BacktraceException

Generated by Doxygen 1.8.11

Contents

1 Main Page

BacktraceException

`BacktraceException` is a C++ exception type that produces a stack backtrace when thrown. It can capture this backtrace with several methods and the backtrace can be disabled. Backtrace exception works on Linux.

Documentation

The BacktraceException Doxygen documentation can be build with the `OPT_DOC` CMake option and is also available on online:

- `BacktraceException HTML Manual`
- `BacktraceException PDF Manual`
- `BacktraceException github repository`

Background

Some exceptions are never meant to be thrown in the absence of programming or system error. Such an exception is thrown, it is normally passed far up the stack before being handled. When debugging an application it can be useful to quickly identify where a critical exception is being thrown. However, once the high-level exception handler has received the critical exception, there may not be sufficient information to determine exactly which part of the execution is producing the exception.

Clearly a debugger can be used to catch the exception as it is being thrown and eventually trace down the problem. But, the process of opening up the debugger, setting catch-points or break-points, restarting the application/computation, and waiting for the failure to reoccur is tedious, especially if the error occurs minutes or hours after the program is started. A full debugging session is often overkill as well. A programmer actively debugging will often know exactly what the problem is immediately on inspection of the throwing line of code or upon a quick scan of the stack-backtrace.

In these situations, BacktraceException prevents the need to break out the debugger every time a critical exception escapes a numerical simulation or long-running application. The BacktraceException object will capture a stack backtrace as it is being constructed and just before it is thrown. The catching code can then log or output a very useful message giving a view of exactly what was happening when it all went south.

This scheme works well in interactive environments like Python and Matlab that are running compiled C++ numerical code. Rather than getting a mysterious `NumericalError("Non finite")` exception message three hours into the computation with no further explanation, you now can get a stack backtrace.

Features

- Linux: Can use `glibc backtrace`, fast efficient, moderately informative.
- Linux: Can use `gdb backtraces`: much slower, but includes arguments and other info not present in glibc backtraces. Uses `gdb info stack`.
  - GDB backtraces also print thread info returned by `gdb info thread`
- Windows: Currently cross-compiles but stack walker traces are not yet implemented.
- Used `Glibc demangling` for C++ symbols name
- Easily installed as a standalone package or built alongside CMake or autotools projects.

Using Backtrace Exception

A `BacktraceException` is used a a base class for application-defined classes of critical errors that would not be recoverable, and a backtrace of the exception throwing site would be useful.

```
struct UnrecoverableError : public BacktraceException {
    UnrecoverableError(std::string what) : BacktraceException("UnrecoverableError",what) {}
};
```

- Use `backtrace_exception::enable_backtraces()` or `backtrace_exception::disable_backtraces()` to control the generation of backtraces. When disabled backtrace_exception behaves like a normal `std::exception`.
- Use `backtrace_exception::set_backtrace_method()` and `backtrace_exception::get_backtrace_method()` to set the method used for backtracing.
- Build examples from the examples sub-directory with CMake option `OPT_EXAMPLES=1`

  $ ./build.debug.sh -DOPT_EXAMPLES=1 $ ./_build/Debug/examples/backtrace_exception_example1

Limitations

- In order to get a backtrace, the exception must derive from `BacktraceException`. Exceptions from third-party libraries or based on *`std::exception`* will not generate a backtrace.
- Windows 64-bit backtrace support is planed in the next release. Currently you can cross-compile applications using `BacktraceException` to Win64, but backtraces are disabled.

CMake configuration options

- `BUILD_SHARED_LIBS` - Build shared libraries [Default: On]
- `BUILD_STATIC_LIBS` - Build static libraries [Default: On]
- `BUILD_TESTING` - Build tests [Default: On if BUILD_TYPE==Debug]
- `OPT_INSTALL_TESTING` - Install tests.
- `OPT_DOC` - Build and install documentation (enables `make doc` and `make pdf`) [Default: Off]
- `OPT_EXAMPLES` - Build examples [Default: Off]
- `OPT_EXPORT_BUILD_TREE` - Enable CMake export and `find_package(BacktraceException)` support from the build-tree.

2 Namespace Index

2.1 Namespace List

Here is a list of all namespaces with brief descriptions:

3 Hierarchical Index

3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

4 Class Index

4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

5 File Index

5.1 File List

Here is a list of all files with brief descriptions:

6 Namespace Documentation

6.1 backtrace_exception Namespace Reference

Classes

- class BacktraceException

  Extension of std::exception that produces saved backtraces for debugging.

Enumerations

- enum BacktraceMethod { BacktraceMethod::glibc, BacktraceMethod::gdb, BacktraceMethod::stackwalk }

Functions

- BacktraceMethod get_backtrace_method ()
- void set_backtrace_method (BacktraceMethod method)
- void disable_backtraces ()
- void enable_backtraces ()
- bool backtraces_enabled ()

6.1.1 Enumeration Type Documentation

6.1.1.1 enum backtrace_exception::BacktraceMethod [strong]

Enumerator

glibc

gdb

stackwalk

Definition at line 16 of file BacktraceException.h.

