

# C++ 11 e o futuro do C++

ISO/IEC 14882:2011

(C++ 11)

Fabio Galuppo

<http://member.acm.org/~fabiogaluppo>

fabiogaluppo@acm.org

# Fabio Galuppo

Fabio Galuppo

About | C++ Renaissance


mvp.microsoft.com/en-us/mvp/Anonymous-9529

Microsoft

Sign in

Most Valuable Professional

Home About Features Find an MVP



**First year awarded:**  
2002

**Number of MVP Awards:**  
10

**Technical Expertise:**  
Visual C++

**Technical Interests:**  
XNA/DirectX, Visual C#, Visual F#

Language

Portuguese, English

Certifications

MCTS, MCSD for .NET, MCSD for VS6, MCAD, MCP, MCT

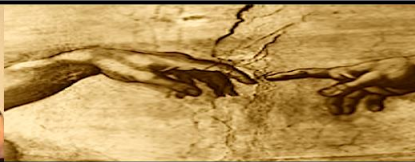

Fabio Galuppo

"The more you know, the more you learn; the more you learn, the more you can do; the more you can do, the more the opportunity..." — Dr. Richard W. Hamming

Biography

Software Engineer, Computer Scientist, Researcher, and Professional Trainer with large experience in Software Development and Architecture. My interests include: Compilers and Programming Languages (C, C++, F#, C#, Scala, Haskell, Objective-C, Python, Go, Lua, and Java), Math and Algorithms, Artificial Intelligence, Game and Graphics Programming, and Parallel/Concurrent Programming. You can find additional information here:  
<http://member.acm.org/~fabiogaluppo>

C++ Renaissance & Functional Revolution



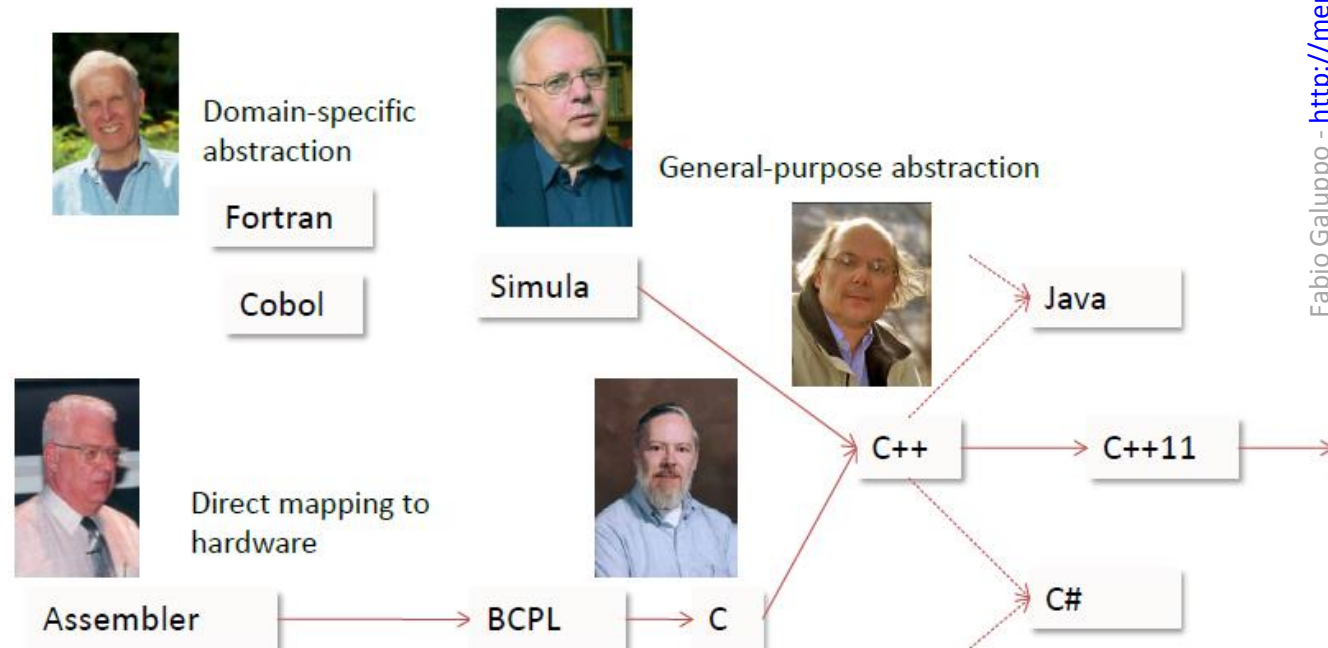
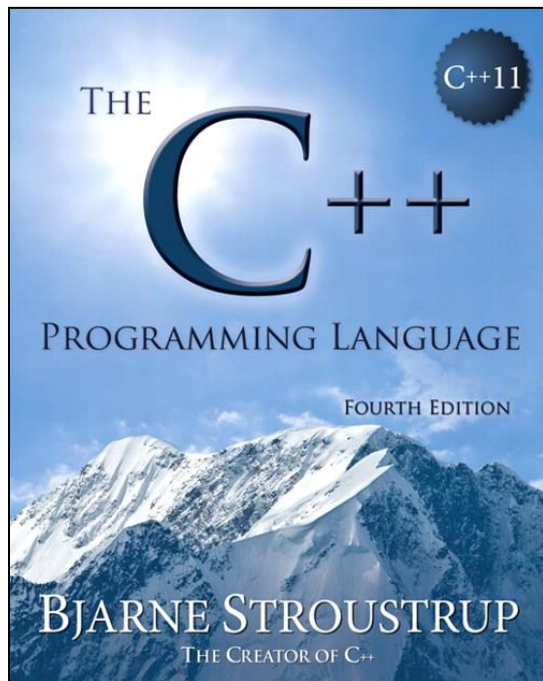
Home About

