

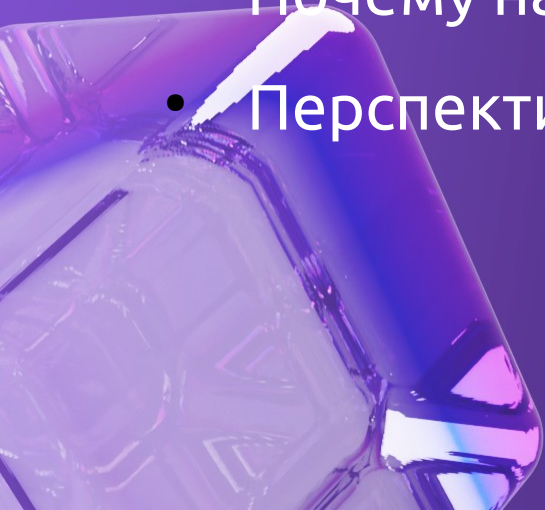


C++ сегодня и завтра



Содержание

- Какой язык выбрать?
- Чем хорош C++
- Почему на C++ жалуются
- Перспективы





Каждой задаче свой
язык!





Современные задачи



Современные задачи

- Сделать логику веб странички



Современные задачи

- Сделать логику веб странички
 - JavaScript



Современные задачи

- Сделать логику веб странички
 - JavaScript
- Собрать прототип решения



Современные задачи

- Сделать логику веб странички
 - JavaScript
- Собрать прототип решения
 - Python



Современные задачи

- Сделать логику веб странички
 - JavaScript
- Собрать прототип решения
 - Python
- Разработка под мобильные устройства



Современные задачи

- Сделать логику веб странички
 - JavaScript
- Собрать прототип решения
 - Python
- Разработка под мобильные устройства
 - Java



Современные задачи

- Сделать логику веб странички
 - JavaScript
- Собрать прототип решения
 - Python
- Разработка под мобильные устройства
 - Java
 - Swift



Современные задачи

- Сделать логику веб странички
 - JavaScript
- Собрать прототип решения
 - Python
- Разработка под мобильные устройства
 - Java
 - Swift
 - C#





А где C++ ?





Где применяется C++





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки



Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос



Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили



Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО

- Компиляторы
- Виртуальные машины





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО

- Компиляторы
- Виртуальные машины
- Драйверы и части ОС





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО

- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО

- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО

- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО

- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО

- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО
- Распознавание образов
- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО
- Распознавание образов
- Обработка изображений
- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО
- Распознавание образов
- Обработка изображений
- Web фреймворки
- Web страницы
- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО
- Распознавание образов
- Обработка изображений
- Web фреймворки
- Web страницы
- Базы данных
- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО
- Распознавание образов
- Обработка изображений
- Web фреймворки
- Web страницы
- Базы данных
- Proxy
- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО
- Распознавание образов
- Обработка изображений
- Web фреймворки
- Web страницы
- Базы данных
- Proxy
- Embedded
- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки
- Торренты
- Игровые движки
- Космос
- Самолеты
- Автомобили
- Медицинское ПО
- Распознавание образов
- Обработка изображений
- Web фреймворки
- Web страницы
- Базы данных
- Proxy
- Embedded
- ...
- Компиляторы
- Виртуальные машины
- Драйверы и части ОС
- Биржа
- Офисные приложения
- Банкоматы
- САПР
- Рендеры
- Химия / физика
- Машинное обучение





Почему выбирают C++ ?





История про 1%





Масштабы современного IT



Масштабы современного IT

1.000.000 серверов (Google в 2011)



Масштабы современного IT

1.000.000 серверов (Google в 2011)

– 1% => 10.000 серверов





Масштабы современного IT

1.000.000 серверов (Google в 2011)

- 1% => 10.000 серверов
- Сервер надо:





Масштабы современного IT

1.000.000 серверов (Google в 2011)

- 1% => 10.000 серверов
- Сервер надо:
 - Питать



Масштабы современного IT

1.000.000 серверов (Google в 2011)

- 1% => 10.000 серверов
- Сервер надо:
 - Питать
 - Охлаждать



Масштабы современного IT

1.000.000 серверов (Google в 2011)

