C++ сегодня и завтра

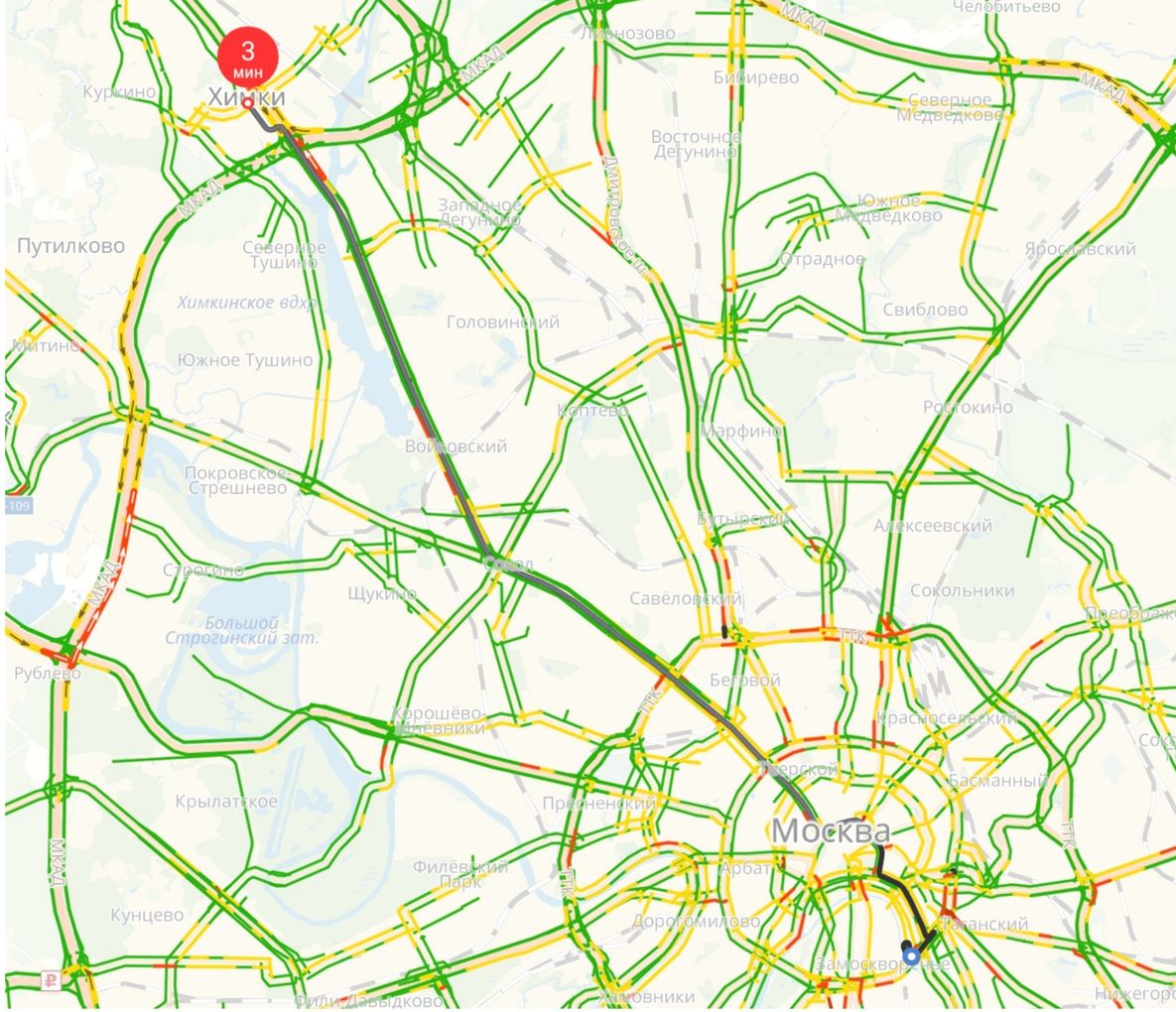
Полухин Антон

Antony Polukhin

Яндекс Go

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, склеить разные решения воедино

