

Package ‘logrx’

July 22, 2025

Title A Logging Utility Focus on Clinical Trial Programming Workflows

Version 0.4.0

Description A utility to facilitate the logging and review of R programs in clinical trial programming workflows.

License MIT + file LICENSE

URL <https://pharmaverse.github.io/logrx/>,
<https://github.com/pharmaverse/logrx>

BugReports <https://github.com/pharmaverse/logrx/issues>

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

Imports dplyr (>= 1.0.0), magrittr, purrr, rlang, stats, tidyr,
stringr, sessioninfo (>= 1.2), stringi, tibble, digest,
lifecycle, methods

Suggests testthat (>= 3.0.0), knitr, rmarkdown, withr, covr, pkgdown,
Tplyr, haven, lintr (>= 3.1.1), xml2, here, readr, rstudioapi,
tidyselect, renv, yaml

VignetteBuilder knitr

Config/testthat/edition 3

Depends R (>= 4.0.0)

NeedsCompilation no

Author Nathan Kosiba [aut, cre],
Thomas Bermudez [aut],
Ben Straub [aut],
Michael Rimler [aut],
Nicholas Masel [aut],
Sam Parmar [aut],
GSK/Atorus JPT [cph, fnd]

Maintainer Nathan Kosiba <nghkosiba@gmail.com>

Repository CRAN

Date/Publication 2025-05-05 13:50:02 UTC

Contents

approved	2
axecute	3
build_approved	4
log_config	5
log_init	6
log_remove	6
log_write	7
read_log_file	8
write_log_header	9
write_unapproved_functions	9
write_used_functions	10
Index	11

approved	<i>Approved packages and functions</i>
----------	--

Description

A dataset that stores approved packages and functions for use. Each row contains a library and function_name. This dataset is used to illustrate the data format to be stored in the log.rx.approved option.

Usage

approved

Format

A tibble with 6 rows and 2 variables:

function_name Name of the function

library Name of the package

Examples

logrx::approved

axecute

*Creation of a log and axecution of a file***Description**

axecute() creates a log, executes a file, and returns 0 if there are no errors or 1 if there are any errors

Usage

```
axecute(
  file,
  log_name = NA,
  log_path = NA,
  include_rds = FALSE,
  quit_on_error = TRUE,
  to_report = c("messages", "output", "result"),
  show_repo_url = FALSE,
  extra_info = NA,
  ...
)
```

Arguments

file	String. Path to file to execute
log_name	String. Name of log file
log_path	String. Path to log file
include_rds	Boolean. Option to export log object as Rds file. Defaults to FALSE
quit_on_error	Boolean. Should the session quit with status 1 on error? Defaults to TRUE
to_report	String vector. Objects to optionally report, may include as many as necessary: • messages: any messages generated by program execution • output: any output generated by program execution • result: any result generated by program execution
show_repo_url	Boolean. Should the repository URLs be reported Defaults to FALSE
extra_info	List. Objects to optionally add on to end of log in a special extra info section. List printed in YAML format. Optional
...	Not used

Value

0 if there are no errors or 1 if there are any errors

Examples

```

dir <- tempdir()
text <- 'print("Hello, logrx-person!")'
fileConn <- file(file.path(dir, "hello.R"))
writeLines(text, fileConn)
close(fileConn)

axecute(file.path(dir, "hello.R"))

fileConn <- file(file.path(dir, "hello.Rmd"))
writeLines(text, fileConn)
close(fileConn)

axecute(file.path(dir, "hello.Rmd"))

```

build_approved	<i>Build approved packages and functions tibble</i>
----------------	---

Description

A utility function to help you build your approved packages and functions list. This can be used by logrx to log unapproved use of packages and functions.

Usage

```
build_approved(pkg_list, file = NULL)
```

Arguments

pkg_list	Named list of character vectors: • Name is the package name • Value is a character vector of approved functions or 'All'
file	String. Name of file where the approved tibble will be written to. If not specified, the tibble is returned Default: NULL Permitted Files: .RDS

Details

For more details see the vignette: `vignette("approved", package = "logrx")`

Value

Tibble with two columns (library, function) and one row per function

Examples

```

approved_pkgs <- list(
  base = c("library", "mean"),
  dplyr = "All"
)

# build and return
build_approved(approved_pkgs)

# build and save
dir <- tempdir()
build_approved(approved_pkgs, file.path(dir, "approved.rds"))

```

log_config

*Configuration of the log.rx environment***Description**

log_config() initialises the log.rx environment, adds its attributes, and sets them

