

C++ Club UK Meeting 116

Gleb Dolgich

2020-10-29

Named Parameters in C++20

Peter Dimov

- [Reddit](#)

C++ in Visual Studio Code reaches version 1.0

Julia Reid

- [Reddit](#)

C++ Talk Index

Website

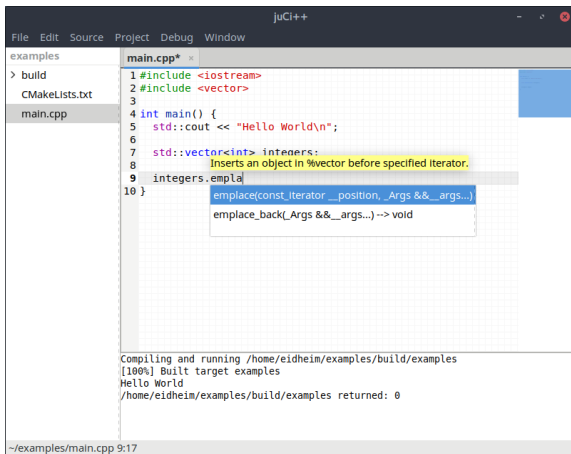
- [Reddit](#)

The terrible **size_t**

Reddit

juCi++: a lightweight, cross-platform IDE

- GitLab
- Installation guide



The screenshot displays the juCi++ IDE interface. The title bar reads "juCi++". The menu bar includes "File", "Edit", "Source", "Project", "Debug", and "Window". On the left, a project explorer shows a tree structure with "examples" as the root, containing "build", "CMakeLists.txt", and "main.cpp". The "main.cpp" file is open in the editor, showing the following code:

```
1 #include <iostream>
2 #include <vector>
3
4 int main() {
5     std::cout << "Hello World\n";
6
7     std::vector<int> integers;
8     // Inserts an object in %vector before specified iterator.
9     integers.emplace
10 }
```

A tooltip is visible over the `emplace` call, showing the signature: `emplace(const_iterator __position, _Args &&__args...)` and `emplace_back(_Args &&__args...) -> void`.

At the bottom of the IDE, a console window shows the output of the program:

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

The status bar at the bottom indicates the current file and line: `~/examples/main.cpp 9:17`.

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

```
1 struct to_string_t {
2     std::string_view s;
3
4     // int from_string(std::string_view s);
5     operator int() const;
6     // bool from_string(std::string_view s);
7     operator bool() const;
8 };
9
10 int i = to_string_t{"7"};
11 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

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

C++20

```
1 template <typename T>  
2 auto operator+(  
3     vec3<T> const& a,  
4     std::convertible_to<T> auto const& b  
5 ) -> vec3<T>;
```

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

```
1 #include <vector>
2 #include <initializer_list>
3 template <class T, class U, class V>
4 struct Foo {
5     T bar;
6     std::vector<U> baz;
7     V foobar;
8 };
9 template <class T, class U, class V>
10 Foo(T, std::initializer_list<U>, V) -> Foo<T, U, V>;
11 int main() {
12     auto test = Foo{42, {3.14, 2.718, 1.618}, 'x'};
13 }
```

116 Likes

18 Retweets

13 Oct 2020 at 17:17

via **Twitter Web App**

How it started/How it's going



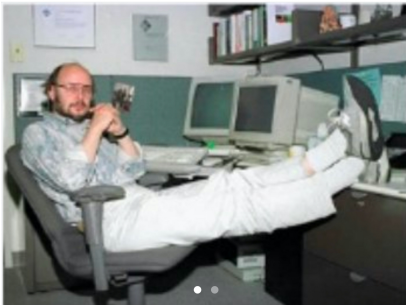
S is for Shafik who stared into the void too long

@shafikyaghamour

How it started:

How it's
going:

#cplusplus pic.twitter.com/cINZFGUJLN



92 Likes

9 Retweets

12 Oct 2020 at 05:10

via **TweetDeck**



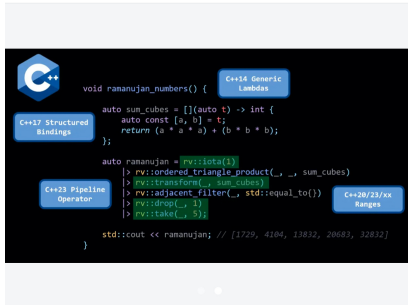
S is for Shafik who stared into the void too long

@shafikyaghamour

How it started:

How it's
going:

#cplusplus pic.twitter.com/cINZFGUJLN



92 Likes

9 Retweets

12 Oct 2020 at 05:10

via **TweetDeck**

Halloween logic

