C++ Club UK Meeting 116

Gleb Dolgich

2020-11-05

mailing2020-10

- mailing2020-10
  - Reddit

Select papers

- P1206R1 `ranges::to`: A function to convert any range to a container
- P2214R0 A Plan for C++23 Ranges
- P2226R0 A proposal for an idiom to move from an object and reset it to its default constructed state
- P2237R0 Metaprogramming

Named Parameters in C++20

Peter Dimov

- Reddit

C++ in Visual Studio Code reaches version 1.0

Julia Reid

- Reddit

C++ Talk Index

Website

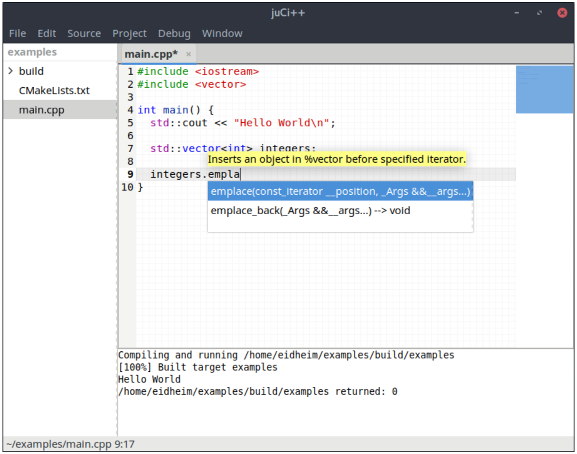
- Reddit

The terrible **size_t**

Reddit

juCi++: a lightweight, cross-platform IDE

- GitLab
- Installation guide


juCi++
File Edit Source Project Debug Window
examples
build
CMakeLists.txt
main.cpp
main.cpp*
emplace(const_iterator __position, _Args &&__args...)
emplace_back(_Args &&__args...) --> void
~/examples/main.cpp 9:17

```cpp
#include <iostream>
#include <vector>

int main() {
  std::cout << "Hello World\n";

  std::vector<int> integers;

  integers.empla
}
```

Inserts an object in %vector before specified iterator.

```
Compiling and running /home/eidheim/examples/build/examples
[100%] Built target examples
Hello World
/home/eidheim/examples/build/examples returned: 0
```

Figure 1: Screenshot

Library: cpp-lazy

GitHub

Cpp-lazy is a fast and easy lazy evaluation library for C++14/17/20.

Lazy evaluation is an evaluation strategy which holds the evaluation of an expression until its value is needed. In this library, all the iterators are lazy evaluated.

This library is not a replacement for `ranges::v3` but rather a (smaller) alternative.

Library: Crypto3

- Home page
- Boost mailing list announcement
- GitHub
- Reddit

Library: AAA - Auxiliary Arithmetic Algorithms

- GitHub (MIT)
- Docs

C#-like events in C++

Reddit

(Unrelated: Variable name prefixes)

- Code on Pastebin
- Signal-Slot library benchmarks
- Boost.Signals2

Uses of immediately invoked function expressions (IIFE) in C++

- Erik Rigtorp
- Jonathan Müller

Overloading by Return Type in C++

Philip Trettner

- Reddit

```cpp
struct to_string_t {
  std::string_view s;

  // int  from_string(std::string_view s);
  operator int() const;
  // bool from_string(std::string_view s);
  operator bool() const;
};

int i = to_string_t{"7"};
bool b = to_string_t{"true"};
```

The Defold game engine code style

Article

Code style

- C-like C++
- No classes (*huh? – GD*)
- No exceptions
- No STL
    - Custom containers
- Data ownership tracking
- C++98

(*It’s the end of 2020, by the way. – GD*)

A Buffers Library for C++20

Colby Pike

vcpkg: Accelerate your team development environment with binary caching and manifests

Microsoft

- Reddit

Related

Why is it such an abysmal pain to use libraries in C++ compared to pretty much anything else?

Raymond Chen on structured bindings

- Structured binding may be the new hotness, but we’ll always have `std::tie`
  - Reddit
- How to add C++ structured binding support to your own types
  - Reddit

dont_deduce

- artificial::mind
    - Reddit

C++11

```cpp
template <class T> struct foo_t { using type = T; };
template <class T> using foo = typename foo_t<T>::type;
```

