[Не]очевидные оптимизации и паттерны из userver

Антон Полухин

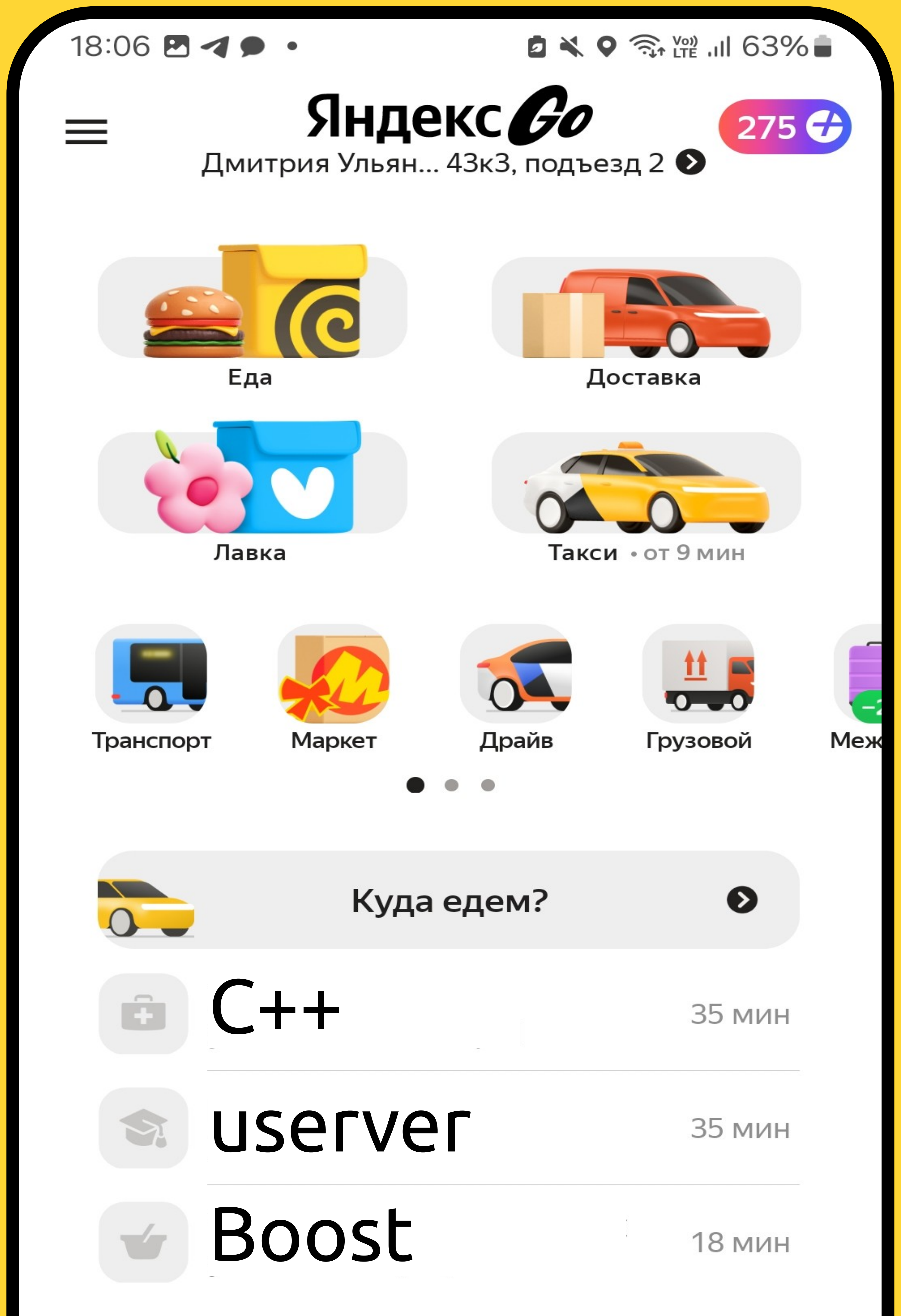
Эксперт разработчик C++

Техплатформа городских сервисов Яндекса

https://userver.tech/

Содержание

- Самый быстрый shared_ptr
- Экономим память не экономя
- Альтернатива 1000 перегрузок
- Правильная балансировка
- return v == std::vector{"*"};



Самый быстрый shared_ptr

Итак, проблема...

Итак, проблема...

```
Thread 1                Thread 2                Thread 3
```

Итак, проблема...

```
Thread 1              Thread 2                Thread 3

| ptr1 = g_ptr
|
|
|
|
|
|
|
|
```

Итак, проблема...

```
Thread 1               Thread 2                 Thread 3

| ptr1 = g_ptr

|

|                      | g_ptr = ptr2

|                      |

|

|

|

|

|
```

Итак, проблема...

```
Thread 1              Thread 2                Thread 3

| ptr1 = g_ptr
|
|                     | g_ptr = ptr2          |  ptr12 = g_ptr
|                     |
|
|
|
|
|
```

Итак, проблема...

```
Thread 1              Thread 2                Thread 3

| ptr1 = g_ptr
|
|                     | g_ptr = ptr2          |  ptr12 = g_ptr
|                     |
|
|                     | ptr2 = g_ptr
|                     |
|
|
```

Итак, проблема...

```
Thread 1              Thread 2                Thread 3

| ptr1 = g_ptr
|
|                     | g_ptr = ptr2          |  ptr12 = g_ptr
|                     |
|
|                     | ptr2 = g_ptr
|                     |
|
|

                      ...
                      | g_ptr = ptr1
```

Решение «в лоб»

Решение «в лоб»

```
namespace {
constinit std::mutex g_mutex{};
constinit std::shared_ptr<Logger> g_ptr = DefaultLogger();
}
```

Решение «в лоб»

```
namespace {
constinit std::mutex g_mutex{};
constinit std::shared_ptr<Logger> g_ptr = DefaultLogger();
}

void SetLogger(std::shared_ptr<Logger> ptr) {
    std::lock_guard _{g_mutex};
    g_ptr.swap(ptr);
}
```

