

C++ Club Meeting Notes

Gleb Dolgich

2018-01-25

Announcement

- ▶ Windows System for Linux (WSL) support
- ▶ If and switch with initializers from C++17
- ▶ CMake changes: open single file/open folder
- ▶ Performance improvements: incremental highlighter
- ▶ MSVC supported automatically if installed

Best C++ build system?

Reddit

- ▶ CMake is suggested most often
- ▶ [build2](#), [QBS](#)
- ▶ [Meson](#)

YouTube

- ▶ [International Components for Unicode \(ICU\)](#)
- ▶ [Boost.Locale](#)

A CMS written in C++

Blog

See also: [Building a website with C++](#)

CppCon 2017: Vinnie Falco “Make Classes Great Again! (Using Concepts for Customization Points)”

- ▶ CppCon 2017: Vinnie Falco “Make Classes Great Again! (Using Concepts for Customization Points)”
 - ▶ Code
- ▶ CppCon 2016: Vinnie Falco “Introducing Beast...”
- ▶ See also: [Boost::Beast: HTTP and WebSocket built on Boost.Asio in C++11](#)

CppCon 2017: Vinnie Falco “Make Classes Great Again!...” (cont.)

Summary

```
/// Holds an HTTP message
template<bool isRequest, class Body, class Fields>
struct message;

/// Holds an HTTP request
template<class Body, class Fields>
struct message<true, Body, Fields>
    : Fields, private Body::value_type
{
    int version;
    string_view method() const
    { return this->get_method(); }
    void method(string_view s)
    { this->set_method(s); }
    string_view target() const
    { return this->get_target(); }
    void target(string_view s)
    { this->set_target(s); }
    typename Body::value_type& body()
    { return *this; }
    typename Body::value_type const& body() const
    { return *this; }
};
```

Body Requirements:

Expression	Type	Description
B::value_type		The type of container used to represent the body in a message.
B::read	void(iostream&, B::value_type&)	A function invoked by the implementation to parse the body from a std::istream.
B::write	void(ostream&, B::value_type const&)	A function invoked by the implementation to serialize the body to a std::ostream.

```
/// A Body that uses a string container
struct string_body
{
    using value_type = string;
    static void read(istream& is, string& body)
    { is >> body; }
    static void write(ostream& os, string const& body)
    { os << body; }
};

// Determine if B meets the requirements of Body
template<class B, class = void>
struct is_body : false_type {};

template<class B>
struct is_body<B, void_t<
    typename B::value_type,
    decltype(
        B::read(
            declval<istream&>(),
            declval<typename B::value_type&>()),
        B::write(
            declval<ostream&>(),
            declval<typename B::value_type const&>()),
            (void)0
        )>> : true_type {};
```

110

cppcon 2017
THE C++ CONFERENCE • BELLEVUE, WASHINGTON



VINNIE FALCO

Make Classes
Great Again!
(Using Concepts
for Customization Points)

CppCon.org

Your own type predicate in C++11

Post

A gentle intro to metaprogramming, metafunctions and type traits.

```
1 static_assert(is_acceptable<X>::value,  
2               "X does not have a desired interface");  
3  
4 template <typename... T>  
5 using void_t = void;
```


YouTube

```
1 #include <new>
2 struct X { const int n; double m; }
3 int main() {
4     X p{3, 8.8};
5     new (&p) X{5, 7.7};
6     int b = std::launder(&p)->n; // 5
7     double c = p.m; // UB!
8 }
```

CppCon 2017: Juan Arrieta “Traveling the Solar System with C++: Programming Rocket Science”


YouTube

Modeling Reality


- ▶ N-body perturbations
- ▶ Ring perturbations
- ▶ Gravitational potential
- ▶ Solar radiation pressure
- ▶ Relativistic corrections
- ▶ Atmospheric drag
- ▶ Spacecraft thrust
- ▶ ...

- ▶ Frames of reference
- ▶ Clocks
- ▶ Shape models
- ▶ Temperature models
- ▶ Radiation models
- ▶ Attitude dynamics
- ▶ Aerothermodynamics
- ▶ Instrument modeling
- ▶ Telecomm
- ▶ Ground stations
- ▶ Avionics
- ▶ ...

