Advanced_Backend / Lesen 043 / code / hw_na_les042_1 / src / Main.java

Evgen959 newLes43 d71a156 · 9 hours ago

93 lines (78 loc) · 3.57 KB

Code Blame Raw

```java
/*
Объявите функциональный интерфейс MyPredicate, в котором должен быть единственный метод boolea
Используя ваш интерфейс реализуйте универсальный метод фильтрации списка Person.
Т.е. ваш метод должен принимать список Person и интерфейс MyPredicate который задает условие (
(подсказка: это очень похоже на код StringTransformer написанный в классе)
Выполните несколько вариантов отбора Person (по началу имени, по возрасту и т.д.).
Попробуйте реализовать MyPredicate с помощью лямбда выражений (стрелочных функций).
*/


import java.util.ArrayList;
import java.util.List;
import java.util.function.Predicate;

public class Main {
    public static void main(String[] args) {
        List<Person> people = List.of(
                new Person("Jack", 19),
                new Person("Ann", 20),
                new Person("Garry", 35),
                new Person("Stann", 10),
                new Person("Duck", 13),
                new Person("Nickolaus", 8)
        );

        List<Person> newList1 = filter(people, new MyPredicateImpl());
        System.out.println("----------newList1-------------");
        System.out.println(newList1);

        System.out.println("--------------- 2 ----------------");
        List<Person> newList2 = filter(people, new MyPredicate() {
            @Override
            public boolean test(Person person) {
                return person.getName().charAt(0) < 'J';
            }
        });
        System.out.println(newList2);

```

```java
        System.out.println("--------------- 3 ----------------");
        List<Person> newList3 = filter(people, new MyPredicate() {
            @Override
            public boolean test(Person person) {
                return person.getName().length() == 3;
            }
        });
        System.out.println(newList3);

        System.out.println("--------------- 4 ----------------");
        List<Person> newList4 = filter(people, person -> person.getName().length() == 3);
        System.out.println(newList4);

        System.out.println("--------------- 5 ----------------");
        MyPredicate myPredicate = person -> person.getName().length() == 3;
        System.out.println(filter(people, myPredicate));

        System.out.println("--------------- 6 ----------------");
        System.out.println(filter2(people, p -> p.getName().length() > 4));

        System.out.println(filter2(List.of(1,3,4,-2,3), i-> i%2==0));


    }

    public static List<Person> filter(List<Person> list, MyPredicate predicate){
        List<Person> resaltList = new ArrayList<>();
        for (Person p : list){
            if (predicate.test(p)){
                resaltList.add(p);
            }
        }
        return resaltList;
    }

    public static <E> List<E> filter2(List<E> list, MyPredicate2<E> predicate){
        List<E> resaltList = new ArrayList<>();
        for (E p : list){
            if (predicate.test(p)){
                resaltList.add(p);
            }
        }
        return resaltList;
    }

    public static <E> List<E> filter3(List<E> list, Predicate<E> predicate){
        List<E> resaltList = new ArrayList<>();
        for (E p : list){
            if (predicate.test(p)){
                resaltList.add(p);
            }
        }
        return resaltList;
    }
```

```
93      }
```