Функциональное программирование
в Java. Часть 1

Skillbox

Лямбда-выражения

Лямбда-выражения

В предыдущем видео вы узнали про функциональный интерфейс **BiFunction**, который позволяет совершать действия над двумя объектами. Посмотрите на него ещё раз.

```
 int r = calculate(a, b, new BiFunction<Integer, Integer, Integer>()
{
 @Override
 public Integer apply(Integer integer, Integer integer2) {
   return a + b;
 }
```

Если посмотреть внимательнее, то в этом примере всё, кроме одной строчки, избыточно. За содержательную часть (логику работы) отвечает только одно выражение return a + b, а всё остальное — код, который обеспечивает работу этого выражения. И вам пришлось написать достаточно много этого кода, даже чтобы просто передать методу код сложения двух чисел.

Изображение: Даниил Пилипенко

Лямбда-выражения

Лямбда строится так: (параметры) → (код метода).
А ваша лямбда будет такой:

BiFunction<Integer, Integer> biFunction = (a, b) → a + b

Этот блок из 10 символов можно передавать как аргумент методу, ожидающему функциональный интерфейс в качестве параметра. Причём чаще всего обходятся без промежуточной переменной — передают напрямую лямбду:

```
calculate(a, b, (integer, integer2) -> integer + integer2);
```

Изображение: Даниил Пилипенко

Лямбда-выражения

Компилятор проверит, что лямбда подходит функциональному интерфейсу — принимает нужное число параметров нужного типа. Например, такой вот вызов метода не скомпилируется, потому что передан всего один параметр:

```
calculate(a, b, (integer) -> integer + integer2);
```

Изображение: Даниил Пилипенко

Лямбда-выражения. Применение

Лямбды применяются во многих случаях, один из них — это **обход коллекции в цикле**.

```
List<Integer> integers = Arrays.asList(1, 2, 3, 4, 5);
integers.forEach(item -> System.out.println(item));
```

Изображение: Даниил Пилипенко

Лямбда-выражения. Применение

Если лямбда-выражения вызывают только один существующий метод, лучше ссылаться на этот метод по его имени. Для методов, у которых уже есть имя, можно применить ссылки на методы — это компактные лямбда-выражения, в которых можно не указывать параметры методов. Например, вот такое выражение:

```
Consumer<String> consumer = str -> System.out.println(str);
```

Вы можете переписать вот так:

```
Consumer<String> consumer = System.out::println;
```

Изображения: Даниил Пилипенко

Лямбда-выражения. Применение

Также можно написать ссылку на статический метод класса:

```
public static void main(String[] args) {
 Consumer<String> consumer = Main::print;
 consumer.accept("Hello!");
}

private static void print(String a) {
 System.out.println(a);
}
```

Изображение: Даниил Пилипенко

Лямбда-выражения. Применение

Кроме того, можно написать ссылку на создание нового объекта:

```
Supplier<String> supplier = String::new;
System.out.println(supplier.get());
```

Изображение: Даниил Пилипенко

Выводы

В этом видео вы научились создавать лямбда-выражения на основе анонимных классов, а также писать ссылки на методы.

Лямбда-выражения — это мощный инструмент, который может существенно помочь в разработке и написании программ.