

# Package ‘downlit’

July 5, 2022

**Title** Syntax Highlighting and Automatic Linking

**Version** 0.4.2

**Description** Syntax highlighting of R code, specifically designed for the needs of 'RMarkdown' packages like 'pkgdown', 'hugodown', and 'bookdown'. It includes linking of function calls to their documentation on the web, and automatic translation of ANSI escapes in output to the equivalent HTML.

**License** MIT + file LICENSE

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

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

**Depends** R (>= 3.4.0)

**Imports** brio, desc, digest, evaluate, fansi, memoise, rlang, vctrs, withr, yaml

**Suggests** covr, htmltools, jsonlite, MASS, MassSpecWavelet, pkgload, rmarkdown, testthat (>= 3.0.0), xml2

**Config/Needs/website** tidyverse/tidytemplate

**Config/testthat/edition** 3

**Encoding** UTF-8

**RoxygenNote** 7.2.0

**NeedsCompilation** no

**Author** Hadley Wickham [aut, cre],  
RStudio [cph, fnd]

**Maintainer** Hadley Wickham <hadley@rstudio.com>

**Repository** CRAN

**Date/Publication** 2022-07-05 03:00:02 UTC

## R topics documented:

autolink . . . . .	2
downlit_html_path . . . . .	3
downlit_md_path . . . . .	4
evaluate_and_highlight . . . . .	5
highlight . . . . .	6
is_low_change . . . . .	7

<b>Index</b>	<b>8</b>
--------------	----------

---

autolink	<i>Automatically link inline code</i>
----------	---------------------------------------

---

### Description

Automatically link inline code

### Usage

```
autolink(text)
```

```
autolink_url(text)
```

### Arguments

text                      String of code to highlight and link.

### Value

If text is linkable, an HTML link for `autolink()`, and or just the URL for `autolink_url()`. Both return NA if the text is not linkable.

### Options

`downlit` provides a number of options to control the details of the linking. They are particularly important if you want to generate "local" links.

- `downlit.package`: name of the current package. Determines when `topic_index` and `article_index`
- `downlit.topic_index` and `downlit.article_index`: named character vector that maps from topic/article name to path.
- `downlit.rdbname`: name of current Rd file being documented (if any); used to avoid self-links.
- `downlit.attached`: character vector of currently attached R packages.
- `downlit.local_packages`: named character vector providing relative paths (value) to packages (name) that can be reached with relative links from the target HTML document.
- `downlit.topic_path` and `downlit.article_path`: paths to reference topics and articles/vignettes relative to the "current" file.

**Examples**

```
autolink("stats::median()")
autolink("vignette('grid', package = 'grid')")

autolink_url("stats::median()")
```

---

downlit\_html\_path      *Syntax highlight and link an HTML page*

---

**Description**

- Code blocks, identified by `<pre>` tags with class `sourceCode` or any `<pre>` tag inside of `<div class='downlit'>`, are processed with `highlight()`.
- Inline code, identified by `<code>` tags that contain only text (and don't have a header tag (e.g. `<h1>`) or `<a>` as an ancestor) are processed with `autolink()`.

Use `downlit_html_path()` to process an `.html` file on disk; use `downlit_html_node()` to process an in-memory `xml_node` as part of a larger pipeline.

**Usage**

```
downlit_html_path(in_path, out_path, classes = classes_pandoc())

downlit_html_node(x, classes = classes_pandoc())
```

**Arguments**

<code>in_path</code> , <code>out_path</code>	Input and output paths for HTML file
<code>classes</code>	A mapping between token names and CSS class names. Bundled <code>classes_pandoc()</code> and <code>classes_chroma()</code> provide mappings that (roughly) match Pandoc and chroma (used by hugo) classes so you can use existing themes.
<code>x</code>	An <code>xml2::xml_node</code>

**Value**

`downlit_html_path()` invisibly returns `output_path`; `downlit_html_node()` modifies `x` in place and returns nothing.

**Examples**

```
node <- xml2::read_xml("<p><code>base::t()</code></p>")
node

# node is modified in place
downlit_html_node(node)
node
```

---

downlit\_md\_path      *Syntax highlight and link a md document*

---

### Description

downlit\_md\_\* works by traversing the markdown AST generated by Pandoc. It applies `highlight()` to CodeBlocks and `autolink()` to inline Code.

Use `downlit_md_path()` to transform a file on disk; use `downlit_md_string()` to transform a string containing markdown as part of a larger pipeline.

Needs pandoc 1.19 or later.

### Usage

```
downlit_md_path(in_path, out_path, format = NULL)
```

```
downlit_md_string(x, format = NULL)
```

### Arguments

in_path, out_path	Input and output paths for markdown file.
format	Pandoc format; defaults to "gfm" if you have pandoc 2.0.0 or greater, otherwise "markdown_github".
x	A string containing markdown.

### Value

`downlit_md_path()` invisibly returns `output_path`; `downlit_md_string()` returns a string containing markdown.

