

Package ‘spdl’

July 23, 2025

Type Package

Title Easier Use of 'RcppSpdlog' Functions via Wrapper

Description Logging functions in 'RcppSpdlog' provide access to the logging functionality from the 'spdlog' 'C++' library. This package offers shorter convenience wrappers for the 'R' functions which match the 'C++' functions, namely via, say, 'spdl::debug()' at the debug level. The actual formatting is done by the 'fmt::format()' function from the 'fmtlib' library (that is also 'std::format()' in 'C++20' or later).

Version 0.0.5

Date 2023-06-18

License GPL (>= 2)

Imports RcppSpdlog (>= 0.0.13)

URL <https://github.com/eddelbuettel/spdl>

BugReports <https://github.com/eddelbuettel/spdl/issues>

RoxygenNote 6.0.1

NeedsCompilation no

Author Dirk Eddelbuettel [aut, cre]

Maintainer Dirk Eddelbuettel <edd@debian.org>

Repository CRAN

Date/Publication 2023-06-18 12:40:02 UTC

Contents

setup	2
Index	4

Description

Several short wrappers for functions from 'RcppSpdlog' package are provided as a convenience. Given the potential for clashing names of common and popular functions names we do *not* recommend the import the whole package but rather do `importFrom(RcppSpdlog, set_pattern)` (or maybe `importFrom(RcppSpdlog, set_pattern)`). After that, functionality can be accessed via a convenient shorter form such as for example `spdlog::info()` to log at the 'info' level. Format strings suitable for the C++ library 'fmtlib::fmt' and its `fmt::format()` (which as of C++20 becomes 'std::fmt') are supported so the `{}` is the placeholder for simple (scalar) arguments (for which the default R formatter is called before passing on a character representation).

Usage

```
setup(name = "default", level = "warn")

init(level = "warn")

log(level = "warn")

filesetup(s, name = "default", level = "warn")

drop(s)

set_pattern(s)

set_level(s)

trace(s, ...)

debug(s, ...)

info(s, ...)

warn(s, ...)

error(s, ...)

critical(s, ...)

fmt(s, ...)

cat(...)

stopwatch()
```

elapsed(w)

Arguments

name	Character value for the name of the logger instance
level	Character value for the logging level
s	Character value for filename, pattern, level, or logging message
...	Supplementary arguments for the logging string
w	Stopwatch object

Value

Nothing is returned from these functions as they are invoked for their side-effects.

Examples

```
spd1::setup("exampleDemo", "warn")
# and spd1::init("warn") and spd1::log("warn") are shortcuts
spd1::info("Not seen as level 'info' below 'warn'")
spd1::warn("This warning message is seen")
spd1::set_level("info")
spd1::info("Now this informational message is seen too")
spd1::info("Calls use fmtlib::fmt {} as we can see {}", "under the hood", 42L)
```

Index

`cat (setup)`, 2
`critical (setup)`, 2

`debug (setup)`, 2
`drop (setup)`, 2

`elapsed (setup)`, 2
`error (setup)`, 2

`filesetup (setup)`, 2
`fmt (setup)`, 2

`info (setup)`, 2
`init (setup)`, 2

`log (setup)`, 2

`set_level (setup)`, 2
`set_pattern (setup)`, 2
`setup`, 2
`stopwatch (setup)`, 2

`trace (setup)`, 2

`warn (setup)`, 2