

# Программирование на современном C++

Основы обобщенного программирования.



# Отметьтесь на портале!

- Посещение необязательное, но тем, кто пришёл, следует отмечаться на портале в начале каждого занятия
- Это позволяет нам анализировать, какие занятия были более или менее интересны студентам, и менять курс в лучшую сторону
- Также это даст возможность вам оставить обратную связь по занятию после его завершения


пн, 1 ноября	вт, 2 ноября	ср, 3 ноября	чт, 4 ноября	пт, 5 ноября	сб, 6 ноября
Нет занятий	<div>18:00 Углубленный C/C++ (w... <span>п</span></div> <div>Обработка исключительных ситуаций. Шаблоны классов и методов. Обобщенное и безопасное программирование</div> <div>А. Халайджи</div> <div>18:00 Углубленный C/C++ (ML) <span>п</span></div> <div>Обработка исключительных ситуаций. Шаблоны классов и методов. Обобщенное и безопасное программирование</div> <div>А. Халайджи</div>	Нет занятий	Нет занятий	Нет занятий	Нет занятий

Ссылка на Zoom РК сегодня в 18:00

Безопасность интернет-приложений (третий семестр)

Подключиться к конференции Zoom  
<https://mailru.zoom.us/j/98586081818?pwd=eWxwSDdCbHhQNnNwQU5iblFvU2dTZz09>

Идентификатор конференции: 985 8608 1818  
Код доступа: 785885

 Алексей Набережный 55 минут назад

★ 0 💬 0 ↓ 0 ↑

Запись прошлой лекции (+ что почитать перед РК)

Безопасность интернет-приложений (третий семестр)

### Углублённый C/C++

Лекция 5

📍 Онлайн - ML

Отметьтесь, что вы пришли на занятие. Так вы улучшите свою посещаемость и вас увидит преподаватель в своём "Журнале посещений".

Отметиться

Оставьте отзыв о занятии и мы сможем улучшить учебный процесс.

Оставить отзыв

# План лекции

- История возникновения шаблонов
- Виды шаблонов
- Параметры шаблонов
- Перерыв
- Специализация шаблонов
- Статический полиморфизм



Бьёрн Страуструп

автор языка C++

## A History of C++

“For many people, the largest single problem using C++ is the lack of an extensive standard library. A major problem in producing such a library is that C++ does not provide a sufficiently general facility for defining “container classes” such as lists, vectors, and associative arrays.”

# Виды шаблонов

- Шаблон класса

# Виды шаблонов

- Шаблон класса
- Шаблон функции

# Виды шаблонов

- Шаблон класса
- Шаблон функции
- Шаблон псевдонима (C++11)

# Виды шаблонов

- Шаблон класса
- Шаблон функции
- Шаблон псевдонима (C++11)
- Шаблон переменной (C++14)



# Виды шаблонов

- Шаблон класса
- Шаблон функции
- Шаблон псевдонима (C++11) `using It = std::vector<int>::iterator` (аналог `typedef`)
- Шаблон переменной (C++14)
- Концепты (C++20)

# Виды шаблонов

Пример

# Итог

```
template<typename T, typename S>  
class SomeClass;
```

```
template<typename T>  
void SomeFunction();
```

```
template<typename T>  
using MyAlias = typename std::vector<T>::iterator;
```

```
template<typename T>  
inline constexpr T SomeVar = 10;
```

 пишется в заголовочных файлах

# Параметры шаблонов

- Параметра шаблона тип

# Параметры шаблонов

- Параметра шаблона тип
- Параметра шаблона не-тип

# Параметры шаблонов

- Параметра шаблона тип
- Параметра шаблона не-тип
- Параметр шаблона шаблон

# Параметры шаблонов

Пример

# Iterator

```
template<typename T>
class G;

enum class Visibility { VISIBLE, INVISIBLE };

template<int G, Visibility V>
void some_function();

template<template<typename> typename Hash>
class MyHashTable;
```



# Перерыв

# Специализация шаблонов

Полная специализация шаблона

Частичная специализация шаблона

# Специализация шаблонов

Пример

# Итог

```
template<>    ПОЛНАЯ СПЕЦИАЛИЗАЦИЯ  
void f<int>();
```

```
template<typename T>    ЧАСТИЧНАЯ СПЕЦИАЛИЗАЦИЯ (ОПРЕДЕЛЕНА КАКИЕ-ЛИБО ТИПЫ, НО  
class MyClass<T, int>; ПАРАМЕТРЫ ШАБЛОНОВ ВСЕ РАВНО ЕСТЬ)
```

# Статический полиморфизм

- Выбор функции происходит на этапе компиляции
- Семантика значения

# Статический полиморфизм

- Выбор функции происходит на этапе компиляции

# Статический полиморфизм

- Выбор функции происходит на этапе компиляции
- Семантика значения
- Сложный для понимания код

# Статический полиморфизм

- Выбор функции происходит на этапе компиляции
- Семантика значения
- Сложный для понимания код
- Сложное описание требований к типу



# Статический полиморфизм

- Выбор функции происходит на этапе компиляции
- Семантика значения при динамическом полиморфизме все передается по указателю и
- Ссылкам то есть нужна доп логика для копирования классов-наследника, ее
- Сложный для понимания код нужно реализовывать во всех потомках
- Сложное описание требований к типу
- Увеличение времени компиляции

# Статический полиморфизм

- Выбор функции происходит на этапе компиляции
- Семантика значения
- Сложный для понимания код <sup>детали реализации должны быть вместе с объявлением</sup>
- Сложное описание требований к типу
- Увеличение времени компиляции
- Увеличение размера кода

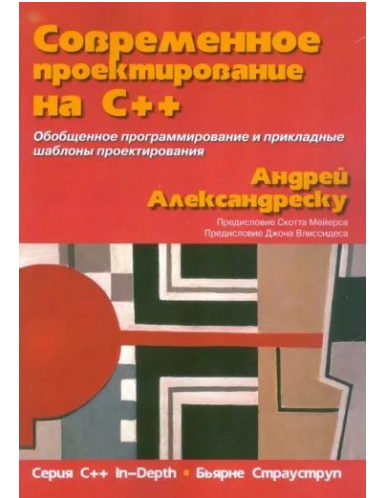
+одинаковые сущности становятся абсолютно разными типами при конкретизации хотя бы одним отличающимся шаблоном

(пример: вектора с 2-мя разными аллокаторами - 2 разных типа с точки зрения компилятора)

непонятно, что подставлять в шаблоны, ошибка возникает только при прокидывании подставленных конкретных параметров в шаблон то есть, как правило, в глубинах реализации. РЕШЕНИЕ - КОНЦЕПТЫ(ТРЕБОВАНИЯ К ТИПУ)

# Литература

- Дэвид Вандевурд  
Шаблоны C++. Справочник разработчика
- **Осторожно!**  
Андрей Александреску  
Современное проектирование на C++



# Спасибо за внимание!

