Package: parsermd (via r-universe)

September 13, 2024

Title Formal Parser and Related Tools for R Markdown Documents

Version 0.2.0

Description An implementation of a formal grammar and parser for R Markdown documents using the Boost Spirit X3 library. It also includes a collection of high level functions for working with the resulting abstract syntax tree.

License MIT + file LICENSE

Encoding UTF-8

Roxygen list(markdown = TRUE)

Depends R (>= 3.5.0)

Imports purrr, Rcpp, cli (>= 2.5.0), checkmate, readr, tidyr, dplyr, tibble, yaml, withr, rmarkdown, pillar, rlang, magrittr, tidyselect (>= 1.2.0), lifecycle, fs, quarto

RoxygenNote 7.3.2

SystemRequirements C++17

LinkingTo Rcpp, BH

Suggests testthat (>= 3.0.0), knitr

Config/testthat/edition 3

VignetteBuilder knitr

URL https://rundel.github.io/parsermd/, https://github.com/rundel/parsermd

BugReports https://github.com/rundel/parsermd/issues

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

RemoteUrl https://github.com/rundel/parsermd

RemoteRef HEAD

RemoteSha d00049e029641b51d2380410d9b5c1b204455ddf

Contents

`as_ast` *Convert an object into an* `rmd_ast`.

Description

Currently only supports conversion of `rmd_tibble` objects back to `rmd_ast`.

Usage

```
as_ast(x, ...)
```

Arguments

`x`	Object to convert
`...`	Unused, for extensibility.

Value

Returns an `rmd_ast` object.

Examples

```
parse_rmd(system.file("examples/hw01.Rmd", package="parsermd")) %>%
  as_tibble() %>%
  as_ast()
```

as_document *Convert an* rmd_ast, rmd_tibble, *or any ast node into text.*

Description

Convert an rmd_ast, rmd_tibble, or any ast node into text.

Usage

```
as_document(x, padding = "", collapse = NULL, ...)
```

Arguments

x	rmd_ast, rmd_tibble, or parsermd node object.
padding	Padding to add between nodes when assembling the text.
collapse	If not NULL, use value to collapse lines.
...	Passed to to_ast() when converting rmd_collection or qmd_collection.

Value

Returns a character vector.

chunk_options *Get and set code chunk options*

Description

Helper functions for obtaining or changing chunk options within an rmd object.

Usage

```
rmd_set_options(x, ...)

rmd_get_options(x, ..., defaults = list())
```

Arguments

x	An rmd_ast, rmd_tibble, or any rmd ast node object.
...	Either a collection of named values for the setter or a character values of the option names for the getter.
defaults	A named list of default values for the options.

Value

`rmd_set_options` returns the modified version of the original object.

`rmd_get_options` returns a list of the requested options (or all options if none are specified). Non-chunk nodes return `NULL`.

Examples

```
rmd = parse_rmd(system.file("examples/minimal.Rmd", package = "parsermd"))

str(rmd_get_options(rmd))
str(rmd_get_options(rmd), "include")

rmd_set_options(rmd, include = TRUE)
```

---

parse_collection *Parse a collection of R Markdown or Quarto document*

Description

[Experimental]

Recursively searches a directory for R Markdown or Quarto documents and parses them into a collection of `rmd_ast` objects

Usage

```
parse_qmd_collection(
  dir = "./",
  pattern = "*.qmd",
  all = FALSE,
  recurse = TRUE,
  regex = FALSE
)

parse_rmd_collection(
  dir = "./",
  pattern = "*.Rmd",
  all = FALSE,
  recurse = TRUE,
  regex = FALSE
)
```

Arguments

`dir`	Directory to search
`pattern`	Pattern to match files, defaults to glob syntax

all	Search includes hidden files
recurse	Search recusively within dir
regex	Treat pattern as a regular expression syntax for pattern

Value

Returns a tibble object with columns for document name, path, and ast.

Examples

```
parse_rmd_collection(system.file("examples/", package="parsermd"))
```

---

parse_rmd *Parse an R Markdown or Quarto document*

Description

Documents are parsed into an rmd_ast object.

Usage

```
parse_rmd(rmd, allow_incomplete = FALSE)

parse_qmd(qmd, allow_incomplete = FALSE)
```

Arguments

rmd	Either the path to an Rmd file or a character vector containing the contents of a R Markdown document.
allow_incomplete	Allow incomplete parsing of the document.
qmd	Either the path to an qmd file or a character vector containing the contents of a Quarto document.

Value

Returns a rmd_ast object.

