C++ Club Meeting 104

Gleb Dolgich

2020-04-09

Follow-up: How I Declare My class And Why, by Howard Hinnant

- Howard Hinnant
- Reddit
- Coding guidelines

Order:

- data members
- destructor
- default constructor
- copy special members
- move special members
- other constructors
- other member functions

Follow-up: References, simply, by Herb Sutter

- Herb Sutter
- Reddit

- Microsoft
- Reddit

Epic 10-hour C++ port of Doom

- YouTube video by Jason Turner

- Home
- GitHub (C++17, MIT)

- GitHub
- Reddit

Continuable V4.0

C++14 asynchronous allocation aware futures (supporting then, exception handling, coroutines and connections), by Denis Blank

- GitHub (C++14, MIT)
- Docs
- Reddit

IceCream-Cpp

Sweet and creamy print debugging. C++ flavored, by Renato Garcia

- GitHub
- Reddit

- Daniel Martin

- Video
- Article by Arthur O'Dwyer
- Reddit

See also: Abseil Tip of the Week #107: Reference Lifetime Extension

```
1 std::string Foo::GetName();
2 const std::string& name = obj.GetName();  // Is this safe/legal?
```

A new decade, a new tool: libman

- Colby Pike (vector-of-bool)
- Reddit
- GitHub
- Specification

libman is a new level of indirection between package management and build systems.

dds is Drop-Dead Simple build and package manager.

- CppCon 2019: Robert Schumacher “How to Herd 1,000 Libraries”

“Making new friends” idiom by Dan Saks

Wikibooks

> *The goal is to simplify creation of friend functions for a class template.*

```
1  #include <iostream>
2  template<typename T>
3  class Foo {
4     T value;
5  public:
6     Foo(const T& t) { value = t; }
7     friend std::ostream& operator <<(std::ostream& os, const Foo<T>& b)
8     {
9        return os << b.value;
10    }
11 };
```

A hidden gem: `inner_product` (1/2)

- Article

A hidden gem: `inner_product` (2/2)

4

Conor Hoekstra @code_report
@cjdb_ns & @TartanLlama

This makes me so incredibly happy! I literally just yesterday googled, C++17 / C++20 zip to see if they had anything, because I wrote some code in both C++ and #Python and Python was so much more beautiful.

```
int solve(int h, vector<int> w, vector<int> l) {
    int p = 0;
    for (int i = 0; i < w.size(); ++i)
        p = max(p, w[i] - l[i] / 4);
    return max(0, p - h);
}
```

```
def solve(h, w, l):
    p = max(a - b//4 for a, b in zip(w, l))
    return max(0, p - h)
```

29w • 03/12/2018 • 17:47

Conor Hoekstra @code_report
@cjdb_ns & @TartanLlama

Also, I just discovered std::inner_product - a beautiful temporary solution to a lack of zip. #cpp #inner_product

```
int solve(int h, vector<int> w, vector<int> l) {
    return max(0, inner_product(begin(w), end(w), begin(l), 0,
        [](auto a, auto b) { return max(a, b); },
        [](auto a, auto b) { return a - b / 4; }) - h);
}
```

27w • 16/12/2018 • 09:30

Structured Exceptions (Win32) and C++

- Raymond Chen: How can I handle both structured exceptions and C++ exceptions potentially coming from the same source?
  - Reddit
- Raymond Chen: Can I throw a C++ exception from a structured exception?

- Article by Jonathan Boccara
- Reddit

STL writes:

> *`mem_fn` is less typing, but lambdas are higher performance (MSVC's optimizer can't see through `mem_fn`'s data member) and can handle overloaded/templated member functions much more easily.*

Parameter passing, by Raymond Chen, Microsoft

- If you plan on keeping the parameter anyway, then there's no need to have separate `T const&` and `T&&` overloads
- If you're not keeping the parameter, then you still want to have separate `T const&` and `T&&` overloads
- Reddit

- Reddit
- Paper PDF
- Paper GitHub
- Reference implementation

Book: The C++ Annotations, V11.4.0

- C++ Annotations Version 11.4.0, by Frank B. Brokken
- Reddit 1
- Reddit 2

High performance SQLite, PostgreSQL, MySQL sync & async drivers

- Lithium
- Reddit

Format specifiers for C++ in the Visual Studio debugger

- Microsoft

Tim Myers @denvercoder

tuple = (1, 2)
threeple = (1,2,3)
fourple = (1,2,3,4)

let's make this happen people.

22h • 27/01/2020 • 17:39

Tim Myers @denvercoder

sheeple = (🐑, 🐑, 🐑, 🐑, 🐑)

thread

22h • 27/01/2020 • 17:44