Denver Tech Center
C++ Developers

Scott Fredericksen

18 Nov 2013

Congratulate yourselves
for being in the top 1%

Introductions

Things you may want to tell us:

- Why are you here?
- What do you want to get out of the group?
- What operating systems do you use with C++?
- How are you able to contribute to the group?
- What software conferences have you attended recently?
- What other Meetups do you attend? Recommend?
- What are you reading?
- What podcasts do you listen to?

C++... in 2013?

Really?

- **Denver Tech Center C++**
  - 20 members (founded 24 Oct 2013)
  - First Meetup – Today (11 attendees)
- Denver C/C++ Developers Group
  - 67 members (founded 24 Oct 2013)
  - 2 meetups (19, 23 attendees)
- San Francisco
  - 335 members (founded 18 Jan 2012)
  - 18 meetups (8, 8, 38, 6, attendees)
- SFBay
  - 338 members (founded 27 Dec 2012)
  - 25 meetups (36, 10, 33, … attendees)
- New York
  - 1,065 members (founded 26 May 2009)
  - 23 meetups (107, 109, 80, attendees)
- Chicago
  - 711 members (founded 13 May 2009)
  - 24 meetups (181, 193, 51 attendees)
- Austin
  - 416 members (12 Sep 2006)
  - 101 meetups (28, 19, 27 attendees)
- San Antonio
  - 36 members (6 Jul 2013)
  - 3 meetups (11, 9, 11 attendees)
- Houston
  - 52 members (founded 6 Jul 2013)
  - 2 meetups (17, 12 attendees)
- DFW
  - 61 members (founded 12 Aug 2012)
  - 6 meetups (7, 8, 7, 2, 6 attendees)
- Charleston
  - 8 members (founded 19 Feb 2013)
  - No meeups scheduled)
- Philly
  - 126 members (founded 5 May 2012)
  - 11 meetups (20, 22, 11 attendees)

- Dutch
  - 34 members (founded 10 Oct 2013)
  - First meetup, 28 Nov
- Sweeden
  - 43 members (founded 19 Aug 2011)
  - No meetings scheduled
- Berlin
  - 96 members (founded 17 Jul 2013)
  - 3 meetups (22, 9, 30 attendees)
- Munich
  - 18 members (founded 15 Oct 2013)
  - No meetups scheduled
- Toronto
  - 79 members (founded 2 Apr 2012)
  - 2 meetups (14, 24 attendees)
- Oslo
  - 178 members (founded 9 Aug 2007)
  - 15 meetups (44, 29, 9,... attendees)
- Dusseldorf
  - 4 members (founded 23 Aug 2013)
  - 2 meetups (1, 3 attendees)

- Only 19 **C++** Meetups in the world
- ~3,620 members in the World
- Only 10 ***active*** **C++** Meetups in the U.S.
- ~3,168 members in the U.S.

Software is no longer an individual sport;
software a team sport

It's important to be able to network with others

- to identify opportunities to collaborate
- to learn from others
- to teach others

CAREERS 2.0
by stackoverflow

in

GitHub

stackoverflow

My goals for the Meetup:

- To learn something that you can take back to your job
- Provide inspiration: passionate programmer
- Network with others: collaborate on other projects
- Share resources, knowledge
- Others?

Potential List of Topics for the C++ Meetup:

- C++11: Move Semantics, RAII, Smart Pointers, Lambdas, Variadic Templates, Memory Management
- C++14: What's coming?
- C++ Testing
- IDEs: Eclipse, NetBeans, MSVC++
- Compiler Comparison: gcc, Intel, LLVM, MSVC++
- Unit test frameworks: GoogleTest/Mock, CppUTest, etc
- Graphical frameworks: Qt, Cinder, wxWidgets
- Build frameworks: SCon, Cmake, Gradle, make
- Quality assurance: SonarQube, cppcheck
- Others?

Denver Tech Center C++ Developers:

- 3rd Monday of the month
- 6:30p – 8p
- Hosted here at Innovation Pavilion
    - note: doors lock at 7p; call if you can't get in
- Additional hosts in the future?
    - food, drink
    - door prizes (books, training, conferences)
- Upload presentations to Google Drive, other?

Meeting Format:

- 6:30p – 6:50p; food / drink
- 6:50p – 8p; formal / informal presentations
- C++ Meetup member presetations
- Lightning talks (10 min show and tell)
- Herb Sutter inspired quiz (Gotw) with teams
- Videos: C++ Now, Going Native, Pluralsight, Channel9
- Author presentations
- Pub quiz format
- Bring a project you're hacking on

Other Topics?

