

Приведение типов

Гусев Илья

Московский физико-технический институт

Москва, 2017

1 Приведение типов

- `static_cast`
- `const_cast`
- `dynamic_cast`
- `reinterpret_cast`
- C-style cast

static_cast

Можно:

- 1 Преобразование указателя на родительский класс к указателю на дочерний класс. Объект по указателю обязан быть правильного дочернего класса, иначе `undefined behaviour`.
- 2 Преобразования между числовыми типами. `int`, `long`, `char`, `unsigned int` — все их можно кастить друг в друга при помощи `static_cast`.
- 3 Можно закастить любое выражение в `void` (ради побочных эффектов).
- 4 Можно привести константу `nullptr` к любому типу-указателю.

Нельзя:

- 1 Преобразование между указателями на в принципе несовместимые типы. Например, указатель на `double` нельзя привести к указателю на `int`.
- 2 Указатели на типы, а также сами типы с несовместимыми атрибутами `const` и/или `volatile`.
- 3 Привести указатель на функцию-член к указателю на обычную функцию, или указатель на код к указателю на данные.

static_cast - примеры

// Пример числовых преобразований.

```
double a = 1.0;  
int b = static_cast<int>(a);
```

// Пример каста вверх по иерархии.

```
class A {};  
class B: public A{};  
B b;  
A* a = static_cast<A*>(&b);
```

// Пример попытки снять константность.

```
const int a = 1;  
int b = static_cast<int>(a); // Ошибка при компиляции!
```

const_cast

- 1 Добавляет или убирает модификатор const/volatile у переменной.
- 2 Если вы изначально определили переменную как const, нельзя на ней использовать const_cast, иначе UB!
- 3 Используется для вызовов правильных перегрузок, например.
- 4 Добавить const к неконстантному объекту могут также static_cast и dynamic_cast.

const_cast - примеры

```
int f(const int* a) {  
    int* b = const_cast<int*>( a );  
    (*b)++;  
    return *b;  
}
```

```
const int c = 1;  
int i = 1;
```

```
f(c); // UB.  
f(i); // Корректно.
```

dynamic_cast

- 1 Нужен для полиморфизма, используется только с классами с виртуальными функциями или виртуальным наследованием.
- 2 Если преобразование по иерархии действительно возможно (подходящий изначальный объект), оно произойдёт.
- 3 Если преобразование невозможно, и преобразовывались указатели, `dynamic_cast` вернёт 0 или `nullptr`.
- 4 Если преобразование невозможно, и преобразовывались ссылки, `dynamic_cast` кинет исключение `std::bad_cast`.

dynamic_cast - примеры

```
struct A {  
    virtual ~A() {}  
};  
struct B : A {  
    int b = 1;  
};  
  
A a;  
B b;  
A* dynamic_b_a = dynamic_cast<A*>(&b); // OK.  
A* static_b_a = static_cast<A*>(&b); // OK.  
B* dynamic_a_b = dynamic_cast<B*>(&a); // 0.  
B* static_a_b = static_cast<B*>(&a); // UB.  
static_a_b->b; // Mycop.
```


reinterpret_cast

- 1 Самое опасное преобразование, используется для грязных хаков.
- 2 Напрямую превращает один тип в другой.
- 3 Ограничения: только pointer-to-pointer, reference-to-reference, pointer-to-integer, integer-to-pointer преобразования.
- 4 Гарантирует, что если преобразовать сначала в одну сторону, а потом обратно, то получится то же значение. Исключение - если размер типа, в который преобразовываем, меньше исходного.

reinterpret_cast

```
int a = 1;
// Верхние 4 байта - мусор.
double b = *reinterpret_cast<double*>(&a); // Случайное значение.

if (sizeof(float) == sizeof(int)) {
    int a = 1;
    // Конкретное значение зависит от платформы.
    // Размер типов, порядок байтов.
    float b = *reinterpret_cast<float*>(&a); // 1.4013e-45
}
```

C-style cast

Порядок вызова:

- 1 `const_cast`.
- 2 Если `const_cast` не может дать нужный результат, то `static_cast` (с игнорированием спецификатора приватного наследования).
- 3 Если и так не выходит, то компилятор пробует в хвост к `static_cast` добавить `const_cast`.
- 4 Если и это не получается, то `reinterpret_cast`.
- 5 Если не выйдет, то к нему дописывается `const_cast`.

Полезные ссылки I



StackOverflow: `static_cast`

<https://ru.stackoverflow.com/questions/387985/Для-чего-нужен-static-cast-как-он-работает-и-где-его-применяют>



StackOverflow: When should `static_cast`, `dynamic_cast`, `const_cast` and `reinterpret_cast` be used?

<https://stackoverflow.com/questions/332030/when-should-static-cast-dynamic-cast-const-cast-and-reinterpret-cast-be-used>