

Онлайн образование

otus.ru



Проверить, идет ли запись

Меня хорошо видно && слышно?



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



Тема вебинара

Полиморфизм и все-все-все



Андрей Рыжиков



@ryzhikovas



План вебинара



Спецификаторы доступа



```
class Shape {  
public:  
    Color getColor() const;  
protected:  
    size_t getId() const;  
private:  
    Color color;  
    size_t id;  
};
```

```
class Protected: protected Shape {  
    void foo() const {  
        getColor();  
        getId();  
        color;  
    }  
};  
//...  
Protected child;  
child.getColor();
```



Спецификаторы доступа

```
class Shape {  
public:  
    Color getColor() const;  
protected:  
    size_t getId() const;  
private:  
    Color color;  
    size_t id;  
};
```

```
class Private: private Shape {  
    void foo() const {  
        getColor();  
        getId();  
        color;  
    }  
};  
//...  
Private child;  
child.getColor();
```



Спецификаторы доступа



```
class Shape {  
public:  
    Color getColor() const;  
    //...  
};
```

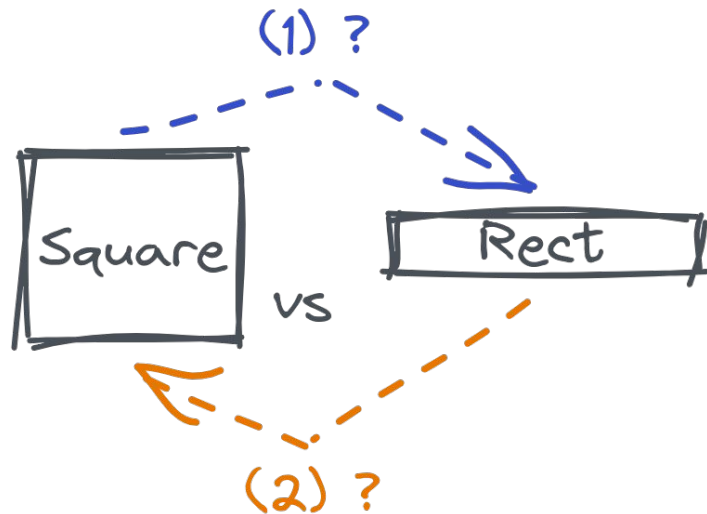
```
class Grandchild: Private {  
public:  
    void bar() const;  
        getColor();  
};
```

```
class Private: private Shape {  
    void foo() const {  
        getColor();  
        getId();  
        color;  
    }  
};  
//...  
Private child;  
child.getColor();
```



Принцип подстановки Барбары Лисков

```
class Rect: public Square{  
    //...  
};  
//или  
class Square: public Rect {  
    //...  
};
```



Наследование и переиспользование

```
struct Shape {  
    Color getColor() const;  
    const char* name() const {  
        return "shape";  
    }  
};  
  
struct Square: Shape {  
    const char* name() const {  
        return "square";  
    }  
};
```

В душе я `class` с `public`-спецификатором!
Но места на слайде хватило только на `struct`

+ уже можем переиспользовать код:

```
mixColor(const Shape& lhs,  
         const Shape& rhs);  
  
Shape shape/*...*/;  
Square square/*...*/;  
mixColor(square, square);  
mixColor(shape, square);  
//...
```

- Но пока не умеем менять поведение



Полиморфизм

```
struct Shape {  
    virtual  
    const char* name() const {  
        return "shape";  
    }  
};  
  
struct Square: Shape {  
    const char* name() const {  
        return "square";  
    }  
};
```

- разрешаем изменить поведение в наследниках
... обязаны сохранить сигнатуру
- это и есть полиморфизм
- помним про принцип Барбары Лисков
- **и виртуальный деструктор!**