On to the Technical Presentation

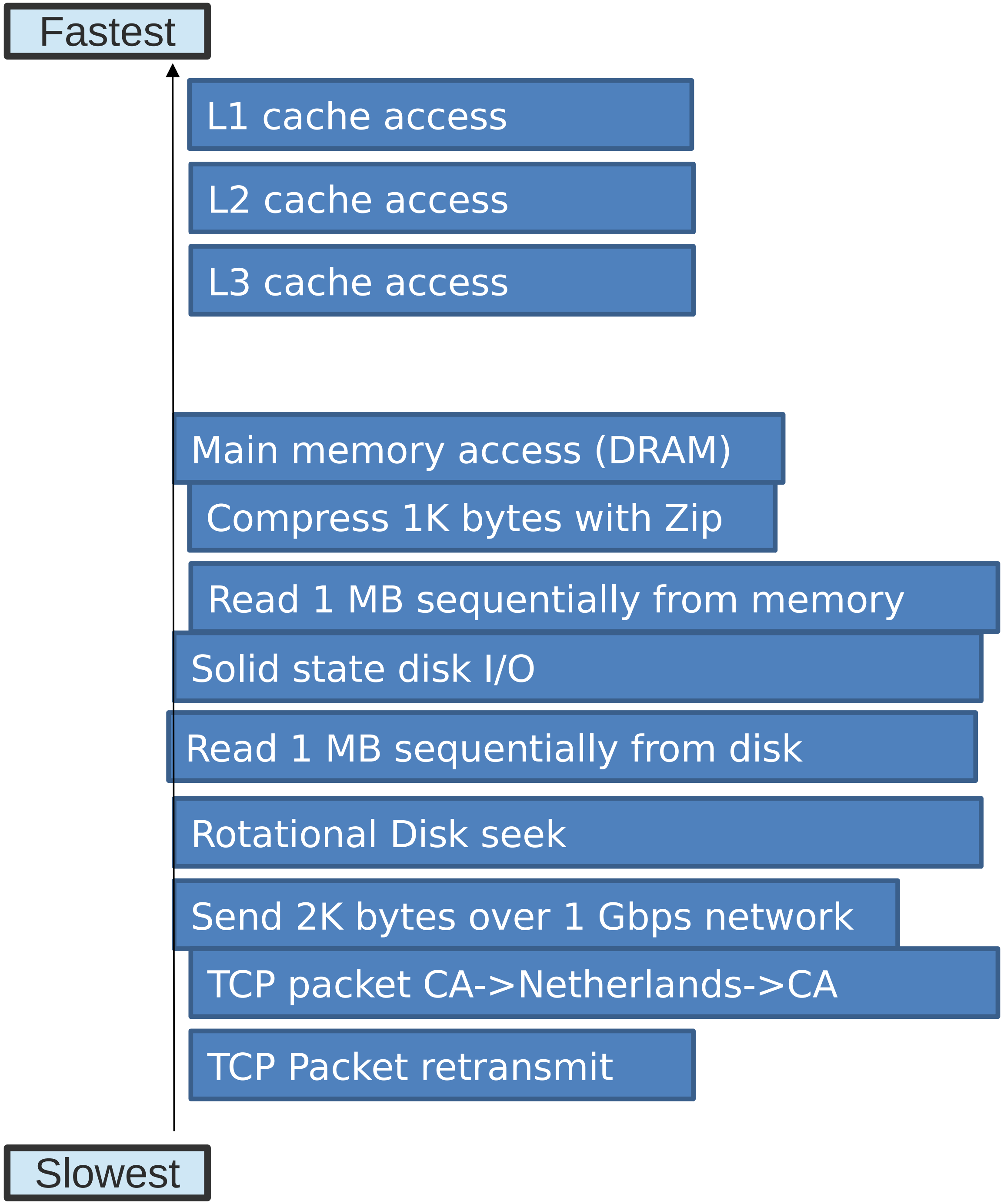
Why are we still using C++ in 2013?

It's (all) about performance!

But, I find that a lot of developer's never really grasped what this means, or lost track of it

Sometimes being off by orders of magnitude...

