%2$s\"",
  ...
)

mdweave_to_tex(
  fn,
  ofn = file_subs_ext(basename(fn), ".tex", FALSE),
  extra_arguments2 = "--self-contained",
  run_in_temp = TRUE,
  cmd2 = "pandoc %3$s -s \"%1$s\" -t latex -o \"%2$s\"",
  ...
)
```

```
mdweave_to_html(
  fn,
  ofn = file_subs_ext(basename(fn), ".html", FALSE),
  extra_arguments2 = "--self-contained",
  run_in_temp = TRUE,
  cmd2 = "pandoc %3$s -s \"%1$s\" -t html -o \"%2$s\"",
  ...
)
```

Arguments

fn	filename of the markdown file (should use pandoc markdown).
ofn	name of the resulting file.
extra_arguments2	extra arguments passed on to pandoc. Should be a length 1 character vector.
run_in_temp	When TRUE the intermediary markdown file and generated figures (when not using custom paths) are created in a temporary directory. Otherwise these will be generated in the same directory as the output file.
cmd2	command used to run pandoc. See details.
...	additional arguments are passed on to mdweave.

Details

These functions first call mdweave to run the code in the original file and convert the original markdown file to a new markdown file. This second markdown file is then converted to the desired output format using a second run of pandoc.

In case of converting to pdf the file is required to have the extension .pdf. In case of converting to LaTeX, the file cannot have the extension .pdf. That is because in both cases the file is first converted to LaTeX. In case of a file with the extension .pdf the file is than further converted to PDF.

Value

Returns the name of the resulting outout file.

---

md_figure *Generate a figure and generate the markdown to include the figure*

Description

Will evaluate the expressions in expr and capture the output on the given plotting device in the given file. It will then generate the markdown needed to include that figure in a markdown document.

Usage

```
md_figure(
  expr,
  name,
  caption = "",
  id = "",
  dir = file.path(Sys.getenv("MDOUTDIR", "."), "figures"),
  device = c("png", "pdf"),
  ...,
  as_character = FALSE,
  echo = FALSE,
  results = FALSE,
  formatter = getOption("md_formatter", default = format_traditional),
  capture_warnings = FALSE,
  capture_messages = results,
  muffle_warnings = FALSE,
  muffle_messages = TRUE
)
```

Arguments

expr	the expressions to evaluate. Will generally contain plotting commands. The expressions are evaluated in the global environment.
name	the name of the figure.
caption	text of the caption. When omitted no caption is added to the figure.
id	id of the figure. When omitted or equal to NULL or an empty character, no id is added to the figure.
dir	name of the directory in which to store the file.
device	the graphics device to use for creating the image.
...	passed on to the graphics device.
as_character	return the figure as a character vector. If FALSE the figure will be written to the standard output.
echo	the code in code is repeated in the output.
results	include the results of running the code in the output. The output of code that explicitly writes to standard output is always included.
formatter	function that will format the R-code and resulting output (if requested). See format_traditional for possible options.
capture_warnings	include warnings in the output.
capture_messages	include messages in the output.
muffle_warnings	do not show warnings in the console.
muffle_messages	do not show messages in the console.

Details

The image is stored in the file `dir/name.device`.

Value

When `as_character = FALSE` a character vector with the markdown needed to include the generated figure in a markdown file is returned. Otherwise, nothing is returned; the markdown is written to the console.

`md_table` *Generate a markdown table from a data.frame*

Description

Generate a markdown table from a data.frame

Usage

```
md_table(tab, caption, as_character = FALSE, ...)
```

Arguments

`tab`	a data frame
`caption`	text of the caption. When omitted no caption is added to the table.
`as_character`	return the table as a character vector. If `FALSE` the table will be written to the standard output.
`...`	unused.

Value

Then `as_character = FALSE` a character vector with the markdown containing the table is returned. Otherwise, nothing is returned; the markdown is then written to the console.

`output_table` *Output filters for code blocks in markdown*

Description

Output filters for code blocks in markdown

Usage