Usage

```
log_config(file = NA, log_name = NA, log_path = NA, extra_info = NA)
```

Arguments

file	String. Path to file executed. Optional
log_name	String. Name of log file. Optional
log_path	String. Path to log file. Optional
extra_info	List. Objects to add on to end of log in a special extra info section. Optional

Value

Nothing

Examples

```

dir <- tempdir()
text <- 'print("Hello, Timberperson!")'
fileConn <- file(file.path(dir, "hello.R"))
writeLines(text, fileConn)
close(fileConn)

file <- file.path(dir, "hello.R")

# Initialise and configure the log.rx environment
log_config(file)

```

```
# Run the script and record results, outputs, messages, errors, and warnings
logrx::run_safely_loudly(file)

# Write the log
log_write(file)
```

log_init	<i>Initialization of the log.rx environment</i>
----------	---

Description

log_init() initialises the log.rx environment

Usage

```
log_init()
```

Value

Nothing

Examples

```
# Initialise the log.rx environment
log_init()

# Remove the log.rx environment
log_remove()
```

log_remove	<i>log.rx object removal</i>
------------	------------------------------

Description

log_remove() removes the log.rx object by setting options("log.rx") to NULL

Usage

```
log_remove()
```

Value

Nothing

Examples

```
# Initialise the log.rx environment
log_init()

# Remove the log.rx environment
log_remove()
```

log_write

*Formatting and writing of the log.rx object to a log file***Description**

log_write() gets and formats the content of the log.rx before writing it to a log file

Usage

```
log_write(
  file = NA,
  remove_log_object = TRUE,
  show_repo_url = FALSE,
  include_rds = FALSE,
  extra_info = NA,
  to_report = c("messages", "output", "result")
)
```

Arguments

file	String. Path to file executed
remove_log_object	Boolean. Should the log object be removed after writing the log file? Defaults to TRUE
show_repo_url	Boolean. Should the repo URLs be reported Defaults to FALSE
include_rds	Boolean. Option to export log object as Rds file. Defaults to FALSE
extra_info	List. Objects to add on to end of log in a special extra info section. Optional
to_report	String vector. Objects to optionally report; additional information in axecute

Value

Nothing

Examples

```

dir <- tempdir()
text <- 'print("Hello, Timberperson!")'
fileConn <- file(file.path(dir, "hello.R"))
writeLines(text, fileConn)
close(fileConn)

file <- file.path(dir, "hello.R")

# Initialise and configure the log.rx environment
log_config(file)

# Run the script and record results, outputs, messages, errors, and warnings
logrx::run_safely_loudly(file)

# Write the log
log_write(file)

```

read_log_file	<i>Read and parse logrx file</i>
---------------	----------------------------------

Description

Read and parse logrx file

Usage

```
read_log_file(file)
```

Arguments

file String. Path to a logrx log file

Value

Tibble. Object that includes nested and parsed content

Examples

```

## Not run:
read_log_file(previous_log_filepath)

## End(Not run)

```

write_log_header	<i>Formatting of log file section headers</i>
------------------	---

Description

write_log_header formats a string and returns it as a formatted log file section header

Usage

```
write_log_header(title_string)
```

Arguments

title_string String. Used as section title

Value

Vector of strings. Formatted log file section header

Examples

```
## Not run:  
write_log_header("Section Header")  
  
## End(Not run)
```

write_unapproved_functions	<i>Formats and returns a vector of unapproved functions</i>
----------------------------	---

Description

write_unapproved_functions() gets log.rx unapproved_packages_functions attribute, formats and returns the list of unapproved functions

Usage

```
write_unapproved_functions()
```

Value

Formatted vector of unapproved functions

Examples

```
## Not run:  
write_unapproved_functions()  
  
## End(Not run)
```

write_used_functions *Formats and returns a vector of used package functions*

Description

write_used_functions() gets log.rx used_packages_functions attribute, formats and returns the list of used package functions

Usage

```
write_used_functions()
```

Value

Formatted vector of used package functions

Examples

```
## Not run:  
write_used_functions()  
  
## End(Not run)
```

Index

* **datasets**

approved, [2](#)

approved, [2](#)

axecute, [3](#), [7](#)

build_approved, [4](#)

log_config, [5](#)

log_init, [6](#)

log_remove, [6](#)

log_write, [7](#)

read_log_file, [8](#)

write_log_header, [9](#)

write_unapproved_functions, [9](#)

write_used_functions, [10](#)