HowToDoInJava

Collecting Stream of Primitives into Collection or Array

Last Updated: March 14, 2022 By: Lokesh Gupta Java 8 Java Stream Basics

Java Collectors class provides many *static* methods to collect items from a *Stream* and store them into a Collection. But these methods do not work with streams of primitives.

Works with Objects; But doesn't work with Primitives

```
//It works perfect !!
List<String> strings = Stream.of("how", "to", "do", "in", "java")
            .collect(Collectors.toList());

//Compilation Error !!
IntStream.of(1,2,3,4,5).collect(Collectors.toList());
```

To convert the Stream of primitives to a Collection, we can adapt one of the following ways. Note that **these methods can be applied to all three primitive streams i.e. *IntStream*, *LongStream*, and *DoubleStream***.

Table Of Contents

1. Collecting IntStream into Collection using Boxed Stream

Using `boxed()` method of `IntStream`, `LongStream` or `DoubleStream` e.g. IntStream.boxed(), we can get stream of wrapper objects which can be collected by `Collectors` methods.

```
//Primitive Stream -> List
List<Integer> listOfIntegers = IntStream.of(1,2,3,4,5)
  .boxed()
  .collect(Collectors.toList());

List<Long> listOfLongs = LongStream.of(1,2,3,4,5)
  .boxed()
  .collect(Collectors.toList());

List<Double> listOfDoubles = DoubleStream.of(1.0,2.0,3.0,4.0,5.0)
  .boxed()
  .collect(Collectors.toList());
```

2. Collecting IntStream to List by Mapping to Objects

Another way is to manually to the boxing using IntStream.mapToObj(), IntStream.mapToLong() or IntStream.mapToDouble() methods. There are other similar methods in other stream classes, which you can use.

```
//Primitive Stream -> List
List<Integer> listOfIntegers = IntStream.of(1,2,3,4,5)
 .mapToObj(Integer::valueOf)
 .collect(Collectors.toList());

List<Long> listOfLongs = LongStream.of(1,2,3,4,5)
 .mapToObj(Long::valueOf)
 .collect(Collectors.toList());

List<Double> listOfDoubles = DoubleStream.of(1.0,2.0,3.0,4.0,5.0)
 .mapToObj(Double::valueOf)
 .collect(Collectors.toList());
```

3. Collecting IntStream to Array

It is also useful to know how to **convert the primitive stream to an array**. Use `IntStream.toArray()` method to convert from **int stream to array**.

```
int[] intArray = IntStream.of(1, 2, 3, 4, 5).toArray();

long[] longArray = LongStream.of(1, 2, 3, 4, 5).toArray();

double[] doubleArray = DoubleStream.of(1, 2, 3, 4, 5).toArray();
```

Similarly, use `toArray()` function of `LongStream` or `DoubleStream` as well.

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. Convert Between Array of Primitives and Array to Objects
2. Collecting Stream Items into Map in Java
3. Collecting Stream Items into List in Java

4. Java Stream reuse – traverse stream multiple times?
5. Java Memory Management – Garbage Collection Algorithms
6. Java MongoDB : Insert Document(s) in Collection Examples
7. Java Garbage Collection Algorithms [till Java 9]
8. Spring HATEOAS – Embedded Collection Model Name
9. GSON – Parse JSON array to Java array or list
0. Convert between Stream and Array

1 thought on “Collecting Stream of Primitives into Collection or Array”

Priya

September 9, 2017 at 12:26 pm

You made the concept very easy to understand. Thanks for sharing.

Reply

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