C++ Club Meeting Notes

Gleb Dolgich

2019-03-07

Kona Trip Reports

- CppCast: Kona Trip Report with Peter Bindels
- Timur Doumler
- Corentin Jabot (Reddit)
  - 177 papers, of which 144 never seen before
  - One contention point of the Coroutines TS is that it always heap-allocates the type-erased coroutine frame and relies on impressive optimization techniques to make these allocations disappear when the frame does not outlive the caller's context. The Core Coroutine proposal offered deterministic stack allocations of coroutines frames. The issue with having the coroutine frame allocated on the stack is that it can not be optimized away without black magic. That black magic now has a name: Deferred layout. (Deemed very complex to implement by compiler vendors.)
  - https://github.com/lewissbaker/cppcoro
  - https://github.com/luncliff/coroutine
  - There will be a paper in the post-mailling which offer a great way to use `await` and `yield` as keyword without breaking any existing code.
  - Voted against Modules.

Are C++ Modules Dead-on-Arrival?

Colby Pike:

- https://vector-of-bool.github.io/2019/03/04/modules-doa-2.html
- https://www.reddit.com/r/cpp/comments/ax81zs/are_modules_doa_a_followup/

This is a follow-up to C++ Modules might be Dead-on-Arrival.

> *We can deduce the answer from Betteridge's law of headlines: No.*
>
> *First and foremost, let me get this out of the way: I do not want C++ modules to fail. I do not know of anyone who wishes for them to crash and burn. I do know people with serious concerns about modules. The purpose of these posts is not to shoot down the modules work, but to make our skeptical voices heard.*

Gabriel Dos Reis:

> *A lot of harm has been done with incomplete understandings or misunderstandings presented as flaws in the modules design.*

CMake + GCC module proof-of-concept

https://www.reddit.com/r/cpp/comments/axnwiz/cmake_gcc_module_proofofconcept/

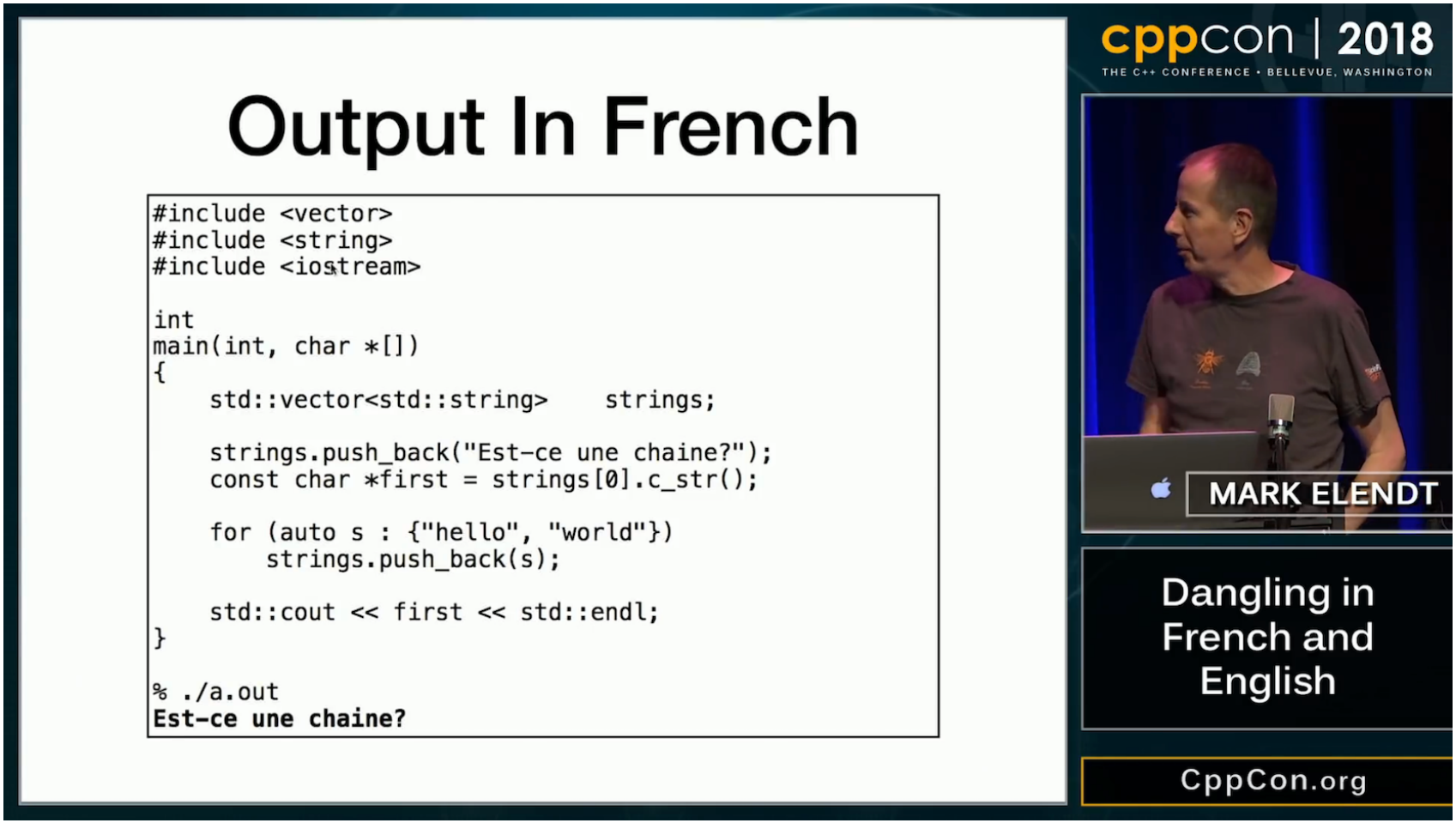
> *Hi all, CMake developer here. There's been a lot of noise and discussion of modules recently, particularly with respect to how build systems will deal with it. There has been* ***build2*** *for quite a while, but it was also designed with modules in mind.*
>
> *At Kona last week, I worked with Nathan Sidwell who is working on GCC's module support to get a minimally viable proof-of-concept for CMake building modules.*

