std::bind

29 июня 2017 г.

std::bind (C++11)

- универсальный «связыватель»
- создает новый объект-функцию с заданным набором аргументов
- использует плейсхолдеры (_1, _2 и т. п.) – показывают, какой по счету переданный аргумент нужно подставлять в функцию

std::bind(функция, плейсхолдеры_или_аргументы)(аргументы)

std::bind (пример 1)

```
#include <iostream>
#include <functional>

void printArgs(const int x, const int y, const int z) {
    std::cout << x << ' ' << y << ' ' << z << std::endl;
}

void f(){
    using namespace std::placeholders;
    auto func1 = std::bind(printArgs, _1, _2, _3);
    func1(10, 11, 12); // 10 11 12
    auto func2 = std::bind(printArgs, _1, _3, _2);
    func2(10, 11, 12); // 10 12 11
    auto func3 = std::bind(printArgs, _3, _2, _1);
    func3(10, 11, 12); // 12 11 10
}
```

std::bind (пример 2)

```
int sum(const int x, const int y) {
    return x + y;
}

void f(){
    auto f_sum1 = std::bind(sum,  3, std::placeholders::_2);
    std::cout << f_sum1(5, 7) << std::endl;

    auto f_sum2 = std::bind(sum, std::placeholders::_1, 10);
    std::cout << f_sum2(5, 7) << std::endl;

    auto f_sum3 = std::bind(sum, std::placeholders::_1,
                                 std::placeholders::_1);
    std::cout << f_sum3(2) << std::endl;
}
```

std::bind (пример 2)

```
int sum(const int x, const int y) {
    return x + y;
}

void f(){
    auto f_sum1 = std::bind(sum,  3, std::placeholders::_2);
    std::cout << f_sum1(5, 7) << std::endl;    // 10

    auto f_sum2 = std::bind(sum, std::placeholders::_1, 10);
    std::cout << f_sum2(5, 7) << std::endl;    // 15

    auto f_sum3 = std::bind(sum, std::placeholders::_1,
                                 std::placeholders::_1);
    std::cout << f_sum3(2) << std::endl;       // 4
}
```

std::bind (пример 3)

```
int sum(const int x, const int y) {
    return x + y;
}

void f(){
    auto f_sum4 = std::bind(sum, std::placeholders::_4,
                                        std::placeholders::_5);
    std::cout << f_sum4(1, 2, 3, 4, 5) << std::endl;

    auto f_sum5 = std::bind(sum, std::placeholders::_1,
                                std::placeholders::_2);
    auto f_sum6 = std::bind(sum, std::placeholders::_3,
                                std::placeholders::_4);
    auto f_sum7 = std::bind(sum, f_sum5, f_sum6);
    std::cout << f_sum7(20, 30, 40, 50) << std::endl;
}
```

std::bind (пример 3)

```
int sum(const int x, const int y) {
    return x + y;
}

void f(){
    auto f_sum4 = std::bind(sum, std::placeholders::_4,
                                 std::placeholders::_5);
    std::cout << f_sum4(1, 2, 3, 4, 5) << std::endl;    // 9

    auto f_sum5 = std::bind(sum, std::placeholders::_1,
                           std::placeholders::_2);
    auto f_sum6 = std::bind(sum, std::placeholders::_3,
                           std::placeholders::_4);
    auto f_sum7 = std::bind(sum, f_sum5, f_sum6);
    std::cout << f_sum7(20, 30, 40, 50) << std::endl;    // 140
}
```

std::bind (пример 4)

```
void f(){
    using namespace std::placeholders;
    std::vector<double> vec = {1, 2, 3, 4, 5, 6, 7, 8, 9};
    std::replace_if(vec.begin(), vec.end(),
                    (std::bind(std::less<double>(), _1, 5)), 0);
    for (auto value : vec) { std::cout  << value << " "; }
}
```

```
void f(){
    using namespace std::placeholders;
    std::vector<double> vec = {1, 2, 3, 4, 5, 6, 7, 8, 9};
    std::replace_if(vec.begin(), vec.end(),
                    (std::bind(std::less<double>(), 5, _1)), 0);
    for (auto value : vec) { std::cout  << value << " "; }
}
```

Вопросы?