ferror

1.2.5

Generated by Doxygen 1.8.11

Contents

1 Main Page 1

1.1 Introduction . . . 1

2 Modules Index 2

2.1 Modules List . . . 2

3 Data Type Index 2

3.1 Data Types List . . . 2

4 Module Documentation 2

4.1 ferror Module Reference . . . 2

4.1.1 Detailed Description . . . 3

4.1.2 Function/Subroutine Documentation . . . 3

4.2 ferror_c_binding Module Reference . . . 7

4.2.1 Detailed Description . . . 8

4.2.2 Function/Subroutine Documentation . . . 8

5 Data Type Documentation 13

5.1 ferror_c_binding::errorhandler Type Reference . . . 13

5.1.1 Detailed Description . . . 13

5.2 ferror::errors Type Reference . . . 13

5.2.1 Detailed Description . . . 15

Index 17

1 Main Page

1.1 Introduction

FERROR is a library to assist with error handling in Fortran projects.

Author

Jason Christopherson

Version

1.2.5

2 Modules Index

2.1 Modules List

Here is a list of all documented modules with brief descriptions:

ferror
ferror 2

ferror_c_binding
ferror 7

3 Data Type Index

3.1 Data Types List

Here are the data types with brief descriptions:

ferror_c_binding::errorhandler
A C compatible type encapsulating an errors object 13

ferror::errors
Defines a type for managing errors and warnings 13

4 Module Documentation

4.1 ferror Module Reference

ferror

Data Types

- type errors

  Defines a type for managing errors and warnings.

Functions/Subroutines

- pure character(len=:) function, allocatable er_get_log_filename (this)

  Gets the name of the error log file.
- subroutine er_set_log_filename (this, str)

  Sets the name of the error log file.
- subroutine er_report_error (this, fcn, msg, flag)

  Reports an error condition to the user.
- subroutine er_report_warning (this, fcn, msg, flag)

  Reports a warning message to the user.
- subroutine er_log_error (this, fcn, msg, flag)

  Writes an error log file.
- pure logical function er_has_error_occurred (this)

  Tests to see if an error has been encountered.
- subroutine er_reset_error_status (this)

  Resets the error status flag to false, and the current error flag to zero.
- pure logical function er_has_warning_occurred (this)

  Tests to see if a warning has been encountered.
- subroutine er_reset_warning_status (this)

  Resets the warning status flag to false, and the current warning flag to zero.
- pure integer(int32) function er_get_error_flag (this)

  Gets the current error flag.
- pure integer(int32) function er_get_warning_flag (this)

  Gets the current warning flag.
- pure logical function er_get_exit_on_error (this)

  Gets a logical value determining if the application should be terminated when an error is encountered.
- subroutine er_set_exit_on_error (this, x)

  Sets a logical value determining if the application should be terminated when an error is encountered.
- pure logical function er_get_suppress_printing (this)

  Gets a logical value determining if printing of error and warning messages should be suppressed.
- subroutine er_set_suppress_printing (this, x)

  Sets a logical value determining if printing of error and warning messages should be suppressed.

4.1.1 Detailed Description

ferror

Purpose

Provides a series of error codes and error handling mechanisms.

4.1.2 Function/Subroutine Documentation

4.1.2.1 pure integer(int32) function ferror::er_get_error_flag (class(errors), intent(in) *this*) [private]

Gets the current error flag.

Parameters

in	*this*	The errors object.

Returns

The current error flag.

4.1.2.2 pure logical function ferror::er_get_exit_on_error (class(errors), intent(in) *this*) [private]

Gets a logical value determining if the application should be terminated when an error is encountered.

Parameters

in	*this*	The errors object.

Returns

Returns true if the application should be terminated; else, false.

4.1.2.3 pure character(len = :) function, allocatable ferror::er_get_log_filename (class(errors), intent(in) *this*)

Gets the name of the error log file.

Parameters

in	*this*	The errors object.

Returns

The filename.

4.1.2.4 pure logical function ferror::er_get_suppress_printing (class(errors), intent(in) *this*) [private]

Gets a logical value determining if printing of error and warning messages should be suppressed.

Parameters

in	*this*	The errors object.

Returns

True if message printing should be suppressed; else, false to allow printing.

4.1.2.5 pure integer(int32) function ferror::er_get_warning_flag (class(errors), intent(in) *this*) [private]

Gets the current warning flag.

Parameters

in	*this*	The errors object.

Returns

The current warning flag.

