

Package ‘dyn.log’

July 22, 2025

Type Package

Title Dynamic Logging for R Inspired by Configuration Driven Development

Version 0.4.0

Maintainer Brandon Moretz <bmoretz@ionicsolutions.net>

Description A comprehensive and dynamic configuration driven logging package for R. While there are several excellent logging solutions already in the R ecosystem, I always feel constrained in some way by each of them. Every project is designed differently to solve it's domain specific problem, and ultimately the utility of a logging solution is its ability to adapt to this design. This is the raison d'être for 'dyn.log': to provide a modular design, template mechanics and a configuration-based integration model, so that the logger can integrate deeply into your design, even though it knows nothing about it.

License MIT + file LICENSE

Encoding UTF-8

Depends R (>= 4.0)

Imports crayon (>= 1.4.1), glue (>= 1.4.2), R6 (>= 2.5.1), rlang (>= 0.4.12), stringr (>= 1.4.0), yaml (>= 2.2.1)

URL <https://bmoretz.github.io/dyn.log/>

BugReports <https://github.com/bmoretz/dyn.log/issues>

Suggests devtools, usethis, rmarkdown, markdown, knitr, covr (>= 3.5.1), testthat (>= 3.0.0), lintr (>= 0.28), remotes, pkgdown, prettydoc, here, fansi, pander, DT

Config/testthat/edition 3

RoxygenNote 7.1.2

Language en-US

VignetteBuilder knitr

NeedsCompilation no

Author Brandon Moretz [aut, cre] (ORCID:
<<https://orcid.org/0000-0001-8832-5173>>)

Repository CRAN

Date/Publication 2022-03-14 19:30:02 UTC

Contents

.onAttach	3
.onDetach	4
.onLoad	4
active	4
apply_active_settings	5
as.character.log_level	5
as.integer.log_level	6
class_scope	6
clean_internal_calls	7
config_specification	7
create_log_levels	8
display_log_levels	8
ensure_logger	9
evaluate_layout	9
exec_context	10
extract_func_name	11
format.fmt_timestamp	11
format.log_level	12
format_fn_call	12
get_active_settings	13
get_call_stack	13
get_configurations	14
get_minimum_severity	14
get_r_version	15
get_system_info	15
init_logger	16
is_logger_call	16
length.log_layout	17
level_description	17
level_info	18
level_name	18
level_severities	19
level_severity	19
load_log_layouts	20
LogDispatch	20
log_layouts	21
log_layout_detail	22
log_levels	22
new_fmt_cls_field	23
new_fmt_exec_scope	23
new_fmt_layout	24
new_fmt_line_break	24

new_fmt_literal	25
new_fmt_log_level	25
new_fmt_log_msg	26
new_fmt_metric	26
new_fmt_timestamp	27
new_log_layout	28
new_log_level	28
style	29
style.fmt_layout	29
style.log_level	30
sys_context	30
value	31
value.fmt_cls_field	32
value.fmt_exec_scope	32
value.fmt_literal	33
value.fmt_log_level	33
value.fmt_log_msg	34
value.fmt_metric	34
value.fmt_newline	35
value.fmt_timestamp	35
wipe_logger	36

Index	37
--------------	-----------

.onAttach	<i>Attach Handler Package initialization routine.</i>
-----------	---

Description

Attach Handler
Package initialization routine.

Usage

.onAttach(libname, pkgname)

Arguments

libname	library name
pkgname	package name

<code>.onDetach</code>	<i>Detach Handler Package initialization routine.</i>
------------------------	---

Description

Detach Handler
Package initialization routine.

Usage

`.onDetach(libpath)`

Arguments

<code>libpath</code>	<code>libpath</code>
----------------------	----------------------

<code>.onLoad</code>	<i>Load Handler Package initialization routine.</i>
----------------------	---

Description

Load Handler
Package initialization routine.

Usage

`.onLoad(libname, pkgname)`

Arguments

<code>libname</code>	library name
<code>pkgname</code>	package name

<code>active</code>	<i>Active Settings</i>
---------------------	------------------------

Description

Package environment variable to hold global level settings.

Usage

`active`

Format

An object of class environment of length 4.

apply_active_settings *Apply Active Logger Settings*

Description

Parses and loads the settings specified in the logger configuration and ensures they are active in the environment.