Examples

```
parse_rmd(system.file("examples/hw01.Rmd", package="parsermd"))
```

render *Render* parsermd *objects*

Description

Object contents are converted to a character vector and written to a temporary directory before rendering via `quarto::quarto_render()` or `rmarkdown::render()`.

Note that this function has the potential to overwrite existing output files (e.g. `.html`, `.pdf`, etc).

Usage

```
render(x, name = NULL, ..., engine = c("quarto", "rmarkdown"))
```

Arguments

x	Object to render, e.g. a `rmd_ast`, `rmd_tibble`, `character` vector, etc.
name	Name of the output file, if not given it will be inferred from the name of `x`.
...	Any additional arguments to be passed to `quarto::quarto_render()` or `rmarkdown::render()`
engine	The rendering engine to use, either "quarto" or "rmarkdown".

Value

Returns the results of the render function.

rmd_ast_append *Append or preprend nodes to an ast*

Description

Functions for adding nodes to the beginning or end of an ast.

Usage

```
rmd_ast_append(x, ...)

rmd_ast_prepend(x, ...)
```

Arguments

x	An object containing an `rmd_ast` of some kind, e.g. `rmd_ast`, `rmd_tibble`, or `rmd_collection`.
...	A collections of ast nodes to append or prepend.

Value

An object of the same class as `x`

rmd_check_template *Check an Rmd against a template*

Description

This function compares the provided Rmd against a template and reports on discrepancies (e.g. missing or unmodified components).

Usage

```
rmd_check_template(rmd, template, ...)
```

Arguments

rmd	The rmd to be check, can be an rmd_ast, rmd_tibble, or text that can be handled by parse_rmd.
template	rmd_template object from rmd_template().
...	Unused, for extensibility.

Value

Invisibly returns TRUE if the rmd matches the template, FALSE otherwise.

Examples

```
tmpl = parse_rmd(system.file("examples/hw01.Rmd", package = "parsermd")) %>%
  rmd_select(by_section(c("Exercise *", "Solution"))) %>%
  rmd_template(keep_content = TRUE)

rmd_check_template(
  system.file("examples/hw01-student.Rmd", package = "parsermd"),
  tmpl
)
```

rmd_create *AST node creation*

Description

Functions for creating ast nodes,

- rmd_ast() - Create an ast container of nodes
- rmd_yaml() - Create a yaml node
- rmd_heading() - Create a heading node

- `rmd_code_blokc()` - Create a markdown code block node
- `rmd_chunk()` - Create a chunk node
- `rmd_raw_chunk()` - Create a raw chunk node
- `rmd_fenced_div_open()` - Create a fenced div open node
- `rmd_fenced_div_close()` - Create a fenced div close node
- `rmd_markdown()` - Create a markdown container node of `rmd_markdown_lines`
- `rmd_markdown_line()` - Create a markdown line node
- `rmd_inline_code()` - Create an inline code node
- `rmd_shortcode()` - Create a shortcode node

Usage

```
rmd_ast(...)

rmd_yaml(...)

rmd_heading(name, level)

rmd_code_block(attr = "", code = character(), indent = "", n_ticks = 3L)

rmd_chunk(
  name = NULL,
  engine = "r",
  options = list(),
  yaml_options = list(),
  code = character(),
  indent = "",
  n_ticks = 3L
)

rmd_raw_chunk(format, code = character(), indent = "", n_ticks = 3L)

rmd_fenced_div_open(attr = character())

rmd_fenced_div_close()

rmd_markdown(...)

rmd_markdown_line(...)

rmd_inline_code(engine = "", code = "")

rmd_shortcode(func, args = character())
```

Arguments

`...`	Elements within the node.

name	Character. Heading or chunk name.
level	Integer. Heading level (1-6).
attr	Character. Attributes for code block or fenced div.
code	Character. Code lines for code block or chunk.
indent	Character. Indentation for code block or chunk.
n_ticks	Integer. Number of backticks for code block or chunk.
engine	Character. Language engine for chunk or inline code
options	List. Chunk options.
yaml_options	List. Chunk yaml options.
format	Character. Format for raw chunk.
func	Character. Shortcode function name.
args	Character. Shortcode arguments.

Value

An object with class matching the function name, e.g. `rmd_ast()` returns an `rmd_ast` object.

`rmd_node` *rmd node utility functions*

Description

Functions for extracting information for Rmd nodes.

Usage

```
rmd_node_label(x, ...)

rmd_node_type(x, ...)

rmd_node_length(x, ...)

rmd_node_content(x, ...)

rmd_node_attr(x, attr, ...)

rmd_node_engine(x, ...)

rmd_node_options(x, ...)

rmd_node_code(x, ...)
```

Arguments

x	An rmd object, e.g. rmd_ast or rmd_tibble.
...	Unused, for extensibility.
attr	Attribute name to extract.