# The C++ Programming Language

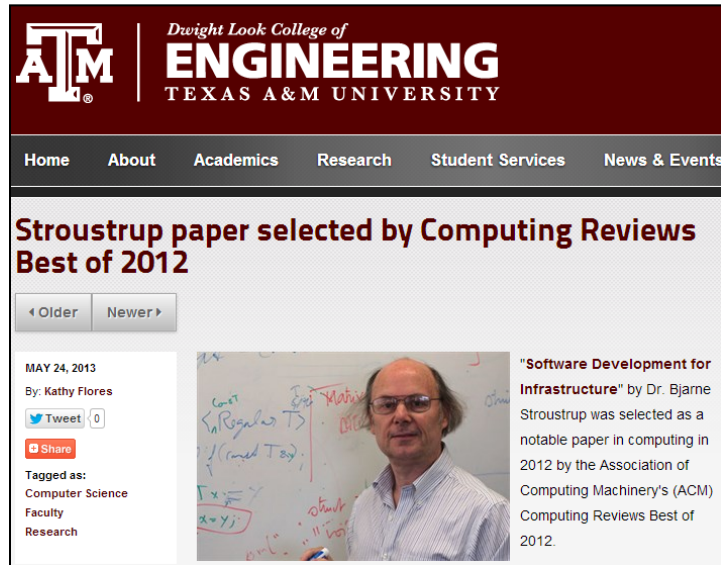
C++ is a general purpose programming language with a bias towards systems programming that

- is a better C
- supports data abstraction
- supports object-oriented programming
- supports generic programming.

C++ is a general purpose programming language designed to make programming more enjoyable for the serious programmer.



# C++



IEEE Computer Magazine Volume:45, Issue: 1

<http://dx.doi.org/10.1109/MC.2011.353>

“A light-weight abstraction programming language

Key strengths:

- software infrastructure
- resource-constrained applications”

C++ 11

The Future is here

-- <http://www.infoq.com/presentations/Cplusplus-11-Bjarne-Stroustrup>

# Linguagem + Biblioteca

C++ 14: <https://github.com/cplusplus/draft>

Document Number: N3797  
Date: 2013-10-13  
Revises: N3691  
Reply to: Stefanus Du Toit  
cxxeditor@gmail.com

Working Draft, Standard for Programming  
Language C++

acesso em: 10 de Novembro de 2013

C++ 11: ISO/IEC 14882:2011

## Abstract

ISO/IEC 14882:2011 specifies requirements for implementations of the C++ programming language. The first such requirement is that they implement the language, and so ISO/IEC 14882:2011 also defines C++. Other requirements and relaxations of the first requirement appear at various places within ISO/IEC 14882:2011.

C++ is a general purpose programming language based on the C programming language as specified in ISO/IEC 9899:1999. In addition to the facilities provided by C, C++ provides additional data types, classes, templates, exceptions, namespaces, operator overloading, function name overloading, references, free store management operators, and additional library facilities.

