

C++ Club UK

Gleb Dolgich

2019-02-14

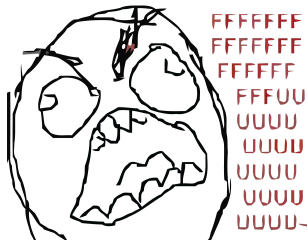
Avoid unsigned

- ▶ P0330R4 Literal Suffixes for ptrdiff_t and size_t
- ▶ P1227R1 Signed ssize() functions, unsigned size() functions
- ▶ P1428R0 Subscripts and sizes should be signed

```
1 template <typename C>
2 constexpr int ssize(const C& c)
3 {
4     const size_t size = c.size();
5     assert(size <= static_cast<size_t>(std::numeric_limits<int>::max()));
6     return static_cast<int>(size);
7 }
```

https://youtu.be/_CaP_xwfAFU

- ▶ Initialisation is broken!
- ▶ Initialiser lists are broken!
- ▶ Auto initialisation is broken!



https://youtu.be/s5PCh_FaMfM

What are we trying to be "safe" from?


Data races?

"The execution of a program contains a *data race* if it contains two potentially concurrent conflicting actions, at least one of which is not atomic...

Two expression evaluations *conflict* if one of them modifies a memory location (4.4) and the other one reads or modifies the same memory location."

— [The C++ Standard](#)

cppcon | 2018
THE C++ CONFERENCE - BELLEVUE, WASHINGTON



GEOFF ROMER

What do you mean
"thread-safe"?

CppCon.org


https://youtu.be/s5PCh_FaMfM

What are we trying to be "safe" from?

Data races?

```
std::string s = "";
```

<pre>void thread1() { ... s.append("foo"); ... }</pre>	<pre>void thread2() { ... std::cout << s; ... }</pre>
--	---

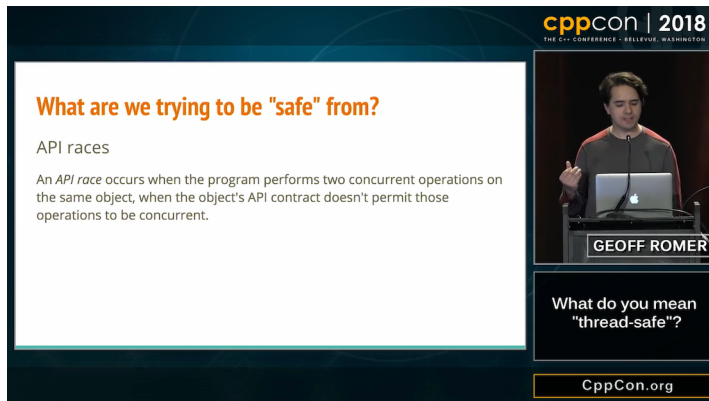


GEOFF ROMER

What do you mean "thread-safe"?

CppCon.org

https://youtu.be/s5PCh_FaMfM



The image is a screenshot of a video player showing a presentation from CppCon 2018. The presentation slide on the left has a dark blue header with the CppCon 2018 logo and text. The main content area of the slide is white and contains the following text:

What are we trying to be "safe" from?

API races

An *API race* occurs when the program performs two concurrent operations on the same object, when the object's API contract doesn't permit those operations to be concurrent.

On the right side of the video player, there is a live video feed of the speaker, Geoff Romer, standing at a podium with a laptop. Below the video feed, there is a name tag that reads "GEOFF ROMER". At the bottom of the video player, there is a dark blue bar with the text "What do you mean 'thread-safe'?" and a yellow bar with the text "CppCon.org".

cppcon | 2018
THE C++ CONFERENCE - BELLEVUE, WASHINGTON

What are we trying to be "safe" from?

API races

An *API race* occurs when the program performs two concurrent operations on the same object, when the object's API contract doesn't permit those operations to be concurrent.

GEOFF ROMER

What do you mean "thread-safe"?

CppCon.org


CppCon 2018 - Geoff Romer - What do you mean "thread-safe"?

https://youtu.be/s5PCh_FaMfM

Identifying API races

```
class LazyStringView {  
    const char* data_;  
    mutable optional<size_t> size_;  
    mutable std::mutex mu_;  
  
public:  
    size_t size() const {  
        std::scoped_lock lock(mu_);  
        if (!size_)  
            size_ = strlen(data_);  
        return *size_;  
    }  
};
```

cppcon | 2018
THE C++ CONFERENCE - BELLEVUE, WASHINGTON



GEOFF ROMER

What do you mean
"thread-safe"?


