Package 'rspdlite'

June 16, 2026

Type Package

Title R and C++ Interfaces to 'spdlite' C++ Header Library for Logging

Version 0.1.0-1

Date 2026-06-08

License GPL (>= 2)

Description The lightweight header-only C++-20 logging library 'spdlite', a lighter version of 'spdlog' and also written by Gabi Melman, provides most of the features of the larger version, and also includes 'fmt' as a fallback if std::format() is not selected.

URL https://github.com/eddelbuettel/rspdlite

BugReports https://github.com/eddelbuettel/rspdlite/issues

LinkingTo Rcpp

Imports Rcpp

Suggests tinytest

RoxygenNote 6.0.1

NeedsCompilation yes

Author Dirk Eddelbuettel [aut, cre] (ORCID:
<https://orcid.org/0000-0001-6419-907X>),
Gabi Melman [aut] (Author of spdlog),
Victor Zverovic [aut] (Author of fmt)

Maintainer Dirk Eddelbuettel <edd@debian.org>

Repository CRAN

Date/Publication 2026-06-16 19:30:02 UTC

Contents

`rspdlite-package` *rspdlite: Lightweight Logging From R and C++*

Description

rspdlite relies on **spdlite**, a lightweight header-only C++20 library for logging. This package offers a simple and consistent interface from both R and C++.

Details

spdlite is the 'little brother' of **spdlog** (which we have available via R packages **RcppSpdlog** and **spdl**). **spdlite** is on purpose smaller and simpler. It has (by choice) fewer options and configuration settings keeping the core small and simple.

By using a global `inline` instance, each shared library (i.e. typically each package using logging) is guaranteed to have exactly one instance. So setting changes such as the logging level affect both the R and C++ side. If however another R package were to be compiled with the **spdlite** header, its instance would be separate as it resides in a different shared library. Similarly, an ad-hoc compilation via e.g. `Rcpp::sourceCpp()` will lead to a distinct instance for the same reason.

Author(s)

Maintainer: Dirk Eddelbuettel <edd@debian.org> (0000-0001-6419-907X)

Authors:

- Gabi Melman (Author of spdlog)
- Victor Zverovic (Author of fmt)

See Also

Useful links:

- https://github.com/eddelbuettel/rspdlite
- Report bugs at https://github.com/eddelbuettel/rspdlite/issues

---

`log_trace` *Logging wrappers for 'spdlite' logging from both R and C++*

Description

Several wrappers for functions from 'spdlite' are provided as a convenience. In general, these can be accessed both from C++ (via the provided C++20 header), and from R via the functions documented here allowing for consitent logging throughout a package.

Usage

```
log_trace(s, ...)

log_debug(s, ...)

log_info(s, ...)

log_warn(s, ...)

log_error(s, ...)

log_critical(s, ...)

set_level(level)

get_level()

set_name(s)

get_name()

set_precision(precision)

show_thread_id(show_thread_id = TRUE)

show_date(show_date = TRUE)

show_utc(utc = TRUE)

set_format(utc = FALSE, show_date = TRUE, show_thread_id = FALSE,
  precision = "ms")
```

Arguments

s	Character value for filename, pattern, level, or logging message
...	Supplementary arguments for the logging string
level	Character value for the logging level
precision	Character value for selected time precision: one of "ms" (the default format), "us", "ns" or "none"
show_thread_id	Boolean flag select display of current thread, default is off
show_date	Boolean flag select display of date part of current, default is on
utc	Boolean flag select display of current time in UTC rather than local, default is off

Details

Logging functions respect a global logging level that defaults to 'info' (meaning that calls to `log_trace` or `log_debug` are ignored). Several formatting options are available as well to control the number of digits on the timestamp, whether or not the thread id is displayed, whether the date portion of the timestamp is to be displayed and whether the display is in local time (the default) or in UTC.

The C++ functionality is illustrated in the example file in the `examples` directory.

Value

Nothing is returned from these functions (with the exception of `get_level()`) as they are invoked for their side-effects.

Author(s)

Dirk Eddelbuettel

Examples

```
lvl <- rspdlite::get_level()
rspdlite::log_debug("This message is ignored by the default level 'info'.")
rspdlite::log_info("This message is show by the default level.")
rspdlite::set_level("warn")
rspdlite::log_info("Now this message at 'info' is ignored too.")
rspdlite::log_warn("A warning messages passes at level warning. {}", 42L)
rspdlite::set_name("my_logger")
rspdlite::log_error("Error messages also pass, and see the name set")
rspdlite::set_format(show_thread_id=TRUE, precision="ns")
rspdlite::log_error("Warning message under changed formatting")
rspdlite::set_level(lvl) # revert to prior level
rspdlite::set_name("") # revert to no name
rspdlite::set_format() # revert to default format
```

Index