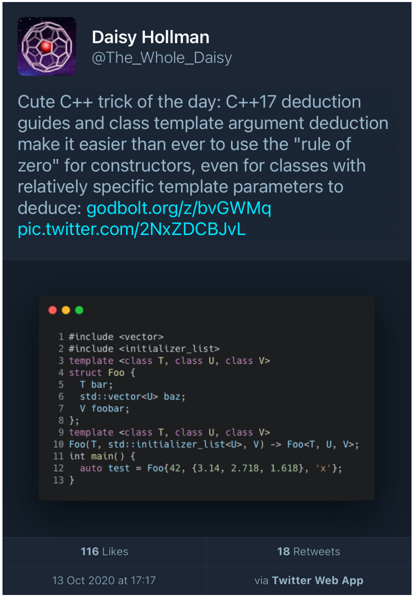
C++20

```cpp
template <typename T>
auto operator+(
    vec3<T> const& a,
    std::convertible_to<T> auto const& b
) -> vec3<T>;
```

Calendar and Time-Zones in C++20: Time of Day

Rainer Grimm

Daisy Hollman’s deduction trick



Daisy Hollman
@The_Whole_Daisy

Cute C++ trick of the day: C++17 deduction guides and class template argument deduction make it easier than ever to use the "rule of zero" for constructors, even for classes with relatively specific template parameters to deduce: godbolt.org/z/bvGWMq pic.twitter.com/2NxZDCBJvL

```cpp
#include <vector>
#include <initializer_list>
template <class T, class U, class V>
struct Foo {
  T bar;
  std::vector<U> baz;
  V foobar;
};
template <class T, class U, class V>
Foo(T, std::initializer_list<U>, V) -> Foo<T, U, V>;
int main() {
  auto test = Foo{42, {3.14, 2.718, 1.618}, 'x'};
}
```

116 Likes 18 Retweets
13 Oct 2020 at 17:17 via Twitter Web App

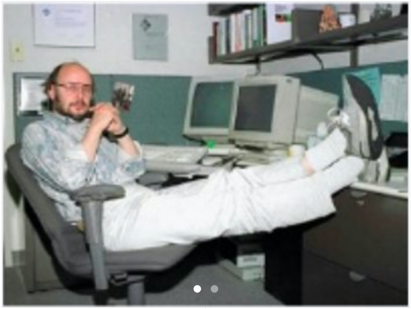
Figure 2: Deduction guides

How it started/How it’s going

S is for Shafik who stared into the void too long
@shafikyaghmour

How it started: How it's going:

#cplusplus pic.twitter.com/cINZFGUJLN



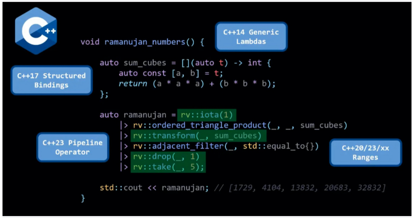
92 Likes **9** Retweets

12 Oct 2020 at 05:10 via **TweetDeck**

S is for Shafik who stared into the void too long
@shafikyaghmour

How it started: How it's going:

#cplusplus pic.twitter.com/cINZFGUJLN

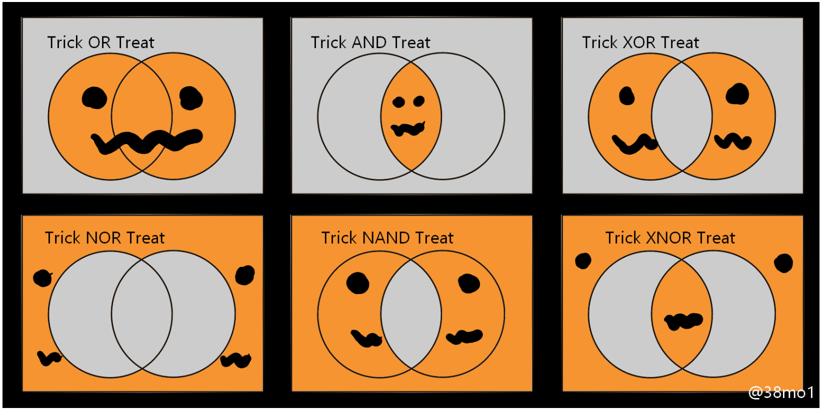


```
void ramanujan_numbers() {

    auto sum_cubes = [](auto t) -> int {
        auto const [a, b] = t;
        return (a * a * a) + (b * b * b);
    };

    auto ramanujan = rv::iota(1)
        |> rv::ordered_triangle_product(_, _, sum_cubes)
        |> rv::transform(_, sum_cubes)
        |> rv::adjacent_filter(_, std::equal_to{})
        |> rv::drop(_, 1)
        |> rv::take(_, 5);

    std::cout << ramanujan; // [1729, 4104, 13832, 20683, 32832]
}
```

92 Likes **9** Retweets

12 Oct 2020 at 05:10 via **TweetDeck**

Halloween logic


Trick OR Treat
Trick AND Treat
Trick XOR Treat
Trick NOR Treat
Trick NAND Treat
Trick XNOR Treat
@38mo1