

Package ‘EventLogger’

October 31, 2019

Title Capture events and timings across multiple objects in a shared
DataTable

Version 1.15.3

Date 2019-10-31

Author Charles Tilford [aut, cre]

Maintainer Charles Tilford <cran@agent.fastmail.com>

Description This is a utility package designed to hold and report
events. It builds a record of manually-defined events and tracks the
timings between each for performance analysis.

URL <https://github.com/maptracker/setfisher/tree/master/EventLogger>

BugReports <https://github.com/maptracker/setfisher/issues>

Depends R (>= 3.1)

Imports CatMisc (>= 1.1.0), crayon, data.table, dymaictable, methods

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

Suggests testthat

NeedsCompilation no

R topics documented:

actionMessage	2
dateMessage	3
debugMessage	4
err	5
EventLogger-class	6
EvLogObj	7
fieldDescriptions	8
log	8

logText	9
message	10
showLog	11
tidyTime	12
vb	13
verbose	14
Index	15

actionMessage	<i>EventLogger Action Message</i>
---------------	-----------------------------------

Description

EventLogger object method to present and record an emphatic message

Arguments

- msg The text to display and show
- prefix Default '[+]', the text to show before the message
- color Default 'red', the color of the displayed text
- ... Passed to [message](#)
- help Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:
myObject$actionMessage( help=TRUE )

myObject$actionMessage(msg="No message provided!", prefix='[+]', color="red")

This is just a wrapper for message where prefix='[+]' and color="red"
```

Value

The [log](#) table, invisibly

See Also

[log](#), [message](#)

Examples

```
e1 <- EventLogger()
e1$actionMessage("Outer airlock door not responding")

# Show the log, nicely formatted:
e1
```

dateMessage	<i>EventLogger Date Message</i>
-------------	---------------------------------

Description

EventLogger object method to present and record a message with a date

Arguments

msg	The text to display and show
...	Passed to message
help	Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:  
myObject$dateMessage( help=TRUE )  
  
myObject$dateMessage(msg="No message provided!", ...)
```

This is just a wrapper for [message](#) where datestamp=TRUE

Value

The [log](#) table, invisibly

See Also

[log](#), [message](#)

Examples

```
el <- EventLogger()  
el$dateMessage("Something auspicious just happened")  
  
# Show the log, nicely formatted:  
el
```

debugMessage

*EventLogger Debug Message***Description**

EventLogger object method to present and record a message for debugging

Arguments

msg	The text to display and show
prefix	Default '[DEBUG]', the text to show before the message
color	Default 'white', the color of the displayed text
bgcolor	Default 'blue', the background color of the displayed text
...	Passed to message
help	Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:
```

```
myObject$debugMessage( help=TRUE )
```

```
myObject$debugMessage(msg="No message provided!", prefix='[DEBUG]',
  color="white", bgcolor="blue")
```

This is just a wrapper for [message](#) where prefix='[DEBUG]', color="white" and bgcolor="blue"

Value

The [log](#) table, invisibly

See Also

[log](#), [message](#)

Examples

```
e1 <- EventLogger()
for (cv in 1:9) {
  e1$debugMessage(c("Chevron", cv, "locked ..."))
}

# Show the log, nicely formatted:
e1
```

err*EventLogger Error Message*

Description

EventLogger object method to present and record an error message

Arguments

msg	The text to display and show
prefix	Default '[ERROR]', the text to show before the message
color	Default 'red', the color of the displayed text
bgcolor	Default 'yellow', the background color of the
...	Passed to message
help	Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:  
myObject$err( help=TRUE )
```

```
myObject$err(msg="No message provided!", prefix='[ERROR]',  
             color="red", bgcolor="yellow",...)
```

This is just a wrapper for [message](#) where prefix='[ERROR]', color="red" and bgcolor="yellow"

Value

The [log](#) table, invisibly

See Also

[log](#), [message](#)

Examples

```
e1 <- EventLogger()  
e1$err("Weight is not defined")  
# Show the log, nicely formatted:  
e1
```

EventLogger-class	<i>Event Logger</i>
-------------------	---------------------

Description

Utility ReferenceClass object for messaging and recording of events. Use `$message()` for general logging (stores event and reports to console), see **Methods** section for variants.

Details

A `$show()` method has been set, so simply evaluating an EventLogger object on the command line will pretty-print the result. The `data.table` holding the log information can be directly accessed in field `$log`.

Fields

`log` The `data.table` holding log messages
`vb` Logical flag indicating if verbose messaging should be active
`EvLogObj` An optional external EventLogger object. Used to centralize log information across multiple inheriting objects in one place.

