

# Package ‘yamlthis’

August 5, 2022

**Title** Write 'YAML' for 'R Markdown', 'bookdown', 'blogdown', and More

**Version** 0.1.7

**Description** Write 'YAML' front matter for R Markdown and related documents. Work with 'YAML' objects more naturally and write the resulting 'YAML' to your clipboard or to 'YAML' files related to your project.

**License** MIT + file LICENSE

**URL** <https://yamlthis.r-lib.org>, <https://github.com/r-lib/yamlthis>

**BugReports** <https://github.com/r-lib/yamlthis/issues>

**Depends** R (>= 3.2)

**Imports** crayon, fs, glue, magrittr, purrr (>= 0.3.2), rlang (>= 0.4.10), rmarkdown (>= 2.10), rstudioapi, stringr, usethis (>= 1.5.0), whoami, withr, yaml

**Suggests** blogdown, bookdown, covr, knitr, miniUI, pkgdown, prettydoc, roxygen2 (>= 7.0.0), shiny, shinyBS, spelling, testthat (>= 3.0.0), xaringan

**VignetteBuilder** knitr

**Config/testthat/edition** 3

**Encoding** UTF-8

**Language** en-US

**RoxygenNote** 7.2.0

**NeedsCompilation** no

**Author** Malcolm Barrett [aut, cre] (<<https://orcid.org/0000-0003-0299-5825>>),  
Richard Iannone [aut] (<<https://orcid.org/0000-0003-3925-190X>>),  
RStudio [cph, fnd]

**Maintainer** Malcolm Barrett <malcolmbarrett@gmail.com>

**Repository** CRAN

**Date/Publication** 2022-08-05 15:00:02 UTC

**R topics documented:**

asis_yaml_output . . . . .	3
as_yaml . . . . .	3
bib2yaml . . . . .	4
blogdown_template . . . . .	5
code_chunk . . . . .	5
draw_yaml_tree . . . . .	6
gitbook_config . . . . .	7
has_field . . . . .	9
includes2 . . . . .	10
is_yaml . . . . .	10
last_yaml . . . . .	11
pagedown_business_card_template . . . . .	11
pandoc_template_types . . . . .	14
pkgdown_template . . . . .	15
read_json . . . . .	16
use_yaml . . . . .	17
use_yaml_defaults . . . . .	18
use_yaml_file . . . . .	19
yaml . . . . .	21
yaml_author . . . . .	23
yaml_blank . . . . .	25
yaml_blogdown_opts . . . . .	26
yaml_bookdown_opts . . . . .	28
yaml_citations . . . . .	31
yaml_clean . . . . .	32
yaml_code . . . . .	33
yaml_distill_opts . . . . .	34
yaml_handlers . . . . .	38
yaml_latex_opts . . . . .	38
yaml_load . . . . .	41
yaml_output . . . . .	42
yaml_pagedown_opts . . . . .	43
yaml_params . . . . .	44
yaml_pkgdown . . . . .	49
yaml_reference . . . . .	53
yaml_replace . . . . .	55
yaml_resource_files . . . . .	56
yaml_rsconnect_email . . . . .	57
yaml_rtticles_opts . . . . .	59
yaml_runtime . . . . .	61
yaml_site_opts . . . . .	62
yaml_toc . . . . .	64
yaml_verbatim . . . . .	65
yaml_vignette . . . . .	66

---

asis_yaml_output	<i>Export yml object as a YAML knitr code chunk</i>
------------------	---

---

### Description

asis\_yaml\_output() exports a yml object as a YAML knitr code chunk instead of as an R object. Doing so adds code highlighting for YAML syntax.

### Usage

```
asis_yaml_output(.yaml, fences = TRUE)
```

### Arguments

.yaml	a yml object created by yml(), as_yaml(), or returned by a yml_*() function
fences	Logical. Write fences ("—") before and after YAML?

### See Also

Other yml: [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

---

as_yaml	<i>Convert to yml object</i>
---------	------------------------------

---

### Description

as\_yaml is a wrapper for [yaml::yaml.load\(\)](#) that stores YAML as a yml object, which prints cleanly to the console and is easy to work with using ymlthis functions.

### Usage

```
as_yaml(x)
```

### Arguments

x	An object, either a character vector of length 1 or list, to convert to yml.
---	--

### Value

a yml object

**Examples**

```
x <- as_yaml("
  author: Hadley Wickham
  date: '2014-09-12'
  title: Tidy Data
  keywords:
  - data cleaning
  - data tidying
  - relational databases
  - R")

x

x %>%
  yaml_subtitle("Hadley's Tidy Data Paper")
```

---

 bib2yaml

 Convert bib files to YAML
 

---

**Description**

bib2yaml() uses pandoc to convert a .bib file to YAML. It also accepts an optional yml object to prepend to the the YAML from the .bib file. If you want to cite several R packages, see [knitr::write\\_bib\(\)](#) to write a bibliography file and convert it with bib2yaml().

**Usage**

```
bib2yaml(.yaml = NULL, path)
```

**Arguments**

.yaml	a yml object created by yml(), as_yaml(), or returned by a yml_*() function
path	a path to the .bib file

**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

Other citations: [yaml\\_citations\(\)](#), [yaml\\_reference\(\)](#)

---

blogdown\_template      *Create YAML based on blogdown theme archetypes*

---

### Description

blogdown\_template() creates YAML based on your blogdown theme archetypes. blogdown is based on Hugo, which supports many custom themes. Each theme uses YAML in a different way. However, many come with archetypes that define the YAML or TOML. To find out which types your theme has, use blogdown\_archetypes() to see what's available. Use blogdown\_template() to specify the archetype and it will convert the template to YAML that you can use in your post.

### Usage

```
blogdown_template(type, path = ".", theme = NULL)
```

```
blogdown_archetypes(path = ".", theme = NULL)
```

### Arguments

type	an archetype
path	the path to your blogdown site
theme	the theme to check for archetypes. By default, blogdown_template() will attempt to read your theme from your config file.

### Value

a yml object

---

code\_chunk      *Write code chunks programmatically*

---

### Description

code\_chunk() assembles a knitr code chunk as a character vector. setup\_chunk() is a wrapper around code\_chunk() to create setup chunks. By default it uses include = FALSE and inserts knitr::opts\_chunk\$set(echo = TRUE) into the chunk body. These are helper functions to write R Markdown bodies for [use\\_rmarkdown\(\)](#).

### Usage

```
code_chunk(chunk_code, chunk_name = NULL, chunk_args = NULL)
```

```
setup_chunk(chunk_code = NULL, chunk_args = list(include = FALSE))
```

**Arguments**

chunk_code	An expression. Surround with {} to capture multiple lines.
chunk_name	The name of the chunk
chunk_args	A list of chunk options

**Value**

a character vector

**Examples**

```
setup_chunk()

code_chunk({
  yml() %>%
    yml_output(pdf_document())
}, chunk_name = "yml_example")
```

---

draw_yaml_tree	<i>Draw an tree of YAML hierarchy</i>
----------------	---------------------------------------

---

**Description**

draw\_yaml\_tree() draws an ASCII tree of the hierarchy of a given yml object to the console.

**Usage**

```
draw_yaml_tree(.yaml = last_yaml(), indent = "")
```

**Arguments**

.yaml	a yml object created by yml(), as_yaml(), or returned by a yml_*() function
indent	a character vector used to indent the tree

**Value**

invisibly, .yaml

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

## Examples

```
# draw the most recently used `yaml`
draw_yaml_tree()

yaml() %>%
  yaml_output(
    pdf_document(keep_tex = TRUE),
    html_document()
  ) %>%
  draw_yaml_tree()
```

---

gitbook\_config

*Configure bookdown::gitbook() output*

---

## Description

`gitbook_config()` is a helper function to specify the config argument in `bookdown::gitbook()`, as described in the [bookdown book](#).

## Usage

```
gitbook_config(
  toc_collapse = yaml_blank(),
  toc_scroll_highlight = yaml_blank(),
  toc_before = yaml_blank(),
  toc_after = yaml_blank(),
  toolbar_position = yaml_blank(),
  edit = yaml_blank(),
  download = yaml_blank(),
  search = yaml_blank(),
  fontsettings_theme = yaml_blank(),
  fontsettings_family = yaml_blank(),
  fontsettings_size = yaml_blank(),
  sharing_facebook = yaml_blank(),
  sharing_twitter = yaml_blank(),
  sharing_google = yaml_blank(),
  sharing_linkedin = yaml_blank(),
  sharing_weibo = yaml_blank(),
  sharing_instapaper = yaml_blank(),
  sharing_vk = yaml_blank(),
  sharing_all = yaml_blank(),
  ...
)
```

**Arguments**

toc_collapse	Collapse some items initially when a page is loaded via the collapse option. Its possible values are "subsection" (the default), "section", "none", or NULL.
toc_scroll_highlight	Logical. Enable highlighting of TOC items as you scroll the book body? The default is TRUE.
toc_before, toc_after	a character vector of HTML to add more items before and after the TOC using the HTML tag <code>&lt;li&gt;</code> . These items will be separated from the TOC using a horizontal divider.
toolbar_position	The toolbar position: "fixed" or "static." The default ("fixed") is that the toolbar will be fixed at the top of the page, whereas when set to "static" the toolbar will not scroll with the page.
edit	If not empty, an edit button will be added to the toolbar.
download	This option takes either a character vector or a list of character vectors with the length of each vector being 2. When it is a character vector, it should be either a vector of filenames or filename extensions. When you only provide the filename extensions, the filename is derived from the book filename of the configuration file <code>_bookdown.yml</code>
search	Include a search bar?
fontsettings_theme	The theme. "White" (the default), "Sepia", or "Night".
fontsettings_family	The font family. "sans" (the default) or "serif".
fontsettings_size	The font size. Default is 2.
sharing_facebook	Logical. Include Facebook share link? Default is TRUE.
sharing_twitter	Logical. Include Twitter share link? Default is TRUE.
sharing_google	Logical. Include Google share link? Default is FALSE.
sharing_linkedin	Logical. Include LinkedIn share link? Default is FALSE.
sharing_weibo	Logical. Include Weibo share link? Default is FALSE.
sharing_instapaper	Logical. Include Instapaper share link? Default is FALSE.
sharing_vk	Logical. Include VK share link? Default is FALSE.
sharing_all	Logical. Include all share links? Default is FALSE.
...	additional named R objects, such as characters or lists, to transform into YAML

**Value**

a list to use in the config argument of `bookdown::gitbook()`



**See Also**

Other bookdown: [yml\\_bookdown\\_opts\(\)](#)

---

`has_field`*Check if field exists in YAML*

---

**Description**

`has_field()` retrieves the names of all fields (including nested fields) and checks if `field` is among them.

**Usage**

```
has_field(.yml, field)
```

**Arguments**

<code>.yml</code>	a <code>yml</code> object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> ( <code>)</code> function
<code>field</code>	A character vector, the name of the field(s) to check for

**Value**

logical

**See Also**

Other `yml`: [asis\\_yaml\\_output\(\)](#), [bib2yml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yml\\_author\(\)](#), [yml\\_blogdown\\_opts\(\)](#), [yml\\_bookdown\\_opts\(\)](#), [yml\\_citations\(\)](#), [yml\\_clean\(\)](#), [yml\\_distill\\_opts\(\)](#), [yml\\_latex\\_opts\(\)](#), [yml\\_output\(\)](#), [yml\\_pagedown\\_opts\(\)](#), [yml\\_params\(\)](#), [yml\\_pkgdown\(\)](#), [yml\\_reference\(\)](#), [yml\\_replace\(\)](#), [yml\\_resource\\_files\(\)](#), [yml\\_rsconnect\\_email\(\)](#), [yml\\_rarticles\\_opts\(\)](#), [yml\\_runtime\(\)](#), [yml\\_site\\_opts\(\)](#), [yml\\_toc\(\)](#), [yml\\_vignette\(\)](#)

**Examples**

```
has_field(yml(), "author")
has_field(yml(), "toc")
```

---

includes2	<i>Include content within output</i>
-----------	--------------------------------------

---

**Description**

includes2() is a version of the includes() helper function from rmarkdown that uses yml\_blank() instead of NULL as the argument defaults, as ymlthis treats NULLs as literal YAML syntax ("null").

**Usage**

```
includes2(  
  in_header = yml_blank(),  
  before_body = yml_blank(),  
  after_body = yml_blank()  
)
```

**Arguments**

in_header	One or more files with content to be included in the header of the document.
before_body	One or more files with content to be included before the document body.
after_body	One or more files with content to be included after the document body.

**Value**

a list

**Examples**

```
yml() %>%  
  yml_output(  
    pdf_document(includes = includes2(after_body = "footer.tex"))  
  )
```

---

is_yaml	<i>Is object a yaml object?</i>
---------	---------------------------------

---

**Description**

Is object a yaml object?

**Usage**

```
is_yaml(x)
```

**Arguments**

x                    An object to test

**Value**

A logical vector

---

last_yaml	<i>Return the most recently printed YAML</i>
-----------	--

---

**Description**

yamlthis stores the most recently printed yaml object; you can use last\_yaml() to retrieve it to modify, pass to use\_\*() functions, and so on.

**Usage**

```
last_yaml()
```

**Examples**

```
yaml() %>%
  yaml_author("Yihui Xie")

last_yaml()
```

---

pagedown_business_card_template	<i>Generate a full YAML template for your pagedown business card</i>
---------------------------------	--

---

**Description**

pagedown has a unique output type to make business cards: pagedown::business\_card(). pagedown\_business\_card\_template creates a YAML template to use for this output. What's unique about this output type is that almost all of the contents are supplied through YAML. An R Markdown file that only contains YAML related to the business card is enough to produce the output, although you can also customize the output in the body of the document (see the [pagedown vignette](#)). A good workflow to write a business card is to use pagedown\_business\_card\_template() to specify the YAML and pass it to [use\\_rmarkdown\(\)](#), which you can then to knit into business cards.

**Usage**

```

pagedown_business_card_template(
  name = yml_blank(),
  person = yml_blank(),
  title = yml_blank(),
  phone = yml_blank(),
  email = yml_blank(),
  url = yml_blank(),
  address = yml_blank(),
  logo = yml_blank(),
  .repeat = yml_blank(),
  paperwidth = yml_blank(),
  paperheight = yml_blank(),
  cardwidth = yml_blank(),
  cardheight = yml_blank(),
  cols = yml_blank(),
  rows = yml_blank(),
  mainfont = yml_blank(),
  googlefonts = yml_blank(),
  ...
)

pagedown_person(...)