- 1% => 10.000 серверов
- Сервер надо:
 - Питать
 - Охлаждать
 - Обслуживать



Масштабы современного IT

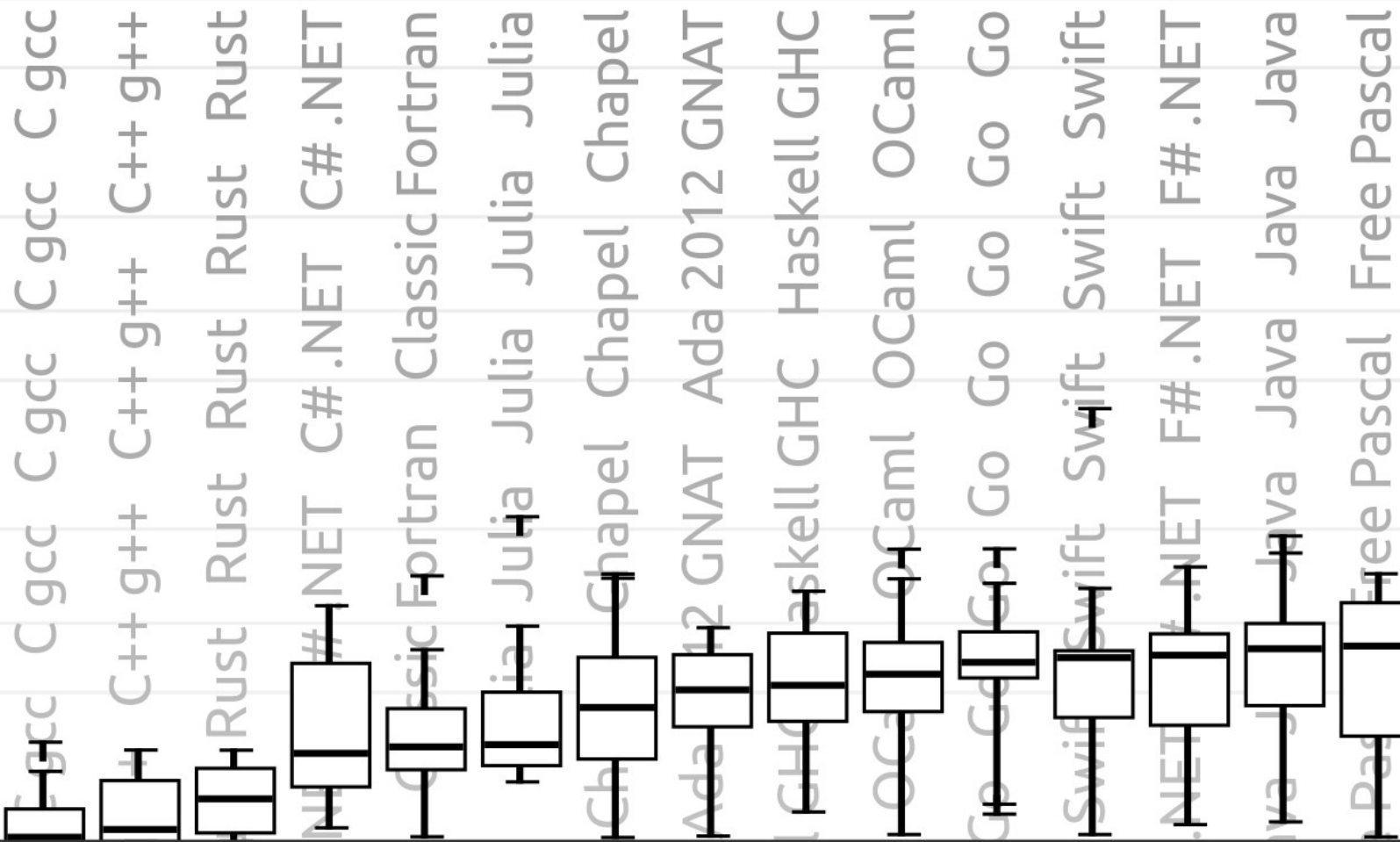
1.000.000 серверов (Google в 2011)

- 1% => 10.000 серверов
- Сервер надо:
 - Питать
 - Охлаждать
 - Обслуживать
 - ...

program cpu time / lowest

300
100
50
30
10
5
3
1

benchmarks game



12 May 2022 u64q



C++



- Экономия в железе и его поддержке



C++



- Экономия в железе и его поддержке
 - Кстати, а сколько “стоит” динамическая аллокация?

