Курс-интенсив

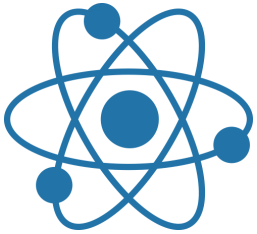
Программирование на C++

Основы языка

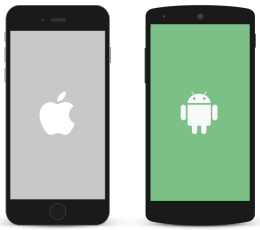
academy.rubius.com
konstantin.dobrychev@rubius.com
Константин Добрычев

Зачем C++?

Desktop



Science

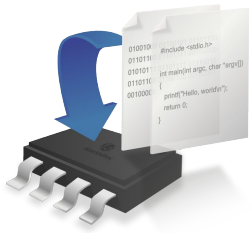


Mobile

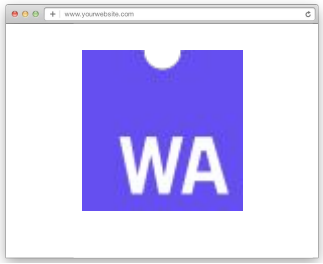
C++

Backend

GameDev



Embedded



Frontend

Немного истории

- 1980 - Си с классами
- 1985 - Cfront 1.0 (Название “C++”)
- 1998 - ISO/IEC 14882:1998 (C++98)
- 2003 - ISO/IEC 14882:2003 (C++03)
- 2011 - C++11
- 2014 - C++14
- 2017 - C++17

Немного идеологии

“you don't pay for what you don't use”

Немного кода

```
#include <iostream>

int main()
{
    std::cout << "Hello, World!" << "\n";
    return 0;
}
```

Текст программы

```
int the_first_value = 5; // значение утром
int the_second_value = 3; // значение вечером

/*
 Какой-то длинный текст,
 описывающий что делает эта операция и почему она так написана.
*/
int total = sum(the_first_value, the_second_value);

/**
 * @brief Возвращает результат сложения двух чисел.
 * @param x первое число
 * @param y второе число
 * @return сумма
 */
int sum(int x, int y)
{
    return x + y;
}
```

Ключевые слова C++

```
alignas
alignof
and
and_eq
asm
auto
bitand
bitor
bool
break
case
catch
char
char16_t
char32_t
class
compl
const
constexpr
const_cast
continue
decltype
default
delete
do
double
dynamic_cast
else
enum
explicit
export
extern
false
float
for
friend
goto
if
inline
int
long
mutable
namespace
new
noexcept
not
not_eq
nullptr
operator
or
or_eq
private
protected
public
register
reinterpret_cast
return
short
signed
sizeof
static
static_assert
static_cast
struct
switch
template
this
thread_local
throw
true
try
typedef
typeid
typename
union
unsigned
using
virtual
void
volatile
wchar_t
while
xor
xor_eq
```

Переменные

Переменная — поименованная, либо адресуемая иным способом область памяти, адрес которой можно использовать для осуществления доступа к данным. Данные, находящиеся в переменной (то есть по данному адресу памяти), называются **значением** этой переменной.

```
int a = 42;
const double x = 0.4;

std::string text = "Text";
text = "Mutable text";
```

Область видимости (англ. *scope*) — часть программы, в пределах которой идентификатор, объявленный как имя некоторой программной сущности, остаётся связанным с этой сущностью, то есть позволяет посредством себя обратиться к ней

```
int global_variable = 12;

int doSomething()
{
    int local_variable = 12;

    int a = -1;
    { int a = 0; }
    { int a = 1; }
}
```

Фундаментальные типы

short
int
long
long long

```
int a = 42;
long b = 42;
```

float
double
long double

```
float a = 42.0;
double b = 3.0E8;
```

char
wchar
char16_t
char32_t

```
char ch = 'a';
unsigned char chd = 'b';
```

signed
unsigned

```
signed short a = -100;
unsigned long b = 42;
auto c = 42UL;
```

bool

```
bool isRunnig = false;
bool isActive = true;
```

Фундаментальные типы

```
// для работы с сырой памятью
#include <cstddef>
std::byte

// числа фиксированного размера
#include <cstdint>
int8_t .. int64_t
uint8_t .. uint64_t

// индекс/размер чего-либо
#include <cstddef>
std::size_t
```

`void` - пустой тип

`std::nullptr_t` - тип `nullptr`

Константы и перечисления

```
// плохо
const double PI = 3.14;
static const double PI = 3.14;
#define PI 3.14

// хорошо
constexpr double PI = 3.14;
```

```
enum class Color {
    Blue,
    Green,
    Red,
    Yellow,
};

Color color = Color::Yellow;
```

```
enum Color {
    Blue,
    Green,
    Red,
    Yellow,
};

int color = Yellow;
```

```
enum Permissions : uint16_t {
    Execute = 0111,
    Write = 0222,
    Read = 0444,
};

int permissions = Read | Write;
```

Массивы и строки

```
int numbers[5] = { 2, 4, 6, 8, 10};
char word[] = {'h', 'e', 'l', 'l', 'o'};
```

```
#include <array>
std::array numbers = {2, 4, 6, 8, 10};
std::array<float, 2> values = {23, 45.01};
std::array<char, 0> empty_array = {};
```

```
const char *cstring = "Hello";
```

```
#include <string>
std::string string = "Hello";
```

