



Core библиотеки: идеи и примеры

Александр Еналдиев

Kaspersky

folly:: absl:: ...

Содержание

- ABI (Application Binary Interface) и текущие подходы к решению проблем с ним ;
- Идеи при построении библиотек:
 - Сборка библиотеки ;
 - “Live at head” Principle ;
 - Решаемые задачи ;

C++ библиотеки: Abseil




Доклады:

- CppCon 2017: Titus Winters “Hands-On With Abseil”
- CppCon 2017: Matt Kulkundis “Designing a Fast, Efficient, Cache-friendly Hash Table, Step by Step”
- CppCon 2018: Titus Winters “Modern C++ Design”
- CppCon 2019: Titus Winters “Maintainability and Refactoring Impact of Higher-Level Design Features”
- CppCon 2019: Matt Kulkundis “Abseil's Open Source Hashtables: 2 Years In”
- ...

C++ библиотеки: Abseil



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



TITUS WINTERS

What is Abseil

- Zero config
- Utility code
- string routines
- Debugging / analysis facilities
- Guidance (Tip of the Week)
- C++11-compatible versions of standard types (pre-adopt)
- Standards-alternatives

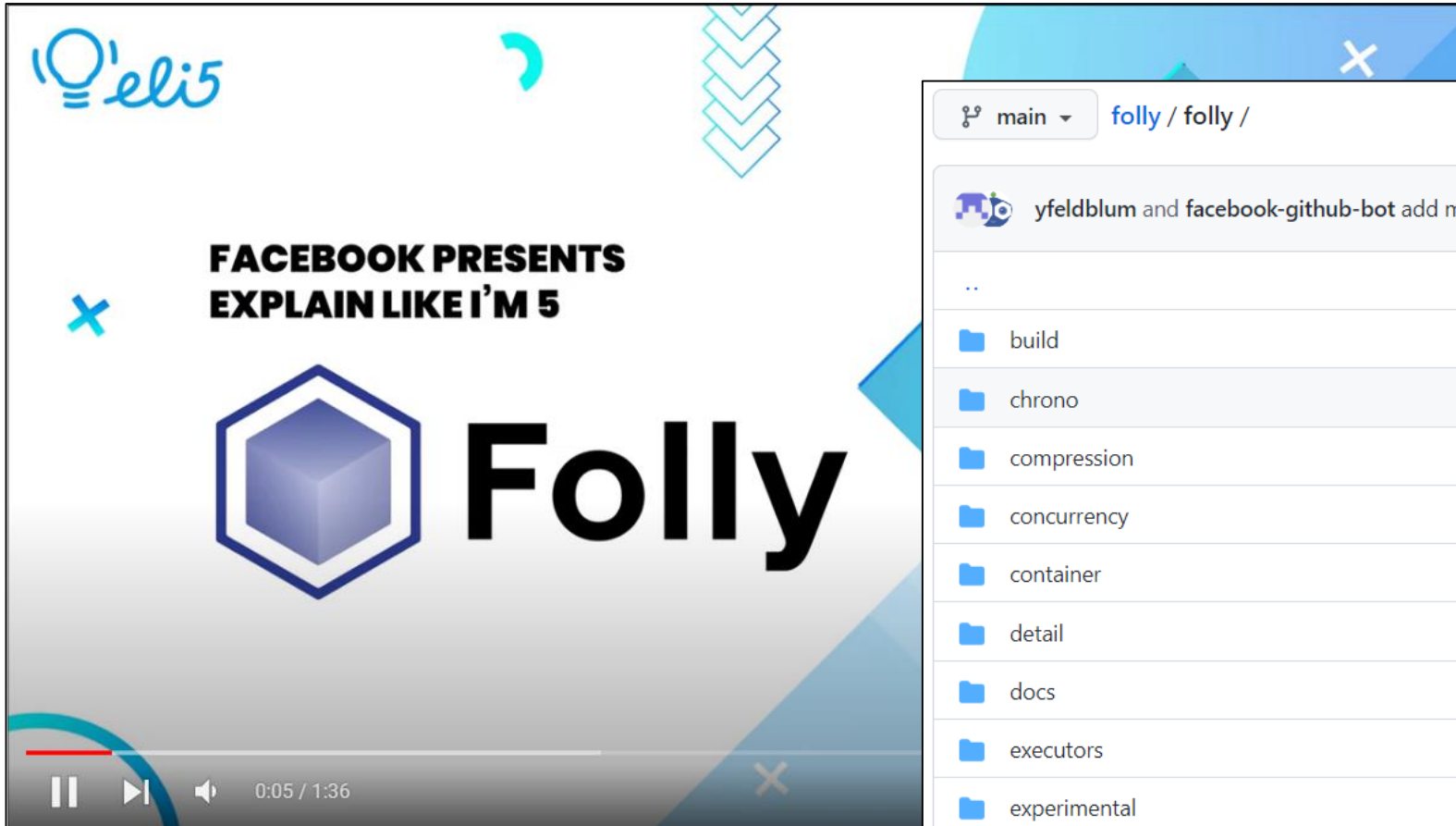
General goal: Support 5 years back where possible.