Современные задачи

Сделать логику веб-странички

- JavaScript

Собрать прототип решения, склеить разные решения воедино

- Python

Современные задачи

Сделать логику веб-странички

- JavaScript

Собрать прототип решения, склеить разные решения воедино

- Python

Разработка под мобильные устройства

Современные задачи

Сделать логику веб-странички

- JavaScript

Собрать прототип решения, склеить разные решения воедино

- Python

Разработка под мобильные устройства

- Java

Современные задачи

Сделать логику веб-странички

- JavaScript

Собрать прототип решения, склеить разные решения воедино

- Python

Разработка под мобильные устройства

- Java
- Swift

Современные задачи

Сделать логику веб-странички

- JavaScript

Собрать прототип решения, склеить разные решения воедино

- Python

Разработка под мобильные устройства

- Java
- Swift
- Objective-C

Современные задачи

Сделать логику веб-странички

- JavaScript

Собрать прототип решения, склеить разные решения воедино

- Python

Разработка под мобильные устройства

- Java
- Swift
- Objective-C
- C#

А где C++ ?

Где применяется C++

Где применяется C++

- Браузеры

Где применяется C++

- Браузеры
- Поисковые сервисы

Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript

Где применяется C++

- Браузеры
- Поисковые сервисы
- JavaScript
- Кодеки

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
- Медицинское ПО

Где применяется C++

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

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

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

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
- 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 современных IT компаний

Масштабы современных IT компаний

- 1.000.000 серверов (Google в 2011)

Масштабы современных IT компаний

- 1.000.000 серверов (Google в 2011)