Table 4 — Keywords

alignas	continue	friend	register	true
alignof	decltype	goto	reinterpret_cast	try
asm	default	if	return	typedef
auto	delete	inline	short	typeid
bool	do	int	signed	typename
break	double	long	sizeof	union
case	dynamic_cast	mutable	static	unsigned
catch	else	namespace	static_assert	using
char	enum	new	static_cast	virtual
char16_t	explicit	noexcept	struct	void
char32_t	export	nullptr	switch	volatile
class	extern	operator	template	wchar_t
const	false	private	this	while
constexpr	float	protected	thread_local	
const_cast	for	public	throw	

2.12 Keywords [lex.key] [2. Lexical conventions]

# Compiladores



Summary of C++11 Feature Availability: <http://www.aristeia.com/C++11/C++11FeatureAvailability.htm>

# Eu quero aprender C++ 11!

## Por onde começar?

Bjarne Stroustrup

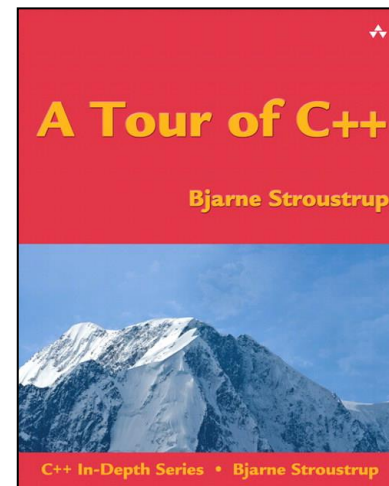
The Essence of C++: With Examples in C++84, C++98, C++11, and C++14

### Abstract

- C++11 is being deployed and the shape of C++14 is becoming clear. This talk examines the foundations of C++. What is essential? What sets C++ apart from other languages? How do new and old features support (or distract from) design and programming relying on this essence.
- I focus on the abstraction mechanisms (as opposed to the mapping to the machine): Classes and templates. Fundamentally, if you understand vector, you understand C++.
- Type safety and resource safety are key design aims for a program. These aims must be met without limiting the range of applications and without imposing significant run-time or space overheads. I address issues of resource management (garbage collection is not an ideal answer and pointers should not be used as resource handles), generic programming (we must make it simpler and safer), compile-time computation (how and when?), and type safety (casts belong in the lowest-level hardware interface). I will touch upon move semantics, exceptions, concepts, type aliases, and more. My aim is not so much to present novel features and technique, but to explore how C++'s feature set supports a new and more effective design and programming style.
- Primary audience
  - Experienced programmers with weak C++ understanding
  - Academics/Teachers/Mentors
  - Architects (?)

Stroustrup - Essence - Going Native'13

2



<http://channel9.msdn.com/Events/GoingNative/2013/Opening-Keynote-Bjarne-Stroustrup>



# C++ 11, hoje! C++ 14, no ano que vem!

<http://isocpp.org/blog/2013/05/gcc-4.8.1-released-c11-feature-complete>

## GCC 4.8.1 released, C++11 feature complete

By Seth I May 31, 2013 11:10 AM | News, Product News | Tags: None

The release of **GCC 4.8.1** was [announced](#) today (31 May 2013) on the gcc mailing list. In addition to many bug fixes, GCC 4.8.1 adds support for C++11 ref-qualifiers, the final missing C++11 feature. This makes GCC the first C++11 [Ed: language specification] feature complete compiler to be released.

Information on the full set of changes is available on the [GCC 4.8 series](#) page.



**Clang 3.3**, also C++11 feature complete, is in release testing and the release is currently scheduled for June 5th. Within a week full C++11 support will be available from two major compilers and on numerous platforms supported by those compilers.

Forging ahead with support for the [next C++ standard](#).



## Clang is (draft) C++14 feature-complete!

By Blog Staff | Nov 7, 2013 02:34 PM | News, Product News | Tags: None

A few hours ago, [Clang completed checkin 194194 to be feature-complete for draft C++14](#) including both language extensions and standard library features. (Note: The library conformance requires using libc++, instead of libstdc++ which is supported on more platforms but is not as conforming.) Congratulations to the Clang team for this achievement!

With this progress, it appears that the next release of Clang and LLVM, expected in December or January, will be draft C++14 feature-complete. C++14 itself may still undergo final changes at the February 2014 ISO C++ meeting, which is expected to be the final meeting for technical tweaks to the contents of C++14.

