Package 'froggeR'

July 8, 2025

Type Package

Title Enhance 'Quarto' Project Workflows and Standards

Version 0.5.0

Maintainer Kyle Grealis <kyleGrealis@icloud.com>

Description Streamlines 'Quarto' workflows by providing tools for consistent project setup and documentation. Enables portability through reusable metadata, automated project structure creation, and standardized templates. Features include enhanced project initialization, preformatted 'Quarto' documents, inclusion of 'Quarto' brand functionality, comprehensive data protection settings, custom styling, and structured documentation generation. Designed to improve efficiency and collaboration in R data science projects by reducing repetitive setup tasks while maintaining consistent formatting across multiple documents.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.5.0)

Imports cli (>= 3.0.0), fs, glue (>= 1.6.0), here (>= 1.0.1), quarto (>= 1.3.0), rappdirs, readr (>= 2.0.0), rstudioapi, stringr (>= 1.5.0), usethis (>= 2.2.0), yaml

Suggests knitr, mockery, rmarkdown, testthat (>= 3.0.0), withr

Config/build/copy-resources inst/gists/basic_quarto.qmd inst/gists/custom_quarto.qmd inst/gists/README.md

Config/testthat/edition 3

VignetteBuilder knitr

URL https://azimuth-project.tech/froggeR/

BugReports https://github.com/kyleGrealis/froggeR/issues

NeedsCompilation no

Author Kyle Grealis [aut, cre] (ORCID: <https://orcid.org/0000-0002-9223-8854>)

Repository CRAN

Date/Publication 2025-07-08 16:00:02 UTC

Contents

`quarto_project` *Create a Custom 'Quarto' Project*

Description

This function creates a new 'Quarto' project directory with additional froggeR features. It first calls `quarto::quarto_create_project()` to set up the basic structure, then enhances it with froggeR-specific files and settings.

Usage

```
quarto_project(name, path = here::here(), example = TRUE)
```

Arguments

`name`	Character string. The name of the 'Quarto' project directory and initial `.qmd` file.
`path`	Character string. Path to the project directory.
`example`	Logical. If TRUE (default), creates a Quarto document with a default to position the brand logo and examples of within-document cross-referencing, links, and references.

Details

This function creates a 'Quarto' project with the following enhancements:

- `_brand.yml`: Stores Quarto project branding
- `logos`: Quarto project branding logos directory
- `_variables.yml`: Stores reusable YAML variables
- `.gitignore`: Enhanced settings for R projects
- `README.md`: Template README file

- `dated_progress_notes.md`: For project progress tracking
- `custom.scss`: Custom 'Quarto' styling

If froggeR settings don't exist, it will prompt to create them.

Value

Invisibly returns the path to the created project directory.

See Also

`write_quarto`, `settings`

Examples

```
if (interactive() && quarto::quarto_version() >= "1.6") {

  # Create the Quarto project with custom YAML & associated files
  quarto_project("frogs", path = tempdir(), example = TRUE)

  # Confirms files were created (optional, for user confirmation)
  file.exists(file.path(tempdir(), "frogs.qmd"))     # Quarto doc
  file.exists(file.path(tempdir(), "_quarto.yml"))  # project YAML file

  # Create a new Quarto project with standard Quarto (no examples)
  # quarto_project('frogs_standard', path = tempdir(), example = FALSE)

}
```

---

save_brand *Save brand YAML for 'Quarto' Projects*

Description

This function saves the current '_brand.yml' file from an existing froggeR Quarto project. This provides a safety catch to prevent unintended overwrite if the system-level configuration file exists.

Usage

```
save_brand(save_logos = TRUE)
```

Arguments

`save_logos`	Logical. Save brand logos from 'logos' directory. Default is 'TRUE'.

Details

The function will attempt to create a system-level configuration file from the current froggeR Quarto project. If the system-level configuration file already exists, the user will be prompted prior to overwrite.

Value

Invisibly returns ‘NULL‘ after creating system-level configuration file.

Examples

```
# Write the _brand.yml file
if (interactive()) save_brand()
```

`save_variables` *Save variables YAML for 'Quarto' Projects*

Description

This function saves the current ‘_variables.yml‘ file from an existing froggeR Quarto project. This provides a safety catch to prevent unintended overwrite if the system-level configuration file exists.

