Package 'downlit'

September 25, 2020

Title Syntax Highlighting and Automatic Linking

Version 0.2.0

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

Depends R (>= 3.2.0)

Imports brio, digest, evaluate, fansi, rlang, vctrs, yaml

Suggests rmarkdown, jsonlite, MASS, testthat, covr, pkgload, xml2

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

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

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

Config/testthat/edition 3

NeedsCompilation no

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

Maintainer Hadley Wickham <hadley@rstudio.com>

Repository CRAN

Date/Publication 2020-09-25 05:00:10 UTC

R topics documented:

`autolink` *Automatically link inline code*

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.

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` and `r`, 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 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

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

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 combines all results into a single HTML div.

Usage

```
evaluate_and_highlight(
  code,
  fig_save,
  classes = downlit::classes_pandoc(),
  env = NULL
)
```

Arguments

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

Value

An string containing HTML.

Examples

```
evaluate_and_highlight("1 + 2")
```

`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)

classes_pandoc()

classes_chroma()
```

Arguments

`text`	String of code to highlight and link.
`classes`	A mapping between token names and CSS class names. Bundled `classes_pandoc()` and `classes_chroma()` 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 <pre>.

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).

Examples

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

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

Index