

NAME

logrd - bash logging and stream redirection library

SYNOPSIS

```
source logrd.bash
logrd-setup --copy-to-console

logrd-set level warn

log-warn "this is a warning"

# log to a file, but copy the messages to the console
logrd-redirect-stream file --copy-to-console stdlog

log-warn "this warning goes to both file and the console"
```

DESCRIPTION

logrd is a bash library which combines a simple logging facility with the ability to redirect program output as well as logging to multiple output streams.

Stream Redirection

logrd improves upon the standard stream redirection schemes by allowing writing to multiple streams. For example,

```
echo "this goes to the console"

logrd-redirect-stream file1 --copy-to-console stdout

echo "this goes to file1 and the console"

logrd-redirect-stream file2 --copy-to-stream stdout

echo "this goes to file1, file2, and the console"

logrd-restore-streams stdout

echo "this goes to the console"
```

Writing to streams other than stderr will be buffered.

Multiple streams may be directed to a single file:

```
logrd-redirect-stream file1 stdout stderr
```

Nested stream redirection

logrd does not keep a history of stream redirection. The *logrd-restore-streams* command restores a stream to the state it was when **logrd** was loaded. If the ability to restore streams to intermediate states is required, use subshells:

```
source logrd.bash
echo "this goes to the console"

logrd-redirect-stream file1 --copy-to-console stdout
echo "this goes to file1 and the console"

(
  logrd-redirect-stream file2 stdout
  echo "this goes to file1, file2, and the console"
)

echo "this goes to file1 and the console"
```

Logging

logrd provides commands which write to the logging stream (`stdlog`). It follows the standard paradigm of multiple levels of logging:

```
error warn notice info debug
```

and provides commands specific to each level, e.g.

```
log-error "this is at level ERROR"
```

as well as the level agnostic command `log-to`:

```
log-to warn "this is a warning"
```

The logging stream, *stdlog*, writes by default to the `stderr` stream, but may be redirected:

```
logrd-redirect-stream file stdlog
```

The file descriptor associated with *stdlog* may be obtained via the *logrd-get* command.

Log Formatting

The *logrd-format-message* function is responsible for formatting the message which will be output. It is passed the log level as well as the message to be logged, and should write the formatted message to the standard output stream. The default version performs no transformation on the message.

Background

Stream redirection in the bash shell is usually accomplished either by redirecting individual commands' output stream

```
echo "to stdout" > file
```

or by redirecting the shell's streams:

```
exec >file
echo "to stdout"
```

In the latter case child processes will inherit the redirected streams, and all further output sent to the standard output stream will end up in *file*.

It's also possible to send multiple streams to the same file. The following redirects the standard output stream, and then makes the standard error stream write to the standard output stream.

```
exec >file
exec 2>&1
```

Thus,

```
echo "to stdout"
echo "to stderr" >&2
```

results in both phrases being written to *file*.

LOADING

To load the **logrd** library, simply source it:

```
source logrd.bash
```

To efficiently initialize settings, use *logrd-setup*

logrd-setup

```
logrd-setup <options>
```

Initialize settings. This should be performed before any logging is performed.

The available options are:

--copy-to-console

--no-copy-to-console

Copy (or don't) all redirected streams to the console (that is, to the streams in existence when **logrd** was loaded). For example, this causes

```
logrd-redirect-stream file stdout
ls
```

to send the output of *ls* to both *file* and to the standard output stream

logrd-redirect-stream will also accept this option to restrict its application to a single redirection.

--copy-to-stream

--no-copy-to-stream

Copy (or don't) all redirected streams to their original destination as well as the new one.

For example, this causes

```
logrd-redirect-stream file1 stdout
logrd-redirect-stream file2 stdout
ls
```

to send the output of *ls* to *file1*, *file2*, and to the standard output stream.

logrd-redirect-stream will also accept this option to restrict its application to a single redirection.

-q

--quiet

Set the logging level to `error`.

`--env-prefix string`

`--env-prefix=string`

Set the prefix for environment variables. See *ENVIRONMENT*.

`--starting-save-fd integer`

`--starting-save-fd=integer`

For older versions of bash (that's you, Apple) **logrd** needs to search for unused file descriptors. This option specifies the first descriptor at which to start looking. It defaults to 20, but may be set higher if there are conflicts with other code.

`--stdlog-fd integer`

`--stdlog-fd=integer`

By default the log stream *stdlog* is sent to the *stderr* stream. This may be changed with this option. The specified file descriptor must already be open.