Usage

```
save_variables()
```

Details

The function will attempt to create a system-level configuration file from the current froggeR Quarto project. If the system-level configuration file already exists, the user will be prompted prior to overwrite.

Value

Invisibly returns ‘NULL‘ after creating system-level configuration file.

Examples

```
# Write the _variables.yml file
if (interactive()) save_variables()
```

settings *Manage 'froggeR' Settings*

Description

This function provides useful instructions and information regarding ‘froggeR‘ settings.

Usage

```
settings()
```

Details

This function can only run in an interactive environment. Choose to: - Check for the existence of project- and global-level configuration files - Display current project-level settings - Feedback for updating settings - Instructions how to reuse settings across multiple ‘froggeR‘ Quarto projects.

Value

No return value; called for side-effects.

Examples

```
# Only run in an interactive environment
if (interactive()) froggeR::settings()
```

write_brand *Write Brand YAML for 'Quarto' Projects*

Description

This function creates or updates the ‘_brand.yml‘ file in a Quarto project directory if they exist in the config path.

Usage

```
write_brand(
  path = here::here(),
  restore_logos = TRUE,
  .initialize_proj = FALSE
)
```

Arguments

`path`	Character string. Path to the project directory.
`restore_logos`	Logical. Restore logo content from system configuration. Default is 'TRUE'.
`.initialize_proj`	Logical. TRUE only if starting a `froggeR::quarto_project()`.

Details

The function will attempt to use the current froggeR settings from the config path. If no global configurations exist, a template '_brand.yml' will be created.

Value

Invisibly returns 'NULL' after creating or updating the '_brand.yml' file.

Examples

```
# Write the _brand.yml file
if (interactive()) {
  temp_dir <- tempdir()
  write_brand(temp_dir)
}
```

`write_ignore` *Create an enhanced* `.gitignore` *file*

Description

This function creates a `.gitignore` file with enhanced security measures designed to help prevent accidental data leaks. The template includes comprehensive rules for common data file types and sensitive information.

Usage

```
write_ignore(path = here::here(), .initialize_proj = FALSE)
```

Arguments

`path`	Character string. Path to the project directory.
`.initialize_proj`	Logical. TRUE only if starting a `froggeR::quarto_project()`.

Details

If a `.gitignore` file already exists, the user will be prompted before any overwriting occurs. The template provides substantial security enhancements over basic `.gitignore` files.

WARNING: Always consult your organization's data security team before using git with any sensitive or protected health information (PHI). This template helps prevent accidental data exposure but should not be considered a complete security solution.

Value

A `.gitignore` file with enhanced security rules. The file includes:
* R data files (.RData, .rda, .rds)
* Common data formats (CSV, Excel, text)
* System and temporary files
* IDE-specific files

Examples

```
# Create a temporary directory for testing
tmp_dir <- tempdir()

# Write the .gitignore file
write_ignore(path = tmp_dir)

# Confirm the file was created (optional, for user confirmation)
file.exists(file.path(tmp_dir, ".gitignore"))

# Clean up: Remove the created file
unlink(file.path(tmp_dir, ".gitignore"))
```

`write_notes` *Create a project README file*

Description

This function streamlines project documentation by creating a dated progress notes file.

Usage

```
write_notes(path = here::here(), .initialize_proj = FALSE)
```

Arguments

`path`	Character string. Path to the project directory.
`.initialize_proj`	Logical. TRUE only if starting a `froggeR::quarto_project()`.

Details

The dated_progress_notes.md file is initialized with the current date and is designed to help track project milestones chronologically. If the progress notes file already exists, the function will stop and warn the user.

Value

CA chronological project progress notes tracker.

Examples

```
# Create a temporary directory for testing
tmp_dir <- tempdir()

# Write the progress notes file
write_notes(path = tmp_dir)

# Confirm the file was created (optional, for user confirmation)
file.exists(file.path(tmp_dir, "dated_progress_notes.md"))

# Clean up: Remove the created file
unlink(file.path(tmp_dir, "dated_progress_notes.md"))
```

---

write_quarto *Create a New 'Quarto' Document*

Description

This function creates a new 'Quarto' document (.qmd file) with either a custom or standard YAML header. When using a custom header, it integrates with 'froggeR::settings()' for reusable metadata across documents.