<http://isocpp.org/blog/2013/11/clang-is-draft-c14-feature-complete>

# “C++14 completes C++11”

## The future of C++

Herb Sutter  
Partner Program Manager  
2-306

<http://channel9.msdn.com/Events/Build/2013/2-306>

<http://msdn.microsoft.com/en-us/library/vstudio/hh567368.aspx>

## C++11 Features (Modern C++)

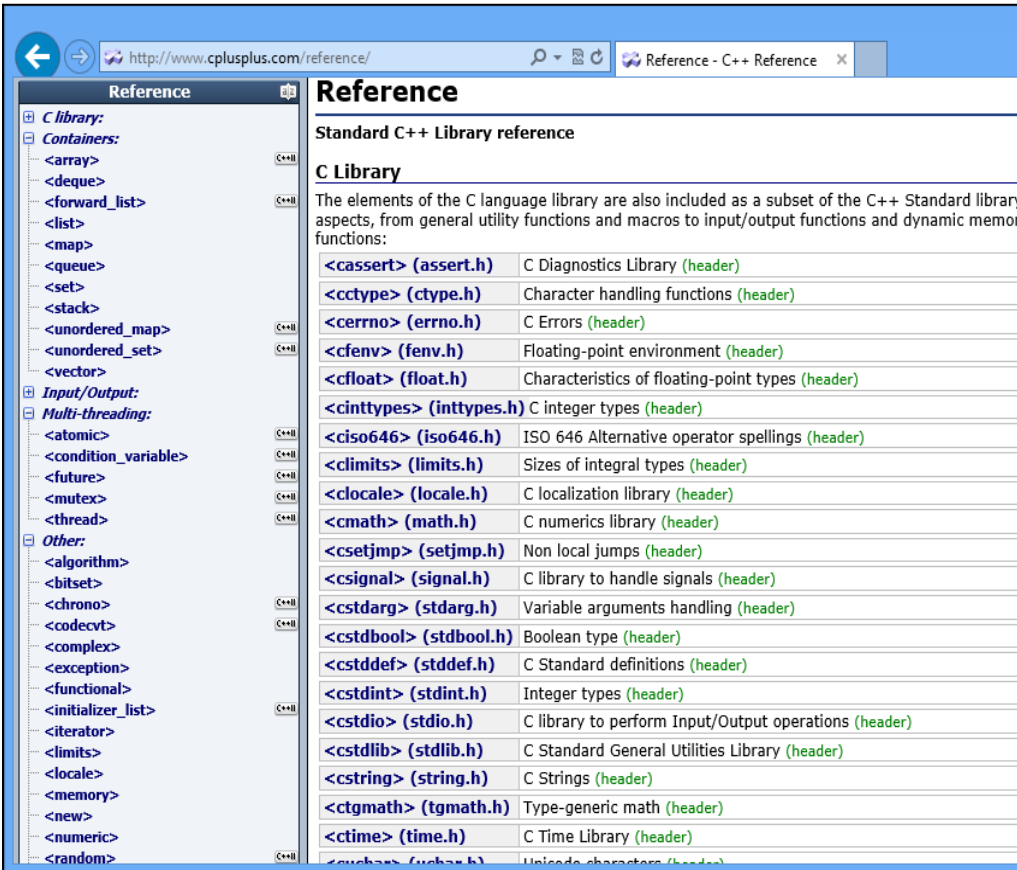
Visual Studio 2012 | Other Versions ▼ | 32 out of 36 rated this helpful - [Rate this topic](#)

This document describes the features of the new C++ Standard—also known as C++11—that are implemented in Visual C++.

### ▲ C++11 Core Language Features

Visual C++ 2010 implemented many features in the C++0x core language specification, which was the precursor to C++11, and Visual C++ in Visual Studio 2012 expands on that to include many C++11 features. The following table lists C++11 core language features and their implementation status in both Visual C++ 2010 and Visual C++ in Visual Studio 2012.

# Standard Template Library



The screenshot shows the 'Reference' page on <http://www.cplusplus.com/reference/>. The page is titled 'Reference' and 'Standard C++ Library reference'. It lists various C++ standard library components categorized into 'C Library', 'Input/Output', 'Multi-threading', and 'Other'.

