Package ‘rsyslog’

July 3, 2018

Title Interface to the 'syslog' System Logger

Version 1.0.1

Maintainer Aaron Jacobs <atheriel@gmail.com>

Description Functions to write messages to the 'syslog' system logger API, available on all 'POSIX'-compatible operating systems. Features include tagging messages with a priority level and application type, as well as masking (hiding) messages below a given priority level.

License GPL (>= 2)

URL https://github.com/atheriel/rsyslog

BugReports https://github.com/atheriel/rsyslog/issues

ByteCompile true

Encoding UTF-8

LazyData true

OS_type unix

RoxygenNote 6.0.1

SystemRequirements POSIX.1-2001

NeedsCompilation yes

Author Aaron Jacobs [aut, cre], Crescendo Technology Ltd. [cph]

Repository CRAN

Date/Publication 2018-07-03 19:30:03 UTC

R topics documented:

open_syslog ................................................................. 2
set_syslog_mask .......................................................... 3

Index 4
open_syslog  Write Messages to the System Log

Description

Write messages to the system log via the POSIX syslog interface. Since this is a thin wrapper around that interface, you may also want to take a look at its documentation. Note that neither open_syslog() nor close_syslog() is actually required, but using them is good practice.

Usage

```c
open_syslog(identifier, open_immediately = FALSE, include_pid = FALSE, fallback_to_console = FALSE, echo = FALSE, facility = NULL)

syslog(message, level = "INFO", facility = NULL)

close_syslog()
```

Arguments

- **identifier**: A string identifying the application.
- **open_immediately**: When TRUE, the connection will be opened immediately (equivalent to using LOG_NDELAY). Otherwise it will be opened when the first message is written to the log (equivalent to using LOG_ODELAY).
- **include_pid**: When TRUE, include the process ID in the log message. Equivalent to using LOG_PID.
- **fallback_to_console**: Write to the system console (e.g. /dev/console) if there is an error while sending to the system logger. Equivalent to using LOG_CONS.
- **echo**: Also log the message to standard error. Equivalent to using LOG_PERROR. Note that this is not actually part of the POSIX specification, and may not be available on your platform. If that is the case, setting this to TRUE will generate a warning.
- **facility**: The type of program doing the logging, according to the guidelines in RFC 5424. Generally one of "USER" or "LOCAL0" through "LOCAL7". When this is NULL, fall back on the default.
- **message**: The message to write to the system log.
- **level**: The priority level of the message. One of "DEBUG", "INFO", "NOTICE", "WARNING", "ERR", "CRITICAL", "ALERT", or "EMERG" – in that order of priority. See RFC 5424 for the basis of this schema.

Examples

```bash
## Not run:
open_syslog("my_script")
```
set_syslog_mask

set_syslog_mask can be used to prevent messages below a priority level from being written to the system log.

Usage

set_syslog_mask(level)

Arguments

level Mask (hide) messages below this priority level. One of "DEBUG", "INFO", "NOTICE", "WARNING", "ERR", "CRITICAL", or "ALERT" – in that order of priority. See RFC 5424 for the basis of this schema.

Examples

## Not run:
open_syslog("my_script")
syslog("This message is visible.", level = "INFO")
set_syslog_mask("WARNING")
syslog("No longer visible.", level = "INFO")
syslog("Still visible.", level = "WARNING")
close_syslog()

## End(Not run)
Index

close_syslog(open_syslog), 2
open_syslog, 2
set_syslog_mask, 3
syslog(open_syslog), 2