C++ as a
"Live at Head"
Language

CppCon.org

<https://www.youtube.com/watch?v=tISy7EJQPzI>

C++ библиотеки: Folly



main - folly / folly /		Fork 4.6k	Star 21.9k
yfeldblum and facebook-github-bot add missing sanitizer suppression in MicroLock ...			
..			
build	use gmock 1.10 instead of 1.8		
chrono	Update copyright headers from Facebook to Meta		
compression	Expose Method to Clear Compression Context Pools		
concurrency	folly::ConcurrentHashMap assign_if		
container	let small_vector handle MaxInline = 0 under MSVC (#1775)		
detail	Remove -inl.h include cycle from folly/detail		
docs	Add an Rcu.md documentation page for folly RCU		
executors	Implement step() in folly Manual Executor		
experimental	Provide a Default Initializer for the Huge Page Allocator		
ext	folly::ext::test_find_resource		
external	update farmhash.cpp to avoid an unused function error on an		

C++ библиотеки: Folly



What is folly?

Folly (acronymed loosely after Facebook Open Source) designed with practicality and efficiency in mind. Folly components used extensively at Facebook. In particular, other open source C++ efforts and place where those

It complements (as opposed to competing against) other. In fact, we embark on defining our own component if available, or does not meet the needed performance. `std` or Boost obsoletes them.

Performance concerns permeate much of Folly, some would otherwise be (see e.g. `PackedSyncPtr.h`, `Small` in all of Folly).

main	folly / folly /	Fork 4.6k	Star 21.9k
yfeldblum and facebook-github-bot add missing sanitizer suppression in MicroLock ...			
..			
build	use gmock 1.10 instead of 1.8		
chrono	Update copyright headers from Facebook to Meta		
compression	Expose Method to Clear Compression Context Pools		
concurrency	folly::ConcurrentHashMap assign_if		
container	let small_vector handle MaxInline = 0 under MSVC (#1775)		
detail	Remove -inl.h include cycle from folly/detail		
docs	Add an Rcu.md documentation page for folly RCU		
executors	Implement step() in folly Manual Executor		
experimental	Provide a Default Initializer for the Huge Page Allocator		
ext	folly::ext::test_find_resource		
external	update farmhash.cpp to avoid an unused function error on an		

C++ библиотеки: Folly



Доклады:

- CppCon 2016: Nicholas Ormrod “The strange details of std::string at Facebook”
- CppCon 2016: David Watson “Experiences with Facebook's C++ library”

C++ библиотекы: The ABI

```
struct IntAndString
{
    int i;
    std::string s;
};
```

```
template<>
struct std::hash<std::string> {
    std::size_t operator() (std::string const& s) const noexcept
    {
        return _FNV1a_append_bytes(std::_FNV_offset_basis,
                                   &*s.begin(), s.size());
    }
};
```

```
template<>
struct std::hash<std::string> {
    std::size_t operator() (std::string const& s) const noexcept
    {
        boost::crc_32_type result;
        result.process_bytes(s.data(), s.size());
        return static_cast<std::size_t>(result.checksum());
    }
};
```

Core библиотеки: The ABI

- N4028 Defining a Portable C++ ABI ; (Herb Sutter)

“...The programmer needs to be able to distinguish between code that uses “the usual `std::library`” (whatever that is today, which usually is different for different C++ implementations on the same platform and could even be a user-provided standard library if they are not using the one that came with their compiler) and code that uses the target platform’s C++ standard library ABI. C++ already has a way to designate the C++ standard library types and functions: namespace `std`.

For symmetry with extern “abi”, we propose the namespace `std::abi` ... This gives us two distinct namespaces for two distinct thing:

- ***`std`** contains “the C++ standard library implementation that can change/evolve.” This is provided by each C++ implementer, exactly as today.*
- ***`std::abi`** contains “the C++ standard library implementation that is binary stable.” This is provided by each OS platform, however it wants, and is shared by all compilers that target that platform. A likely choice is to make this the release build of a snapshot of the OS platform owner’s own C++ product’s implementation of `std`, taken at the time they support the C++ ABI.*

For convenience and safety, in an extern “abi” block `std::` means `std::abi::` ...”