**Reference**

**Standard C++ Library reference**

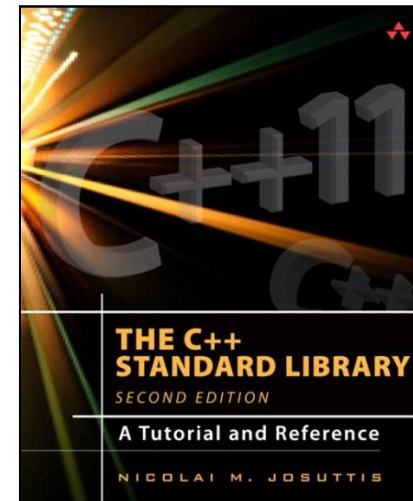
**C Library**

The elements of the C language library are also included as a subset of the C++ Standard library, aspects, from general utility functions and macros to input/output functions and dynamic memory functions:

<code>&lt;cassert&gt;</code> ( <code>assert.h</code> )	C Diagnostics Library (header)
<code>&lt;cctype&gt;</code> ( <code>ctype.h</code> )	Character handling functions (header)
<code>&lt;cerrno&gt;</code> ( <code>errno.h</code> )	C Errors (header)
<code>&lt;cfenv&gt;</code> ( <code>fenv.h</code> )	Floating-point environment (header)
<code>&lt;cfloat&gt;</code> ( <code>float.h</code> )	Characteristics of floating-point types (header)
<code>&lt; cinttypes&gt;</code> ( <code>inttypes.h</code> )	C integer types (header)
<code>&lt;ciso646&gt;</code> ( <code>iso646.h</code> )	ISO 646 Alternative operator spellings (header)
<code>&lt;climits&gt;</code> ( <code>limits.h</code> )	Sizes of integral types (header)
<code>&lt;locale&gt;</code> ( <code>locale.h</code> )	C localization library (header)
<code>&lt;cmath&gt;</code> ( <code>math.h</code> )	C numerics library (header)
<code>&lt; csetjmp&gt;</code> ( <code>setjmp.h</code> )	Non local jumps (header)
<code>&lt;csignal&gt;</code> ( <code>signal.h</code> )	C library to handle signals (header)
<code>&lt;cstdarg&gt;</code> ( <code>stdarg.h</code> )	Variable arguments handling (header)
<code>&lt;cstdbool&gt;</code> ( <code>stdbool.h</code> )	Boolean type (header)
<code>&lt;cstddef&gt;</code> ( <code>stddef.h</code> )	C Standard definitions (header)
<code>&lt;cstdint&gt;</code> ( <code>stdint.h</code> )	Integer types (header)
<code>&lt;cstdio&gt;</code> ( <code>stdio.h</code> )	C library to perform Input/Output operations (header)
<code>&lt;cstdlib&gt;</code> ( <code>stdlib.h</code> )	C Standard General Utilities Library (header)
<code>&lt;cstring&gt;</code> ( <code>string.h</code> )	C Strings (header)
<code>&lt;ctgmath&gt;</code> ( <code>tgmath.h</code> )	Type-generic math (header)
<code>&lt;ctime&gt;</code> ( <code>time.h</code> )	C Time Library (header)
<code>&lt;uchar&gt;</code> ( <code>uchar.h</code> )	Unicode characters (header)

<http://www.cplusplus.com>

Selected Standard Library Headers		
<code>&lt;algorithm&gt;</code>	<code>copy()</code> , <code>find()</code> , <code>sort()</code>	§iso.25
<code>&lt;array&gt;</code>	<code>array</code>	§iso.23.3.2
<code>&lt;chrono&gt;</code>	<code>duration</code> , <code>time_point</code>	§iso.20.11.2
<code>&lt;cmath&gt;</code>	<code>sqrt()</code> , <code>pow()</code>	§iso.26.8
<code>&lt;complex&gt;</code>	<code>complex</code> , <code>sqrt()</code> , <code>pow()</code>	§iso.26.8
<code>&lt;forward_list&gt;</code>	<code>forward_list</code>	§iso.23.3.4
<code>&lt;fstream&gt;</code>	<code>fstream</code> , <code>ifstream</code> , <code>ofstream</code>	§iso.27.9.1
<code>&lt;future&gt;</code>	<code>future</code> , <code>promise</code>	§iso.30.6
<code>&lt;iostream&gt;</code>	<code>hex</code> , <code>dec</code> , <code>scientific</code> , <code>fixed</code> , <code>defaultfloat</code>	§iso.27.5
<code>&lt;istream&gt;</code>	<code>istream</code> , <code>ostream</code> , <code>cin</code> , <code>cout</code>	§iso.27.4
<code>&lt;map&gt;</code>	<code>map</code> , <code>multimap</code>	§iso.23.4.4
<code>&lt;memory&gt;</code>	<code>unique_ptr</code> , <code>shared_ptr</code> , <code>allocator</code>	§iso.20.6
<code>&lt;random&gt;</code>	<code>default_random_engine</code> , <code>normal_distribution</code>	§iso.26.5
<code>&lt;regex&gt;</code>	<code>regex</code> , <code>smatch</code>	§iso.28.8
<code>&lt;string&gt;</code>	<code>string</code> , <code>basic_string</code>	§iso.21.3
<code>&lt;set&gt;</code>	<code>set</code> , <code>multiset</code>	§iso.23.4.6
<code>&lt;sstream&gt;</code>	<code>istringstream</code> , <code>ostringstream</code>	§iso.27.8
<code>&lt;stdexcept&gt;</code>	<code>length_error</code> , <code>out_of_range</code> , <code>runtime_error</code>	§iso.19.2
<code>&lt;thread&gt;</code>	<code>thread</code>	§iso.30.3
<code>&lt;unordered_map&gt;</code>	<code>unordered_map</code> , <code>unordered_multimap</code>	§iso.23.5.4
<code>&lt;utility&gt;</code>	<code>move()</code> , <code>swap()</code> , <code>pair</code>	§iso.20.1
<code>&lt;vector&gt;</code>	<code>vector</code>	§iso.23.3.6






