C++ Club Meeting Notes

Gleb Dolgich

2019-07-04

Elements of Programming Authors' Edition (free ebook)

http://componentsprogramming.com/elements-of-programming-authors-edition/

https://www.reddit.com/r/cpp/comments/c6fjjg/elements_of_programming_authors_edition/

> *Alex Stepanov and Paul McJones have just released Elements of Programming Authors' Edition.*

PDF download:

http://elementsofprogramming.com/

C++17 - The Complete Guide by Nicolai Josuttis

https://leanpub.com/cpp17

CLion 2019.2 EAP: MSVC Debugger, Unused Includes Check, and More

https://blog.jetbrains.com/clion/2019/06/clion-2019-2-eap-msvc-debugger-unused-includes-check-and-more/

- Experimental feature: LLDB-based Debugger for the Microsoft Visual C++ toolchain
- The ‘unused includes’ check is back
- Memory view: ASCII view
- Better performance for code completion

https://www.reddit.com/r/cpp/comments/c5vnhw/clion_20192_eap_brings_experimental_lldbbased/

A dbg(...) macro for C++

https://github.com/sharkdp/dbg-macro

https://www.reddit.com/r/cpp/comments/c2ysa7/a_dbg_macro_for_c/

https://doc.rust-lang.org/std/macro.dbg.html

Algorithms/Data Structure course for C++

- Stanford CS106B - Programming Abstractions
- MIT 6.006 Introduction to Algorithms, Fall 2011
- MIT 6.046J Design and Analysis of Algorithms, Spring 2015
- Alex Stepanov Efficient Programming with Components
- Udemy Mastering Data Structures & Algorithms using C and C++

mimalloc

Microsoft **mimalloc** is a compact general purpose allocator with excellent performance.

https://github.com/microsoft/mimalloc

https://www.reddit.com/r/programming/comments/c3ox2r/mimalloc_is_a_compact_general_purpose_allocator/

Mimalloc: Free List Sharding in Action

Serenity OS

https://github.com/SerenityOS/serenity (BSD-2-Clause)

https://www.reddit.com/r/programming/comments/c13vph/serenityos_a_marriage_between_the_aesthetic_of/

Serenity OS Patterns: The Badge

(aka The Client-Attorney Idiom)

https://awesomekling.github.io/Serenity-C++-patterns-The-Badge/

- Reddit
- SO: Granular friend
- Dr. Dobbs - Friendship and the Attorney-Client Idiom

```cpp
class Foo;
class Bar { public: void special(int a, Key<Foo>); };
Bar().special(1, {}); // at call site in Foo
```

Catching use-after-move bugs with Clang’s consumed annotations

Article by Andreas Kling | Reddit

- Clang consumed annotation checking

```cpp
class [[clang::consumable(unconsumed)]] CleverObject {
public:
  CleverObject() {}
  CleverObject(CleverObject&& other) { other.invalidate(); }
  [[clang::callable_when(unconsumed)]]
  void do_something() { assert(m_valid); }
private:
  [[clang::set_typestate(consumed)]]
  void invalidate() { m_valid = false; }
  bool m_valid { true };
};
```

Catching use-after-move bugs with Clang's consumed annotations

Article by Andreas Kling | Reddit

- Clang consumed annotation checking
- Clang-tidy bugprone-use-after-move

```cpp
class [[clang::consumable(unconsumed)]] CleverObject {
public:
  CleverObject() {}
  CleverObject(CleverObject&& other) { other.invalidate(); }
  [[clang::callable_when(unconsumed)]]
  void do_something() { assert(m_valid); }
private:
  [[clang::set_typestate(consumed)]]
  void invalidate() { m_valid = false; }
  bool m_valid { true };
};
```

What are some uses of decltype(auto)?

https://stackoverflow.com/questions/24109737/what-are-some-uses-of-decltypeauto

- https://stackoverflow.com/a/24109800/10154
- https://stackoverflow.com/a/24109944/10154

LibTom

https://www.libtom.net/

https://github.com/libtom/libtomcrypt

The Power of Hidden Friends in C++

Article by Anthony Williams

https://www.justsoftwaresolutions.co.uk/cplusplus/hidden-friends.html

```cpp
namespace A{
  class X{
  public:
    X(int i):data(i){}
  private:
    int data;
    friend bool operator==(X const& lhs,X const& rhs){
      return lhs.data==rhs.data;
    }
  };
}
```

strong_typedef - Create distinct types for distinct purposes

Article by Anthony Williams

https://www.justsoftwaresolutions.co.uk/cplusplus/strong_typedef.html

https://github.com/anthonywilliams/strong_typedef ()

```cpp
using transaction_id =
  jss::strong_typedef<struct transaction_tag, std::string>;

bool is_a_foo(transaction_id id)
{
  auto &s = id.underlying_value();
  return s.find("foo") != s.end();
}
```

Introducing the Rule of DesDeMovA

Blog post by Peter Sommerlad

https://blog.safecpp.com

https://accu.org/content/conf2014/Howard_Hinnant_Accu_2014.pdf

Rule of Zero:

> *Code that you do not write cannot be wrong.*

A closer look at **bake**: a tool that makes building C/C++ code effortless

https://medium.com/@cortoproject/a-closer-look-at-bake-a-tool-that-makes-building-c-c-code-effortless-b2e0409fad8f

- https://www.reddit.com/r/C_Programming/comments/a85f6w/meet_bake_a_new_build_system_package_manager_for/
- https://www.reddit.com/r/cpp/comments/a8d7ny/meet_bake_a_new_build_system_package_manager_for/
- https://news.ycombinator.com/item?id=18787777

https://github.com/SanderMertens/bake (GPLv3)

> *A cargo-like buildsystem and package manager for C/C++*

Magic.