C++ source #1

C++

```
1 #include <array>
2 #include <string>
3
4
5 auto example5() {
6     std::array<std::string, 5> a = {
7         "One",
8         "Two",
9         "Three",
10        "Four",
11        "Five",
12    };
13    return a;
14 }
```

x86-64 gcc 12.2 (Editor #1)

x86-64 gcc 12.2

-std=c++2a -O2

Output... Filter... Libraries (1) + Add new... + Add tool...

```
1 example5[abi:cxx11]():
2     lea rdx, [rdi+16]
3     mov ecx, 30548
4     mov BYTE PTR [rdi+18], 101
5     mov rax, rdi
6     mov QWORD PTR [rdi], rdx
7     mov edx, 28239
8     mov WORD PTR [rdi+16], dx
9     lea rdx, [rdi+48]
10    mov QWORD PTR [rdi+32], rdx
11    lea rdx, [rdi+80]
12    mov QWORD PTR [rdi+64], rdx
13    lea rdx, [rdi+112]
14    mov QWORD PTR [rdi+96], rdx
15    lea rdx, [rdi+144]
16    mov QWORD PTR [rdi+8], 3
17    mov BYTE PTR [rdi+19], 0
18    mov WORD PTR [rdi+48], cx
19    mov BYTE PTR [rdi+50], 111
20    mov QWORD PTR [rdi+40], 3
21    mov BYTE PTR [rdi+51], 0
22    mov DWORD PTR [rdi+80], 1701996628
23    mov BYTE PTR [rdi+84], 101
24    mov QWORD PTR [rdi+72], 5
25    mov BYTE PTR [rdi+85], 0
26    mov DWORD PTR [rdi+112], 1920298822
27    mov QWORD PTR [rdi+104], 4
28    mov BYTE PTR [rdi+116], 0
29    mov QWORD PTR [rdi+128], rdx
30    mov DWORD PTR [rdi+144], 1702259014
31    mov QWORD PTR [rdi+136], 4
32    mov BYTE PTR [rdi+148], 0
33    ret
```

x86-64 clang 15.0.0 (Editor #1)

x86-64 clang 15.0.0

-std=c++2a -O2

Output... Filter... Libraries (1) + Add new... + Add tool...

```
1 example5[abi:cxx11](): # @example5
2     mov rax, rdi
3     lea rcx, [rdi + 16]
4     mov qword ptr [rdi], rcx
5     mov dword ptr [rdi + 16], 6647375
6     mov qword ptr [rdi + 8], 3
7     lea rcx, [rdi + 48]
8     mov qword ptr [rdi + 32], rcx
9     mov dword ptr [rdi + 48], 7305044
10    mov qword ptr [rdi + 40], 3
11    lea rcx, [rdi + 80]
12    mov qword ptr [rdi + 64], rcx
13    mov dword ptr [rdi + 80], 1701996628
14    mov word ptr [rdi + 84], 101
15    mov qword ptr [rdi + 72], 5
16    lea rcx, [rdi + 112]
17    mov qword ptr [rdi + 96], rcx
18    mov dword ptr [rdi + 112], 1920298822
19    mov qword ptr [rdi + 104], 4
20    mov byte ptr [rdi + 116], 0
21    lea rcx, [rdi + 144]
22    mov qword ptr [rdi + 128], rcx
23    mov dword ptr [rdi + 144], 1702259014
24    mov qword ptr [rdi + 136], 4
25    mov byte ptr [rdi + 148], 0
26    ret
```





C++



- Экономия в железе и его поддержке



C++



- Экономия в железе и его поддержке
- Экономия батареечки в вашем телефоне



C++



- Экономия в железе и его поддержке
- Экономия батарейки в вашем телефоне
- Меньше время отклика



C++



- Экономия в железе и его поддержке
- Экономия батареи в вашем телефоне
- Меньше время отклика
- Предсказуемое время отклика



C++

- Экономия в железе и его поддержке
- Экономия батареи в вашем телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock



C++

- Экономия в железе и его поддержке
- Экономия батареи в вашем телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock
- Поддержка множества платформ



C++

- Экономия в железе и его поддержке
- Экономия батареи в вашем телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock
- Поддержка множества платформ
- Совместимость с C





Почему на C++ жалуются?