Stack these from the fastest (shortest time) to slowest (longest time)



Fastest

L1 cache access

L2 cache access

L3 cache access

Main memory access (DRAM)

Compress 1K bytes with Zip

Read 1 MB sequentially from memory

Solid state disk I/O

Read 1 MB sequentially from disk

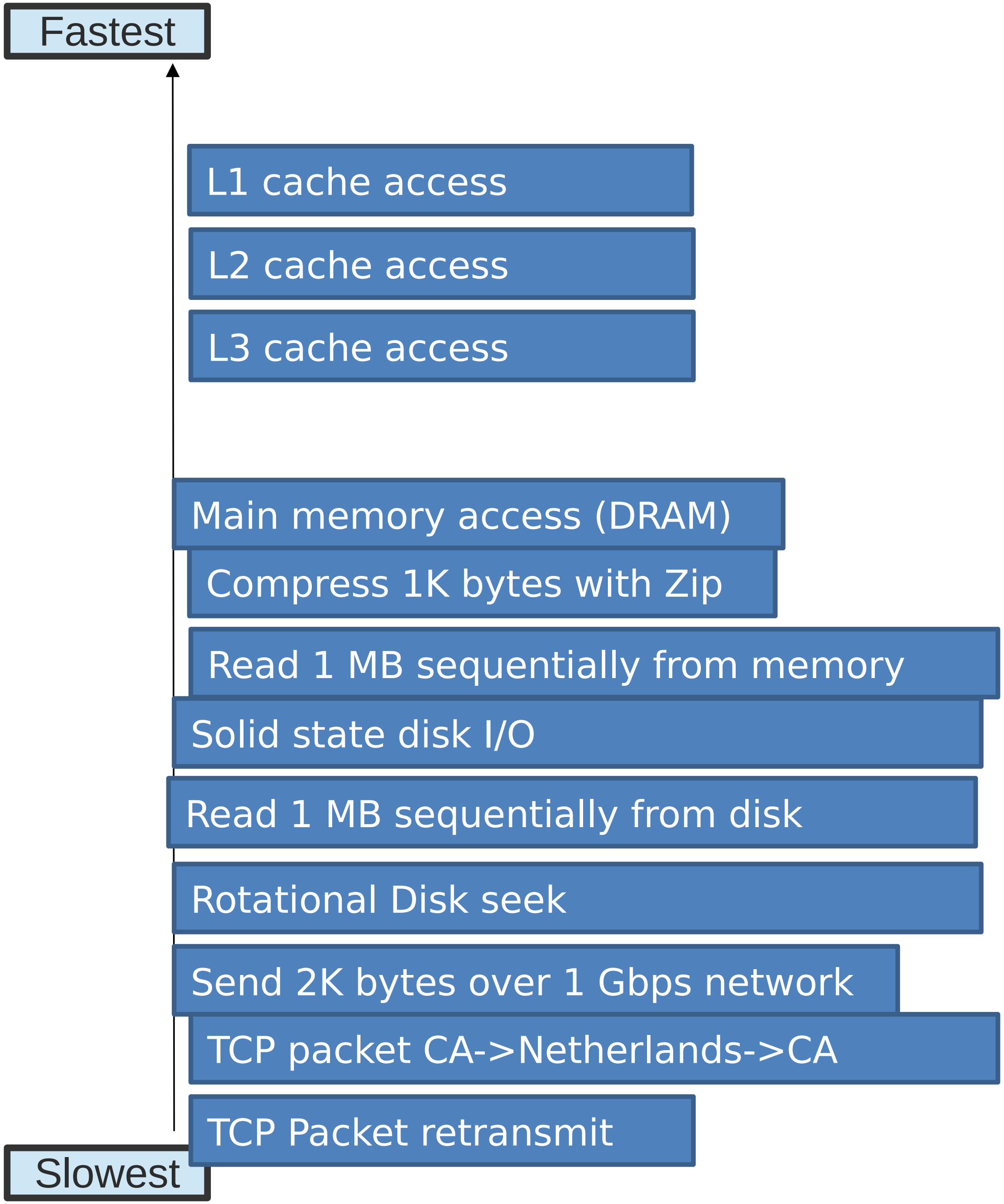
Rotational Disk seek

Send 2K bytes over 1 Gbps network

TCP packet CA->Netherlands->CA

TCP Packet retransmit

Slowest



Fastest

L1 cache access

L2 cache access

L3 cache access

Main memory access (DRAM)

Compress 1K bytes with Zip

Read 1 MB sequentially from memory

Solid state disk I/O

Read 1 MB sequentially from disk

Rotational Disk seek

Send 2K bytes over 1 Gbps network

TCP packet CA->Netherlands->CA

TCP Packet retransmit

Slowest

Key
L1 cache access
L2 cache access
L3 cache access
Main memory access (DRAM, from CPU)
Compress 1K bytes w Zip
Send 2K bytes over 1 Gbps network
Solid state disk I/O (flash memory)
Read 1 MB sequentially from memory
Rotational Disk seek
Read 1 MB sequentially from disk
TCP packet CA->Netherlands->CA
TCP packet retransmit

So, we feel pretty good, we got them in the right order.... from fastest to slowest

How much faster is one to the next?