```
output_table(code, language = "R", id = "", ...)

output_figure(code, language = "R", id = "", ...)

output_eval(
  code,
  language = "R",
  id = "",
  echo = TRUE,
  results = TRUE,
  drop_empty = TRUE,
  eval = TRUE,
  formatter = getOption("md_formatter", default = format_traditional),
  capture_warnings = FALSE,
  capture_messages = results,
  muffle_warnings = FALSE,
  muffle_messages = TRUE,
  ...
)

output_raw(code, language = "R", id = "", ...)

output_str(code, language = "R", id = "", ...)

output_shell(
  code,
  language,
  id = "",
  cmd = language,
  echo = TRUE,
  results = TRUE,
  comment_char = "# ",
  ...
)
```

Arguments

code	character vector containing the code of the code block
language	the language in which the code is written (as specified in the markdown file).
id	the identifier of the code block.
...	additional arguments specified as arguments in the code block are passed on to the filter function. Often these are ignored, or they are passed on to other functions (see 'Details').
echo	the code in code is repeated in the output.
results	include the results of running the code in the output. The output of code that explicitly writes to standard output is always included.

drop_empty
: do not include any output if the resulting code block would be empty.

eval
: if FALSE do not run the code and just include the code in the output.

formatter
: function that will format the R-code and resulting output (if requested). See format_traditional for possible options.

capture_warnings
: include warnings in the output.

capture_messages
: include messages in the output.

muffle_warnings
: do not show warnings in the console.

muffle_messages
: do not show messages in the console.

cmd
: the command to use in the system2 call. Only needed when different than the language.

comment_char
: string prepended to output of commands run by output_shell.

Details

The filter functions output_table and output_figure call md_table and md_figure respectively; additional arguments are passed on to those functions. Other filter functions ignore the additional arguments.

It is also possible to write custom output filter. An output filter should have code, language and id as its first three arguments. It should either return a character vector containing the markdown that should be included in the resulting markdown file or an object that can be directly included in the pandoc parse tree. If the function does not return a character vector it is assumed the latter is returned. simplermarkdown defines a small number of valid object constructors: raw_block and markdown_block.

The custom function should be available when running the markdown document through pandoc. The easiest way is to source or define the function in the markdown document before using it.

The filter function output_shell can be used to process chunks of code from other languages than R. These chunks of code are written to a temporary file which is then ran using the cmd using a call to system2. The output of that is captured and when results = TRUE included in the output. The output lines are prepended by comment_char. When echo = TRUE the code is also included before the output.

Value

The functions either return a character vector with markdown, or return a list with the correct structure to include in the pandoc parse tree.

raw_block *Return a raw chunk of text that can be included in the pandoc parse tree*

Description

Return a raw chunk of text that can be included in the pandoc parse tree

Usage

```
raw_block(content, language = "markdown")
```

Arguments

content	a character vector containing the content to include in the final document.
language	language of the content

Details

A raw block is included as is into the final markdown document. This can be used for example to include raw chunks of markdown.

Value

Returns a `list` with the correct structure for a `RawBlock` in the pandoc parse tree.

run_and_capture *Run code and capture the output*

Description

Run code and capture the output

Usage

```
run_and_capture(
  code,
  echo = TRUE,
  results = TRUE,
  output = results,
  capture_warnings = FALSE,
  capture_messages = results,
  muffle_warnings = FALSE,
  muffle_messages = TRUE
)
```

Arguments

code	character vector or expression with the code to run
echo	the code in code is repeated in the output.
results	include the results of running the code in the output.
output	include output that is explicitly written to the output, for example using print statements.
capture_warnings	include warnings in the output.
capture_messages	include messages in the output.
muffle_warnings	do not show warnings in the console.
muffle_messages	do not show messages in the console.

Value

Returns a list. Each item of the list contains a list with elements input and output. input contains the command/code and output the corresponding output. These are empty vectors when there is no output or when input and output are suppressed using one of the echo/results/output statements.

Index