Решение «в лоб»

```
namespace {
constinit std::mutex g_mutex{};
constinit std::shared_ptr<Logger> g_ptr = DefaultLogger();
}

void SetLogger(std::shared_ptr<Logger> ptr) {
    std::lock_guard _{g_mutex};
    g_ptr.swap(ptr);
}

std::shared_ptr<Logger> GetLogger() {
    std::lock_guard _{g_mutex};
    return g_ptr;
}
```

Решение «в лоб»

```
namespace {
constinit std::mutex g_mutex{};
constinit std::shared_ptr<Logger> g_ptr = DefaultLogger();
}

void SetLogger(std::shared_ptr<Logger> ptr) {
    std::lock_guard _{g_mutex};
    g_ptr.swap(ptr);
}

std::shared_ptr<Logger> GetLogger() {
    std::lock_guard _{g_mutex}; // Ой! Оёёй!!!
    return g_ptr;
}
```

Решение «в лоб»

```
namespace {
constinit std::mutex g_mutex{};
constinit std::shared_ptr<Logger> g_ptr = DefaultLogger();
}

void SetLogger(std::shared_ptr<Logger> ptr) {
    std::lock_guard _{g_mutex};
    g_ptr.swap(ptr);
}

std::shared_ptr<Logger> GetLogger() {
    std::lock_guard _{g_mutex}; // Ой! Оёёй!!!
    return g_ptr;               // Ой!
}
```

Решение C++20

Решение C++20

```
namespace {
constinit std::atomic<std::shared_ptr<Logger>> g_ptr = // ...
}
```

Решение C++20

```
namespace {
constinit std::atomic<std::shared_ptr<Logger>> g_ptr = // ...
}


void SetLogger(std::shared_ptr<Logger> ptr) {
    g_ptr = std::move(ptr);
}

std::shared_ptr<Logger> GetLogger() {
    return g_ptr.load();
}
```

Решение C++20

```
namespace {
constinit std::atomic<std::shared_ptr<Logger>> g_ptr = // ...
}


void SetLogger(std::shared_ptr<Logger> ptr) {
    g_ptr = std::move(ptr);
}

std::shared_ptr<Logger> GetLogger() {
    return g_ptr.load();       // Ой! Оёёй!!!
}
```

Решение C++26

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {

    auto* old = g_ptr.exchange(ptr.release());

    old->retire();
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {

    auto* old = g_ptr.exchange(ptr.release());

    old->retire();
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();

void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {

    auto* old = g_ptr.exchange(ptr.release());

    old->retire();
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}


auto GetLogger() {
    std::hazard_pointer h = std::make_hazard_pointer();
    auto* p = h.protect(g_ptr);
    return {h, p};
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();

void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}

auto GetLogger() {
    std::hazard_pointer h = std::make_hazard_pointer();
    auto* p = h.protect(g_ptr);
    return {h, p};
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}


auto GetLogger() {
    std::hazard_pointer h = std::make_hazard_pointer();
    auto* p = h.protect(g_ptr);
    return {h, p};
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();

void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}

auto GetLogger() {
    std::hazard_pointer h = std::make_hazard_pointer();
    auto* p = h.protect(g_ptr);
    return {h, p};
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}


auto GetLogger() {
    std::hazard_pointer h = std::make_hazard_pointer();
    auto* p = h.protect(g_ptr);
    return {h, p};
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}


auto GetLogger() {
    std::hazard_pointer h = std::make_hazard_pointer();
    auto* p = h.protect(g_ptr);
    return {h, p};
}
```

Решение C++26

```
constinit std::atomic<Logger*> g_ptr = DefaultLoggerPtr();


void SetLogger(std::unique_ptr<Logger> ptr) {
    auto* old = g_ptr.exchange(ptr.release());
    old->retire();
}


auto GetLogger() {
    std::hazard_pointer h = std::make_hazard_pointer();
    auto* p = h.protect(g_ptr);
    return {h, p};
}
```

Проблема, ещё раз...

Проблема, ещё раз...

```
Thread 1                Thread 2                 Thread 3

| ptr1 = g_ptr
|
|                       | g_ptr = ptr2           |  ptr12 = g_ptr
|                       |
|
|                       | ptr2 = g_ptr
|                       |
|
|

                        ...
                        | g_ptr = ptr1
```

Проблема, ещё раз...

```
logger1                 logger2             logger3
|
|                       |
|                       |                   |
|                        ...
|                       |                   |
|                       |                   ?
|                       ?                   ?
|                       ?                   ?
|
|
```

Проблема, ещё раз...

```
   logger1                logger2             logger3
   |
   |                      |
   |                      |                   |
   |                       ...
   |                      |                   |
   |                      |                   ?
   |                      ?                   ?
   |                      ?                   ?
-------------------------------------------------------------------------------------
   |
   |
```

Проблема, ещё раз…

```
logger1                 logger2             logger3
|
|                       |
|                       |                   |
|                        ...
|                       |                   |
|                       |                   ?
|                       ?                   ?
|                       ?                   ?
|
|
```

Проблема, ещё раз...