Usage

```
apply_active_settings(settings)
```

Arguments

settings	defined in the configuration
----------	------------------------------

See Also

Other Configuration: [config_specification\(\)](#), [create_log_levels\(\)](#), [display_log_levels\(\)](#), [ensure_logger\(\)](#), [get_active_settings\(\)](#), [load_log_layouts\(\)](#), [wipe_logger\(\)](#)

as.character.log_level

Get Log Level Name

Description

gets the name of the log level though casting to a character and forwarding the call to get_level_name.

Usage

```
## S3 method for class 'log_level'
as.character(x, ...)
```

Arguments

x	log level
...	ignored

Value

log level name

```
as.integer.log_level
```

Gets the severity of a log level.

Description

Gets the severity of a log level.

Usage

```
## S3 method for class 'log_level'
as.integer(x, ...)
```

Arguments

x	log level
...	ignored

Value

log level

```
class_scope
```

Calling Class Scope

Description

Gets the exposed public field scope of a R6 class. Used for evaluating cls field execution scopes.

Usage

```
class_scope(cls)
```

Arguments

cls	R6 class to export.
-----	---------------------

Value

system context for evaluating fmt_metric objects.

See Also

Other Context: [exec_context\(\)](#), [format_fn_call\(\)](#), [get_call_stack\(\)](#), [get_r_version\(\)](#), [get_system_info\(\)](#), [sys_context\(\)](#)

clean_internal_calls	<i>Clean System Calls</i>
----------------------	---------------------------

Description

Cleans up any internal system calls from inside the package from the call stack.

Usage

```
clean_internal_calls(call_stack)
```

Arguments

call_stack	call stack
------------	------------

Value

string representation of a func call.

See Also

Other Internal: [extract_func_name\(\)](#), [is_logger_call\(\)](#)

config_specification	<i>Config Specification</i>
----------------------	-----------------------------

Description

Loads & attaches a logger with the specified config.

Usage

```
config_specification()
```

Value

Nothing.

See Also

Other Configuration: [apply_active_settings\(\)](#), [create_log_levels\(\)](#), [display_log_levels\(\)](#), [ensure_logger\(\)](#), [get_active_settings\(\)](#), [load_log_layouts\(\)](#), [wipe_logger\(\)](#)

create_log_levels	<i>Attach Log Levels</i>
-------------------	--------------------------

Description

Parses and loads the levels specified in the logging configuration and registers them with the dispatcher via the log_levels active binding.

Usage

```
create_log_levels(definitions)
```

Arguments

definitions	defined in the configuration
-------------	------------------------------

See Also

Other Configuration: [apply_active_settings\(\)](#), [config_specification\(\)](#), [display_log_levels\(\)](#), [ensure_logger\(\)](#), [get_active_settings\(\)](#), [load_log_layouts\(\)](#), [wipe_logger\(\)](#)

display_log_levels	<i>Display Log Levels</i>
--------------------	---------------------------

Description

A utility function that dynamically displays the configured log levels (loaded from config), and outputs them in a simple layout with only the log level and msg formatted in their crayon styles.

Usage

```
display_log_levels()
```

See Also

Other Configuration: [apply_active_settings\(\)](#), [config_specification\(\)](#), [create_log_levels\(\)](#), [ensure_logger\(\)](#), [get_active_settings\(\)](#), [load_log_layouts\(\)](#), [wipe_logger\(\)](#)

ensure_logger	<i>Ensure Instance</i>
---------------	------------------------

Description

Ensures there is an active dispatcher attached to the specified environment.

Usage

ensure_logger(variable)

Arguments

variable variable name.

Value

None.

See Also

Other Configuration: [apply_active_settings\(\)](#), [config_specification\(\)](#), [create_log_levels\(\)](#), [display_log_levels\(\)](#), [get_active_settings\(\)](#), [load_log_layouts\(\)](#), [wipe_logger\(\)](#)

evaluate_layout	<i>Evaluate Layout</i>
-----------------	------------------------

Description

Evaluates a log layout, which is simply a container for a set of formats that specify the log entry layout.

Usage

evaluate_layout(detail, context)

Arguments

detail The details of the layout specified for evaluation.
context a list of contexts needed to evaluate formats in the the layout.