```

**Arguments**

name	The name
person	When you are creating business cards for numerous people with shared information, passing values to the person field can override the default values, which can be any of the values accepted by this function. Use <code>pagedown_person()</code> to do so or manually provide them using <code>list(field = value)</code> .
title	The title of the person
phone	A phone number
email	An email address
url	A website URL
address	The address
logo	A path to a logo file
.repeat	The number of cards to repeat. Note that the actual YAML field is <code>repeat</code> .
paperwidth	The paper width
paperheight	The paper height
cardwidth	The width of the card
cardheight	The height of the card
cols	The number of columns in the card grid
rows	The rows of columns in the card grid

mainfont	The font
googlefonts	A character vector of Google Fonts
...	additional named R objects, such as characters or lists, to transform into YAML

**Value**

a yml object

**See Also**

[use\\_rmarkdown\(\)](#)

Other pagedown: [yaml\\_pagedown\\_opts\(\)](#)

**Examples**

```

pagedown_business_card_template(
  name = "Jane Doe",
  title = "Miss Nobody",
  phone = "+1 123-456-7890",
  email = "jane.doe@example.com",
  url = "www.example.com",
  address = "2020 South Street,
Sunshine, CA 90000",
  logo = "logo.png",
  .repeat = 12
)

pagedown_business_card_template(
  phone = "+1 123-456-7890",
  url = "www.example.com",
  address = "2020 South Street,
Sunshine, CA 90000",
  logo = "logo.png",
  person = list(
    pagedown_person(
      name = "Jane Doe",
      title = "Miss Nobody",
      email = "jane.doe@example.com",
      .repeat = 6
    ),
    pagedown_person(
      name = "John Doe",
      title = "Mister Nobody",
      phone = "+1 777-777-7777", # overrides the default phone
      email = "john.doe@example.com",
      .repeat = 6
    )
  ),
  paperwidth = "8.5in",
  paperheight = "11in",
  cols = 4,

```

```
  rows = 3
)
```

---

pandoc\_template\_types *Use pandoc templates and custom highlight themes*

---

## Description

Pandoc has several built in templates and code highlighting themes that can be customized and included in the `template` and `highlight-style` YAML fields, respectively. `pandoc_template_types()` and `pandoc_highlight_styles()` return the available templates and highlight styles in pandoc, respectively. `use_pandoc_template()` creates a new file based on a template from pandoc or R Markdown and `use_pandoc_highlight_style()` creates a new highlight theme file based on an existing pandoc theme.

## Usage

```
pandoc_template_types()

pandoc_highlight_styles()

use_pandoc_template(type, path, source = c("rmarkdown", "pandoc"))

use_pandoc_highlight_style(theme, path)
```

## Arguments

<code>type</code>	The template type
<code>path</code>	The path to write the file to
<code>source</code>	The template source ("pandoc" or "rmarkdown")
<code>theme</code>	The name of the theme

## Value

a character vector

---

pkgdown_template	<i>Generate a full YAML template for your pkgdown site</i>
------------------	--

---

## Description

pkgdown includes three helpful `pkgdown::template_*`() functions to generate the navbar, reference, and article YAML for the `_pkgdown.yml` file. `pkgdown_template()` is a wrapper function that runs all three, combines them, and converts them to a `yml` object. You may also pass `pkgdown::template_*`() functions to `as_yml()` to convert the individual sections. `pkgdown_template()` is particularly useful with `use_pkgdown_yml()` to write directly to the `_pkgdown.yml` file.

## Usage

```
pkgdown_template(path = ".")
```

## Arguments

path	The path to your package directory
------	------------------------------------

## Value

a `yml` object

## See Also

[use\\_pkgdown\\_yml\(\)](#)

Other pkgdown: [yml\\_pkgdown\(\)](#)

## Examples

```
## Not run:  
# requires this to be a package directory  
pkgdown_template() %>%  
  use_pkgdown_yml()  
  
## End(Not run)
```

---

 read\_json

*Read and write to JSON and TOML*


---

**Description**

Read JSON and TOML files in as yml objects with `read_*`(`path`). Write yml objects out as JSON and YAML files with `write_as_*`(`path`). You can also provide `write_as_*`(`path`) a path to an existing `.yaml` file to translate to JSON or TOML. These functions rely on Hugo and blogdown, so you must have blogdown installed.

**Usage**

```
read_json(path)
```

```
read_toml(path)
```

```
write_as_json(
  .yaml = NULL,
  path = NULL,
  out = NULL,
  build_ignore = FALSE,
  git_ignore = FALSE,
  quiet = FALSE
)
```

```
write_as_toml(
  .yaml = NULL,
  path = NULL,
  out = NULL,
  build_ignore = FALSE,
  git_ignore = FALSE,
  quiet = FALSE
)
```

**Arguments**

<code>path</code>	a path to a JSON or TOML file
<code>.yaml</code>	a yml object created by <code>yaml()</code> , <code>as_yaml()</code> , or returned by a <code>yaml_*</code> ( <code>path</code> ) function
<code>out</code>	The path to write out to. If NULL, will write to the path but change the file extension to <code>.toml</code> or <code>.json</code> .
<code>build_ignore</code>	Logical. Should the file be added to the <code>.Rbuildignore</code> file?
<code>git_ignore</code>	Logical. Should the file be added to the <code>.gitignore</code> file?
<code>quiet</code>	Logical. Whether to message about what is happening.

**Value**

a yml object (if reading) or the path (if writing)



**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

use\_yaml

*Copy YAML code to your clipboard or write to a new R Markdown file***Description**

`use_yaml()` takes a yml object and puts the resulting YAML on your clipboard to paste into an R Markdown or YAML file. `use_rmarkdown()` takes the yml object and writes it to a new R Markdown file. You can add text to include in the body of the file. If it's not specified, `use_rmarkdown()` will use [setup\\_chunk\(\)](#) by default. You can also set a default for body using `options(ymlthis.rmd_body = "{your text}")`; see [use\\_rmd\\_defaults\(\)](#). Together with specifying default YAML (see [use\\_yaml\\_defaults\(\)](#)), `use_rmarkdown()` also serves as an ad-hoc way to make R Markdown templates. You may also supply `use_rmarkdown()` with an existing R Markdown file from which to read the YAML header; the YAML header from the template is then combined with `.yaml`, if it's supplied, and written to a new file. `use_index_rmd()` is a wrapper around `use_rmarkdown()` that specifically writes to a file called `index.Rmd`. By default, `use_yaml()` and `use_rmarkdown()` use the most recently printed YAML via [last\\_yaml\(\)](#).

**Usage**

```
use_yaml(.yaml = last_yaml())

use_rmarkdown(
  .yaml = last_yaml(),
  path,
  template = NULL,
  include_yaml = TRUE,
  include_body = TRUE,
  body = NULL,
  quiet = FALSE,
  open_doc = interactive(),
  overwrite = getOption("usethis.overwrite", FALSE)
)

use_index_rmd(
  .yaml = last_yaml(),
  path,
  template = NULL,
  include_yaml = TRUE,
```

```

include_body = TRUE,
body = NULL,
quiet = FALSE,
open_doc = interactive()
)

```

### Arguments

.yaml	a yaml object created by <code>yaml()</code> , <code>as_yaml()</code> , or returned by a <code>yaml_*</code> ( <code>)</code> function
path	A file path to write R Markdown file to
template	An existing R Markdown file to read YAML from
include_yaml	Logical. Include the template YAML?
include_body	Logical. Include the template body?
body	A character vector to use in the body of the R Markdown file. If no template is set, checks <code>getOption("yamlthis.rmd_body")</code> (see <code>use_rmd_defaults()</code> ) and otherwise uses <code>setup_chunk()</code> .
quiet	Logical. Whether to message about what is happening.
open_doc	Logical. Open the document after it's created? By default, this is TRUE if it is an interactive session and FALSE if not. Also checks that RStudio is available.
overwrite	Logical. If TRUE, overwrites the file without asking for permission. If FALSE, asks interactively if the user wishes to do so. Checks the user's <code>usethis.overwrite</code> option if set and is otherwise FALSE by default.

### Value

`use_yaml()` invisibly returns the input `yaml` object

### See Also

`code_chunk()` `setup_chunk()`

Other `yaml`: `asis_yaml_output()`, `bib2yaml()`, `draw_yaml_tree()`, `has_field()`, `read_json()`, `use_yaml_defaults()`, `use_yaml_file()`, `yaml_author()`, `yaml_blogdown_opts()`, `yaml_bookdown_opts()`, `yaml_citations()`, `yaml_clean()`, `yaml_distill_opts()`, `yaml_latex_opts()`, `yaml_output()`, `yaml_pagedown_opts()`, `yaml_params()`, `yaml_pkgdown()`, `yaml_reference()`, `yaml_replace()`, `yaml_resource_files()`, `yaml_rsconnect_email()`, `yaml_rarticles_opts()`, `yaml_runtime()`, `yaml_site_opts()`, `yaml_toc()`, `yaml_vignette()`

**Description**

use\_yaml\_defaults() takes a yml object and places code on the clipboard that will save the resulting YAML as the default for yml(). The code that is placed on the clipboard is raw YAML passed to ymlthis.default\_yaml via options(). Saving this code to your .Rprofile (see `usethis::edit_r_profile()`) will allow yml() or get\_yaml\_defaults() to return the saved YAML. use\_rmd\_defaults() does the same for ymlthis.rmd\_body, which is used in use\_rmarkdown() as the body text of the created R Markdown file.

**Usage**

```
use_yaml_defaults(.yaml)
```

```
use_rmd_defaults(x)
```

```
get_yaml_defaults()
```

```
get_rmd_defaults()
```

**Arguments**

.yaml a yml object created by yml(), as\_yaml(), or returned by a yml\_\*() function  
 x a character vector to use as the body text in use\_rmarkdown().

**See Also**

yml() get\_yaml\_defaults()

Other yml: asis\_yaml\_output(), bib2yaml(), draw\_yaml\_tree(), has\_field(), read\_json(), use\_yaml\_file(), use\_yaml(), yml\_author(), yml\_blogdown\_opts(), yml\_bookdown\_opts(), yml\_citations(), yml\_clean(), yml\_distill\_opts(), yml\_latex\_opts(), yml\_output(), yml\_pagedown\_opts(), yml\_params(), yml\_pkgdown(), yml\_reference(), yml\_replace(), yml\_resource\_files(), yml\_rsconnect\_email(), yml\_rarticles\_opts(), yml\_runtime(), yml\_site\_opts(), yml\_toc(), yml\_vignette()

---

use_yaml_file	<i>Write YAML to file</i>
---------------	---------------------------

---

**Description**

Write yml objects to a file. use\_yaml\_file() writes to any given file name. use\_output\_yaml() creates file \_output.yml, which can be used by multiple R Markdown documents. All documents located in the same directory as \_output.yml will inherit its output options. Options defined within document YAML headers will override those specified in \_output.yml. Note that use\_output\_yaml() plucks the output field from yml; any other YAML top-level fields will be ignored. use\_site\_yaml creates \_site.yml for use with R Markdown websites and third-party tools like the distill package (see [the R Markdown book for more](#)). use\_navbar\_yaml is a special type of site YAML that only specifies the navbar in \_navbar.yml use\_pkgdown\_yaml() and use\_bookdown\_yaml() write YAML files specific to those packages; see the [pkgdown](#) and [blogdown](#) documentation for more.

**Usage**

```
use_yaml_file(  
  .yaml = NULL,  
  path,  
  build_ignore = FALSE,  
  git_ignore = FALSE,  
  quiet = FALSE  
)
```

```
use_output_yaml(  
  .yaml = NULL,  
  path = ".",  
  build_ignore = FALSE,  
  git_ignore = FALSE,  
  quiet = FALSE  
)
```

```
use_site_yaml(  
  .yaml = NULL,  
  path = ".",  
  build_ignore = FALSE,  
  git_ignore = FALSE,  
  quiet = FALSE  
)
```

```
use_navbar_yaml(  
  .yaml = NULL,  
  path = ".",  
  build_ignore = FALSE,  
  git_ignore = FALSE,  
  quiet = FALSE  
)
```

```
use_pkgdown_yaml(  
  .yaml = NULL,  
  path = ".",  
  build_ignore = TRUE,  
  git_ignore = FALSE,  
  quiet = FALSE  
)
```

```
use_bookdown_yaml(  
  .yaml = NULL,  
  path = ".",  
  build_ignore = FALSE,  
  git_ignore = FALSE,  
  quiet = FALSE  
)
```

**Arguments**

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> () function
<code>path</code>	a file path to write the file to
<code>build_ignore</code>	Logical. Should the file be added to the <code>.Rbuildignore</code> file?
<code>git_ignore</code>	Logical. Should the file be added to the <code>.gitignore</code> file?
<code>quiet</code>	Logical. Whether to message about what is happening.

**Details**

By default, the yml package adds a new line to the end of files. Some environments, such as RStudio Projects, allow you to append new lines automatically. Thus, you may end up with 2 new lines at the end of your file. If you'd like to automatically remove the last new line in the file, set `options(ymlthis.remove_blank_line = TRUE)`.