```
logger1                logger2               logger3
|
|                      |
|                      |                     |
|                       ...
|                      |                     |
|                      |                     |
|                      |                     |
|                      |                     |
|
|
```

Решение [не]очевидное

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

void SetLogger(std::shared_ptr<Logger> ptr) {
    static std::mutex g_mutex{};
    static std::vector<std::shared_ptr<Logger>> g_all_loggers{};

    UASSERT(ptr);
    g_ptr = ptr.get();

    std::lock_guard _{g_mutex};
    g_all_loggers.push_back(std::move(ptr));
}
```

Решение [не]очевидное

```cpp
constinit std::atomic<Logger*> g_ptr{nullptr};

void SetLogger(std::shared_ptr<Logger> ptr) {
    static std::mutex g_mutex{};
    static std::vector<std::shared_ptr<Logger>> g_all_loggers{};    // не production решение
    UASSERT(ptr);
    g_ptr = ptr.get();

    std::lock_guard _{g_mutex};
    g_all_loggers.push_back(std::move(ptr));
}
```

Решение [не]очевидное

```cpp
constinit std::atomic<Logger*> g_ptr{nullptr};

void SetLogger(std::shared_ptr<Logger> ptr) {
    static std::mutex g_mutex{};
    static std::vector<std::shared_ptr<Logger>> g_all_loggers{};

    UASSERT(ptr);
    g_ptr = ptr.get();

    std::lock_guard _{g_mutex};

    g_all_loggers.push_back(std::move(ptr));
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

void SetLogger(std::shared_ptr<Logger> ptr) {
    static std::mutex g_mutex{};
    static std::vector<std::shared_ptr<Logger>> g_all_loggers{};

    UASSERT(ptr);
    g_ptr = ptr.get();

    std::lock_guard _{g_mutex};
    g_all_loggers.push_back(std::move(ptr));
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

void SetLogger(std::shared_ptr<Logger> ptr) {
    static std::mutex g_mutex{};
    static std::vector<std::shared_ptr<Logger>> g_all_loggers{};

    UASSERT(ptr);
    g_ptr = ptr.get();

    std::lock_guard _{g_mutex};
    g_all_loggers.push_back(std::move(ptr));
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

void SetLogger(std::shared_ptr<Logger> ptr) {
    static std::mutex g_mutex{};
    static std::vector<std::shared_ptr<Logger>> g_all_loggers{};

    UASSERT(ptr);
    g_ptr = ptr.get();

    std::lock_guard _{g_mutex};
    g_all_loggers.push_back(std::move(ptr));
}
```

Решение [не]очевидное

```cpp
constinit std::atomic<Logger*> g_ptr{nullptr};

void SetLogger(std::shared_ptr<Logger> ptr) {
    static std::mutex g_mutex{};
    static std::vector<std::shared_ptr<Logger>> g_all_loggers{};

    UASSERT(ptr);
    g_ptr = ptr.get();

    std::lock_guard _{g_mutex};
    g_all_loggers.push_back(std::move(ptr));
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();
    UASSERT(p);
    return *p;
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();
    UASSERT(p);
    return *p;
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();
    UASSERT(p);
    return *p;
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();
    UASSERT(p);
    return *p;
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();
    UASSERT(p);
    return *p;
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();  // mov rax, QWORD PTR g_ptr[rip]
    UASSERT(p);
    return *p;
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();  // mov rax, QWORD PTR g_ptr[rip]
    UASSERT(p);
    return *p;
}
```

Решение [не]очевидное

```
constinit std::atomic<Logger*> g_ptr{nullptr};

Logger& GetLogger() {
    auto* p = g_ptr.load();  // mov rax, QWORD PTR g_ptr[rip]
    UASSERT(p);
    return *p;
}
```

Экономим память не экономя

Новая проблема

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();


// ...
g_queue.push(ptr);
```

Новая проблема

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};

// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();

// ...
g_queue.push(ptr);
```

Новая проблема
Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();


// ...
g_queue.push(ptr);
```

Новая проблема
Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();


// ...
g_queue.push(ptr);
```

Новая проблема

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();


// ...
g_queue.push(ptr);
```

Новая проблема
Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();


// ...
g_queue.push(ptr);
```

Новая проблема

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();


// ...
g_queue.push(ptr);
```

Новая проблема
Пул корутин

Новая проблема

Пул корутин

01

Недостаточно корутин в пуле — тратится много CPU на выделения и разметку памяти

Новая проблема

Пул корутин

01

Недостаточно корутин в пуле — тратится много CPU на выделения и разметку памяти

02

Избыток корутин в пуле — тратится оперативная память

Новая проблема

Пул корутин

01

Недостаточно корутин в пуле — тратится много CPU на выделения и разметку памяти

02

Избыток корутин в пуле — тратится оперативная память

03

В зависимости от приложения нужны разные размеры пулов — требуется ручная настройка

Новая проблема

Пул корутин

01

Недостаточно корутин в пуле — тратится много CPU на выделения и разметку памяти

02

Избыток корутин в пуле — тратится оперативная память

03

В зависимости от приложения нужны разные размеры пулов — требуется ручная настройка

04

Ручная настройка пулов — боль и страдание

[Не]очевидное решение

Пул корутин

[Не]очевидное решение

Пул корутин

1 Выставляем большое число корутин на старте

[Не]очевидное решение

Пул корутин

1 Выставляем большое число корутин на старте

2 Размечаем кусок памяти под корутины

[Не]очевидное решение
Пул корутин

1. Выставляем большое число корутин на старте
2. Размечаем кусок памяти под корутины
3. ???????

[Не]очевидное решение

Пул корутин

1. Выставляем большое число корутин на старте
2. Размечаем кусок памяти под корутины
3. ???????
4. PROFIT

[Не]очевидное решение

Пул корутин

1. Выставляем большое число корутин на старте
2. Размечаем кусок памяти под корутины
3. Не обращаемся к нетронутой памяти
4. PROFIT

[Не]очевидное решение

Пул корутин

1. Выставляем большое число корутин на старте
2. Размечаем кусок памяти под корутины
3. **Не обращаемся к нетронутой памяти**
4. PROFIT

[Не]очевидное решение

Пул корутин

1. Выставляем большое число корутин на старте
2. Размечаем кусок памяти под корутины
3. Не обращаемся к нетронутой памяти
4. PROFIT

Было

Пул корутин

```cpp
ConcurrentFiloQueue<Coro*> g_queue{InitialCoroPoolSize};

// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : NewCoro();

// ...
g_queue.push(ptr);
```

[Не]очевидное решение
Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue_initial{InitialCoroPoolSize};
ConcurrentFiloQueue<Coro*> g_queue{};

// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : (ptr = g_queue_initial.try_extract() ? ptr : NewCoro());

// ...
g_queue.push(ptr);
```