Use constexpr for faster, smaller, and safer code

https://blog.trailofbits.com/2019/06/27/use-constexpr-for-faster-smaller-and-safer-code/

https://www.reddit.com/r/cpp/comments/c646ng/use_constexpr_for_faster_smaller_and_safer_code/

https://github.com/trailofbits/constexpr-everything (Apache 2.0)

How to try the new coroutines TS?

https://www.reddit.com/r/cpp/comments/c6ag3l/how_to_try_the_new_coroutines_ts/

MSVC

```
/await /std:c++latest
```

Clang

```
-std=c++2a -stdlib=libc++ -fcoroutines-ts
```

- CppCoro - https://github.com/lewissbaker/cppcoro
- coroutine - https://github.com/luncliff/coroutine
- continuable - https://github.com/Naios/continuable

Discussion: member variable naming

https://www.reddit.com/r/cpp/comments/c6rnel/discussion_member_variable_naming/

- `m_foo`
- `foo_`
- `_foo`

How do you get the benefits of Rust in C++?

https://www.reddit.com/r/cpp/comments/c6gtd4/how_do_you_get_the_benefits_of_rust_in_c/

How do C++ developers manage dependencies

https://www.reddit.com/r/cpp/comments/c6l3eg/how_do_c_developers_manage_dependencies/

> *Through much pain and anguish.*

Just started learning C++ coming from Python

https://www.reddit.com/r/cpp/comments/c6vnb3/just_started_learning_c_coming_from_python_and/

> *The new GCC compiler with colour highlighting is a little bit better at pointing out errors. It's generally quite helpful for pure C/C++ until you make an error with the standard library and you get 200 lines about std:: whatever<random characters>*
>
> *In C++ a trick I always use when the error message is massive is to just focus on the first error.*

Scott Meyers’ TD trick

https://www.reddit.com/r/cpp/comments/c6vnb3/just_started_learning_c_coming_from_python_and/eshq8vb?utm_source=share&utm_medium=web2x

```
template <typename T> struct TD; // no definition
```

Now you write something like `TD<decltype(thing)>` and the error message tells you the type of thing (as deduced by `decltype`, of course, but in this case that’s probably what you want).

Why std::expected is not in the standard yet? Is it bad practice?

https://www.reddit.com/r/cpp/comments/c75ipk/why_stdexpected_is_not_in_the_standard_yet_is_it/

- `std::expected` https://github.com/TartanLlama/expected
- Boost Outcome https://www.boost.org/doc/libs/1_70_0/libs/outcome/doc/html/index.html
- Outcome without Boost https://ned14.github.io/outcome/
- Leaf https://github.com/zajo/leaf

Go-like error handling in C++

https://github.com/hellozee/errors

https://www.reddit.com/r/cpp/comments/c7il5n/an_idiots_attempt_to_do_a_go_like_error_handling/

> *It looks like you invented something similar to `std::expected`.*

Simplify Your Code With Rocket Science: C++20’s Spaceship Operator

https://devblogs.microsoft.com/cppblog/simplify-your-code-with-rocket-science-c20s-spaceship-operator/

https://www.reddit.com/r/cpp/comments/c68457/simplify_your_code_with_rocket_science_c20s/

Better Ways to Test with **doctest** – the Fastest C++ Unit Testing Framework

https://blog.jetbrains.com/rscpp/better-ways-testing-with-doctest/

The Best Book to Read as a Developer

https://dev.to/taillogs/the-best-book-to-read-as-a-developer-1h4m

https://www.reddit.com/r/programming/comments/c8aaov/the_best_book_to_read_as_a_developer/

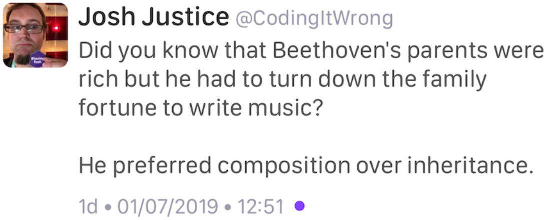
- Inside the Machine by Jon Stokes http://joe90.yolasite.com/resources/InsidetheMachine.pdf
- The Pragmatic Programmer
- ”Working Effectively with Legacy Code” by Michael Feathers
- Charles Petzold’s Code https://www.goodreads.com/book/show/44882.Code
- Tao of Programming http://canonical.org/~kragen/tao-of-programming.html
- Game Engine Architecture https://www.amazon.com/Game-Engine-Architecture-Jason-Gregory/dp/1568814135

Splitting a string in C++

https://medium.com/@bkey76/splitting-a-string-in-c-23e2547e6451

- C++ String Toolkit Library (MIT)
  http://www.partow.net/programming/strtk/index.html

Twitter



Josh Justice @CodingItWrong

Did you know that Beethoven's parents were rich but he had to turn down the family fortune to write music?

He preferred composition over inheritance.

1d • 01/07/2019 • 12:51 ●