It's about performance!

Event	Time (ns)
CPU Cycle (3.3GHz)	0.3
L1 cache access	0.9
L2 cache access	2.8
L3 cache access	12.9
Main memory access (DRAM, from CPU)	120.0
Compress 1K bytes w Zip	3,000.0
Send 2K bytes over 1 Gbps network	20,000.0
Solid state disk I/O (flash memory)	50,000 – 150,000
Read 1 MB sequentially from memory	250,000.0
Rotational Disk seek	10,000,000.0
Read 1 MB sequentially from disk	20,000,000.0
TCP packet CA->Netherlands->CA	150,000,000.0
TCP packet retransmit	2,000,000,000.0

- Jeff Dean (circa 2009) Google Fellow, Numbers Everyone should know
- Brendan Gregg: Systems Performance (circa 2013)

It's about performance!

Event	Time (ns)	Scaled Relative to CPU Cycle
CPU Cycle (3.3GHz)	0.3	1 sec
L1 cache access	0.9	3 sec
L2 cache access	2.8	9 sec
L3 cache access	12.9	43 sec
Main memory access (DRAM, from CPU)	120.0	6.5 minutes
Compress 1K bytes w Zip	3,000.0	2.7 hours
Send 2K bytes over 1 Gbps network	20,000.0	18.5 hours
Solid state disk I/O (flash memory)	50,000 – 150,000	2 – 6 days
Read 1 MB sequentially from memory	250,000.0	9 days 6 hrs
Rotational Disk seek	10,000,000.0	12 months
Read 1 MB sequentially from disk	20,000,000.0	24 months
TCP packet CA->Netherlands->CA	150,000,000.0	15 years
TCP packet retransmit	2,000,000,000.0	211 years

- Jeff Dean (circa 2009) Google Fellow, Numbers Everyone should know
- Brendan Gregg: Systems Performance (circa 2013)

15 minute video

http://channel9.msdn.com/Events/Lang-NEXT/Lang-NEXT-2012/-No

Move Semantics

- When can we take advantage of Move Semantics?
  - C++ compilers that support C++0x, C++11
  - Rvalue references
  - Standard library support

Wait, what did you say?

Rvalue Reference? What's that?

An Rvalue Reference is anything that does not have a name and that you can not take the address of

Huh? Don't worry about it, for now

Trust that the compiler will do the right thing

- Why Move Semantics?
  - Performance, of course
  - Designed to optimize data transfer between objects (or collections)
  - Especially, heap allocated objects

- How do we get Move operations?
  - Write a move constructor
  - Write a move assignment operator
  - The Standard Library already supports
    - Including std::string and the collection containers

- When you define your own types, if you have dynamically allocated memory that forces you to write the Big-3, you'll now need to write the Big-5.
  - Dtor – usually the easiest for us to recognize that we need to write
  - Copy Ctor
  - Assignment Operator
  - Move Ctor
  - Move Operator

- So, what happens if I compile our code with a new C++11 compiler, but I don't define any move constructors?
  - Most likely nothing; backward compatible
  - If your classes don't define any of the Big-3, the new compiler will auto generate them for you
  - If your classes define a single one of the Big-3, you won't get any move functions automatically generated