Twitter bot that posts updates when compilers add support for new features

- https://twitter.com/CompilerCpp/status/1102565726970363904
- Programmed in Go
- https://github.com/therocode/CppCompilerCompliance

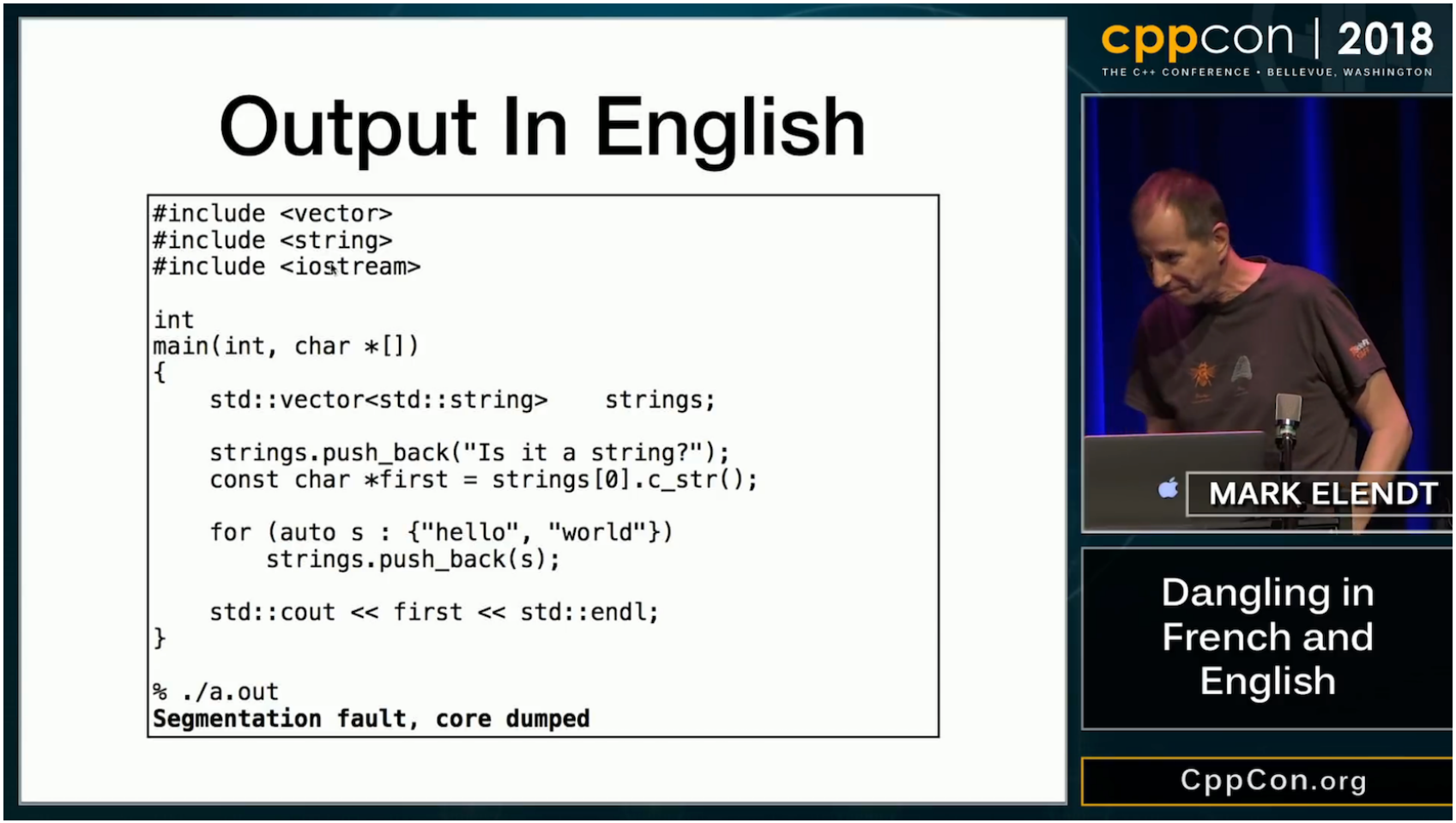
CppCon 2018 - Mark Elendt - Dangling in French and English

https://youtu.be/jieYLTcmTS0



CppCon 2018 - Mark Elendt - Dangling in French and English

https://youtu.be/jieYLTcmTS0



Moving iterators in C++

https://cukic.co/2019/02/09/moving-iterators-in-cxx/

> *`std::move_iterator` is an iterator adaptor which behaves exactly like the underlying iterator, except that dereferencing converts the value returned by the underlying iterator into an* rvalue. *(https://en.cppreference.com/w/cpp/iterator/move_iterator)*

```
1 std::vector<fs::directory_entry> results;
2 auto dir_items = files_in_dir(...);
3 results.insert(results.end(), dir_items.cbegin(), dir_items.cend());
```

Alternatively (see https://en.cppreference.com/w/cpp/algorithm/move):

```
1 std::move(dir_items.begin(), dir_items.end(), std::back_inserter(results));
```

Low-cost Deterministic C++ Exceptions for Embedded Systems

https://www.research.ed.ac.uk/portal/files/78829292/low_cost_deterministic_C_exceptions_for_embedded_systems.pdf

James Renwick, Tom Spink, Björn Franke (University of Edinburgh)

> *In our novel C++ exception implementation we make use of a stack-allocated object that records the necessary run-time information for throwing an exception, such as the type and size of the exception object. This state is allocated in a single place and is passed between functions via an implicit function parameter injected into functions which support exceptions. The state is initialised by* `throw` *expressions, and is re-used to enable re-throwing.* `catch` *statements use the state in order to determine whether they can handle the exception. After a call to a function which may throw exceptions, a run-time check is inserted to test whether the state containsan active exception.*

Fast_ber: ASN.1 BER serialization library

- https://github.com/Samuel-Tyler/fast_ber
- https://www.reddit.com/r/cpp/comments/anwlrs/fast_ber_asn1_ber_serialization_library_written/
- https://en.wikipedia.org/wiki/Abstract_Syntax_Notation_One

