HowToDoInJava

Java Stream – Find, Count and Remove Duplicates

Last Updated: March 10, 2022 By: Lokesh Gupta Java 8 Java Stream Basics

Few simple examples to find and count the duplicates in a *Stream* and remove those duplicates since Java 8. We will use ArrayList to provide a *Stream* of elements including duplicates.

Table Of Contents

1. Stream.distinct() – To Remove Duplicates

1.1. Remove Duplicate Strings

The `distinct()` method returns a *Stream* consisting of the distinct elements of the given stream. The **object equality is checked according to the object's `equals()` method**.

Find all distinct strings

```
List<String> list = Arrays.asList("A", "B", "C", "D", "A", "B", "C");

// Get list without duplicates
List<String> distinctItems = list.stream()
                                 .distinct()
                                 .collect(Collectors.toList());

// Let's verify distinct elements
System.out.println(distinctItems);
```

Program output:

Output

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

1.2. Remove Duplicate Custom Objects

The same syntax can be used to remove the duplicate objects from *List*. To do so, we need to be very careful about the object's *equals()* method, because it will decide if an object is duplicate or unique.

Consider the below example where two *Person* instances are considered equal if both have the same *id* value.

```
public class Person
{
    private Integer id;
    private String fname;
    private String lname;
}
```

Let us see an example of how we can remove duplicate *Person* objects from a *List*.

Get Distinct Objects using Default Equality

```
//Add some random persons
Collection<Person> list = Arrays.asList(p1, p2, p3, p4, p5, p6);

// Get distinct people by id
List<Person> distinctElements = list.stream()
        .distinct()
        .collect( Collectors.toList() );
```

To **find all unique objects using a different equality condition**, we can take the help of the following *distinctByKey()* method. For example, we are finding all unique objects by Person's full name.

Get Distinct Objects using Custom Equality

```
//Add some random persons
List<Person> list = Arrays.asList(p1, p2, p3, p4, p5, p6);

// Get distinct people by full name
List<Person> distinctPeople = list.stream()
             .filter( distinctByKey(p -> p.getFname() + " " + p.getLname()) )
             .collect( Collectors.toList() );

//********The distinctByKey() method need to be created**********

public static <T> Predicate<T> distinctByKey(Function<? super T, Object> keyExtractor)
{
  Map<Object, Boolean> map = new ConcurrentHashMap<>();
  return t -> map.putIfAbsent(keyExtractor.apply(t), Boolean.TRUE) == null;
}
```

2. Collectors.toSet() – To Remove Duplicates

Another simple and very useful way is to store all the elements in a `Set`. **Sets, by definition, store only distinct elements.** Note that a *Set* stores distinct items by comparing the objects with *equals()* method.

Here, *we cannot compare the objects using a custom equality condition*.

```
ArrayList<Integer> numbersList
= new ArrayList<>(Arrays.asList(1, 1, 2, 3, 3, 3, 4, 5, 6, 6, 6, 7, 8));

Set<Integer> setWithoutDuplicates = numbersList.stream()
.collect(Collectors.toSet());

System.out.println(setWithoutDuplicates);
```

Program output:

```
[1, 2, 3, 4, 5, 6, 7, 8]
```

3. Collectors.toMap() – To Count Duplicates

Sometimes, we are interested in finding out which elements are duplicates and how many times they appeared in the original list. We can use a `Map` to store this information.

We have to iterate over the list, put the element as the Map key, and all its occurrences in the Map value.

```
// ArrayList with duplicate elements
ArrayList<Integer> numbersList
= new ArrayList<>(Arrays.asList(1, 1, 2, 3, 3, 3, 4, 5, 6, 6, 6, 7, 8));

Map<Integer, Long> elementCountMap = numbersList.stream()
.collect(Collectors.toMap(Function.identity(), v -> 1L, Long::sum));

System.out.println(elementCountMap);
```

Program output:

```
{1=2, 2=1, 3=3, 4=1, 5=1, 6=3, 7=1, 8=1}
```

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 – Remove Duplicates from Array
2. Java Stream count() Matches with filter()
3. Java Stream reuse – traverse stream multiple times?
4. String indent(count) – Left indent lines in Java
5. Java program to count vowels and consonants in a String
6. Count Number of Lines in a File in Java
7. Spring Batch Count of Processed Records Example
8. Python String count()
9. Hibernate count, min, max, sum, avg Functions
0. Java remove trailing whitespaces from String

Join 7000+ Awesome Developers

Get the latest updates from industry, awesome resources, blog updates and much more.

Email Address

Subscribe

** We do not spam !!*

Leave a Comment

Name *

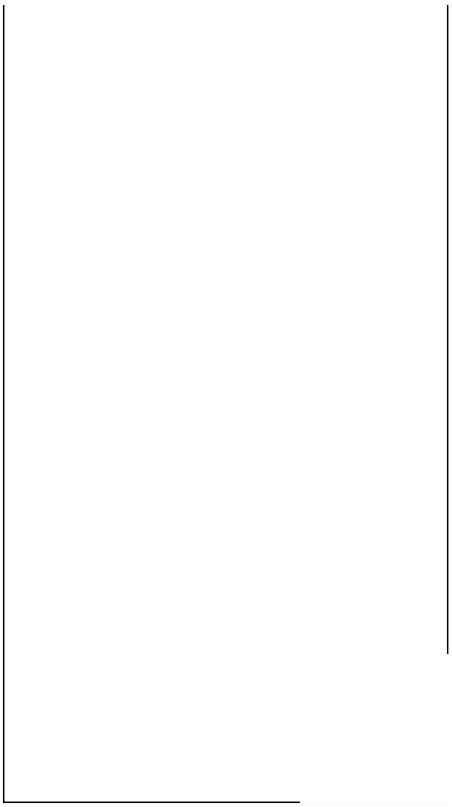
Email *

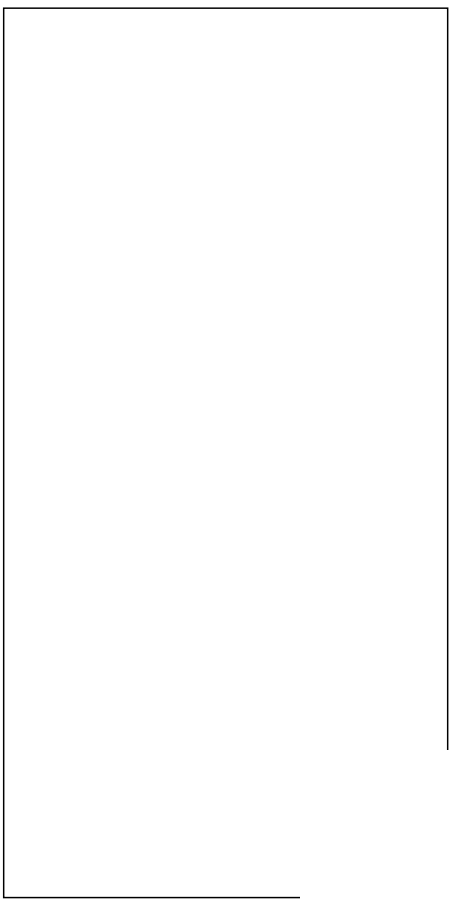
Website

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

Post Comment

Search ...





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