**See Also**

`yml_bookdown_opts` `yml_bookdown_site` `yml_pkgdown` `yml_pkgdown_articles` `yml_pkgdown_docsearch`  
`yml_pkgdown_figures` `yml_pkgdown_news` `yml_pkgdown_reference`

Other yml: [asis\\_yml\\_output\(\)](#), [bib2yml\(\)](#), [draw\\_yml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yml\\_defaults\(\)](#), [use\\_yml\(\)](#), [yml\\_author\(\)](#), [yml\\_blogdown\\_opts\(\)](#), [yml\\_bookdown\\_opts\(\)](#), [yml\\_citations\(\)](#), [yml\\_clean\(\)](#), [yml\\_distill\\_opts\(\)](#), [yml\\_latex\\_opts\(\)](#), [yml\\_output\(\)](#), [yml\\_pagedown\\_opts\(\)](#), [yml\\_params\(\)](#), [yml\\_pkgdown\(\)](#), [yml\\_reference\(\)](#), [yml\\_replace\(\)](#), [yml\\_resource\\_files\(\)](#), [yml\\_rsconnect\\_email\(\)](#), [yml\\_rarticles\\_opts\(\)](#), [yml\\_runtime\(\)](#), [yml\\_site\\_opts\(\)](#), [yml\\_toc\(\)](#), [yml\\_vignette\(\)](#)

---

yml *Create a new yml object*

---

**Description**

`yml()` initializes a yml object. yml objects create valid YAML and print it cleanly to the console. By default, `yml()` looks for your name (using `getOption("usethis.full_name")`, `getOption("devtools.name")`, and `whoami::fullname()`) and uses today's date to use in the author and date fields, respectively. If you've set default YAML in `getOption("ymlthis.default_option")` (see [use\\_yml\\_defaults\(\)](#)), `yml()` will also use include those fields by default. `yml_empty()` is a wrapper that doesn't use any of these default YAML fields. `yml()` and all related `yml_*`() functions validate that the results are indeed valid YAML syntax, although not every function is able to check that the input fields are valid for the setting they are used in.

**Usage**

```
yml(.yml = NULL, get_yml = TRUE, author = TRUE, date = TRUE)
```

```
yml_empty()
```

**Arguments**

<code>.yml</code>	a character vector, yml object, or YAML-like list. See details.
<code>get_yml</code>	logical. Use YAML stored in <code>getOption("yamlthis.default_option")</code> ? By default, <code>yml()</code> includes if it exists.
<code>author</code>	logical. Get default author name?
<code>date</code>	logical. Get default date?

**Details**

`.yml` accepts a character vector of YAML, such as "author: Hadley Wickham", an object returned by `yamlthis` functions that start with `yaml_*()`, or a list object (e.g. `list(author = "Hadley Wickham")`). `.yml` objects are processed with `as_yml()`, a wrapper around `yaml::yaml.load()`. See that function for more details.

**Value**

a yml object

**Examples**

```

yml()

yml(date = FALSE)

"author: Hadley Wickham\n\ndate: 2014-09-12" %>%
  yml() %>%
  yml_title("Tidy Data") %>%
  yml_keywords(
    c("data cleaning", "data tidying", "relational databases", "R")
  )

yml() %>%
  yml_author(
    c("Yihui Xie", "Hadley Wickham"),
    affiliation = rep("RStudio", 2)
  ) %>%
  yml_date("07/04/2019") %>%
  yml_output(
    pdf_document(
      keep_tex = TRUE,
      includes = includes2(after_body = "footer.tex")
    )
  ) %>%
  yml_latex_opts(biblio_style = "apalike")

```

## Description

These functions add common top-level YAML fields for R Markdown documents, such as `author`, `date`, and `title`. Each takes a `yml` object and adds fields related to the function, as well as checking for duplicate fields and (where possible) checking for valid entries. `yml_toplevel()` is a catch-all function that will take any named R object and put in the top level of the YAML; it checks for duplicate fields but is unable to validate the input beyond that it is valid YAML syntax. Some R Markdown templates allow for additional variations of the YAML here. For instance, the `distill` package adds `url` and `affiliation_url` to the `author` field (see [yml\\_distill\\_author](#), which wraps [yml\\_author](#)). Several `yml_*`() functions also contain `...` which allow for these unique fields.

## Usage

```
yml_author(.yml, name = NULL, affiliation = NULL, email = NULL, ...)
```

```
yml_date(.yml, date = NULL, format = "")
```

```
yml_title(.yml, title)
```

```
yml_subtitle(.yml, subtitle)
```

```
yml_abstract(.yml, abstract)
```

```
yml_keywords(.yml, keywords)
```

```
yml_subject(.yml, subject)
```

```
yml_description(.yml, description)
```

```
yml_category(.yml, category)
```

```
yml_lang(.yml, lang)
```

```
yml_toplevel(.yml, ...)
```

## Arguments

<code>.yml</code>	a <code>yml</code> object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> () function
<code>name</code>	A character vector, name of the author(s)
<code>affiliation</code>	The author's affiliation; must match length of <code>name</code> , e.g. if <code>name</code> has length of two, <code>affiliation</code> must as well; use <code>NA</code> if you don't want to include an affiliation for a given author. Note that not all formats support the <code>affiliation</code> field.
<code>email</code>	The author email address. Note that not all formats support the <code>email</code> field.

...	additional named R objects, such as characters or lists, to transform into YAML
date	The date; by default this is " <code>r format(Sys.Date())`</code> ", which will populate the date automatically.
format	When the default date is used, the format passed to <code>format.Date()</code> .
title	A character vector, the title of the document
subtitle	A character vector, the subtitle of the document. Not all R Markdown formats use subtitles, so it may depend on what you use in the output field (see <code>yml_output()</code> ). It is available in <code>pdf_document()</code> , <code>html_document()</code> , and <code>word_document()</code> by default.
abstract	A character vector, the abstract. Long character vectors are automatically wrapped using valid YAML syntax. This field is not available in all output formats; it is available in <code>pdf_document()</code> and <code>html_document()</code> by default.
keywords	A character vector of keywords. This field is not available in all output formats; it is available in <code>pdf_document()</code> , <code>html_document()</code> , <code>word_document()</code> , <code>odt_document()</code> , and <code>powerpoint_presentation()</code> by default.
subject	A character vector, the subject of the document. This field is not available in all output formats; it is available in <code>pdf_document()</code> , <code>html_document()</code> , <code>word_document()</code> , <code>odt_document()</code> , and <code>powerpoint_presentation()</code> by default.
description	A character vector, a description of the document. This field is not available in all output formats; it is available in <code>word_document()</code> , <code>odt_document()</code> , and <code>powerpoint_presentation()</code> by default.
category	A character vector, the category of the document. This field is not available in all output formats; it is available in <code>word_document()</code> and <code>powerpoint_presentation()</code> by default.
lang	The document language using IETF language tags such as "en" or "en-US". The <a href="#">Language subtag lookup tool</a> can help find the appropriate tag.

**Value**

a yml object

**See Also**

Other yml: `asis_yaml_output()`, `bib2yaml()`, `draw_yaml_tree()`, `has_field()`, `read_json()`, `use_yaml_defaults()`, `use_yaml_file()`, `use_yaml()`, `yml_blogdown_opts()`, `yml_bookdown_opts()`, `yml_citations()`, `yml_clean()`, `yml_distill_opts()`, `yml_latex_opts()`, `yml_output()`, `yml_pagedown_opts()`, `yml_params()`, `yml_pkgdown()`, `yml_reference()`, `yml_replace()`, `yml_resource_files()`, `yml_rsconnect_email()`, `yml_rarticles_opts()`, `yml_runtime()`, `yml_site_opts()`, `yml_toc()`, `yml_vignette()`

**Examples**

```
yml_empty() %>%
  yml_author("Yihui Xie") %>%
  yml_date("02-02-2002") %>%
```



```

yml_title("R Markdown: An Introduction") %>%
yml_subtitle("Introducing ymlthis") %>%
yml_abstract("This paper will discuss a very important topic") %>%
yml_keywords(c("r", "reproducible research")) %>%
yml_subject("R Markdown") %>%
yml_description("An R Markdown reader") %>%
yml_category("r") %>%
yml_lang("en-US")

```

---

yml\_blank

*Return a blank object to be discarded from YAML*


---

## Description

ymlthis treats NULL, NA, and other common argument defaults as literal (e.g. author = NULL will produce "author: null"). yml\_blank() is a helper function to indicate that the field should not be included. yml\_blank() is primarily used as a default argument for fields that should not be included by default.

## Usage

```

yml_blank()

is_yml_blank(x)

```

## Arguments

x                    a field from a yml object

## Value

a yml\_blank object

## See Also

[yml\\_discard\(\)](#), [yml\\_replace\(\)](#)

## Examples

```

yml() %>%
  yml_replace(author = yml_blank()) %>%
  yml_discard(~is_yml_blank(.x))

```

---

yml\_blogdown\_opts      *Set Top-level YAML options for blogdown*

---

## Description

YAML in blogdown comes from a variety of sources. Most YAML will be for your posts, as described in the [blogdown book](#)). Common R Markdown fields can be used, but there are two other main sources for YAML fields: Hugo itself and the Hugo theme you are using. Hugo has numerous top-level YAML to control the output (see the [Hugo documentation](#)). `yml_blogdown_opts()` supports Hugo YAML. Your Hugo theme may also add fields to use. To find YAML specific to your theme, see [blogdown\\_template\(\)](#). In addition to these sources of YAML, the configuration file for your blog can also be in YAML, but this is not very common; most use a `config.toml` file, based on TOML (see the [blogdown book](#) for more).

## Usage

```
yml_blogdown_opts(  
  .yml,  
  draft = yml_blank(),  
  publishdate = yml_blank(),  
  weight = yml_blank(),  
  slug = yml_blank(),  
  aliases = yml_blank(),  
  audio = yml_blank(),  
  date = yml_blank(),  
  description = yml_blank(),  
  expiration_date = yml_blank(),  
  headless = yml_blank(),  
  images = yml_blank(),  
  keywords = yml_blank(),  
  layout = yml_blank(),  
  lastmod = yml_blank(),  
  link_title = yml_blank(),  
  resources = yml_blank(),  
  series = yml_blank(),  
  summary = yml_blank(),  
  title = yml_blank(),  
  type = yml_blank(),  
  url = yml_blank(),  
  videos = yml_blank(),  
  ...  
)
```

## Arguments

`.yml`      a yml object created by `yml()`, `as_yml()`, or returned by a `yml_*`(`)` function

draft	Logical. Set post as a draft? Draft posts will not be rendered if the site is built via <code>blogdown::build_site()</code> or <code>blogdown::hugo_build()</code> but will be rendered in the local preview mode. See <a href="#">Section D.3 of the blogdown book</a> .
publishdate	A future date to publish the post. Future posts are only rendered in the local preview mode
weight	This field can take a numeric value to tell Hugo the order of pages when sorting them, e.g., when you generate a list of all pages under a directory, and two posts have the same date, you may assign different weights to them to get your desired order on the list
slug	A character string used as the tail of the post URL. It is particularly useful when you define custom rules for permanent URLs. See <a href="#">Section 2.2.2 of the blogdown book</a> .
aliases	A character vector of one or more aliases (e.g., old published paths of renamed content) that will be created in the output directory structure
audio	A character vector of paths to audio files related to the page
date	The date assigned to this page. This is usually fetched from the date field in front matter, but this behavior is configurable.
description	The description for the content
expiration_date	the date at which the content should no longer be published by Hugo. Note that the actual YAML field is <code>expiryDate</code>
headless	if TRUE, sets a leaf bundle to be <a href="#">headless</a> .
images	A character vector of paths to images related to the page
keywords	A character vector of the keywords for the content.
layout	The layout Hugo should use while rendering the content. By default, layout matches type and is thus based on the directory. However, it's possible to use additional layouts within a type. See <a href="#">Hugo's Defining a Content Type documentation</a> .
lastmod	The date the content was last modified at
link_title	used for creating links to content. Note that the actual YAML field is <code>linkTitle</code>
resources	A named list. Used for configuring page bundle resources. See <a href="#">Hugo's Page Resources documentation</a>
series	A character vector of series this page belongs to
summary	A summary of the content in the <code>.Summary</code> Hugo page variable; see the <a href="#">content-summaries</a> section of Hugo's documentation.
title	The title for the content
type	The type of the content, which is based on the from the directory of the content if not specified
url	The full path to the content from the web root
videos	A character vector of paths to videos related to the page
...	additional named R objects, such as characters or lists, to transform into YAML

**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yml\\_author\(\)](#), [yml\\_bookdown\\_opts\(\)](#), [yml\\_citations\(\)](#), [yml\\_clean\(\)](#), [yml\\_distill\\_opts\(\)](#), [yml\\_latex\\_opts\(\)](#), [yml\\_output\(\)](#), [yml\\_pagedown\\_opts\(\)](#), [yml\\_params\(\)](#), [yml\\_pkgdown\(\)](#), [yml\\_reference\(\)](#), [yml\\_replace\(\)](#), [yml\\_resource\\_files\(\)](#), [yml\\_rsconnect\\_email\(\)](#), [yml\\_rarticles\\_opts\(\)](#), [yml\\_runtime\(\)](#), [yml\\_site\\_opts\(\)](#), [yml\\_toc\(\)](#), [yml\\_vignette\(\)](#)

**Examples**

```
yml() %>%
  yml_bookdown_opts(
    draft = TRUE,
    slug = "blog-post"
  )
```

---

yml\_bookdown\_opts

*Set Top-level YAML options for bookdown*


---

**Description**

bookdown uses YAML in three main places, as described in the [bookdown book](#): `index.Rmd`, `_output.yml`, and `_bookdown.yml`. `index.Rmd` can take most YAML. `_output.yml` is intended for output-related YAML, such as that produced by `yml() %>% yml_output(bookdown::pdf_book())`. `_bookdown.yml` is intended for configuring the build of the book. Pass the results of the `yml_*()` functions to `use_index_rmd()`, `use_bookdown_yaml()`, `use_output_yaml()` to write them to these files. `yml_bookdown_site()` adds the site: `"bookdown::bookdown_site"` to the YAML metadata.