Value

- rmd_node_label() - returns a character vector of node labels, nodes without labels return NA.
- rmd_node_type() - returns a character vector of node types.
- rmd_node_length() - returns an integer vector of node lengths (i.e. lines of code, lines of text, etc.), nodes without a length return NA.
- rmd_node_content() - returns a character vector of node textual content, nodes without content return NA.
- rmd_node_attr() - returns a list of node attribute values.
- rmd_node_engine() - returns a character vector of chunk engines, NA for all other node types.
- rmd_node_options() - returns a list of chunk node options (named list), MULL for all other node types.
- rmd_node_code() - returns a list of chunk node code (character vector), NULL for all other node types.

Examples

```
rmd = parse_rmd(system.file("examples/hw01.Rmd", package="parsermd"))

rmd_node_label(rmd)
rmd_node_type(rmd)
rmd_node_content(rmd)
rmd_node_attr(rmd, "level")
rmd_node_engine(rmd)
rmd_node_options(rmd)
rmd_node_code(rmd)
```

---

rmd_node_sections *Find the sections for each rmd object node*

Description

Uses the section headings of an rmd object to identify the hierarchical structure of the document.

Usage

```
rmd_node_sections(x, levels = 1:6, drop_na = FALSE)
```

Arguments

x	An rmd object, e.g. rmd_ast or rmd_tibble.
levels	Limit which section heading levels to return.
drop_na	Should NA sections be dropped.

Value

A list of section names for each node.

---

rmd_select *Select nodes of an Rmd ast*

Description

This function is implemented using tidyselect::eval_select() which enables a variety of useful syntax for selecting nodes from the ast.

Additionally, a number of additional parsermd specific selection helpers are available: by_section(), has_type(), has_label(), and has_option().

Usage

```
rmd_select(x, ...)
```

Arguments

x	Rmd object, e.g. rmd_ast or rmd_tibble.
...	One or more unquoted expressions separated by commas. Chunk labels can be used as if they were positions in the data frame, so expressions like x:y can be used to select a range of nodes.

Value

Returns a subset Rmd object (either rmd_ast or rmd_tibble depending on input).

Examples

```
rmd = parse_rmd(system.file("examples/hw01.Rmd", package = "parsermd"))

rmd_select(rmd, "plot-dino", "cor-dino")
rmd_select(rmd, "plot-dino":"cor-dino")
rmd_select(rmd, `plot-dino`:`cor-dino`)

rmd_select(rmd, has_type("rmd_chunk"))

rmd_select(rmd, by_section(c("Exercise *", "Solution")))
```

`rmd_select_helpers` *Rmd selection helper functions*

Description

These functions are used in conjunction with `rmd_select()` to select nodes from an Rmd ast.

- `by_section()` - uses section selectors to select nodes.
- `has_type()` - selects all nodes that have the given type(s).
- `has_label()` - selects nodes with labels matching the given glob.
- `has_option()` - selects nodes that have the given option(s) set.

Usage

```
has_type(types)

by_section(sec_ref, keep_parents = TRUE)

has_label(label)

has_code(code)

has_option(...)
```

Arguments

`types`	Vector of character type names, e.g. `rmd_chunk`, `rmd_heading`, etc.
`sec_ref`	character vector, a section reference selector. See details below for further details on how these are constructed.
`keep_parents`	Logical, retain the parent headings of selected sections. Default: `TRUE`
`label`	character vector, glob patterns for matching chunk labels.
`code`	character vector, regex patterns for matching chunk code line(s)
`...`	Either option names represented by a scalar string or a named argument with the form `opt = value` where `opt` is the option name and `value` is the value to be checked. For example `eval = TRUE` would check for the option `eval` being set to `TRUE`.

Details

Section reference selectors:

Section reference selectors are a simplified version of CSS selectors that are designed to enable the selection nodes in a way that respects the implied hierarchy of a document's section headings.

They consist of a character vector of heading names where each subsequent value is assumed to be nested within the preceding value. For example, the section selector `c("Sec 1", "Sec 2")` would

select all nodes that are contained within a section named `Sec 2` that is in turn contained within a section named `Sec 1` (or a section contained within a section named `Sec 1`, and so on).

The individual section names can be specified using wildcards (aka globbing patterns), which may match one or more sections within the document, e.g. `c("Sec 1", "Sec *")`. See `utils::glob2rx()` or wikipedia for more details on the syntax for these patterns.