Hello world





Hello world

```
char* get1();
```

```
char* get2();
```

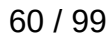
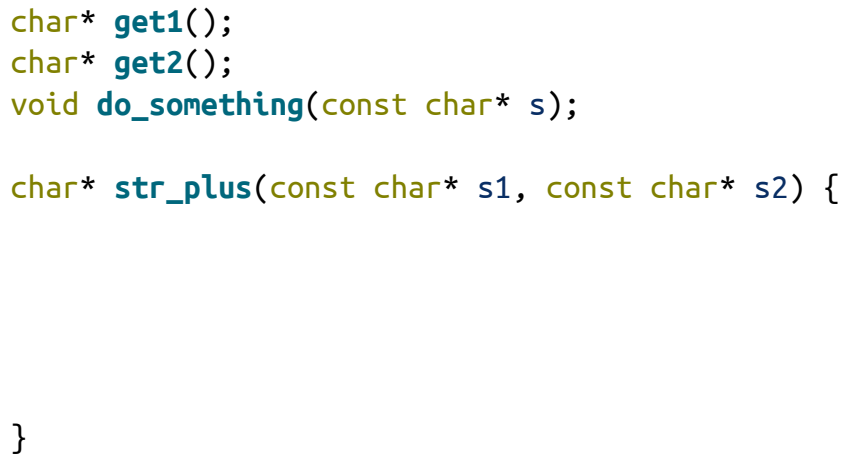




Hello world

```
char* get1();  
char* get2();  
void do_something(const char* s);
```







Hello world



```
char* get1();  
char* get2();  
void do_something(const char* s);  
  
char* str_plus(const char* s1, const char* s2) {  
    unsigned len = strlen(s1) + strlen(s2) + 1;  
  
}
```



Hello world



```
char* get1();  
char* get2();  
void do_something(const char* s);  
  
char* str_plus(const char* s1, const char* s2) {  
    unsigned len = strlen(s1) + strlen(s2) + 1;  
    char* result = (char*)malloc(len);  
  
}
```



Hello world



```
char* get1();
char* get2();
void do_something(const char* s);

char* str_plus(const char* s1, const char* s2) {
    unsigned len = strlen(s1) + strlen(s2) + 1;
    char* result = (char*)malloc(len);
    strcat(result, s1);
    strcat(result, s2);
}
```



Hello world

```
char* get1();  
char* get2();  
void do_something(const char* s);  
  
char* str_plus(const char* s1, const char* s2) {  
    unsigned len = strlen(s1) + strlen(s2) + 1;  
    char* result = (char*)malloc(len);  
    strcat(result, s1);  
    strcat(result, s2);  
    return result;  
}
```




Hello world

```
char* get1();
char* get2();
void do_something(const char* s);

char* str_plus(const char* s1, const char* s2) {
    unsigned len = strlen(s1) + strlen(s2) + 1;
    char* result = (char*)malloc(len);
    strcat(result, s1);
    strcat(result, s2);
    return result;
}

void example1() {
```



Hello world

```
char* get1();
char* get2();
void do_something(const char* s);

char* str_plus(const char* s1, const char* s2) {
    unsigned len = strlen(s1) + strlen(s2) + 1;
    char* result = (char*)malloc(len);
    strcat(result, s1);
    strcat(result, s2);
    return result;
}

void example1() {
    char *s1 = get1(), *s2 = get2();

    // free(s1); ???
    // free(s2); ???
}
```





Hello world

```
char* get1();
char* get2();
void do_something(const char* s);

char* str_plus(const char* s1, const char* s2) {
    unsigned len = strlen(s1) + strlen(s2) + 1;
    char* result = (char*)malloc(len);
    strcat(result, s1);
    strcat(result, s2);
    return result;
}

void example1() {
    char *s1 = get1(), *s2 = get2();
    char* result = str_plus(s1, s2);

    free(result);
    // free(s1); ???
    // free(s2); ???
}
```