CppCon.org

https://youtu.be/s5PCh_FaMfM

Identifying API races

```
struct Counter {  
    int c = 0;  
    void operator()() { ++c; }  
};  
const std::function<void()> f = Counter{};  
  
void thread1() {  
    f();  
}  
  
void thread2() {  
    f();  
}
```

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON



GEOFF ROMER

What do you mean
"thread-safe"?

CppCon.org

https://youtu.be/s5PCh_FaMfM

Identifying API races


If a live object has a **thread-safe type**, it can't be the site of an API race.

If a live object has a **thread-compatible type**, it can't be the site of an API race if it's not being mutated.

If a live object has **any type**, it can't be the site of an API race if it's not being accessed concurrently.

... but beware of **thread-hostile functions**, which can cause API races at sites other than their inputs.

cppcon | 2018
THE C++ CONFERENCE - BELLEVUE, WASHINGTON

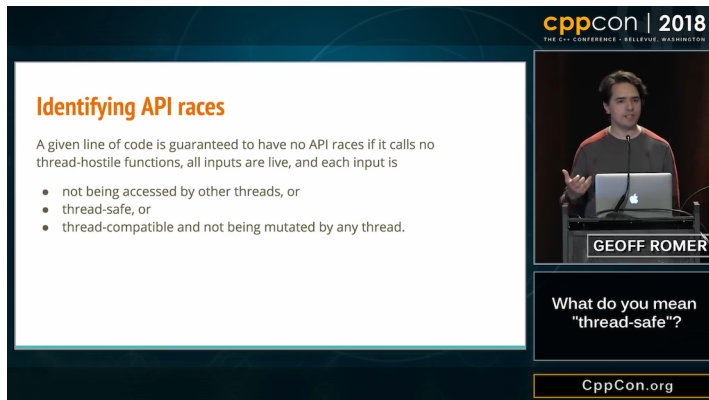


GEOFF ROMER

What do you mean
"thread-safe"?

CppCon.org

https://youtu.be/s5PCh_FaMfM



The image is a screenshot of a video player showing a presentation from CppCon 2018. The presentation slide on the left has a dark blue header with the CppCon 2018 logo and text. The main content area is white with the title "Identifying API races" in orange. Below the title is a paragraph explaining that a line of code is guaranteed to have no API races if it calls no thread-hostile functions, all inputs are live, and each input is safe. A bulleted list follows, listing three conditions: not being accessed by other threads, being thread-safe, or being thread-compatible and not being mutated by any thread. On the right side of the video player, there is a live video feed of Geoff Romer speaking at a podium. Below the video feed, the text "What do you mean 'thread-safe'?" is displayed. At the bottom right of the video player, the website "CppCon.org" is shown.

cppcon | 2018
THE C++ CONFERENCE - BELLEVUE, WASHINGTON

Identifying API races

A given line of code is guaranteed to have no API races if it calls no thread-hostile functions, all inputs are live, and each input is

- not being accessed by other threads, or
- thread-safe, or
- thread-compatible and not being mutated by any thread.

GEOFF ROMER

What do you mean "thread-safe"?

CppCon.org

https://youtu.be/s5PCh_FaMfM


Identifying API races

```
vector<int> shared_vec = {0, 0};

void thread1() {
    ...
    ++shared_vec[0];
    ...
}

void thread2() {
    ...
    ++shared_vec[1];
    ...
}
```

cppcon | 2018
THE C++ CONFERENCE • BELLEVUE, WASHINGTON



GEOFF ROMER

What do you mean
"thread-safe"?

CppCon.org

https://youtu.be/s5PCh_FaMfM


Identifying API races

```
vector<bool> shared_vec = {false, false};

void thread1() {
    ...
    shared_vec[0] = true;
    ...
}

void thread2() {
    ...
    shared_vec[1] = true;
    ...
}
```

cppcon | 2018
THE C++ CONFERENCE - BELLEVUE, WASHINGTON



GEOFF ROMER

What do you mean
"thread-safe"?

CppCon.org

https://youtu.be/s5PCh_FaMfM

Recommendations

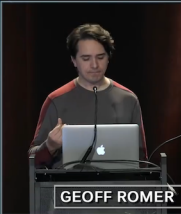
For library code:

- Make your types thread-compatible if possible, thread-safe if necessary.
- Clearly document any types that are thread-safe, or thread-incompatible.
 - Prefer to explicitly document the rest as thread-compatible.
- Be thoughtful about directly exposing sub-objects.
- Never define or use thread-hostile functions.
 - Avoid hidden mutable shared state
 - Be very careful with private pointers to shared data.

For application code:

- Make shared objects thread-safe, or thread-compatible and immutable.

cppcon | 2018
THE C++ CONFERENCE - BELLEVUE, WASHINGTON



GEOFF ROMER

What do you mean
"thread-safe"?

