

# C++ developer. Basic

## Структуры и классы



Проверить, идет ли запись

# Меня хорошо видно && слышно?



Ставим "+", если все хорошо  
"-", если есть проблемы

Тема вебинара

# Структуры и классы



**Карина Дорожкина**

**Research Development Team Lead**

Более 10 лет опыта разработки на C/C++.

Долгое время занималась развитием ПО в области безопасности транспортного сектора.

Имею опыт руководства несколькими командами и проектами с разнообразным стеком технологий.

Спикер конференций C++ Russia, escar Europe.

**[dorozhkinak@gmail.com](mailto:dorozhkinak@gmail.com)**

# План вебинара



# Мультипарадигменный C++

Как можем писать на C++?

- процедурное программирование
- **объектно-ориентированное программирование**
- обобщенное программирование
- метапрограммирование



# Структура

- пользовательский тип данных
- реализует принцип абстракции
- хранит поля

```
1 struct Person {  
2     std::string name;  
3     int age;  
4 };
```



# Перегрузка операторов

<https://en.cppreference.com/w/cpp/language/operators>

```
std::ostream& operator<< (std::ostream& os, const Person& person)
{
    os << person.name << " " << person.surname << " " << person.age;
    return os;
}
```



# Класс

Класс определяет данные и методы работы с ними

```
1 class Point {  
2 private:  
3     int x;  
4     int y;  
5 public:  
6     std::string toString() const;  
7 };
```





# Конструктор

Это особая функция класса, которая вызывается при создании объекта этого класса

```
1 class Point {  
2     public:  
3         Point(int x_, int y_) {  
4             x = x_;  
5             y = y_;  
6         }  
7     private:  
8         int x;  
9         int y;  
10 };
```



# Деструктор и сотоварищи

Вызывается

- деструктор - при разрушении объекта
- конструктор копирования - при создании копии другого объекта
- оператор присваивания - при присваивании нового значения существующему объекту



```
1 class String {  
2     ~String();  
3     String(const String& other);  
4     String& operator=(const String& other);  
5 };
```



# Наследование

```
1 class Shape {  
2 public:  
3     Color GetColor();  
4  
5 private:  
6     Color color;  
7 };
```

```
1 struct Square: public Shape {  
2 public:  
3     double Square() const {  
4         return length * length;  
5     }  
6 private:  
7     double length;  
8 };
```



# Подводим итоги



**Заполните, пожалуйста,  
опрос о занятии  
по ссылке в чате**

# Вопросы?

Слушаю/читаю в чате