Hello world

```
char* get1();
char* get2();
void do_something(const char* s);

char* str_plus(const char* s1, const char* s2) {
    unsigned len = strlen(s1) + strlen(s2) + 1;
    char* result = (char*)malloc(len);
    strcat(result, s1);
    strcat(result, s2);
    return result;
}

void example1() {
    char *s1 = get1(), *s2 = get2();
    char* result = str_plus(s1, s2);
    do_something(result);
    free(result);
    // free(s1); ???
    // free(s2); ???
}
```





Hello world

```
char* get1();
char* get2();
void do_something(const char* s);

char* str_plus(const char* s1, const char* s2) {
    unsigned len = strlen(s1) + strlen(s2) + 1;
    char* result = (char*)malloc(len);
    strcat(result, s1);
    strcat(result, s2);
    return result;
}

void example1() {
    char *s1 = get1(), *s2 = get2();
    char* result = str_plus(s1, s2);
    do_something(result);
    free(result);
    // free(s1); ???
    // free(s2); ???
}
```





Это был не C++!



Bot C++ :



C++ Hello world

```
std::string get_str1();  
std::string get_str2();  
void do_something(std::string_view s);  
  
void example2() {  
    auto result = get_str1() + get_str2();  
    do_something(result);  
}
```




C++ Hello world



```
std::string get_str1();  
std::string get_str2();  
void do_something(std::string_view s);  
  
void example2() {  
    auto result = get_str1() + get_str2();  
    do_something(result);  
}
```



C++ Hello world

```
std::string get_str1();  
std::string get_str2();  
void do_something(std::string_view s);  
  
void example2() {  
    auto result = get_str1() + get_str2();  
    do_something(result);  
}
```



C++ Hello world

```
std::string get_str1();  
std::string get_str2();  
void do_something(std::string_view s);  
  
void example2() {  
    auto result = get_str1() + get_str2();  
    do_something(result);  
}
```



C++ Hello world

```
std::string get_str1();  
std::string get_str2();  
void do_something(std::string_view s);  
  
void example2() {  
    auto result = get_str1() + get_str2();  
    do_something(result);  
}
```



RAII + exceptions



He C++ Hello world (найдите ошибки)

```
char* get1();
char* get2();
void do_something(const char* s);

01: char* str_plus(const char* s1, const char* s2) {
02:     unsigned len = strlen(s1) + strlen(s2) + 1;
03:     char* result = (char*)malloc(len);
04:     strcat(result, s1);
05:     strcat(result, s2);
06:     return result;
07: }
08:
09: void example1() {
10:     char *s1 = get1(), *s2 = get2();
11:     char* result = str_plus(s1, s2);
12:     do_something(result);
13:     free(result);
14:     // free(s1); ???
15:     // free(s2); ???
16: }
```





C++ Hello world

```
std::string get_str1();  
std::string get_str2();  
void do_something(const char* s);  
  
void example2() {  
    std::string result = get_str1() + get_str2();  
    do_something(result.c_str());  
}
```



C++ Hello world

```
std::string get_str1();  
std::string get_str2();  
void do_something(const char* s);  
  
void example2() {  
    std::string result = get_str1() + get_str2();  
    do_something(result.c_str());    // А если исключение ???  
}
```




C++ Hello world

```
std::string get_str1();  
std::string get_str2();  
void do_something(const char* s);  
  
void example2() {  
    std::string result = get_str1() + get_str2();  
    do_something(result.c_str());    // А если исключение, то деструктор строки отработает  
}
```



C++ Hello world



```
std::string get_str1();  
std::string get_str2();  
void do_something(const char* s);  
  
void example2() {  
    std::string result = get_str1() + get_str2();  
    do_something(result.c_str());  
}
```