Value

evaluated log layout

See Also

Other Log Layout: [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

 exec_context

Execution Context

Description

Wrapper around Sys.info() and get_r_version that provides a consolidated list of variables used for logging contexts.

Usage

```
exec_context(
  keep_args = FALSE,
  max_calls = 5,
  call_subset = c(-1, -1),
  filter_internal = TRUE
)
```

Arguments

keep_args	bool to specify keep function all arguments
max_calls	maximum number of calls to keep from the stack
call_subset	offset index into system calls
filter_internal	filter out internal calls?

Value

system context for evaluating fmt_metric objects.

See Also

Other Context: [class_scope\(\)](#), [format_fn_call\(\)](#), [get_call_stack\(\)](#), [get_r_version\(\)](#), [get_system_info\(\)](#), [sys_context\(\)](#)

extract_func_name	<i>Extract Function Name</i>
-------------------	------------------------------

Description

Extracts the name of the function from a deparse call.

Usage

```
extract_func_name(func)
```

Arguments

func	function name
------	---------------

Value

function name without arguments

See Also

Other Internal: [clean_internal_calls\(\)](#), [is_logger_call\(\)](#)

format.fmt_timestamp	<i>Gets the format of a format object.</i>
----------------------	--

Description

Gets the format of a format object.

Usage

```
## S3 method for class 'fmt_timestamp'  
format(x, ...)
```

Arguments

x	object to extract value from.
...	further arguments passed to or from other methods.

Value

object's value

format.log_level	<i>Log Level Format</i>
------------------	-------------------------

Description

formats a message with the style of the log level.

Usage

```
## S3 method for class 'log_level'
format(x, message = character(0), ...)
```

Arguments

x	log level
message	message to format
...	further arguments passed to or from other methods.

Value

styled level information

Examples

```
## Not run:
level_info(LEVEL)

## End(Not run)
```

format_fn_call	<i>Format Function Call</i>
----------------	-----------------------------

Description

Formats a function call into a deparsed string.

Usage

```
format_fn_call(expr, cutoff = 100L)
```

Arguments

expr	function call
cutoff	max width cutoff

Value

string representation of a func call.

See Also

Other Context: [class_scope\(\)](#), [exec_context\(\)](#), [get_call_stack\(\)](#), [get_r_version\(\)](#), [get_system_info\(\)](#), [sys_context\(\)](#)

get_active_settings	<i>Active Logger Settings</i>
---------------------	-------------------------------

Description

Gets the active global settings for the logger.

Usage

```
get_active_settings()
```

See Also

Other Configuration: [apply_active_settings\(\)](#), [config_specification\(\)](#), [create_log_levels\(\)](#), [display_log_levels\(\)](#), [ensure_logger\(\)](#), [load_log_layouts\(\)](#), [wipe_logger\(\)](#)

get_call_stack	<i>Formatted Call Stack</i>
----------------	-----------------------------

Description

Placeholder for the formatted call stack in a log layout.

Usage

```
get_call_stack(
    keep_args = FALSE,
    call_subset = c(-1, -1),
    filter_internal = TRUE
)
```

Arguments

keep_args	T/F to indicate if you want to keep call arguments or not.
call_subset	offset index into system calls
filter_internal	filter out internal calls?

Value

formatted call stack

See Also

Other Context: [class_scope\(\)](#), [exec_context\(\)](#), [format_fn_call\(\)](#), [get_r_version\(\)](#), [get_system_info\(\)](#), [sys_context\(\)](#)

<code>get_configurations</code>	<i>Get Configurations</i>
---------------------------------	---------------------------

Description

Gets all available logging configurations exposed by the package.

Usage

```
get_configurations(pkgname = "dyn.log")
```

Arguments

<code>pkgname</code>	package name to get configs for.
----------------------	----------------------------------

See Also

Other Logging: [LogDispatch](#), [init_logger\(\)](#)

<code>get_minimum_severity</code>	<i>Get Minimum Severity</i>
-----------------------------------	-----------------------------

Description

Gets the log level with the highest threshold which is used internally to log 'trace/info' level messages.