CppCon.org

Proper way to iterate backwards in C++

https://www.reddit.com/r/cpp/comments/947a1z/proper_way_to_do_backward_iteration_in_c/

```
1 for (size_t i = data.size() - 1; i >= 0; --i) { ... } // Nope
2 for (size_t i = data.size(); i--;) { ... } // The C way
3
4 // C++17
5 std::vector<int> vec;
6 for (auto [value, idx] : reverse_index_adapter(vec)) {
7     // idx = n-1, n-2, ... 0
8 }
9
10 std::for_each(vec.rbegin(), vec.rend(), []( ) { ... }); // No index
11
12 for (auto it = data.rbegin(); it != data.rend(); ++it) {
13     auto i = std::distance(it, data.rend()) - 1;
14 }
```

https://youtu.be/S7l66lZX_zM

Inverse two-phase initialisation

```
1 class Foo
2 {
3     static expected<construction_token>
4     preconstruct(Arg n_arg) noexcept
5     {
6         construction_token t;
7         t.state = make_unique_nothrow(n_arg);
8         if (!t.state) return unexpected(my_errc::error);
9         return t;
10    }
11
12    Foo(construction_token&& t) noexcept
13    : m_state(std::move(t.state)) {}
14 };
```

https://youtu.be/S7l66lZX_zM

Inverse two-phase initialisation: usage

```
1 // 1
2 expected<Foo::construction_token> t1 = Foo::preconstruct(args);
3 if (!t1.has_value()) { /* get out */ }
4 Foo obj(std::move(*t1));
5
6 // 2
7 auto t2 = Foo::preconstruct(args);
8 auto obj_ptr = std::make_shared<Foo>(std::move(*t2));
9
10 // 3
11 auto t3 = Foo::preconstruct(args);
12 std::vector<Foo> objects;
13 objects.emplace_back(std::move(*t3));
```


CppCon 2018 - Timur Doumler - I can has grammar?

<https://youtu.be/tsG95Y-C14k>

```
1 // MSVC rejects, GCC accepts, Clang accepts
2 extern extern "C++" extern "C" extern "C++" int x;
3
4 // MSVC accepts, GCC rejects, Clang accepts
5 extern "C++" extern "C" extern "C++" extern int x;
```

<https://youtu.be/tsG95Y-C14k>

selection-statement:

`if constexpropt (init-statementopt condition) statement`

```
1 if (class foo; !ret.second) /* ... */;  
2  
3 if (false; true) /* ... */;  
4  
5 if (; true) /* ... */;
```

<https://youtu.be/tsG95Y-C14k>

Declare and initialise a variable of type "function pointer":

```
1 | auto (*fp)() -> int(&f);
```

<https://youtu.be/tsG95Y-C14k>

```
1 struct foo;  
2 void bar(foo foo);  
3 void bar(foo(foo)); // vexing parse  
4 void bar(foo((foo))); // more vexing parse
```

<https://youtu.be/tsG95Y-C14k>

```
1 class bar {};  
2 int bar;    // OK  
3 bar b;      // error  
4 class bar b; // OK  
5 class std::vector<class bar> bars; // OK  
6  
7 // also acts as a forward declaration  
8 void foo(struct S* x);  
9  
10 // weird scoping rules  
11 class C { void foo(struct S* x); };  
12 S* s;
```

<https://youtu.be/tsG95Y-C14k>

pseudo-destructor-name

```
1 int i;  
2 i.~int(); // error: int is not type-name but type-specifier  
3 using foo = int;  
4 i.~foo(); // OK  
5 ~int(); // OK (not a destructor!)
```

<https://youtu.be/tsG95Y-C14k>

Alternative tokens

```
1 struct Foo
2 {
3     Foo();
4     compl Foo();
5     Foo(const Foo bitand);
6     Foo(Foo and);
7 };
```

CppCon 2018: Matt Godbolt “The Bits Between the Bits: How We Get to main()”

<https://youtu.be/dOfucXtyEsU>

<https://arne-mertz.de/2019/02/extern-template-reduce-compile-times/>

Move smart pointers in and out functions in modern C++

- ▶ Move smart pointers in and out functions in modern C++
 - ▶ Reddit: https://www.reddit.com/r/cpp/comments/aaux96/move_smart_pointers_in_and_out_functions_in/



Lynn
@chordbug

🌶️ HOT JAVASCRIPT TIP: 🌶️

to increment some counter on the page,

```
node.innerText += 1
```

doesn't work ($0 \rightarrow 01 \rightarrow 011 \rightarrow \dots$), but

```
node.innerText -= -1
```

works fine ($0 \rightarrow 1 \rightarrow 2 \rightarrow \dots$)

4,127 Likes

1,490 Retweets

5 Feb 2019 at 16:36

via **Twitter Web Client**