Programowanie fukcyjne w Javie

K.S R.A

08 luty 2019

Spis treści

Spis treści

Spis treści

Java 8

Java Development Kit 8 jest jedną z najważniejszych aktualizacji języka rozwijanego przez Oracle. Niesie ona za sobą przede wszystkim istotne zmiany znacznie upraszczające zapis logiki programu. Jest to ważne zwłaszcza dla deweloperów oprogramowania rozwijających projekty zespołowe

Wyrażenia lambda

Wyrażenia lambda są bardzo przydatną zmianą jaka weszła do Javy 8. Nadal możemy programować “po staremu”, jednak czystość kodu wynikająca z prawidłowego stosowania lambd jest naprawdę fajnym zyskiem. Dlatego zachęcam do zapoznania się z tym elementem języka i wcieleniem go do swojego codziennego programowania.

Wyrażenia lambda

```
public class Celebrity {
    private String name;
    private boolean canSing;
    private boolean canAct;
    private boolean canDance;

    public Celebrity(String starName, boolean singer, boolean actor, boolean dancer) {
        this.name = starName;
        this.canSing = singer;
        this.canAct = actor;
        this.canDance = dancer;
    }

    public boolean canSing() {
        return canSing;
    }

    public boolean canDance() {
        return canDance;
    }

    public boolean canAct() {
        return canAct;
    }

    public String getName() {
        return name;
    }

    public String toString() {
        return getName();
    }
}
```

Wyrażenia lambda

```java
public interface CheckTalent {
    boolean test(Celebrity celebrity);
}


public class CheckIfSinger implements CheckTalent {
    boolean test(Celebrity celebrity) {
        return celebrity.canSing();
    }
}
```

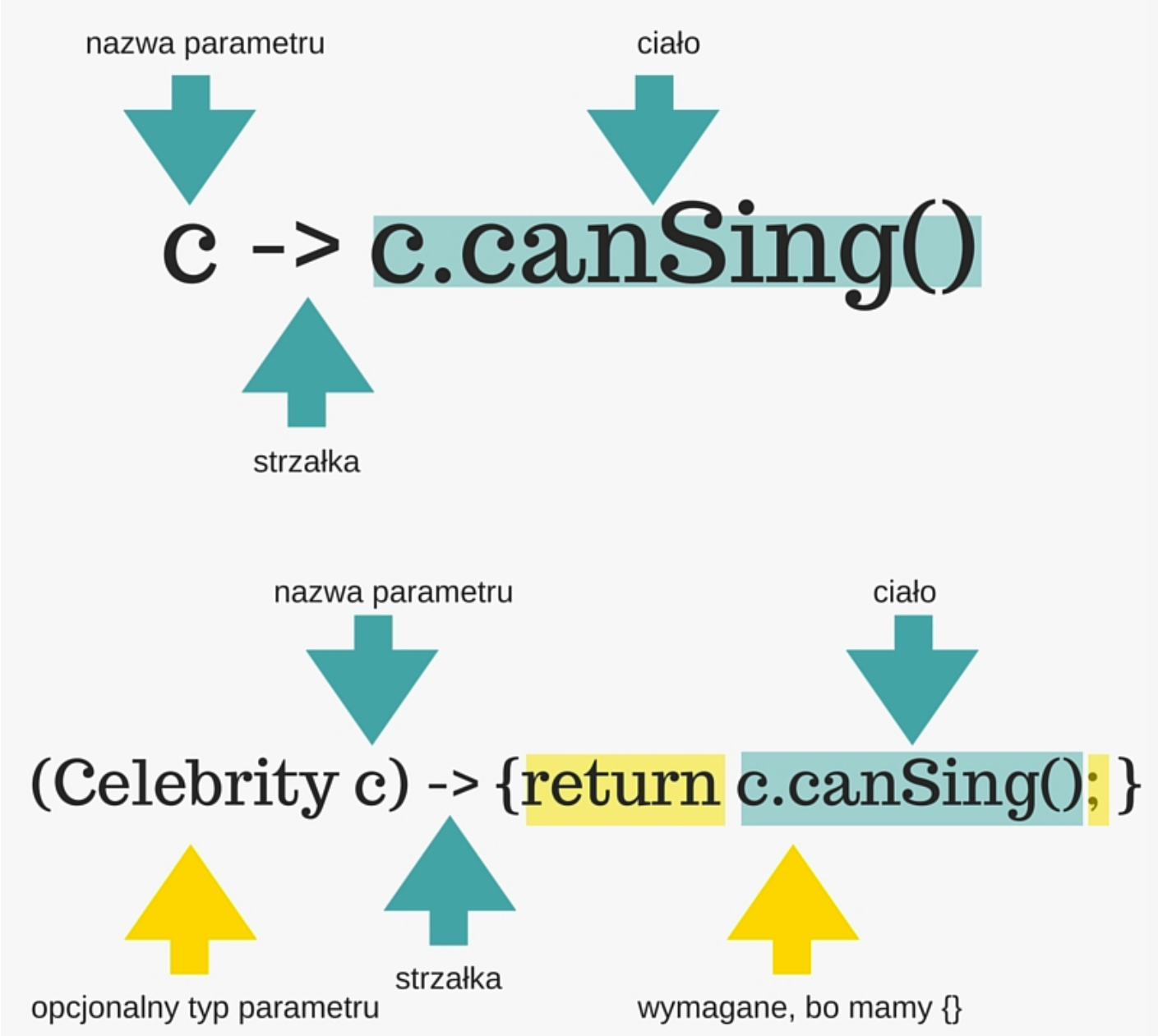
```java
public class Controller {
    public static void main(String[] args) {
        List<Celebrity> celebrities = new ArrayList<Celebrity>();
        celebrities.add(new Celebrity("Michał", true, false, true));
        celebrities.add(new Celebrity("Anna", false, false, false));
        celebrities.add(new Celebrity("Grzesiek", true, true, false));
        print(celebrities, new CheckIfSinger());
    }
```

