Package: logrittr (via r-universe)

June 2, 2026

Title A Verbose Pipe Operator for Logging dplyr Pipelines

Version 0.2.0

Maintainer Guillaume Pressiat <guillaume.pressiat@gmail.com>

Description Provides the `%>=%` pipe operator that logs row counts, column counts, added/dropped columns, and timing at each step of a dplyr pipeline, inspired by the SAS DATA step log. Supports nested pipelines and configurable options for display width, number formatting, and language.

License GPL-2

URL https://guillaumepressiat.github.io/logrittr/

BugReports https://github.com/GuillaumePressiat/logrittr/issues

Encoding UTF-8

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Imports cli (>= 3.0.0), stringr (>= 1.4.0), magrittr

Suggests dplyr, knitr, rmarkdown, lumberjack, R6, testthat (>= 3.0.0)

Config/testthat/edition 3

VignetteBuilder knitr

Config/pak/sysreqs libicu-dev

Repository https://guillaumepressiat.r-universe.dev

Date/Publication 2026-05-03 08:54:11 UTC

RemoteUrl https://github.com/GuillaumePressiat/logrittr

RemoteRef HEAD

RemoteSha 031bf9f364302b4275e56b36769993d2cabebd9e

Contents

`%>=%` *Logging pipe operator*

Description

A drop-in replacement for `%>%` that logs row counts, column counts, added/dropped column names, and step timing at each stage of a dplyr / tidyr pipeline. Inspired by the SAS DATA step log and mostly for educational context in R.

Nested pipelines (e.g. inside a `semi_join()` or `filter()` argument) are automatically detected and displayed with increasing indentation, so the main pipeline and its sub-pipelines are visually distinct.

Usage

```
lhs %>=% rhs
```

Arguments

`lhs`	A data frame (or tibble) passed as the left-hand side.
`rhs`	An unevaluated dplyr-style function call.

Details

Depth tracking uses `options(.LPipe_depth)` which is incremented around the evaluation of `rhs` only, ensuring that the steps of the *main* pipeline always log at depth 0 and nested `%>=%` calls at depth 1, 2, etc.

Display options (`wrap_width`, `big_mark`, `lang`) are controlled via `logrittr_options()`.

Value

The result of applying `rhs` to `lhs`, invisibly from the logging perspective — the value is returned normally so pipelines compose as expected.

See Also

`logrittr_options()`

Examples

```
## Not run:
library(dplyr)

logrittr_options(lang = "en", big_mark = ",", wrap_width = 30)

iris %>=%
  as_tibble() %>=%
  filter(Sepal.Length < 5) %>=%
  mutate(rn = row_number()) %>=%
  group_by(Species) %>=%
  summarise(n = n_distinct(rn))

## End(Not run)
```

`logrittr_activate` *Activate logrittr logging on the %>% pipe*

Description

Replaces %>% in the global environment with %>=% so that existing pipelines using %>% are automatically logged without any code change.

Call `logrittr_deactivate()` to restore the original %>% behaviour.

Usage

```
logrittr_activate()
```

Value

Invisibly returns the previous definition of %>% in the global environment (or NULL if none existed).

See Also

`logrittr_deactivate()`, `logrittr_hook()`

Examples

```
## Not run:
library(logrittr)
library(dplyr)

logrittr_activate()

iris %>%
  filter(Sepal.Length < 5) %>%
  group_by(Species) %>%
```

```
  summarise(n = n())

logrittr_deactivate()

## End(Not run)
```

`logrittr_deactivate` *Deactivate logrittr logging on the %>% pipe*

Description

Restores %>% in the global environment to its original definition (magrittr's pipe if available, otherwise removes the binding).

Usage

```
logrittr_deactivate()
```

Value

Invisibly returns NULL.

See Also

logrittr_activate(), logrittr_hook()

Examples

```
## Not run:
logrittr_activate()
# ... work ...
logrittr_deactivate()

## End(Not run)
```

`logrittr_hook` *knitr source hook for pipe logging*

Description

Registers a knitr source hook that enables logrittr logging in R Markdown and Quarto documents, with no change to pipeline code.

- "native" (default): rewrites |> to %>=% in chunks where the chunk option logrittr = TRUE is set.
- "magrittr": calls logrittr_activate() so %>% logs globally.
- "both": does both of the above.

Usage

```
logrittr_hook(pipe = c("native", "magrittr", "both"))
```

Arguments

pipe	Character. Which pipe(s) to intercept. One of "native", "magrittr", or "both". Default: "native".

Value

Invisibly returns NULL.

