C++ Club UK

Gleb Dolgich

2019-01-24

Deducing your intentions

Deducing your intentions by Andrzej Krzemieński

> *In this post we will briefly describe what class template argument deduction is, and why it works differently than what people often expect.*

- Class template argument deduction (CTAD) in C++17: P0091R2

```
1 auto q = std::make_optional(std::optional<int>{});
2 // q is optional<optional<int>>
3 std::optional q (std::optional<int>{});
4 // q is optional<int> !
```

Contra CTAD

Contra CTAD by Arthur O'Dwyer

> *"I like features that work 100% of the time. I hate features that work 99% of the time. Working 99% of the time is much worse than working 50% of the time." And CTAD is the poster child for a feature that works 99% of the time. That is, it works 100% of the time... until it doesn't.*

Stop with the CTAD FUD!

```
1 std::optional maybe_string{"Hello!"s}; // optional<string>
2 std::optional other_thing{maybe_string}; // optional<string>
3
4 // Deduction guides:
5 // [1] Explicit
6 template <typename T> auto __deduce(T) -> optional<T>;
7 // [2] Implicit from std::optional copy ctor
8 template <typename T> auto __deduce(const optional<T>&) -> optional<T>;
```

”Modern” C++ Lamentations

”Modern” C++ Lamentations by Aras Pranckevičius (Unity)

> *C++ compilation times have been a source of pain in every non-trivial-size codebase I've worked on. <...> Yet it feels like the C++ community at large pretends that is not an issue, with each revision of the language putting even more stuff into header files, and even more stuff into templated code that has to live in header files.*
>
> *Some of the "all feedback will be ignored unless it comes in form of a paper presented at a C++ committee meeting" replies that I've seen in the past few days sound to me like a not very productive approach.*

- Reddit: https://www.reddit.com/r/programming/comments/aac4hg/modern_c_lamentations/
  - https://www.reddit.com/r/programming/comments/aac4hg/modern_c_lamentations/ecrwgep/

Thoughts on Modern C++ and Game Dev by Ben Deane

> *TL;DR: The C++ committee isn't following some sort of agenda to ignore the needs of game programmers, and "modern" C++ isn't going to become undebuggable.*
>
> *I think it's a shame that the STL is so strongly identified with its containers. They tend to be what gets taught first, so when we say "the STL" we often first think of `std::vector` when we should really be thinking of `std::find_if`.*

- Reddit: https://www.reddit.com/r/cpp/comments/abnoke/thoughts_on_modern_c_and_game_dev/

Gamedev introduction to 'Modern' C++ by Mathieu Ropert

- Reddit: https://www.reddit.com/r/cpp/comments/abvw6c/gamedev_introduction_to_modern_c/

”Modern” C++ Ruminations

”Modern” C++ Ruminations by Sean Parent

> *Programming is a profession. It is an ethical obligation to work to improve our profession. The more senior and talented you are, the more you owe to the community. Giving back can take many forms; mentoring, lecturing, publishing, serving on committees and furthering open source projects. Part of that obligation is to continue to study, to read papers and work through books.*

- Reddit: https://www.reddit.com/r/cpp/comments/ac3ntu/modern_c_ruminations/

Another cool MSVC flag: /d1reportTime

- Another cool MSVC flag: /d1reportTime by Aras Pranckevičius
  - Reddit: https://www.reddit.com/r/cpp/comments/aij9h4/another_cool_msvc_flag_d1reporttime/
- Complete list of MSVC flags

Compile Time, Binary Size Reductions and C++'s future for sol3 by ThePhD

Cpp-Taskflow

- Code: https://github.com/cpp-taskflow/cpp-taskflow
- Docs: https://cpp-taskflow.github.io/cpp-taskflow-documentation.github.io/
- Reddit: https://www.reddit.com/r/cpp/comments/9b01ek/cpptaskflow_v20_a_new_taskbased_parallel/

Quote

Eagleson's Law:

> *Any code of your own that you haven't looked at for six or more months might as well have been written by someone else.*