Package ‘rmarkdown’

June 11, 2018

Type Package
Title Dynamic Documents for R
Version 1.10
Maintainer Yihui Xie <xie@yihui.name>
Description Convert R Markdown documents into a variety of formats.
Depends R (>= 3.0)
Imports tools, utils, knitr (>= 1.18), yaml (>= 2.1.5), htmltools (>= 0.3.5), evaluate (>= 0.8), base64enc, jsonlite, rprojroot, mime, tinytex (>= 0.3), methods, stringr (>= 1.2.0)
Suggests shiny (>= 0.11), tufte, testthat, digest, dygraphs, tibble
SystemRequirements pandoc (>= 1.12.3) - http://pandoc.org
URL https://rmarkdown.rstudio.com,
https://github.com/rstudio/rmarkdown
BugReports https://github.com/rstudio/rmarkdown/issues
License GPL-3
RoxygenNote 6.0.1
NeedsCompilation no
Author JJ Allaire [aut],
Yihui Xie [aut, cre] (https://orcid.org/0000-0003-0645-5666),
Jonathan McPherson [aut],
Javier Luraschi [aut],
Kevin Ushey [aut],
Aron Atkins [aut],
Hadley Wickham [aut],
Joe Cheng [aut],
Winston Chang [aut],
Jeff Allen [ctb],
Roy Storey [ctb],
Rob Hyndman [ctb],
Ruben Arslan [ctb],
Frederik Aust [ctb] (https://orcid.org/0000-0003-4900-788X),
R topics documented:

RSudio, Inc. [cph],
jQuery Foundation [cph] (jQuery library),
jQuery contributors [ctb, cph] (jQuery library; authors listed in
inst/rmd/h/jquery-AUTHORS.txt),
jQuery UI contributors [ctb, cph] (jQuery UI library; authors listed in
inst/rmd/h/jqueryui-AUTHORS.txt),
Mark Otto [ctb] (Bootstrap library),
Jacob Thornton [ctb] (Bootstrap library),
Bootstrap contributors [ctb] (Bootstrap library),
Twitter, Inc [cph] (Bootstrap library),
Alexander Farkas [ctb, cph] (html5shiv library),
Scott Jehl [ctb, cph] (Respond.js library),
Ivan Sagalaev [ctb, cph] (highlight.js library),
Greg Franko [ctb, cph] (tocify library),
John MacFarlane [ctb, cph] (Pandoc templates),
Google, Inc. [ctb, cph] (ioslides library),
Dave Raggett [ctb] (slidy library),
W3C [cph] (slidy library),
Dave Gandy [ctb, cph] (Font-Awesome),
Ben Sperry [ctb] (Ionicons),
Drifty [cph] (Ionicons),
Aidan Lister [ctb, cph] (jQuery StickyTabs)

Repository  CRAN
Date/Publication  2018-06-11 06:12:56 UTC

R topics documented:

rmarkdown-package .................................................. 3
all_output_formats ............................................... 4
beamer_presentation ............................................. 5
compile_notebook ................................................. 7
default_output_format ........................................... 8
draft ............................................................... 9
find_external_resources ........................................... 10
github_document .................................................. 11
html-dependencies ................................................ 12
html_document ...................................................... 13
html_document_base .............................................. 17
html_fragment ..................................................... 18
html_notebook ..................................................... 20
html_notebook_metadata ......................................... 21
html_notebook_output ........................................... 22
html_vignette .................................................... 23
includes ........................................................... 24
ioslides_presentation ........................................... 25
knitr_options ...................................................... 25
knitr_options_html ................................................ 31
**R Markdown Document Conversion**

**Description**

Convert R Markdown documents into a variety of formats including HTML, MS Word, PDF, and Beamer.
Details

The `rmarkdown` package includes high level functions for converting to a variety of formats. For example:

```r
render("input.Rmd", html_document())
render("input.Rmd", pdf_document())
```

You can also specify a plain markdown file in which case knitting will be bypassed:

```r
render("input.md", html_document())
```

Additional options can be specified along with the output format:

```r
render("input.Rmd", html_document(toc = TRUE))
render("input.Rmd", pdf_document(latex_engine = "lualatex"))
render("input.Rmd", beamer_presentation(incremental = TRUE))
```

You can also include arbitrary pandoc command line arguments along with the other options:

```r
render("input.Rmd", pdf_document(toc = TRUE, pandoc_args = "--listings"))
```

See Also

`render`, `html_document`, `pdf_document`, `word_document`, `beamer_presentation`
beamer_presentation

Convert to a Beamer presentation

Description

Format for converting from R Markdown to a Beamer presentation.

Usage

beamer_presentation(toc = FALSE, slide_level = NULL, incremental = FALSE,
fig_width = 10, fig_height = 7, fig_crop = TRUE, fig_caption = TRUE,
dev = "pdf", df_print = "default", theme = "default",
colortheme = "default", fonttheme = "default", highlight = "default",
template = "default", keep_tex = FALSE, latex_engine = "pdflatex",
citation_package = c("none", "natbib", "biblatex"), self_contained = TRUE,
includes = NULL, md_extensions = NULL, pandoc_args = NULL)

Arguments

toc TRUE to include a table of contents in the output (only level 1 headers will be
included in the table of contents).

slide_level The heading level which defines individual slides. By default this is the highest
header level in the hierarchy that is followed immediately by content, and not
another header, somewhere in the document. This default can be overridden by
specifying an explicit slide_level.

incremental TRUE to render slide bullets incrementally. Note that if you want to reverse the
default incremental behavior for an individual bullet you can precede it with >.
For example: > - Bullet Text

fig_width Default width (in inches) for figures

fig_height Default width (in inches) for figures

fig_crop TRUE to automatically apply the pdfcrop utility (if available) to pdf figures

fig_caption TRUE to render figures with captions

dev Graphics device to use for figure output (defaults to pdf)

df_print Method to be used for printing data frames. Valid values include "default",
"kable", "tibble", and "paged". The "default" method uses print.data.frame. The
"kable" method uses the knitr::kable function. The "tibble" method uses the
tibble package to print a summary of the data frame. The "paged" method
creates a paginated HTML table (note that this method is only valid for formats
that produce HTML). In addition to the named methods you can also pass an ar-
bitrary function to be used for printing data frames. You can disable the df_print
behavior entirely by setting the option rmarkdown.df_print to FALSE.

theme Beamer theme (e.g. "AnnArbor").

colortheme Beamer color theme (e.g. "dolphin").

fonttheme Beamer font theme (e.g. "structurebold").
**beamer_presentation**

- **highlight**: Syntax highlighting style. Supported styles include "default", "tango", "pygments", "kate", "monochrome", "espresso", "zenburn", and "haddock". Pass `NULL` to prevent syntax highlighting.

- **template**: Pandoc template to use for rendering. Pass "default" to use the rmarkdown package default template; pass `NULL` to use pandoc’s built-in template; pass a path to use a custom template that you’ve created. See the documentation on pandoc online documentation for details on creating custom templates.

- **keep_tex**: Keep the intermediate tex file used in the conversion to PDF.

- **latex_engine**: LaTeX engine for producing PDF output. Options are "pdflatex", "lualatex", and "xelatex".

- **citation_package**: The LaTeX package to process citations, natbib or biblatex. Use `none` if neither package is to be used.

- **self_contained**: Whether to generate a full LaTeX document (`TRUE`) or just the body of a LaTeX document (`FALSE`). Note the LaTeX document is an intermediate file unless `keep_tex = TRUE`.

- **includes**: Named list of additional content to include within the document (typically created using the `includes` function).

- **md_extensions**: Markdown extensions to be added or removed from the default definition or R Markdown. See the `rmarkdown_format` for additional details.

- **pandoc_args**: Additional command line options to pass to pandoc.

**Details**

See the online documentation for additional details on using the `beamer_presentation` format.

Creating Beamer output from R Markdown requires that LaTeX be installed.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

**Value**

R Markdown output format to pass to `render`.

**Examples**

```r
## Not run:
library(rmarkdown)

# simple invocation
derender("pres.Rmd", beamer_presentation())

# specify an option for incremental rendering
derender("pres.Rmd", beamer_presentation(incremental = TRUE))
```
## Description

R Markdown can also compile R scripts to a notebook which includes commentary, source code, and script output. Notebooks can be compiled to any output format including HTML, PDF, and MS Word.

### Overview

To compile a notebook from an R script you simply pass the script to `render`. For example:

```r
rmarkdown::render('analysis.R')
rmarkdown::render('analysis.R', 'pdf_document')
```

The first call to `render` creates an HTML document, whereas the second creates a PDF document.

By default the name of the script, username, and current date and time are included in the header of the generated notebook. You can override this default behavior by including explicit metadata in a specially formatted R comment:

```r
#' ---
# title: "Crop Analysis Q3 2013"
# author: "John Smith"
# date: "May 3rd, 2014"
#' ---
```

### Including Markdown

Note that the R comment used above to add a title, author, and date includes a single-quote as a special prefix character. This is a `roxygen2` style comment, and it’s actually possible to include many such comments in an R script, all of which will be converted to markdown content within the generated notebook. For example:

```r
#' A script comment that includes **markdown** formatting.
```

Rather than displaying as an R comment in the compiled notebook any `roxygen2` style comment will be treated as markdown and rendered accordingly.
knitr Spin

Including markdown within R comments is possible because `render` calls the `knitr::spin` function to convert the R script to an Rmd file. The `spin` function also enables you to add knitr chunk options with another special comment prefix (`#+`).

Here’s an example of a script that uses the various features of `spin`:


For more details on `knitr::spin` see the following documentation:

http://yihui.name/knitr/demo/stitch/

default_output_format

Determine the default output format for an R Markdown document

Description

Read the YAML metadata (and any common _output.yml file) for the document and return the output format that will be generated by a call to `render`.

Usage

default_output_format(input, encoding = getOption("encoding"))

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Input file (Rmd or plain markdown)</td>
</tr>
<tr>
<td>encoding</td>
<td>The encoding of the input file; see <code>file</code></td>
</tr>
</tbody>
</table>

Details

This function is useful for front-end tools that require additional knowledge of the output to be produced by `render` (e.g. to customize the preview experience).

Value

A named list with a `name` value containing the format name and an `options` value that is a list containing all the options for the format and their values. An option’s default value will be returned if the option isn’t set explicitly in the document.
Create a new document based on a template

Description
Create (and optionally edit) a draft of an R Markdown document based on a template.

Usage
draft(file, template, package = NULL, create_dir = "default", edit = TRUE)

Arguments
- file: File name for the draft
- template: Template to use as the basis for the draft. This is either the full path to a template
directory or the name of a template directory within the rmarkdown/templates directory of a package.
- package: (Optional) Name of package where the template is located.
- create_dir: TRUE to create a new directory for the document (the "default" setting leaves this
behavior up to the creator of the template).
- edit: TRUE to edit the template immediately

Details
The `draft` function creates new R Markdown documents based on templates that are either located
on the filesystem or within an R package. The template and its supporting files will be copied to the
location specified by `file`.

Value
The file name of the new document (invisibly).

Note
An R Markdown template consists of a directory that contains a description of the template, a
skeleton Rmd file used as the basis for new documents, and optionally additional supporting files
that are provided along with the skeleton (e.g. a logo graphic).

If the template directory is contained within a package then it should be located at inst/rmarkdown/templates.
For example, a package named `pubtools` that wanted to provide a template named `quarterly_report`
would need to provide the following files within the pubtools/inst/rmarkdown/templates directory:

`quarterly_report/template.yaml`
`quarterly_report/skeleton/skeleton.Rmd`
The `template.yaml` file should include a `name` field. If you want to ensure that a new directory is always created for a given template, then you can add the `create_dir` field to the `template.yaml` file. For example:

```
create_dir: true
```

The `skeleton/skeleton.Rmd` file should include the initial contents you want for files created from this template. Additional files can be added to the skeleton directory, for example:

```
skeleton/logo.png
```

These files will automatically be copied to the directory containing the new R Markdown draft.