**Usage**

```
yml_bookdown_opts(
  .yml,
  book_filename = yml_blank(),
  delete_merged_file = yml_blank(),
  before_chapter_script = yml_blank(),
  after_chapter_script = yml_blank(),
  edit = yml_blank(),
  history = yml_blank(),
  rmd_files = yml_blank(),
  rmd_subdir = yml_blank(),
  output_dir = yml_blank(),
```

```

      clean = yml_blank(),
      ...
    )

yml_bookdown_site(.yml)

```

### Arguments

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> function
<code>book_filename</code>	A character vector, the filename of the main <code>.Rmd</code> file, the <code>.Rmd</code> file that is created by merging all chapters. By default, it is called <code>"_main.Rmd"</code> .
<code>delete_merged_file</code>	Logical. Delete the main <code>.Rmd</code> file if it exists?
<code>before_chapter_script</code> , <code>after_chapter_script</code>	A character vector of one or more R scripts to be executed before or after each chapter
<code>edit</code>	A URL that collaborators can click to edit the <code>.Rmd</code> source document of the current page, usually a link to a GitHub repository. This link should have <code>%s</code> where the actual <code>.Rmd</code> filename for each page will go.
<code>history</code>	Similar to <code>edit</code> , a link to the edit/commit history of the current page.
<code>rmd_files</code>	A character vector, the order order of <code>.Rmd</code> files for the book. <code>rmd_files</code> can also be a named list where each element of the list is named for the output type, e.g. <code>"html"</code> or <code>"latex"</code> . By default, bookdown merges all <code>.Rmd</code> files by the order of filenames.
<code>rmd_subdir</code>	whether to search for book source <code>.Rmd</code> files in subdirectories (by default, only the root directory is searched). This may be either a boolean (e.g. <code>TRUE</code> will search for book source <code>.Rmd</code> files in the project directory and all subdirectories) or vector of paths if you want to search for book source <code>.Rmd</code> files in a subset of subdirectories.
<code>output_dir</code>	the output directory of the book (" <code>_book</code> " by default)
<code>clean</code>	a character vector of files and directories to be cleaned by the <code>bookdown::clean_book()</code> function.
<code>...</code>	additional named R objects, such as characters or lists, to transform into YAML

### Value

a yml object

### See Also

[use\\_index\\_rmd\(\)](#) [use\\_bookdown\\_yml\(\)](#) [use\\_output\\_yml\(\)](#)

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yml\(\)](#), [draw\\_yml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yml\\_defaults\(\)](#), [use\\_yml\\_file\(\)](#), [use\\_yml\(\)](#), [yml\\_author\(\)](#), [yml\\_blogdown\\_opts\(\)](#), [yml\\_citations\(\)](#), [yml\\_clean\(\)](#), [yml\\_distill\\_opts\(\)](#), [yml\\_latex\\_opts\(\)](#), [yml\\_output\(\)](#), [yml\\_pagedown\\_opts\(\)](#), [yml\\_params\(\)](#), [yml\\_pkgdown\(\)](#), [yml\\_reference\(\)](#), [yml\\_replace\(\)](#), [yml\\_resource\\_files\(\)](#),

```
yml_rsconnect_email(), yml_rarticles_opts(), yml_runtime(), yml_site_opts(), yml_toc(),
yml_vignette()
```

Other bookdown: `gitbook_config()`

## Examples

```
yml_empty() %>%
  yml_bookdown_opts(
    book_filename = "my-book.Rmd",
    before_chapter_script = c("script1.R", "script2.R"),
    after_chapter_script = "script3.R",
    edit = "https://github.com/rstudio/bookdown-demo/edit/master/%s",
    output_dir = "book-output",
    clean = c("my-book.bbl", "R-packages.bib")
  )
```

```
yml_empty() %>%
  yml_bookdown_opts(
    rmd_files = list(
      html = c("index.Rmd", "abstract.Rmd", "intro.Rmd"),
      latex = c("abstract.Rmd", "intro.Rmd")
    )
  )
```

```
x <- yml_empty() %>%
  yml_title("A Minimal Book Example") %>%
  yml_date(yml_code(Sys.Date())) %>%
  yml_author("Yihui Xie") %>%
  yml_bookdown_site() %>%
  yml_latex_opts(
    documentclass = "book",
    bibliography = c("book.bib", "packages.bib"),
    biblio_style = "apalike"
  ) %>%
  yml_citations(
    link_citations = TRUE
  ) %>%
  yml_description("This is a minimal example of using
the bookdown package to write a book.")
```

x

```
output_yml <- yml_empty() %>%
  yml_output(
    bookdown::gitbook(
      lib_dir = "assets",
      split_by = "section",
      config = gitbook_config(toolbar_position = "static")
    ),
```

```

      bookdown::pdf_book(keep_tex = TRUE),
      bookdown::html_book(css = "toc.css")
    )
  output_yml

```

---

yml\_citations

*Set citation-related YAML options*


---

## Description

`yml_citations()` sets citation-related YAML fields, such as specifying a bibliography file or style. For controlling the citation engine in PDF documents, see the `citation_package` argument in `rmarkdown::pdf_document()`.

## Usage

```

yml_citations(
  .yml,
  bibliography = yml_blank(),
  biblio_style = yml_blank(),
  biblio_title = yml_blank(),
  csl = yml_blank(),
  citation_abbreviations = yml_blank(),
  link_citations = yml_blank(),
  nocite = yml_blank(),
  suppress_bibliography = yml_blank()
)

```

## Arguments

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> ( <code>)</code> function
<code>bibliography</code>	a path to a bibliography file, such as a <code>.bib</code> file
<code>biblio_style</code>	bibliography style, when used with <b>natbib</b> and <b>biblatex</b> . Note that the actual YAML field is <code>biblio-style</code>
<code>biblio_title</code>	bibliography title, when used with <b>natbib</b> and <b>biblatex</b> . Note that the actual YAML field is <code>biblio-title</code>
<code>csl</code>	a path to a Citation Style Language (CSL) file. CSL files are used to specify the citation style; see the <b>CSL repository</b> for the CSL files of dozens of journals.
<code>citation_abbreviations</code>	Path to a CSL abbreviations JSON file. See the <b>pandoc-citeproc documentation</b> . Note that the actual YAML field is <code>citation-abbreviations</code> .
<code>link_citations</code>	Logical. Add citations hyperlinks to the corresponding bibliography entries? Note that the actual YAML field is <code>link-citations</code> .

nocite	Citation IDs (" <code>@item1</code> ") to include in the bibliography even if they are not cited in the document. Including the wildcard pattern " <code>@*</code> " will include all citations in the bibliography regardless of if they're cited in the document.
suppress_bibliography	Logical. Suppress bibliography?

**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

Other citations: [bib2yaml\(\)](#), [yaml\\_reference\(\)](#)

**Examples**

```
yml() %>%
  yml_citations(bibliography = "references.bib", csl = "aje.csl")
```

---

yml_clean	<i>Remove intermediate rendering files</i>
-----------	--

---

**Description**

R Markdown may create many documents while rendering the final product, for instance by using knitr to turn the R Markdown file to a Markdown file and then using Pandoc to convert to the final output. The `clean` field tells R Markdown whether or not to remove these files.

**Usage**

```
yml_clean(.yaml, clean)
```

**Arguments**

<code>.yaml</code>	a yml object created by <code>yml()</code> , <code>as_yaml()</code> , or returned by a <code>yml_*()</code> function
<code>clean</code>	Logical. Remove intermediate files that are created while making the R Markdown document?



**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

Other R Markdown: [yaml\\_params\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_vignette\(\)](#)

**Examples**

```
yml() %>%  
  # keep intermediate files  
  yml_clean(FALSE)
```

---

yml\_code

*Take code and write it as valid YAML*

---

**Description**

`yml_code()` takes R code and writes it as valid YAML to be evaluated during knitting. Note that `yml_code()` does not evaluate or validate the R code but only captures it to use in the YAML field. R code needs to be formatted differently when using in the `params` field for parameterized reports; `yml_params_code` will format this correctly for you.

**Usage**

```
yml_code(x)
```

```
yml_params_code(x)
```

**Arguments**

x                    valid R code

**Value**

a character vector with class `verbatim`

**See Also**

[yml\\_verbatim\(\)](#)

## Examples

```
yml_empty() %>%
  yml_date(yml_code(sys.Date()))

yml_empty() %>%
  yml_params(date = yml_params_code(sys.Date()))
```

---

<code>yml_distill_opts</code>	<i>Set Top-level YAML options for distill</i>
-------------------------------	---

---

## Description

`distill` uses many custom YAML fields to create some of its unique features, such as article metadata and citations. In addition to the arguments in `yml_distill_opts()`, `ymlthis` supports `distill` in a number of other ways. `yml_distill_author()` wraps `yml_author()` to include these extra used in `distill`. For a `distill` blog, you can specify the listings page a post belongs to, including an optional vector of other posts to list with it; `distill_listing()` is a helper function to pass to the `listing` argument to specify such pages. `distill` uses the same approach to navbars as R Markdown. `yml_navbar()` and friends will help you write the YAML for that. YAML specifying the site build, like the output field and navbars, can also be placed in `_site.yml`; see `yml_site_opts()` for further R Markdown website build options and `use_site_yaml()` for creating that file based on a `yml` object. `distill`'s YAML options are discussed in greater detail in the [articles on the distill website](#).

## Usage

```
yml_distill_opts(
  .yml,
  draft = yml_blank(),
  slug = yml_blank(),
  categories = yml_blank(),
  listing = yml_blank(),
  collection = yml_blank(),
  citation_url = yml_blank(),
  preview = yml_blank(),
  repository_url = yml_blank(),
  base_url = yml_blank(),
  compare_updates_url = yml_blank(),
  creative_commons = yml_blank(),
  twitter_site = yml_blank(),
  twitter_creator = yml_blank(),
  journal_title = yml_blank(),
  journal_issn = yml_blank(),
  journal_publisher = yml_blank(),
  volume = yml_blank(),
```

```

    issue = yml_blank(),
    doi = yml_blank(),
    resources = yml_blank(),
    ...
)

yml_distill_author(
  .yml,
  name = yml_blank(),
  url = yml_blank(),
  affiliation = yml_blank(),
  affiliation_url = yml_blank(),
  orcid_id = yml_blank()
)

distill_listing(listing_name = "posts", slugs = NULL)

distill_collection(
  collection_name = "post",
  feed_items_max = yml_blank(),
  disqus_name = yml_blank(),
  disqus_hidden = yml_blank(),
  share = yml_blank(),
  citations = yml_blank(),
  subscribe = yml_blank()
)

distill_resources(include = yml_blank(), exclude = yml_blank())

```

## Arguments

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> ( <code>)</code> function
<code>draft</code>	Logical. Set the post to be a draft? Draft posts won't be published.
<code>slug</code>	The abbreviated version of the citation included in the BibTeX entry. If you don't provide a slug then one will be automatically generated.
<code>categories</code>	A character vector, the post categories
<code>listing</code>	The listing a post is under; either a character vector, the output of <code>distill_listing()</code> , or a named list.
<code>collection</code>	Specify the RSS, sharing, and other settings of a listing; use <code>distill_collection()</code> or a named list.
<code>citation_url</code>	A URL to the article; automatically generated for blog articles
<code>preview</code>	a path or link to the preview image for your article. You can also set this by including <code>preview = TRUE</code> in an R Markdown code chunk in your document.
<code>repository_url</code>	A URL where the source code for your article can be found
<code>base_url</code>	Base (root) URL for the location where the website will be deployed (used for providing preview images for Open Graph and Twitter Card)

compare_updates_url	a URL that will show the differences between the article's current version and the version that was initially published
creative_commons	Designate articles that you create as Creative Commons licensed by specifying one of the standard Creative Commons licenses. Common options include "CC BY", "CC BY-SA", "CC BY-ND", and "CC BY-NC". See the <a href="#">distill vignette</a> for more details.
twitter_site	The Twitter handle for the site
twitter_creator	The Twitter handle for the creator
journal_title	The title of the journal
journal_issn	The issn of the journal
journal_publisher	The publisher of the journal
volume	The volume the article is on
issue	The issue the article is on
doi	The article Digital Object Identifier (DOI)
resources	Files to include or exclude while publishing. Use <code>distill_resources()</code> or a named list to specify.
...	additional named R objects, such as characters or lists, to transform into YAML
name	A character vector, name of the author(s)
url	the author URL
affiliation	The author's affiliation; must match length of name, e.g. if name has length of two, affiliation must as well; use NA if you don't want to include an affiliation for a given author. Note that not all formats support the affiliation field.
affiliation_url	the affiliation URL
orcid_id	the author's ORCID ID
listing_name	A character vector, the name of the listing
slugs	A character vector of the posts to include in the listing
collection_name	A character vector, the name of the collection
feed_items_max	Number of articles to include in the RSS feed (default: 20). Specify FALSE to have no limit on the number of items included in the feed.
disqus_name	A shortname for the disqus comments section ( <code>base_url</code> field is required in order to use Disqus)
disqus_hidden	Logical. Show full text of disqus comments? By default, this is FALSE so as not to obscure the bibliography and other appendices.
share	Share buttons to include. Choices: "twitter", "linkedin", "facebook", "google-plus", and "pinterest". ( <code>base_url</code> field is required in order to use sharing links)

citations	Logical. If your <code>_site.yml</code> file provides a <code>base_url</code> field, an article citation appendix and related metadata will be included automatically within all published posts. Set to <code>FALSE</code> to disable this behavior.
subscribe	a path to a HTML file enabling readers to subscribe. See the <a href="#">distill vignette on blog posts</a> for more details.
include, exclude	a character vector of files to explicitly include or exclude when publishing a post. Can use wild cards, such as <code>"*.csv"</code> .

**Value**

a yml object

**See Also**

[use\\_site\\_yaml\(\)](#) [use\\_rmarkdown\(\)](#)

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

Other websites: [yaml\\_pkgdown\(\)](#), [yaml\\_site\\_opts\(\)](#)

**Examples**

```
post_listing <- distill_listing(
  slugs = c(
    "2016-11-08-sharpe-ratio",
    "2017-11-09-visualizing-asset-returns",
    "2017-09-13-asset-volatility"
  )
)

yaml() %>%
  yaml_title("Gallery of featured posts") %>%
  yaml_distill_opts(listing = post_listing)

yaml_empty() %>%
  yaml_title("Reproducible Finance with R") %>%
  yaml_description("Exploring reproducible finance with the R statistical,
  computing environment.") %>%
  yaml_site_opts(name = "reproducible-finance-with-r") %>%
  yaml_distill_opts(
    base_url = "https://beta.rstudioconnect.com/content/3776/",
    collection = distill_collection(
      feed_items_max = 30,
      disqus_name = "reproducible-finance-with-r",
      disqus_hidden = FALSE,
      share = c("twitter", "linkedin")
    )
  )
```

```
)
)
```

---

yml_handlers	<i>Set handlers to process the way YAML is printed</i>
--------------	--

---

### Description

ymlthis uses the yml package to process and validate YAML; this package also lets you specify how fields and values are printed using a list of handler functions. `yml_handlers()` specifies defaults for the package used in the print statement. See `yml::yml.load()` for more on specifying handlers.

### Usage

```
yml_handlers()
```

---

yml_latex_opts	<i>Set LaTeX YAML options for PDF output</i>
----------------	--

---

### Description

`yml_latex_opts()` sets top-level YAML fields for LaTeX options used by pandoc ([see the documentation](#), from which these descriptions were derived), as when making a PDF document with `pdf_document()`.

