Package: tl (via r-universe)

June 4, 2026

Type Package

Title Tiny logging interface to 'rspdlite' wrapping 'spdlite' C++20 logging

Version 0.0.1

Date 2026-05-31

Description Just how 'spdl' provides a nice and consistent interface to 'spdlog' (via 'RcppSpdlog'), this package does so for 'spdlite', the lightweight header-only C++-20 logging library, a lighter version of 'spdlog' and also written by Gabi Melman, providing most of the features of the larger version, and also including 'fmt' as fallback if 'std::format' is not selected. This package is essentially a thin shim around it for a more compact interface from both R and C++.

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

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

License GPL (>= 2)

Imports rspdlite

Suggests tinytest

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

Date/Publication 2026-06-04 10:43:29 UTC

RemoteUrl https://github.com/eddelbuettel/tl

RemoteRef HEAD

RemoteSha db5e8530013bc7226996e4cb4d3a379ef741d302

Contents

tl-package *Tiny logging interface to 'rspdlite' wrapping 'spdlite' C++20 logging*

Description

Just how 'spdl' provides a nice and consistent interface to 'spdlog' (via 'RcppSpdlog'), this package does so for 'spdlite', the lightweight header-only C++-20 logging library, a lighter version of 'spdlog' and also written by Gabi Melman, providing most of the features of the larger version, and also including 'fmt' as fallback if 'std::format' is not selected. This package is essentially a thin shim around it for a more compact interface from both R and C++.

Package Content

Index: This package was not yet installed at build time.

Maintainer

Dirk Eddelbuettel <edd@debian.org>

Author(s)

Dirk Eddelbuettel [aut, cre]

trace *Tiny logging wrapper for 'rspdlite'*

Description

These functions all pass on their arguments to the corresonding function in the **rspdlite** package implementing them. The core purpose of these functions is to provide a 'tighter' interface via the `tl::` prefix from both R and C++, i.e. `tl::debug("Condition met, value {}", val)` works from both. See the **rspdlite** package for more.

Usage

```
trace(...)

debug(...)

info(...)

warn(...)

error(...)
```

```
critical(...)

set_level(...)

get_level()

set_name(...)

get_name()

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

Arguments

...	Argument(s) passed along
utc	Boolean flag to select display of current time in UTC rather than local, default is off
show_date	Boolean flag to select display of date part of current, default is on
show_thread_id	Boolean flag to select display of current thread, default is off
precision	Character value for selected time precision: one of "ms" (the default format), "us", "ns" or "none"

Value

In general, nothing is returned as the functions are invoked for their side effect of logging.

See Also

rspdlite

Examples

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

Index