### Examples

```r
## Not run:
rmrmarkdown::draft("Q4Report.Rmd",
    template="/opt/rmd/templates/quarterly_report")

rmrmarkdown::draft("Q4Report.Rmd",
    template="quarterly_report", package="pubtools")

## End(Not run)
```

---

**find_external_resources**

*Find External Resource References*

---

**Description**

Given an R Markdown document or HTML file, attempt to determine the set of additional files needed in order to render and display the document.

**Usage**

```
find_external_resources(input_file, encoding = getOption("encoding"))
```

**Arguments**

- `input_file`: path to the R Markdown document or HTML file to process
- `encoding`: the encoding of the document

**Details**

This routine applies heuristics in order to scan a document for possible resource references.

In R Markdown documents, it looks for references to files implicitly referenced in Markdown (e.g. `![](img.png)`), in the document's YAML header, in raw HTML chunks, and as quoted strings in R code chunks (e.g. `read.csv("data.csv")`).
Resources specified explicitly in the YAML header for R Markdown documents are also returned. To specify resources in YAML, use the `resource_files` key:

```yaml
---
title: My Document
author: My Name
resource_files:
- data/mydata.csv
- images/figure.png
---
```

Each item in the `resource_files` list can refer to:

1. A single file, such as `images/figure.png`, or
2. A directory, such as `resources/data`, in which case all of the directory’s content will be recursively included, or
3. A wildcard pattern, such as `data/*.csv`, in which case all of the files matching the pattern will be included. No recursion is done in this case.

In HTML files (and raw HTML chunks in R Markdown documents), this routine searches for resources specified in common tag attributes, such as `<img src="...">`, `<link href="...">`, etc. In all cases, only resources that exist on disk and are contained in the document’s directory (or a child thereof) are returned.

**Value**

A data frame with the following columns:

- **path** The relative path from the document to the resource
- **explicit** Whether the resource was specified explicitly (TRUE) or discovered implicitly (FALSE)
- **web** Whether the resource is needed to display a Web page rendered from the document

---

**github_document**

*Convert to GitHub Flavored Markdown*

**Description**

Format for converting from R Markdown to GitHub Flavored Markdown.

**Usage**

```r
github_document(toc = FALSE, toc_depth = 3, fig_width = 7,
fig_height = 5, dev = "png", df_print = "default", includes = NULL,
md_extensions = NULL, hard_line_breaks = TRUE, pandoc_args = NULL,
html_preview = TRUE)
```
Arguments

- **toc**: TRUE to include a table of contents in the output
- **toc_depth**: Depth of headers to include in table of contents
- **fig_width**: Default width (in inches) for figures
- **fig_height**: Default width (in inches) for figures
- **dev**: Graphics device to use for figure output (defaults to png)
- **df_print**: Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the df_print behavior entirely by setting the option `rmarkdown::df_print` to FALSE.
- **includes**: Named list of additional content to include within the document (typically created using the `includes` function).
- **md_extensions**: Markdown extensions to be added or removed from the default definition or R Markdown. See the `rmarkdown_format` for additional details.
- **hard_line_breaks**: TRUE to generate markdown that uses a simple newline to represent a line break (as opposed to two-spaces and a newline).
- **pandoc_args**: Additional command line options to pass to pandoc
- **html_preview**: TRUE to also generate an HTML file for the purpose of locally previewing what the document will look like on GitHub.

Details

See the online documentation for additional details on using the `github_document` format.

Value

R Markdown output format to pass to `render`
html_document

**Usage**

```r
html_dependency_jquery()
html_dependency_bootstrap(theme)
html_dependency_jqueryui()
html_dependency_tocify()
html_dependency_font_awesome()
html_dependency_ionicons()
html_dependency_pagedtable()
html_dependency_highlightjs(highlight)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>theme</code></td>
<td>Visual theme (&quot;default&quot;, &quot;cerulean&quot;, &quot;journal&quot;, &quot;flatly&quot;, &quot;readable&quot;, &quot;spacelab&quot;, &quot;united&quot;, &quot;cosmo&quot;, &quot;lumen&quot;, &quot;paper&quot;, &quot;sandstone&quot;, &quot;simplex&quot;, or &quot;yeti&quot;). Pass NULL for no theme (in this case you can use the <code>css</code> parameter to add your own styles).</td>
</tr>
<tr>
<td><code>highlight</code></td>
<td>Highlighter to use</td>
</tr>
</tbody>
</table>

---

**html_document**

*Convert to an HTML document*

**Description**

Format for converting from R Markdown to an HTML document.

**Usage**

```r
html_document(toc = FALSE, toc_depth = 3, toc_float = FALSE,
              number_sections = FALSE, section_divs = TRUE, fig_width = 7,
              fig_height = 5, fig_retina = 2, fig_caption = TRUE, dev = "png",
              df_print = "default", code_folding = c("none", "show", "hide"),
              code_download = FALSE, smart = TRUE, self_contained = TRUE,
              theme = "default", highlight = "default", mathjax = "default",
              template = "default", extra_dependencies = NULL, css = NULL,
              includes = NULL, keep_md = FALSE, lib_dir = NULL,
              md_extensions = NULL, pandoc_args = NULL, ...)
```
Arguments

- **toc**: TRUE to include a table of contents in the output
- **toc_depth**: Depth of headers to include in table of contents
- **toc_float**: TRUE to float the table of contents to the left of the main document content. Rather than TRUE you may also pass a list of options that control the behavior of the floating table of contents. See the *Floating Table of Contents* section below for details.
- **number_sections**: TRUE to number section headings
- **section_divs**: Wrap sections in <div> tags (or <section> tags in HTML5), and attach identifiers to the enclosing <div> (or <section>) rather than the header itself.
- **fig_width**: Default width (in inches) for figures
- **fig_height**: Default height (in inches) for figures
- **fig_retina**: Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).
- **fig_caption**: TRUE to render figures with captions
- **dev**: Graphics device to use for figure output (defaults to png)
- **df_print**: Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses print.data.frame. The "kable" method uses the knitr::kable function. The "tibble" method uses the tibble package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the df_print behavior entirely by setting the option rmarkdown.df_print to FALSE.
- **code_folding**: Enable document readers to toggle the display of R code chunks. Specify "none" to display all code chunks (assuming they were knit with echo = TRUE). Specify "hide" to hide all R code chunks by default (users can show hidden code chunks either individually or document-wide). Specify "show" to show all R code chunks by default.
- **code_download**: Embed the Rmd source code within the document and provide a link that can be used by readers to download the code.
- **smart**: Produce typographically correct output, converting straight quotes to curly quotes, — to em-dashes, – to en-dashes, and ... to ellipses.
- **self_contained**: Produce a standalone HTML file with no external dependencies, using data: URIs to incorporate the contents of linked scripts, stylesheets, images, and videos. Note that even for self contained documents MathJax is still loaded externally (this is necessary because of its size).
- **theme**: Visual theme ("default", "cerulean", "journal", "flatly", "readable", "spacelab", "united", "cosmo", "lumen", "paper", "sandstone", "simplex", or "yeti"). Pass NULL for no theme (in this case you can use the css parameter to add your own styles).
highlight


mathjax

Include mathjax. The "default" option uses an https URL from a MathJax CDN. The "local" option uses a local version of MathJax (which is copied into the output directory). You can pass an alternate URL or pass NULL to exclude MathJax entirely.

template

Pandoc template to use for rendering. Pass "default" to use the rmarkdown package default template; pass NULL to use pandoc’s built-in template; pass a path to use a custom template that you’ve created. Note that if you don’t use the "default" template then some features of html_document won’t be available (see the Templates section below for more details).

extra_dependencies, ...

Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base

css

One or more css files to include

includes

Named list of additional content to include within the document (typically created using the includes function).

keep_md

Keep the markdown file generated by knitting.

lib_dir

Directory to copy dependent HTML libraries (e.g. jquery, bootstrap, etc.) into. By default this will be the name of the document with _files appended to it.

md_extensions

Markdown extensions to be added or removed from the default definition or R Markdown. See the rmarkdown_format for additional details.

pandoc_args

Additional command line options to pass to pandoc

Details

See the online documentation for additional details on using the html_document format.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

Value

R Markdown output format to pass to render

Navigation Bars

If you have a set of html documents which you’d like to provide a common global navigation bar for, you can include a "_navbar.yml" or "_navbar.html" file within the same directory as your html document and it will automatically be included at the top of the document.

The "_navbar.yml" file includes title, type, left, and right fields (to define menu items for the left and right of the navbar respectively). Menu items include title and href fields. For example:
The type field is optional and can take the value "default" or "inverse" (which provides a different color scheme for the navigation bar).

Alternatively, you can include a "_navbar.html" file which is a full HTML definition of a bootstrap navigation bar. For a simple example of including a navigation bar see https://github.com/rstudio/rmarkdown-website/blob/master/_navbar.html. For additional documentation on creating Bootstrap navigation bars see http://getbootstrap.com/components/#navbar.

### Floating Table of Contents

You may specify a list of options for the toc_float parameter which control the behavior of the floating table of contents. Options include:

- **collapsed** (defaults to `TRUE`) controls whether the table of contents appears with only the top-level (H2) headers. When collapsed the table of contents is automatically expanded inline when necessary.
- **smooth_scroll** (defaults to `TRUE`) controls whether page scrolls are animated when table of contents items are navigated to via mouse clicks.
- **print** (defaults to `TRUE`) controls whether the table of contents appears when user prints out the HTML page.

### Tabbed Sections

You can organize content using tabs by applying the `.tabset` class attribute to headers within a document. This will cause all sub-headers of the header with the `.tabset` attribute to appear within tabs rather than as standalone sections. For example:

```yaml
## Quarterly Results {.tabset}

### By Product

### By Region
```

You can also specify two additional attributes to control the appearance and behavior of the tabs. The `.tabset-fade` attribute causes the tabs to fade in and out when switching. The `.tabset-pills` attribute causes the visual appearance of the tabs to be "pill" rather than traditional tabs. For example:

```yaml
## Quarterly Results {.tabset .tabset-fade .tabset-pills}
```
Templates

You can provide a custom HTML template to be used for rendering. The syntax for templates is described in the pandoc documentation. You can also use the basic pandoc template by passing template = NULL.

Note however that if you choose not to use the "default" HTML template then several aspects of HTML document rendering will behave differently:

- The theme parameter does not work (you can still provide styles using the css parameter).
- For the highlight parameter, the default highlighting style will resolve to "pygments" and the "textmate" highlighting style is not available
- The toc_float parameter will not work.
- The code_folding parameter will not work.
- Tabbed sections (as described above) will not work.
- Navigation bars (as described above) will not work.
- MathJax will not work if self_contained is TRUE (these two options can’t be used together in normal pandoc templates).

Due to the above restrictions, you might consider using the includes parameter as an alternative to providing a fully custom template.

Examples

```r
## Not run:
library(rmarkdown)

render("input.Rmd", html_document())

render("input.Rmd", html_document(toc = TRUE))

## End(Not run)
```

---

**html_document_base**  
Base output format for HTML-based output formats

**Description**

Creates an HTML base output format suitable for passing as the base_format argument of the output_format function.

**Usage**

```r
html_document_base(smart = TRUE, theme = NULL, self_contained = TRUE, 
lib_dir = NULL, mathjax = "default", pandoc_args = NULL, 
template = "default", dependency_resolver = NULL, 
copy_resources = FALSE, extra_dependencies = NULL, 
bootstrap_compatible = FALSE, ...)```
## Arguments

**smart**

Produce typographically correct output, converting straight quotes to curly quotes, — to en-dashes, — to em-dashes, and ... to ellipses.