### Usage

```
yml_latex_opts(
  .yml,
  block_headings = yml_blank(),
  classoption = yml_blank(),
  documentclass = yml_blank(),
  geometry = yml_blank(),
  indent = yml_blank(),
  linestretch = yml_blank(),
  margin_left = yml_blank(),
  margin_right = yml_blank(),
  margin_top = yml_blank(),
  margin_bottom = yml_blank(),
  pagestyle = yml_blank(),
  papersize = yml_blank(),
  secnumdepth = yml_blank(),
  fontenc = yml_blank(),
  fontfamily = yml_blank(),
```

```

fontfamilyoptions = yml_blank(),
fontsize = yml_blank(),
mainfont = yml_blank(),
sansfont = yml_blank(),
monofont = yml_blank(),
mathfont = yml_blank(),
CJKmainfont = yml_blank(),
mainfontoptions = yml_blank(),
sansfontoptions = yml_blank(),
monofontoptions = yml_blank(),
mathfontoptions = yml_blank(),
CJKoptions = yml_blank(),
microtypeoptions = yml_blank(),
colorlinks = yml_blank(),
linkcolor = yml_blank(),
filecolor = yml_blank(),
citecolor = yml_blank(),
urlcolor = yml_blank(),
toccolor = yml_blank(),
links_as_notes = yml_blank(),
lof = yml_blank(),
lot = yml_blank(),
thanks = yml_blank(),
toc = yml_blank(),
toc_depth = yml_blank(),
biblatexoptions = yml_blank(),
biblio_style = yml_blank(),
biblio_title = yml_blank(),
bibliography = yml_blank(),
natbiboptions = yml_blank()
)

```

## Arguments

.yml	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> ( <code>)</code> function
block_headings	make paragraph and subparagraph (fourth- and fifth-level headings, or fifth- and sixth-level with book classes) free-standing rather than run-in; requires further formatting to distinguish from subsection (third- or fourth-level headings). Note that the YAML field is actually called <code>block-headings</code> .
classoption	a character vector of options for document class, e.g. "oneside"
documentclass	the document class usually "article", "book", or "report"
geometry	a character vector of options for the <a href="#">geometry LaTeX package</a> , e.g. "margin=lin"
indent	Logical. Use document class settings for indentation? The default LaTeX template otherwise removes indentation and adds space between paragraphs.
linestretch	adjusts line spacing using the <a href="#">setspace LaTeX package</a> , e.g. 1.25, 1.5

margin_left, margin_right, margin_top, margin_bottom	sets margins if geometry is not used, otherwise geometry overrides these. Note that the actual YAML fields use - instead of _, e.g. margin-left.
pagestyle	control the pagestyle LaTeX command: the default article class supports "plain" (default), "empty" (no running heads or page numbers), and "headings" (section titles in running heads)
papersize	paper size, e.g. letter, a4
secnumdepth	numbering depth for sections (with --number-sections pandoc)
fontenc	allows font encoding to be specified through <b>fontenc LaTeX package</b> (with pdfflatex); default is "T1" (see <b>LaTeX font encodings guide</b> )
fontfamily	font package for use with pdfflatex: TeX Live includes many options, documented in the <b>LaTeX Font Catalogue</b> . The default is "Latin Modern".
fontfamilyoptions	a character vector of options for fontfamily.
fontsize	font size for body text. The standard classes allow "10pt", "11pt", and "12pt".
mainfont, sansfont, monofont, mathfont, CJKmainfont	font families for use with xelatex or lualatex: take the name of any system font, using the <b>fontspec LaTeX package</b> . CJKmainfont uses the <b>xecjk LaTeX package</b> .
mainfontoptions, sansfontoptions, monofontoptions, mathfontoptions, CJKoptions	a character vector of options to use with mainfont, sansfont, monofont, mathfont, CJKmainfont in xelatex and lualatex. Allow for any choices available through fontspec.
microtypeoptions	a character vector of options to pass to the <b>microtype LaTeX package</b> .
colorlinks	Logical. Add color to link text? Automatically enabled if any of linkcolor, filecolor, citecolor, urlcolor, or toccolor are set.
linkcolor, filecolor, citecolor, urlcolor, toccolor	color for internal links, external links, citation links, linked URLs, and links in table of contents, respectively: uses options allowed by <b>xcolor</b> , including the dvipsnames, svgnames, and x11names lists
links_as_notes	Logical. Print links as footnotes? Note that the actual YAML field is links-as-notes
lof, lot	Logical. Include list of figures or list of tables?
thanks	contents of acknowledgments footnote after document title
toc	include table of contents
toc_depth	level of section to include in table of contents. Note that the actual YAML field is toc-depth
biblatexoptions	list of options for <b>biblatex</b> .
biblio_style	bibliography style, when used with <b>natbib</b> and <b>biblatex</b> . Note that the actual YAML field is biblio-style
biblio_title	bibliography title, when used with <b>natbib</b> and <b>biblatex</b> . Note that the actual YAML field is biblio-title
bibliography	a path to the bibliography file to use for references
natbiboptions	a character vector of options for <b>natbib</b>



**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

**Examples**

```
yml() %>%
  yaml_output(pdf_document()) %>%
  yaml_latex_opts(
    fontfamily = "Fira Sans Thin",
    fontsize = "11pt",
    links_as_notes = TRUE
  )
```

---

yml\_load

*Load YAML from string*


---

**Description**

yml\_load() is a wrapper for [yaml::yaml.load\(\)](#) that also converts the object to the yml class.

**Usage**

```
yml_load(x)
```

**Arguments**

x an object to pass to [yaml::yaml.load\(\)](#)

**Examples**

```
c("title: my title", "author: Malcolm Barrett") %>%
  yml_load()
```

---

yml\_output

*Capture, validate, and write output YAML*


---

## Description

`yml_output()` writes valid YAML for the output field of R Markdown YAML. `yml_output()` captures the actual output functions, such as `pdf_document()`, and translates them to YAML. This function accepts multiple output formats (separated by commas) and validates each by evaluating the function internally. The YAML fields in under output come from arguments in their respective R functions. If you wanted to see the available fields in `pdf_document()`, for instance, you would read the documentation for that function using `?pdf_document`.

## Usage

```
yml_output(.yml, ...)
```

## Arguments

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> () function
<code>...</code>	valid R code calling functions that return objects of class <code>rmarkdown_output_format</code> , such as the <code>*_document()</code> functions in <code>rmarkdown</code> .

## Value

a yml object

## See Also

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

## Examples

```
yml() %>%
  yml_output(html_document())

yml() %>%
  yml_output(
    pdf_document(keep_tex = TRUE, includes = includes2(after_body = "footer.tex")),
    bookdown::html_document2()
  )
```

---

yml\_pagedown\_opts      *Top-level YAML options for pagedown*


---

## Description

pagedown offers several output functions for paginated output, resumes, business cards, theses, and more as described in the [pagedown vignette](#). pagedown also accepts a few custom top-level YAML. See [pagedown\\_business\\_card\\_template\(\)](#) for more on setting up the YAML for a business card.

## Usage

```
yml_pagedown_opts(
  .yml,
  toc = yml_blank(),
  toc_title = yml_blank(),
  lot = yml_blank(),
  lot_title = yml_blank(),
  chapter_name = yml_blank(),
  links_to_footnotes = yml_blank(),
  paged_footnotes = yml_blank()
)
```

## Arguments

.yml	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> () function
toc	Logical. Use a table of contents?
toc_title	The title for the table of contents. Note that the actual YAML field is <code>toc-title</code>
lot	Logical. Use a list of figures?
lot_title	The title for the list of figures. Note that the actual YAML field is <code>lot-title</code>
chapter_name	The chapter title prefix
links_to_footnotes	Logical. Transform all the URLs to footnotes? Note that the actual YAML field is <code>links-to-footnotes</code>
paged_footnotes	Logical. Render notes as footnotes? Note that the actual YAML field is <code>paged-footnotes</code>

## Value

a yml object

**See Also**

Other yml: `asis_yaml_output()`, `bib2yaml()`, `draw_yaml_tree()`, `has_field()`, `read_json()`, `use_yaml_defaults()`, `use_yaml_file()`, `use_yaml()`, `yaml_author()`, `yaml_blogdown_opts()`, `yaml_bookdown_opts()`, `yaml_citations()`, `yaml_clean()`, `yaml_distill_opts()`, `yaml_latex_opts()`, `yaml_output()`, `yaml_params()`, `yaml_pkgdown()`, `yaml_reference()`, `yaml_replace()`, `yaml_resource_files()`, `yaml_rsconnect_email()`, `yaml_rarticles_opts()`, `yaml_runtime()`, `yaml_site_opts()`, `yaml_toc()`, `yaml_vignette()`

Other pagedown: `pagedown_business_card_template()`

**Examples**

```
yml() %>%
  yml_pagedown_opts(
    toc = TRUE,
    toc_title = "TOC",
    chapter_name = c("CHAPTER\\ ", "."),
    links_to_footnotes = TRUE
  )
```

---

yml\_params

---

*Parameterize an R Markdown report using Shiny components*


---

**Description**

R Markdown lets you add dynamic parameters to your report using the `params` YAML field (see the [R Markdown book](#) for examples); parameterized reports are also used in RStudio Connect. The values of these variables can be called inside your R Markdown document using `params$field_name`. **There are several ways to change the values of the parameters:** manually change the YAML, use the `params` argument in `rmarkdown::render()`, or knit with `parameters`, which launches a Shiny app to select values for each. `yml_params()` accepts any number of named R objects to set as YAML fields. You can also pass arguments to the underlying Shiny functions using YAML. To set a shiny component, use the `shiny_*`() helper functions. `shiny_params()` captures a Shiny output function and transforms it to YAML. However, R Markdown supports only a limited number of components; each of these is included as a function starting with `shiny_*`(), e.g. `shiny_checkbox()`

**Usage**

```
yml_params(.yaml, ...)
```

```
shiny_params(.shiny)
```

```
shiny_checkbox(label, value = FALSE, width = NULL)
```

```
shiny_numeric(label, value, min = NA, max = NA, step = NA, width = NULL)
```

```
shiny_slider(
```

```
    label,  
    min,  
    max,  
    value,  
    step = NULL,  
    round = FALSE,  
    format = NULL,  
    ticks = TRUE,  
    animate = FALSE,  
    width = NULL,  
    sep = ",",  
    pre = NULL,  
    post = NULL,  
    timeFormat = NULL,  
    timezone = NULL,  
    dragRange = TRUE  
  )  
  
  shiny_date(  
    label,  
    value = NULL,  
    min = NULL,  
    max = NULL,  
    format = "yyyy-mm-dd",  
    startview = "month",  
    weekstart = 0,  
    language = "en",  
    width = NULL,  
    autoclose = TRUE,  
    datesdisabled = NULL,  
    daysofweekdisabled = NULL  
  )  
  
  shiny_text(label, value = "", width = NULL, placeholder = NULL)  
  
  shiny_file(  
    label,  
    multiple = FALSE,  
    accept = NULL,  
    width = NULL,  
    buttonLabel = "Browse...",  
    placeholder = "No file selected"  
  )  
  
  shiny_radio(  
    label,  
    choices = NULL,  
    selected = NULL,  
  )
```

```

    inline = FALSE,
    width = NULL,
    choiceNames = NULL,
    choiceValues = NULL
  )

shiny_select(
  label,
  choices,
  selected = NULL,
  multiple = FALSE,
  selectize = TRUE,
  width = NULL,
  size = NULL
)

shiny_password(label, value = "", width = NULL, placeholder = NULL)

```

### Arguments

.yml	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*()</code> function
...	additional named R objects, such as characters or lists, to transform into YAML
.shiny	a Shiny function call to capture and convert to YAML
label	Display label for the control, or NULL for no label.
value	Initial value (TRUE or FALSE).
width	The width of the input, e.g. '400px', or '100%'; see <a href="#">shiny::validateCssUnit()</a>
min	Minimum allowed value
max	Maximum allowed value
step	Interval to use when stepping between min and max
round	TRUE to round all values to the nearest integer; FALSE if no rounding is desired; or an integer to round to that number of digits (for example, 1 will round to the nearest 10, and -2 will round to the nearest .01). Any rounding will be applied after snapping to the nearest step.
format	The format of the date to display in the browser. Defaults to "yyyy-mm-dd".
ticks	FALSE to hide tick marks, TRUE to show them according to some simple heuristics.
animate	TRUE to show simple animation controls with default settings; FALSE not to; or a custom settings list, such as those created using <a href="#">shiny::animationOptions()</a>
sep	Separator between thousands places in numbers.
pre	A prefix string to put in front of the value.
post	A suffix string to put after the value.
timeFormat	Only used if the values are Date or POSIXt objects. A time format string, to be passed to the Javascript strftime library. See <a href="https://github.com/samsonjs/strftime">https://github.com/samsonjs/strftime</a> for more details. The allowed format specifications are very similar,

	but not identical, to those for R's <code>base::strftime()</code> function. For Dates, the default is "%F" (like "2015-07-01"), and for POSIXt, the default is "%F %T" (like "2015-07-01 15:32:10").
timezone	Only used if the values are POSIXt objects. A string specifying the time zone offset for the displayed times, in the format "+HHMM" or "-HHMM". If NULL (the default), times will be displayed in the browser's time zone. The value "+0000" will result in UTC time.
dragRange	This option is used only if it is a range slider (with two values). If TRUE (the default), the range can be dragged. In other words, the min and max can be dragged together. If FALSE, the range cannot be dragged.
startview	The date range shown when the input object is first clicked. Can be "month" (the default), "year", or "decade".
weekstart	Which day is the start of the week. Should be an integer from 0 (Sunday) to 6 (Saturday).
language	The language used for month and day names. Default is "en". Other valid values include "ar", "az", "bg", "bs", "ca", "cs", "cy", "da", "de", "el", "en-AU", "en-GB", "eo", "es", "et", "eu", "fa", "fi", "fo", "fr-CH", "fr", "gl", "he", "hr", "hu", "hy", "id", "is", "it-CH", "it", "ja", "ka", "kh", "kk", "ko", "kr", "lt", "lv", "me", "mk", "mn", "ms", "nb", "nl-BE", "nl", "no", "pl", "pt-BR", "pt", "ro", "rs-latin", "rs", "ru", "sk", "sl", "sq", "sr-latin", "sr", "sv", "sw", "th", "tr", "uk", "vi", "zh-CN", and "zh-TW".
autoclose	Whether or not to close the datepicker immediately when a date is selected.
datesdisabled	Which dates should be disabled. Either a Date object, or a string in yyyy-mm-dd format.
daysofweekdisabled	Days of the week that should be disabled. Should be a integer vector with values from 0 (Sunday) to 6 (Saturday).
placeholder	A character string giving the user a hint as to what can be entered into the control. Internet Explorer 8 and 9 do not support this option.
multiple	Whether the user should be allowed to select and upload multiple files at once. <b>Does not work on older browsers, including Internet Explorer 9 and earlier.</b>
accept	A character vector of "unique file type specifiers" which gives the browser a hint as to the type of file the server expects. Many browsers use this prevent the user from selecting an invalid file. A unique file type specifier can be: <ul style="list-style-type: none"> <li>• A case insensitive extension like .csv or .rds.</li> <li>• A valid MIME type, like text/plain or application/pdf</li> <li>• One of audio/*, video/*, or image/* meaning any audio, video, or image type, respectively.</li> </ul>
buttonLabel	The label used on the button. Can be text or an HTML tag object.
choices	List of values to select from (if elements of the list are named then that name rather than the value is displayed to the user). If this argument is provided, then choiceNames and choiceValues must not be provided, and vice-versa. The values should be strings; other types (such as logicals and numbers) will be coerced to strings.

