

# Лямбды

Профессия Java-разработчик на Hexlet

Преподаватель: Яковлев Егор

# Вопросы к лекции

- Почему у Stream API такой странный аргумент со стрелочкой (->)?
- Лямбда-функция - метод или аргумент?
- Какие бывают виды лямбда-функций?

# План

1. Где мы встречали лямбда функции?
2. Лямбда-функции
3. Структура лямбда-функций
4. Method reference
5. Лямбда в Java: in deer
6. Виды функциональных интерфейсов

# Где мы встречали лямбда функции?

Stream API!

```
list.stream()  
    .map (x -> Math.abs(x)) // 1  
    .filter(x -> x > 10) // 2  
    .forEach(x -> System.out.println(x)); // 3
```

$x \rightarrow x > 10$  - лямбда-функция

# Лямбда-функции

- Лямбда-функции - это функции, у которой фактически нет имени. (анонимная функция)
- Лямбда-выражение представляет собой блок кода, который можно передать в другое место, поэтому он может быть выполнен позже, один или несколько раз.

# Лямбда-функции

И в чём смысл?

- пишем метод и сразу используем его
- зачем создавать класс и метод, если можно вызвать функцию прямо сейчас?!
- передаём лямбда-функцию как аргумент в методы

# Структура лямбда-функций

- лямбда-функция может иметь 0 и более входных параметров
- тип параметров можно указать явно или получить из контекста
- параметры ( > 1) заключаются в круглые скобки и указываются через запятые

```
// берём модуль числа  
.map(x -> Math.abs(x)) // 1  
// считаем сумму элементов стрима  
.reduce((x, y) -> x + y) // 2
```

# Структура лямбда-функций

- если параметров нет - используем круглые скобки
- тело лямбда-выражения может содержать сколько угодно выражений
- если лямбда-функции содержать только одно выражение - ключевое слово `return` опускаем

```
// если стрим остался пустым - создаём нового пользователя  
.orElse(() -> new User()) // 1
```



# Структура лямбда-функций

```
// определяем время года
.map(x -> {
    if ((x > 0 && x < 2) || (x == 11)) {
        return "Зима";
    } else if (x > 1 && x < 5) {
        return "Весна";
    } else if (x > 4 && x < 8) {
        return "Лето";
    } else if (x > 7 && x < 11) {
        return "Осень"
    }
    return null;
}))
```

# Method reference

```
.forEach(System.out::println) // method reference
```

```
.map(User::toString) // method reference too
```

```
.filter(User::isValid) // method reference too
```

## Демо

# Лямбда в Java: in deer

Лямбда-выражение не выполняется само по себе, а образует реализацию метода, определенного в функциональном интерфейсе.

Функциональный интерфейс - интерфейс, который содержит единственный метод без реализации

# Виды функциональных интерфейсов:

- `Consumer<T>` - принимает один параметр `T` и не возвращает никакого значения
- `Supplier<R>` - не принимает параметров, возвращает значение `R`
- `Function<T, R>` - принимает один параметр `T` и возвращает результат `R`
- `Predicate<T>` - принимает параметр `T` и возвращает значение типа `boolean`

# Домашнее задание

```
hexlet program download java lambdas
```

```
hexlet program submit java lambdas
```

# Вопросы?