Usage

```
get_minimum_severity()
```

Value

styled level information

get_r_version	<i>R Version</i>
---------------	------------------

Description

Wrapper around `R.Version()` to produce a nicely formatted string for use use in `sys_context`.

Usage

```
get_r_version()
```

Value

R environment version is (major).(minor) format

See Also

Other Context: [class_scope\(\)](#), [exec_context\(\)](#), [format_fn_call\(\)](#), [get_call_stack\(\)](#), [get_system_info\(\)](#), [sys_context\(\)](#)

get_system_info	<i>Get System Info</i>
-----------------	------------------------

Description

Wrapper around `Sys.info()` that provides the values in a named list format.

Usage

```
get_system_info()
```

Value

`Sys.info()` as a named list

See Also

Other Context: [class_scope\(\)](#), [exec_context\(\)](#), [format_fn_call\(\)](#), [get_call_stack\(\)](#), [get_r_version\(\)](#), [sys_context\(\)](#)

init_logger	<i>Init Logger</i>
-------------	--------------------

Description

Loads the configuration passed in, or uses the default if nothing is specified, and attaches a reference to the singleton dispatcher to the global environment.

Usage

```
init_logger(file_path = NULL)
```

Arguments

file_path logging configuration to use.

See Also

Other Logging: [LogDispatch](#), [get_configurations\(\)](#)

is_logger_call	<i>Is Logger Call</i>
----------------	-----------------------

Description

Determines if a call came from the logger, so we can exclude it from the call stack.

Usage

```
is_logger_call(call)
```

Arguments

call function call

Value

string representation of a func call.

See Also

Other Internal: [clean_internal_calls\(\)](#), [extract_func_name\(\)](#)

length.log_layout	<i>Log Layout Length</i>
-------------------	--------------------------

Description

Generic override for length of a log layout that returns the number of individual format objects in the layout.

Usage

```
## S3 method for class 'log_layout'
length(x, ...)
```

Arguments

- x log format
- ... further arguments passed to or from other methods.

Value

number of formats in the layout.

level_description	<i>Log Level Description</i>
-------------------	------------------------------

Description

Gets the description of a log level.
Gets the description of a log level.

Usage

```
level_description(level)

level_description(level)
```

Arguments

- level log level

Value

level description
level severity

Examples

```
## Not run:
level_description(LEVEL)

## End(Not run)
## Not run:
level_description(LEVEL)

## End(Not run)
```

level_info	<i>Log Level Information</i>
------------	------------------------------

Description

Gets log level information.

Usage

```
level_info(level)
```

Arguments

level log level

Value

log level information

Examples

```
## Not run:
level_info(LEVEL)

## End(Not run)
```

level_name	<i>Get Level Name</i>
------------	-----------------------

Description

gets the name of the log level.

Usage

```
level_name(level)
```

Arguments

level log level

Value

log level name

level_severities	<i>Level Severities</i>
------------------	-------------------------

Description

Gets the severity associated with each log level.

Usage

level_severities()

Value

styled level information

level_severity	<i>get level severity</i>
----------------	---------------------------

Description

Gets the severity of a log level.

Usage

level_severity(level)

level_severity(level)

Arguments

level log level

Value

level severity

level severity

Examples

```
## Not run:
level_severity(LEVEL)

## End(Not run)
## Not run:
level_severity(LEVEL)

## End(Not run)
```

load_log_layouts	<i>Load Log Layouts</i>
------------------	-------------------------

Description

Parses and loads layouts specified in the logging configuration and registers them with the log dispatcher via the log_layouts active binding.

Usage

```
load_log_layouts(layouts)
```

Arguments

layouts defined in the configuration

Value

None.

See Also

Other Configuration: [apply_active_settings\(\)](#), [config_specification\(\)](#), [create_log_levels\(\)](#), [display_log_levels\(\)](#), [ensure_logger\(\)](#), [get_active_settings\(\)](#), [wipe_logger\(\)](#)

LogDispatch	<i>Log Dispatch</i>
-------------	---------------------

Description

R6 Class that dispatches log messages throughout the application.

Details

This object is designed to a centralized logging dispatcher that renders log messages with the appropriate context of the calling object. The log_layout() object is used to generate log message layouts (render formats), which are used by the LogDispatcher to render highly-customizable and detailed log messages.