# Programação Concorrente com C++ 11

```
#include <thread>
#include <future>
#include <atomic>
#include <mutex>
#include <condition_variable>
```



artigo online: <http://msdn.microsoft.com/en-us/magazine/hh852594.aspx>  
código fonte: <http://archive.msdn.microsoft.com/mag201203CPP>

library	
<b>Multi-threading</b> 	
<b>Atomic and thread support</b>	
Support for atomics and threads:	
● <b>Headers</b>	
<atomic>	Atomic (header)
<thread>	Thread (header)
<mutex>	Mutex (header)
<condition_variable>	Condition variable (header)
<future>	Future (header)

This article discusses Visual C++ 11, a prerelease technology. All related information is subject to change.

This article discusses:

- Parallel execution
- Asynchronous tasks
- Threads
- Variables and exceptions
- Synchronization
- Atomic types
- Mutexes and locks
- Condition variables

Technologies discussed:

Visual C++ 11

Code download available at:

[code.msdn.microsoft.com/mag201203CPP](http://code.msdn.microsoft.com/mag201203CPP)

**DIEGO DAGUM** is a software developer with more than 20 years of experience. He's currently a Visual C++ community program manager with Microsoft.

**THANKS** to the following technical experts for reviewing this article:  
David Cravey, Alon Fliess, **Fabio Galuppo** and Marc Gregoire

# Quem depende de C++?

- Node.js



GitHub, Inc. [US] <https://github.com/joyent/node/tree/master/src>

fs_event_wrap.cc	lib, src: upgrade
handle_wrap.cc	lib, src: upgrade
handle_wrap.h	lib, src: upgrade
macros.py	Improve gyp build
node.cc	Merge remote-tr
node.d	dtrace: add miss
node.h	build: fix window
node.js	buffer: use small
node.stp	systemtap: add
node_buffer.cc	build: fix window
node_buffer.h	buffer: write strin
node_constants.cc	constants: add C

- HipHop for PHP (HPHP)



<http://developers.facebook.com/blog/post/2010/02/02/hiphop-for-php--move-fast/>

HipHop for PHP isn't technically a compiler itself. Rather it is a source code transformer. HipHop programmatically transforms your PHP source code into highly optimized C++ and then uses g++ to compile it. HipHop executes the source code in a semantically equivalent manner and sacrifices some rarely used features — such as eval() — in exchange for improved performance. HipHop includes a code transformer, a reimplement of PHP's runtime system, and a rewrite of many common PHP Extensions to take advantage of these performance optimizations.