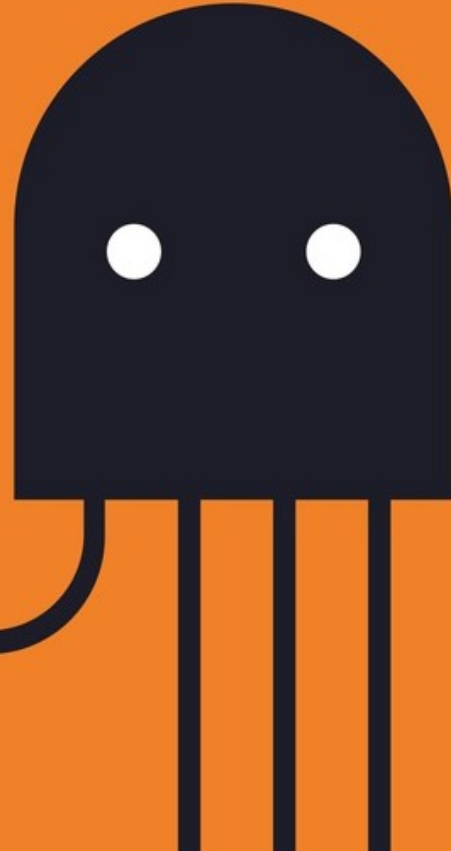


Невыполнимая C++ задача:



За 10 минут напишите
работающий HTTP сервер

<https://userver.tech/>





C++ HTTP Hello world

```
#include <userver/components/minimal_server_component_list.hpp>
#include <userver/server/handlers/http_handler_base.hpp>
#include <userver/utils/daemon_run.hpp>

struct Hello final : public server::handlers::HttpHandlerBase {
    static constexpr std::string_view kName = "handler-hello-sample";

    using HttpHandlerBase::HttpHandlerBase;

    std::string HandleRequestThrow(
        const server::http::HttpRequest&,
        server::request::RequestContext&) const override {
        return "Hello world!\n";
    }
};

int main(int argc, char* argv[]) {
    const auto component_list = components::MinimalServerComponentList().Append<Hello>();
    return utils::DaemonMain(argc, argv, component_list);
}
```





C++ HTTP Hello world

```
#include <userver/components/minimal_server_component_list.hpp>
#include <userver/server/handlers/http_handler_base.hpp>
#include <userver/utils/daemon_run.hpp>

struct Hello final : public server::handlers::HttpHandlerBase {
    static constexpr std::string_view kName = "handler-hello-sample";

    using HttpHandlerBase::HttpHandlerBase;

    std::string HandleRequestThrow(
        const server::http::HttpRequest&,
        server::request::RequestContext&) const override {
        return "Hello world!\n";
    }
};

int main(int argc, char* argv[]) {
    const auto component_list = components::MinimalServerComponentList().Append<Hello>();
    return utils::DaemonMain(argc, argv, component_list);
}
```





C++ HTTP Hello world

```
#include <userver/components/minimal_server_component_list.hpp>
#include <userver/server/handlers/http_handler_base.hpp>
#include <userver/utils/daemon_run.hpp>

struct Hello final : public server::handlers::HttpHandlerBase {
    static constexpr std::string_view kName = "handler-hello-sample";

    using HttpHandlerBase::HttpHandlerBase;

    std::string HandleRequestThrow(
        const server::http::HttpRequest&,
        server::request::RequestContext&) const override {
        return "Hello world!\n";
    }
};

int main(int argc, char* argv[]) {
    const auto component_list = components::MinimalServerComponentList().Append<Hello>();
    return utils::DaemonMain(argc, argv, component_list);
}
```





C++ HTTP Hello world

```
#include <userver/components/minimal_server_component_list.hpp>
#include <userver/server/handlers/http_handler_base.hpp>
#include <userver/utils/daemon_run.hpp>

struct Hello final : public server::handlers::HttpHandlerBase {
    static constexpr std::string_view kName = "handler-hello-sample";

    using HttpHandlerBase::HttpHandlerBase;

    std::string HandleRequestThrow(
        const server::http::HttpRequest&,
        server::request::RequestContext&) const override {
        return "Hello world!\n";
    }
};

int main(int argc, char* argv[]) {
    const auto component_list = components::MinimalServerComponentList().Append<Hello>();
    return utils::DaemonMain(argc, argv, component_list);
}
```





C++ HTTP Hello world

```
#include <userver/components/minimal_server_component_list.hpp>
#include <userver/server/handlers/http_handler_base.hpp>
#include <userver/utils/daemon_run.hpp>

struct Hello final : public server::handlers::HttpHandlerBase {
    static constexpr std::string_view kName = "handler-hello-sample";

    using HttpHandlerBase::HttpHandlerBase;

    std::string HandleRequestThrow(
        const server::http::HttpRequest&,
        server::request::RequestContext&) const override {
        return "Hello world!\n";
    }
};

int main(int argc, char* argv[]) {
    const auto component_list = components::MinimalServerComponentList().Append<Hello>();
    return utils::DaemonMain(argc, argv, component_list);
}
```