C++ библиотекы: The ABI

- N4028 Defining a Portable C++ ABI ;
- P1863 ABI - Now Or Never ;
- P2028 What is ABI, and What Should

WG21 Do About It? ; (Titus Winters)

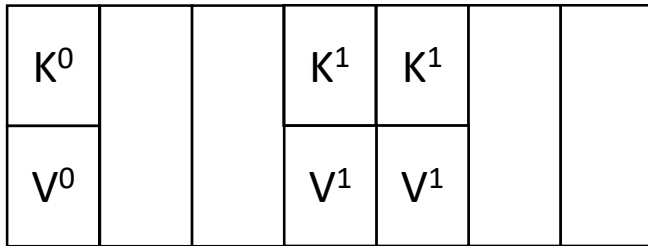
- `std::hash`, `std::unordered_map` impl;
- `std::regex` impl ;
- `lock_guard` vs `scoped_lock` ;
- `push_back` vs `emplace_back` ;
- ...

(Herb Sutter)



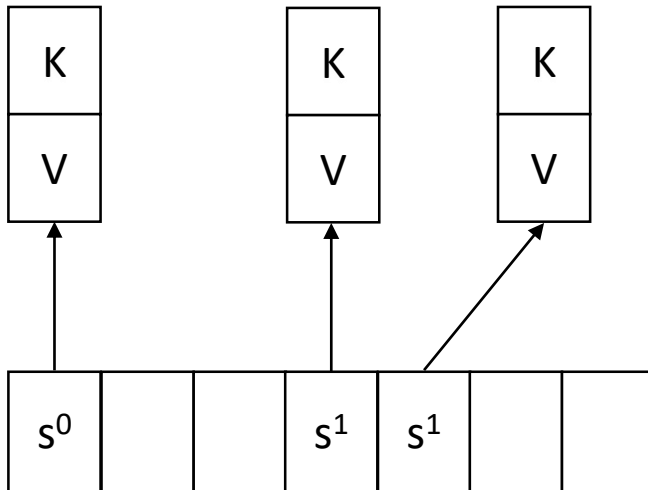
C++ библиотечи: The ABI

- `absl::Hash`, `absl::HashOf` ;
- `absl::flash_hash_set` / `absl::flat_hash_map` ;



```
struct A {  
    int m;  
  
    template <typename H>  
    friend H AbslHashValue(H state, const A& v) {  
        return H::combine(std::move(state), v.m);  
    }  
};
```

- `absl::node_hash_set`
`absl::node_hash_map` ;

