C++ Club Meeting Notes

Gleb Dolgich

2017-11-02

Website

CppCon 2017: Matt Godbolt "What Has My Compiler Done for Me Lately? Unbolting the Compiler's Lid"

YouTube

- ▶ x86 Assembly (Wikibooks)
- ▶ Reading assembly alone can be misleading, need to measure too
    - ▶ Google Benchmark
    - ▶ quick-bench.com
- ▶ Target specific CPU architecture (`-march=i486` vs. `-march=haswell`)
- ▶ Coming soon: code execution
- ▶ Written in Node.js
- ▶ Costs: $200/month

YouTube

- ▶ Versions in library paths: not ideal
- ▶ Masterconfig: Premake solves this much more elegantly, IMHO
- ▶ EA package server: directory of packages
- ▶ Disting is tricky (manual versioning)
- ▶ Module example (MSVC-style); discusses how to find modules during build
- ▶ EA packages can have circular link dependencies
- ▶ Modules should have package as part of the name to avoid conflicts (naming convention). Better yet, there should be a package manager.
- ▶ Q: "I'm coming from Rust and I don't get the point of these modules"

YouTube

- ▶ Updating and fixing bugs is a problem, especially when the device is not supported anymore or the manufacturer goes out of business
- ▶ Why would anyone want their fridge to connect to the Internet?
- ▶ Early IoT adopters have quite a few paperweights
- ▶ IoT security is a big issue and it is hard
- ▶ Using exceptions may not be possible
- ▶ IoT projects are excellent for getting children interested in programming and technology

Intel® System Studio 2018 Beta User Guide for IoT C/C++ Development

▶ CodeProject article
▶ Download Intel System Studio IoT Edition

Supported boards:

▶ Intel® IoT Gateway
▶ MinnowBoard MAX

CppCon 2017: John D. Woolverton "C Pointers"

YouTube

A lightning talk about learning from mistakes.

```
1  /* add polygon to current grid position and advance */
2  *((*(gridfill++))++) = poly;
```

DRES: Destruct Resources on Exit Scope (aka *RAII*)

Clara: A simple-to-use composable command line parser by Phil Nash

- ▶ GitHub
- ▶ YouTube
- ▶ C++11
- ▶ Monadic binding for composability, no exceptions
- ▶ Used by Catch, combines with user's parsers

```cpp
int width = 0;
using namespace clara;
auto cli = Opt(width, "width")["-w"]["--width"]("How wide?");
auto result = cli.parse(Args(argc, argv));
if (!result) {
    std::cerr << "Error: " << result.errorMessage() << std::endl;
    exit(1);
}
```

DLL: Deep Learning Library (!)

GitHub (MIT)

DLL is a library that aims to provide a C++ implementation of Restricted Boltzmann Machine (RBM) and Deep Belief Network (DBN) and their convolution versions as well. It also has support for some more standard neural networks.

- ▶ Header-only
- ▶ C++14
- ▶ CUDA
- ▶ Windows not supported
- ▶ Dependencies:
    - ▶ Catch
    - ▶ cifar – Simple C++ reader for CIFAR-10 dataset
    - ▶ etl – Expression Templates Library (ETL) with GPU support
    - ▶ mnist – Simple C++ reader for MNIST dataset
    - ▶ libsvm – A Library for Support Vector Machines

Post

Goal: to retire C++/CX

Download Windows Insider Preview SDK build 17025

Post

Case study

▶ Anomalistic compile times
▶ Caching stats
▶ Code generation summary
▶ Name demangling: undname.exe or online demangler

ARM GCC Cross Compilation in Visual Studio

Post

Download the Visual Studio 2017 Preview, install the **Linux C++ Workload**, select the option for **Embedded and IoT Development** and give it a try with your projects.

CLion 2017.3 EAP

Post

- ▶ Valgrind memcheck integration (except on Windows)
- ▶ YouTrack ticket: Sanitizer support
- ▶ Improved support for multiple toolchains

Guide into OpenMP: Easy multithreading programming for C++

Link

```cpp
1  #include <cmath>
2  int main()
3  {
4      const int size = 256;
5      double sinTable[size];
6
7      #pragma omp parallel for
8      for(int n=0; n<size; ++n)
9          sinTable[n] = std::sin(2 * M_PI * n / size);
10
11     // the table is now initialized
12 }
```

A polymorphic value-type for C++

P0201R2 by Jonathan Coe and Sean Parent

```cpp
// Copyable composite with mutable polymorphic components
class CompositeObject {
  std::polymorphic_value<IComponent1> c1_;
  std::polymorphic_value<IComponent2> c2_;
public:
  CompositeObject(std::polymorphic_value<IComponent1> c1,
                  std::polymorphic_value<IComponent2> c2) :
                  c1_(std::move(c1)), c2_(std::move(c2)) {}
  void foo() { c1_->foo(); }
  void bar() { c2_->bar(); }
};
```

Printing boolean values

Pierre Habouzit on Twitter:

```
1  printf("%c", boolean_expr["NY"])
```