Package 'froggeR'

March 17, 2026

Type Package

Title Project Scaffolding for R and 'Quarto'

Version 1.0.1

Description Creates structured R and 'Quarto' projects with a consistent directory layout: scripts in R/, analysis documents in analysis/, and web assets in www/. The primary entry point, init(), downloads the latest template from a companion 'GitHub' repository so that project structure evolves independently of package releases. Supports persistent author metadata and 'Quarto' brand configuration that carry across projects automatically.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 3.5.0)

Imports cli (>= 3.0.0), fs, here (>= 1.0.1), rappdirs, rlang, rstudioapi, usethis (>= 2.2.0)

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

Config/testthat/edition 3

VignetteBuilder knitr

URL https://www.kyleGrealis.com/froggeR/, https://github.com/kyleGrealis/froggeR

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

NeedsCompilation no

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

Maintainer Kyle Grealis <kyleGrealis@proton.me>

Repository CRAN

Date/Publication 2026-03-17 08:10:08 UTC

Contents

`init` *Initialize a froggeR Project from the Template*

Description

Downloads the latest project scaffold from the frogger-templates repository and restores any saved user configuration. This is the recommended way to start a new froggeR project.

Usage

```
init(path = here::here())
```

Arguments

`path`	Character. Directory where the project will be created. If the directory does not exist, it will be created. Default is current project root via `here`.

Details

The function performs these steps:

1. Creates the target directory if it does not exist
2. Downloads the latest template zip from GitHub
3. Copies only files that do not already exist (never overwrites)
4. Restores saved user config (`_variables.yml`, `_brand.yml`, `logos/`) from `~/.config/froggeR/` if present
5. Creates a `data/` directory (gitignored by default)

Existing files are never overwritten. Each created and skipped file is reported individually so you can see exactly what changed.

Global configuration is saved via `save_variables` and `save_brand`. If no saved config exists, the template defaults are used as-is.

Value

Invisibly returns the normalized `path`.

See Also

`write_variables`, `write_brand`, `save_variables`, `save_brand`

Examples

```
## Not run:
# Create a new project (directory is created automatically)
init(path = file.path(tempdir(), "my-project"))

## End(Not run)
```

`quarto_project` *Create a Custom Quarto Project*

Description

Deprecated. Use `init()` instead.

Usage

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

Arguments

`name`	Character. Ignored.
`path`	Character. Ignored.
`example`	Logical. Ignored.

Value

Does not return. Always errors with a deprecation message.

See Also

`init`

Examples

```
## Not run:
# Use init() instead
init(path = "my-project")

## End(Not run)
```

save_brand *Save Brand Configuration to Global froggeR Settings*

Description

This function saves the current `_brand.yml` file from an existing froggeR Quarto project to your global (system-wide) froggeR configuration. This allows you to reuse brand settings across multiple projects.

Usage

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

Arguments

`save_logos`	Logical. Should logo files from the `logos` directory also be saved to global configuration? Default is `TRUE`.

Details

This function:

- Reads the project-level `_brand.yml` file
- Saves it to your system-wide froggeR config directory
- Optionally copies the `logos` directory for reuse in future projects
- Prompts for confirmation if a global configuration already exists

The saved configuration is stored in `rappdirs::user_config_dir('froggeR')` and will automatically be used in new froggeR projects created with `init` or `write_brand`.

Value

Invisibly returns `NULL` after saving configuration file.

See Also

`write_brand`, `save_variables`

Examples

```
# Save brand settings from current project to global config
if (interactive()) save_brand()

# Save brand settings but skip logos
if (interactive()) save_brand(save_logos = FALSE)
```

save_variables *Save Metadata Configuration to Global froggeR Settings*

Description

This function saves the current `_variables.yml` file from an existing froggeR Quarto project to your global (system-wide) froggeR configuration. This allows you to reuse metadata across multiple projects.

Usage

```
save_variables()
```

Details

This function:

- Reads the project-level `_variables.yml` file
- Saves it to your system-wide froggeR config directory
- Prompts for confirmation if a global configuration already exists

The saved configuration is stored in `rappdirs::user_config_dir('froggeR')` and will automatically be used in new froggeR projects created with `init` or `write_variables`.

This is useful for maintaining consistent author metadata (name, email, affiliations, etc.) across all your projects without having to re-enter it each time.

Value

Invisibly returns `NULL` after saving configuration file.