Wyrażenia lambda

```
private static void print(List<Celebrity> celebrities, CheckTalent checker) {
    for (Celebrity celebrity : celebrities) {
        if (checker.test(celebrity)) {
            System.out.println(celebrity + " ");
        }
        System.out.println();
    }
}
```

```
    private static void print(List<Celebrity> celebrities, CheckTalent checker) {
        for (Celebrity celebrity : celebrities) {
            print(celebrities,c -> c.canSing());
        }
    }
}
```

Wyrażenia lambda



Wyrażenia lambda

```
new Thread(new Runnable() {
    @Override
    public void run() {
        System.out.println("Hello world!");
    }
}).start();
new Thread(() -> System.out.println("Hello world!")).start();
```

Strumienie

Zaraz na naprawdę użyteczny przykład stosowania lambd, ale pierw czym jest strumień?

Strumienie

```java
void convertWithOutStream(String binarna) {
    int dec = 0;
    int potega = 1;
    for (int i = binarna.length(); i >= 1; i--) {
        if (binarna.charAt(i - 1) == '1') {
            dec = dec + (1 * potega);
            System.out.println(dec);
        }
        potęga = potęga * 2;
    }
    System.out.println("Liczba dziesiętna to: " + dec);
}
```

```java
int convertToDecimal(List<Integer> binaries) {
    return binaries.stream().reduce((sum, digit) -> 2 * sum + digit).get();
}
```

Strumienie

```
List fruits = Arrays.asList("apple", "banana", "cherry", "plum", "pear", "pinapple");
fruits.stream()
        .filter(s -> {
            return s.toString()
                    .startsWith("p");
        })
        .map(s -> {
            return s.toString()
                    .toUpperCase();
        }).sorted()
        .forEach(System.out::println);
```

```
List fruits = Arrays.asList("apple", "banana", "cherry", "plum", "pear", "pinapple");
fruits.stream()
        .filter(s -> s.toString()
                .startsWith("p"))
        .map(s -> s.toString()
                .toUpperCase()).sorted()
        .forEach(System.out::println);
```

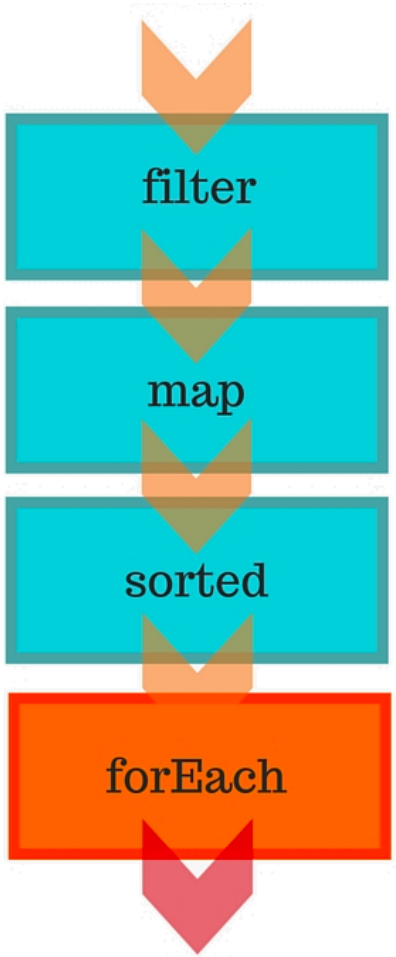
Strumienie

```
List fruits = Arrays.asList("apple", "banana", "cherry", "plum", "pear", "pinapple");
fruits.stream()
        .filter(s -> s.toString()
                .startsWith("p"))
        .map(s -> s.toString()
                .toUpperCase()).sorted()
        .forEach(System.out::println);
```

```
List<String> fruits = new ArrayList();

List<String> fruitsStartWithPAndUpperCase = new ArrayList();
for (String fruit : fruits) {
    if (fruit.startsWith("p")) {
        fruitsStartWithPAndUpperCase.add(fruit.toUpperCase());
        Collections.sort(fruitsStartWithPAndUpperCase);
    }
}
for (String fruit: fruitsStartWithPAndUpperCase){
    System.out.println(fruit);
}
```

Strumienie



Strumienie

Za nami krótkie wprowadzenie do pracy ze strumieniami i wyrażeniami lambda.

Naprawdę ułatwiają one pracę programisty.