Methods

Public methods:

- [LogDispatch\\$new\(\)](#)
- [LogDispatch\\$attach_log_level\(\)](#)

Method `new()`: Creates a new instance of a log config.

Usage:

```
LogDispatch$new()
```

Returns: A new LogLayout object.

Method `attach_log_level()`: Attaches a S3 `log_level` object to the log dispatcher by creating a new function wrapping the specified log level, and binding an instance of the dispatcher quote block with the context of the log level.

Usage:

```
LogDispatch$attach_log_level(log_level)
```

Arguments:

`log_level` log level to attach

See Also

Other Logging: [get_configurations\(\)](#), [init_logger\(\)](#)

log_layouts

Log Layouts

Description

an active binding to keep track of log layouts created with `new_log_layout`.

Usage

```
log_layouts(association = character(0), layout = NULL)
```

Arguments

<code>association</code>	named association to the layout
<code>layout</code>	log layout to add if not already existing.

Value

defined log layouts

log_layout_detail	<i>Log Layout Detail</i>
-------------------	--------------------------

Description

Gets the layout formats and the distinct format types in a log layout instance, which is useful for determining the appropriate amount of log context to construct.

Usage

log_layout_detail(layout)

Arguments

layout object to extract layout detail from.

Value

layout format

log_levels	<i>Log Levels</i>
------------	-------------------

Description

an active binding to keep track of log levels created with new_log_level.

Usage

log_levels(name = character(0), level = NULL)

Arguments

name name associated with the log level
level log level to add if not already existing.

Value

defined log levels

new_fmt_cls_field	<i>Formatted field from the calling class scope.</i>
-------------------	--

Description

Placeholder for a container class field

Usage

new_fmt_cls_field(style, field)

Arguments

style	crayon::style()
field	field in the object to display

Value

new_fmt_cls_field

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

new_fmt_exec_scope	<i>Formatted variable from the execution scope.</i>
--------------------	---

Description

Placeholder for an execution scope variable.

Usage

new_fmt_exec_scope(style, field)

Arguments

style	crayon::style()
field	execution scope field

Value

new_fmt_cls_field

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

<code>new_fmt_layout</code>	<i>Format Layout</i>
-----------------------------	----------------------

Description

Base type for log format objects.

Usage

`new_fmt_layout(style)`

Arguments

`style` [crayon](#) that the layout will use in log generation.

Value

new log format

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

<code>new_fmt_line_break</code>	<i>Formatted Line Break</i>
---------------------------------	-----------------------------

Description

Placeholder for a new line in a log layout.

Usage

`new_fmt_line_break()`

Value

log layout newline.

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

new_fmt_literal	<i>Formatted Literal</i>
-----------------	--------------------------

Description

Placeholder for a formatted literal in a log layout.

Usage

```
new_fmt_literal(style, literal)
```

Arguments

style	format style (crayon)
literal	log value

Value

log metric layout.

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

Examples

```
## Not run:  
new_fmt_literal(red $ bold, "literal text")  
  
new_fmt_literal(blue $ italic, "literal text")  
  
## End(Not run)
```

new_fmt_log_level	<i>Formatted Log Level</i>
-------------------	----------------------------

Description

Placeholder for the formatted log level in a log layout.

Usage

```
new_fmt_log_level()
```

Value

a fmt_log_level.

<code>new_fmt_log_msg</code>	<i>Formatted Messaged, based on log level</i>
------------------------------	---

Description

Placeholder for the log msg in a log layout.

Usage

`new_fmt_log_msg()`

Value

log layout newline.

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

<code>new_fmt_metric</code>	<i>Formatted Metric</i>
-----------------------------	-------------------------

Description

Inserts a formatted log metric.

Usage

`new_fmt_metric(style, metric)`

Arguments

<code>style</code>	that the layout will use in log generation
<code>metric</code>	the metric to log.

Value

a new formatted metric

See Also

[LogDispatch](#)
Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_timestamp\(\)](#), [new_log_layout\(\)](#)

Examples

```
## Not run:
new_fmt_metric(bold $ green, "sysname")

new_fmt_metric(bold $ red, "release")

## End(Not run)
```

<code>new_fmt_timestamp</code>	<i>Formatted Time stamp</i>
--------------------------------	-----------------------------

Description

Placeholder for a formatted time stamp in a log layout.