- mongoDB




GitHub, Inc. [US] <https://github.com/mongodb/mongo/tree/master/src/mongo/db>

background.cpp	clear namespace string so that it can be
background.h	namespace string -> namespace_string
btree.cpp	SERVER-10084 New logging implementat
btree.h	SERVER-8791 SERVER-9165 SERVER-9
btree_stats.cpp	SERVER-9242 Use MONGO_INITIALIZER
btree_stats.h	SERVER-9242 Use MONGO_INITIALIZER
btreebuilder.cpp	SERVER-10084 New logging implementat
btreebuilder.h	SERVER-3067 Add killop support for foreg
btreecursor.cpp	SERVER-8791 SERVER-9212 keep btree
btreecursor.h	SERVER-8791 SERVER-9212 keep btree
btreeposition.cpp	SERVER-1752 Fix rhel compile by includi
btreeposition.h	SERVER-1752 Optimize simple indexed c

MongoDB (from "humongous") is an open-source document database, and the leading NoSQL database. Written in C++, MongoDB features:

# Cinder C++

<http://libcinder.org/>



## CINDER

ABOUT FEATURES GALLERY REFERENCE FORUM **BLOG** DOWNLOAD

---

**JUNE 29, 2013**  
**01:29 PM**

### CINDER WINS INNOVATION LION

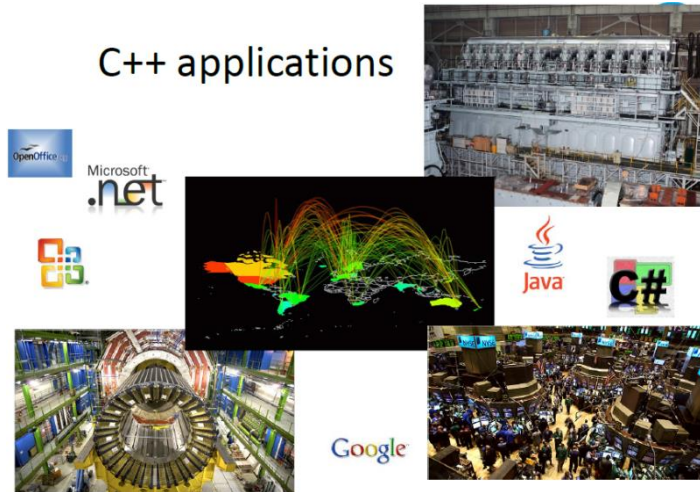
A pretty remarkable thing happened recently. Cinder won the highest honor in the advertising industry, a Grand Prix at the [Cannes Lions](#) festival. In fact, Cinder was awarded the first ever Grand Prix for a new brand-new category, *Innovation*. We're obviously quite proud of this fact, and have been amazed at all of the new attention Cinder is getting.

You can read all about it in various publications like [Fast Company](#), [Creativity](#), and [Advertising Age](#).

[http://libcinder.org/blog/posts/8\\_cinder-wins-innovation-lion/](http://libcinder.org/blog/posts/8_cinder-wins-innovation-lion/)

# Mais aplicações de C++

## C++ applications



## C++ Applications

- [www.research.att.com/~bs/applications.html](http://www.research.att.com/~bs/applications.html)