[Не]очевидное решение

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue_initial{InitialCoroPoolSize};
ConcurrentFiloQueue<Coro*> g_queue{};

// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : (ptr = g_queue_initial.try_extract() ? ptr : NewCoro());

// ...
g_queue.push(ptr);
```

[Не]очевидное решение

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue_initial{InitialCoroPoolSize};
ConcurrentFiloQueue<Coro*> g_queue{};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : (ptr = g_queue_initial.try_extract() ? ptr : NewCoro());


// ...
g_queue.push(ptr);
```

[Не]очевидное решение
Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue_initial{InitialCoroPoolSize};
ConcurrentFiloQueue<Coro*> g_queue{};

// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : (ptr = g_queue_initial.try_extract() ? ptr : NewCoro());

// ...
g_queue.push(ptr);
```

[Не]очевидное решение

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue_initial{InitialCoroPoolSize};
ConcurrentFiloQueue<Coro*> g_queue{};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : (ptr = g_queue_initial.try_extract() ? ptr : NewCoro());


// ...
g_queue.push(ptr);
```

[Не]очевидное решение

Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue_initial{InitialCoroPoolSize};
ConcurrentFiloQueue<Coro*> g_queue{};

// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : (ptr = g_queue_initial.try_extract() ? ptr : NewCoro());

// ...
g_queue.push(ptr);
```

[Не]очевидное решение
Пул корутин

```
ConcurrentFiloQueue<Coro*> g_queue_initial{InitialCoroPoolSize};
ConcurrentFiloQueue<Coro*> g_queue{};


// ...
auto* ptr = g_queue.try_extract();
return ptr ? ptr : (ptr = g_queue_initial.try_extract() ? ptr : NewCoro());


// ...
g_queue.push(ptr);
```

Альтернатива 1000 перегрузок

Хотим сделать красиво

easy

```cpp
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    userver::easy::HttpWith<>(argc, argv)
        .DefaultContentType(userver::http::content_type::kTextPlain)
        .Route("/hello", [](const userver::server::http::HttpRequest& /*req*/) {
            return "Hello world";  // Возвращаем строчку как ответ на запрос
        });
}
```

Хотим сделать красиво

easy

```cpp
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    userver::easy::HttpWith<>(argc, argv)
        .DefaultContentType(userver::http::content_type::kTextPlain)
        .Route("/hello", [](const userver::server::http::HttpRequest& /*req*/) {
            return "Hello world";  // Возвращаем строчку как ответ на запрос
        });
}
```

Хотим сделать красиво
easy

```
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    userver::easy::HttpWith<>(argc, argv)
        .DefaultContentType(userver::http::content_type::kTextPlain)
        .Route("/hello", [](const userver::server::http::HttpRequest& /*req*/) {
            return "Hello world";  // Возвращаем строчку как ответ на запрос
        });
}
```

Хотим сделать красиво

easy

```cpp
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    userver::easy::HttpWith<>(argc, argv)
        .DefaultContentType(userver::http::content_type::kTextPlain)
        .Route("/hello", [](const userver::server::http::HttpRequest& /*req*/) {
            return "Hello world";  // Возвращаем строчку как ответ на запрос
        });
}
```

Хотим сделать красиво
easy

```cpp
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    userver::easy::HttpWith<>(argc, argv)
        .DefaultContentType(userver::http::content_type::kTextPlain)
        .Route("/hello", [](const userver::server::http::HttpRequest& /*req*/) {
            return "Hello world";  // Возвращаем строчку как ответ на запрос
        });
}
```

Хотим сделать красиво
Но есть сложности...

```
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    using namespace userver;
    easy::HttpWith<userver::easy::PgDep>(argc, argv)
        .DbSchema(kSchema)
        .Get("/kv",      [](formats::json::Value request_json) {/* ... */})
        .Post("/kv",     [](const formats::json::Value& request_json, easy::PgDep dep) { /*...*/ })
        .Route("/hello", [](const server::http::HttpRequest& req) { /*...*/ })
        .Del("/hi",      [](const server::http::HttpRequest& req, const easy::PgDep& dep) {/*...*/});
}
```

Хотим сделать красиво

Но есть сложности…

```
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    using namespace userver;
    easy::HttpWith<userver::easy::PgDep>(argc, argv)
        .DbSchema(kSchema)
        .Get("/kv",      [](formats::json::Value request_json) {/* ... */})
        .Post("/kv",     [](const formats::json::Value& request_json, easy::PgDep dep) { /*...*/ })
        .Route("/hello", [](const server::http::HttpRequest& req) { /*...*/ })
        .Del("/hi",      [](const server::http::HttpRequest& req, const easy::PgDep& dep) {/*...*/});
}
```

Хотим сделать красиво
Но есть сложности…

```cpp
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    using namespace userver;
    easy::HttpWith<userver::easy::PgDep>(argc, argv)
        .DbSchema(kSchema)
        .Get("/kv",      [](formats::json::Value request_json) {/* ... */})
        .Post("/kv",     [](const formats::json::Value& request_json, easy::PgDep dep) { /*...*/ })
        .Route("/hello", [](const server::http::HttpRequest& req) { /*...*/ })
        .Del("/hi",      [](const server::http::HttpRequest& req, const easy::PgDep& dep) {/*...*/});
}
```

Хотим сделать красиво

Но есть сложности…

```
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    using namespace userver;
    easy::HttpWith<userver::easy::PgDep>(argc, argv)
        .DbSchema(kSchema)
        .Get("/kv",      [](formats::json::Value request_json) {/* ... */})
        .Post("/kv",     [](const formats::json::Value& request_json, easy::PgDep dep) { /*...*/ })
        .Route("/hello", [](const server::http::HttpRequest& req) { /*...*/ })
        .Del("/hi",      [](const server::http::HttpRequest& req, const easy::PgDep& dep) {/*...*/});
}
```

Хотим сделать красиво

Но есть сложности...