Usage

```
new_fmt_timestamp(style, format = "[%x %H:%M:%S %z]")
```

Arguments

<code>style</code>	format style (crayon)
<code>format</code>	time stamp format, defaults to: <code>%x %H:%M:%S %z</code> , e.g., 12/04/21 14:31:25 -0500

Value

log metric layout.

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_log_layout\(\)](#)

Examples

```
## Not run:
fmt_timestamp(red $ bold, "%Y-%m-%d %H:%M:%S")

fmt_timestamp(blue $ italic, "%x %H:%M:%S %z")

## End(Not run)
```

<i>new_log_layout</i>	<i>Log Layout</i>
-----------------------	-------------------

Description

a class that stores a collection of log format objects and understands how to associate a given format to a class of objects.

Usage

```
new_log_layout(  
    format = list(),  
    seperator = " ",  
    new_line = "\n",  
    association = character()  
)
```

Arguments

- format collection of format objects to initialize with.
- seperator format entry separator, defaults to a single space.
- new_line the layout separator that is inserted between lines.
- association objects to associate this log format with.

Value

object's value

See Also

Other Log Layout: [evaluate_layout\(\)](#), [new_fmt_cls_field\(\)](#), [new_fmt_exec_scope\(\)](#), [new_fmt_layout\(\)](#), [new_fmt_line_break\(\)](#), [new_fmt_literal\(\)](#), [new_fmt_log_msg\(\)](#), [new_fmt_metric\(\)](#), [new_fmt_timestamp\(\)](#)

<i>new_log_level</i>	<i>Log Level</i>
----------------------	------------------

Description

S3 object to represent a typed & predefined log level.

Usage

```
new_log_level(name, description, severity, log_style = NULL, msg_style = NULL)
```

Arguments

name	name of the log level is the string representation.
description	description of the log level & limited info on appropriate usage.
severity	log severity is used in determining if a message should get displayed according to the currently set evaluation threshold.
log_style	is a <code>crayon::style()</code> that will colorize the log level.
msg_style	is a <code>crayon::style()</code> style that will gray scale the log message, with typically inverted strength, according to the severity.

Value

log_level

style	<i>Style</i>
-------	--------------

Description

Generic style method used for overriding to get style information from various logging objects.

Usage

```
style(obj)
```

Arguments

obj	object to extract value from.
-----	-------------------------------

Value

object's value

style.fmt_layout	<i>Style</i>
------------------	--------------

Description

Gets the style of a format object.

Usage

```
## S3 method for class 'fmt_layout'
style(obj, ...)
```

Arguments

obj object to extract value from.
... further arguments passed to or from other methods.

Value

object's value

style.log_level	<i>Get Level Style</i>
-----------------	------------------------

Description

gets the style of the log level.

Usage

```
## S3 method for class 'log_level'  
style(obj, ...)
```

Arguments

obj log level
... further arguments passed to or from other methods.

Value

log level name

sys_context	<i>System Context</i>
-------------	-----------------------

Description

Wrapper around Sys.info() and get_r_version that provides a consolidated list of variables used for logging contexts.

Usage

```
sys_context()
```

Value

system context for evaluating fmt_metric objects.

Metrics

System Context

- "sysname" : The operating system name.
- "release" : The OS release.
- "version" : The OS version.
- "nodename" : A name by which the machine is known on the network (if any).
- "machine" : A concise description of the hardware, often the CPU type.
- "login" : The user's login name, or "unknown" if it cannot be ascertained.
- "user" : The name of the real user ID, or "unknown" if it cannot be ascertained.
- "r-ver" : R Version in (major).(minor) format.

See Also

Other Context: [class_scope\(\)](#), [exec_context\(\)](#), [format_fn_call\(\)](#), [get_call_stack\(\)](#), [get_r_version\(\)](#), [get_system_info\(\)](#)

value	<i>Value</i>
-------	--------------

Description

Base method for getting the value of a format object.

Usage

value(obj, ...)

Arguments

- | | |
|-----|--|
| obj | object to extract value from. |
| ... | further arguments passed to or from other methods. |

Value

object's value

value.fmt_cls_field	<i>Value</i>
---------------------	--------------

Description

Generic override for getting the value of an enclosing class variable.