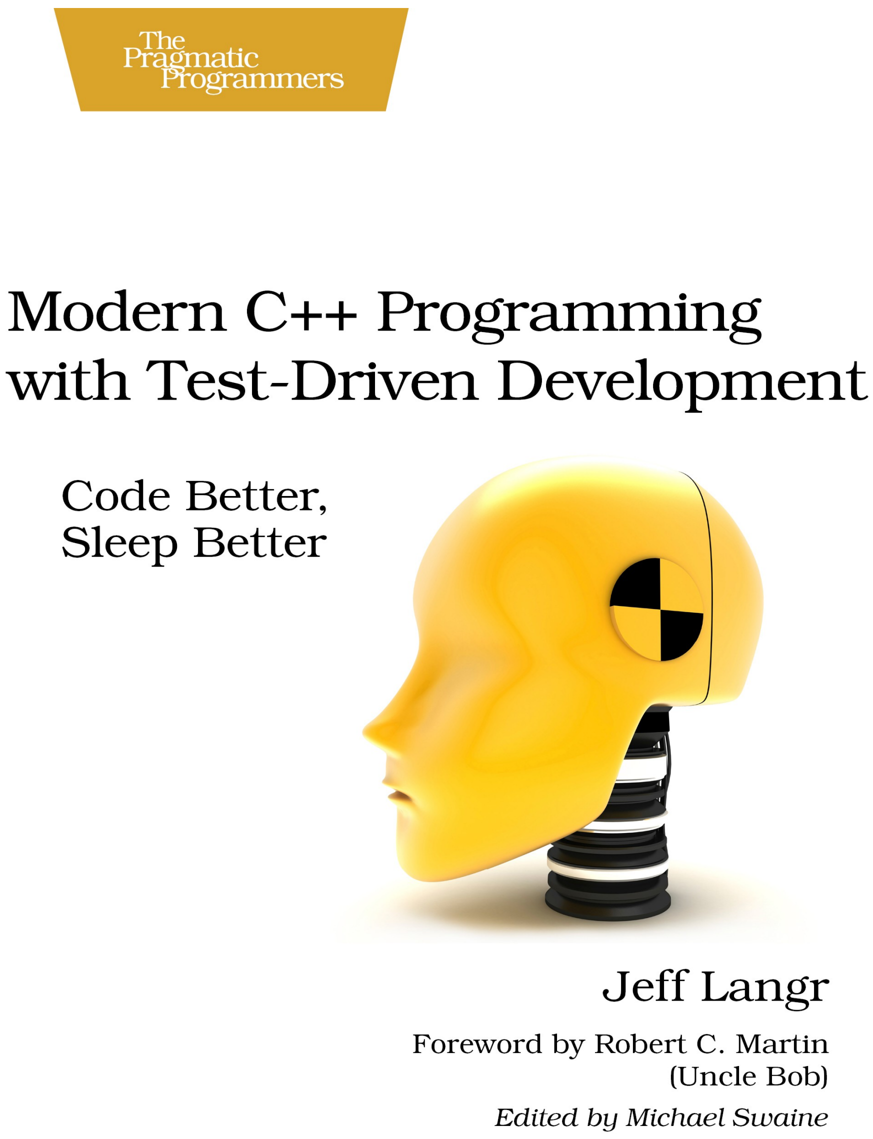
- When do I want to avoid move semantics?
  - I haven't found an instance yet...
  - The compiler is smart, it will know when it can use it and when it must avoid it

- Scott Meyer's example of move semantics at ACCU 2011 is very good... but 90 minutes

http://skillsmatter.com/podcast/home/move-semanticsperfect-forwarding-and-rvalue-references

Next Month

Leff Langr
Mon, 18 Dec
6:30pm



The Pragmatic Programmers

Modern C++ Programming with Test-Driven Development

Code Better, Sleep Better

Jeff Langr

Foreword by Robert C. Martin (Uncle Bob)

Edited by Michael Swaine

In C++14 we'll just write:

```
auto auto(auto auto) { auto; }
```

The compiler will infer the rest from context.

References

- http://eli.thegreenplace.net/2011/12/15/understanding-lvalues-and-rvalues-in-c-and-c/
- http://stackoverflow.com/questions/3601602/what-are-rvalues-lvalues-xvalues-glvalues-and-prvalues
- http://stackoverflow.com/questions/15482508/what-is-an-example-of-a-difference-in-allowed-usage-or-behavior-between-an-xv
- http://www.artima.com/cppsource/rvalue.html
- http://www.codeproject.com/Articles/397492/Move-Semantics-and-Perfect-Forwarding-in-Cplusplus
- http://www.aristeia.com/TalkNotes/ACCU2011_MoveSemantics.pdf
- http://skillsmatter.com/podcast/home/move-semanticsperfect-forwarding-and-rvalue-references
- http://stackoverflow.com/questions/3106110/what-is-move-semantics
- http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2002/n1377.htm (one of the original papers regarding Move Semantics)
- http://stackoverflow.com/questions/161053/c-which-is-faster-stack-allocation-or-heap-allocation (great comparison including a small program that can measure the performance difference)

- http://martin-moene.blogspot.com/2013/01/to-move-or-not-to-move.html
- http://thbecker.net/articles/rvalue_references/section_01.html (Thomas Becker Overview recommended by Herb Sutter)
- http://isocpp.org/blog/2012/11/universal-references-in-c11-scott-meyers (Herb Sutter)
- http://www.artima.com/cppsource/rvalue.html
- http://stackoverflow.com/questions/5481539/what-does-t-mean-in-c11

- http://www.gotw.ca/
- http://en.wikipedia.org/wiki/Performance_per_watt
- http://bulldozer00.com/category/c11/
- http://www.slideshare.net/olvemaudal/deep-c (pub quiz inspiration)
- http://isocpp.org
- http://cppreference.com