```cpp
#include <userver/easy.hpp>


int main(int argc, char* argv[]) {
    using namespace userver;
    easy::HttpWith<userver::easy::PgDep>(argc, argv)
        .DbSchema(kSchema)
        .Get("/kv",      [](formats::json::Value request_json) {/* ... */})
        .Post("/kv",     [](const formats::json::Value& request_json, easy::PgDep dep) { /*...*/ })
        .Route("/hello", [](const server::http::HttpRequest& req) { /*...*/ })
        .Del("/hi",      [](const server::http::HttpRequest& req, const easy::PgDep& dep) {/*...*/});
}
```

Сложности

Сложности

1. 5+1 HTTP методов

Сложности

1. 5+1 HTTP методов
2. 6+ сигнатур функций

Сложности

1. 5+1 HTTP методов
2. 6+ сигнатур функций
3. Мы не любим копипасту

Сложности

1. 5+1 HTTP методов
2. 6+ сигнатур функций
3. Мы не любим копипасту
4. Нужно всё документировать

Сложности

1. 5+1 HTTP методов
2. 6+ сигнатур функций
3. Мы не любим копипасту
4. Нужно всё документировать
5. Нужно всё тестировать

Сложности

1. 5+1 HTTP методов
2. 6+ сигнатур функций
3. Мы не любим копипасту
4. Нужно всё документировать
5. Нужно всё тестировать
6. Нужна понятная диагностика

Хотим сделать красиво

Варианты

Хотим сделать красиво

Варианты

```
template <class Callback>
HttpWith& Get(std::string_view path, Callback&& func);
```

Хотим сделать красиво
Варианты

```
template <class Callback>
HttpWith& Get(std::string_view path, Callback&& func);



template <CallbackCompatibleFunction Callback>
HttpWith& Get(std::string_view path, Callback&& func);
```

Хотим сделать красиво
Варианты

```
template <class Callback>
HttpWith& Get(std::string_view path, Callback&& func);



template <CallbackCompatibleFunction Callback>
HttpWith& Get(std::string_view path, Callback&& func);



class Callback;
HttpWith& Get(std::string_view path, Callback&& func);
```

Хотим сделать красиво

[Не]очевидное решение

```
/// Helper class that can store any callback of the following signatures:
/// * formats::json::Value(formats::json::Value, const Dependency&)
/// ...
class Callback final {
public:
    template <class Function>
    Callback(Function func);
    HttpBase::Callback Extract() && noexcept { return std::move(func_); }

private:
    HttpBase::Callback func_;
};
```

Хотим сделать красиво

[Не]очевидное решение

```
/// Helper class that can store any callback of the following signatures:
/// * formats::json::Value(formats::json::Value, const Dependency&)
/// ...
class Callback final {
public:
    template <class Function>
    Callback(Function func);
    HttpBase::Callback Extract() && noexcept { return std::move(func_); }
private:
    HttpBase::Callback func_;
};
```

Хотим сделать красиво

[Не]очевидное решение

```
/// Helper class that can store any callback of the following signatures:
/// * formats::json::Value(formats::json::Value, const Dependency&)
/// ...
class Callback final {
public:
    template <class Function>
    Callback(Function func);
    HttpBase::Callback Extract() && noexcept { return std::move(func_); }

private:
    HttpBase::Callback func_;
};
```

Хотим сделать красиво
[Не]очевидное решение

```
template <class Dependency>
template <class Function>
HttpWith<Dependency>::Callback::Callback(Function func) {
    namespace json = formats::json;
    constexpr unsigned kMatches =

        (std::is_invocable_r_v<json::Value, Function, json::Value, const Dependency&> << 0) |
        (std::is_invocable_r_v<json::Value, Function, json::Value> << 1) |
        (std::is_invocable_r_v<json::Value, Function, const HttpRequest&, const Dependency&> << 2) |
        (std::is_invocable_r_v<std::string, Function, const HttpRequest&, const Dependency&> << 3) |
        (std::is_invocable_r_v<json::Value, Function, const HttpRequest&> << 4) |
        (std::is_invocable_r_v<std::string, Function, const HttpRequest&> << 5);
```

Хотим сделать красиво

[Не]очевидное решение

```
template <class Dependency>
template <class Function>
HttpWith<Dependency>::Callback::Callback(Function func) {
    namespace json = formats::json;
    constexpr unsigned kMatches =

        (std::is_invocable_r_v<json::Value, Function, json::Value, const Dependency&> << 0) |
        (std::is_invocable_r_v<json::Value, Function, json::Value> << 1) |
        (std::is_invocable_r_v<json::Value, Function, const HttpRequest&, const Dependency&> << 2) |
        (std::is_invocable_r_v<std::string, Function, const HttpRequest&, const Dependency&> << 3) |
        (std::is_invocable_r_v<json::Value, Function, const HttpRequest&> << 4) |
        (std::is_invocable_r_v<std::string, Function, const HttpRequest&> << 5);
```

Хотим сделать красиво

[Не]очевидное решение

```cpp
template <class Dependency>
template <class Function>
HttpWith<Dependency>::Callback::Callback(Function func) {
    namespace json = formats::json;
    constexpr unsigned kMatches =

        (std::is_invocable_r_v<json::Value, Function, json::Value, const Dependency&> << 0) |
        (std::is_invocable_r_v<json::Value, Function, json::Value> << 1) |
        (std::is_invocable_r_v<json::Value, Function, const HttpRequest&, const Dependency&> << 2) |
        (std::is_invocable_r_v<std::string, Function, const HttpRequest&, const Dependency&> << 3) |
        (std::is_invocable_r_v<json::Value, Function, const HttpRequest&> << 4) |
        (std::is_invocable_r_v<std::string, Function, const HttpRequest&> << 5);
```

Хотим сделать красиво
[Не]очевидное решение

```
static_assert(
    kMatches,
    "Failed to find a matching signature. See the easy::HttpWith::Callback docs for info on "
    "supported signatures"
);

constexpr bool has_single_match = ((kMatches & (kMatches - 1)) == 0);
static_assert(
    has_single_match,
    "Found more than one matching signature, probably due to `auto` usage in parameters. See "
    "the easy::HttpWith::Callback docs for info on supported signatures");
```

Хотим сделать красиво
[Не]очевидное решение