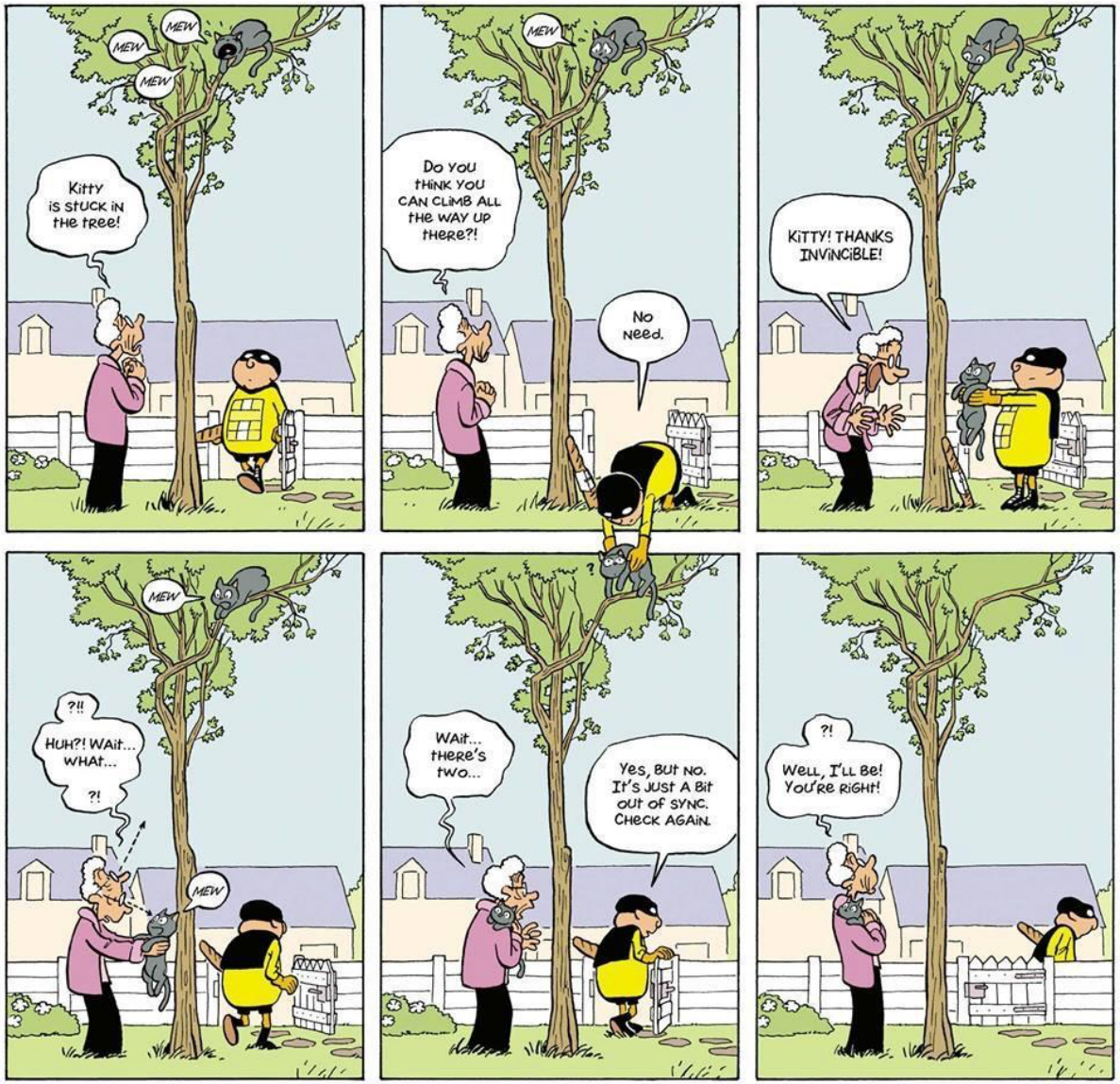
Formatting user-defined types with {fmt} library

https://wgml.pl/blog/formatting-user-defined-types-fmt.html

Don’t Use `std::endl`

https://accu.org/index.php/journals/2619

Eventual consistency



Twitter

Christopher Di Bella
@cjdb_ns

C++ gives you grey hairs, but UB can send you back in time to get your colour back.

22 Likes	**2** Retweets
27 Feb 2019 at 05:38	via **Twitter for Android**