### Examples

```
if (rmarkdown::pandoc_available("1.19")) {
  downlit_md_string("`base::t`")
  downlit_md_string("`base::t`")
  downlit_md_string("* `base::t`")

  # But don't highlight in headings
  downlit_md_string("## `base::t`")
}
```

---

 evaluate\_and\_highlight

*Evaluate code and syntax highlight the results*


---

## Description

This function runs code and captures the output using `evaluate::evaluate()`. It syntax highlights code with `highlight()`, and intermingles it with output.

## Usage

```
evaluate_and_highlight(
  code,
  fig_save,
  classes = downlit::classes_pandoc(),
  env = NULL,
  output_handler = evaluate::new_output_handler(),
  highlight = TRUE
)
```

## Arguments

<code>code</code>	Code to evaluate (as a string).
<code>fig_save</code>	A function with arguments <code>plot</code> and <code>id</code> that is responsible for saving plot to a file (using <code>id</code> to disambiguate multiple plots in the same chunk). It should return a list with components <code>path</code> , <code>width</code> , and <code>height</code> .
<code>classes</code>	A mapping between token names and CSS class names. Bundled <code>classes_pandoc()</code> and <code>classes_chroma()</code> provide mappings that (roughly) match Pandoc and chroma (used by hugo) classes so you can use existing themes.
<code>env</code>	Environment in which to evaluate code; if not supplied, defaults to a child of the global environment.
<code>output_handler</code>	Custom output handler for <code>evaluate::evaluate()</code> .
<code>highlight</code>	Optionally suppress highlighting. This is useful for tests.

## Value

An string containing HTML with a `dependencies` attribute giving an additional `htmltools` dependencies required to render the HTML.

## Examples

```
cat(evaluate_and_highlight("1 + 2"))
cat(evaluate_and_highlight("x <- 1:10\nmean(x)"))

# -----
# evaluate_and_highlight() powers pkgdown's documentation formatting so
```

```
# here I include a few examples to make sure everything looks good
# -----

blue <- function(x) paste0("\033[34m", x, "\033[39m")
f <- function(x) {
  cat("This is some output. My favourite colour is ", blue("blue"), ".\n", sep = "")
  message("This is a message. My favourite fruit is ", blue("blueberries"))
  warning("Now at stage ", blue("blue"), "!")
}
f()

plot(1:10)
```

---

highlight

*Highlight and link a code block*


---

## Description

This function:

- syntax highlights code
- links function calls to their documentation (where possible)
- in comments, translates ANSI escapes in to HTML equivalents.

## Usage

```
highlight(text, classes = classes_chroma(), pre_class = NULL, code = FALSE)
```

```
classes_pandoc()
```

```
classes_chroma()
```

## Arguments

text	String of code to highlight and link.
classes	A mapping between token names and CSS class names. Bundled <code>classes_pandoc()</code> and <code>classes_chroma()</code> provide mappings that (roughly) match Pandoc and chroma (used by hugo) classes so you can use existing themes.
pre_class	Class(es) to give output <code>&lt;pre&gt;</code> .
code	If TRUE, wrap output in a <code>&lt;code&gt;</code> block

## Value

If text is valid R code, an HTML `<pre>` tag. Otherwise, NA.

A string containing syntax highlighted HTML or NA (if text isn't parseable).

## Options

downlit provides a number of options to control the details of the linking. They are particularly important if you want to generate "local" links.

- `downlit.package`: name of the current package. Determines when `topic_index` and `article_index`
- `downlit.topic_index` and `downlit.article_index`: named character vector that maps from topic/article name to path.
- `downlit.rdbname`: name of current Rd file being documented (if any); used to avoid self-links.
- `downlit.attached`: character vector of currently attached R packages.
- `downlit.local_packages`: named character vector providing relative paths (value) to packages (name) that can be reached with relative links from the target HTML document.
- `downlit.topic_path` and `downlit.article_path`: paths to reference topics and articles/vignettes relative to the "current" file.

## Examples

```
cat(highlight("1 + 1"))
cat(highlight("base::t(1:3)"))

# Unparseable R code returns NA
cat(highlight("base::t("))
```

---

is_low_change	<i>Compare two recorded plots</i>
---------------	-----------------------------------

---

## Description

Compare two recorded plots

## Usage

```
is_low_change(p1, p2)
```

## Arguments

p1, p2                  Plot results

## Value

Logical value indicating whether p2 is a low-level update of p1.

# Index

autolink, 2  
autolink(), 3, 4  
autolink\_url (autolink), 2

classes\_chroma (highlight), 6  
classes\_pandoc (highlight), 6

downlit\_html\_node (downlit\_html\_path), 3  
downlit\_html\_path, 3  
downlit\_md\_path, 4  
downlit\_md\_string (downlit\_md\_path), 4

evaluate::evaluate(), 5  
evaluate\_and\_highlight, 5

highlight, 6  
highlight(), 3-5

is\_low\_change, 7