```
static_assert(
    kMatches,
    "Failed to find a matching signature. See the easy::HttpWith::Callback docs for info on "
    "supported signatures"
);

constexpr bool has_single_match = ((kMatches & (kMatches - 1)) == 0);
static_assert(
    has_single_match,
    "Found more than one matching signature, probably due to `auto` usage in parameters. See "
    "the easy::HttpWith::Callback docs for info on supported signatures");
```

Хотим сделать красиво

[Не]очевидное решение

```
static_assert(
    kMatches,
    "Failed to find a matching signature. See the easy::HttpWith::Callback docs for info on "
    "supported signatures"
);

constexpr bool has_single_match = ((kMatches & (kMatches - 1)) == 0);
static_assert(
    has_single_match,
    "Found more than one matching signature, probably due to `auto` usage in parameters. See "
    "the easy::HttpWith::Callback docs for info on supported signatures");
```

Хотим сделать красиво

[Не]очевидное решение

```
static_assert(
    kMatches,
    "Failed to find a matching signature. See the easy::HttpWith::Callback docs for info on "
    "supported signatures"
);

constexpr bool has_single_match = ((kMatches & (kMatches - 1)) == 0);
static_assert(
    has_single_match,
    "Found more than one matching signature, probably due to `auto` usage in parameters. See "
    "the easy::HttpWith::Callback docs for info on supported signatures");
```

Хотим сделать красиво
[Не]очевидное решение

```
static_assert(
    kMatches,
    "Failed to find a matching signature. See the easy::HttpWith::Callback docs for info on "
    "supported signatures"
);

constexpr bool has_single_match = ((kMatches & (kMatches - 1)) == 0);
static_assert(
    has_single_match,
    "Found more than one matching signature, probably due to `auto` usage in parameters. See "
    "the easy::HttpWith::Callback docs for info on supported signatures");
```

Хотим сделать красиво

[Не]очевидное решение

```
if constexpr (kMatches & 1) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase& deps) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody()), GetDependencies(deps)));
    };
} else if constexpr (kMatches & 2) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase&) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody())));
    };
} else if constexpr (kMatches & 4) {
```

Хотим сделать красиво

[Не]очевидное решение

```
if constexpr (kMatches & 1) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase& deps) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody()), GetDependencies(deps)));
    };
} else if constexpr (kMatches & 2) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase&) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody())));
    };
} else if constexpr (kMatches & 4) {
```

Хотим сделать красиво

[Не]очевидное решение

```
if constexpr (kMatches & 1) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase& deps) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody()), GetDependencies(deps)));
    };
} else if constexpr (kMatches & 2) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase&) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody())));
    };
} else if constexpr (kMatches & 4) {
```

Хотим сделать красиво

[Не]очевидное решение

```
if constexpr (kMatches & 1) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase& deps) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody()), GetDependencies(deps)));
    };
} else if constexpr (kMatches & 2) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase&) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody())));
    };
} else if constexpr (kMatches & 4) {
```

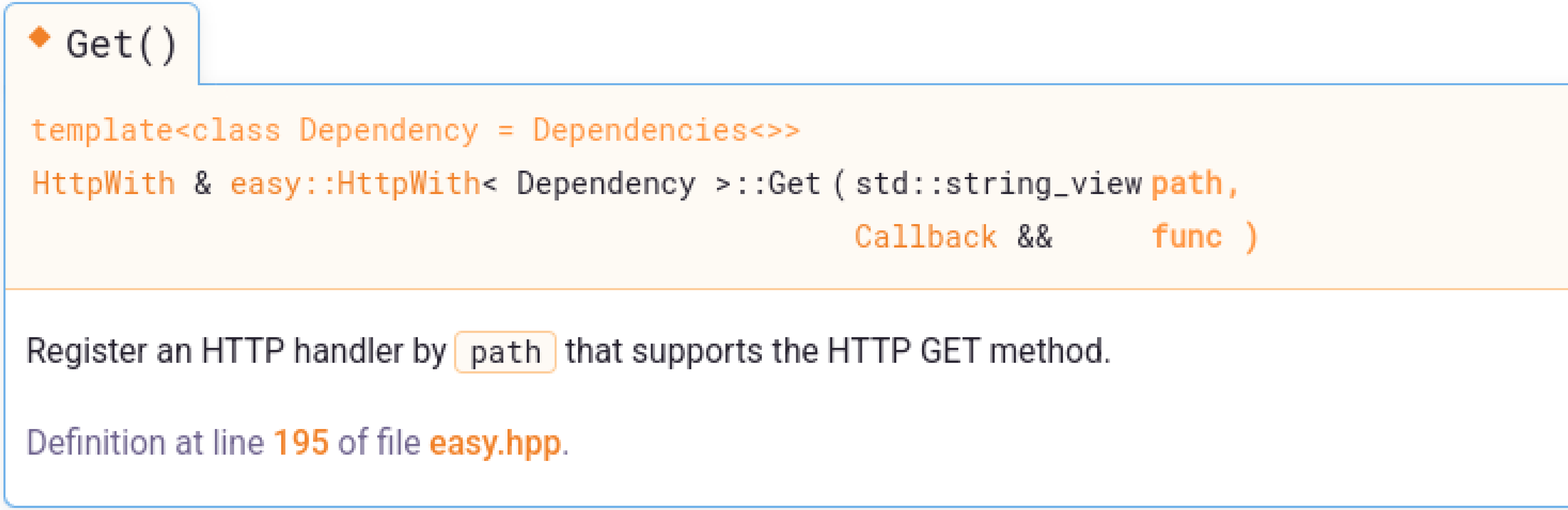
Хотим сделать красиво

[Не]очевидное решение