Methods

`actionMessage(msg = "No message provided!", prefix = "[+]", color = "red", help = FALSE, ...)`
 Calls message with a '+' prefix and red coloring
`dateMessage(msg = "No message provided!", help = FALSE, ...)` Calls message() with `datestamp=TRUE`
`debugMessage(msg = "No message provided!", prefix = "[DEBUG]", color = "white", bgcolor = "blue", h`
 Calls message with a '[DEBUG]' prefix and white/blue coloring
`err(msg = "No message provided!", prefix = "[ERROR]", color = "red", bgcolor = "yellow", help = FAL`
 Calls message with an '[ERROR]' prefix and red/yellow coloring
`fieldDescriptions(help = FALSE)` A static list of brief descriptions for each field in this object
`helpSections(help = FALSE)` Static list organizing object methods into conceptual sections
`initialize(useColor = TRUE, ...)` Create a new RefClassHelper Reference Class object
`logText(width = 0.7 * getOption("width"), relative = TRUE, pad = 11, n = 0, help = FALSE)`
 Formats the log as a human readable two-column table
`message(msg = "No message provided!", prefix = NULL, color = NULL, bgcolor = NULL, datestamp = FALS`
 Display a formatted message, and store it in the log table
`show(help = FALSE)` Pretty-print the object
`showLog(help = FALSE, ...)` Pretty-prints the log, including total elapsed time
`tidyTime(x = NULL, pad = 0, help = FALSE)` Reports a time interval with unit management and colorization
`verbose(newval = NULL, help = FALSE)` Get or set the flag determining if messages are displayed

Examples

```

myEL <- EventLogger()
myEL$message("Did something important")
Sys.sleep(3)
myEL$actionMessage("Something emphatic has happened")
Sys.sleep(1)
myEL$dateMessage("Here's a date stamp")
myEL$debugMessage("Remember to comment this out in production")

# Pretty print the log, including an elapsed time:
myEL
# Expose the underlying data.table:
myEL$log

## Demo with inheritance of the class:
demo("objectInheritance", package="EventLogger", ask=FALSE)

```

EvLogObj

*EventLogger Object***Description**

Internal EventLogger field pointing to another object that holds a shared [log](#)

Details

This package was designed to be inherited by other ReferenceClass objects, and was also designed such that many objects could share a common log. An object that inherits (contains) EventLogger can choose to reference a different object with this field. If so, operations that would normally alter the EventLogger fields will instead act on the referenced object.

The other object is defined by the log parameter on creation.

```

## NORMALLY YOU WILL NOT WANT TO ACCESS THIS FIELD DIRECTLY
## ALTERING IT MAY RESULT IN CODE INSTABILITY

```

Value

An empty field if not set, or an EventLogger-compliant object

fieldDescriptions	<i>Field Descriptions</i>
-------------------	---------------------------

Description

A list of brief descriptions for each object field

Arguments

help Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:
myObject$fieldDescriptions( help=TRUE )

myObject$fieldDescriptions( )
```

This method returns a simple list of descriptive text for each object field. It is designed to help the user understand the role of each field.

Value

A list of character strings

See Also

[help](#)

Examples

```
myEL <- EventLogger()
myEL$fieldDescriptions()
```

log	<i>EventLogger Log</i>
-----	------------------------

Description

Internal EventLogger field holding the data.table of events

Details

A [data.table](#) (data.frame-compliant) table with two columns: \$Date (which is automatically populated at time of event) and \$Message

This field is intended to be accessed, but alterations (additions) should be performed using the [message](#) functions

Value

A data.table

See Also

[message](#), [showLog](#)

Examples

```
el <- EventLogger( )
el$message("An entry")
el$message("A second entry")
el$log
```

logText	<i>Log Text</i>
---------	-----------------

Description

EventLogger object method to generate pretty-formatted text of the log

Arguments

- widthDefault 70 length to be used when strwrap()ing the event text
- relative - Default TRUE, will show the time elapsed between events. If FALSE will show absolute time stamps
- pad - Default 11, character padding used to align the time column
- n Default 0, if greater will only show that number of most recent events
- help Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:
myObject$logText( help=TRUE )

myObject$logText( width=0.7 * getOption("width"),
                  relative=TRUE, pad=11, n=0 )
```

This method will parse the [log](#) data.table and generate a human-friendly, colorized table of events. The left column will report the time difference between log events, intending to help gauge how long individual events are taking. The right column will be event message text, strwrap()’ed to the user’s width option.

This method will return the ’raw’ character strings of the text. In general you will likely wish to instead use [showLog](#) to have the message shown in a way such that the ANSI color codes are properly evaluated for display.

