Функциональное программирование
в Java. Часть 1

Skillbox

Функциональные интерфейсы

Функциональные интерфейсы

Функциональный интерфейс — это интерфейс, который содержит ровно один абстрактный метод, то есть описание метода без тела. Статические методы и методы по умолчанию при этом не в счёт, их в функциональном интерфейсе может быть сколько угодно.

Когда параметром метода является функциональный интерфейс, при вызове этого метода одним из аргументов должен быть блок кода.

Функциональный интерфейс

Написать функциональный интерфейс легко — достаточно создать интерфейс с одним методом и пометить его аннотацией **@FunctionalInterface**. Аннотацию писать необязательно, это пометка для компилятора.

Компилятор выбросит ошибку, если интерфейс имеет эту аннотацию, но не отвечает требованиям функциональных интерфейсов.

```
@FunctionalInterface
public interface Action<T extends Number> {

 T calculate(T t1, T t2);
}
```

Изображение: Даниил Пилипенко

Примеры функциональных интерфейсов в JDK. Function

Однако на практике вам чаще всего придётся пользоваться уже готовыми функциональными интерфейсами, реализованными в JDK. Поэтому очень важно знать об их назначении и особенностях.

В Java на данный момент существует четыре основных функциональных интерфейса и множество производных от них. Самый распространённый — это **Function**. Он содержит единственный метод apply, который принимает один аргумент и возвращает значение.

```
@FunctionalInterface
public interface Function<T, R> {

    /**
     * Applies this function to the given argument.
     *
     * @param t the function argument
     * @return the function result
     */
    R apply(T t);

}
```

Изображение: Даниил Пилипенко

Примеры функциональных интерфейсов в JDK. Supplier

Следующий пример — это **Supplier**. Метод get возвращает значение, но не принимает каких-либо параметров.

```
@FunctionalInterface
public interface Supplier<T> {

    /**
     * Gets a result.
     *
     * @return a result
     */
    T get();
}
```

Изображение: Даниил Пилипенко

Примеры функциональных интерфейсов в JDK. Consumer

Интерфейс **Consumer** работает другим образом — его метод accept принимает аргумент, производит с ним какие-либо действия, но не возвращает никакого результата. С помощью Consumer, например, можно сделать вывод в консоль.

```
@FunctionalInterface
public interface Consumer<T> {

 /**
    * Performs this operation on the given argument.
    *
    * @param t the input argument
    */
   void accept(T t);

}
```

Изображение: Даниил Пилипенко

Примеры функциональных интерфейсов в JDK. Predicate

И, наконец, последний из основных функциональных интерфейсов — **Predicate**. Он реализует функцию проверки на соответствие значения какому-либо условию. Его единственный абстрактный метод называется **test**, принимает один аргумент и возвращает значение типа **boolean**.

```
@FunctionalInterface
public interface Predicate<T> {

    /**
     * Evaluates this predicate on the given argument.
     *
     * @param t the input argument
     * @return {@code true} if the input argument matches the
predicate,
     * otherwise {@code false}
     */
    boolean test(T t);

}
```

Изображение: Даниил Пилипенко

Выводы

В этом видео вы узнали, что такое функциональные интерфейсы, научились передавать действия в методы, писать анонимные классы, а также познакомились с наиболее распространёнными функциональными интерфейсами в Java.