

## Append or Prepend Items to a Stream

📅 Last Updated: March 14, 2022    👤 By: Lokesh Gupta    📁 Java 8    💎 Java Stream Basics, Java Stream Methods

Learn to add items to a [Java Stream](#). Remember that a `Stream` is not a data structure or collection that can store values. To add items to an existing `Stream`, we need to :

- Create a new Stream with items to be added
- Concatenate with the first stream to get a merged stream.

### Table Of Contents

- 1. Concatenating Streams
- 2. Examples of Adding Items
  - 2.1. Appending Items
  - 2.2. Prepending Items
- 3. Conclusion

## 1. Concatenating Streams

The `Stream.concat(stream1, stream2)` is used to merge two streams into one stream which consists of all the elements of both streams.

- The `concat(s1, s2)` method creates a **lazily concatenated stream** whose elements are all the elements of the `s1` followed by all the elements of the `s2`.
- The resulting stream is **ordered if both of the input streams are ordered**.
- The resulting stream is **parallel if either of the input streams is parallel**.

## 2. Examples of Adding Items

### 2.1. Appending Items

To append items at the start of a Stream, create a new stream of items and pass the new `Stream` as the first method argument in the `concat()` method.

```
Stream<Integer> stream = Stream.of(1, 2, 3, 4);
```

```
//Append 5 and 6 to the stream
Stream<Integer> appenededStream = Stream.concat(stream, Stream.of(5, 6));

//Verify Stream
appenededStream.forEach(System.out::print); //Prints 123456
```

## 2.2. Prepending Items

To prepend the items at the end of a Stream, create a new stream of the items and pass the new Stream as the second method argument in the `concat()` method.

```
Stream<Integer> stream = Stream.of(1, 2, 3, 4);

//Prepend 0 to the stream
Stream<Integer> prependedStream = Stream.concat(Stream.of(0), stream);

//Verify Stream
prependingStream.forEach(System.out::print); //Prints 01234
```

## 3. Conclusion

The Stream API provides lots of useful methods which can be used to solve many problems. In the above case, **adding new objects to the Java stream** has been demonstrated using the `concat()` API whose original purpose is to merge two streams.

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. [Getting Distinct Stream Items by Comparing Multiple Fields](#)
2. [Collecting Stream Items into Map in Java](#)
3. [Collecting Stream Items into List in Java](#)

4. [Java 8 – Join or append stream of strings](#)
5. [Java Stream reuse – traverse stream multiple times?](#)
6. [Join String Array Items with Delimiter](#)
7. [Python JSON – Append JSON to a File](#)
8. [Finding Top N Items in Array](#)
9. [Removing Items from an Array in Java](#)
0. [Finding Sum and Average of Array Items](#)

### Join 7000+ Awesome Developers

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

**Email Address**

**Subscribe**

*\* We do not spam !!*

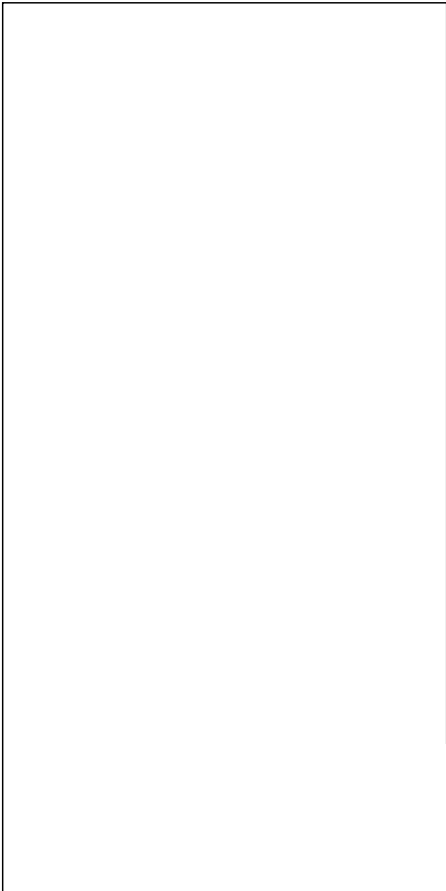
---

**Leave a Comment**

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

### Post Comment





## 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](#)



Copyright © 2022 · Hosted on [Cloudways](#) · [Sitemap](#)