See Also

[showLog](#), [message](#), [log](#)

Examples

```
e1 <- EventLogger()
e1$message("A sample message")
e1$logText()
```

message	<i>EventLogger Message</i>
---------	----------------------------

Description

EventLogger object method to present and record a message

Arguments

msg	The text to display and show
prefix	Default NULL, Optional text to display in front of message. Will not be recorded in the log.
color	Default NULL, foreground (text) color of message, not logged.
bgcolor	Default NULL, background color of message, not logged.
datestamp	Default FALSE; If TRUE, then a datestamp will be displayed as well. Datestamps are always recorded in the log , regardless of this value.
fatal	Default FALSE; If TRUE, then stop() execution as well.
collapse	Default ”, text to use when collapsing msg vector.
help	Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:
myObject$message( help=TRUE )

myObject$message(msg="No message provided!", prefix=NULL,
                  color=NULL, bgcolor=NULL, datestamp=FALSE,
                  fatal=FALSE, collapse=" ")
```

This method is used to both display a message to the terminal and to record it in the [log](#)

Several wrapper functions exist with pre-configured display options:

- [dateMessage](#) Will display a datestamp
- [actionMessage](#) Prefix with '[+]', red color
- [debugMessage](#) Prefix with '[DEBUG]', white FG, blue BG
- [err](#) Prefix with 'ERROR', red FG, yellow BG

Bear in mind that for all functions the appearance is merely cosmetic - only the contents of msg will go into the log.

Value

The [log](#) table, invisibly

See Also

[log](#) [dateMessage](#) (just sets datestamp=TRUE) [actionMessage](#) (just sets prefix='[+]' and color='red') [debugMessage](#) (sets prefix='[DEBUG]', color="white", bgcolor="blue") [err](#) (sets prefix='[ERROR]', color="red", bgcolor="yellow")

Examples

```
e1 <- EventLogger()
e1$message("A generic message")

# Show the log, nicely formatted:
e1
```

showLog

Show Log

Description

EventLogger object method to pretty-print the event log

Arguments

- ... Will be passed to [logText](#)
- help Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:
myObject$showLog( help=TRUE )

myObject$showLog( ... )
```

A simple wrapper for `logText()`, the contents of which are displayed in the shell using `cat`. This method is auto-invoked if an `EventLogger` object is evaluated in the shell (and there are no other `ReferenceClass` classes contained in the object that would take precedence)

See Also

[logText](#)

Examples

```
e1 <- EventLogger()
e1$message("A sample message")
e1$showLog()
```

tidyTime	<i>Tidy Time</i>
----------	------------------

Description

EventLogger object method to pretty-format a time difference

Arguments

- x Default NULL. Expects a single numeric value in seconds
- pad Default 0, a minimum width that the final string should occupy
- help Default FALSE. If TRUE, show this help and perform no other actions.

Details

```
## Method Usage:
myObject$tidyTime( help=TRUE )

myObject$tidyTime( (x=NULL, pad=0 )
```

Takes a character vector and applies foreground and/or background color to it. This method is used by [showLog](#) to highlight shorter or longer time frames.

Value

A character string with ANSI color codes injected by [crayon](#)

See Also

[showLog](#), [logText](#)

Examples

```
e1 <- EventLogger()  
base::message( e1$tidyTime(0.00013) )  
base::message( e1$tidyTime(35232) )
```

vb

EventLogger Verbosity Flag

Description

Internal EventLogger field holding the data.table of events

Details

A logical flag setting if EventLogger is verbose or not.

```
## NORMALLY YOU WILL NOT WANT TO ACCESS THIS FIELD DIRECTLY  
## Instead, use the \link{verbose} method to check and alter the value
```

Value

A logical value

See Also

[message](#)

Examples

```
e1 <- EventLogger( )  
e1$actionMessage("HELLO")  
e1$verbose( FALSE )  
e1$message("please no shouting")  
e1
```

verbose

EventLogger Verbosity

Description

EventLogger object method to get/set verbosity flag

Arguments

newval	Default NULL. If provided and can be made logical, will set the flag. Inability to cast as logical will emit a non-fatal error.
help	Default FALSE. If TRUE, show this help and perform no other actions.

Details

Method Usage:

```
myObject$verbose( help=TRUE )
```

```
myObject$verbose( newval=NULL )
```

Gets or sets the flag determining if messages should be displayed to the terminal, or just logged.

Value

A single logical value, invisibly

See Also

[message](#), [colorize](#), [vb](#)

Examples

```
e1 <- EventLogger()
e1$err("Please be aware that something has gone wrong")
e1$verbose(FALSE)
e1$err("Another problem! But not on your screen. You'll need to check the log")
e1
```

Index

actionMessage, [2](#), [11](#)

colorize, [14](#)

crayon, [13](#)

data.table, [9](#)

dateMessage, [3](#), [11](#)

debugMessage, [4](#), [11](#)

err, [5](#), [11](#)

EventLogger (EventLogger-class), [6](#)

EventLogger-class, [6](#)

EvLogObj, [7](#)

fieldDescriptions, [8](#)

help, [8](#)

log, [2–5](#), [7](#), [8](#), [10](#), [11](#)

logText, [9](#), [12](#), [13](#)

message, [2–5](#), [9](#), [10](#), [10](#), [13](#), [14](#)

showLog, [9](#), [10](#), [11](#), [12](#), [13](#)

tidyTime, [12](#)

vb, [13](#), [14](#)

verbose, [14](#)