4.1.2.6 pure logical function ferror::er_has_error_occurred (class(errors), intent(in) *this*) [private]

Tests to see if an error has been encountered.

Parameters

in	*this*	The errors object.

Returns

Returns true if an error has been encountered; else, false.

4.1.2.7 pure logical function ferror::er_has_warning_occurred (class(errors), intent(in) *this*) [private]

Tests to see if a warning has been encountered.

Parameters

in	*this*	The errors object.

Returns

Returns true if a warning has been encountered; else, false.

4.1.2.8 subroutine ferror::er_log_error (class(errors), intent(in) *this,* character(len = ∗), intent(in) *fcn,* character(len = ∗), intent(in) *msg,* integer(int32), intent(in) *flag*) [private]

Writes an error log file.

Parameters

in	*this*	The errors object.
in	*fcn*	The name of the function or subroutine in which the error was encountered.
in	*msg*	The error message.
in	*flag*	The error flag.

4.1.2.9 subroutine ferror::er_report_error (class(errors), intent(inout) *this,* character(len = ∗), intent(in) *fcn,* character(len = ∗), intent(in) *msg,* integer(int32), intent(in) *flag*) [private]

Reports an error condition to the user.

Parameters

in,out	*this*	The errors object.
in	*fcn*	The name of the function or subroutine in which the error was encountered.

Parameters

in	*msg*	The error message.
in	*flag*	The error flag.

Remarks

The default behavior prints an error message, appends the supplied information to a log file, and terminates the program.

4.1.2.10 subroutine ferror::er_report_warning (class(errors), intent(inout) *this,* character(len = ∗), intent(in) *fcn,* character(len = ∗), intent(in) *msg,* integer(int32), intent(in) *flag*) [private]

Reports a warning message to the user.

Parameters

in,out	*this*	The errors object.
in	*fcn*	The name of the function or subroutine from which the warning was issued.
in	*msg*	The warning message.
in	*flag*	The warning flag.

Remarks

The default behavior prints the warning message, and returns control back to the calling code.

4.1.2.11 subroutine ferror::er_reset_error_status (class(errors), intent(inout) *this*) [private]

Resets the error status flag to false, and the current error flag to zero.

Parameters

in,out	*this*	The errors object.

4.1.2.12 subroutine ferror::er_reset_warning_status (class(errors), intent(inout) *this*) [private]

Resets the warning status flag to false, and the current warning flag to zero.

Parameters

in,out	*this*	The errors object.

4.1.2.13 subroutine ferror::er_set_exit_on_error (class(errors), intent(inout) *this,* logical, intent(in) *x*) [private]

Sets a logical value determining if the application should be terminated when an error is encountered.

Parameters

in,out	*this*	The errors object.
in	*x*	Set to true if the application should be terminated when an error is reported; else, false.

4.1.2.14 subroutine ferror::er_set_log_filename (class(errors), intent(inout) *this*, character(len = :), allocatable *str*) [private]

Sets the name of the error log file.

Parameters

in,out	*this*	The errors object.
in	*str*	The filename.

4.1.2.15 subroutine ferror::er_set_suppress_printing (class(errors), intent(inout) *this*, logical, intent(in) *x*) [private]

Sets a logical value determining if printing of error and warning messages should be suppressed.

Parameters

in,out	*this*	The errors object.
in	*x*	Set to true if message printing should be suppressed; else, false to allow printing.

4.2 ferror_c_binding Module Reference

ferror

Data Types

- type errorhandler

  A C compatible type encapsulating an errors object.

Functions/Subroutines

- subroutine alloc_errorhandler (obj)

  Initializes a new error handler object.
- subroutine free_errorhandler (obj)

  Frees resources held by the errorhandler object.
- subroutine get_errorhandler (obj, eobj)

  Retrieves the errors object from the C compatible data structure.
- subroutine get_log_filename (err, fname, nfname)

  Gets the name of the error log file.
- subroutine set_log_filename (err, fname)

  Sets the error log filename.
- subroutine report_error (err, fcn, msg, flag)

Reports an error condition to the user.