`--log-level level`

`--log-level=level`

Set the logging level. It defaults to `warn`. It may be one of
`error warn notice info debug`

FUNCTIONS

Errors

All functions return a status code, and push error messages onto the `logrd_ERRORS` array. There may be multiple error messages in the array; the messages at the end of the array arise from higher up the calling sequence within the library.

`logrd_has-error`

```
if logrd_has-error ; then
  handle errors
fi
```

Returns true if the `logrd_ERRORS` array contains errors.

Stream functions

There are three streams upon which **logrd** operates:

`stdout`

the standard output stream (file descriptor 1)

`stderr`

the standard error stream (file descriptor 2)

`stdlog`

the "standard logging" stream. By default this writes to *stderr*. Use *logrd-get* to obtain its file descriptor.

logrd-redirect-streams

```
logrd-redirect-streams [-fd|-file] target [global options] stream [ stream
options ] ...
```

Redirect one or more streams to a *target*. A target may be either a file descriptor or a file name. It defaults to a filename; use the `-fd` option to indicate it is a file descriptor.

The following options are available. If they are specified *before* the streams, they will apply to all streams, otherwise they apply to the stream which precedes them.

```
--copy-to-console
```

```
--no-copy-to-console
```

Copy (or don't) redirected streams to the console (that is, to the streams in existence when **logrd** was loaded). For example, this

```
logrd-redirect-stream file --copy-to-console stdout stderr
```

causes all output sent to *stdout* and *stderr* to be written to *file* as well as to the console *stdout* and *stderr* streams.

```
--copy-to-stream
```

```
--no-copy-to-stream
```

Copy (or don't) redirected streams to their original destination as well as the new one.

For example,

```
logrd-redirect-stream file1 stdout
logrd-redirect-stream file2 --copy-to-stream stdout
```

causes output sent to *stdout* to be written to both *file1* and *file2*

logrd-restore-streams

```
logrd-restore-streams stream stream
```

Restore the streams to their state when **logrd** was loaded.

Logging functions**log-to**

```
log-to $log_level $message
```

Write the message to the *stdlog* stream if the log level is at least that specified. The *log-level* may be one of

```
error warn notice info debug
```

The current log level may be obtained via *logrd-get*

log-error**log-warn****log-notice**

log-info**log-debug**

```
log-error message
log-warn message
log-notice message
log-info message
log-debug message
```

Write the message to the *stdlog* stream if the log level is at least that specified.

logrd-format-message

```
logrd-format-message $level $message
```

Format a message which is to be logged at level *\$level* and write it to the standard output stream. This function is used by **logrd** to format log messages. It may be redefined to change the behavior from the default, which is to output the message without transformation.

Global Attributes**logrd-set**

```
logrd-set $attribute $value
```

Set a global attribute. The following attributes are available:

level

Set the logging level; it may be one of

```
error warn notice info debug
```

logrd-get

```
logrd-get attribute
```

Retrieve an attribute. Boolean attributes are returned as status values, e.g.

```
logrd-get copy_to_console && echo 'global copy_to_console is set!'
```

Other attributes are output to *stdout*, e.g.

```
log_level=$(logrd-get level)
```

The attributes are

level

The logging level

copy_to_console

A boolean which is true if the global *copy_to_console* flag is set.

copy_to_stream

A boolean which is true if the global *copy_to_stream* flag is set.

copied_to_console

`logrd-get copied_to_console stream`

True if the specified *stream* is being copied to the console.

`starting_save_fd`

The starting file descriptor used. Pertinent only for older versions of **bash**

`stdlog`

The file descriptor for the *stdlog* stream.

ENVIRONMENT

Default values for global parameters may be read from environment variables. By default the variables have a prefix of `LOGRD_` (note the trailing underscore); that may be changed by the `--env-prefix` option when loading **logrd**

The following variables are recognized; see the similarly named options in *Loading* for more information

```
<PREFIX>COPY_TO_CONSOLE  
<PREFIX>COPY_TO_STREAM  
<PREFIX>STARTING_SAVE_FD  
<PREFIX>STDLOG_FD  
<PREFIX>LOG_LEVEL
```

AUTHOR

Diab Jerius <djerius@cfa.harvard.edu>

COPYRIGHT AND LICENSE

This software is Copyright (c) 2016 by Smithsonian Astrophysical Observatory.

This is free software, licensed under:

The GNU General Public License, Version 3, June 2007