

Программирование на языке C++

Лекция 5

Ключевое слово `friend`

Александр Смаль

Дружественные классы

```
➔ struct String {  
    ...  
    ➔ friend struct StringBuffer;  
private:  
    ➔ char * data_;  
    size_t len_;  
};  
  
➔ struct StringBuffer {  
    ➔ void append(String const& s) {  
        ➔ append(s.data_);  
    }  
    ➔ void append(char const* s) {...}  
    ...  
};
```

Дружественные функции

Дружественные функции можно определять прямо внутри описания класса (они становятся inline).

```
struct String {  
    ...  
    friend std::ostream&  
        operator<<(std::ostream & os,  
                    StringBuffer const& sb)  
    {  
        return os << sb.data_;  
    }  
  
private:  
    char * data_  
    size_t len_  
};
```

Дружественные методы

```
→ struct String;
→ struct StringBuffer {
    → void append(String const& s); ←
        void append(char const* s) {...}
    ...
};

→ struct String {
    ...
    friend
        void StringBuffer::append(String const& s);
};

→ inline void StringBuffer::append(String const& s) {
    append(s.data_);
}
```

Отношение дружбы

Отношение дружбы можно охарактеризовать следующими утверждениями:

- Отношение дружбы не симметрично. A - друг B \nRightarrow B - друг A
- Отношение дружбы не транзитивно. A - друг B, B - друг C \nRightarrow A - друг C
- Отношение наследования не задаёт отношение дружбы.
- Отношение дружбы сильнее, чем отношение наследования.

Вывод

Стоит избегать ключевого слова `friend`, так как оно нарушает инкапсуляцию.