Usage

```
write_quarto(
  filename = "Untitled-1",
  path = here::here(),
  example = FALSE,
  .initialize_proj = FALSE
)
```

Arguments

filename	Character string. The name of the file without the '.qmd' extension. Only letters, numbers, hyphens, and underscores are allowed.
path	Character string. Path to the project directory.

example Logical. If TRUE, creates a Quarto document with a default to position the brand logo and examples of within-document cross-referencing, links, and references.

.initialize_proj
Logical. TRUE only if starting a froggeR::quarto_project().

Value

Invisibly returns NULL after creating the 'Quarto' document.

Examples

```
if (interactive()) {
  # Create a temporary directory for testing
  tmp_dir <- tempdir()

  # Write the Quarto & associated files for a custom YAML with reusable metadata
  write_quarto(path = tempdir(), filename = "analysis")

  # Write the Quarto file with a template requiring more DIY
  write_quarto(path = tempdir(), filename = "analysis_basic", example = FALSE)

  # Confirm the file was created (optional, for user confirmation)
  file.exists(file.path(tmp_dir, "analysis.qmd"))
  file.exists(file.path(tmp_dir, "analysis_basic.qmd"))

  # Clean up: Remove the created file
  unlink(list.files(tempdir(), full.names = TRUE), recursive = TRUE)
}
```

---

write_readme *Create a project README file*

Description

This function streamlines project documentation by a README.md file.

Usage

```
write_readme(path = here::here(), .initialize_proj = FALSE)
```

Arguments

path Character string. Path to the project directory.

.initialize_proj
Logical. TRUE only if starting a froggeR::quarto_project().

Details

The README.md template includes structured sections for:

- Project description (study name, principal investigator, author)
- Project setup steps for reproducibility
- File and directory descriptions
- Miscellaneous project notes

Value

Creates a comprehensive README template for project documentation.

Examples

```
# Create a temporary directory for testing
tmp_dir <- tempdir()

# Write the README file
write_readme(path = tmp_dir)

# Confirm the file was created (optional, for user confirmation)
file.exists(file.path(tmp_dir, "README.md"))

# Clean up: Remove the created file
unlink(file.path(tmp_dir, "README.md"))
```

---

write_scss *Create a 'Quarto' SCSS file*

Description

This function creates the `.scss` file so that any 'Quarto' project can be easily customized with SCSS styling variables, mixins, and rules.

Usage

```
write_scss(path = here::here(), .initialize_proj = FALSE)
```

Arguments

`path`	Character string. Path to the project directory.
`.initialize_proj`	Logical. TRUE only if starting a `froggeR::quarto_project()`.

Details

The function includes a robust YAML handling mechanism that safely adds new SCSS file.

See `vignette("customizing-quarto", package = "froggeR")` vignette for more help.

Value

A SCSS file to customize 'Quarto' styling.

Examples

```
# Create a temporary directory for testing
tmp_dir <- tempdir()

# Write the SCSS file
write_scss(path = tmp_dir)

# Confirm the file was created (optional, for user confirmation)
file.exists(file.path(tmp_dir, "custom.scss"))

# Clean up: Remove the created file
unlink(file.path(tmp_dir, "custom.scss"))
```

---

write_variables *Write Variables YAML for 'Quarto' Projects*

Description

This function creates or updates the '_variables.yml' file in a Quarto project directory using froggeR settings, if they exist in the config path.

Usage

```
write_variables(path = here::here(), .initialize_proj = FALSE)
```

Arguments

`path`	Character string. Path to the project directory.
`.initialize_proj`	Logical. TRUE only if starting a `froggeR::quarto_project()`.

Details

The function will attempt to use the current froggeR settings from the config path. If no global configurations exist, a template '_variables.yml' will be created.

Value

Invisibly returns 'NULL' after creating or updating the '_variables.yml' file.

Examples

```
# Write the _variables.yml file
if (interactive()) {
  temp_dir <- tempdir()
  write_variables(temp_dir)
}
```

Index