Usage

```
## S3 method for class 'fmt_cls_field'
value(obj, cls_context, ...)
```

Arguments

- obj object to extract value from.
- cls_context class scope to evaluate with.
- ... further arguments passed to or from other methods.

Value

object's value

value.fmt_exec_scope	<i>Value</i>
----------------------	--------------

Description

Generic override for getting the value of an execution scope variable.

Usage

```
## S3 method for class 'fmt_exec_scope'
value(obj, env_context, ...)
```

Arguments

- obj object to extract value from.
- env_context class scope to evaluate with.
- ... further arguments passed to or from other methods.

Value

object's value

value.fmt_literal	<i>Value</i>
-------------------	--------------

Description

Generic override for getting the value of a literal log message.

Usage

```
## S3 method for class 'fmt_literal'  
value(obj, ...)
```

Arguments

obj	object to extract value from.
...	further arguments passed to or from other methods.

Value

object's value

value.fmt_log_level	<i>Value</i>
---------------------	--------------

Description

Generic override for getting the value for log level information.

Usage

```
## S3 method for class 'fmt_log_level'  
value(obj, lvl_context, ...)
```

Arguments

obj	object to extract value from.
lvl_context	context to evaluate log level.
...	further arguments passed to or from other methods.

Value

object's value

value.fmt_log_msg	<i>Value</i>
-------------------	--------------

Description

Generic override for getting the value of an log format message.

Usage

```
## S3 method for class 'fmt_log_msg'
value(obj, msg_context, ...)
```

Arguments

- obj object to extract value from.
- msg_context context to evaluate log message.
- ... further arguments passed to or from other methods.

Value

object's value

value.fmt_metric	<i>Value</i>
------------------	--------------

Description

Generic override for getting the value of an system info variable.

Usage

```
## S3 method for class 'fmt_metric'
value(obj, sys_context, ...)
```

Arguments

- obj object to extract value from.
- sys_context context to evaluate the metric.
- ... further arguments passed to or from other methods.

Value

object's value

value.fmt_newline	<i>Value</i>
-------------------	--------------

Description

Generic override for getting the value of a new line placeholder.

Usage

```
## S3 method for class 'fmt_newline'  
value(obj, ...)
```

Arguments

- obj object to extract value from.
- ... further arguments passed to or from other methods.

Value

object's value

value.fmt_timestamp	<i>Value</i>
---------------------	--------------

Description

Generic override for getting the value of a formatted timestamp.

Usage

```
## S3 method for class 'fmt_timestamp'  
value(obj, ...)
```

Arguments

- obj object to extract value from.
- ... further arguments passed to or from other methods.

Value

object's value

`wipe_logger`*Wipe the Logger Instance*

Description

Cleans up any dangling global instance from a previous load.

Usage

```
wipe_logger()
```

Value

None.

See Also

Other Configuration: [apply_active_settings\(\)](#), [config_specification\(\)](#), [create_log_levels\(\)](#), [display_log_levels\(\)](#), [ensure_logger\(\)](#), [get_active_settings\(\)](#), [load_log_layouts\(\)](#)