## C++ Applications



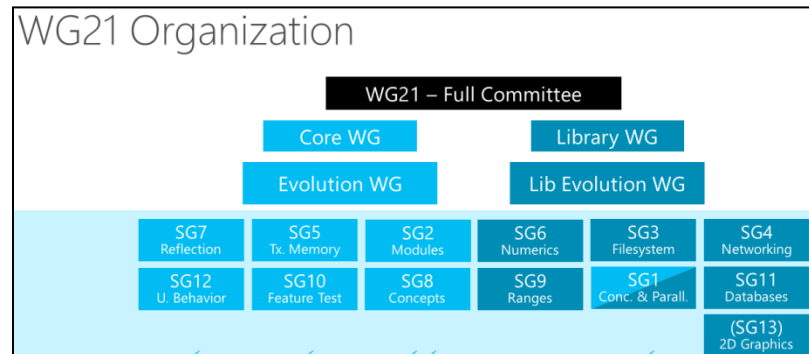
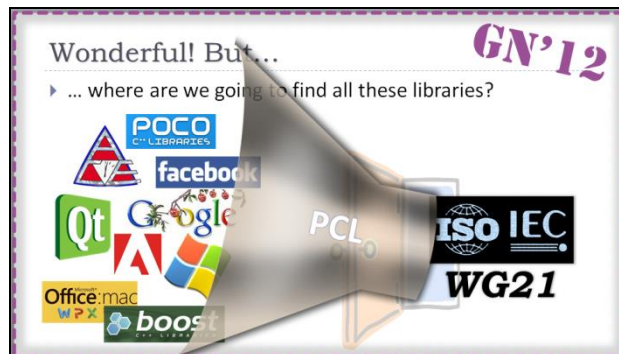
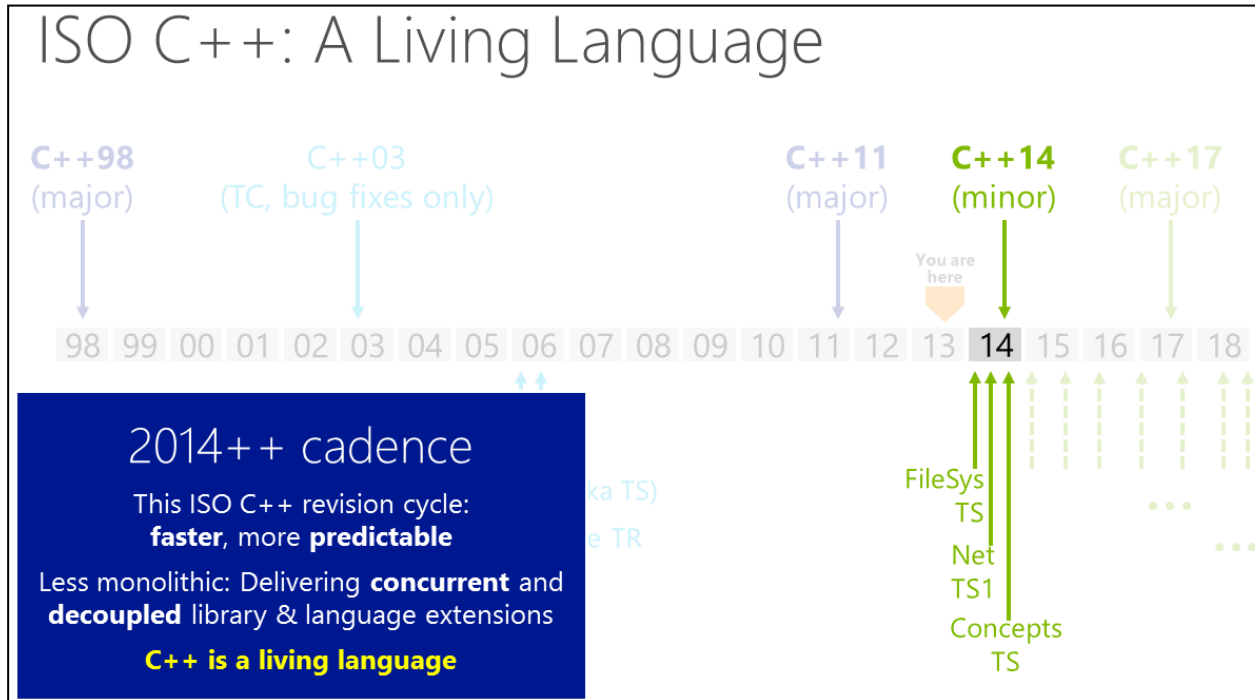
<http://www.stroustrup.com/applications.html>

<http://www.lextrait.com/vincent/implementations.html>

# ISO C++

<http://isocpp.org/>

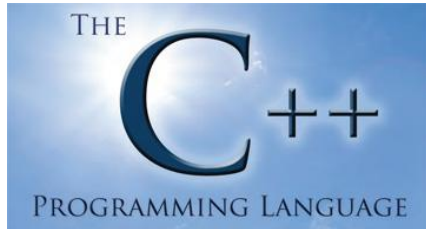
Herb Sutter  
One C++



<http://channel9.msdn.com/Events/GoingNative/2013/Keynote-Herb-Sutter-One-Cpp>



# It's all about Polyglot Programming!



C++ supports systems programming. This implies that C++ code is able to effectively interoperate with software written in other languages on a system. The idea of writing all software in a single language is a fantasy. From the beginning, C++ was designed to interoperate simply and efficiently with C, assembler, and Fortran. By that, I meant that a C++, C, assembler, or Fortran function could call functions in the other languages without extra overhead or conversion of data structures passed among them.

<http://www.youtube.com/watch?v=NvWTnIoQZj4>



Bjarne Stroustrup: The 5 Programming Languages You Need to Know

“Nobody should call themselves a professional if they only knew one language.”

...**C++**, of course; **Java**; maybe **Python** for mainline work... And if you know those, you can't help know sort of a little bit about **Ruby** and **JavaScript**, you can't help knowing **C** because that's what fills out the domain and of course **C#**. But again, **these languages create a cluster so that if you knew either five of the ones that I said, you would actually know the others...**

“Inclua a esta lista **F#**, **Scala**, **Haskell**, **Erlang**, **Clojure**, e/ou **Racket**” – Fabio Galuppo