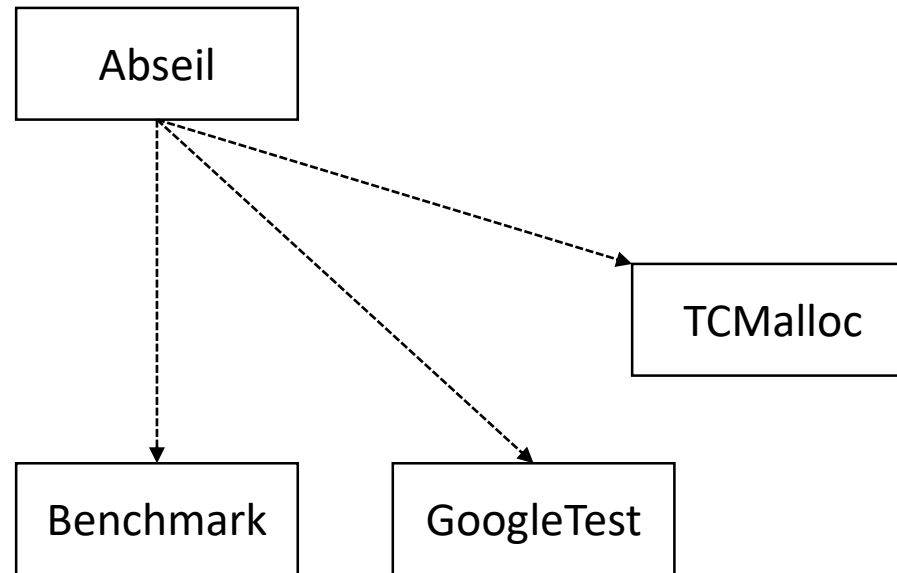


<https://www.youtube.com/watch?v=M2fKMP47sIQ>

C++ библиотеки: Идеи



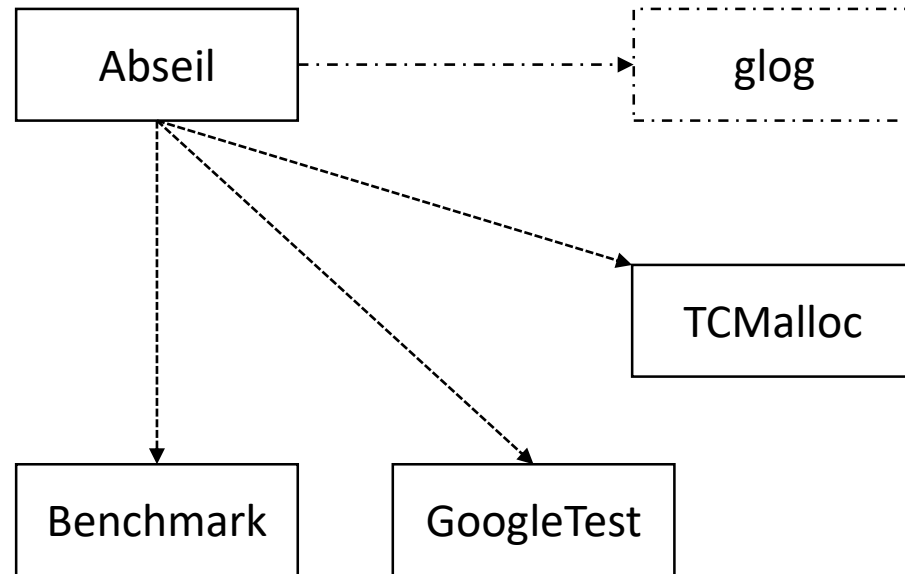
- Модульность



С++ библиотеки: Идеи



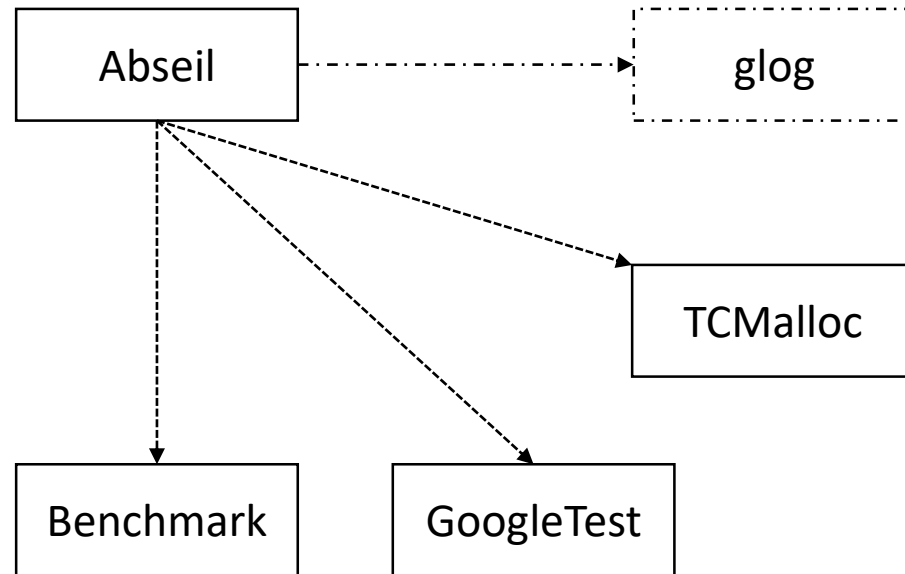
- Модульность



C++ библиотеки: Идеи



- Модульность

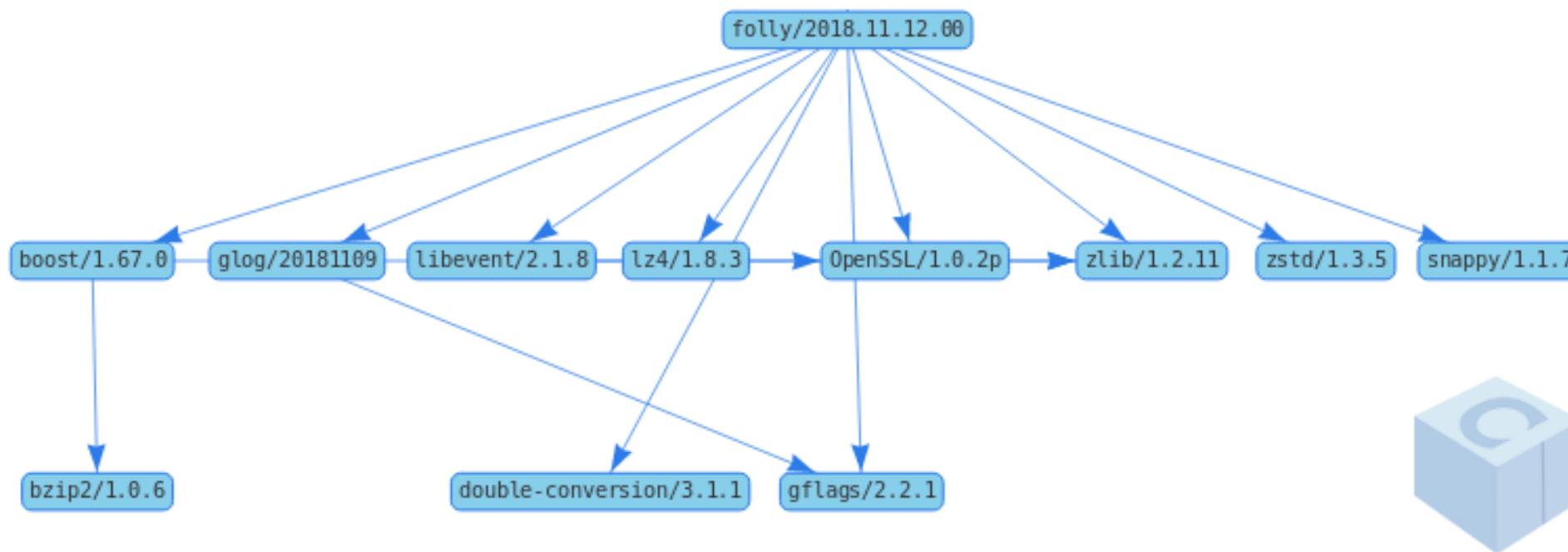


- Core lib
 - Allocator ;
 - Logging ;
 - Container ;
 - Util ;
- Tiny lib
 - Has logging?!;

C++ библиотеки: Идеи



- Модульность



<https://blog.conan.io/2018/12/03/Using-Facebook-Folly-with-Conan.html>

C++ библиотеки: Идеи



- Модульность

The screenshot displays the Compiler Explorer interface. On the left, the C++ source code is shown, featuring a function `absl::flat_hash_set<int> get_mock_set()` that initializes a vector `s` with the values `{5, 1, 2, 8, -1}`, creates a `flat_hash_set` object `result`, and inserts the elements from `s` into `result`. The right pane shows the assembly output for `x86-64 gcc 12.1`, with lines 52 through 65 highlighted in red, corresponding to the function's execution. The assembly includes instructions for moving registers, loading memory, and calling the `absl::flat_hash_set` constructor and `insert_iterator` function.

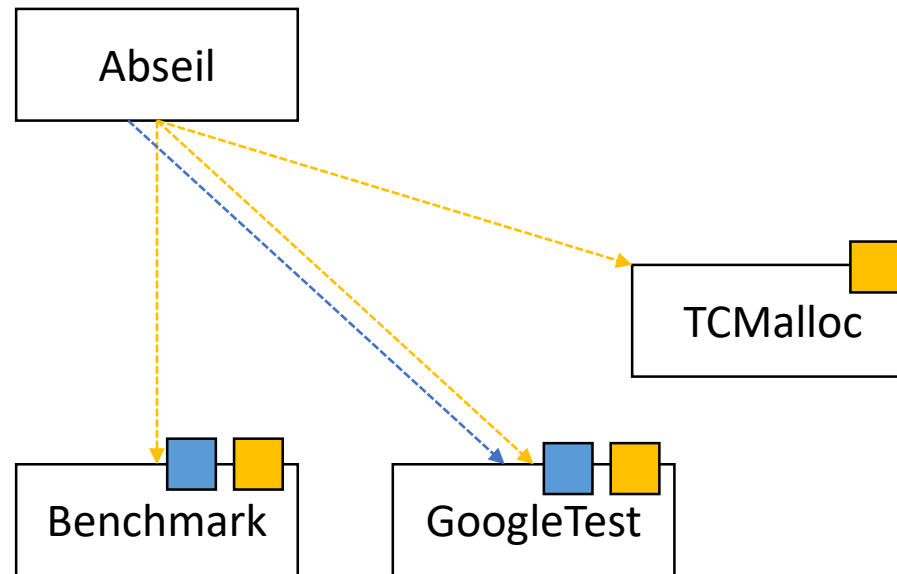
```
1 #include <absl/container/flat_hash_set.h>