- subroutine report_warning (err, fcn, msg, flag)

  Reports a warning condition to the user.
- subroutine log_error (err, fcn, msg, flag)

  Writes an error log file.
- logical(c_bool) function has_error_occurred (err)

  Tests to see if an error has been encountered.
- subroutine reset_error_status (err)

  Resets the error status flag to false, and the current error flag to zero.
- logical(c_bool) function has_warning_occurred (err)

  Tests to see if a warning has been encountered.
- subroutine reset_warning_status (err)

  Resets the warning status flag to false, and the current warning flag to zero.
- integer(c_int) function get_error_flag (err)

  Gets the current error flag.
- integer(c_int) function get_warning_flag (err)

  Gets the current warning flag.
- logical(c_bool) function get_exit_on_error (err)

  Gets a logical value determining if the application should be terminated when an error is encountered.
- subroutine set_exit_on_error (err, x)

  Sets a logical value determining if the application should be terminated when an error is encountered.
- logical(c_bool) function get_suppress_printing (err)

  Gets a logical value determining if printing of error and warning messages should be suppressed.
- subroutine set_suppress_printing (err, x)

  Sets a logical value determining if printing of error and warning messages should be suppressed.
- character(len=:) function, allocatable cstr_2_fstr (cstr)

  Copies a C string (null terminated) to a Fortran string.
- subroutine fstr_2_cstr (fstr, cstr, csize)

  Copies a Fortran string into a C string.

4.2.1 Detailed Description

ferror

Purpose

Provides C bindings to the ferror library.

4.2.2 Function/Subroutine Documentation

4.2.2.1 subroutine ferror_c_binding::alloc_errorhandler (type(errorhandler), intent(inout) *obj*)

Initializes a new error handler object.

Parameters

in	*obj*	The errorhandler object to allocate.

4.2.2.2 character(len = :) function, allocatable ferror_c_binding::cstr_2_fstr (character(kind = c_char), dimension(*), intent(in) *cstr*)

Copies a C string (null terminated) to a Fortran string.

Parameters

in	*cstr*	The null-terminated C string.

Returns

The Fortran copy.

4.2.2.3 subroutine ferror_c_binding::free_errorhandler (type(**errorhandler**), intent(inout), target *obj*)

Frees resources held by the errorhandler object.

Parameters

in,out	*obj*	The errorhandler object.

4.2.2.4 subroutine ferror_c_binding::fstr_2_cstr (character(len = *), intent(in) *fstr,* character(kind = c_char), dimension(*), intent(out) *cstr,* integer, intent(inout) *csize*)

Copies a Fortran string into a C string.

Parameters

in	*fstr*	The Fortran string to copy.
out	*cstr*	The null-terminated C string.
in,out	*csize*	On input, the size of the character buffer `cstr`. On output, the actual number of characters (not including the null character) writen to `cstr`.

4.2.2.5 integer(c_int) function ferror_c_binding::get_error_flag (type(**errorhandler**), intent(in) *err*)

Gets the current error flag.

Parameters

in	*err*	The errorhandler object.

Returns

The current error flag.

4.2.2.6 subroutine ferror_c_binding::get_errorhandler (type(**errorhandler**), intent(in), target *obj,* type(**errors**), intent(out), pointer *eobj*)

Retrieves the errors object from the C compatible data structure.

Parameters

in	*obj*	The C compatible errorhandler data structure.
out	*eobj*	The resulting errors object.

4.2.2.7 logical(c_bool) function ferror_c_binding::get_exit_on_error (type(errorhandler), intent(in) *err*)

Gets a logical value determining if the application should be terminated when an error is encountered.

Parameters

in	*err*	The errorhandler object.

Returns

Returns true if the application should be terminated; else, false.

4.2.2.8 subroutine ferror_c_binding::get_log_filename (type(errorhandler), intent(in) *err,* character(kind = c_char), dimension(*), intent(out) *fname,* integer(c_int), intent(inout) *nfname*)

Gets the name of the error log file.

Parameters

in	*err*	The errorhandler object.
out	*fname*	A character buffer where the filename will be written. It is recommended that this be in the neighborhood of 256 elements.
in,out	*nfname*	On input, the actual size of the buffer. Be sure to leave room for the null terminator character. On output, the actual numbers of characters written to `fname` (not including the null character).

4.2.2.9 logical(c_bool) function ferror_c_binding::get_suppress_printing (type(errorhandler), intent(in) *err*)

Gets a logical value determining if printing of error and warning messages should be suppressed.

Parameters

in	*err*	The errorhandler object.

Returns

True if message printing should be suppressed; else, false to allow printing.

4.2.2.10 integer(c_int) function ferror_c_binding::get_warning_flag (type(errorhandler), intent(in) *err*)

Gets the current warning flag.

Parameters

in	*err*	The errorhandler object.

Returns

The current warning flag.

4.2.2.11 logical(c_bool) function ferror_c_binding::has_error_occurred (type(errorhandler), intent(in) *err*)

Tests to see if an error has been encountered.