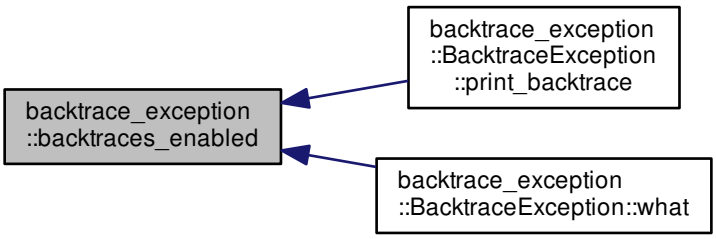
6.1.2 Function Documentation

6.1.2.1 bool backtrace_exception::backtraces_enabled ( )

Definition at line 84 of file BacktraceException.cpp.

Referenced by backtrace_exception::BacktraceException::print_backtrace(), and backtrace_exception::BacktraceException::what().

Here is the caller graph for this function:



6.1.2.2 void backtrace_exception::disable_backtraces ( )

Definition at line 74 of file BacktraceException.cpp.

6.1.2.3 void backtrace_exception::enable_backtraces ( )

Definition at line 79 of file BacktraceException.cpp.

6.1.2.4 BacktraceMethod backtrace_exception::get_backtrace_method ( )

Definition at line 46 of file BacktraceException.cpp.

6.1.2.5 void backtrace_exception::set_backtrace_method (BacktraceMethod *method*)

Definition at line 51 of file BacktraceException.cpp.

References gdb, glibc, and stackwalk.

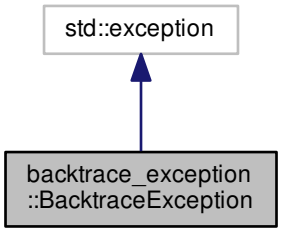
7 Class Documentation

7.1 backtrace_exception::BacktraceException Class Reference

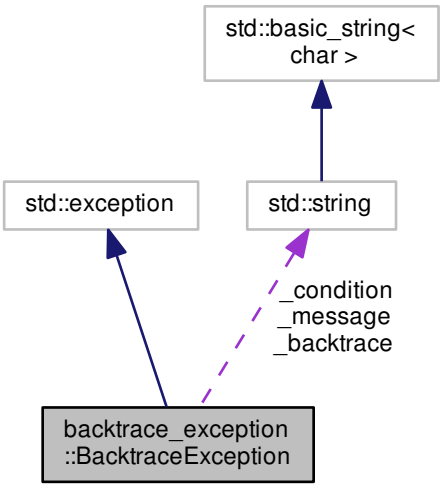
Extension of std::exception that produces saved backtraces for debugging.

`#include </home/travis/build/markjolah/BacktraceException/include/Backtrace↩ Exception/BacktraceException.h>`

Inheritance diagram for backtrace_exception::BacktraceException:



Collaboration diagram for backtrace_exception::BacktraceException:



Public Member Functions

- BacktraceException (std::string message)
- BacktraceException (std::string condition, std::string message)

  Create a BacktraceException with specified condition.
- virtual const char * condition () const noexcept
- virtual const char * message () const noexcept
- virtual const char * backtrace () const noexcept
- const char * what () const noexceptoverride

Static Public Member Functions

- static std::string print_backtrace ()

Protected Attributes

- std::string _condition
- std::string _message
- std::string _backtrace

7.1.1 Detailed Description

Extension of std::exception that produces saved backtraces for debugging.

Definition at line 28 of file BacktraceException.h.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 backtrace_exception::BacktraceException::BacktraceException (std::string *message*)

Definition at line 195 of file BacktraceException.cpp.

7.1.2.2 backtrace_exception::BacktraceException::BacktraceException (std::string *condition,* std::string *message*)

Create a BacktraceException with specified condition.

Parameters

condition	A string classifying the error condition
message	A message string describing the error condition.

Definition at line 199 of file BacktraceException.cpp.

7.1.3 Member Function Documentation

7.1.3.1 const char * backtrace_exception::BacktraceException::backtrace ( ) const [inline], [virtual], [noexcept]

Definition at line 59 of file BacktraceException.h.

7.1.3.2 const char * backtrace_exception::BacktraceException::condition ( ) const [inline], [virtual], [noexcept]

Definition at line 51 of file BacktraceException.h.

7.1.3.3 const char * backtrace_exception::BacktraceException::message ( ) const [inline], [virtual], [noexcept]

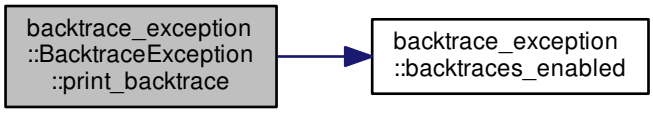
Definition at line 55 of file BacktraceException.h.