selected	The initially selected value. If not specified, then it defaults to the first item in choices. To start with no items selected, use character(0).
inline	If TRUE, render the choices inline (i.e. horizontally)
choiceNames, choiceValues	List of names and values, respectively, that are displayed to the user in the app and correspond to the each choice (for this reason, choiceNames and choiceValues must have the same length). If either of these arguments is provided, then the other <i>must</i> be provided and choices <i>must not</i> be provided. The advantage of using both of these over a named list for choices is that choiceNames allows any type of UI object to be passed through (tag objects, icons, HTML code, ...), instead of just simple text. See Examples.
selectize	Whether to use <b>selectize.js</b> or not.
size	Number of items to show in the selection box; a larger number will result in a taller box. Not compatible with selectize=TRUE. Normally, when multiple=FALSE, a select input will be a drop-down list, but when size is set, it will be a box instead.

**Value**

a yml object

**See Also**

[yml\\_params\\_code\(\)](#)

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

Other R Markdown: [yaml\\_clean\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_vignette\(\)](#)

Other shiny: [yaml\\_runtime\(\)](#)

**Examples**

```
yml() %>%
  yml_params(
    z = "z",
    x = shiny_numeric("Starting value", 23),
    no = shiny_checkbox("No option?"),
    y = shiny_slider("Data range", 0, 1, .5, round = TRUE)
  )
```



**Description**

These functions set YAML for various pkgdown options to be used in `_pkgdown.yml`. The options are described in greater depth in the [pkgdown vignette](#) and in the help pages for `pkgdown::build_site()`, `pkgdown::build_articles()`, `pkgdown::build_reference()`, and `pkgdown::build_tutorials()`. Essentially, they control the build of vignettes and function references. pkgdown also uses the same approach to navbars as R Markdown. `yml_navbar()` and friends will help you write the YAML for that. A useful approach to writing pkgdown YAML might be to use `pkgdown_template()` to build a template based on your package directory, modify with `yml_pkgdown_*`() and `pkgdown_*`() functions or `yml_replace()` and `yml_discard()`, then pass the results to `use_pkgdown_yml()` to write to `_pkgdown.yml`

**Usage**

```
yml_pkgdown(.yml, as_is = yml_blank(), extension = yml_blank())
```

```
yml_pkgdown_opts(
  .yml,
  site_title = yml_blank(),
  destination = yml_blank(),
  url = yml_blank(),
  toc_depth = yml_blank()
)
```

```
yml_pkgdown_development(
  .yml,
  mode = yml_blank(),
  dev_destination = yml_blank(),
  version_label = yml_blank(),
  version_tooltip = yml_blank()
)
```

```
yml_pkgdown_template(
  .yml,
  bootswatch = yml_blank(),
  ganalytics = yml_blank(),
  noindex = yml_blank(),
  package = yml_blank(),
  path = yml_blank(),
  assets = yml_blank(),
  default_assets = yml_blank()
)
```

```
yml_pkgdown_reference(.yml, ...)
```

```
pkgdown_ref(  
  title = yml_blank(),  
  desc = yml_blank(),  
  contents = yml_blank(),  
  exclude = yml_blank(),  
  ...  
)  
  
yml_pkgdown_news(.yml, one_page = yml_blank())  
  
yml_pkgdown_articles(.yml, ...)  
  
pkgdown_article(  
  title = yml_blank(),  
  desc = yml_blank(),  
  contents = yml_blank(),  
  exclude = yml_blank(),  
  ...  
)  
  
yml_pkgdown_tutorial(.yml, ...)  
  
pkgdown_tutorial(  
  name = yml_blank(),  
  title = yml_blank(),  
  tutorial_url = yml_blank(),  
  source = yml_blank(),  
  ...  
)  
  
yml_pkgdown_figures(  
  .yml,  
  dev = yml_blank(),  
  dpi = yml_blank(),  
  dev.args = yml_blank(),  
  fig.ext = yml_blank(),  
  fig.width = yml_blank(),  
  fig.height = yml_blank(),  
  fig.retina = yml_blank(),  
  fig.asp = yml_blank(),  
  ...  
)  
  
yml_pkgdown_docsearch(  
  .yml,  
  api_key = yml_blank(),  
  index_name = yml_blank(),
```

```
  doc_url = yml_blank()
)
```

### Arguments

.yml	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> ( <code>)</code> function
as_is	Logical. Use the <code>output_format</code> and options that you have specified?
extension	The output extension, e.g. "pdf".
site_title	The title of the website (by default, this is the package name). Note that the actual YAML is <code>title</code> (specified as <code>site_title</code> to avoid duplication with content titles).
destination	The path where the site should be rendered ("docs/" by default)
url	URL where the site will be published; setting the URL will allow other pkgdown sites to link to your site when needed, generate a <code>sitemap.xml</code> to increase the searchability of your site, and generate a CNAME.
toc_depth	The depth of the headers included in the Table of Contents. Note that the actual YAML is <code>depth</code> and is nested under <code>toc</code> .
mode	The development mode of the site, one of: "auto", "release", "development", or "unreleased". <code>development</code> controls where the site is built; the color of the package version; the optional tooltip associated with the version; and the indexing of the site by search engines. See <code>?pkgdown::build_site()</code> for more details.
dev_destination	The subdirectory used for the development site, which defaults to "dev/". Note that the actual YAML is <code>destination</code> and is nested under <code>development</code> .
version_label	Label to display for "development" and "unreleased" mode. One of: "danger" (the default), "default", "info", or "warning".
version_tooltip	A custom message to include in the version tooltip
bootswatch	A bootswatch theme for the site. See the options at <a href="https://rstudio.github.io/shinythemes/">https://rstudio.github.io/shinythemes/</a> .
ganalytics	A Google Analytics tracking id
noindex	Logical. Suppress indexing of your pages by web robots?
package	an R package with with directories <code>inst/pkgdown/assets</code> and <code>inst/pkgdown/templates</code> to override the default templates and add additional assets; alternatively, you can specify this in <code>path</code> and <code>assets</code>
path	A path to templates with which to override the default pkgdown templates
assets	A path to additional assets to include
default_assets	Logical. Include default assets?
...	additional named R objects, such as characters or lists, to transform into YAML
title	The title of the article, reference, tutorial, or other resource
desc	A description of the article or reference
contents	The contents, which can also be dplyr-style tidy selectors (e.g. " <code>contains('index')</code> ").

<code>exclude</code>	What to exclude of the what's captured by contents
<code>one_page</code>	Logical. Create one page per release for NEWS.md?
<code>name</code>	The name of the file
<code>tutorial_url</code>	The tutorial URL to embed in an iframe
<code>source</code>	A URL to the source code of the tutorial
<code>dev</code>	The graphics device (default: "grDevices::png")
<code>dpi</code>	The DPI (default: 96)
<code>dev.args</code>	A vector of arguments to pass to dev
<code>fig.ext</code>	The figure extension (default: "png")
<code>fig.width</code>	The figure width (default: 7.2916667)
<code>fig.height</code>	The figure height (default: NULL)
<code>fig.retina</code>	The figure retina value (default: 2)
<code>fig.asp</code>	The aspect ratio (default: 1.618)
<code>api_key</code>	The API key provided by docsearch (see the <a href="#">pkgdown vignette</a> )
<code>index_name</code>	The index name provided by docsearch (see the <a href="#">pkgdown vignette</a> )
<code>doc_url</code>	the URL specifying the location of your documentation. Note that the actual YAML field is <code>url</code> but is nested.

**Value**

a yml object

**See Also**

[use\\_pkgdown\\_yml\(\)](#) [yml\\_navbar\(\)](#)

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yml\\_defaults\(\)](#), [use\\_yml\\_file\(\)](#), [use\\_yml\(\)](#), [yml\\_author\(\)](#), [yml\\_blogdown\\_opts\(\)](#), [yml\\_bookdown\\_opts\(\)](#), [yml\\_citations\(\)](#), [yml\\_clean\(\)](#), [yml\\_distill\\_opts\(\)](#), [yml\\_latex\\_opts\(\)](#), [yml\\_output\(\)](#), [yml\\_pagedown\\_opts\(\)](#), [yml\\_params\(\)](#), [yml\\_reference\(\)](#), [yml\\_replace\(\)](#), [yml\\_resource\\_files\(\)](#), [yml\\_rsconnect\\_email\(\)](#), [yml\\_rarticles\\_opts\(\)](#), [yml\\_runtime\(\)](#), [yml\\_site\\_opts\(\)](#), [yml\\_toc\(\)](#), [yml\\_vignette\(\)](#)

Other pkgdown: [pkgdown\\_template\(\)](#)

Other websites: [yml\\_distill\\_opts\(\)](#), [yml\\_site\\_opts\(\)](#)

**Examples**

```
yml_empty() %>%
  yml_pkgdown(
    as_is = TRUE,
    extension = "pdf"
  ) %>%
  yml_pkgdown_reference(
    pkgdown_ref(
      title = "pkgdown functions",
```

```

      contents = "contains('function_name')"
    )
  ) %>%
  yml_pkgdown_articles(
    pkgdown_article(
      title = "Introduction to the package"
    )
  )
)

```

---

yml\_reference

*Write references as YAML fields*


---

### Description

`yml_reference()` creates YAML fields for references to be used in citation. `reference()` is a simple function to add references to `yml_reference()`. The easiest way to add references to an R Markdown file is to use a bibliography file, such as `.bib`, in the bibliography field (see [yml\\_citations\(\)](#)). For documents with very few references, however, it might be useful to make the references self-contained in the YAML. `yml_reference()` can also transform to YAML bibentry and citation objects created by `bibentry()` and `citation()`. To cite many R packages and convert the references to YAML, it may be better to use `knitr::write_bib()` to write a bibliography file and convert it with `bib2yaml()`.

### Usage

```
yml_reference(.yaml, ..., .bibentry = NULL)
```

```
reference(id = NULL, ...)
```

### Arguments

<code>.yaml</code>	a yaml object created by <code>yaml()</code> , <code>as_yaml()</code> , or returned by a <code>yaml_*()</code> function
<code>...</code>	Fields relevant to the citation (e.g. bibtex fields)
<code>.bibentry</code>	An object created by <code>bibentry()</code> or <code>citation()</code> . Note that this requires <code>pandoc-citeproc</code> to be installed.
<code>id</code>	a character vector to use as the reference ID

### Value

a yaml object

**See Also**

Other yml: `asis_yaml_output()`, `bib2yaml()`, `draw_yaml_tree()`, `has_field()`, `read_json()`, `use_yaml_defaults()`, `use_yaml_file()`, `use_yaml()`, `yaml_author()`, `yaml_blogdown_opts()`, `yaml_bookdown_opts()`, `yaml_citations()`, `yaml_clean()`, `yaml_distill_opts()`, `yaml_latex_opts()`, `yaml_output()`, `yaml_pagedown_opts()`, `yaml_params()`, `yaml_pkgdown()`, `yaml_replace()`, `yaml_resource_files()`, `yaml_rsconnect_email()`, `yaml_rarticles_opts()`, `yaml_runtime()`, `yaml_site_opts()`, `yaml_toc()`, `yaml_vignette()`

Other citations: `bib2yaml()`, `yaml_citations()`

**Examples**

```
ref <- reference(
  id = "fenner2012a",
  title = "One-click science marketing",
  author = list(
    family = "Fenner",
    given = "Martin"
  ),
  `container-title` = "Nature Materials",
  volume = 11L,
  URL = "https://doi.org/10.1038/nmat3283",
  DOI = "10.1038/nmat3283",
  issue = 4L,
  publisher = "Nature Publishing Group",
  page = "261-263",
  type = "article-journal",
  issued = list(
    year = 2012,
    month = 3
  )
)

yaml() %>%
  yaml_reference(ref)

# from ?bibentry
bref <- c(
  bibentry(
    bibtype = "Manual",
    title = "boot: Bootstrap R (S-PLUS) Functions",
    author = c(
      person("Angelo", "Canty", role = "aut",
        comment = "S original"),
      person(c("Brian", "D."), "Ripley", role = c("aut", "trl", "cre"),
        comment = "R port, author of parallel support",
        email = "ripley@stats.ox.ac.uk")
    ),
    year = "2012",
    note = "R package version 1.3-4",
    url = "https://CRAN.R-project.org/package=boot",
    key = "boot-package"
```

```

    ),
    bibentry(
      bibtype = "Book",
      title = "Bootstrap Methods and Their Applications",
      author = as.person("Anthony C. Davison [aut], David V. Hinkley [aut]"),
      year = "1997",
      publisher = "Cambridge University Press",
      address = "Cambridge",
      isbn = "0-521-57391-2",
      url = "http://statwww.epfl.ch/davison/BMA/",
      key = "boot-book"
    )
  )

# requires pandoc-citeproc to be installed
yml() %>%
  yml_reference(.bibentry = bref)

yml() %>%
  yml_reference(.bibentry = citation("purrr"))

```

---

yml\_replace

*Replace, pluck, or discard top-level YAML fields*


---

## Description

`yml_replace()` replaces a named field with another value. As opposed to duplicating top-level fields with other functions, explicitly replacing them with `yml_replace()` will not raise a warning. `yml_discard()` removes values given either a character vector of names or a purrr-style lambda with a predicate (`~ predicate`); see the examples. `yml_pluck()` and `yml_chuck()` are wrappers around `purrr::pluck()` and `purrr::chuck()` that return `yml` objects.