Parameters

in	*err*	A pointer to the error handler object.

Returns

Returns true if an error has been encountered; else, false.

4.2.2.12 logical(c_bool) function ferror_c_binding::has_warning_occurred (type(errorhandler), intent(in) *err*)

Tests to see if a warning has been encountered.

Parameters

in	*err*	The errorhandler object.

Returns

Returns true if a warning has been encountered; else, false.

4.2.2.13 subroutine ferror_c_binding::log_error (type(errorhandler), intent(in) *err,* character(kind = c_char), intent(in) *fcn,* character(kind = c_char), intent(in) *msg,* integer(c_int), intent(in), value *flag*)

Writes an error log file.

Parameters

in	*err*	The errorhandler object.
in	*fcn*	The name of the function or subroutine in which the error was encountered.
in	*msg*	The error message.
in	*flag*	The error flag.

4.2.2.14 subroutine ferror_c_binding::report_error (type(errorhandler), intent(inout) *err,* character(kind = c_char), intent(in) *fcn,* character(kind = c_char), intent(in) *msg,* integer(c_int), intent(in), value *flag*)

Reports an error condition to the user.

Parameters

in,out	*err*	A pointer to the error handler object.
in	*fcn*	The name of the function or subroutine in which the error was encountered.
in	*msg*	The error message.
in	*flag*	The error flag.

4.2.2.15 subroutine ferror_c_binding::report_warning (type(errorhandler), intent(inout) *err,* character(kind = c_char), intent(in) *fcn,* character(kind = c_char), intent(in) *msg,* integer(c_int), intent(in), value *flag*)

Reports a warning condition to the user.

Parameters

in,out	*err*	The errorhandler object.
in	*fcn*	The name of the function or subroutine in which the warning was encountered.
in	*msg*	The warning message.
in	*flag*	The warning flag.

4.2.2.16 subroutine ferror_c_binding::reset_error_status (type(errorhandler), intent(inout) *err*)

Resets the error status flag to false, and the current error flag to zero.

Parameters

in,out	*err*	The errorhandler object.

4.2.2.17 subroutine ferror_c_binding::reset_warning_status (type(errorhandler), intent(inout) *err*)

Resets the warning status flag to false, and the current warning flag to zero.

Parameters

in,out	*err*	The errorhandler object.

4.2.2.18 subroutine ferror_c_binding::set_exit_on_error (type(errorhandler), intent(inout) *err,* logical(c_bool), intent(in), value *x*)

Sets a logical value determining if the application should be terminated when an error is encountered.

Parameters

in,out	*err*	The errorhandler object.
in	*x*	Set to true if the application should be terminated when an error is reported; else, false.

4.2.2.19 subroutine ferror_c_binding::set_log_filename (type(errorhandler), intent(inout) *err,* character(kind = c_char), dimension(∗), intent(in) *fname*)

Sets the error log filename.

Parameters

in,out	*err*	The errorhandler object.
in	*fname*	A null-terminated string containing the filename.

4.2.2.20 subroutine ferror_c_binding::set_suppress_printing (type(errorhandler), intent(inout) *err,* logical(c_bool), intent(in), value *x*)

Sets a logical value determining if printing of error and warning messages should be suppressed.

Parameters

in,out	*err*	The errorhandler object.
in	*x*	Set to true if message printing should be suppressed; else, false to allow printing.

5 Data Type Documentation

5.1 ferror_c_binding::errorhandler Type Reference

A C compatible type encapsulating an errors object.

Public Attributes

- type(c_ptr) ptr

  A pointer to the errors object.
- integer(c_int) n

  The size of the errors object, in bytes.

5.1.1 Detailed Description

A C compatible type encapsulating an errors object.

The documentation for this type was generated from the following file:

- /home/jason/Documents/Code/ferror/src/ferror_c_binding.f90

5.2 ferror::errors Type Reference

Defines a type for managing errors and warnings.