  1% => 10.000 серверов

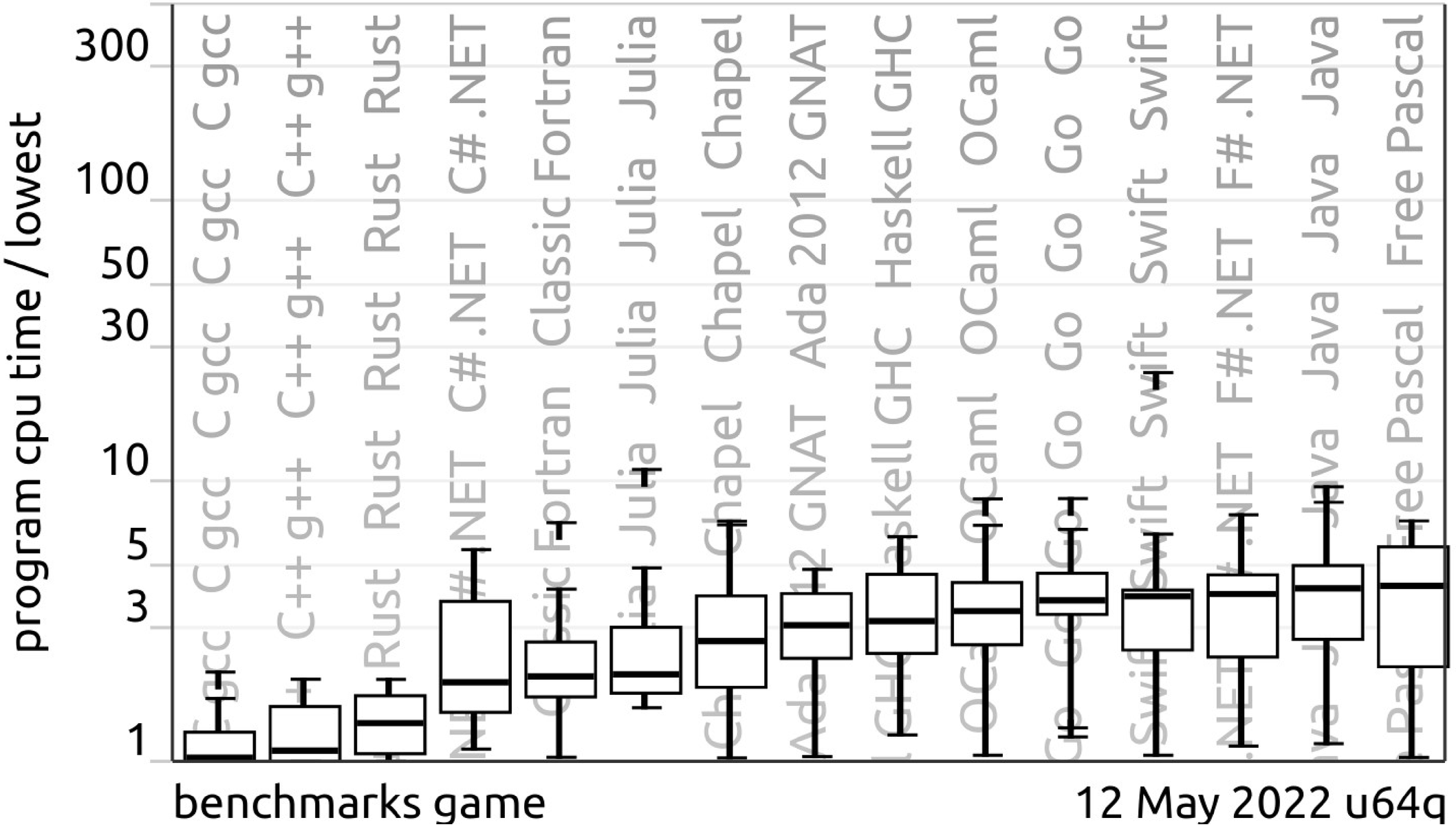
Масштабы современных IT компаний

- 1.000.000 серверов (Google в 2011)

  1% => 10.000 серверов

Каждый сервер надо питать, охлаждать, обслуживать, ...

Современные языки программирования



Почему выбирают C++?