2 #include <algorithm>
3 #include <iterator>
4 #include <vector>
5
6 absl::flat_hash_set<int> get_mock_set()
7 {
8     std::vector<int> s{ 5, 1, 2, 8, -1};
9     absl::flat_hash_set<int> result;
10    std::copy(s.begin(), s.end(), std::inserter(result, result.end()));
11    return result;
12 }
13
```

```
37 mov     rsi, r12
38 mov     rdi, r13
39 mov     rcx, r12
40 mov     rbx, r13
41 mov     rdi, rbx
42 lea    rax, [rbp-160]
43 mov     rcx, rdx
44 mov     rdx, rdi
45 mov     rdi, rax
46 call   std::vector<int, std::allocator<int> >::vector(std::initializer_
47 lea    rax, [rbp-97]
48 mov     rdi, rax
49 call   std::allocator<int>::~~allocator() [complete object destructor]
50 mov     rax, QWORD PTR [rbp-168]
51 mov     rdi, rax
52 call   absl::flat_hash_set<int, absl::hash_internal::Hash<int>, std::eq
53 mov     rax, QWORD PTR [rbp-168]
54 mov     rdi, rax
55 call   absl::container_internal::raw_hash_set<absl::container_internal:
56 lea    rdi, [rbp-64]
57 mov     rsi, QWORD PTR [rbp-168]
58 mov     rcx, rdx
59 mov     rdx, rax
60 call   std::insert_iterator<absl::flat_hash_set<int, absl::hash_interna
61 lea    rax, [rbp-160]
62 mov     rdi, rax
63 call   std::vector<int, std::allocator<int> >::end()
64 mov     rbx, rax
65 lea    rax, [rbp-160]
```

