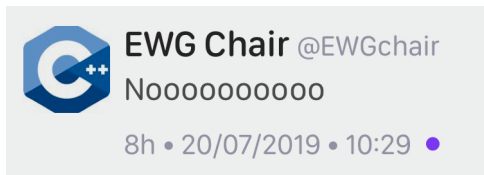


# C++ Club

Gleb Dolgich

2019-07-25

# C++20 is feature-complete



I wonder what was this tweet about.

C++20 -= Contracts perhaps?

- ▶ Herb Sutter
  - ▶ Reddit
- ▶ Bryce Lebach
  - ▶ On the proposed coroutine keywords
  - ▶ On the madness that is `std::web_view`
  - ▶ On the `constinit` keyword
  - ▶ On the spaceship operator

## std::format in C++20

<http://www.zverovich.net/2019/07/23/std-format-cpp20.html>

[https://www.reddit.com/r/cpp/comments/cgqo14/stdformat\\_in\\_c20/](https://www.reddit.com/r/cpp/comments/cgqo14/stdformat_in_c20/)

# Why `std::expected` is not in the standard yet?

[https://www.reddit.com/r/cpp/comments/c75ipk/why\\_stdexpected\\_is\\_not\\_in\\_the\\_standard\\_yet\\_is\\_it/](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](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>

# 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/](https://www.reddit.com/r/cpp/comments/c6gtd4/how_do_you_get_the_benefits_of_rust_in_c/)

- ▶ SaferCPlusPlus <https://github.com/duneroadrunner/SaferCPlusPlus/blob/master/README.md>
- ▶ Clang 10 thread safety analysis  
<https://clang.llvm.org/docs/ThreadSafetyAnalysis.html>
- ▶ Sanitizers <https://github.com/google/sanitizers>
- ▶ Escher C++ Verifier <http://www.eschertech.com/papers/ecvpp2016.pdf>
- ▶ Lifetime profile
  - ▶ Paper <https://github.com/isocpp/CppCoreGuidelines/blob/master/docs/Lifetime.pdf>
  - ▶ Clang implementation  
<https://eurollvm2019.sched.com/event/MGHd/implementing-the-c-core-guidelines-lifetime-safety-profile-in-clang>
  - ▶ Lifetime Profile Update in Visual Studio 2019 Preview 2  
<https://devblogs.microsoft.com/cppblog/lifetime-profile-update-in-visual-studio-2019-preview-2/>

# Microsoft to explore using Rust

<https://www.zdnet.com/article/microsoft-to-explore-using-rust/>

[https://www.reddit.com/r/cpp/comments/cegbhj/microsoft\\_looking\\_into\\_rust\\_as\\_an\\_alternative\\_to\\_c/](https://www.reddit.com/r/cpp/comments/cegbhj/microsoft_looking_into_rust_as_an_alternative_to_c/)



# How do C++ developers manage dependencies?

[https://www.reddit.com/r/cpp/comments/c6l3eg/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.*



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

```
1 | 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).

# Just started learning C++ coming from Python

[https://www.reddit.com/r/cpp/comments/c6vnb3/just\\_started\\_learning\\_c\\_coming\\_from\\_python\\_and/](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.*

# 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://www.reddit.com/r/cpp/comments/c646ng/use_constexpr_for_faster_smaller_and_safer_code/)

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

# 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/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://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.

# Introducing the Rule of DesDeMovA (1/4)

Blog post by Peter Sommerlad

<https://blog.safecpp.com/2019/07/01/initial.html>

[https://accu.org/content/conf2014/Howard\\_Hinnant\\_Accu\\_2014.pdf](https://accu.org/content/conf2014/Howard_Hinnant_Accu_2014.pdf)

Rule of Zero:

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

# Introducing the Rule of DesDeMovA (2/4)

C++ now

2019  
MAY 6-10  
cppnow.org



Peter Sommerlad

Rule of DesDeMovA

Video Sponsorship  
Provided By:



03:54

Voting Closed

Do you Remember: What Special Member Functions Do You Get?

3

What you write	What you get					
	default constructor	destructor	copy constructor	copy assignment	move constructor	move assignment
nothing	defaulted	defaulted	defaulted	defaulted	defaulted	defaulted
any constructor	not declared	defaulted	defaulted	defaulted	defaulted	defaulted
default constructor	user declared	defaulted	defaulted	defaulted	defaulted	defaulted
destructor	defaulted	user declared	defaulted (!)	defaulted (!)	not declared	not declared
copy constructor	not declared	defaulted	user declared	defaulted (!)	not declared	not declared
copy assignment	defaulted	defaulted	defaulted (!)	user declared	not declared	not declared
move constructor	not declared	defaulted	deleted	deleted	user declared	not declared
move assignment	defaulted	defaulted	deleted	deleted	not declared	user declared

Howard Hinnant's Table: [https://ericniebler.com/2016/06/02/Howard\\_Hinnant\\_August\\_2014.pdf](https://ericniebler.com/2016/06/02/Howard_Hinnant_August_2014.pdf)

Note: Getting the defaulted special members denoted with a (!) is a bug in the standard.

© Peter Sommerlad

# Introducing the Rule of DesDeMovA (3/4)

C++ now

2019  
MAY 6-10  
cppnow.org



Peter Sommerlad

Rule of DesDeMovA

Video Sponsorship  
Provided By:



02:38

Voting Closed

Rule of DesDeMovA: T&& operator=(T&&) noexcept=delete;

6

What you write	Rule of 5		Rule of 4		Rule of 3		Rule of 2	
	default constructor	copy constructor	copy assignment	move constructor	move assignment	operator=	operator+=	operator-=
nothing	defaulted	defaulted	defaulted	defaulted	defaulted	defaulted	defaulted	defaulted
any constructor	not declared	not declared	not declared	not declared	not declared	not declared	not declared	not declared
default constructor	user declared	defaulted	defaulted	defaulted	defaulted	defaulted	defaulted	defaulted
destructor	defaulted	user declared	defaulted (!)	defaulted (!)	not declared	not declared	not declared	not declared
copy constructor	not declared	defaulted	user declared	defaulted (!)	not declared	not declared	not declared	not declared
copy assignment	defaulted	defaulted	defaulted (!)	user declared	not declared	not declared	not declared	not declared
move constructor	not declared	defaulted	deleted	deleted	user declared	not declared	not declared	not declared
move assignment	defaulted	user declared	deleted	deleted	not declared	deleted	deleted	deleted


**DesDeMovA**  
Rule of if  
Destructor defined  
Deleted  
Move Assignment

Howard Hinnant's Table: <https://ericniebler.com/2015/04/24/rule-of-five/>  
Note: Getting the defaulted special members denoted with a (!) is a bug in the standard.

# Introducing the Rule of DesDeMovA (3/4)

C++ now


**2019**  
MAY 6-10  
cppnow.org



**Peter Sommerlad**

Rule of DesDeMovA

Video Sponsorship  
Provided By:





02:19


Voting Closed

Summary 🍌 7

1. Rule of Zero
2. Rule of DesDeMovA (no copy, no move for SBRM/RAII and OO-Base classes)
3. Rule of Unique Resource Managers (move-only, no copy)
4. Rule of Five for Resource Managers with Value Semantics, or other really special cases



Download IDE at:  
[www.cevloop.com](http://www.cevloop.com)



Sponsors welcome!

Commercial licensing possible!



# strong\_typedef - Create distinct types for distinct purposes

Article by Anthony Williams

[https://www.justsoftwaresolutions.co.uk/cplusplus/strong\\_typedef.html](https://www.justsoftwaresolutions.co.uk/cplusplus/strong_typedef.html)

[https://github.com/anthonywilliams/strong\\_typedef](https://github.com/anthonywilliams/strong_typedef)