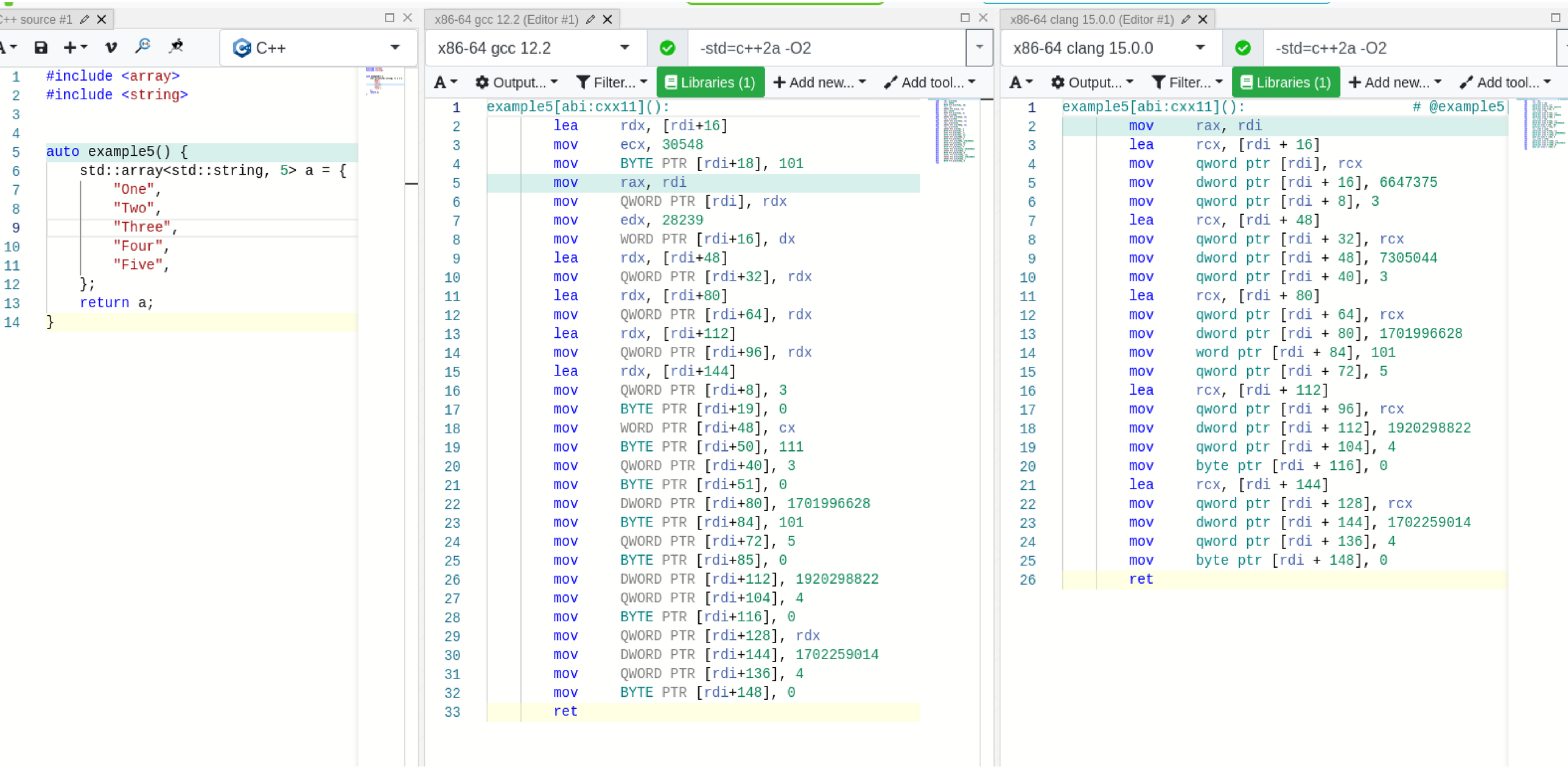
C++

C++

- Экономия в количестве железа и его поддержке

C++

- Экономия в количестве железа и его поддержке
  - Кто знает сколько «стоит» динамическая аллокация?


C++ source #1
C++
#include <array>
#include <string>
auto example5() {
std::array<std::string, 5> a = {
"One",
"Two",
"Three",
"Four",
"Five",
};
return a;
}
x86-64 gcc 12.2 (Editor #1)
x86-64 gcc 12.2
-std=c++2a -O2
Output...
Filter...
Libraries (1)
Add new...
Add tool...
example5[abi:cxx11]():
lea rdx, [rdi+16]
mov ecx, 30548
mov BYTE PTR [rdi+18], 101
mov rax, rdi
mov QWORD PTR [rdi], rdx
mov edx, 28239
mov WORD PTR [rdi+16], dx
lea rdx, [rdi+48]
mov QWORD PTR [rdi+32], rdx
lea rdx, [rdi+80]
mov QWORD PTR [rdi+64], rdx
lea rdx, [rdi+112]
mov QWORD PTR [rdi+96], rdx
lea rdx, [rdi+144]
mov QWORD PTR [rdi+8], 3
mov BYTE PTR [rdi+19], 0
mov WORD PTR [rdi+48], cx
mov BYTE PTR [rdi+50], 111
mov QWORD PTR [rdi+40], 3
mov BYTE PTR [rdi+51], 0
mov DWORD PTR [rdi+80], 1701996628
mov BYTE PTR [rdi+84], 101
mov QWORD PTR [rdi+72], 5
mov BYTE PTR [rdi+85], 0
mov DWORD PTR [rdi+112], 1920298822
mov QWORD PTR [rdi+104], 4
mov BYTE PTR [rdi+116], 0
mov QWORD PTR [rdi+128], rdx
mov DWORD PTR [rdi+144], 1702259014
mov QWORD PTR [rdi+136], 4
mov BYTE PTR [rdi+148], 0
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...
example5[abi:cxx11](): # @example5
mov rax, rdi
lea rcx, [rdi + 16]
mov qword ptr [rdi], rcx
mov dword ptr [rdi + 16], 6647375
mov qword ptr [rdi + 8], 3
lea rcx, [rdi + 48]
mov qword ptr [rdi + 32], rcx
mov dword ptr [rdi + 48], 7305044
mov qword ptr [rdi + 40], 3
lea rcx, [rdi + 80]
mov qword ptr [rdi + 64], rcx
mov dword ptr [rdi + 80], 1701996628
mov word ptr [rdi + 84], 101
mov qword ptr [rdi + 72], 5
lea rcx, [rdi + 112]
mov qword ptr [rdi + 96], rcx
mov dword ptr [rdi + 112], 1920298822
mov qword ptr [rdi + 104], 4
mov byte ptr [rdi + 116], 0
lea rcx, [rdi + 144]
mov qword ptr [rdi + 128], rcx
mov dword ptr [rdi + 144], 1702259014
mov qword ptr [rdi + 136], 4
mov byte ptr [rdi + 148], 0
ret

