Search...

Java 8 Predicate Examples

By mkyong | Last updated: February 11, 2020

Viewed: 132,284 (+1,505 pv/w)

Tags: filter | functional interface | higher order function | java 8 | predicate | stream

In Java 8, Predicate is a functional interface, which accepts an argument and returns a boolean. Usually, it used to apply in a filter for a collection of objects.

```
@FunctionalInterface
public interface Predicate<T> {
  boolean test(T t);
}
```

Further Reading

Java 8 BiPredicate Examples

1. Predicate in filter()

`filter()` accepts predicate as argument.

Java8Predicate.java

```
package com.mkyong.java8;

import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;

public class Java8Predicate {
```

```
    public static void main(String[] args) {

        List<Integer> list = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

        List<Integer> collect = list.stream().filter(x -> x > 5).collect(Collectors.toL

        System.out.println(collect); // [6, 7, 8, 9, 10]

    }

}
```

Output

```
[6, 7, 8, 9, 10]
```

Java8Predicate.java

```
package com.mkyong.java8;

import java.util.Arrays;
import java.util.List;
import java.util.function.Predicate;
import java.util.stream.Collectors;

public class Java8Predicate {

    public static void main(String[] args) {

        Predicate<Integer> noGreaterThan5 =  x -> x > 5;

        List<Integer> list = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

        List<Integer> collect = list.stream()
                .filter(noGreaterThan5)
                .collect(Collectors.toList());

        System.out.println(collect); // [6, 7, 8, 9, 10]
```

```
    }

}
```

Output

```
[6, 7, 8, 9, 10]
```

2. Predicate.and()

2.1 Multiple filters.

Java8Predicate2.java

```
package com.mkyong.java8;

import java.util.Arrays;
import java.util.List;
import java.util.stream.Collectors;

public class Java8Predicate2 {

    public static void main(String[] args) {

        List<Integer> list = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

        // multiple filters
        List<Integer> collect = list.stream()
                .filter(x -> x > 5 && x < 8).collect(Collectors.toList());

        System.out.println(collect);

    }

}
```

Output

```
[6, 7]
```

2.1 Replace with `Predicate.and()`

Java8Predicate2.java

```
package com.mkyong.java8;

import java.util.Arrays;
import java.util.List;
import java.util.function.Predicate;
import java.util.stream.Collectors;

public class Java8Predicate2 {

    public static void main(String[] args) {

        Predicate<Integer> noGreaterThan5 = x -> x > 5;
        Predicate<Integer> noLessThan8 = x -> x < 8;

        List<Integer> list = Arrays.asList(1, 2, 3, 4, 5, 6, 7, 8, 9, 10);

        List<Integer> collect = list.stream()
                .filter(noGreaterThan5.and(noLessThan8))
                .collect(Collectors.toList());

        System.out.println(collect);

    }

}
```

Output

```
[6, 7]
```

3. Predicate.or()

Java8Predicate3.java

```
package com.mkyong.java8;

import java.util.Arrays;
import java.util.List;
import java.util.function.Predicate;
import java.util.stream.Collectors;

public class Java8Predicate3 {

    public static void main(String[] args) {

        Predicate<String> lengthIs3 = x -> x.length() == 3;
        Predicate<String> startWithA = x -> x.startsWith("A");

        List<String> list = Arrays.asList("A", "AA", "AAA", "B", "BB", "BBB");

        List<String> collect = list.stream()
                .filter(lengthIs3.or(startWithA))
                .collect(Collectors.toList());

        System.out.println(collect);

    }

}
```

Output

```
[A, AA, AAA, BBB]
```

4. Predicate.negate()

Find all elements not start with 'A'.

Java8Predicate4.java

```
package com.mkyong.java8;

import java.util.Arrays;
import java.util.List;
import java.util.function.Predicate;
import java.util.stream.Collectors;

public class Java8Predicate4 {

    public static void main(String[] args) {

        Predicate<String> startWithA = x -> x.startsWith("A");

        List<String> list = Arrays.asList("A", "AA", "AAA", "B", "BB", "BBB");

        List<String> collect = list.stream()
                .filter(startWithA.negate())
                .collect(Collectors.toList());

        System.out.println(collect);

    }

}
```

Output

```
[B, BB, BBB]
```

5. Predicate.test() in function

Predicate in function.

