Stack Traces made easy

Down the C++ Rabbit Hole

About me

- Jonathan O’Connor
- Started C++ in 1988
- Switched to Java in 2000, Ruby in 2010
- Came back to Modern C++ 2015
- @ninkibah
- https://github.com/ninkibah

A journey into C++17

- RAII
- Limiting use of tracing class
- Auto logging of exceptions
- Variadic MACROS

Why

- Debugger wouldn’t work on code
- Signal raised
- Exception thrown
- No idea where

What makes C++ different

- Constructors
- Destructors
- Lots of languages have constructors: Java, Ruby, Python
- Very few have destructors
- Stack based vs Garbage Collected

Resource Acquisition

- RAII – ctor acquires, dtor releases
- Allocating and deallocating memory
- Opening and closing files
- Starting and committing transactions
- Acquiring locks

All about the scope

- Stack-based objects die when the scope ends
- Functions
- Begin-end blocks
- lambdas

Trace initial version

- Trace class remembers location of it’s creation.
  - __FILE__
  - __LINE__
  - __func__
- Linked list of nested Trace objects
- Dtor moves the head to the caller Trace.

Trace – TRACE Macro

- Trace trace{__FILE__, __func__, __LINE__};
- Simple macro TRACE

Trace – Better function traces

- __FUNCTION__
- __func__
- __PRETTY_FUNCTION__

Std::uncaught_exceptions()

- Normal return from function
- Stack unwinding because of exception
- C++17 int std::uncaught_exceptions()
- Pre-C++17 bool std::uncaught_exception()

Restricting uses of Trace

- No copying
  - Trace(Trace const&) = delete;
- No assignment
  - Trace& operator=(Trace const&) = delete;
- No allocating on the heap
  - Private void* operator new(size_t);

Multi-thread support

- thread_local
- static thread_local inline Trace* mostRecentCaller = nullptr;

Logging parameters

- TRACE(x, y)
- Trace trace(__FILE__, __func__, __LINE__, “x”, x, “y”, y)
- Need Variadic Macros
- Template function with parameter pack

Future directions

- Stack traces only show the location of the Trace objects
- execinfo.h
  - backtrace(void **frames, int nFrames)
  - backtrace_symbols(void **, int)
- Difficult to get unmangled names

Other approaches

- http://code-freeze.blogspot.ie/2012/01/generating-stack-traces-from-c.html
- execinfo.h
-

Summary

- RAII is great!
- __PRETTY_FUNCTION__
- Deleting standard functions
- Making operator new private
- thread_local inline
- Variadic macros – Yeuch!!!

Questions?

- https://github.com/ninkibah/trace
- Pull requests are very welcome