

C++ Club UK

Gleb Dolgich

2019-09-12

C++20 Concepts Are Here in Visual Studio 2019 version 16.3

<https://devblogs.microsoft.com/cppblog/c20-concepts-are-here-in-visual-studio-2019-version-16-3/>

https://www.reddit.com/r/cpp/comments/d2alin/c20_concepts_in_visual_studio_2019_version_163/

Dropbox abandons C++, uses Swift, Kotlin, JavaScript and Electron instead

- Eyal Guthmann (Dropbox): The (not so) hidden cost of sharing code between iOS and Android
 - [Reddit](#)
 - [HackerNews](#)

It's possible we could have done a better job at leveraging open source C++ libraries, but the open source culture in the C++ development community was (is still?) not as strong as it is in the mobile development community <...>

It seems like the real issue was that Dropbox lost all of their senior C++ engineers.

- The Register: Dropbox would rather write code twice than try to make C++ work on both iOS and Android
 - [Reddit](#)

Dropbox abandons C++, uses Swift, Kotlin, JavaScript and Electron instead

Previously

- Djinni
- CppCon 2014: Alex Allain & Andrew Twyman "Practical Cross-Platform Mobile C++ Development"
- CppCon 2017: Stephen Spann "Cross-Platform Apps with Dropbox's Djinni..."

Barry Revzin

- What is unified function call syntax anyway?
 - [Reddit](#)
- UFCS Customization and Extension
 - [Reddit](#)
 - [P1282R0 Ceci N'est Pas Une Pipe: Adding a workflow operator to C++](#)

- * Bjarne Stroustrup. [A bit of background for the unified call proposal](#)

Based on real input from code and users, I reluctantly agreed that for compatibility reasons, $x.f(y)$ and $f(x,y)$ could not mean exactly the same. The only feasible way forward was to do a traditional lookup based on the syntax used, and then try the other syntax if the first one failed. Stability – backwards compatibility – is an important feature, overruling my desire for perfection.

- * [P0131 Unified call syntax concerns](#)

Approval tests (1/2)

Also known as **Golden Master Tests** or **Snapshot Testing** (locking down current behaviour)

- CppOnSea 2019 - Clare Macrae - Quickly testing legacy code
<https://youtu.be/dtm8V3TIB6k>
 - Slides <https://slideshare.net/ClareMacrae>
 - CppCast with Clare Macrae <https://cppcast.com/clare-macrae/>
 - r/cpp https://www.reddit.com/r/cpp/comments/ckzc11/cppcast_approval_tests_with_clare_macrae/
- Code <https://github.com/approvals/ApprovalTests.cpp> (Apache 2.0)
- Approval Tests Library - Capturing Human Intelligence [available for Java, C#, VB.Net, PHP, Ruby, Node.JS and Python] <https://approvaltests.com/> by Llevelyn Falco
 - Supports Catch, Catch 2, Google Test, [Okra](#)
- Mutation tests: sabotage the code
 - Mutate++ https://github.com/nlohmann/mutate_cpp

Books

- Modern C++ Programming with Test-Driven Development, by Jeff Langr [[Safari Books Online](#)]
- Your Code as a Crime Scene, by Adam Tornhill [[Safari Books Online](#)]
- Software Design X-Rays, by Adam Tornhill [[Safari Books Online](#)]

Tools

- OpenCoverage <https://github.com/OpenCppCoverage>
- BullseyeCoverage <https://www.bullseye.com>

<https://www.cleanqt.io/blog/crash-course-in-qt-for-c%2B%2B-developers,-part-1>

Modern Qt Development: The Top 10 Tools You Should Be Using

<https://blog.qt.io/blog/2018/10/12/modern-qt-development-top-10-tools-using/>

[https:](https://www.reddit.com/r/cpp/comments/9njw5n/is_there_an_easytouse_gui_library/)

[//www.reddit.com/r/cpp/comments/9njw5n/is_there_an_easytouse_gui_library/](https://www.reddit.com/r/cpp/comments/9njw5n/is_there_an_easytouse_gui_library/)

<https://www.cleanqt.io/blog/cmake-it-modern-using-c%2B%2B-and-qt,-part-1>

A new SQLite C++ wrapper

<https://blog.trailofbits.com/2019/08/26/wrappers-delight/>

https://www.reddit.com/r/cpp/comments/cxxk4b/a_new_c_sqlite_wrapper/

The Reddit thread also includes a heated discussion on how to handle errors and if exceptions are a good thing (eyeroll).

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

```
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 }
```

cppcon bingo

Herb Sutter playing piano	Gripes about exceptions	Allocators	Monday WiFi issues	Unicode printing errors on badges
Memes on slides	Another hipster presentation uses reveal.js	Strategies for talking to C programmers	Attendees try to file feature requests in person	Template meta programming
Visual Studio demos	Zero cost abstractions	Boost	Concepts	Live coding demo crashes or doesn't compile
Bryce with a flock of volunteers following him	Assurances that X will be in the next standard	A lunch group grows way too big for any one restaurant	Java hate	Subtle bugs on concurrency slides
Last minute slide making	Cherry Coke	Monads	"JS/Swift/Rust has X, why doesn't C++?"	(Re)definition of modern C++