## Usage

```
yml_replace(.yml, ...)
```

```
yml_discard(.yml, .rid)
```

```
yml_pluck(.yml, ...)
```

```
yml_chuck(.yml, ...)
```

## Arguments

<code>.yml</code>	a <code>yml</code> object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*()</code> function
<code>...</code>	additional named R objects, such as characters or lists, to transform into YAML
<code>.rid</code>	a character vector of fields to remove or a purrr-style lambda with a predicate ( <code>~ predicate</code> ) where fields that are TRUE will be discarded

**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

**Examples**

```
yml() %>%
  yaml_clean(TRUE) %>%
  yaml_replace(clean = FALSE) %>%
  yaml_discard("author")

yml() %>%
  yaml_output(
    pdf_document(),
    html_document()
  ) %>%
  yaml_discard(~ length(.x) > 1)
```

---

`yml_resource_files`      *Add external resource files to R Markdown document*

---

**Description**

The `resource_files` field specifies a character vectors of paths to external resources to include in the output, e.g. files that are necessary for rendering. These files are handled with `rmarkdown::find_external_resources()`

**Usage**

```
yml_resource_files(.yaml, resource_files)
```

**Arguments**

`.yaml`                    a yml object created by `yml()`, `as_yaml()`, or returned by a `yaml_*()` function  
`resource_files`      A path to a file, directory, or a wildcard pattern (such as "data/\*.csv")

**Value**

a yml object



**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

**Examples**

```
yml() %>%
  yml_resource_files(c("data/mydata.csv", "images/figure.png"))
```

---

yml\_rsconnect\_email    *Set YAML for Scheduled Emails in RStudio Connect*

---

**Description**

RStudio Connect allows you to schedule emails to send using R Markdown. It uses a special type of YAML using the top-level field `rmd_output_metadata` that tells RStudio Connect about the email output. Several `rsc_*` fields exist to specify different components of the email, which can be set in the YAML header or programmatically using `rmarkdown::output_metadata()`. See the [RStudio Connect documentation](#) for more. `yml_output_metadata()` allows you to add any type of content to the `rmd_output_metadata` field.

**Usage**

```
yml_rsconnect_email(
  .yaml,
  rsc_email_subject = yml_blank(),
  rsc_email_body_html = yml_blank(),
  rsc_email_body_text = yml_blank(),
  rsc_email_images = yml_blank(),
  rsc_output_files = yml_blank(),
  rsc_email_attachments = yml_blank(),
  rsc_email_suppress_scheduled = yml_blank(),
  rsc_email_suppress_report_attachment = yml_blank(),
  resource_files = yml_blank(),
  ...
)

yml_output_metadata(.yaml, ...)
```

**Arguments**

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> ( <code>)</code> function
<code>rsc_email_subject</code>	The subject of the email. A report without an <code>rsc_email_subject</code> entry uses its published document name.
<code>rsc_email_body_html</code> , <code>rsc_email_body_text</code>	The body of the email, either in plain text or HTML. A report with neither entry uses an automatically generated, plain-text body with a link to the report's URL.
<code>rsc_email_images</code>	Images to embed in the email. The embedded image must have a Content ID that is used in the body of the HTML and when providing the image to <code>rsc_email_images</code> , and the image itself must be base64-encoded, e.g. with the <code>base64enc</code> package.
<code>rsc_output_files</code>	A vector of file names that should be available after the report has rendered. If you list a file that does not exist after rendering your report, Connect will log a message but continue trying to processing the other files listed. If the output files are not generated during the rendering of your report, then you will also need to list them in <code>resource_files</code> when you upload your report to Connect.
<code>rsc_email_attachments</code>	A vector of file names that should be attached to the email.
<code>rsc_email_suppress_scheduled</code>	Logical. Should the email schedule be suppressed? Default is FALSE.
<code>rsc_email_suppress_report_attachment</code>	Logical. Should the rendered document be included as an attachment? Default is TRUE.
<code>resource_files</code>	A file or files to host on RStudio Connect that is <i>not</i> generated by your report, e.g. an existing file.
<code>...</code>	additional named R objects, such as characters or lists, to transform into YAML

**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

**Examples**

```
yml() %>%
  yml_rsconnect_email(
```

```

    rsc_email_subject = "Quarterly report",
    rsc_output_files = "data.csv",
    rsc_email_attachments = c("attachment_1.csv", "attachment_2.csv")
  )

```

---

yml\_rticles\_opts      *Set YAML related to rticles output formats*

---

### Description

The rticles package include numerous output formats specific to academic journals. All of these can take YAML similar to pdf\_document(). Additionally, two templates include custom YAML, rticles::sage\_article() and rticles::sim\_article(). See the help pages for these functions for more details and the sources of the LaTeX templates used for each.

### Usage

```

yml_rticles_opts(
  .yml,
  title = yml_blank(),
  runninghead = yml_blank(),
  author = yml_blank(),
  authormark = yml_blank(),
  address = yml_blank(),
  corrauth = yml_blank(),
  corres = yml_blank(),
  email = yml_blank(),
  abstract = yml_blank(),
  received = yml_blank(),
  revised = yml_blank(),
  accepted = yml_blank(),
  keywords = yml_blank(),
  bibliography = yml_blank(),
  longtable = yml_blank(),
  classoption = yml_blank(),
  header_includes = yml_blank(),
  include_after = yml_blank(),
  ...
)

rticles_author(name = yml_blank(), num = yml_blank())

rticles_address(name = yml_blank(), org = yml_blank())

rticles_corr_author(
  name = yml_blank(),
  author = yml_blank(),

```

```

    address = yml_blank()
  )

```

### Arguments

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> function
<code>title</code>	Title of the manuscript
<code>runninghead</code>	A character vector, a short author list for the header ( <code>sage_article</code> )
<code>author</code>	A list of authors, containing name and num fields ( <code>sage_article</code> , <code>sim_article</code> ). Use <code>rticles_author()</code> or a list to specify.
<code>authormark</code>	A character vector, the short author list for the header ( <code>sim_article</code> )
<code>address</code>	list containing num and org for defining author affiliations ( <code>sage_article</code> , <code>sim_article</code> ). Use <code>rticles_address()</code> or a list to specify.
<code>corrauth</code>	corresponding author name and address ( <code>sage_article</code> ). Use <code>rticles_corr_author()</code> or a list to specify.
<code>corres</code>	author and address for correspondence ( <code>sim_article</code> ). Use <code>rticles_corr_author()</code> or a list to specify.
<code>email</code>	The email of the correspondence author ( <code>sage_article</code> )
<code>abstract</code>	The abstract, limited to 200 words ( <code>sage_article</code> ), 250 words ( <code>sim_article</code> )
<code>received, revised, accepted</code>	The dates of submission, revision, and acceptance of the manuscript ( <code>sim_article</code> )
<code>keywords</code>	The keywords for the article ( <code>sage_article</code> ), up to 6 keywords ( <code>sim_article</code> )
<code>bibliography</code>	BibTeX .bib file name ( <code>sage_article</code> , <code>sim_article</code> )
<code>longtable</code>	Logical. Include the longtable package? Used by default from pandoc to convert markdown to LaTeX code ( <code>sim_article</code> )
<code>classoption</code>	a character vector of classoption options for the sagej class ( <code>sage_article</code> )
<code>header_includes</code>	additional LaTeX code to include in the header, before the <code>\begin\{document\}</code> statement ( <code>sage_article</code> , <code>sim_article</code> ). Note that the actual YAML field is <code>header-includes</code>
<code>include_after</code>	additional LaTeX code to include before the <code>\end\{document\}</code> statement ( <code>sage_article</code> , <code>sim_article</code> ). Note that the actual YAML field is <code>include-after</code> .
<code>...</code>	additional named R objects, such as characters or lists, to transform into YAML
<code>name</code>	The author's name
<code>num</code>	The author's number or address number
<code>org</code>	The author's organization

### Value

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

**Examples**

```
yml() %>%
  yml_rarticles_opts(received = "09-12-2014")
```

---

yml\_runtime

*Activate Shiny in R Markdown*


---

**Description**

The runtime field lets you use Shiny in your R Markdown document, making it interactive. See the [R Markdown book](#) for examples.

**Usage**

```
yml_runtime(.yaml, runtime = c("static", "shiny", "shiny_prerendered"))
```

**Arguments**

<code>.yaml</code>	a yml object created by <code>yml()</code> , <code>as_yaml()</code> , or returned by a <code>yml_*()</code> function
<code>runtime</code>	The runtime target for rendering. <code>static</code> , the default, renders static documents; <code>shiny</code> allows you to include use Shiny in your document. <code>shiny_prerendered</code> is a subset of the shiny runtime that allows pre-rendering of app components (see the <a href="#">R Markdown site</a> for more)

**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

Other R Markdown: [yaml\\_clean\(\)](#), [yaml\\_params\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_vignette\(\)](#)

Other shiny: [yaml\\_params\(\)](#)

## Examples

```
yml() %>%
  yml_runtime("shiny")
```

---

yml\_site\_opts

*Add site options for \_site.yml and navbars for R Markdown websites*


---

## Description

R Markdown has a simple website builder baked in (see the [R Markdown book](#) for a detailed description). An R Markdown website must have at least have an `index.Rmd` file and a `_site.yml` file (which can be empty). Including YAML in `_site.yml` will apply it to all R Markdown files for the website, e.g. setting the output format here will tell R Markdown to use that format across the website. R Markdown websites also support navbars, which you can specify with YAML (see `yml_navbar()`, as well as `?rmarkdown::render_site` and `?rmarkdown::html_document`). Pass `navbar_page()` to the left or right field to set up page tabs and use `navbar_separator()` to include a separators. In addition to writing YAML with `yml_*`() functions, `use_site_yml()` will take the a yml object and write it to a `_site.yml` file for you.

## Usage

```
yml_site_opts(
  .yml,
  name = yml_blank(),
  favicon = yml_blank(),
  output_dir = yml_blank(),
  include = yml_blank(),
  exclude = yml_blank(),
  new_session = yml_blank(),
  ...
)
```

```
yml_navbar(
  .yml,
  title = yml_blank(),
  type = yml_blank(),
  left = yml_blank(),
  right = yml_blank(),
  ...
)
```

```
navbar_page(
  text = yml_blank(),
  href = yml_blank(),
  icon = yml_blank(),
```

```

    menu = yml_blank(),
    ...
)

navbar_separator()

```

## Arguments

.yml	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> () function
name	The name of the website
favicon	Path to a file to use as the favicon
output_dir	Directory to copy site content into (" <code>_site</code> " is the default if none is specified)
include, exclude	Files to include or exclude from the copied into <code>output_dir</code> . You can use <code>*</code> to indicate a wildcard selection, e.g. <code>"*.csv"</code> .
new_session	Logical. Should each website file be rendered in a new R session?
...	additional named R objects, such as characters or lists, to transform into YAML
title	The title of the website
type	The color scheme for the navigation bar: either "default" or "inverse".
left, right	the side of the navbar a <code>navbar_page()</code> should go (see example)
text	The link text
href	The link URL
icon	An icon to include
menu	drop-down menus specified by including another <code>navbar_page()</code>

## Value

a yml object

## See Also

[use\\_site\\_yml\(\)](#) [use\\_navbar\\_yml\(\)](#) [use\\_index\\_rmd\(\)](#)

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_toc\(\)](#), [yaml\\_vignette\(\)](#)

Other R Markdown: [yaml\\_clean\(\)](#), [yaml\\_params\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_vignette\(\)](#)

Other websites: [yaml\\_distill\\_opts\(\)](#), [yaml\\_pkgdown\(\)](#)

**Examples**

```

yml_empty() %>%
  yml_site_opts(
    name = "my-website",
    output_dir = "_site",
    include = "demo.R",
    exclude = c("docs.txt", "*.csv")
  ) %>%
  yml_navbar(
    title = "My Website",
    left = list(
      navbar_page("Home", href = "index.html"),
      navbar_page(navbar_separator(), href = "about.html")
    )
  ) %>%
  yml_output(html_document(toc = TRUE, highlight = "textmate"))

```

---

yml\_toc

*Specify Table of Contents options*


---

**Description**

It's generally better to specify Table of Contents in the output function you are using so you have a clearer idea of your options (e.g. `html_document(toc = TRUE, toc_float = TRUE)`). However, you can also generally specify at the top level of YAML.

**Usage**

```

yml_toc(
  .yml,
  toc = yml_blank(),
  toc_depth = yml_blank(),
  toc_title = yml_blank(),
  ...
)

```

**Arguments**

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*()</code> function
<code>toc</code>	Logical. Use a Table of Contents?
<code>toc_depth</code>	An integer. The depth of headers to use in the TOC. Note that the actual YAML field is <code>toc-depth</code> .
<code>toc_title</code>	The title of the TOC. Note that the actual YAML field is <code>toc-title</code> .
<code>...</code>	additional named R objects, such as characters or lists, to transform into YAML



**Value**

a yml object

**See Also**

Other yml: [asis\\_yaml\\_output\(\)](#), [bib2yaml\(\)](#), [draw\\_yaml\\_tree\(\)](#), [has\\_field\(\)](#), [read\\_json\(\)](#), [use\\_yaml\\_defaults\(\)](#), [use\\_yaml\\_file\(\)](#), [use\\_yaml\(\)](#), [yaml\\_author\(\)](#), [yaml\\_blogdown\\_opts\(\)](#), [yaml\\_bookdown\\_opts\(\)](#), [yaml\\_citations\(\)](#), [yaml\\_clean\(\)](#), [yaml\\_distill\\_opts\(\)](#), [yaml\\_latex\\_opts\(\)](#), [yaml\\_output\(\)](#), [yaml\\_pagedown\\_opts\(\)](#), [yaml\\_params\(\)](#), [yaml\\_pkgdown\(\)](#), [yaml\\_reference\(\)](#), [yaml\\_replace\(\)](#), [yaml\\_resource\\_files\(\)](#), [yaml\\_rsconnect\\_email\(\)](#), [yaml\\_rarticles\\_opts\(\)](#), [yaml\\_runtime\(\)](#), [yaml\\_site\\_opts\(\)](#), [yaml\\_vignette\(\)](#)

**Examples**

```
yml() %>%
  yml_toc(toc = TRUE, toc_depth = 1, toc_title = "Article Outline")
```

---

yml_verbatim	<i>Write YAML field or content verbatim</i>
--------------	---

---

**Description**

`yml_verbatim()` is a helper function to write YAML precisely as given to the `yml_*()` function rather than the defaults in `ymlthis` and `yaml`. `ymlthis` uses the `yaml` package to check for valid syntax; `yaml` and `ymlthis` together make decisions about how to write syntax, which can often be done in numerous valid ways. See [`yaml::as.yaml\(\)`](#) for more details.

