Программирование на языке C++
Лекция 3

Константность

Александр Смаль

Определение констант

- Ключевое слово `const` позволяет определять типизированные константы.

```
double const pi = 3.1415926535;
int const day_seconds = 24 * 60 * 60;
// массив констант
int const days[12] = {31, 28, 31,
                      30, 31, 30,
                      31, 31, 30,
                      31, 30, 31};
```

- Попытка изменить константные данные приводит к неопределённому поведению.

```
    int * may = (int *) &days[4];
    *may = 30;
```

Указатели и const

В C++ можно определить как константный указатель, так и указатель на константу:

```
int a = 10;
const int * p1 = &a; // указатель на константу
int const * p2 = &a; // указатель на константу
*p1 = 20; // ошибка
p2  = 0;  // OK

int * const p3 = &a; // константный указатель
*p3 = 30; // OK
p3  = 0;  // ошибка

// константный указатель на константу
int const * const p4 = &a;
*p4 = 30; // ошибка
p4  = 0;  // ошибка
```

Указатели и const

Можно использовать следующее правило:

- “слово const делает неизменяемым тип слева от него”.

```
int a = 10;
int * p = &a;

// указатель на указатель на const int
int const ** p1 = &p;

// указатель на константный указатель на int
int * const * p2 = &p;

// константный указатель на указатель на int
int ** const p3 = &p;
```

Ссылки и const

- Ссылка сама по себе является неизменяемой.

```
int a = 10;
int & const b = a; // ошибка
int const & c = a; // ссылка на константу
```

- Использование константных ссылок позволяет избежать копирования объектов при передаче в функцию.

```
Point midpoint(Segment const & s);
```

- По константной ссылке можно передавать rvalue.

```
Point p = midpoint(Segment(Point(0,0),
                           Point(1,1)));
```

Константные методы

- Методы классов могут быть объявлены как `const`.

```
struct IntArray {
    size_t size() const;
};
```

IntArray const *

- Такие методы не могут менять поля объекта (тип `this` — указатель на `const`).
- У константных объектов (через указатель или ссылку на константу) можно вызывать только константные методы:

```
IntArray const * p = foo();
p->resize(); // ошибка
```

- Внутри константных методов можно вызывать только константные методы.

Две версии одного метода

- Слово `const` является частью сигнатуры метода.

```
size_t IntArray::size() const {return size_;}
```

- Можно определить две версии одного метода:

```
struct IntArray {
    int get(size_t i) const {
        return data_[i];
    }
    int & get(size_t i) {
        return data_[i];
    }
private:
    size_t size_;
    int *  data_;
};
```

Синтаксическая и логическая константность

- Синтаксическая константность: константные методы не могут менять поля (обеспечивается компилятором).
- Логическая константность — нельзя менять те данные, которые определяют состояние объекта.

```
struct IntArray {
    void foo() const {
        // нарушение логической константности
        data_[10] = 1;
    }
private:
    size_t size_;
    int *  data_;
};
```

int * const

Ключевое слово mutable

Ключевое слово `mutable` позволяет определять поля, которые можно изменять внутри константных методов:

```
struct IntArray {
  size_t size() const {
      ++counter_;
      return size_;
    }

private:
    size_t size_;
    int *  data_;

    mutable size_t counter_;
};
```