**theme**

Visual theme ("default", "cerulean", "journal", "flatly", "readable", "spacelab", "united", "cosmo", "lumen", "paper", "sandstone", "simplex", or "yeti"). Pass `NULL` for no theme (in this case you can use the `css` parameter to add your own styles).

**self_contained**

Produce a standalone HTML file with no external dependencies, using data: URLs to incorporate the contents of linked scripts, stylesheets, images, and videos. Note that even for self contained documents MathJax is still loaded externally (this is necessary because of its size).

**lib_dir**

Directory to copy dependent HTML libraries (e.g. jquery, bootstrap, etc.) into. By default this will be the name of the document with `_files` appended to it.

**mathjax**

Include mathjax. The "default" option uses an https URL from a MathJax CDN. The "local" option uses a local version of MathJax (which is copied into the output directory). You can pass an alternate URL or pass `NULL` to exclude MathJax entirely.

**pandoc_args**

Additional command line options to pass to pandoc.

**template**

Pandoc template to use for rendering. Pass "default" to use the rmarkdown package default template; pass `NULL` to use pandoc’s built-in template; pass a path to use a custom template that you’ve created. Note that if you don’t use the "default" template then some features of `html_document` won’t be available (see the Templates section below for more details).

**dependency_resolver**

A dependency resolver

**copy_resources**

Copy resources

**extra_dependencies**

Extra dependencies

**bootstrap_compatible**

Bootstrap compatible

... Ignored

## Value

HTML base output format.

---

**html_fragment** *Convert to an HTML fragment.*

## Description

An html fragment is suitable for inclusion into an external html page. See `html_document` for full details - this is a minor variation that assumes you will include the output into an existing document (e.g. a blog post).
**Usage**

```r
df_print = "default", smart = TRUE, mathjax = TRUE, includes = NULL,
keep_md = FALSE, md_extensions = NULL, pandoc_args = NULL, ...)```

**Arguments**

- **number_sections**
  - TRUE to number section headings
- **section_divs**
  - Wrap sections in `<div>` tags (or `<section>` tags in HTML5), and attach identifiers to the enclosing `<div>` (or `<section>`) rather than the header itself.
- **fig_width**
  - Default width (in inches) for figures
- **fig_height**
  - Default width (in inches) for figures
- **fig_retina**
  - Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).
- **fig_caption**
  - TRUE to render figures with captions
- **dev**
  - Graphics device to use for figure output (defaults to png)
- **df_print**
  - Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the df_print behavior entirely by setting the option `rmarkdown::df_print` to FALSE.
- **smart**
  - Produce typographically correct output, converting straight quotes to curly quotes, — to em-dashes, – to en-dashes, and ... to ellipses.
- **mathjax**
  - TRUE to convert $ and $$ math blocks into MathJax compatible output. Note that you'll still need to ensure that the page where the fragment is included loads the required MathJax scripts.
- **includes**
  - Named list of additional content to include within the document (typically created using the `includes` function).
- **keep_md**
  - Keep the markdown file generated by knitting.
- **md_extensions**
  - Markdown extensions to be added or removed from the default definition or R Markdown. See the `rmarkdown_format` for additional details.
- **pandoc_args**
  - Additional command line options to pass to pandoc
- **...**
  - Additional arguments passed to `html_document`

**Details**

See the online documentation for additional details on using the `html_fragment` format.
Value

R Markdown output format to pass to render

html_notebook

Convert to an HTML notebook

Description

Format for converting from R Markdown to an HTML notebook.

Usage

html_notebook(toc = FALSE, toc_depth = 3, toc_float = FALSE,
number_sections = FALSE, fig_width = 7, fig_height = 5,
fig_retina = 2, fig_caption = TRUE, code_folding = "show",
smart = TRUE, theme = "default", highlight = "textmate",
mathjax = "default", extra_dependencies = NULL, css = NULL,
includes = NULL, md_extensions = NULL, pandoc_args = NULL,
output_source = NULL, self_contained = TRUE, ...)

Arguments

toc TRUE to include a table of contents in the output
toc_depth Depth of headers to include in table of contents
toc_float TRUE to float the table of contents to the left of the main document content. Rather than TRUE you may also pass a list of options that control the behavior of the floating table of contents. See the Floating Table of Contents section below for details.
number_sections TRUE to number section headings
fig_width Default width (in inches) for figures
fig_height Default width (in inches) for figures
fig_retina Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).
fig_caption TRUE to render figures with captions
code_folding Enable document readers to toggle the display of R code chunks. Specify "none" to display all code chunks (assuming they were knit with echo = TRUE). Specify "hide" to hide all R code chunks by default (users can show hidden code chunks either individually or document-wide). Specify "show" to show all R code chunks by default.
smart Produce typographically correct output, converting straight quotes to curly quotes, — to em-dashes, – to en-dashes, and ... to ellipses.
html_notebook_metadata

theme
Visual theme ("default", "cerulean", "journal", "flatly", "readable", "spacelab", "united", "cosmo", "lumen", "paper", "sandstone", "simplex", or "yeti"). Pass NULL for no theme (in this case you can use the css parameter to add your own styles).

highlight

mathjax
Include mathjax. The "default" option uses an https URL from a MathJax CDN. The "local" option uses a local version of MathJax (which is copied into the output directory). You can pass an alternate URL or pass NULL to exclude MathJax entirely.

extra_dependencies
Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base

css
One or more css files to include

includes
Named list of additional content to include within the document (typically created using the includes function).

md_extensions
Markdown extensions to be added or removed from the default definition or R Markdown. See the rmarkdown_format for additional details.

pandoc_args
Additional command line options to pass to pandoc

output_source
Define an output source for R chunks (ie, outputs to use instead of those produced by evaluating the underlying R code). See html_notebook_output for more details.

self_contained
Produce a standalone HTML file with no external dependencies. Defaults to TRUE. In notebooks, setting this to FALSE is not recommended, since the setting does not apply to embedded notebook output such as plots and HTML widgets.

... Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base

Details
See the online documentation for additional details on using the html_notebook format.
html_notebook_output

Arguments

iframe  Boolean; should output be shown in an `<iframe>`?

Description

Utilities for generating output for the html_notebook format, through the output_source function attached to a output_format.

Usage

```r
html_notebook_output_html(html, meta = NULL)
html_notebook_output_img(path = NULL, bytes = NULL, attributes = NULL, meta = NULL, format = c("png", "jpeg"))
html_notebook_output_png(path = NULL, bytes = NULL, attributes = NULL, meta = NULL, format = c("png", "jpeg"))
html_notebook_output_code(code, attributes = list(class = "r"), meta = NULL)
```

Arguments

html  Arbitrary HTML content to insert.
meta  An R list of arbitrary meta-data. The data will be converted to JSON, base64-encoded, and injected into the header comment.
path  A path to a file. For functions accepting both path and bytes, if bytes is NULL, the bytewise contents will be obtained by reading the file.
bytes  The bytewise representation of content.
attributes  A named R list of HTML attributes. These will be escaped and inserted into the generated HTML as appropriate.
format  The image format; one of "png" or "jpeg".
code  Source code.

Details

See the online documentation for additional details on using the html_notebook format.
**html_vignette**

Convert to an HTML vignette

**Description**

A HTML vignette is a lightweight alternative to html_document suitable for inclusion in packages to be released to CRAN. It reduces the size of a basic vignette from 100k to around 10k.

**Usage**

```r
html_vignette(fig_width = 3, fig_height = 3, dev = "png",
              df_print = "default", css = NULL, keep_md = FALSE, readme = FALSE,
              ...)
```

**Arguments**

- `fig_width`: Default width (in inches) for figures
- `fig_height`: Default width (in inches) for figures
- `dev`: Graphics device to use for figure output (defaults to png)
- `df_print`: Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the `df_print` behavior entirely by setting the option `rmarkdown::df_print` to `false`.
- `css`: One or more css files to include
- `keep_md`: Keep the markdown file generated by knitting.
- `readme`: Use this vignette as the package README.md file (i.e. render it as README.md to the package root). Note that if there are image files within your vignette you should be sure to add README_files to .Rbuildignore
- `...`: Additional arguments passed to html_document. Please note that theme, fig_retina and highlight are hard coded. Setting any of those will yield an error.

**Details**

Compared to html_document, it:

- never uses retina figures
- never uses a theme
- has a smaller default figure size
- uses a custom css stylesheet
- uses a custom highlight scheme

See the online documentation for additional details on using the html_vignette format.
Value

R Markdown output format to pass to `render`.

includes

Include content within output

Description

Specify additional content to be included within an output document.

Usage

```r
includes(in_header = NULL, before_body = NULL, after_body = NULL)
includes_to_pandoc_args(includes, filter = identity)
```

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.
- `includes` Includes to convert to pandoc args.
- `filter` Filter to pre-process includes with.

Details

Non-absolute paths for resources referenced from the `in_header`, `before_body`, and `after_body` parameters are resolved relative to the directory of the input document.

Value

Includes list or pandoc args.

Examples

```r
## Not run:
library(rmarkdown)

text_document(includes = includes(before_body = "header.txt"))

document(includes = includes(after_body = "footer.tex"))

## End(Not run)
```
Description

Format for converting from R Markdown to an ioslides presentation.

Usage

ioslides_presentation(logo = NULL, slide_level = 2, incremental = FALSE, fig_width = 7.5, fig_height = 4.5, fig_retina = 2, fig_caption = TRUE, dev = "png", df_print = "default", smart = TRUE, self_contained = TRUE, widescreen = FALSE, smaller = FALSE, transition = "default", mathjax = "default", analytics = NULL, template = NULL, css = NULL, includes = NULL, keep_md = FALSE, lib_dir = NULL, md_extensions = NULL, pandoc_args = NULL, extra_dependencies = NULL, ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>logo</td>
<td>Path to file that includes a logo for use in the presentation (should be square and at least 128x128)</td>
</tr>
<tr>
<td>slide_level</td>
<td>Header level to consider as slide separator (Defaults to header 2)</td>
</tr>
<tr>
<td>incremental</td>
<td>True to render slide bullets incrementally. Note that if you want to reverse the default incremental behavior for an individual bullet you can preceded it with &gt;. For example: &gt; - Bullet Text</td>
</tr>
<tr>
<td>fig_width</td>
<td>Default width (in inches) for figures</td>
</tr>
<tr>
<td>fig_height</td>
<td>Default width (in inches) for figures</td>
</tr>
<tr>
<td>fig_retina</td>
<td>Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).</td>
</tr>
<tr>
<td>fig_caption</td>
<td>TRUE to render figures with captions</td>
</tr>
<tr>
<td>dev</td>
<td>Default graphics device to use for figure output</td>
</tr>
<tr>
<td>df_print</td>
<td>Method to be used for printing data frames. Valid values include &quot;default&quot;, &quot;kable&quot;, and &quot;tibble&quot;. The &quot;default&quot; method uses print.data.frame. The &quot;kable&quot; method uses the knitr::kable function. The &quot;tibble&quot; method uses the tibble package to print a summary of the data frame. In addition to the named methods you can also pass an arbitrary function to be used for printing data frames.</td>
</tr>
<tr>
<td>smart</td>
<td>Produce typographically correct output, converting straight quotes to curly quotes, — to em-dashes, – to en-dashes, and ... to ellipses.</td>
</tr>
<tr>
<td>self_contained</td>
<td>Produce a standalone HTML file with no external dependencies, using data: URLs to incorporate the contents of linked scripts, stylesheets, images, and videos. Note that even for self contained documents MathJax is still loaded externally (this is necessary because of it's size).</td>
</tr>
</tbody>
</table>
widescreen Display presentation with wider dimensions.
smaller Use smaller text on all slides. You can also enable this for individual slides by adding the .smaller attribute to the slide header (see Presentation Size below for details).
transition Speed of slide transitions. This can be "default", "slower", "faster", or a numeric value with a number of seconds (e.g. 0.5)
mathjax Include mathjax. The "default" option uses an https URL from the official MathJax CDN. The "local" option uses a local version of MathJax (which is copied into the output directory). You can pass an alternate URL or pass NULL to exclude MathJax entirely.
analytics A Google analytics property ID
template Path to a pandoc template to use instead of the default bundled template.
css One or more css files to include
includes Named list of additional content to include within the document (typically created using the includes function). If a before_body include is specified then it will replace the standard title slide entirely.
keep_md Keep the markdown file generated by knitting.
lib_dir Directory to copy dependent HTML libraries (e.g. jquery, bootstrap, etc.) into. By default this will be the name of the document with _files appended to it.
md_extensions Markdown extensions to be added or removed from the default definition or R Markdown. See the rmarkdown_format for additional details.
pandoc_args Additional command line options to pass to pandoc
extra_dependencies, ... Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base

Details
See the online documentation for additional details on using the ioslides_presentation format.

Value
R Markdown output format to pass to render

Slide Basics
You can create a slide show broken up into sections by using the # and ## heading tags (you can also create a new slide without a header using a horizontal rule (----------)). For example here's a simple slide show:

---
title: "Habits"
author: John Doe
date: March 22, 2005
output: ioslides_presentation
---

# In the morning

## Getting up

- Turn off alarm
- Get out of bed

## Breakfast

- Eat eggs
- Drink coffee

# In the evening

## Dinner

- Eat spaghetti
- Drink wine

---------

![picture of spaghetti](images/spaghetti.jpg)

## Going to sleep

- Get in bed
- Count sheep

You can add a subtitle to a slide or section by including text after the pipe (!) character. For example:

## Getting up | What I like to do first thing

### Display Modes

The following single character keyboard shortcuts enable alternate display modes:

- 'f' enable fullscreen mode
- 'w' toggle widescreen mode
- 'o' enable overview mode
- 'h' enable code highlight mode
- 'p' show presenter notes

Pressing Esc exits all of these modes. See the sections below on Code Highlighting and Presenter Mode for additional detail on those modes.
Incremental Bullets

You can render bullets incrementally by adding the incremental option:

```yaml
output:
  ioslides_presentation:
    incremental: true
```

If you want to render bullets incrementally for some slides but not others you can use this syntax:

```yaml
> - Eat eggs
> - Drink coffee
```

Presentation Size

You can display the presentation using a wider form factor using the widescreen option. You can specify that smaller text be used with the smaller option. For example:

```yaml
output:
  ioslides_presentation:
    widescreen: true
    smaller: true
```

You can also enable the smaller option on a slide-by-slide basis by adding the .smaller attribute to the slide header:

```yaml
## Getting up {.smaller}
```

Adding a Logo

You can add a logo to the presentation using the logo option (the logo should be square and at least 128x128). For example:

```yaml
output:
  ioslides_presentation:
    logo: logo.png
```

A 128x128 version of the logo graphic will be added to the title slide and an icon version of the logo will be included in the bottom-left footer of each slide.
**Build Slides**

Slides can also have a `.build` attribute that indicate that their content should be displayed incrementally. For example:

```markdown
## Getting up {.build}
```

Slide attributes can be combined if you need to specify more than one, for example:

```markdown
## Getting up {.smaller .build}
```

**Code Highlighting**

It’s possible to select subsets of code for additional emphasis by adding a special "highlight" comment around the code. For example:

```markdown
### <b>
x <- 10
y <- x * 2
### </b>
```

The highlighted region will be displayed with a bold font. When you want to help the audience focus exclusively on the highlighted region press the ‘h’ key and the rest of the code will fade away.

**Tables**

The ioslides template has an attractive default style for tables so you shouldn’t hesitate to add tables for presenting more complex sets of information. Pandoc markdown supports several syntaxes for defining tables which are described in the [pandoc online documentation](http://pandoc.org/).

**Advanced Layout**

You can center content on a slide by adding the `.flexbox` and `.vcenter` attributes to the slide title. For example:

```markdown
## Dinner {.flexbox .vcenter}
```

You can horizontally center content by enclosing it in a `div` tag with class `centered`. For example:

```html
<div class="centered">
  This text is centered.
</div>
```

You can do a two-column layout using the `columns-2` class. For example:
<div class="columns-2">
  ![Image](image.png)
  - Bullet 1
  - Bullet 2
  - Bullet 3
</div>

Note that content will flow across the columns so if you want to have an image on one side and text on the other you should make sure that the image has sufficient height to force the text to the other side of the slide.

**Text Color**

You can color content using base color classes red, blue, green, yellow, and gray (or variations of them e.g. red2, red3, blue2, blue3, etc.). For example:

```html
<div class="red">
  This text is red
</div>
```

**Presenter Mode**

A separate presenter window can also be opened (ideal for when you are presenting on one screen but have another screen that’s private to you). The window stays in sync with the main presentation window and also shows presenter notes and a thumbnail of the next slide. To enable presenter mode add `?presentme=true` to the URL of the presentation, for example:

```
mypresentation.html?presentme=true
```

The presenter mode window will open and will always re-open with the presentation until it’s disabled with:

```
mypresentation.html?presentme=false
```

To add presenter notes to a slide you include it within a "notes" div. For example:

```html
<div class="notes">
  This is my note.
  - It can contain markdown
  - like this list
</div>
```

**Printing and PDF Output**

You can print an ioslides presentation from within browsers that have good support for print CSS (i.e. as of this writing Google Chrome has the best support). Printing maintains most of the visual styles of the HTML version of the presentation.

To create a PDF version of a presentation you can use Print to PDF from Google Chrome.
**knitr_options**

Knitr options for an output format

**Description**

Define the knitr options for an R Markdown output format.

**Usage**

```r
knitr_options(opts_knit = NULL, opts_chunk = NULL, knit_hooks = NULL, 
              opts_hooks = NULL, opts_template = NULL)
```

**Arguments**

- `opts_knit` List of package level knitr options (see `opts_knit`)
- `opts_chunk` List of chunk level knitr options (see `opts_chunk`)
- `knit_hooks` List of hooks for R code chunks, inline R code, and output (see `knit_hooks`)
- `opts_hooks` List of hooks for code chunk options (see `opts_hooks`)
- `opts_template` List of templates for chunk level knitr options (see `opts_template`)

**Value**

An list that can be passed as the `knitr` argument of the `output_format` function.

**See Also**

- `output_format`

---

**knitr_options_html**

Knitr options for an HTML output format

**Description**

Define knitr options for an R Markdown output format that creates HTML output.

**Usage**

```r
knitr_options_html(fig_width, fig_height, fig_retina, keep_md, dev = "png")
```
knitr_options_pdf

Arguments

fig_width  Default width (in inches) for figures
fig_height Default width (in inches) for figures
fig_retina Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).
keep_md    Keep the markdown file generated by knitting.
develop    Graphics device to use for figure output (defaults to png)

Value

An list that can be passed as the knitr argument of the output_format function.

See Also

knitr_options, output_format

knitr_options_pdf Knitr options for a PDF output format

Description

Define knitr options for an R Markdown output format that creates PDF output.

Usage

knitr_options_pdf(fig_width, fig_height, fig_crop, dev = "pdf")

Arguments

fig_width  Default width (in inches) for figures
fig_height Default width (in inches) for figures
fig_crop   TRUE to automatically apply the pdfcrop utility (if available) to pdf figures
develop    Graphics device to use for figure output (defaults to png)

Value

An list that can be passed as the knitr argument of the output_format function.

See Also

knitr_options, output_format
**knit_params_ask**

Run a shiny application asking for parameter configuration for the given document.

**Description**

Run a shiny application asking for parameter configuration for the given document.

**Usage**

```r
knit_params_ask(file = NULL, input_lines = NULL, params = NULL,
shiny_args = NULL, save_caption = "Save",
encoding = getOption("encoding"))
```

**Arguments**

- **file**: Path to the R Markdown document with configurable parameters.
- **input_lines**: Content of the R Markdown document. If NULL, the contents of file will be read.
- **params**: A named list of optional parameter overrides used in place of the document defaults.
- **shiny_args**: Additional arguments to `runApp`.
- **save_caption**: Caption to use use for button that saves/confirms parameters.
- **encoding**: The encoding of the input file; see `file`.

**Value**

named list with overridden parameter names and value.

---

**latex_dependency**

Define a LaTeX package dependency

**Description**

Define a LaTeX package dependency

**Usage**

```r
latex_dependency(name, options = NULL)
```

**Arguments**

- **name**: The LaTeX package name
- **options**: The LaTeX options for the package
Convert to a markdown document

Description

Format for converting from R Markdown to another variant of markdown (e.g. strict markdown or github flavored markdown)

Usage

```r
md_document(variant = "markdown_strict", preserve_yaml = FALSE,
             toc = FALSE, toc_depth = 3, fig_width = 7, fig_height = 5,
             fig_retina = NULL, dev = "png", df_print = "default", includes = NULL,
             md_extensions = NULL, pandoc_args = NULL)
```

Arguments

- **variant**: Markdown variant to produce (defaults to "markdown_strict"). Other valid values are "markdown_github", "markdown_mmd", markdown_phpextra", or even "markdown" (which produces pandoc markdown). You can also compose custom markdown variants, see the pandoc online documentation for details.

- **preserve_yaml**: Preserve YAML front matter in final document.

- **toc**: TRUE to include a table of contents in the output

- **toc_depth**: Depth of headers to include in table of contents

- **fig_width**: Default width (in inches) for figures

- **fig_height**: Default width (in inches) for figures

- **fig_retina**: Scaling to perform for retina displays. Defaults to NULL which performs no scaling. A setting of 2 will work for all widely used retina displays, but will also result in the output of `<img>` tags rather than markdown images due to the need to set the width of the image explicitly.

- **dev**: Graphics device to use for figure output (defaults to png)

- **df_print**: Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the df_print behavior entirely by setting the option `rmarkdown::df_print` to FALSE.

- **includes**: Named list of additional content to include within the document (typically created using the `includes` function).

- **md_extensions**: Markdown extensions to be added or removed from the default definition or R Markdown. See the `rmarkdown_format` for additional details.

- **pandoc_args**: Additional command line options to pass to pandoc
**metadata**

Details

See the online documentation for additional details on using the md_document format.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

Value

R Markdown output format to pass to `render`

Examples

```r
## Not run:
library(rmarkdown)
render("input.Rmd", md_document())
render("input.Rmd", md_document(variant = "markdown_github"))
## End(Not run)
```

---

**metadata**

The YAML metadata of the current R Markdown document

Description

The object `metadata` stores the YAML metadata of the current R Markdown document as a list, which you may use in the R code chunks, e.g. `rmarkdown::metadata$title` (the title of the document), `rmarkdown::metadata$author`, and `rmarkdown::metadata$foo` (if you have a YAML field named foo), etc.

Format

An object of class `list` of length 0.

Examples

`rmarkdown::metadata`
odt_document

Convert to an OpenDocument Text (ODT) document

Description

Format for converting from R Markdown to an ODT document.

Usage

odt_document(fig_width = 5, fig_height = 4, fig_caption = TRUE, template = "default", reference_odt = "default", includes = NULL, keep_md = FALSE, md_extensions = NULL, pandoc_args = NULL)

Arguments

- **fig_width** Default width (in inches) for figures
- **fig_height** Default width (in inches) for figures
- **fig_caption** TRUE to render figures with captions
- **template** Pandoc template to use for rendering. Pass "default" to use the rmarkdown package default template; pass NULL to use pandoc's built-in template; pass a path to use a custom template that you've created. See the documentation on pandoc online documentation for details on creating custom templates.
- **reference_odt** Use the specified file as a style reference in producing an odt file. For best results, the reference odt should be a modified version of an odt file produced using pandoc. Pass "default" to use the rmarkdown default styles.
- **includes** Named list of additional content to include within the document (typically created using the includes function).
- **keep_md** Keep the markdown file generated by knitting.
- **md_extensions** Markdown extensions to be added or removed from the default definition or R Markdown. See the rmarkdown_format for additional details.
- **pandoc_args** Additional command line options to pass to pandoc

Details

See the online documentation for additional details on using the odt_document format.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

Value

R Markdown output format to pass to render
Examples