C++ - это очень весело





C++ - это весело



C++ - ЭТО весело

- `int array[Fib<5>::value];`



C++ - ЭТО ВЕСЕЛО

- `int array[Fib<5>::value];`
- `boost::pfr::get<1>(some_variable);`



C++ - ЭТО ВЕСЕЛО

- `int array[Fib<5>::value];`
- `boost::pfr::get<1>(some_variable);`
- `#define PP_ITERATIONS 5`



А что будет “завтра”?



А “завтра” всё ещё будет актуально

- Экономия в железе и его поддержке
- Экономия батареи в вашем телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock
- Поддержка множества платформ
- Совместимость с С

Спасибо

Полухин Антон

Эксперт-разработчик C++ в Яндекс 



antoshkka@gmail.com



antoshkka@yandex-team.ru



<https://github.com/apolukhin>

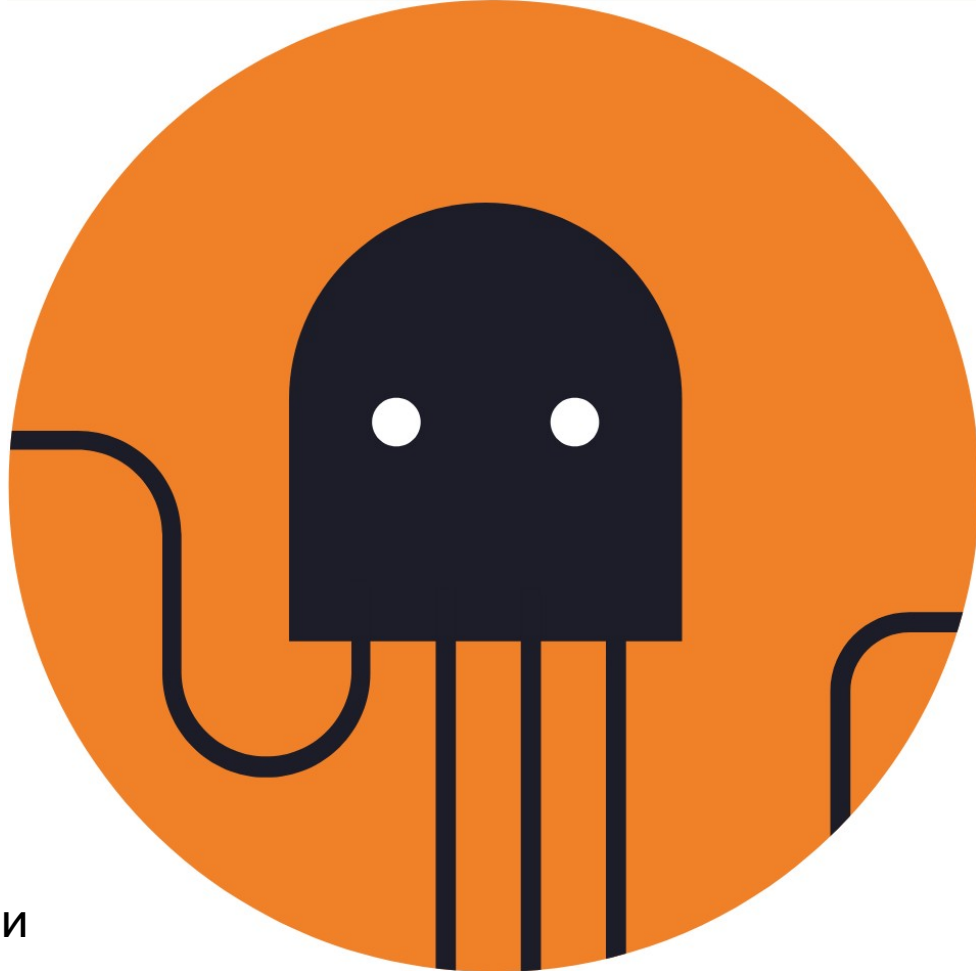


<https://stdcpp.ru/>

PF21 C++ РОССИЯ



Форма обратной связи



<https://github.com/userver-framework>