7.1.3.4 std::string backtrace_exception::BacktraceException::print_backtrace ( ) [static]

Definition at line 211 of file BacktraceException.cpp.

References backtrace_exception::backtraces_enabled(), backtrace_exception::gdb, and backtrace_exception::glibc.

Here is the call graph for this function:

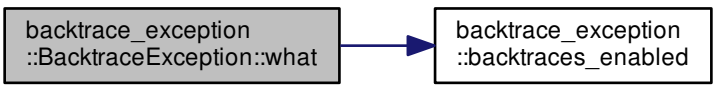


7.1.3.5 const char * backtrace_exception::BacktraceException::what ( ) const [override], [noexcept]

Definition at line 203 of file BacktraceException.cpp.

References _backtrace, _condition, _message, and backtrace_exception::backtraces_enabled().

Here is the call graph for this function:



7.1.4 Member Data Documentation

7.1.4.1 std::string backtrace_exception::BacktraceException::_backtrace [protected]

Definition at line 47 of file BacktraceException.h.

Referenced by what().

7.1.4.2 std::string backtrace_exception::BacktraceException::_condition [protected]

Definition at line 45 of file BacktraceException.h.

Referenced by what().

7.1.4.3 std::string backtrace_exception::BacktraceException::_message [protected]

Definition at line 46 of file BacktraceException.h.

Referenced by what().

The documentation for this class was generated from the following files:

- BacktraceException.h
- BacktraceException.cpp

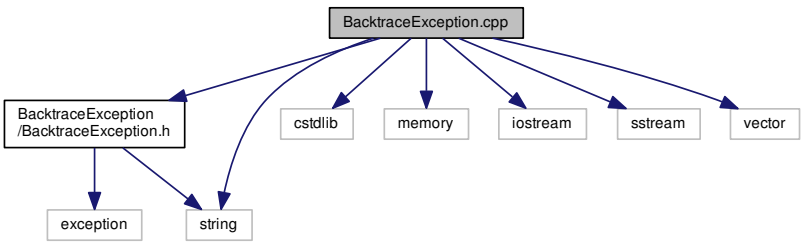
8 File Documentation

8.1 BacktraceException.cpp File Reference

BacktraceException class member function definitions.

```
#include "BacktraceException/BacktraceException.h"
#include <cstdlib>
#include <memory>
#include <iostream>
#include <string>
#include <sstream>
#include <vector>
```

Include dependency graph for BacktraceException.cpp:



Namespaces

- backtrace_exception

Functions

- BacktraceMethod backtrace_exception::get_backtrace_method ()
- void backtrace_exception::set_backtrace_method (BacktraceMethod method)
- void backtrace_exception::disable_backtraces ()
- void backtrace_exception::enable_backtraces ()
- bool backtrace_exception::backtraces_enabled ()

8.1.1 Detailed Description

BacktraceException class member function definitions.

Author

Mark J. Olah (mjo@cs.unm DOT edu)

Date

2017 - 2019

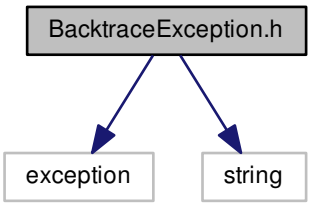
Copyright

8.2 BacktraceException.h File Reference

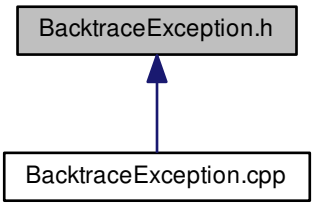
BacktraceException class declaration and inline member functions.

```
#include <exception>
#include <string>
```

Include dependency graph for BacktraceException.h:



This graph shows which files directly or indirectly include this file:



Classes

- class backtrace_exception::BacktraceException

  Extension of std::exception that produces saved backtraces for debugging.

Namespaces

- backtrace_exception

Enumerations

- enum backtrace_exception::BacktraceMethod { backtrace_exception::BacktraceMethod::glibc, backtrace_exception::BacktraceMethod::gdb, backtrace_exception::BacktraceMethod::stackwalk }

Functions

- void backtrace_exception::disable_backtraces ()
- void backtrace_exception::enable_backtraces ()
- bool backtrace_exception::backtraces_enabled ()
- BacktraceMethod backtrace_exception::get_backtrace_method ()
- void backtrace_exception::set_backtrace_method (BacktraceMethod method)

8.2.1 Detailed Description

BacktraceException class declaration and inline member functions.

Author

Mark J. Olah (mjo@cs.unm DOT edu)

Date

2017 - 2019

Copyright

8.3 README.md File Reference

Index