```
if constexpr (kMatches & 1) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase& deps) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody()), GetDependencies(deps)));
    };
} else if constexpr (kMatches & 2) {
    func_ = [f = std::move(func)](const HttpRequest& req, const impl::DependenciesBase&) {
        req.GetHttpResponse().SetContentType(http::content_type::kApplicationJson);
        return json::ToString(f(json::FromString(req.RequestBody())));
    };
} else if constexpr (kMatches & 4) {
```

Хотим сделать красиво

[Не]очевидное решение



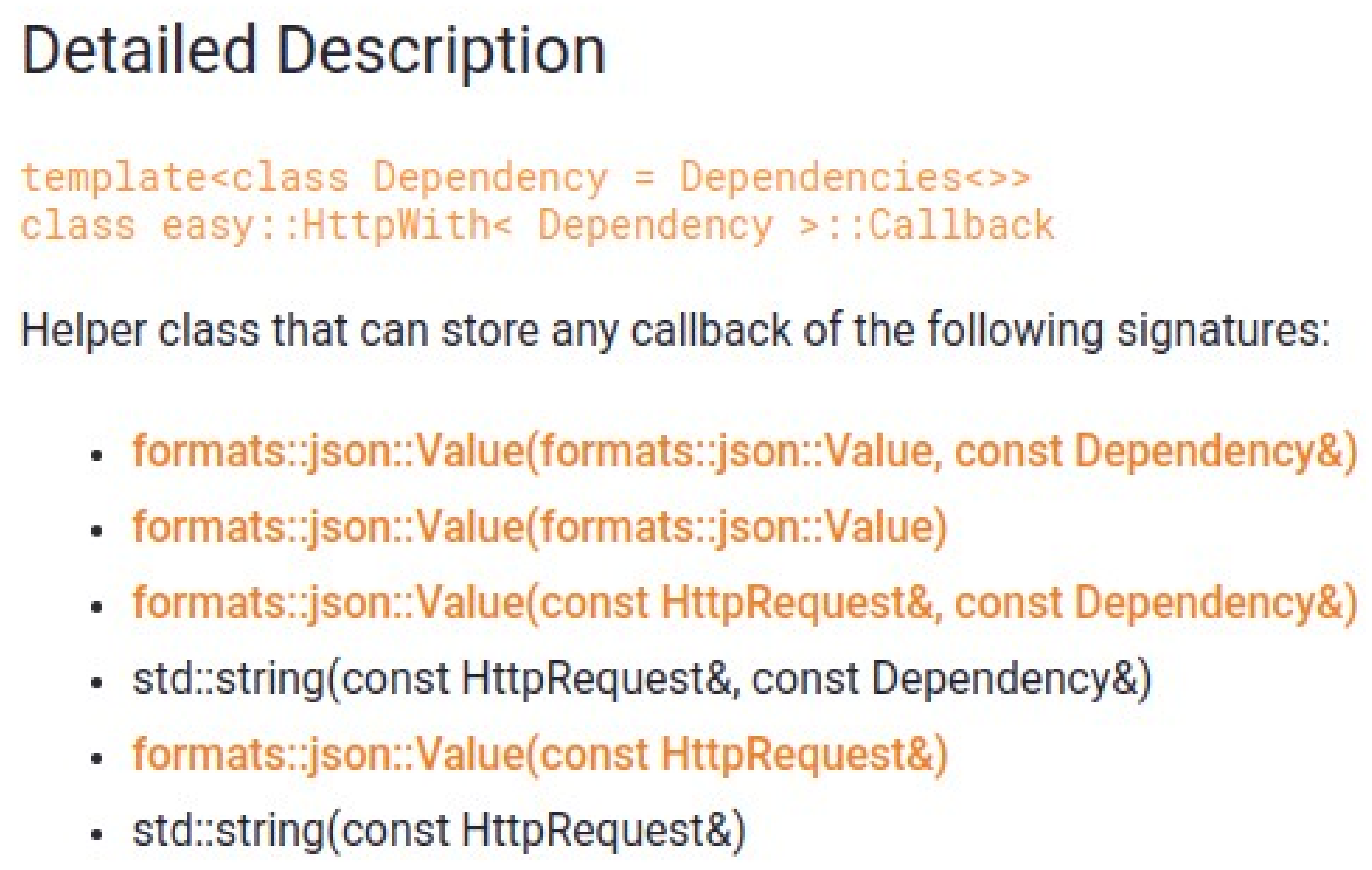
◆ Get()

```
template<class Dependency = Dependencies<>>
HttpWith & easy::HttpWith< Dependency >::Get ( std::string_view path,
                                                Callback &&      func )
```

Register an HTTP handler by `path` that supports the HTTP GET method.

Definition at line 195 of file easy.hpp.

Хотим сделать красиво
[Не]очевидное решение



Detailed Description

```
template<class Dependency = Dependencies<>>
class easy::HttpWith< Dependency >::Callback
```

Helper class that can store any callback of the following signatures:

- formats::json::Value(formats::json::Value, const Dependency&)
- formats::json::Value(formats::json::Value)
- formats::json::Value(const HttpRequest&, const Dependency&)
- std::string(const HttpRequest&, const Dependency&)
- formats::json::Value(const HttpRequest&)
- std::string(const HttpRequest&)

Правильная балансировка

Простой подход

```
FdControl::FdControl()
    : read_(current_task::GetEventThread())
    , write_(current_task::GetEventThread())
{}
```

Простой подход

И его проблемы

```
18272 antoshk+  20   0 3009228 155936  52192 R  87.6   1.0   0:16.39 event-worker_0
18273 antoshk+  20   0 3009228 155936  52192 R  34.0   1.0   0:05.87 event-worker_1
```

Простой подход

И его проблемы

```
18272 antoshk+  20   0 3009228 155936  52192 R  87.6   1.0   0:16.39 event-worker_0
18273 antoshk+  20   0 3009228 155936  52192 R  34.0   1.0   0:05.87 event-worker_1
```

Простой подход
И его проблемы