C++ библиотеки: Идеи



- Сборка из сорцов



■ CMake
■ Bazel

C++ библиотечки: Идеи



Morgan Stanley



Bjarne Stroustrup

C++20: Reaching for the Aims of C++

Video Sponsorship Provided By:



Modules – simplify library use

- We don't yet have modules for the standard library
 - Having 100+ standard headers is a barrier to entry
 - I'd like

```
import std;
int main()
{
    std::cout << "Hello modern world!\n";
}
```

- We can afford that
 - Small experiment
- The idea generalizes
 - Larger more logical modules
- Prefer named modules
 - Rather than header units

	#include needed headers	import needed headers	import std	#include all headers	import all headers
"Hello world" (<iostream>)	0.87s	0.32s	0.08s	3.43s	0.62s
"Mix" (9 popular headers)	2.20s	0.77s	0.44s	3.53s	0.99s

Stroustrup - CppCon 2021

40

C++ библиотеки: Идеи



- Менеджеры пакетов



HomeBrew



MacPorts

Core библиотеки: Идеи



Опрос: Как быстро вы начинаете использовать новинки в боевом коде?


- Еще до попадания в стандарт ;
- Как только попало в стандарт ;
- Как только в следующем стандарте пофиксят баги ;



C++ библиотеки: Идеи



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



TITUS WINTERS

What is Abseil

- Zero config
- Utility code
- string routines
- Debugging / analysis facilities
- Guidance (Tip of the Week)
- C++11-compatible versions of standard types (pre-adopt)
- Standards-alternatives

General goal: Support 5 years back where possible.

**C++ as a
"Live at Head"
Language**

CppCon.org

<https://www.youtube.com/watch?v=tISy7EJQPzI>

Core библиотеки: Идеи



- Live at Head Principle

Long-Term Support (LTS) Branches

By [Tom Manshreck](#), Abseil Tech Writer

Abseil encourages developers to “live at head” but we understand that philosophy may not work for everyone. We are therefore providing snapshots of the Abseil codebase. These snapshots are available as “Long Term Support” (LTS) branches of Abseil, and **we intend to provide a new snapshot every 6 months or so.**

We pledge to support these LTS snapshots **for at least 2 years**. If critical bug fixes, such as security issues, require us to change Abseil, we will also change them within any supported LTS snapshot.

NOTE: we don’t want you to think of these snapshots as “versions.” They are simply a snapshot of the codebase at a specific point in time. If you cannot build from source or otherwise live at head, prefer to use the latest LTS branch of Abseil instead.

<https://abseil.io/blog/20180618-lts-branches>

C++ библиотеки: Идеи



- Live at Head Principle

Releases Tags

03 Nov 2021
derekmauro
20211102.0
2151058

Compare ▾

Abseil LTS branch, Nov 2021 Latest

Abseil LTS 20211102

What's New:

- `absl::Cord` is now implemented as a b-tree. The new implementation offers improved performance in most workloads.
- `absl::SimpleHexAtoi()` has been added to `strings` library for parsing hexadecimal strings.

Breaking Changes:

- Bazel builds now depend on the [bazelbuild/platforms](#) repository. See Abseil's [WORKSPACE](#) file for an example of how to add this dependency.

Baseline: [2151058](#)

<https://github.com/abseil/abseil-cpp/releases>

Core библиотеки: Идеи



- Live at Head Principle

Releases Tags

7 hours ago
github-actions
v2022.05.30.00
a7f0878
Compare

v2022.05.30.00 Latest

Automated release from TagIt

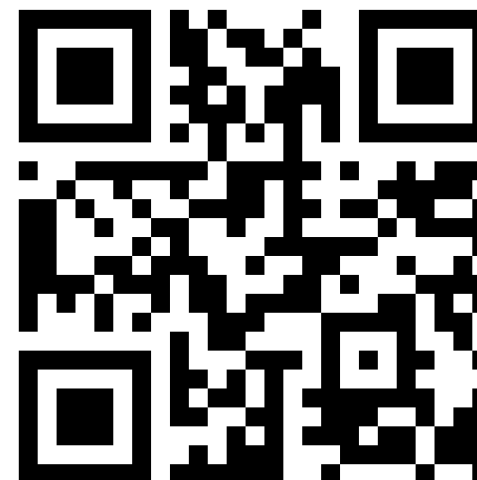
▶ File Hashes

▼ Assets 4

folly-v2022.05.30.00.tar.gz	3.47 MB	7 hours ago
folly-v2022.05.30.00.zip	4.84 MB	7 hours ago
Source code (zip)		3 days ago
Source code (tar.gz)		3 days ago

<https://github.com/facebook/folly/releases>

А что с опросом?