**Usage**

```
yml_verbatim(x)
```

**Arguments**

`x` a character vector

**Value**

an object of class `verbatim`

**Examples**

```
# "yes" and "no" serve as alternatives to `true` and `false`. This writes
# "yes" literally.
yml_verbatim("yes")
```

---

yml\_vignette

*Set up a package vignette*


---

## Description

To use an R Markdown file as a vignette, you need to specify an output format appropriate for inclusion in a package (for example, the lightweight `html_vignette()` output function included in `rmarkdown`) and to specify the `vignette` field, which specifies the title, engine, and encoding type of the vignette. See also [`usethis::use\_vignette\(\)`](#) for setting up a package vignette.

## Usage

```
yml_vignette(.yml, title, engine = "knitr::rmarkdown", encoding = "UTF-8")
```

## Arguments

<code>.yml</code>	a yml object created by <code>yml()</code> , <code>as_yml()</code> , or returned by a <code>yml_*</code> () function
<code>title</code>	The title of the vignette
<code>engine</code>	The rendering engine for the vignette ("knitr::rmarkdown" by default)
<code>encoding</code>	The character encoding for the document ("UTF-8" by default).

## Value

a yml object

## See Also

Other yml: [`asis\_yaml\_output\(\)`](#), [`bib2yaml\(\)`](#), [`draw\_yaml\_tree\(\)`](#), [`has\_field\(\)`](#), [`read\_json\(\)`](#), [`use\_yaml\_defaults\(\)`](#), [`use\_yaml\_file\(\)`](#), [`use\_yaml\(\)`](#), [`yml\_author\(\)`](#), [`yml\_blogdown\_opts\(\)`](#), [`yml\_bookdown\_opts\(\)`](#), [`yml\_citations\(\)`](#), [`yml\_clean\(\)`](#), [`yml\_distill\_opts\(\)`](#), [`yml\_latex\_opts\(\)`](#), [`yml\_output\(\)`](#), [`yml\_pagedown\_opts\(\)`](#), [`yml\_params\(\)`](#), [`yml\_pkgdown\(\)`](#), [`yml\_reference\(\)`](#), [`yml\_replace\(\)`](#), [`yml\_resource\_files\(\)`](#), [`yml\_rsconnect\_email\(\)`](#), [`yml\_rarticles\_opts\(\)`](#), [`yml\_runtime\(\)`](#), [`yml\_site\_opts\(\)`](#), [`yml\_toc\(\)`](#)

Other R Markdown: [`yml\_clean\(\)`](#), [`yml\_params\(\)`](#), [`yml\_runtime\(\)`](#), [`yml\_site\_opts\(\)`](#)

## Examples

```
yml() %>%
  yml_output(html_vignette()) %>%
  yml_vignette("An introduction to R Markdown")
```

# Index

- \* **R Markdown**
  - yml\_clean, 32
  - yml\_params, 44
  - yml\_runtime, 61
  - yml\_site\_opts, 62
  - yml\_vignette, 66
- \* **bookdown**
  - gitbook\_config, 7
  - yml\_bookdown\_opts, 28
- \* **citations**
  - bib2yaml, 4
  - yml\_citations, 31
  - yml\_reference, 53
- \* **distill**
  - yml\_distill\_opts, 34
- \* **pagedown**
  - pagedown\_business\_card\_template, 11
  - yml\_pagedown\_opts, 43
- \* **pkgdown**
  - pkgdown\_template, 15
  - yml\_pkgdown, 49
- \* **shiny**
  - yml\_params, 44
  - yml\_runtime, 61
- \* **websites**
  - yml\_distill\_opts, 34
  - yml\_pkgdown, 49
  - yml\_site\_opts, 62
- \* **yml**
  - asis\_yaml\_output, 3
  - bib2yaml, 4
  - draw\_yaml\_tree, 6
  - has\_field, 9
  - read\_json, 16
  - use\_yaml, 17
  - use\_yaml\_defaults, 18
  - use\_yaml\_file, 19
  - yml\_author, 23
  - yml\_blogdown\_opts, 26
  - yml\_bookdown\_opts, 28
  - yml\_citations, 31
  - yml\_clean, 32
  - yml\_distill\_opts, 34
  - yml\_latex\_opts, 38
  - yml\_output, 42
  - yml\_pagedown\_opts, 43
  - yml\_params, 44
  - yml\_pkgdown, 49
  - yml\_reference, 53
  - yml\_replace, 55
  - yml\_resource\_files, 56
  - yml\_rsconnect\_email, 57
  - yml\_rarticles\_opts, 59
  - yml\_runtime, 61
  - yml\_site\_opts, 62
  - yml\_toc, 64
  - yml\_vignette, 66
- as\_yaml, 3
- as\_yaml(), 22
- asis\_yaml\_output, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- base::strftime(), 47
- bib2yaml, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- bib2yaml(), 53
- bibentry(), 53
- blogdown\_archetypes (blogdown\_template), 5
- blogdown\_template, 5
- blogdown\_template(), 26
- bookdown::gitbook(), 7, 8
- citation(), 53
- code\_chunk, 5

- code\_chunk(), 18
- distill\_collection (yml\_distill\_opts), 34
- distill\_listing (yml\_distill\_opts), 34
- distill\_resources (yml\_distill\_opts), 34
- draw\_yml\_tree, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- format.Date(), 24
- get\_rmd\_defaults (use\_yml\_defaults), 18
- get\_yml\_defaults (use\_yml\_defaults), 18
- get\_yml\_defaults(), 19
- gitbook\_config, 7, 30
- has\_field, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- includes2, 10
- is\_yml, 10
- is\_yml\_blank (yml\_blank), 25
- knitr::write\_bib(), 4, 53
- last\_yml, 11
- last\_yml(), 17
- navbar\_page (yml\_site\_opts), 62
- navbar\_separator (yml\_site\_opts), 62
- pagedown\_business\_card\_template, 11, 44
- pagedown\_business\_card\_template(), 43
- pagedown\_person (pagedown\_business\_card\_template), 11
- pandoc\_highlight\_styles (pandoc\_template\_types), 14
- pandoc\_template\_types, 14
- pkgdown\_article (yml\_pkgdown), 49
- pkgdown\_ref (yml\_pkgdown), 49
- pkgdown\_template, 15, 52
- pkgdown\_tutorial (yml\_pkgdown), 49
- purrr::chuck(), 55
- purrr::pluck(), 55
- read\_json, 3, 4, 6, 9, 16, 18, 19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- read\_toml (read\_json), 16
- reference (yml\_reference), 53
- rticles\_address (yml\_rticles\_opts), 59
- rticles\_author (yml\_rticles\_opts), 59
- rticles\_corr\_author (yml\_rticles\_opts), 59
- setup\_chunk (code\_chunk), 5
- setup\_chunk(), 17, 18
- shiny::animationOptions(), 46
- shiny::validateCssUnit(), 46
- shiny\_checkbox (yml\_params), 44
- shiny\_date (yml\_params), 44
- shiny\_file (yml\_params), 44
- shiny\_numeric (yml\_params), 44
- shiny\_params (yml\_params), 44
- shiny\_password (yml\_params), 44
- shiny\_radio (yml\_params), 44
- shiny\_select (yml\_params), 44
- shiny\_slider (yml\_params), 44
- shiny\_text (yml\_params), 44
- use\_bookdown\_yml (use\_yml\_file), 19
- use\_bookdown\_yml(), 29
- use\_index\_rmd (use\_yml), 17
- use\_index\_rmd(), 29, 63
- use\_navbar\_yml (use\_yml\_file), 19
- use\_navbar\_yml(), 63
- use\_output\_yml (use\_yml\_file), 19
- use\_output\_yml(), 29
- use\_pandoc\_highlight\_style (pandoc\_template\_types), 14
- use\_pandoc\_template (pandoc\_template\_types), 14
- use\_pkgdown\_yml (use\_yml\_file), 19
- use\_pkgdown\_yml(), 15, 49, 52
- use\_rmarkdown (use\_yml), 17
- use\_rmarkdown(), 5, 11, 13, 19, 37
- use\_rmd\_defaults (use\_yml\_defaults), 18
- use\_rmd\_defaults(), 17, 18
- use\_site\_yml (use\_yml\_file), 19
- use\_site\_yml(), 34, 37, 63
- use\_yml, 3, 4, 6, 9, 17, 17, 19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- use\_yml\_defaults, 3, 4, 6, 9, 17, 18, 18, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- use\_yml\_defaults(), 17, 21

- use\_yml\_file, 3, 4, 6, 9, 17–19, 19, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- usethis::edit\_r\_profile(), 19
- usethis::use\_vignette(), 66
- write\_as\_json(read\_json), 16
- write\_as\_toml(read\_json), 16
- yaml::as.yaml(), 65
- yaml::yaml.load(), 3, 22, 38, 41
- yml, 21
- yml(), 19
- yml\_abstract(yml\_author), 23
- yml\_author, 3, 4, 6, 9, 17–19, 21, 23, 23, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_author(), 34
- yml\_blank, 25
- yml\_blogdown\_opts, 3, 4, 6, 9, 17–19, 21, 24, 26, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_bookdown\_opts, 3, 4, 6, 9, 17–19, 21, 24, 28, 28, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_bookdown\_site(yml\_bookdown\_opts), 28
- yml\_category(yml\_author), 23
- yml\_chuck(yml\_replace), 55
- yml\_citations, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 31, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_citations(), 53
- yml\_clean, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 32, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_code, 33
- yml\_date(yml\_author), 23
- yml\_description(yml\_author), 23
- yml\_discard(yml\_replace), 55
- yml\_discard(), 25, 49
- yml\_distill\_author, 23
- yml\_distill\_author(yml\_distill\_opts), 34
- yml\_distill\_opts, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 34, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_empty(yml), 21
- yml\_handlers, 38
- yml\_keywords(yml\_author), 23
- yml\_lang(yml\_author), 23
- yml\_latex\_opts, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 38, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_load, 41
- yml\_navbar(yml\_site\_opts), 62
- yml\_navbar(), 34, 49, 52, 62
- yml\_output, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_output(), 24
- yml\_output\_metadata(yml\_rsconnect\_email), 57
- yml\_pagedown\_opts, 3, 4, 6, 9, 13, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 43, 48, 52, 54, 56–58, 61, 63, 65, 66
- yml\_params, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 44, 52, 54, 56–58, 61, 63, 65, 66
- yml\_params\_code(yml\_code), 33
- yml\_params\_code(), 48
- yml\_pkgdown, 3, 4, 6, 9, 15, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 49, 54, 56–58, 61, 63, 65, 66
- yml\_pkgdown\_articles(yml\_pkgdown), 49
- yml\_pkgdown\_development(yml\_pkgdown), 49
- yml\_pkgdown\_docsearch(yml\_pkgdown), 49
- yml\_pkgdown\_figures(yml\_pkgdown), 49
- yml\_pkgdown\_news(yml\_pkgdown), 49
- yml\_pkgdown\_opts(yml\_pkgdown), 49
- yml\_pkgdown\_reference(yml\_pkgdown), 49
- yml\_pkgdown\_template(yml\_pkgdown), 49
- yml\_pkgdown\_tutorial(yml\_pkgdown), 49
- yml\_pluck(yml\_replace), 55
- yml\_reference, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 53, 56–58, 61, 63, 65, 66
- yml\_replace, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 55, 57, 58, 61, 63, 65, 66
- yml\_replace(), 25, 49
- yml\_resource\_files, 3, 4, 6, 9, 17–19, 21, 24, 28, 29, 32, 33, 37, 41, 42, 44, 48, 52, 54, 56, 56, 58, 61, 63, 65, 66
- yml\_rsconnect\_email, 3, 4, 6, 9, 17–19, 21, 24, 28, 30, 32, 33, 37, 41, 42, 44, 48,

*52, 54, 56, 57, 57, 61, 63, 65, 66*  
yml\_rarticles\_opts, *3, 4, 6, 9, 17–19, 21, 24,*  
*28, 30, 32, 33, 37, 41, 42, 44, 48, 52,*  
*54, 56–58, 59, 61, 63, 65, 66*  
yml\_runtime, *3, 4, 6, 9, 17–19, 21, 24, 28, 30,*  
*32, 33, 37, 41, 42, 44, 48, 52, 54,*  
*56–58, 61, 61, 63, 65, 66*  
yml\_site\_opts, *3, 4, 6, 9, 17–19, 21, 24, 28,*  
*30, 32, 33, 37, 41, 42, 44, 48, 52, 54,*  
*56–58, 61, 62, 65, 66*  
yml\_site\_opts(), *34*  
yml\_subject (yml\_author), *23*  
yml\_subtitle (yml\_author), *23*  
yml\_title (yml\_author), *23*  
yml\_toc, *3, 4, 6, 9, 17–19, 21, 24, 28, 30, 32,*  
*33, 37, 41, 42, 44, 48, 52, 54, 56–58,*  
*61, 63, 64, 66*  
yml\_toplevel (yml\_author), *23*  
yml\_verbatim, *65*  
yml\_verbatim(), *33*  
yml\_vignette, *3, 4, 6, 9, 17–19, 21, 24, 28,*  
*30, 32, 33, 37, 41, 42, 44, 48, 52, 54,*  
*56–58, 61, 63, 65, 66*