```
18272 antoshk+  20   0 3009228 155936  52192 R  87.6   1.0   0:16.39 event-worker_0
18273 antoshk+  20   0 3009228 155936  52192 R  34.0   1.0   0:05.87 event-worker_1
```

Простой подход
И его проблемы

```
FdControl::FdControl()
    : read_(current_task::GetEventThread())
    , write_(current_task::GetEventThread())
{}
```

Простой подход

И его проблемы

```
FdControl::FdControl()
    : read_(current_task::GetEventThread())
    , write_(current_task::GetEventThread())
{}
```

Простой подход

И его проблемы

```
FdControl::FdControl()
    : read_(current_task::GetEventThread())
    , write_(current_task::GetEventThread())
{}
```

Простой подход

[Не]очевидное решение

```
FdControl::FdControl()
    : read_(current_task::GetEventThread())
    , write_(current_task::GetEventThread())
{}


FdControl::FdControl(const ev::ThreadControl& control)
    : read_(control)
    , write_(control)
{}
```

Простой подход
[Не]очевидное решение

```
FdControl::FdControl()
    : read_(current_task::GetEventThread())
    , write_(current_task::GetEventThread())
{}
```

```
FdControl::FdControl(const ev::ThreadControl& control)
    : read_(control)
    , write_(control)
{}
```

Простой подход

[Не]очевидное решение

```
FdControl::FdControl()
    : read_(current_task::GetEventThread())
    , write_(current_task::GetEventThread())
{}


FdControl::FdControl(const ev::ThreadControl& control)
    : read_(control)
    , write_(control)
{}
```

Простой подход

[Не]очевидное решение

```
18272 antoshk+  20   0 3009228 155936  52192 R  87.6   1.0   0:16.39 event-worker_0
18273 antoshk+  20   0 3009228 155936  52192 R  34.0   1.0   0:05.87 event-worker_1
```

```
23495 antoshk+  20   0 3008264 155180  52268 R  66.7   1.0   0:31.73 event-worker_1
23494 antoshk+  20   0 3008264 155180  52268 R  61.9   1.0   0:31.07 event-worker_0
```

Простой подход

[Не]очевидное решение

```
18272 antoshk+  20   0 3009228 155936  52192 R  87.6   1.0   0:16.39 event-worker_0
18273 antoshk+  20   0 3009228 155936  52192 R  34.0   1.0   0:05.87 event-worker_1
```

```
23495 antoshk+  20   0 3008264 155180  52268 R  66.7   1.0   0:31.73 event-worker_1
23494 antoshk+  20   0 3008264 155180  52268 R  61.9   1.0   0:31.07 event-worker_0
```

Простой подход

[Не]очевидное решение

```
18272 antoshk+  20   0 3009228 155936  52192 R  87.6   1.0   0:16.39 event-worker_0
18273 antoshk+  20   0 3009228 155936  52192 R  34.0   1.0   0:05.87 event-worker_1
```

```
23495 antoshk+  20   0 3008264 155180  52268 R  66.7   1.0   0:31.73 event-worker_1
23494 antoshk+  20   0 3008264 155180  52268 R  61.9   1.0   0:31.07 event-worker_0
```

v == vector{"*"}

C++ source #1
C++
C++ source #2
C++
x86-64 gcc (trunk) (Editor #1)
x86-64 gcc (trunk)
-std=c++23 -O2
x86-64 gcc (trunk) (Editor #2)
x86-64 gcc (trunk)
-std=c++23 -O2
Output...
Filter...
Libraries
Overrides
Add new...
Add tool...
Output (0/0)
x86-64 gcc (trunk)
- 1351ms (69959B) ~4560 lines filtered
Compiler License

```cpp
#include <vector>
bool test1(const std::vector<int>& in) {
    return in == std::vector<int>{42};
}
```

```cpp
#include <vector>
bool test2(const std::vector<int>& in) {
    return in.size() == 1 && in[0] == 42;
}
```

```asm
test1(std::vector<int, std::allocator<int> > const&):
        push    rbx
        mov     rbx, rdi
        mov     edi, 4
        call    operator new(unsigned long)
        mov     rdx, QWORD PTR [rbx]
        mov     DWORD PTR [rax], 42
        mov     rdi, rax
        mov     rax, QWORD PTR [rbx+8]
        xor     ebx, ebx
        sub     rax, rdx
        cmp     rax, 4
        je      .L7
        mov     esi, 4
        call    operator delete(void*, unsigned long)
        mov     eax, ebx
        pop     rbx
        ret
.L7:
        cmp     DWORD PTR [rdx], 42
        mov     esi, 4
        sete    bl
        call    operator delete(void*, unsigned long)
        mov     eax, ebx
        pop     rbx
        ret
```

```asm
test2(std::vector<int, std::allocator<int> > const&):
        mov     rcx, QWORD PTR [rdi]
        mov     rax, QWORD PTR [rdi+8]
        xor     edx, edx
        sub     rax, rcx
        cmp     rax, 4
        jne     .L1
        cmp     DWORD PTR [rcx], 42
        sete    dl
.L1:
        mov     eax, edx
        ret
```

Хочется поправить

```
v == std::vector{42}
```

Хочется поправить

v == std::vector{42}

1 Везде!

Хочется поправить

v == std::vector{42}

1. Везде!

2. И не обязательно прям такой паттерн!

Хочется поправить

```
v == std::vector{42}
```

1. Везде!
2. И не обязательно прям такой паттерн!
3. Сделать чтобы подобная проблема больше не возникала

Хочется поправить

v == std::vector{42}

1. Везде!
2. И не обязательно прям такой паттерн!
3. Сделать чтобы подобная проблема больше не возникала
4. Работало и для кастомных контейнеров

Feature request в компилятор!

Спасибо

Полухин Антон

Эксперт-разработчик C++

antoshkka@gmail.com

antoshkka@yandex-team.ru

https://github.com/apolukhin

https://stdcpp.ru/

https://github.com/userver-framework