Public Member Functions

- procedure, public get_log_filename => er_get_log_filename

  Gets the name of the error log file.
- procedure, public set_log_filename => er_set_log_filename

  Sets the name of the error log file.
- procedure, public report_error => er_report_error

  eports an error condition to the user.
- procedure, public report_warning => er_report_warning

  Reports a warning message to the user.
- procedure, public log_error => er_log_error

  Writes an error log file.
- procedure, public has_error_occurred => er_has_error_occurred

  Tests to see if an error has been encountered.
- procedure, public reset_error_status => er_reset_error_status

  Resets the error status flag to false.
- procedure, public has_warning_occurred => er_has_warning_occurred

  Tests to see if a warning has been encountered.
- procedure, public reset_warning_status => er_reset_warning_status

  Resets the warning status flag to false.
- procedure, public get_error_flag => er_get_error_flag

  Gets the current error flag.
- procedure, public get_warning_flag => er_get_warning_flag

  Gets the current warning flag.
- procedure, public get_exit_on_error => er_get_exit_on_error

  Gets a logical value determining if the application should be terminated when an error is encountered.
- procedure, public set_exit_on_error => er_set_exit_on_error

  Sets a logical value determining if the application should be terminated when an error is encountered.
- procedure, public get_suppress_printing => er_get_suppress_printing

  Gets a logical value determining if printing of error and warning messages should be suppressed.
- procedure, public set_suppress_printing => er_set_suppress_printing

  Sets a logical value determining if printing of error and warning messages should be suppressed.

Public Attributes

- character(len=256) m_fname = "error_log.txt"

  A maximum of 256 character error log filename.
- logical m_founderror = .false.

  Found an error.
- logical m_foundwarning = .false.

  Found a warning.
- integer(int32) m_errorflag = 0

  The error flag.
- integer(int32) m_warningflag = 0

  The warning flag.
- logical m_exitonerror = .true.

  Terminate the application on error.
- logical m_suppressprinting = .false.

  Suppress printing of error and warning messages.

5.2.1 Detailed Description

Defines a type for managing errors and warnings.

The documentation for this type was generated from the following file:

- /home/jason/Documents/Code/ferror/src/ferror.f90

Index

alloc_errorhandler
    ferror_c_binding, 8

cstr_2_fstr
    ferror_c_binding, 8

er_get_error_flag
    ferror, 3
er_get_exit_on_error
    ferror, 4
er_get_log_filename
    ferror, 4
er_get_suppress_printing
    ferror, 4
er_get_warning_flag
    ferror, 4
er_has_error_occurred
    ferror, 5
er_has_warning_occurred
    ferror, 5
er_log_error
    ferror, 5
er_report_error
    ferror, 5
er_report_warning
    ferror, 6
er_reset_error_status
    ferror, 6
er_reset_warning_status
    ferror, 6
er_set_exit_on_error
    ferror, 6
er_set_log_filename
    ferror, 7
er_set_suppress_printing
    ferror, 7

ferror, 2
    er_get_error_flag, 3
    er_get_exit_on_error, 4
    er_get_log_filename, 4
    er_get_suppress_printing, 4
    er_get_warning_flag, 4
    er_has_error_occurred, 5
    er_has_warning_occurred, 5
    er_log_error, 5
    er_report_error, 5
    er_report_warning, 6
    er_reset_error_status, 6
    er_reset_warning_status, 6
    er_set_exit_on_error, 6
    er_set_log_filename, 7
    er_set_suppress_printing, 7
ferror::errors, 13
ferror_c_binding, 7
    alloc_errorhandler, 8
    cstr_2_fstr, 8
    free_errorhandler, 9
    fstr_2_cstr, 9
    get_error_flag, 9
    get_errorhandler, 9
    get_exit_on_error, 10
    get_log_filename, 10
    get_suppress_printing, 10
    get_warning_flag, 10
    has_error_occurred, 11
    has_warning_occurred, 11
    log_error, 11
    report_error, 11
    report_warning, 12
    reset_error_status, 12
    reset_warning_status, 12
    set_exit_on_error, 12
    set_log_filename, 12
    set_suppress_printing, 13
ferror_c_binding::errorhandler, 13
free_errorhandler
    ferror_c_binding, 9
fstr_2_cstr
    ferror_c_binding, 9

get_error_flag
    ferror_c_binding, 9
get_errorhandler
    ferror_c_binding, 9
get_exit_on_error
    ferror_c_binding, 10
get_log_filename
    ferror_c_binding, 10
get_suppress_printing
    ferror_c_binding, 10
get_warning_flag
    ferror_c_binding, 10

has_error_occurred
    ferror_c_binding, 11
has_warning_occurred
    ferror_c_binding, 11

log_error
    ferror_c_binding, 11

report_error
    ferror_c_binding, 11
report_warning
    ferror_c_binding, 12
reset_error_status
    ferror_c_binding, 12
reset_warning_status
    ferror_c_binding, 12

set_exit_on_error

ferror_c_binding, 12
set_log_filename
ferror_c_binding, 12
set_suppress_printing
ferror_c_binding, 13