Wyrażenia lambda

Wyrażenie lambda jest metodą. Metodą, którą można przypisać do zmiennej. Można ją także wywołać czy przekazać jako argument do innej metody. Wyrażenia lambda można także porównać do klas anonimowych. Mają one jednak dużo bardziej czytelną i zwięzłą składnię.

Składnia wyrażenia lambda

(lista parametrów) -> (wyrażenie)

Np.:

(x) - > x*x;

Wykonuje potęgowanie.

Od klasy anonimowej do wyrażenia lambda

Załóżmy, że wykonujemy sprawdzanie parzystości liczby przy użyciu interfejsu

https://github.com/idzikpro/JavaBasics/blob/master/src/main/java/pl/idzikpro/lambda/Lambda.java

```
@FunctionalInterface
public interface Lambda<T> {
    boolean check(T object);
}
```

```
Lambda<Integer> isEven = new Lambda<Integer>() {
    @Override
    public boolean check(Integer object) {
        return object % 2 == 0;
    }
};
System.out.println(isEven.check(21));
```

Jak widać robimy to przy użyciu anonimowej klasy interfejsu. Jak to zamienić na wyrażenie lambda?

https://github.com/idzikpro/JavaBasics/blob/master/src/main/java/pl/idzikpro/lambda/LambdaMain.java

```
Lambda<Integer> isEven = object -> object % 2 == 0;
System.out.println(isEven.check(21));
```

Każde wyrażenie lambda jest instancją dowolnego interfejsu funkcyjnego. Interfejs powinien być zatem oznaczony adnotacją **@FunctionalInterface.** Interfejs funkcyjny to interfejs, który ma jedną abstrakcyjną metodę.

Przykładowe interfejsy funkcyjne

Function<T,R> zawiera metodę **R apply(T)**

Consumer<T> zawiera metodę **accept(T)**

Predicate<T> zawiera metodę **test(T)**

Supplier<T> zawiera metodę **T get()**

UnaryOperator<T> to przypadek interfejsu Function, tylko, że przyjmuje i zwraca ten sam typ T

Zalety

Wyrażenia lambda pokazują swoją przydatność

1) Podczas pracy na kolekcjach
2) W trakcie pracy na strumieniach

Kod zajmuje o wiele mniej miejsca, a przez to staje się przez to bardziej czytelny.

Method reference

Dzięki dwóm dwukropkom :: jesteśmy w stanie przypisać metodę do zmiennej bez jej wywołania. Pozwala to na przekazanie takiej metody i wywołanie jej w zupełnie innym miejscu kodu.

Przykład: Klasa::metoda

Jak odwołać się do konstruktora? Klasa::new