```r
## Not run:
library(rmarkdown)

# simple invocation
render("input.Rmd", odt_document())

# specify an option for syntax highlighting
render("input.Rmd", odt_document(highlight = "zenburn"))

## End(Not run)
```

### output_format

**Define an R Markdown output format**

#### Description

Define an R Markdown output format based on a combination of knitr and pandoc options.

#### Usage

```r
output_format(knitr, pandoc, keep_md = FALSE, clean_supporting = TRUE,
    df_print = NULL, pre_knit = NULL, post_knit = NULL,
    pre_processor = NULL, intermediates_generator = NULL,
    post_processor = NULL, on_exit = NULL, base_format = NULL)
```

#### Arguments

- `knitr` Knitr options for an output format (see `knitr_options`)
- `pandoc` Pandoc options for an output format (see `pandoc_options`)
- `keep_md` Keep the markdown file generated by knitting. Note that if this is `TRUE` then `clean_supporting` will always be `FALSE`.
- `clean_supporting` Cleanup any supporting files after conversion see `render_supporting_files`
- `df_print` Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the `df_print` behavior entirely by setting the option `rmarkdown::df_print` to `FALSE`.
- `pre_knit` An optional function that runs before knitting which receives the input (input filename passed to `render`) and ... (for future expansion) arguments.
post_knit  An optional function that runs after knitting which receives the metadata, input_file, runtime, and ... (for future expansion) arguments. This function can return additional arguments to pass to pandoc and can call knitr::knit_meta_add to add additional dependencies based on the contents of the input_file or on other assets side by side with it that may be used to produce html with dependencies during subsequent processing.

pre_processor  An optional pre-processor function that receives the metadata, input_file, runtime, knit_meta, files_dir, and output_dir and can return additional arguments to pass to pandoc.

intermediates_generator  An optional function that receives the original input_file, its encoding, and the intermediates directory (i.e. the intermediates_dir argument to render). The function should generate and return the names of any intermediate files required to render the input_file.

post_processor  An optional post-processor function that receives the metadata, input_file, output_file, clean, and verbose parameters, and can return an alternative output_file.

on_exit  A function to call when rmarkdown::render() finishes execution (as registered with a on.exit handler).

base_format  An optional format to extend.

Value

An R Markdown output format definition that can be passed to render.

See Also

render, knitr_options, pandoc_options

Examples

## Not run:
output_format(knitr = knitr_options(opts_chunk = list(dev = 'png')),
             pandoc = pandoc_options(to = "html"))

## End(Not run)

---

output_metadata  The output metadata object

Description

This object provides a mechanism for users to attach metadata as an attribute (named rmd_output_metadata) of the returned value of render(). The initial value of the metadata comes from in the rmd_output_metadata field of the YAML frontmatter of an R Markdown document. The metadata can be queried via the output_metadata$get() method, and modified via the output_metadata$set() method.
paged_table

Create a table in HTML with support for paging rows and columns

Description
Create a table in HTML with support for paging rows and columns

Usage
paged_table(x, options = NULL)

Arguments
x: a data frame to be rendered as a paged table.
options: options for printing the paged table

pandoc_args
Functions for generating pandoc command line arguments

Description
Functions that assist in creating various types of pandoc command line arguments (e.g. for templates, table of contents, highlighting, and content includes).

Usage
pandoc_variable_arg(name, value)
pandoc_include_args(in_header = NULL, before_body = NULL, after_body = NULL)
pandoc_highlight_args(highlight, default = "tango")
pandoc_latex_engine_args(latex_engine)
pandoc_toc_args(toc, toc_depth = 3)

Arguments
name: Name of template variable to set.
value: Value of template variable (defaults to true if missing).
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.
highlight The name of a pandoc syntax highlighting theme.
default The highlighting theme to use if "default" is specified.
latex_engine LaTeX engine for producing PDF output. Options are "pdflatex", "lualatex", and "xelatex".
toc TRUE to include a table of contents in the output.
toc_depth Depth of headers to include in table of contents.

Details

Non-absolute paths for resources referenced from the in_header, before_body, and after_body parameters are resolved relative to the directory of the input document.

Value

A character vector with pandoc command line arguments.

Examples

```r
## Not run:
library(rmarkdown)

pandoc_include_args(before_body = "header.htm")
pandoc_include_args(before_body = "header.tex")

pandoc_highlight_args("kate")
pandoc_latex_engine_args("pdflatex")
pandoc_toc_args(toc = TRUE, toc_depth = 2)
## End(Not run)
```

---

pandoc_available Check pandoc availability and version

Description

Determine whether pandoc is currently available on the system (optionally checking for a specific version or greater). Determine the specific version of pandoc available.

Usage

```r
pandoc_available(version = NULL, error = FALSE)
pandoc_version()
```
Arguments

- **version**: Required version of pandoc
- **error**: Whether to signal an error if pandoc with the required version is not found

Details

The system environment variable ‘PATH’ as well as the version of pandoc shipped with RStudio (its location is set via the environment variable ‘RSTUDIO_PANDOC’ by RStudio products like the RStudio IDE, RStudio Server, Shiny Server, and RStudio Connect, etc) are scanned for pandoc and the highest version available is used. Please do not modify the environment variable ‘RSTUDIO_PANDOC’ unless you know what it means.

Value

- **pandoc_available**: returns a logical indicating whether the required version of pandoc is available.
- **pandoc_version**: returns a **numeric_version** with the version of pandoc found.

Examples

```r
## Not run:
library(rmarkdown)

if (pandoc_available())
  cat("pandoc", as.character(pandoc_version()), "is available!\n")

if (pandoc_available("1.12.3"))
  cat("required version of pandoc is available!\n")

## End(Not run)
```

---

**pandoc_citeproc_convert**

*Convert a bibliography file*

Description

Convert a bibliography file (e.g. a BibTeX file) to an R list, JSON text, or YAML text.

Usage

```r
pandoc_citeproc_convert(file, type = c("list", "json", "yaml"))
```

Arguments

- **file**: Bibliography file
- **type**: Conversion type
**pandoc_convert**  
*Convert a document with pandoc*

**Description**

Convert documents to and from various formats using the pandoc utility.

**Usage**

```r
pandoc_convert(input, to = NULL, from = NULL, output = NULL, citeproc = FALSE, options = NULL, verbose = FALSE, wd = NULL)
```

**Arguments**

- `input`  
  Character vector containing paths to input files (files must be UTF-8 encoded)
- `to`  
  Format to convert to (if not specified, you must specify `output`)
- `from`  
  Format to convert from (if not specified then the format is determined based on the file extension of `input`).
- `output`  
  Output file (if not specified then determined based on format being converted to).
- `citeproc`  
  TRUE to run the pandoc-citeproc filter (for processing citations) as part of the conversion.
- `options`  
  Character vector of command line options to pass to pandoc.
- `verbose`  
  TRUE to show the pandoc command line which was executed
- `wd`  
  Working directory in which code will be executed. If not supplied, defaults to the common base directory of `input`.

**Details**

Supported input and output formats are described in the [pandoc user guide](https://pandoc.org/). The system path as well as the version of pandoc shipped with RStudio (if running under RStudio) are scanned for pandoc and the highest version available is used.

**Value**

For `type = "list"`, and R list. For `type = "json"` or `type = "yaml"`, a character vector with the specified format.
Examples

```r
## Not run:
library(rmarkdown)

# convert markdown to various formats
pandoc_convert("input.md", to = "html")
pandoc_convert("input.md", to = "pdf")

# process citations
pandoc_convert("input.md", to = "html", citeproc = TRUE)

# add some pandoc options
pandoc_convert("input.md", to="pdf", options = c("--listings"))

## End(Not run)
```

### pandoc_options

#### Pandoc options for an output format

**Description**

Define the pandoc options for an R Markdown output format.

**Usage**

```r
pandoc_options(to, from = rmarkdown_format(), args = NULL,
keep_tex = FALSE, latex_engine = c("pdflatex", "lualatex", "xelatex"),
ext = NULL)
```

**Arguments**

- **to**
  - Pandoc format to convert to
- **from**
  - Pandoc format to convert from
- **args**
  - Character vector of command line arguments to pass to pandoc
- **keep_tex**
  - Keep the intermediate tex file used in the conversion to PDF (applies only to 'latex' and 'beamer' target formats)
- **latex_engine**
  - LaTeX engine to producing PDF output (applies only to 'latex' and 'beamer' target formats)
- **ext**
  - File extension (e.g. ".tex") for output file (if NULL chooses default based on to). This is typically used to force the final output of a latex or beamer conversion to be .tex rather than .pdf.

**Details**

The `from` argument should be used very cautiously as it's important for users to be able to rely on a stable definition of supported markdown extensions.
pandoc_self_contained_html

Value
An list that can be passed as the pandoc argument of the output_format function.

See Also
output_format, rmarkdown_format

pandoc_path_arg Transform path for passing to pandoc

Description
Transform a path for passing to pandoc on the command line. Calls path.expand on all platforms. On Windows, transform it to a short path name if it contains spaces, and then convert forward slashes to backslashes (as required by pandoc for some path references).

Usage
pandoc_path_arg(path, backslash = TRUE)

Arguments

path Path to transform
backslash Whether to replace forward slashes in path with backslashes on Windows.

Value
Transformed path that can be passed to pandoc on the command line.

pandoc_self_contained_html Create a self-contained HTML document using pandoc.

Description
Create a self-contained HTML document by base64 encoding images, scripts, and stylesheets referred by the input document.

Usage
pandoc_self_contained_html(input, output)

Arguments

input Input html file to create self-contained version of.
output Path to save output.
pandoc_template

Value

(Invisibly) The path of the generated file.

Description

Use the pandoc templating engine to render a text file. Substitutions are done using the metadata list passed to the function.

Usage

pandoc_template(metadata, template, output, verbose = FALSE)

Arguments

- metadata A named list containing metadata to pass to template.
- template Path to a pandoc template.
- output Path to save output.
- verbose TRUE to show the pandoc command line which was executed.

Value

(Invisibly) The path of the generated file.

parse_html_notebook

Parse an HTML Notebook

Description

Parse an HTML notebook, retrieving annotation information related to generated outputs in the document, as well as the original R Markdown source document.

Usage

parse_html_notebook(path, encoding = "UTF-8")

Arguments

- path The path to an R Notebook file (with extension .nb.html).
- encoding The document’s encoding (assumed as "UTF-8" by default).

Details

See the online documentation for additional details on using the html_notebook format.
pdf_document

Convert to a PDF/LaTeX document

Description

Formats for converting from R Markdown to a PDF or LaTeX document.

Usage

