

Package ‘parsermd’

July 23, 2025

Title Formal Parser and Related Tools for R Markdown Documents

Version 0.1.3

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

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

RoxygenNote 7.3.0

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>

NeedsCompilation yes

Author Colin Rundel [aut, cre]

Maintainer Colin Rundel <rundel@gmail.com>

Repository CRAN

Date/Publication 2024-01-16 16:20:02 UTC

Contents

as_ast	2
as_document	3
chunk_options	3
parse_rmd	4
render	5
rmd_check_template	5
rmd_node	6
rmd_node_sections	7
rmd_select	8
rmd_select_helpers	9
rmd_source	10
rmd_subset	11
rmd_subset_util	12
rmd_template	13
Index	14

as_ast	<i>Convert an object into an rmd_ast.</i>
--------	---

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("hw01.Rmd", package="parsermd")) %>%  
  as_tibble() %>%  
  as_ast()
```

as_document	<i>Convert an rmd_ast, rmd_tibble, or any ast node into text.</i>
-------------	---

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.
...	Unused, for extensibility.

Value

Returns a character vector.

chunk_options	<i>Get and set code chunk options</i>
---------------	---------------------------------------

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("minimal.Rmd", package = "parsermd"))

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

rmd_set_options(rmd, include = TRUE)
```

parse_rmd

Parse an R Markdown document

Description

Documents are parse into an `rmd_ast` object.

Usage

```
parse_rmd(rmd, allow_incomplete = FALSE, parse_yaml = TRUE)
```

Arguments

<code>rmd</code>	Either the path to an Rmd file or a character vector containing the contents of a R Markdown document.
<code>allow_incomplete</code>	Allow incomplete parsing of the document.
<code>parse_yaml</code>	Use the yaml package to parse the document's yaml.

Value

Returns a `rmd_ast` object.

Examples

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

render	Render parsermd objects using rmarkdown::render()
--------	---

Description

Object contents are converted to a character vector and written to a temporary directory before rendering.

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

Usage

```
render(x, name = NULL, ...)
```

Arguments

x	Object to render, e.g. a <code>rmd_ast</code> , <code>rmd_tibble</code> , 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 rmarkdown::render()

Value

Returns the results of [rmarkdown::render\(\)](#).

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 <code>rmd_ast</code> , <code>rmd_tibble</code> , or text that can be handled by <code>parse_rmd</code> .
template	<code>rmd_template</code> 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("hw01.Rmd", package = "parsermd")) %>%
  rmd_select(by_section(c("Exercise *", "Solution")) %>%
    rmd_template(keep_content = TRUE)

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

```

rmd_node	<i>rmd node utility functions</i>
----------	-----------------------------------

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

<code>x</code>	An rmd object, e.g. <code>rmd_ast</code> or <code>rmd_tibble</code> .
<code>...</code>	Unused, for extensibility.
<code>attr</code>	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), NULL 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("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)
```

<code>rmd_node_sections</code>	<i>Find the sections for each rmd object node</i>
--------------------------------	---

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

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

Value

A list of section names for each node.

rmd_select	<i>Select nodes of an Rmd ast</i>
------------	-----------------------------------

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

<code>x</code>	Rmd object, e.g. <code>rmd_ast</code> or <code>rmd_tibble</code> .
<code>...</code>	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 <code>x:y</code> 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("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")))
```


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_option(...)
```

Arguments

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

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("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("minimal.Rmd", package = "parsermd"), echo=TRUE)
```

rmd_subset	<i>Subset the nodes of an rmd object</i>
------------	--

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	<i>rmd_subset utility functions</i>
-----------------	-------------------------------------

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

<code>x</code>	rmd object, e.g. <code>rmd_ast</code> or <code>rmd_tibble</code> .
<code>sec_refs</code>	Section references, TODO - add details.
<code>type_refs</code>	Node type references, TODO - add details.
<code>name_refs</code>	Name references, TODO - add details.
<code>...</code>	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	<i>Create a template from an rmd object.</i>
--------------	--

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 <code>rmd_ast</code> or <code>rmd_tibble</code> .
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("hw01.Rmd", package="parsermd"))

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

Index

as_ast, [2](#)
as_document, [3](#)

by_section(rmd_select_helpers), [9](#)
by_section(), [8](#)

chunk_options, [3](#)

has_label(rmd_select_helpers), [9](#)
has_label(), [8](#)
has_option(rmd_select_helpers), [9](#)
has_option(), [8](#)
has_type(rmd_select_helpers), [9](#)
has_type(), [8](#)

parse_rmd, [4](#)

render, [5](#)
rmarkdown::render(), [5](#)
rmd_check_template, [5](#)
rmd_get_chunk(rmd_subset_util), [12](#)
rmd_get_markdown(rmd_subset_util), [12](#)
rmd_get_node(rmd_subset_util), [12](#)
rmd_get_options(chunk_options), [3](#)
rmd_has_chunk(rmd_subset_util), [12](#)
rmd_has_markdown(rmd_subset_util), [12](#)
rmd_has_node(rmd_subset_util), [12](#)
rmd_node, [6](#)
rmd_node_attr(rmd_node), [6](#)
rmd_node_code(rmd_node), [6](#)
rmd_node_content(rmd_node), [6](#)
rmd_node_engine(rmd_node), [6](#)
rmd_node_label(rmd_node), [6](#)
rmd_node_length(rmd_node), [6](#)
rmd_node_options(rmd_node), [6](#)
rmd_node_sections, [7](#)
rmd_node_type(rmd_node), [6](#)
rmd_select, [8](#)
rmd_select(), [9](#)
rmd_select_helpers, [9](#)
rmd_set_options(chunk_options), [3](#)

rmd_source, [10](#)
rmd_subset, [11](#)
rmd_subset(), [12](#)
rmd_subset_util, [12](#)
rmd_template, [13](#)
rmd_template(), [5](#)

source, [10](#)
source(), [10](#)

tidyselect::eval_select(), [8](#)

utils::glob2rx(), [9](#)

yaml, [4](#)