Package 'memoiR'

September 3, 2021

Title R Markdown and Bookdown Templates to Publish Documents

Version 1.1-2

URL https://github.com/EricMarcon/memoiR

BugReports https://github.com/EricMarcon/memoiR/issues

Description Producing high-quality documents suitable for publication directly from R is made possible by the R Markdown ecosystem.
'memoiR' makes it easy.
It provides templates to knit memoirs, articles and slideshows with helpers to publish the documents on GitHub Pages and activate continuous integration.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.1.1

SystemRequirements pandoc

VignetteBuilder knitr

Imports bookdown, rmarkdown, usethis

Suggests knitr, pkgdown, rmdformats, testthat

NeedsCompilation no

Author Eric Marcon [aut, cre] (<https://orcid.org/0000-0002-5249-321X>)

Maintainer Eric Marcon <eric.marcon@agroparistech.fr>

Repository CRAN

Date/Publication 2021-09-03 15:00:02 UTC

R topics documented:

`build_ghworkflow` *Build GitHub Action Workflow*

Description

Build a YAML file (`.github/workflows/memoir.yml`) to knit the documents of the project to GitHub Pages. The workflow knits all R Markdown files according their header: all output formats are produced and stored into the `gh-pages` branch of the project.

Usage

```
build_ghworkflow()
```

Details

All HTML outputs have the same name so the last one knitted overwrites the previous ones. Keep only one HTML format in the header of each RMarkdown file.

No `DESCRIPTION` file is necessary in the project to install packages. They must be declared in the options code chunk of each .Rmd file (index.Rmd for the memoir template).

Two secrets must have been stored in the GitHub account:

- GH_PAT: a valid access token,
- EMAIL: the email address to send the workflow results to.

Value

The content of the YAML file as a vector of characters, invisibly. Each element is a line of the file.

Examples

```
## Simulate the creation of a new project
# Save working directory
original_wd <- getwd()
# Get a temporary working directory
wd <- tempfile("example")
# Simulate File > New File > R Markdown... > From Template > Simple Article
rmarkdown::draft(wd, template="simple_article", package="memoiR", edit=FALSE)
# Go to temp directory
setwd(wd)
# Make it the current project
usethis::proj_set(path = ".", force = TRUE)

# Build GitHub Actions Workflow script
build_ghworkflow()
# Content
readLines(".github/workflows/memoir.yml")

## End of the example: cleanup
```

```
# Return to the original working directory and clean up
setwd(original_wd)
unlink(wd, recursive=TRUE)
```

`build_githubpages` *Build GitHub Pages*

Description

Copy the files produced by knitting to the destination folder.

Usage

```
build_githubpages(destination = usethis::proj_path("docs"))
```

Arguments

`destination`	destination folder of the knitted documents.

Details

Produced files are HTML pages and their companions (css, figures, libraries) and PDF documents. The function moves them all and the `README.md` file into the destination folder. GitHub Pages allow making a website to present them:

- `README.md` is the home page. Make it with `build_readme()` to have links to the HTML and PDF outputs.
- knit both HTML and PDF versions to avoid dead links.
- run `build_githubpages()` when a document is knitted to move the outputs into the `docs` folder.
- push to GitHub and activate GitHub Pages on the main branch and the `docs` folder. The function is useless in book projects: the *Build the Book* (i.e. the `bookdown::render_book()` function) takes care of every step.

Value

A vector with the names of the files and directory that were copied if they existed (some may not be knitted), invisibly.

Examples

```
## Not run:
## Simulate the creation of a new project
# Save working directory
original_wd <- getwd()
# Get a temporary working directory
wd <- tempfile("example")
```

```
# Simulate File > New File > R Markdown... > From Template > Simple Article
rmarkdown::draft(wd, template="simple_article", package="memoiR", edit=FALSE)
# Go to temp directory
setwd(wd)
# Make it the current project
usethis::proj_set(path = ".", force = TRUE)

## Sequence of actions to build a complete project
# Build .gitignore
build_gitignore()
## Activate source control, edit your files, commit
# Build README, link to HTML output only in this example
build_readme(PDF=FALSE)
# render: knit to HTML Document (interactively: click the Knit button)
rmarkdown::render(input=list.files(pattern="*.Rmd"),
                  output_format="bookdown::html_document2")
# Build GitHub Pages
build_githubpages()
# List the GitHub Pages files
setwd("docs")
list.files(recursive=TRUE)
## Commit and push. Outputs will be in /docs of the master branch.

## End of the example: cleanup
# Return to the original working directory and clean up
setwd(original_wd)
unlink(wd, recursive=TRUE)

## End(Not run)
```

`build_gitignore` *Build .gitignore*

Description

Build a `.gitignore` file suitable for R Markdown projects.

