

## BacktraceException

Generated by Doxygen 1.8.11

## Contents

<b>1</b>	<b>Main Page</b>	<b>2</b>
<b>2</b>	<b>Namespace Index</b>	<b>4</b>
2.1	Namespace List . . . . .	4
<b>3</b>	<b>Hierarchical Index</b>	<b>4</b>
3.1	Class Hierarchy . . . . .	4
<b>4</b>	<b>Class Index</b>	<b>4</b>
4.1	Class List . . . . .	4
<b>5</b>	<b>File Index</b>	<b>4</b>
5.1	File List . . . . .	4
<b>6</b>	<b>Namespace Documentation</b>	<b>5</b>
6.1	backtrace_exception Namespace Reference . . . . .	5
6.1.1	Enumeration Type Documentation . . . . .	5
6.1.2	Function Documentation . . . . .	5
<b>7</b>	<b>Class Documentation</b>	<b>6</b>
7.1	backtrace_exception::BacktraceException Class Reference . . . . .	6
7.1.1	Detailed Description . . . . .	7
7.1.2	Constructor & Destructor Documentation . . . . .	8
7.1.3	Member Function Documentation . . . . .	8
7.1.4	Member Data Documentation . . . . .	9
<b>8</b>	<b>File Documentation</b>	<b>9</b>
8.1	BacktraceException.cpp File Reference . . . . .	9
8.1.1	Detailed Description . . . . .	10
8.2	BacktraceException.h File Reference . . . . .	10
8.2.1	Detailed Description . . . . .	11
8.3	README.md File Reference . . . . .	12

## 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 as 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 planned 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:

[backtrace\\_exception](#) 5

## 3 Hierarchical Index

### 3.1 Class Hierarchy

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

`std::exception`  
**`backtrace_exception::BacktraceException`** 6

## 4 Class Index

### 4.1 Class List

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

[backtrace\\_exception::BacktraceException](#)  
**Extension of `std::exception` that produces saved backtraces for debugging** 6

## 5 File Index

### 5.1 File List

Here is a list of all files with brief descriptions:

[BacktraceException.cpp](#)  
**BacktraceException class member function definitions** 9

[BacktraceException.h](#)  
**BacktraceException class declaration and inline member functions** 10

## 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.

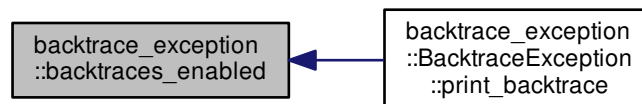
#### 6.1.2 Function Documentation

##### 6.1.2.1 bool backtrace\_exception::backtraces\_enabled ( )

Definition at line 84 of file BacktraceException.cpp.

Referenced by `backtrace_exception::BacktraceException::print_backtrace()`.

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.

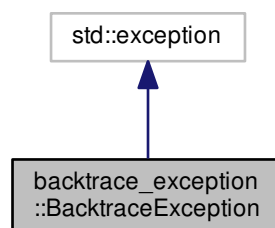
## 7 Class Documentation

### 7.1 backtrace\_exception::BacktraceException Class Reference

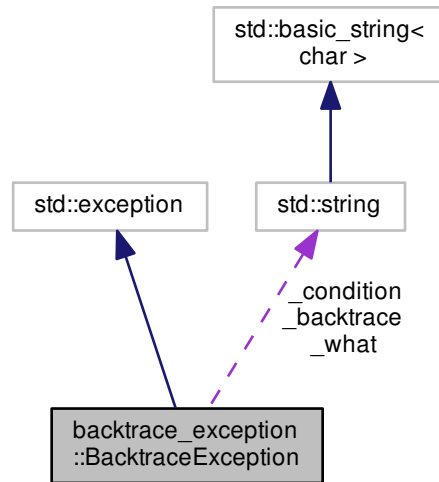
Extension of std::exception that produces saved backtraces for debugging.

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

Inheritance diagram for backtrace\_exception::BacktraceException:



Collaboration diagram for `backtrace_exception::BacktraceException`:



#### Public Member Functions

- [BacktraceException](#) (std::string [what](#))
- [BacktraceException](#) (std::string [condition](#), std::string [what](#))  
*Create a [BacktraceException](#) with specified condition.*
- virtual const char \* [condition](#) () const noexcept
- const char \* [what](#) () const noexcept override
- virtual const char \* [backtrace](#) () const noexcept

#### Static Public Member Functions

- static std::string [print\\_backtrace](#) ()

#### Protected Attributes

- std::string [\\_condition](#)
- std::string [\\_what](#)
- 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 ( std::string what )`

Definition at line 195 of file `BacktraceException.cpp`.

#### 7.1.2.2 `backtrace_exception::BacktraceException ( std::string condition, std::string what )`

Create a [BacktraceException](#) with specified condition.