Code библиотеки: Идеи

- Live at Head Principle

The screenshot shows a search interface for a code library. At the top, there is a search bar with the text "This project" and "vector_t" entered, followed by a magnifying glass icon. Below the search bar, there are tabs for "Code" (11K+), "Work item" (11), and "Wiki" (0). To the right of these tabs is a link "Search this organization". Below the tabs, there are three filter boxes: "Repo: All" with a dropdown arrow, "Branch:" with a question mark and a dropdown arrow, and "Path:" with a question mark and a dropdown arrow. At the bottom left, a red-bordered box contains the text "Showing 25 of 11270 code results". At the bottom right, the text "vector.h" is displayed.

C++ библиотеки: Идеи

- Live at Head Principle

```
int http_get(std::string const& uri);
```



```
int http_get(std::string const& uri, std::vector<std::uint8_t>& data);
```

C++ библиотеки: Идеи

- Live at Head Principle

```
[[deprecated]]  
int http_get(std::string const& uri);  
  
// deprecated  
int http_get(std::string const& uri);
```

(C++14)



```
int http_get(std::string const& uri, std::vector<std::uint8_t>& data);
```

C++ библиотеки: Идеи



- Live at Head Principle

Clang Tidy

```
// Original - Comparison in the integer domain  
int x;  
absl::Time t;  
if (x < absl::ToUnixSeconds(t)) ...
```

```
// Suggested - Compare in the absl::Time domain instead  
if (absl::FromUnixSeconds(x) < t) ...
```

C++ библиотеки: Идеи

- Live at Head Principle

Clang Tidy

```
// Original - Comparison in the integer domain
int x;
absl::Time t;
if (x < absl::ToUnixSeconds(t)) ...
```

```
// Suggested - Compare in the absl::Time domain
instead
if (absl::FromUnixSeconds(x) < t) ...
```