Абстрактный класс

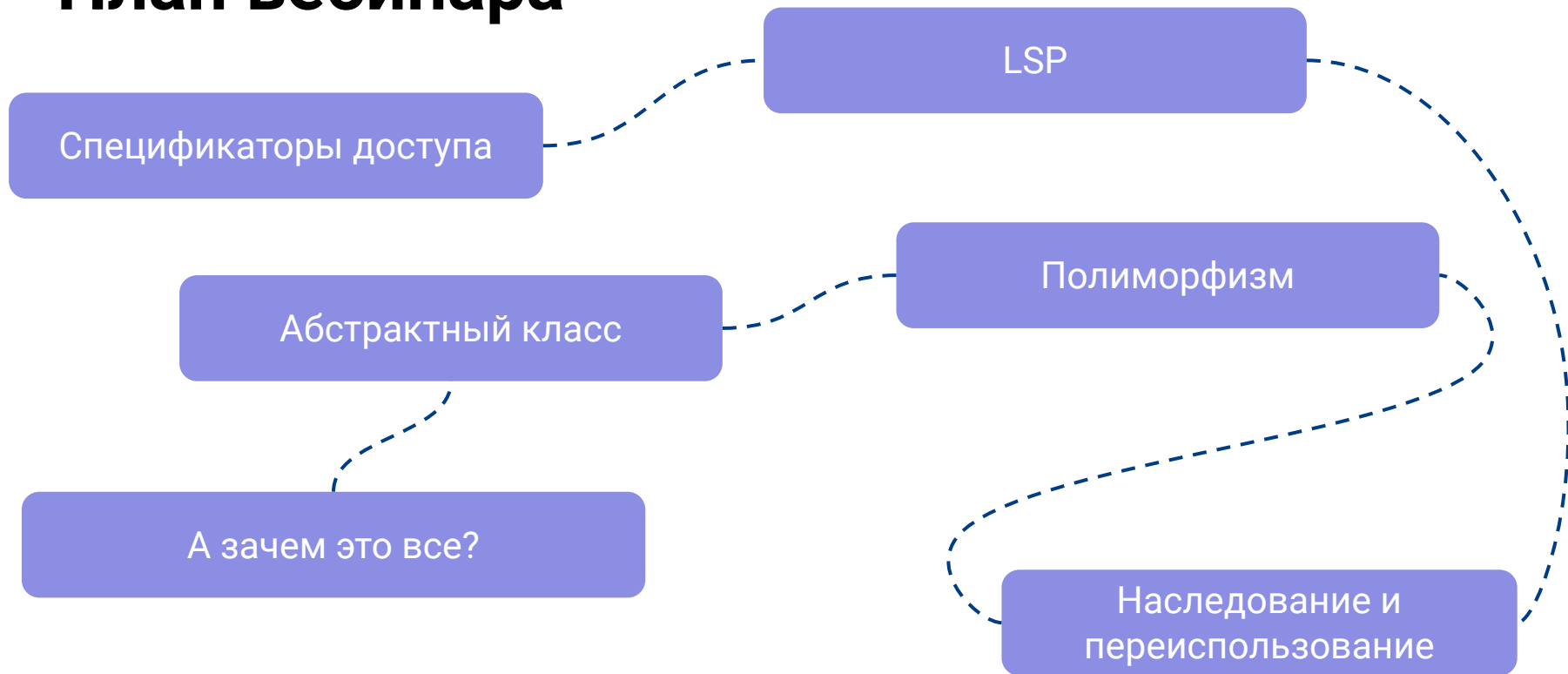
```
struct Shape {  
    virtual  
    const char* name() const = 0;  
};  
  
struct Square: Shape {  
    const char* name() const {  
        return "square";  
    }  
};
```

Я чисто виртуальный

Если есть хоть один чисто виртуальный метод,
то нельзя создать объект класса



План вебинара



Заполните, пожалуйста,
опрос о занятии
по [ссылке](#) в чате

Заключение

Вопросы?

Читаю в чате