##### Parameters

<i>condition</i>	A string further classifying the error condition
<i>what</i>	A general 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 58 of file `BacktraceException.h`.

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

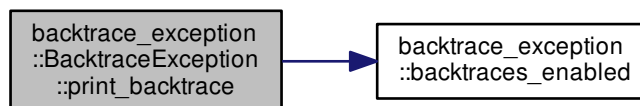
Definition at line 50 of file `BacktraceException.h`.

#### 7.1.3.3 `std::string backtrace_exception::BacktraceException::print_backtrace ( )` `[static]`

Definition at line 203 of file `BacktraceException.cpp`.

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

Here is the call graph for this function:



7.1.3.4 `const char * backtrace_exception::BacktraceException::what ( ) const` `[inline],[override],[noexcept]`

Definition at line 54 of file BacktraceException.h.

#### 7.1.4 Member Data Documentation

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

Definition at line 46 of file BacktraceException.h.

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

Definition at line 44 of file BacktraceException.h.

7.1.4.3 `std::string backtrace_exception::BacktraceException::_what` `[protected]`

Definition at line 45 of file BacktraceException.h.

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

- [BacktraceException.h](#)
- [BacktraceException.cpp](#)

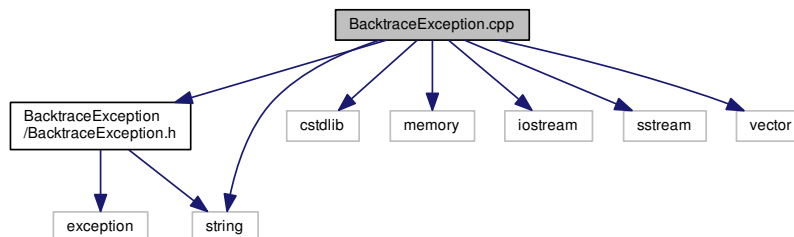
## 8 File Documentation

### 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 - 2018

#### Copyright

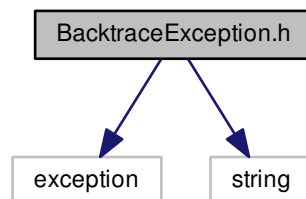
Licensed under the Apache License, Version 2.0. See LICENSE file.

## 8.2 BacktraceException.h File Reference

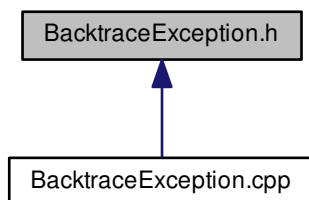
BacktraceException class declaration and inline member functions.

```
#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 - 2018

#### Copyright

Licensed under the Apache License, Version 2.0. See LICENSE file.

### 8.3 README.md File Reference

## Index

- `_backtrace`
    - `backtrace_exception::BacktraceException`, 9
  - `_condition`
    - `backtrace_exception::BacktraceException`, 9
  - `_what`
    - `backtrace_exception::BacktraceException`, 9
- `backtrace`
  - `backtrace_exception::BacktraceException`, 8
- `backtrace_exception`, 5
  - `BacktraceMethod`, 5
  - `backtraces_enabled`, 5
  - `disable_backtraces`, 5
  - `enable_backtraces`, 6
  - `gdb`, 5
  - `get_backtrace_method`, 6
  - `glibc`, 5
  - `set_backtrace_method`, 6
  - `stackwalk`, 5
- `backtrace_exception::BacktraceException`, 6
  - `_backtrace`, 9
  - `_condition`, 9
  - `_what`, 9
  - `backtrace`, 8
  - `BacktraceException`, 8
  - `condition`, 8
  - `print_backtrace`, 8
  - `what`, 8
- `BacktraceException`
  - `backtrace_exception::BacktraceException`, 8
- `BacktraceException.cpp`, 9
- `BacktraceException.h`, 10
- `BacktraceMethod`
  - `backtrace_exception`, 5
- `backtraces_enabled`
  - `backtrace_exception`, 5
- `condition`
  - `backtrace_exception::BacktraceException`, 8
- `disable_backtraces`
  - `backtrace_exception`, 5
- `enable_backtraces`
  - `backtrace_exception`, 6
- `gdb`
  - `backtrace_exception`, 5
- `get_backtrace_method`
  - `backtrace_exception`, 6
- `glibc`
  - `backtrace_exception`, 5
- `print_backtrace`
  - `backtrace_exception::BacktraceException`, 8
- `README.md`, 12
- `set_backtrace_method`
  - `backtrace_exception`, 6
- `stackwalk`
  - `backtrace_exception`, 5
- `what`
  - `backtrace_exception::BacktraceException`, 8