C++

- Экономия в количестве железа и его поддержке

C++

- Экономия в количестве железа и его поддержке
- Экономия вашей батарейки в мобильном телефоне

C++

- Экономия в количестве железа и его поддержке
- Экономия вашей батарейки в мобильном телефоне
- Меньше время отклика

C++

- Экономия в количестве железа и его поддержке
- Экономия вашей батарейки в мобильном телефоне
- Меньше время отклика
- Предсказуемое время отклика

C++

- Экономия в количестве железа и его поддержке
- Экономия вашей батарейки в мобильном телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock

C++

- Экономия в количестве железа и его поддержке
- Экономия вашей батарейки в мобильном телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock
- Поддержка множества платформ

C++

- Экономия в количестве железа и его поддержке
- Экономия вашей батарейки в мобильном телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock
- Поддержка множества платформ
- Совместимость с C

Почему на C++ жалуются

C++ Hello World

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

C++ Hello World

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

char* str_plus(const char* s1, const char* s2) {




}
```

C++ 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;




}
```

C++ 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;




}
```

C++ 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;




}
```

C++ 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;




}
```

C++ 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);




}
```

C++ 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);

}
```

C++ 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;
}
```

C++ 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() {









}
```

C++ 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();









}
```

C++ 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);




}
```

C++ 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);



}
```

C++ 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);



}
```

C++ 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++ !!!

Вот C++ код:

C++ Hello World

```
std::string get_str1();
std::string get_str2();
void do_something(const char* s);

void example2() {
    auto 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() {
    auto 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() {
    auto result = get_str1() + get_str2();
    do_something(result.c_str());
}
```

RAII

НЕ C++ Hello World (найдите баги)

```
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: }
```

НЕ C++ Hello World (найдите баги)