Java8Predicate5.java

```
package com.mkyong.java8;

import java.util.Arrays;
import java.util.List;
import java.util.function.Predicate;
import java.util.stream.Collectors;

public class Java8Predicate5 {

    public static void main(String[] args) {

        List<String> list = Arrays.asList("A", "AA", "AAA", "B", "BB", "BBB");

        System.out.println(StringProcessor.filter(
                list, x -> x.startsWith("A")));                    // [A, AA, AAA]

        System.out.println(StringProcessor.filter(
                list, x -> x.startsWith("A") && x.length() == 3)); // [AAA]

    }

}

class StringProcessor {
    static List<String> filter(List<String> list, Predicate<String> predicate) {
        return list.stream().filter(predicate::test).collect(Collectors.toList());
    }
}
```

Output

```
[A, AA, AAA]
[AAA]
```

6. Predicate Chaining

We can chain predicates together.

Java8Predicate6.java

```
package com.mkyong.java8;

import java.util.function.Predicate;

public class Java8Predicate6 {

    public static void main(String[] args) {

        Predicate<String> startWithA = x -> x.startsWith("a");

        // start with "a" or "m"
        boolean result = startWithA.or(x -> x.startsWith("m")).test("mkyong");
        System.out.println(result);     // true

        // !(start with "a" and length is 3)
        boolean result2 = startWithA.and(x -> x.length() == 3).negate().test("abc");
        System.out.println(result2);    // false

    }

}
```

Output

```
true
false
```

7. Predicate in Object

Hosting.java

```
package com.mkyong.java8;

public class Hosting {

    private int Id;
    private String name;
    private String url;

    public Hosting(int id, String name, String url) {
        Id = id;
        this.name = name;
        this.url = url;
    }

    //... getters and setters, toString()
}
```

HostingRespository.java

```
package com.mkyong.java8;

import java.util.List;
import java.util.function.Predicate;
import java.util.stream.Collectors;

public class HostingRespository {

    public static List<Hosting> filterHosting(List<Hosting> hosting,
                                              Predicate<Hosting> predicate) {
        return hosting.stream()
                .filter(predicate)
                .collect(Collectors.toList());
    }
```

```
    }
```

Java8Predicate7.java

```
com.mkyong.java8;

java.util.Arrays;
java.util.List;
java.util.function.Predicate;

class Java8Predicate7 {

lic static void main(String[] args) {

Hosting h1 = new Hosting(1, "amazon", "aws.amazon.com");
Hosting h2 = new Hosting(2, "linode", "linode.com");
Hosting h3 = new Hosting(3, "liquidweb", "liquidweb.com");
Hosting h4 = new Hosting(4, "google", "google.com");

List<Hosting> list = Arrays.asList(new Hosting[]{h1, h2, h3, h4});

List<Hosting> result = HostingRespository.filterHosting(list, x -> x.getName().startsWith
System.out.println("result : " + result);  // google

List<Hosting> result2 = HostingRespository.filterHosting(list, isDeveloperFriendly());
System.out.println("result2 : " + result2); // linode



lic static Predicate<Hosting> isDeveloperFriendly() {
return n -> n.getName().equals("linode");
```

Output

```
result : [Hosting{Id=4, name='google', url='google.com'}]
result2 : [Hosting{Id=2, name='linode', url='linode.com'}]
```

Done.

References

- Predicate JavaDoc
- Java 8 Streams filter examples
- Java 8 BiPredicate Examples

Related Articles

- Java 8 BiPredicate Examples
- Java 8 - Convert a Stream to List
- Java 8 Stream findFirst() and findAny()
- Java 8 Consumer Examples
- Java 8 BiConsumer Examples

mkyong

Founder of Mkyong.com, love Java and open source stuff. Follow him on Twitter. If you like my tutorials, consider make a donation to these charities.

Comments

Join the discussion

3000

B I </> {}

7 COMMENTS Most Voted

Ainur 2 years ago

Awesome, you eased my struggle in understanding conditions in lambda expressions.

6 Reply

Henry Retter 5 months ago

Just what I was looking for, thanks.

1 Reply

Bhaskar 11 months ago

I was a old schooler , I read your article a while ago but for the first time I had used it in real time, the code looked clean and simple with these predicates.Your examples helped me understand it with ease. I feel guilty if I don't ack you on this. Appreciate your efforts.

1 Reply

Author

mkyong 11 months ago

Reply to Bhaskar

Thanks for your kind words; it means a lot to me.

1 Reply

sarath 4 minutes ago

superb no words to say. But it is in pdf form it will very useful for further

0 Reply

Ana 9 months ago

Thank you so much for the explanation!

0 Reply

Jack Mason 1 year ago

Hi Mkyong,

I've gotten so much out of your post over the years. Thank you so much!

0 Reply