```r
pdf_document(toc = FALSE, toc_depth = 2, number_sections = FALSE,
fig_width = 6.5, fig_height = 4.5, fig_crop = TRUE,
fig_caption = TRUE, dev = "pdf", df_print = "default",
highlight = "default", template = "default", keep_tex = FALSE,
latex_engine = "pdflatex", citation_package = c("none", "natbib",
"biblatex"), includes = NULL, md_extensions = NULL, pandoc_args = NULL,
extra_dependencies = NULL)
latex_document(...)
latex_fragment(...)
```

Arguments

- `toc` 
  TRUE to include a table of contents in the output
- `toc_depth` 
  Depth of headers to include in table of contents
- `number_sections` 
  TRUE to number section headings
- `fig_width` 
  Default width (in inches) for figures
- `fig_height` 
  Default width (in inches) for figures
- `fig_crop` 
  TRUE to automatically apply the pdfcrop utility (if available) to pdf figures
- `fig_caption` 
  TRUE to render figures with captions
- `dev` 
  Graphics device to use for figure output (defaults to pdf)
- `df_print` 
  Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the df_print behavior entirely by setting the option `rmarkdown::df_print` to FALSE.
- `highlight` 
**template** Pandoc template to use for rendering. Pass "default" to use the rmarkdown package default template; pass NULL to use pandoc’s built-in template; pass a path to use a custom template that you’ve created. See the documentation on [pandoc online documentation](#) for details on creating custom templates.

**keep_tex** Keep the intermediate tex file used in the conversion to PDF

**latex_engine** LaTeX engine for producing PDF output. Options are "pdflatex", "lualatex", and "xelatex".

**citation_package** The LaTeX package to process citations, natbib or biblatex. Use none if neither package is to be used.

**includes** Named list of additional content to include within the document (typically created using the includes function).

**md_extensions** Markdown extensions to be added or removed from the default definition or R Markdown. See the [rmarkdown_format](#) for additional details.

**pandoc_args** Additional command line options to pass to pandoc

**extra_dependencies** A LaTeX dependency latex_dependency(), a list of LaTeX dependencies, a character vector of LaTeX package names (e.g. c("framed", "hyperref")), or a named list of LaTeX package options with the names being package names (e.g. list(hyperref = c("unicode=true", "breaklinks=true"), lmodern = NULL)). It can be used to add custom LaTeX packages to the .tex header.

... Arguments passed to pdf_document().

**Details**

See the [online documentation](#) for additional details on using the pdf_document format.

Creating PDF output from R Markdown requires that LaTeX be installed.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

Many aspects of the LaTeX template used to create PDF documents can be customized using metadata. For example:

```r
---
title: "Crop Analysis Q3 2013"
fontsize: 11pt
geometry: margin=1in
---
```

Available metadata variables include:

**lang** Document language code (e.g. "es", "fr", "pt-BR")

**fontsize** Font size (e.g. 10pt, 11pt, 12pt)
documentclass  LaTeX document class (e.g. article)
classoption  Option for documentclass (e.g. onesiside); may be repeated
geometry  Options for geometry class (e.g. margin=lin); may be repeated
mainfont, sansfont, monofont, mathfont  Document fonts (works only with xelatex and luatex, see the latex_engine option)
linkcolor, urlcolor, citecolor  Color for internal, external, and citation links (red, green, magenta, cyan, blue, black)
linestretch  Options for line spacing (e.g. 1, 1.5, 3)

Value

R Markdown output format to pass to render

Examples

```r
## Not run:
library(rmarkdown)

# simple invocation
render("input.Rmd", pdf_document())

# specify an option for latex engine
render("input.Rmd", pdf_document(latex_engine = "lualatex"))

# add a table of contents and pass an option to pandoc
render("input.Rmd", pdf_document(toc = TRUE, "--listings"))

## End(Not run)
```

---

**powerpoint_presentation**

*Convert to a PowerPoint presentation*

Description

Format for converting from R Markdown to a PowerPoint presentation. Pandoc v2.0.5 or above is required.

Usage

```r
powerpoint_presentation(toc = FALSE, toc_depth = 2, fig_width = 5,
fig_height = 4, fig_caption = TRUE, df_print = "default",
smart = TRUE, keep_md = FALSE, md_extensions = NULL,
slide_level = NULL, reference_doc = "default", pandoc_args = NULL)
```
Arguments

- **toc**: TRUE to include a table of contents in the output
- **toc_depth**: Depth of headers to include in table of contents
- **fig_width**: Default width (in inches) for figures
- **fig_height**: Default width (in inches) for figures
- **fig_caption**: TRUE to render figures with captions
- **df_print**: Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the `df_print` behavior entirely by setting the option `rmarkdown::df_print` to FALSE.
- **smart**: Produce typographically correct output, converting straight quotes to curly quotes, — to em-dashes, – to en-dashes, and ... to ellipses.
- **keep_md**: Keep the markdown file generated by knitting.
- **md_extensions**: Markdown extensions to be added or removed from the default definition or R Markdown. See the `rmarkdown_format` for additional details.
- **slide_level**: The heading level which defines individual slides. By default this is the highest header level in the hierarchy that is followed immediately by content, and not another header, somewhere in the document. This default can be overridden by specifying an explicit `slide_level`.
- **reference_doc**: Path to a PowerPoint template.
- **pandoc_args**: Additional command line options to pass to pandoc

Value

R Markdown output format to pass to `render`
#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>dir</td>
<td>Directory</td>
</tr>
<tr>
<td>file</td>
<td>File</td>
</tr>
</tbody>
</table>

#### Value

Relative path from the directory to the file (or the unmodified file path if the file does not appear to be in the directory).

---

**render**  
*Render R Markdown*

#### Description

Render the input file to the specified output format using pandoc. If the input requires knitting then *knit* is called prior to pandoc.

#### Usage

```r
render(input, output_format = NULL, output_file = NULL, output_dir = NULL, 
       output_options = NULL, intermediates_dir = NULL, 
       knit_root_dir = NULL, 
       runtime = c("auto", "static", "shiny", "shiny_prerendered"), 
       clean = TRUE, params = NULL, knit_meta = NULL, envir = parent.frame(), 
       run_pandoc = TRUE, quiet = FALSE, encoding = getOption("encoding"))
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Input file (R script, Rmd, or plain markdown).</td>
</tr>
<tr>
<td>output_format</td>
<td>R Markdown output format to convert to. Pass &quot;all&quot; to render all formats defined within the file. Pass the name of a format (e.g. &quot;html_document&quot;) to render a single format or pass a vector of format names to render multiple formats. Alternatively you can pass an output format object; e.g. html_document(). If NULL is passed then the output format is the first one defined in the YAML metadata of the input file (defaulting to HTML if none is specified).</td>
</tr>
<tr>
<td>output_options</td>
<td>List of output options that can override the options specified in metadata (e.g. could be used to force self_contained or mathjax = &quot;local&quot;). Note that this is only valid when the output format is read from metadata (i.e. not a custom format object passed to output_format).</td>
</tr>
<tr>
<td>output_file</td>
<td>Output file. If NULL then a default based on the name of the input file is chosen.</td>
</tr>
<tr>
<td>output_dir</td>
<td>Output directory. An alternate directory to write the output file to (defaults to the directory of the input file).</td>
</tr>
<tr>
<td>intermediates_dir</td>
<td>Intermediate files directory. If NULL, intermediate files are written to the same directory as the input file; otherwise.</td>
</tr>
</tbody>
</table>
knit_root_dir: The working directory in which to knit the document; uses knitr's root.dir
knit option. NULL means to follow the knitr default, which is to use the parent
directory of the document.

runtime: The runtime target for rendering. static produces output intended for static
files; shiny produces output suitable for use in a Shiny document (see run).
The default, auto, allows the runtime target specified in the YAML metadata
to take precedence, and renders for a static runtime target otherwise.

clean: TRUE to clean intermediate files created during rendering.

params: List of named parameters that override custom params specified within the YAML
front-matter (e.g. specifying a dataset to read or a date range to confine output
to). Pass "ask" to start an application that helps guide parameter configuration.

knit_meta: (For expert use) Meta data generated by knitr.

envir: The environment in which the code chunks are to be evaluated during knitting
(can use new.env() to guarantee an empty new environment).

run_pandoc: Whether to run Pandoc to convert Markdown output.

quiet: TRUE to suppress printing of the pandoc command line.

encoding: The encoding of the input file; see file.

Details

Note that the knitr error option is set to FALSE during rendering (which is different from the knitr
default value of TRUE).

For additional details on rendering R scripts see Compiling R scripts to a notebook.

If no output_format parameter is specified then the output format is read from the YAML front-
matter of the input file. For example, the following YAML would yield a PDF document:

output: pdf_document

Additional format options can also be specified in metadata. For example:

output:
  pdf_document:
    toc: true
    highlight: zenburn

Multiple formats can be specified in metadata. If no output_format is passed to render then the
first one defined will be used:

output:
  pdf_document:
    toc: true
    highlight: zenburn
  html_document:
    toc: true
    theme: united
Formats specified in metadata can be any one of the built in formats (e.g. `html_document`, `pdf_document`) or a format defined in another package (e.g. `pkg::custom_format`). If there is no format defined in the YAML then `html_document` will be used.

**Value**

When `run_pandoc = TRUE`, the compiled document is written into the output file, and the path of the output file is returned.

When `run_pandoc = FALSE`, the path of the Markdown output file, with attributes `knit_meta` (the `knitr` meta data collected from code chunks) and `intermediates` (the intermediate files.directories generated by `render()`).

**R Markdown**

R Markdown supports all of the base pandoc markdown features as well as some optional features for compatibility with GitHub Flavored Markdown (which previous versions of R Markdown were based on). See `rmarkdown_format` for details.

**See Also**

`knit`, `output_format`, `pandoc`

**Examples**

```r
## Not run:

library(rmarkdown)

# render the default (first) format defined in the fileender("input.Rmd")

# render all formats defined in the fileender("input.Rmd", "all")

# render a single format
render("input.Rmd", "html_document")

# render multiple formats
render("input.Rmd", c("html_document", "pdf_document"))

## End(Not run)
```

---

**render_delayed**

*Delay Rendering for an Expression*

**Description**

In a Shiny document, evaluate the given expression after the document has finished rendering, instead of during render.
**render_delayed**

### Usage

`render_delayed(expr)`

### Arguments

- **expr**: The expression to evaluate.

### Details

This function is useful inside Shiny documents. It delays the evaluation of its argument until the document has finished its initial render, so that the document can be viewed before the calculation is finished.

Any expression that returns HTML can be wrapped in `render_delayed`.

### Value

An object representing the expression.

### Note

`expr` is evaluated in a **copy** of the environment in which the `render_delayed` call appears. Consequently, no side effects created by `expr` are visible in succeeding expressions, nor are changes to the environment after the call to `render_delayed` visible to `expr`.

`expr` must be an expression that produces HTML.

### Examples

```r
## Not run:
# Add the following code to an R Markdown document

div(Sys.time())

render_delayed(
  Sys.sleep(3)  # simulate an expensive computation
  div(Sys.time())
)

div(Sys.time())

## End(Not run)
```
**render_site**

**Render multiple documents as a website**

**Description**

Render all of the R Markdown documents within a directory as a website.

**Usage**

```
render_site(input = ".", output_format = "all", envir = parent.frame(),
            quiet = FALSE, encoding = getOption("encoding"))
```

```
clean_site(input = ".", preview = FALSE, quiet = FALSE,
           encoding = getOption("encoding"))
```

```
site_config(input = ".", encoding = getOption("encoding"))
```

```
site_generator(input = ".", output_format = NULL,
               encoding = getOption("encoding"))
```

```
default_site_generator(input, encoding = getOption("encoding"), ...) 
```

**Arguments**

- **input**: Website directory (or the name of a file within the directory)
- **output_format**: R Markdown format to convert to (defaults to "all").
- **envir**: The environment in which the code chunks are to be evaluated during knitting (can use `new.env` to guarantee an empty new environment).
- **preview**: Whether to list the files to be removed rather than actually removing them.
- **quiet**: TRUE to suppress messages and other output.
- **encoding**: The encoding of the input file; see `file`.
- **...**: Currently unused

**Details**

The `render_site` function enables you to render a collection of markdown documents within a directory as a website. There are two requirements for a directory to be rendered as a website:

1. It must contain either an "index.Rmd" or "index.md" file.
2. It must contain a site configuration file ("_site.yml").

The most minimal valid website is an empty "index.Rmd" and an empty "_site.yml". With this configuration a single empty webpage would be generated via a call to `render_site`. If you add additional markdown documents to the directory they will also be rendered. By default a site is rendered in the following fashion:
1. R Markdown (.Rmd) and plain markdown (.md) files in the root directory are rendered. Note however that markdown files beginning with "_" are not rendered (this is a convention to designate files that are included by top level documents).