Usage

```
build_gitignore()
```

Details

The .gitignore file contains the list of files (file name patterns) that must not be controlled by git. Run this function once in each project created from a memoiR template, before activating version control.

Value

The content of the `.gitignore` file as a vector of characters, invisibly. Each element is a line of the file.

Examples

```
## Simulate the creation of a new project
# Save working directory
original_wd <- getwd()
# Get a temporary working directory
wd <- tempfile("example")
# Simulate File > New File > R Markdown... > From Template > Simple Article
rmarkdown::draft(wd, template="simple_article", package="memoiR", edit=FALSE)
# Go to temp directory
setwd(wd)
# Make it the current project
usethis::proj_set(path = ".", force = TRUE)

# Build .gitignore file
build_gitignore()
# Content
readLines(".gitignore")

## End of the example: cleanup
# Return to the original working directory and clean up
setwd(original_wd)
unlink(wd, recursive=TRUE)
```

`build_readme` *Build README*

Description

Build a `README.md` file that will be used as index of GitHub Pages.

Usage

```
build_readme(PDF = TRUE)
```

Arguments

`PDF`	if `TRUE` (by default), a link to the PDF output is added.

Details

R Markdown files of the project are used to get the title and abstract of the published documents. Run this function once in each project created from a memoiR template, before `build_githubpages()`. A link to their HTML and, optionally, PDF versions is added. Metadata fields are read in the .Rmd files YAML header: title, abstract and `URL`.

Value

The content of the `README.md` file as a vector of characters, invisibly. Each element is a line of the file.

Examples

```
## Simulate the creation of a new project
# Save working directory
original_wd <- getwd()
# Get a temporary working directory
wd <- tempfile("example")
# Simulate File > New File > R Markdown... > From Template > Simple Article
rmarkdown::draft(wd, template="simple_article", package="memoiR", edit=FALSE)
# Go to temp directory
setwd(wd)
# Make it the current project
usethis::proj_set(path = ".", force = TRUE)

# Build README.md file
build_readme()
# Content
readLines("README.md")

## End of the example: cleanup
# Return to the original working directory and clean up
setwd(original_wd)
unlink(wd, recursive=TRUE)
```

Knit *Knit*

Description

Create documents from templates

Usage

```
knit_all(destination = usethis::proj_path("docs"), gallery = "gallery")

knit_template(
  template,
  output_format,
  destination = usethis::proj_path("docs"),
  gallery = "gallery"
)
```

Arguments

destination	name of the folder containing GitHub pages or equivalent.
gallery	name of the subfolder of destination to store the knitted documents.
template	name of the template to knit, e.g. "simple_article".
output_format	A character vector of the output formats to convert to. Each value must be the name of a function producing an output format object, such as "bookdown::pdf_book".

Details

These functions are used to test the templates and produce a gallery.

- `knit_template()` produces an HTML and a PDF output of the chosen template.
- `knit_all()` runs knit_template() on all templates of the package. The `output_format` argument selects the way templates are rendered:
- articles may be rendered in HTML by bookdown::html_document2, bookdown::gitbook, rmdformats::downcute (and others, see the package **rmdformats**) and in PDF by bookdown::pdf_book.
- books may be rendered in HTML by bookdown::gitbook and in PDF by bookdown::pdf_book.
- slides may be rendered in HTML by bookdown::ioslides_presentation2, bookdown::ioslides_presentation2 and in PDF by bookdown::beamer_presentation2.

These functions are mainly used for test and documentation purposes. In projects based on the templates, use the *Knit* button (articles, presentations) or the *Build the Book* button (memoirs) or `bookdown::render_book()`.

Value

`TRUE` if all documents have been knitted and copied to the gallery, invisibly.

`memoiR` *Package memoiR*

Description

R Bookdown templates to publish documents, especially relying on the memoir LaTeX package

Index