Структуры и объединения

```
struct Location {
    std::string street;
    std::string country;
};

struct Person {
    std::string name;
    int age;
    Location location;
};
```

```
Person person = {
  "Alice", 25, { "Russia", "Moscow" }
};

person.age = 26;
person.location.city = "Tomsk";
```

```
union Value {
    int integer;
    double real;
    const char *string;
};

Value value;
value.integer = 42;
value.string = "text";
```

```
#include <variant>

std::variant<int, double, std::string>
value;
value = 42;
value = "text";
```

std::vector

```
std::vector<std::string> people = {
    "Alice",
    "Bob",
    "Carlos",
};

// вставка в конец
people.push_back("Eve");

// вставка в середину
people.insert(people.begin() + 3, "Dave");

// удаление
people.erase(people.begin() + 1);
```

std::map

```
std::map<Color, std::string> colors = {
    {Blue, "#0000FF"},
    {Green, "#00FF00"},
    {Red, "#FF0000"},
};

// чтение
std::cout << colors[Red] << "\n";

// модификация/вставка
colors[Yellow] = "#FFFF00";

// удаление
colors.erase(Yellow);
```

Ещё немного про типы

```
// вывод типа
auto a = 42;
auto b = 42ULL;
auto str = "text";

// определение типа
decltype(a) c = 0;

// определение размера
sizeof(int);
sizeof(a);

 // псевдонимы типов
 using BigInteger = long long;
 using StringList = std::vector<std::string>;
```

```
// приведение типа
static_cast
dynamic_cast
const_cast
reinterpret_cast

static_cast<std::size_t>(45);
```

Выражения и операторы

Выражение — это последовательность операторов и операндов, задающая вычисление. Выражение может иметь своим результатом значение и может вызывать побочные эффекты.

```
+a      a = b    ++a  !a      a == b  ~a      a[b]   a(...)
-a      a += b   --a  a && b  a != b  a & b   *a     a, b
a + b   a -= b   a++  a || b  a < b   a | b   &a     ? :
a - b   a *= b   a--          a > b   a ^ b   a->b
a * b   a /= b                a <= b  a << b  a.b
a / b   a %= b                a >= b  a >> b  a->*b
a % b   a &= b                                a.*b
        a |= b
        a ^= b
        a <<= b
        a >>= b
```

Ветвление: if

```
if (expression) {
    doSomething();
}
```

```
if (expression) {
    doSomething();
} else {
    doAnotherThing();
}
```

```
if (FileInfo info = open("data.txt"); !info.isOpen) {
    // …
} else if (info.isReadable) {
    // …
} else if (info.isWritable){
    // …
} else {
    // …
}
```

```
int min = (a < b) ? a : b;
```

Ветвление: switch

```
switch (Color color = getColor(); color) {
    case Blue:
       // …
       break;

    case Red:
    case Yellow:
        // …
        break;

    default:
        break;
}
```

Циклы

```
while (expression) {
    // …
}

do {
    // …
} while (expression);
```

```
for (int i = 0; i < MAX_SIZE; ++i) {
    // …
}

for (int i : {10, 20, 30, 40}) {
    // …
}
```

Функции

```
void printHello()
{
    std::cout << "Hello" << "\n";
}
```

```
int sum(int a, int b)
{
    return a + b;
}
```

```
int sum(int a, int b, int c)
{
    return sum(sum(a, b), c);
}
```

```
std::vector<int> sum(std::vector<int> a, std::vector<int> b)
{
    std::vector<int> result(std::min(a.size(), b.size()));

    for (int i = 0; i < result.size(); ++i) {
      result[i] = sum(a[i], b[i]);
    }

    return result;
}
```

Немного про файлы

Math.h

```
#pragma once

double sin(double x);

double cos(double x);

double tan(double x);

double exp(double x);
```

Math.cpp

```
#include “Math.h”

double sin(double x) {
     return ...
}

double cos(double x) {
     return ...
}

double tan(double x) {
     return ...
}

double exp(double x) {
     return ...
}
```

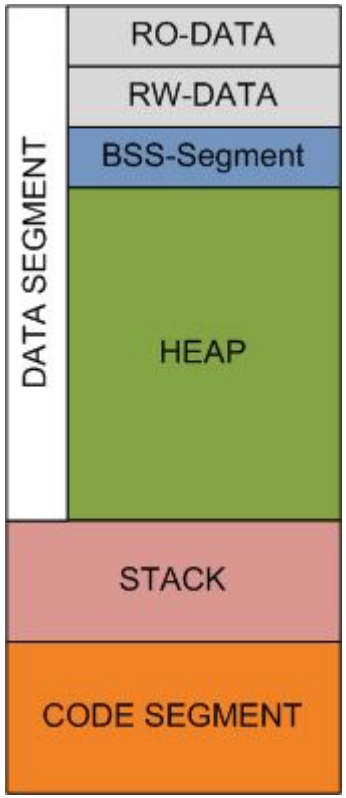
Указатели и ссылки

```
int a = 42;
int *p = &a;
int b = *p;
```

```
// плохо
int *p = 0;
int *p = NULL;

// хорошо
int *p = nullptr;
```

```
int &lvalue = a;
const int &lvalue = a;

int &&rvalue = getValue();
int &&rvalue = std::move(a);

std::string description(const Person &person);
```

Немного про память




```
const char *string = "Text";

static const std::vector = {2, 3, 5};

void doSomething()
{
    int a = 42;

    {
        Person person;
    }

    int *p = new int;
    delete p;

    int *array = new int[10];
    delete[] array;
}
```

Куда копать дальше?

https://cppreference.com/

https://coliru.stacked-crooked.com/

Курс-интенсив

Программирование на C++

Основы языка

academy.rubius.com
konstantin.dobrychev@rubius.com
Константин Добрычев