HowToDoInJava

Java Stream noneMatch()

Last Updated: April 17, 2022 By: Lokesh Gupta Java 8 Java Stream Methods

Java **Stream noneMatch()** method is a *short-circuiting terminal operation*. The *noneMatch()* is used to check that **Stream does not contain any element matching the provided Predicate.**

Table Of Contents

1. Stream noneMatch() Method

1.1. Syntax

```
boolean noneMatch(Predicate<? super T> predicate)
```

The `noneMatch()` returns:

- **true** – if no element in the stream matches the given predicate, or the stream is empty.

- **false** – if at least one element matches the given predicate.

Here `predicate` a **non-interfering and stateless predicate** to apply to elements of the stream.

It is **short-circuiting** operation. A terminal operation is short-circuiting if, when presented with infinite input, it may terminate in finite time.

1.2. Description

- The `noneMatch()` is a **short-circuiting terminal** operation.
- It returns whether no element of the stream match the provided predicate.
- It **may not evaluate the predicate on all elements if not necessary for determining the result**. The method returns `true` if no stream element matches the given predicate, else method returns `false`.
- If the stream is empty then `true` is returned and the predicate is not evaluated.
- It is pretty much opposite to method allMatch().

2. Stream noneMatch() Example

Java example of `Stream.noneMatch()` method to check if no element in the Stream contains any numeric/digit character.

```
import java.util.stream.Stream;

public class Main
{
    public static void main(String[] args)
    {
        Stream<String> stream = Stream.of("one", "two", "three", "fou

        boolean match = stream.noneMatch( s -> s.contains("\\d+") );

        System.out.println(match);      //true
    }
}
```

Program output.

```
true
```

3. Conclusion

Stream.noneMatch() method can be useful in certain cases where we need to run a check on all stream elements. For example, we can use *noneMatch()* function on a stream of *Employee* objects to validate that all employees are NOT below a certain age.

Happy Learning !!

Sourcecode on Github

Was this post helpful?

Let us know if you liked the post. That's the only way we can improve.

Yes

No

Recommended Reading:

1. Java Stream reuse – traverse stream multiple times?
2. Java Stream forEach()

3. Java Stream forEachOrdered()
4. Java Stream filter()
5. Java Stream min()
6. Java Stream map()
7. Java Stream flatMap()
8. Java Stream skip()
9. Java Stream anyMatch()
0. Java Stream toArray()

Leave a Comment

Name *

Email *

Website

□ Add me to your newsletter and keep me updated whenever you publish new blog posts

Post Comment

Search …

rated people
DO THE
WORK YOU LIKE,
WHEN IT SUIT
FIND OUT M

HowToDoInJava

A blog about Java and related technologies, the best practices, algorithms, and interview questions.

Meta Links

- About Me
- Contact Us
- Privacy policy
- Advertise
- Guest Posts

Blogs

REST API Tutorial