See Also

`write_variables`, `save_brand`

Examples

```
# Save metadata from current project to global config
if (interactive()) save_variables()
```

`write_brand` *Write Brand YAML for Quarto Projects*

Description

Creates or opens a `_brand.yml` file in a Quarto project for editing. If the file already exists, it is opened directly. If global froggeR brand settings exist, those are used as the starting point. Otherwise, the template is downloaded from the frogger-templates repository.

Usage

```
write_brand(path = here::here(), restore_logos = TRUE)
```

Arguments

`path`	Character. Path to the project directory. Default is current project root via `here`.
`restore_logos`	Logical. Restore logo content from system configuration. Default is `TRUE`.

Details

The `_brand.yml` file defines your visual identity for Quarto documents: colors, logos, typography, and more. See the brand.yml specification for the full list of available options.

Use `save_brand` to persist your project-level brand configuration to the global config directory for reuse across projects.

Value

Invisibly returns the path to the file.

See Also

`save_brand`, `write_variables`, `init`

Examples

```
## Not run:
write_brand()

## End(Not run)
```

write_ignore *Create a .gitignore File*

Description

Creates or opens a `.gitignore` file in a Quarto project. If the file already exists, it is opened directly. Otherwise, the opinionated template is downloaded from the frogger-templates repository.

Usage

```
write_ignore(path = here::here())
```

Arguments

`path`	Character. Path to the project directory. Default is current project root via `here`.

Value

Invisibly returns the path to the file.

See Also

`init`, `write_quarto`

Examples

```
## Not run:
write_ignore()

## End(Not run)
```

write_quarto *Create a New Quarto Document*

Description

Downloads the Quarto template from the frogger-templates repository and writes it to the `analysis/` directory. Errors if a file with the same name already exists.

Usage

```
write_quarto(filename = "Untitled-1", path = here::here())
```

Arguments

filename	Character. The name of the file without the .qmd extension. Only letters, numbers, hyphens, and underscores are allowed. Default is "Untitled-1".
path	Character. Path to the project directory. Default is current project root via here.

Details

The file is written to analysis/<filename>.qmd. The analysis/ directory is created automatically if it does not exist.

Value

Invisibly returns the path to the created Quarto document.

See Also

init, write_variables, write_brand

Examples

```
## Not run:
# Create a Quarto document with the default name
write_quarto()

# Create a Quarto document with a custom name
write_quarto(filename = "analysis")

## End(Not run)
```

---

write_scss *Create a Quarto SCSS File*

Description

Creates or opens an SCSS file in the www/ directory. If the file already exists, it is opened directly. Otherwise, the template is downloaded from the frogger-templates repository.

Usage

```
write_scss(filename = "custom", path = here::here())
```

Arguments

filename	Character. The name of the file without the .scss extension. A www/ prefix and .scss extension are stripped automatically if provided, so "custom2" and "www/custom2.scss" are equivalent. Only letters, numbers, hyphens, and underscores are allowed. Default is "custom".
path	Character. Path to the project directory. Default is current working directory via here.

Details

The file is written to `www/<filename>.scss`. The `www/` directory is created automatically if it does not exist.

Value

Invisibly returns the path to the file.

See Also

`init`, `write_quarto`

Examples

```
## Not run:
# Create the default custom.scss
write_scss()

# Create a second SCSS file
write_scss("custom2")
# These are equivalent
write_scss("www/custom2.scss")

## End(Not run)
```

`write_variables` *Write Variables YAML for Quarto Projects*

Description

Creates or opens a `_variables.yml` file in a Quarto project for editing. If the file already exists, it is opened directly. If global froggeR settings exist, those are used as the starting point. Otherwise, the template is downloaded from the frogger-templates repository.

Usage

```
write_variables(path = here::here())
```

Arguments

`path`	Character. Path to the project directory. Default is current project root via `here`.

Details

The `_variables.yml` file stores reusable author metadata that Quarto documents can reference. Available fields:

- `name`: Your full name as it appears in publications
- `email`: Contact email address
- `orcid`: ORCID identifier (e.g., 0000-0001-2345-6789)
- `url`: Personal website or profile URL
- `github`: GitHub username
- `affiliations`: Institution, department, etc.

Use `save_variables` to persist your project-level metadata to the global config directory for reuse across projects.

Value

Invisibly returns the path to the file.

See Also

`save_variables`, `write_brand`, `init`

Examples

```
## Not run:
write_variables()

## End(Not run)
```

Index