Value

All helper functions return an integer vector of selected indexes.

Examples

```
rmd = parse_rmd(system.file("examples/hw01.Rmd", package="parsermd"))

rmd_select(rmd, has_type("rmd_chunk"))

rmd_select(rmd, has_label("*dino"))

rmd_select(rmd, has_option("message"))
rmd_select(rmd, has_option(message = FALSE))
rmd_select(rmd, has_option(message = TRUE))
```

---

rmd_source *Source the code chunks of an Rmd document*

Description

This is the equivalent of the `source()` function for Rmd files or their resulting asts.

Usage

```
rmd_source(x, local = FALSE, ..., label_comment = TRUE, use_eval = TRUE)
```

Arguments

`x`	An Rmd document (e.g. `rmd_ast`, `rmd_tibble`, Rmd file path, etc.)
`local`	`TRUE`, `FALSE` or an environment, determining where the parsed expressions are evaluated. `FALSE` (the default) corresponds to the user's workspace (the global environment) and `TRUE` to the environment from which source is called.
`...`	Additional arguments passed to `source`.
`label_comment`	Attach chunk labels as comment before each code block.
`use_eval`	Use the `eval` chunk option to determine if code is included.

Value

Returns the result of `source()` for any R code chunks.

Examples

```
rmd_source(system.file("examples/minimal.Rmd", package = "parsermd"), echo=TRUE)
```

`rmd_subset` *Subset the nodes of an rmd object*

Description

[Deprecated] Subset an rmd object based on sections, node types, or names.

Usage

```
rmd_subset(
  x,
  sec_refs = NULL,
  type_refs = NULL,
  name_refs = NULL,
  exclude = FALSE,
  keep_yaml = TRUE,
  keep_setup = FALSE,
  ...
)
```

Arguments

`x`	rmd object, e.g. `rmd_ast` or `rmd_tibble`.
`sec_refs`	Section references, TODO - add details.
`type_refs`	Node type references, TODO - add details.
`name_refs`	Name references, TODO - add details.
`exclude`	Should the matching nodes be excluded.
`keep_yaml`	Should the document yaml be kept.
`keep_setup`	Should the document setup chunk be kept.
`...`	Unused, for extensibility.

Value

Returns a subset Rmd object (either `rmd_ast` or `rmd_tibble` depending on input).

`rmd_subset_util` *rmd_subset utility functions*

Description

[Deprecated]

Tools for selecting or checking a single node using `rmd_subset()` selection.

Usage

```
rmd_get_node(x, sec_refs = NULL, type_refs = NULL, name_refs = NULL, ...)

rmd_get_chunk(x, sec_refs = NULL, name_refs = NULL)

rmd_get_markdown(x, sec_refs = NULL)

rmd_has_node(x, sec_refs = NULL, type_refs = NULL, name_refs = NULL, ...)

rmd_has_chunk(x, sec_refs = NULL, name_refs = NULL, ...)

rmd_has_markdown(x, sec_refs = NULL, ...)
```

Arguments

`x`	rmd object, e.g. `rmd_ast` or `rmd_tibble`.
`sec_refs`	Section references, TODO - add details.
`type_refs`	Node type references, TODO - add details.
`name_refs`	Name references, TODO - add details.
`...`	Unused, for extensibility.

Value

- `rmd_get_*()` functions returns a single Rmd node object (e.g. `rmd_heading`, `rmd_chunk`, `rmd_markdown`, etc.)
- `rmd_has_*()` functions return `TRUE` if a matching node exists, `FALSE` otherwise.

rmd_template *Create a template from an* rmd *object.*

Description

Templates are objects which are meant to capture the structure of an R Markdown document and facilitate the comparison between the template and new Rmd documents, usually to ensure the structure and/or content matches sufficiently.

Usage

```
rmd_template(
  rmd,
  keep_content = FALSE,
  keep_labels = TRUE,
  keep_headings = FALSE,
  keep_yaml = FALSE,
  ...
)
```

Arguments

rmd	R Markdown document in the form of an rmd_ast or rmd_tibble.
keep_content	Should the template keep the document's content (markdown text and chunk code).
keep_labels	Should the template keep the document's code chunk labels.
keep_headings	Should the template keep the document's headings.
keep_yaml	Should the template keep the document's yaml.
...	Unused, for extensibility.

Value

Returns an rmd_template object, which is a derived tibble containing relevant structural details of the document.

Examples

```
rmd = parse_rmd(system.file("examples/hw01.Rmd", package="parsermd"))

rmd_select(rmd, by_section(c("Exercise *", "Solution"))) %>%
  rmd_template()
```

Index