```
1 using transaction_id =  
2     jss::strong_typedef<struct transaction_tag, std::string>;  
3  
4 bool is_a_foo(transaction_id id)  
5 {  
6     auto &s = id.underlying_value();  
7     return s.find("foo") != s.end();  
8 }
```

<https://www.cycfi.com/2019/07/photon-micro-gui/>

[https:](https://www.reddit.com/r/cpp/comments/ccq9pn/elemental_c_gui_library/)

[//www.reddit.com/r/cpp/comments/ccq9pn/elemental\\_c\\_gui\\_library/](https://www.reddit.com/r/cpp/comments/ccq9pn/elemental_c_gui_library/)

## Are there any good C++ libraries for data visualization?

- ▶ VTK <https://vtk.org/>
- ▶ ROOT <https://root.cern.ch/>
- ▶ matplotlib-cpp <https://github.com/lava/matplotlib-cpp>
  - ▶ matplotlib (Python) <https://matplotlib.org/>
- ▶ QCustomPlot (QT, GPL/commercial) <https://www.qcustomplot.com/>

<http://cppcast.com/2019/07/robert-maynard/>

[https://www.reddit.com/r/cpp/comments/c9bpxb/cppcast\\_cmake\\_and\\_vtk\\_with\\_robert\\_maynard/](https://www.reddit.com/r/cpp/comments/c9bpxb/cppcast_cmake_and_vtk_with_robert_maynard/)

## CMake line by line - creating a header-only library

<http://dominikberner.ch/cmake-interface-lib/>

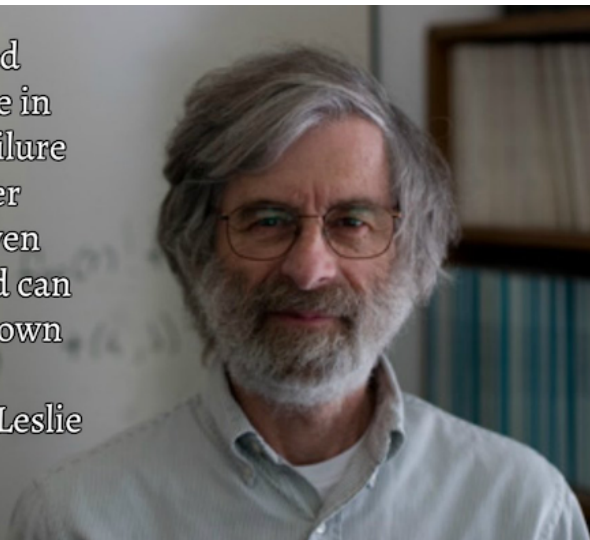
[https://www.reddit.com/r/cpp/comments/c8ty2h/a\\_line\\_by\\_line\\_explanation\\_how\\_to\\_create\\_a/](https://www.reddit.com/r/cpp/comments/c8ty2h/a_line_by_line_explanation_how_to_create_a/)

<https://github.com/bernedom/Sl>


Professional CMake: A Practical Guide, 4th ed., CMake 3.15


<https://crascit.com/professional-cmake/> \$30


"A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusable." - Leslie Lamport






Vittorio Romeo:


 [abbycin](#) -13 points · 1 day ago

 meta programming is destroying c++

 Reply   Give Award   Share   Report   Save

 [SuperV1234](#) 23 points · 1 day ago 

 *laughs in constexpr*

 Reply   Give Award   Share   Report   Save