Index

- * **Configuration**
 - apply_active_settings, [5](#)
 - config_specification, [7](#)
 - create_log_levels, [8](#)
 - display_log_levels, [8](#)
 - ensure_logger, [9](#)
 - get_active_settings, [13](#)
 - load_log_layouts, [20](#)
 - wipe_logger, [36](#)
- * **Context**
 - class_scope, [6](#)
 - exec_context, [10](#)
 - format_fn_call, [12](#)
 - get_call_stack, [13](#)
 - get_r_version, [15](#)
 - get_system_info, [15](#)
 - sys_context, [30](#)
- * **Formats**
 - new_fmt_log_level, [25](#)
- * **Internal**
 - clean_internal_calls, [7](#)
 - extract_func_name, [11](#)
 - is_logger_call, [16](#)
- * **Log Layout**
 - evaluate_layout, [9](#)
 - new_fmt_cls_field, [23](#)
 - new_fmt_exec_scope, [23](#)
 - new_fmt_layout, [24](#)
 - new_fmt_line_break, [24](#)
 - new_fmt_literal, [25](#)
 - new_fmt_log_msg, [26](#)
 - new_fmt_metric, [26](#)
 - new_fmt_timestamp, [27](#)
 - new_log_layout, [28](#)
- * **Log Level**
 - new_log_level, [28](#)
- * **Logging**
 - get_configurations, [14](#)
 - init_logger, [16](#)
 - LogDispatch, [20](#)
- * **datasets**
 - active, [4](#)
 - .onAttach, [3](#)
 - .onDetach, [4](#)
 - .onLoad, [4](#)
- active, [4](#)
- apply_active_settings, [5](#), [7–9](#), [13](#), [20](#), [36](#)
- as.character.log_level, [5](#)
- as.integer.log_level, [6](#)
- class_scope, [6](#), [10](#), [13–15](#), [31](#)
- clean_internal_calls, [7](#), [11](#), [16](#)
- config_specification, [5](#), [7](#), [8](#), [9](#), [13](#), [20](#), [36](#)
- crayon, [24](#)
- create_log_levels, [5](#), [7](#), [8](#), [8](#), [9](#), [13](#), [20](#), [36](#)
- display_log_levels, [5](#), [7](#), [8](#), [8](#), [9](#), [13](#), [20](#), [36](#)
- ensure_logger, [5](#), [7](#), [8](#), [9](#), [13](#), [20](#), [36](#)
- evaluate_layout, [9](#), [23–28](#)
- exec_context, [6](#), [10](#), [13–15](#), [31](#)
- extract_func_name, [7](#), [11](#), [16](#)
- format.fmt_timestamp, [11](#)
- format.log_level, [12](#)
- format_fn_call, [6](#), [10](#), [12](#), [14](#), [15](#), [31](#)
- get_active_settings, [5](#), [7–9](#), [13](#), [20](#), [36](#)
- get_call_stack, [6](#), [10](#), [13](#), [13](#), [15](#), [31](#)
- get_configurations, [14](#), [16](#), [21](#)
- get_minimum_severity, [14](#)
- get_r_version, [6](#), [10](#), [13–15](#), [15](#), [31](#)
- get_system_info, [6](#), [10](#), [13–15](#), [15](#), [31](#)
- init_logger, [14](#), [16](#), [21](#)
- is_logger_call, [7](#), [11](#), [16](#)
- length.log_layout, [17](#)
- level_description, [17](#)

- level_info, [18](#)
- level_name, [18](#)
- level_severities, [19](#)
- level_severity, [19](#)
- load_log_layouts, [5](#), [7–9](#), [13](#), [20](#), [36](#)
- log_layout_detail, [22](#)
- log_layouts, [21](#)
- log_levels, [22](#)
- LogDispatch, [14](#), [16](#), [20](#), [26](#)

- new_fmt_cls_field, [10](#), [23](#), [24–28](#)
- new_fmt_exec_scope, [10](#), [23](#), [23](#), [24–28](#)
- new_fmt_layout, [10](#), [23](#), [24](#), [24](#), [25–28](#)
- new_fmt_line_break, [10](#), [23](#), [24](#), [24](#), [25–28](#)
- new_fmt_literal, [10](#), [23](#), [24](#), [25](#), [26–28](#)
- new_fmt_log_level, [25](#)
- new_fmt_log_msg, [10](#), [23–26](#), [26](#), [27](#), [28](#)
- new_fmt_metric, [10](#), [23–26](#), [26](#), [27](#), [28](#)
- new_fmt_timestamp, [10](#), [23–26](#), [27](#), [28](#)
- new_log_layout, [10](#), [23–27](#), [28](#)
- new_log_level, [28](#)

- style, [29](#)
- style_fmt_layout, [29](#)
- style_log_level, [30](#)
- sys_context, [6](#), [10](#), [13–15](#), [30](#)

- value, [31](#)
- value_fmt_cls_field, [32](#)
- value_fmt_exec_scope, [32](#)
- value_fmt_literal, [33](#)
- value_fmt_log_level, [33](#)
- value_fmt_log_msg, [34](#)
- value_fmt_metric, [34](#)
- value_fmt_newline, [35](#)
- value_fmt_timestamp, [35](#)

- wipe_logger, [5](#), [7–9](#), [13](#), [20](#), [36](#)