```
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

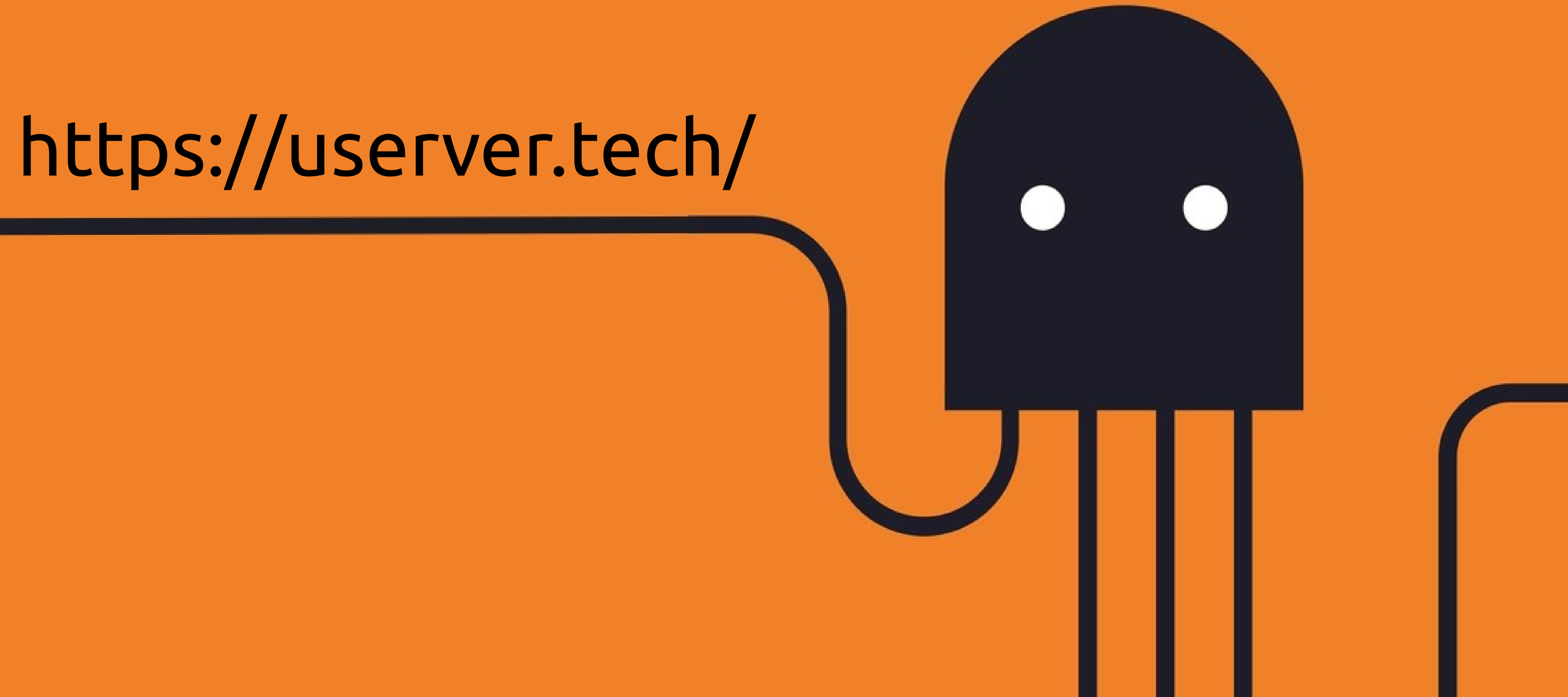
```
std::string get_str1();
std::string get_str2();
void do_something(const char* s);

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

Невыполнимая C++ задача:

У вас есть 10 минут на написание HTTP «Hello World»

https://userver.tech/



Hello world

```cpp
#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);
}
```

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);
}
```

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<7>::value];

C++ - это весело

- int array[Fib<7>::value];

- boost::pfr::get<1>(some_structure);

C++ - это весело

- int array[Fib<7>::value];

- boost::pfr::get<1>(some_structure);

- #define PP_ITERATIONS 5
  #define PP_ACTION /*...*/
  #include <pp_repeat.hpp>

А что будет «завтра»?