2. All output and supporting files are copied to a "_site" subdirectory of the website directory (this is configurable, see discussion below).

3. The following files are **not** copied to the "_site" sub-directory:
   - Files beginning with "." (hidden files).
   - Files beginning with "_"
   - Files known to contain R source code (e.g. ".R", ".s", ".Rmd"), R data (e.g. ".RData", ".rds"), or configuration data (e.g. ".Rproj", "rsconnect").

   Note that you can override which files are included or excluded via settings in "_site.yml" (described below)

4. Normally R Markdown renders documents as self-contained HTML. However, render_site ensures that dependencies (e.g. CSS, JavaScript, images, etc.) remain in external files. CSS/JavaScript libraries are copied to a "site_libs" sub-directory and plots/images are copied to "_files" sub-directories.

   You can remove the files generated by render_site using the clean_site function.

**Value**

render_site returns the name of the site output file (relative to the input directory). clean_site returns the names of the generated files removed during cleaning. site_config returns the contents of _site.yml as an R list default_site_generator returns the default site generator for R Markdown websites

**Configuration**

A "_site.yml" file can be used to configure the behavior of site generation. Here is an example configuration file:

```yaml
name: my-website
output_dir: _site
include: ["demo.R"]
exclude: ["docs.txt", ".csv"]
navigation:
  title: "My Website"
  left:
    - text: "Home"
      href: index.html
    - text: "About"
      href: about.html
output:
  html_document:
    toc: true
    highlight: textmate
```
The name field provides a suggested URL path for your website when it is published (by default this is just the name of the directory containing the site). The output_dir indicates which directory to copy site content into ("site" is the default if none is specified). Note that this can be "." to keep all content within the root website directory alongside the source code.

The include and exclude fields enable you to override the default behavior vis-a-vis what files are copied into the "site" directory (wildcards can be used as in the above example).

The navbar field can be used to define a navigation bar for websites based on the html_document format.

Finally, the output field enables you to specify output options that are common to all documents within the website (you can also still provide local options within each document that override any common options).

**Custom Site Generation**

The behavior of the default site generation function (rmarkdown::default_site) is described above. It is also possible to define a custom site generator that has alternate behavior. A site generator is an R function that is bound to by including it in the "site:" field of the "index.Rmd" or "index.md" file. For example:

```r
title: "My Book"
output: bookdown::gitbook
site: bookdown::bookdown_site
```

A site generation function should return a list with the following elements:

- **name** The name for the website (e.g. the parent directory name).
- **output_dir** The directory where the website output is written to. This path should be relative to the site directory (e.g. "." or "site")
- **render** An R function that can be called to generate the site. The function should accept the input_file, output_format, envir, quiet, and encoding arguments.
- **clean** An R function that returns relative paths to the files generated by render_site (these files are the ones which will be removed by the clean_site function).

Note that the input_file argument will be NULL when the entire site is being generated. It will be set to a specific file name if a front-end tool is attempting to preview it (e.g. RStudio IDE via the Knit button).

When quiet = FALSE the render function should also print a line of output using the message function indicating which output file should be previewed, for example:

```r
if (!quiet)
    message("\nOutput created: ", output)
```

Emitting this line enables front-ends like RStudio to determine which file they should open to preview the website.

See the source code of the rmarkdown::default_site function for a example of a site generation function.
**render_supporting_files**

*Render supporting files for an input document*

**Description**

Render (copy) required supporting files for an input document to the _files directory associated with the document.

**Usage**

```r
render_supporting_files(from, files_dir, rename_to = NULL)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>from</td>
<td>Directory to copy from</td>
</tr>
<tr>
<td>files_dir</td>
<td>Directory to copy files into</td>
</tr>
<tr>
<td>rename_to</td>
<td>Optional rename of source directory after it is copied</td>
</tr>
</tbody>
</table>

**Value**

The relative path to the supporting files. This path is suitable for inclusion in HTML `href` and `src` attributes.

---

**resolve_output_format**  
*Resolve the output format for an R Markdown document*

**Description**

Read the YAML metadata (and any common _output.yml file) for the document and return an output format object that can be passed to the `render` function.

**Usage**

```r
resolve_output_format(input, output_format = NULL, output_options = NULL, encoding = getOption("encoding"))
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>input</td>
<td>Input file (Rmd or plain markdown)</td>
</tr>
<tr>
<td>output_format</td>
<td>Name of output format (or NULL to use the default format for the input file).</td>
</tr>
<tr>
<td>output_options</td>
<td>List of output options that should override the options specified in metadata.</td>
</tr>
<tr>
<td>encoding</td>
<td>The encoding of the input file; see <code>file</code>.</td>
</tr>
</tbody>
</table>
Details

This function is useful for front-end tools that need to modify the default behavior of an output format.

Value

An R Markdown output format definition that can be passed to `render`.

```
 rmarkdown_format(R Markdown input format definition)
```

Description

Compose a pandoc markdown input definition for R Markdown that can be passed as the `from` argument of `pandoc_options`.

Usage

`rmarkdown_format(extensions = NULL)`

`from_rmarkdown(implicit_figures = TRUE, extensions = NULL)`

Arguments

- `extensions`: Markdown extensions to be added or removed from the default definition of R Markdown.
- `implicit_figures`: Automatically make figures from images (defaults to `TRUE`).

Details

By default R Markdown is defined as all pandoc markdown extensions with the following tweaks for backward compatibility with the markdown package (+ features are added, - features are removed):

```
+autolink_bare_uris
+ascii_identifier
+tex_math_single_backslash
```

For more on pandoc markdown see the pandoc online documentation.

Value

Pandoc markdown format specification
R Markdown Metadata

Description

Rmd files include a metadata section (typically located at the top of the file) that can specify (among other things) the title, author, and date of the document. Metadata adheres to the YAML format and is delimited by lines containing three dashes (---). Here is an example metadata section:

```yaml
---
title: "Crop Analysis Q3 2013"
author: Martha Smith
date: October 23rd, 2013
---
```

Note that the title field is quoted. This is because titles often contained embedded colons (:) and colons followed by a space need to be quoted in YAML.

Details

When title, author, and date metadata is provided it’s used to automatically create a title section within output documents. If you don’t want this section included in your document then you should remove the corresponding metadata fields.

When generating PDF and Beamer output there are also a number of other metadata fields that can be included to customize the appearance and theme of PDF output. For more details see the documentation for `pdf_document` and `beamer.presentation`.

See Also

`output_format`, `pandoc_options`

Examples

```r
## Not run:
rmkdown_format("-implicit_figures")
## End(Not run)
```
Description

Format for converting from R Markdown to an RTF document.

Usage

```r
tft_document(toc = FALSE, toc_depth = 3, fig_width = 5, fig_height = 4,
             keep_md = FALSE, md_extensions = NULL, pandoc_args = NULL)
```

Arguments

- `toc` | TRUE to include a table of contents in the output
- `toc_depth` | Depth of headers to include in the table of contents
- `fig_width` | Default width (in inches) for figures
- `fig_height` | Default height (in inches) for figures
- `keep_md` | Keep the markdown file generated by knitting.
- `md_extensions` | Markdown extensions to be added or removed from the default definition or R Markdown. See the `rmarkdown_format` for additional details.
- `pandoc_args` | Additional command line options to pass to pandoc

Details

See the online documentation for additional details on using the `rtf_document` format.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

Value

R Markdown output format to pass to `render`

Examples

```r
## Not run:

library(rmarkdown)

# simple invocation
render("input.Rmd", rtf_document())

# specify table of contents option
```
run

```
render("input.Rmd", rtf_document(toc = TRUE))
```

## End(Not run)

---

### run

**Run a Shiny document**

**Description**

Start a Shiny server for the given document, and render it for display.

**Usage**

```
run(file = "index.Rmd", dir = dirname(file), default_file = NULL,
    auto_reload = TRUE, shiny_args = NULL, render_args = NULL)
```

**Arguments**

- **file**: Path to the R Markdown document to launch in a web browser. Defaults to `index.Rmd` in the current working directory, but may be `NULL` to skip launching a browser.
- **dir**: The directory from which to read input documents. Defaults to the parent directory of `file`.
- **default_file**: The file to serve at the Shiny server’s root URL. If `NULL` (the default), a sensible default is chosen (see Details).
- **auto_reload**: If `TRUE` (the default), automatically reload the Shiny application when the file currently being viewed is changed on disk.
- **shiny_args**: Additional arguments to `runApp`.
- **render_args**: Additional arguments to `render`.

**Details**

The `run` function runs a Shiny document by starting a Shiny server associated with the document. The `shiny_args` parameter can be used to configure the server; see the `runApp` documentation for details.

Once the server is started, the document will be rendered using `render`. The server will initiate a render of the document whenever necessary, so it is not necessary to call `run` every time the document changes: if `auto_reload` is `TRUE`, saving the document will trigger a render. You can also manually trigger a render by reloading the document in a Web browser.

The server will render any R Markdown (.Rmd) document in `dir`; the `file` argument specifies only the initial document to be rendered and viewed. You can therefore link to other documents in the directory using standard Markdown syntax, e.g. `[Analysis Page 2](page2.Rmd)`.

If `default_file` is not specified, nor is a file specified on the URL, then the default document to serve at `/` is chosen from (in order of preference):

- If `dir` contains only one Rmd, that Rmd.
• The file `index.Rmd`, if it exists in `dir`
• The file `index.html`, if it exists in `dir`

If you wish to share R code between your documents, place it in a file named `global.R` in `dir`; it will be sourced into the global environment.

Value

Invisible NULL.

Note

Unlike `render`, `run` does not render the document to a file on disk. In most cases a Web browser will be started automatically to view the document; see `launch.browser` in the `runApp` documentation for details.

When using an external web browser with the server, specify the name of the R Markdown file to view in the URL (e.g. `http://127.0.0.1:1234/`foo.Rmd`). A URL without a filename will show the `default_file` as described above.

Examples

```r
## Not run:
# Run the Shiny document "index.Rmd" in the current directory
rmarkdown::run()

# Run the Shiny document "shiny_doc.Rmd" on port 8241
rmarkdown::run("shiny_doc.Rmd", shiny_args = list(port = 8241))

## End(Not run)
```

---

**shiny_prerendered_chunk**

*Add code to a shiny_prerendered context*

Description

Programmatic equivalent to including a code chunk with a context in a runtime: `shiny_prerendered` document.

Usage

`shiny_prerendered_chunk(context, code, singleton = FALSE)`

Arguments

- `context`  
  Context name (e.g. "server", "server-start")
- `code`  
  Character vector with code
- `singleton`  
  Collapse multiple identical versions of this chunk into a single chunk.
shiny_prerendered_clean

Clean prerendered content for the specified Rmd input file

Description

Remove the associated html file and supporting _files directory for a shiny_prerendered document.

Usage

shiny_prerendered_clean(input)

Arguments

input Rmd input file to clean content for

site_resources

Determine website resource files for a directory

Description

Determine which files within a given directory should be copied in order to serve a website from the directory. Attempts to automatically exclude source, data, hidden, and other files not required to serve website content.

Usage

site_resources(site_dir, include = NULL, exclude = NULL, recursive = FALSE)

Arguments

site_dir Site directory to analyze
include Additional files to include (glob wildcards supported)
exclude Files to exclude (glob wildcards supported)
recursive ‘TRUE’ to return a full recursive file listing; ‘FALSE’ to just provide top-level files and directories.

Value

Character vector of files and directories to copy
slidy_presentation  

Convert to a slidy presentation

Description

Format for converting from R Markdown to a slidy presentation.