32 of 59

 2017

cppcon | 2017
THE C++ CONFERENCE • BELLEVUE, WASHINGTON

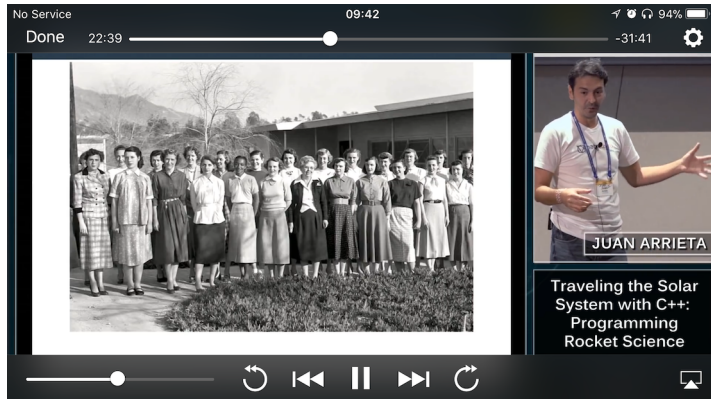


JUAN ARRIETA

**Traveling the Solar System with C++:
Programming Rocket Science**

CppCon.org

CppCon 2017: Juan Arrieta “Traveling the Solar System with C++: Programming Rocket Science” (cont.)



CppCon 2017: Juan Arrieta “Traveling the Solar System with C++: Programming Rocket Science” (cont.)

A Dream of Fields

```
1 namespace monte {  
2   class BOA; // JPL's Binary Object Archive  
3 }  
4 // read from file and return Handle<BOA>  
5 auto boa = BOA::FromFile("my.boa");  
6 // store a new design element in the BOA  
7 auto shape = BOASphere(boa, "Moon", 1737.0 * km);  
8 // find and process events  
9 auto timespan = TimeInterval("01-Jan-2017 ET",  
10                             "01-Jan-2018 ET");  
11 auto finder = OccultationFinder(boa, "Sun", "Moon",  
12                                "DSS-14");  
13 for (auto && event : finder.find(timespan, 60 * sec))  
14   process_event(event)
```

45 of 59



cppcon | 2017
THE C++ CONFERENCE • BELLEVUE, WASHINGTON



JUAN ARRIETA

Traveling the Solar
System with C++:
Programming
Rocket Science

CppCon.org

CppCon 2017: Lars Knoll “Qt as a C++ Framework: History, Present State and Future”

YouTube

Comments are divided:

- ▶ “A good presentation, lots of people use Qt”
- ▶ “This is more of an ad than a talk, why do we waste our time?”
- ▶ “I really hate how they try to hide that Qt is free for commercial development”
- ▶ “Qt is more and more a dead end in GUI development. It’s not done like this anymore. Use Qt for basic operations of your common code but move to the platform specific code for GUI. They look so ugly now and even GTK is better then Qt.”
- ▶ “new - this is why your library is so bad. Just stop.”

A C++ Hello World And the Cute Heartless Rainbow

- ▶ Part 1: A C++ Hello World And A Glass Of Wine, Oh My!
- ▶ Part 2

Article

- ▶ Function pointer
- ▶ Pointer to member function
- ▶ Functor
- ▶ Lambda

- ▶ **SObjectizer** – a framework for building solid multithreaded applications. It is based on async message exchange and uses a mixture high-level abstractions: Actor-Model, Publish-Subscribe and Communicating Sequential Processes.
- ▶ **REStinio** – header-only C++14 library that gives you an embedded HTTP server with nice express-like routing (although it is not mandatory to use router) and websockets on board.
- ▶ **timertt** – The timertt (Timer Thread Template) library was created as a lightweight alternative of ACE Framework's timers for SObjectizer.

Boost 1.66 released

[Release page](#)

[GitHub](#) – A future and stream library for modern C++ (MIT)

Cereal – a C++11 serialization library

Website

- ▶ Header-only
- ▶ Similar to Boost serialization, with easy transition
- ▶ Formats: binary, XML, JSON, custom
- ▶ No external dependencies
- ▶ Supports `std::shared_ptr` and `std::unique_ptr`
- ▶ BSD

A powerful library for writing beautiful command line interfaces in C++11. No dependencies, header-only, BSD 3-Clause.

- ▶ [Announcement](#)
- ▶ [GitHub](#)
- ▶ [Docs](#)

Light-weight, simple and fast XML parser for C++ with XPath support

- ▶ [Website](#)
- ▶ [GitHub](#) (MIT)

Unknown:

Java: write once, run away.

Unknown from Twitter:

Debuggers don't remove bugs. They only show them in slow motion.

Jon Purdy @whyevernotso:

People who deeply understand C++ are great to have on a team—not for their knowledge of C++, but for their ability to accept, cope with, and pragmatically manage things that other people would balk at and call insane, bug-prone, abject horrors.

Slava Pestov @slava_pestov, Apple Swift programmer:

This is very true, but there are like five people who deeply understand C++