C++

- Экономия в количестве железа и его поддержке
- Экономия вашей батарейки в мобильном телефоне
- Меньше время отклика
- Предсказуемое время отклика
- Отсутствие vendor-lock
- Поддержка множества платформ
- Совместимость с C

Спасибо

Полухин Антон

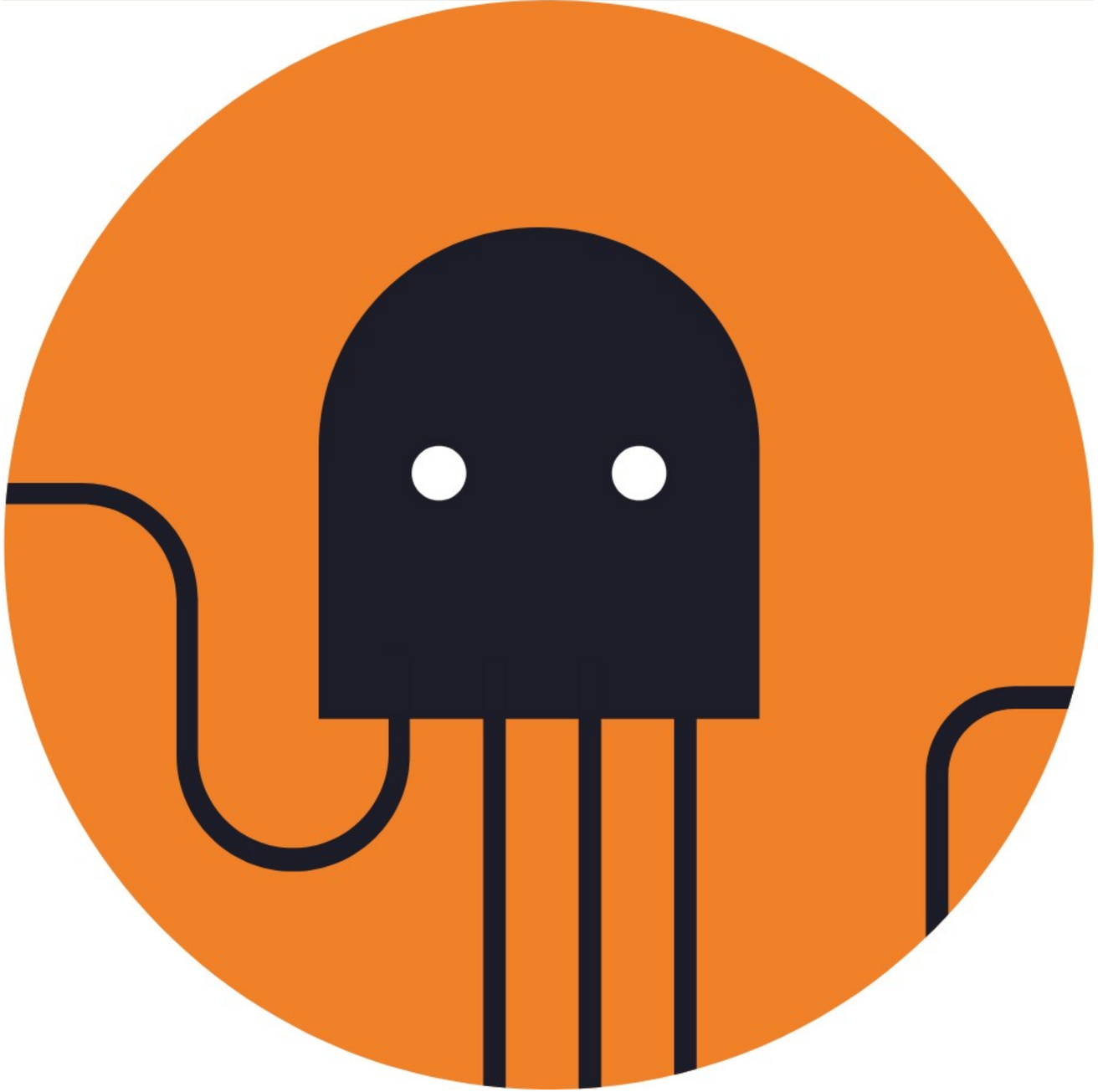
Эксперт-разработчик C++

antoshkka@gmail.com

antoshkka@yandex-team.ru

https://github.com/apolukhin

C++ РГ21 C++ РОССИЯ https://stdcpp.ru/



https://github.com/userver-framework

https://userver.tech/