See Also

logrittr_activate(), logrittr_deactivate()

Examples

```
## Not run:
# In your setup chunk:
library(logrittr)

knitr::opts_chunk$set(
collapse  = TRUE,
comment   = "#>",
message   = TRUE   # needed to show logrittr output (uses message())
)

# For |> pipes (opt-in per chunk with logrittr = TRUE):
logrittr_hook()

# For %>% pipes (global):
logrittr_hook("magrittr")

# For both:
logrittr_hook("both")

# Then in any chunk you want logged (native pipe):
# ```{r, logrittr = TRUE}
# iris |>
#   filter(Sepal.Length < 5) |>
#   group_by(Species) |>
#   summarise(n = n())
# ```

## End(Not run)
```

`logrittr_logger` *logrittr logger for lumberjack*

Description

An R6 logger that plugs into the lumberjack `%L>%` pipe and renders the same console output as `%>=%`: row/column counts, signed deltas, added/dropped column names, and approximate step timing.

Requires the `R6` and `lumberjack` packages (both in `Suggests`). If either is missing, the class is unavailable but the rest of logrittr works normally.

Timing

lumberjack calls `$add()` after each step without providing a start time. Elapsed time is measured as the interval between two consecutive `$add()` calls. The first step always shows `NA ms`.

Public fields

`label` Set by lumberjack to the name of the tracked object.

Methods

Public methods:

- `logrittr_logger$new()`
- `logrittr_logger$add()`
- `logrittr_logger$dump()`
- `logrittr_logger$clone()`

Method `new()`**:** Create a new `logrittr_logger`.

Usage:

```
logrittr_logger$new(verbose = TRUE, src_name = NULL)
```

Arguments:

`verbose` Logical. Whether to print log messages to the console. Default `TRUE`.

`src_name` Character. Optional name of the source object, displayed as a header rule before the first step.

Method `add()`**:** Called by lumberjack after each pipe step.

Usage:

```
logrittr_logger$add(meta, input, output)
```

Arguments:

`meta` List with elements `expr` and `src` (the step expression).

`input` Data frame before the step.

`output` Data frame after the step.

Method `dump()`**:** Called by `dump_log()`. Writes accumulated log to a CSV file. If `verbose = TRUE`, also prints the file path.

Usage:

```
logrittr_logger$dump(file = NULL, ...)
```

Arguments:

`file` Character. Output file path. Defaults to `"simple.csv"` or `"<label>_simple.csv"` when a label is set.

`...` Additional arguments passed to `write.csv()`.

Method `clone()`**:** The objects of this class are cloneable with this method.

Usage:

```
logrittr_logger$clone(deep = FALSE)
```

Arguments:

`deep` Whether to make a deep clone.

See Also

`logrittr_options()`, `%>=%()`, pipe_log

Examples

```
## Not run:
library(lumberjack)
library(dplyr)

iris %L>%
  start_log(log = logrittr_logger$new()) %L>%
  as_tibble() %L>%
  filter(Sepal.Length < 5) %L>%
  mutate(rn = row_number()) %L>%
  group_by(Species) %L>%
  summarise(n = n_distinct(rn)) %L>%
  dump_log(stop = TRUE)


logfile <- tempfile(fileext="r.log.csv")

iris %L>%
  start_log(log = logrittr_logger$new(verbose = FALSE,
  label = "A reel simple example on iris df")) %L>%
  as_tibble()  %L>%
  filter(Sepal.Length < 5) %L>%
  mutate(rn = row_number()) %L>%
  group_by(Species) %L>%
  summarise(n = n_distinct(rn)) %L>%
  dump_log(file=logfile)

logdata <- read.csv(logfile)

head(logdata)
```

```
## End(Not run)
```

`logrittr_options` *Get or set logrittr global options*

Description

Configure the behaviour of the `%>=%` pipe. Call with named arguments to set options, or with no arguments to return current values.

Usage

```
logrittr_options(
  wrap_width = NULL,
  big_mark = NULL,
  lang = NULL,
  max_cols = NULL
)
```

Arguments

`wrap_width`	Integer. Maximum width (in characters) of the step label before it wraps to the next line. Default: `32`.
`big_mark`	Character. Thousands separator used when formatting counts. Default: `" "` (thin space). Use `","` for English convention or `""` to disable.
`lang`	Character. Display language for the metrics line. One of `"en"` (default) or `"fr"`.
`max_cols`	Integer. Maximum number of column names to display in the `added`/`dropped` lines. If there are more, the first `max_cols` are shown followed by `"and N others"`. Default: `5`. Use `Inf` to always show all columns.

Value

Invisibly returns the previous values of any option that was changed, as a named list. When called with no arguments, returns all current options as a named list (visibly).

See Also

`%>=%()`, pipe_log

Examples

```
# French defaults
logrittr_options()

# Switch to English, comma thousands separator, show up to 3 col names
logrittr_options(lang = "en", big_mark = ",", max_cols = 3)

# Reset to defaults
logrittr_options(wrap_width = 32, big_mark = " ", lang = "en", max_cols = 5)
```

Index