Usage

```
slidy_presentation(incremental = FALSE, duration = NULL, footer = NULL,
                  font_adjustment = 0, fig_width = 8, fig_height = 6, fig_retina = 2,
                  fig_caption = TRUE, dev = "png", df_print = "default", smart = TRUE,
                  self_contained = TRUE, highlight = "default", mathjax = "default",
                  template = "default", css = NULL, includes = NULL, keep_md = FALSE,
                  lib_dir = NULL, md_extensions = NULL, pandoc_args = NULL,
                  extra_dependencies = NULL, ...)
```

Arguments

- **incremental**  
  TRUE to render slide bullets incrementally. Note that if you want to reverse the default incremental behavior for an individual bullet you can precede it with >. For example: > - Bullet Text
- **duration**  
  Duration (in minutes) of the slide deck. This value is used to add a countdown timer to the slide footer.
- **footer**  
  Footer text (e.g. organization name and/or copyright)
- **font_adjustment**  
  Increase or decrease the default font size (e.g. -1 or +1). You can also manually adjust the font size during the presentation using the 'S' (smaller) and 'B' (bigger) keys.
- **fig_width**  
  Default width (in inches) for figures
- **fig_height**  
  Default width (in inches) for figures
- **fig_retina**  
  Scaling to perform for retina displays (defaults to 2, which currently works for all widely used retina displays). Set to NULL to prevent retina scaling. Note that this will always be NULL when keep_md is specified (this is because fig_retina relies on outputting HTML directly into the markdown document).
- **fig_caption**  
  TRUE to render figures with captions
- **dev**  
  Graphics device to use for figure output (defaults to pdf)
- **df_print**  
  Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses `print.data.frame`. The "kable" method uses the `knitr::kable` function. The "tibble" method uses the `tibble` package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the df_print behavior entirely by setting the option `rmarkdown.df_print` to FALSE.
slidy_presentation

- **smart**: Produce typographically correct output, converting straight quotes to curly quotes, — to em-dashes, – to en-dashes, and ... to ellipses.

- **self_contained**: Produce a standalone HTML file with no external dependencies, using data: URLs to incorporate the contents of linked scripts, stylesheets, images, and videos. Note that even for self contained documents MathJax is still loaded externally (this is necessary because of its size).

- **highlight**: Syntax highlighting style. Supported styles include "default", "tango", "pygments", "kate", "monochrome", "espresso", "zenburn", and "haddock". Pass NULL to prevent syntax highlighting.

- **mathjax**: Include mathjax. The "default" option uses an https URL from a MathJax CDN. The "local" option uses a local version of MathJax (which is copied into the output directory). You can pass an alternate URL or pass NULL to exclude MathJax entirely.

- **template**: Pandoc template to use for rendering. Pass "default" to use the rmarkdown package default template; pass NULL to use pandoc’s built-in template; pass a path to use a custom template that you’ve created. See the documentation on pandoc online documentation for details on creating custom templates.

- **css**: One or more css files to include

- **includes**: Named list of additional content to include within the document (typically created using the includes function).

- **keep_md**: Keep the markdown file generated by knitting.

- **lib_dir**: Directory to copy dependent HTML libraries (e.g. jquery, bootstrap, etc.) into. By default this will be the name of the document with _files appended to it.

- **md_extensions**: Markdown extensions to be added or removed from the default definition or R Markdown. See the rmarkdown_format for additional details.

- **pandoc_args**: Additional command line options to pass to pandoc

- **extra_dependencies**: A \LaTeX\ dependency latex dependency(), a list of \LaTeX\ dependencies, a character vector of \LaTeX\ package names (e.g. c("framed", "hyperref")), or a named list of \LaTeX\ package options with the names being package names (e.g. list(hyperref = c("unicode=true", "breaklinks=true"), \pmodern = NULL)). It can be used to add custom \LaTeX\ packages to the .\text\ header.

- **...**: Additional function arguments to pass to the base R Markdown HTML output formatter html_document_base

**Details**

See the online documentation for additional details on using the slidy_presentation format.

For more information on markdown syntax for presentations see the pandoc online documentation.

**Value**

R Markdown output format to pass to render
Examples

```r
## Not run:
library(rmarkdown)

# simple invocation
render("pres.Rmd", slidy_presentation())

# specify an option for incremental rendering
render("pres.Rmd", slidy_presentation(incremental = TRUE))

## End(Not run)
```

**tufte_handout**  
*Tufte handout format (PDF)*

**Description**

Template for creating a handout according to the style of Edward R. Tufte and Richard Feynman.

**Usage**

```
tufte_handout(fig_width = 4, fig_height = 2.5, fig_crop = TRUE, 
dev = "pdf", highlight = "default", keep_tex = FALSE, 
citation_package = c("none", "natbib", "biblatex"), includes = NULL, 
md_extensions = NULL, pandoc_args = NULL)
```

**Arguments**

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>fig_width</code></td>
<td>Default width (in inches) for figures</td>
</tr>
<tr>
<td><code>fig_height</code></td>
<td>Default width (in inches) for figures</td>
</tr>
<tr>
<td><code>fig_crop</code></td>
<td>TRUE to automatically apply the pdfcrop utility (if available) to pdf figures</td>
</tr>
<tr>
<td><code>dev</code></td>
<td>Graphics device to use for figure output (defaults to pdf)</td>
</tr>
<tr>
<td><code>keep_tex</code></td>
<td>Keep the intermediate tex file used in the conversion to PDF</td>
</tr>
<tr>
<td><code>citation_package</code></td>
<td>The LaTeX package to process citations, natbib or biblatex. Use none if neither package is to be used.</td>
</tr>
<tr>
<td><code>includes</code></td>
<td>Named list of additional content to include within the document (typically created using the includes function).</td>
</tr>
<tr>
<td><code>md_extensions</code></td>
<td>Markdown extensions to be added or removed from the default definition or R Markdown. See the rmarkdown_format for additional details.</td>
</tr>
<tr>
<td><code>pandoc_args</code></td>
<td>Additional command line options to pass to pandoc</td>
</tr>
</tbody>
</table>
Details

See the online documentation for additional details.

Creating Tufte handout output from R Markdown requires that LaTeX be installed.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

Description

Format for converting from R Markdown to an MS Word document.

Usage

word_document(toc = FALSE, toc_depth = 3, fig_width = 5, fig_height = 4, fig_caption = TRUE, df_print = "default", smart = TRUE, highlight = "default", reference_docx = "default", keep_md = FALSE, md_extensions = NULL, pandoc_args = NULL)

Arguments

toc TRUE to include a table of contents in the output
toc_depth Depth of headers to include in table of contents
fig_width Default width (in inches) for figures
fig_height Default width (in inches) for figures
fig_caption TRUE to render figures with captions
df_print Method to be used for printing data frames. Valid values include "default", "kable", "tibble", and "paged". The "default" method uses print.data.frame. The "kable" method uses the knitr::kable function. The "tibble" method uses the tibble package to print a summary of the data frame. The "paged" method creates a paginated HTML table (note that this method is only valid for formats that produce HTML). In addition to the named methods you can also pass an arbitrary function to be used for printing data frames. You can disable the df_print behavior entirely by setting the option rmarkdown.df_print to FALSE.
smart Produce typographically correct output, converting straight quotes to curly quotes, — to em-dashes, – to en-dashes, and ... to ellipses.
reference_docx  Use the specified file as a style reference in producing a docx file. For best results, the reference docx should be a modified version of a docx file produced using pandoc. Pass "default" to use the rmarkdown default styles.

keep_md  Keep the markdown file generated by knitting.

md_extensions  Markdown extensions to be added or removed from the default definition or R Markdown. See the rmarkdown_format for additional details.

pandoc_args  Additional command line options to pass to pandoc

Details

See the online documentation for additional details on using the word_document format.

R Markdown documents can have optional metadata that is used to generate a document header that includes the title, author, and date. For more details see the documentation on R Markdown metadata.

R Markdown documents also support citations. You can find more information on the markdown syntax for citations in the Bibliographies and Citations article in the online documentation.

Value

R Markdown output format to pass to render

Examples

```r
## Not run:
library(rmarkdown)

# simple invocation
render("input.Rmd", word_document())

# specify an option for syntax highlighting
render("input.Rmd", word_document(highlight = "zenburn"))

## End(Not run)
```
Index

*Topic datasets
  metadata, 35

all_output_formats, 4
beamer_presentation, 4, 5, 59
clean_site (render_site), 54
compile_notebook, 7
Compiling R scripts to a notebook, 51
default_output_format, 8
default_site (render_site), 54
default_site_generator (render_site), 54
draft, 9
file, 4, 8, 33, 51, 54, 57
find_external_resources, 10
from_rmarkdown (rmarkdown_format), 58
github_document, 11
html-dependencies, 12
html_dependency_bootstrap (html-dependencies), 12
html_dependency_font_awesome (html-dependencies), 12
html_dependency_highlightjs (html-dependencies), 12
html_dependency_ionicons (html-dependencies), 12
html_dependency_jquery (html-dependencies), 12
html_dependency_jqueryui (html-dependencies), 12
html_dependency_pagedtable (html-dependencies), 12
html_dependency_tocify (html-dependencies), 12
html_document, 4, 13, 18, 19, 23, 52, 56
html_document_base, 15, 17, 21, 26, 65
html_fragment, 18
html_notebook, 20
html_notebook_metadata, 21
html_notebook_output, 21, 22
html_notebook_output_code (html_notebook_output), 22
html_notebook_output_html (html_notebook_output), 22
html_notebook_output_img (html_notebook_output), 22
html_notebook_output_png (html_notebook_output), 22
html_vignette, 23
includes, 6, 12, 15, 19, 21, 24, 26, 34, 36, 47, 65, 66
includes_to_pandoc_args (includes), 24
ioslides_presentation, 25
knit, 50, 52
knit_hooks, 31
knit::kable, 5, 12, 14, 19, 23, 25, 34, 37, 46, 49, 64, 67
knitr_options, 31, 32, 37, 38
knitr_options_html, 31
knitr_options_pdf, 32
latex_dependency, 33
latex_document (pdf_document), 46
latex_fragment (pdf_document), 46
md_document, 34
message, 56
metadata, 6, 15, 35, 36, 47, 60, 67, 68
new.env, 51, 54
numeric_version, 41
odt_document, 36
on.exit, 38
opts_chunk, 31
opts_hooks, 31
opts_knit, 31
opts_template, 31
output_format, 17, 22, 31, 32, 37, 44, 52, 59
output_metadata, 38
paged_table, 39
pandoc_args, 39
pandoc_available, 40
pandoc_citeproc_convert, 41
pandoc_convert, 42
pandoc_highlight_args (pandoc_args), 39
pandoc_include_args (pandoc_args), 39
pandoc_latex_engine_args (pandoc_args), 39
pandoc_options, 37, 38, 43, 58, 59
pandoc_path_arg, 44
pandoc_self_contained_html, 44
pandoc_template, 45
pandoc_toc_args (pandoc_args), 39
pandoc_variable_arg (pandoc_args), 39
pandoc_version (pandoc_available), 40
parse_html_notebook, 45
path.expand, 44
pdf_document, 4, 46, 52, 59
powerpoint_presentation, 48
relative_to, 49
render, 4, 6–8, 12, 15, 20, 24, 26, 35, 36, 38,
        48, 49, 50, 57, 58, 60–62, 65, 68
render_delayed, 52
render_site, 54
render_supporting_files, 37, 57
resolve_output_format, 57
rmarkdown (rmarkdown-package), 3
rmarkdown-package, 3
rmarkdown_format, 6, 12, 15, 19, 21, 26, 34,
        36, 44, 47, 49, 52, 58, 60, 65, 66, 68
rmd_metadata, 59
rtf_document, 60
run, 51, 61
runApp, 33, 61, 62
shiny_prerendered_chunk, 62
shiny_prerendered_clean, 63
site_config (render_site), 54
site_generator (render_site), 54
site_resources, 63
slidy_presentation, 64
tufte_handout, 66
word_document, 4, 67