<https://clang.llvm.org/extra/clang-tidy/>

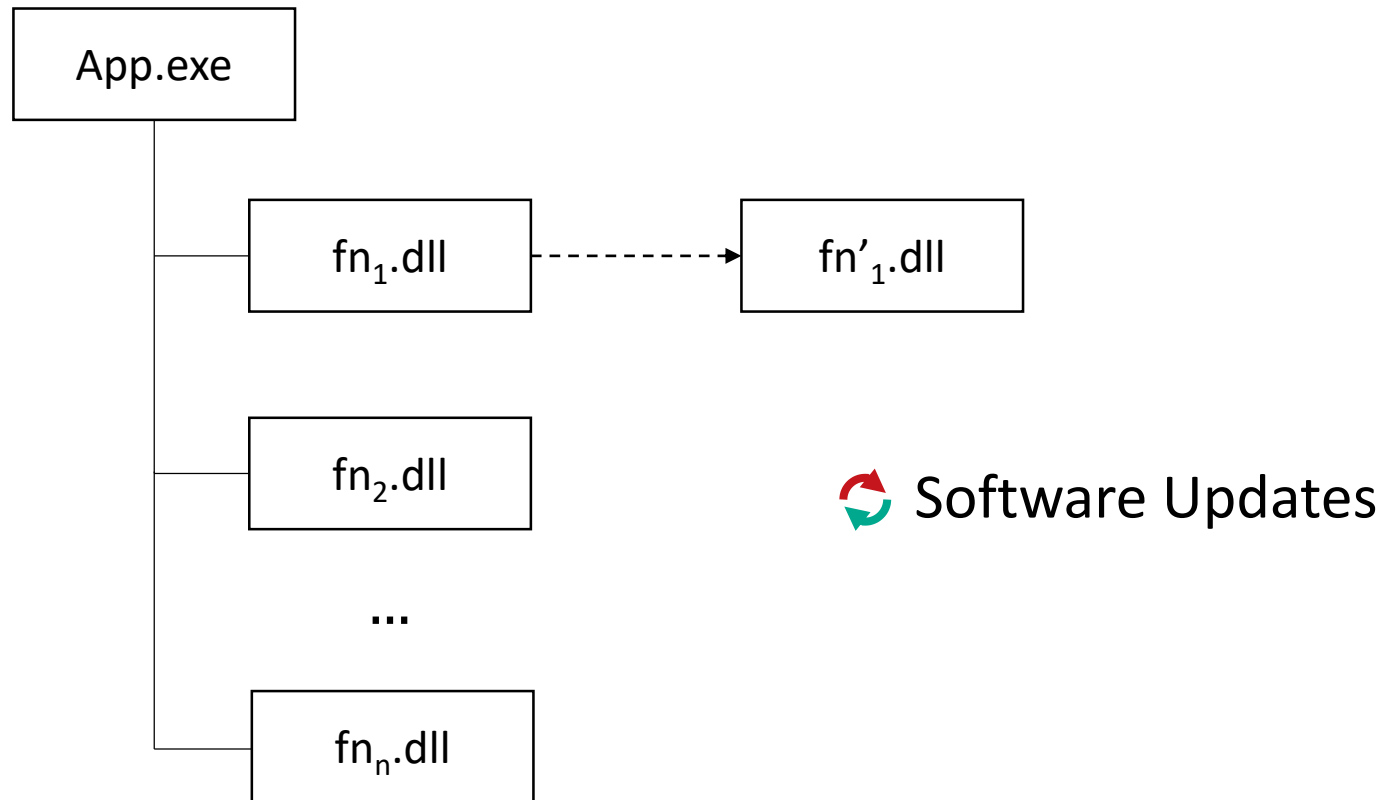


Clang-Tidy Checks

Name	Offers fixes
abseil-cleanup-ctad	Yes
abseil-duration-addition	Yes
abseil-duration-comparison	Yes
abseil-duration-conversion-cast	Yes
abseil-duration-division	Yes
abseil-duration-factory-float	Yes
abseil-duration-factory-scale	Yes
abseil-duration-subtraction	Yes
abseil-duration-unnecessary-conversion	Yes
abseil-faster-strsplit-delimiter	Yes
abseil-no-internal-dependencies	
abseil-no-namespace	
abseil-redundant-strcat-calls	Yes
abseil-str-cat-append	Yes
abseil-string-find-startswith	Yes
abseil-string-find-str-contains	Yes
abseil-time-comparison	Yes
abseil-time-subtraction	Yes
abseil-upgrade-duration-conversions	Yes
altera-id-dependent-backward-branch	
altera-kernel-name-restriction	

Core библиотеки: Идеи

- Цели создания: Boost vs Abseil vs ?



Core библиотеки: Источник вдохновения



...



Спасибо





Core библиотеки: Ещё

folly/fbvector

"...The initial HP implementation by Stepanov used a growth factor of 2; i.e., whenever you'd push_back into a vector without there being room, it would double the current capacity. This was not a good choice: it can be mathematically proven that a growth factor of 2 is *rigorously the worst possible* because it never allows the vector to reuse any of its previously other compilers *reducing the growth factor to 1.5*, maintained its factor of 2. This makes std::vector cac manager unfriendly..." @ github folly

folly::fbVector – улучшенный std::vector от Facebook
(<https://habr.com/ru/company/infopulse/blog/238131/>)

```
1  #include <iostream>
2  #include <iterator>
3  #include <algorithm>
4
5  #include <folly/FBVector.h>
6
7  int main(int argc, char** argv)
8  {
9      folly::fbvector<int> vec;
10     for (int ix = 1; ix <= 4096; ++ix)
11     {
12         vec.emplace_back(ix); // hello, jemalloc
13     }
14     copy(vec.begin(), vec.end(), std::ostream_iterator<int>(std::cout, " "));
15
16     return 0;
17 }
```



Core библиотеки: Ещё

folly/Thread pools & Executors ?

How do I use the thread pools?

Wangle provides two concrete thread pools (IOThreadPoolExecutor, CPUThreadPoolExecutor) as well as building them in as part of a complete async framework...or maybe you need to construct a thrift/memcache client, and need an event base:

```
auto f = getClient(getIOExecutor()->getEventBase()->callSomeFunction(args...))
    .via(getCPUExecutor())
    .then([](Result r) { .... do something with result });
```



C++ библиотеки: Ещё

folly/Thread pools & Executors ?

How do I use the thread pools?

Wangle provides two concrete (CPUThreadPoolExecutor) as well as a framework...or maybe you need a base:

```
auto f = getClient(getClient  
    .via(getCPUExecutor)  
    .then([](Result r)
```

Cppcon | Working with Asynchrony Generically:
A Tour of C++ Executors (part 1 of 2)

Eric Niebler

**Working with Asynchrony
Generically:
A Tour of C++ Executors**

ERIC NIEBLER

2021 | October 24-29

Cppcon | The C++ Conference

2021 | October 24-29

<https://www.youtube.com/watch?v=xLboNif7BTg>