

# Streams

## Stream Splitter

# How Splitterator Works

- you start with some kind of collection with n elements
  - imagine a box containing number of toys
- now you want to split this collection in two (or more collections)
  - imagine moving some toys from the original box to the new box
- in order to do this you have to create an object of type `Splitterator`
- once you have `Splitterator` object, you can apply some common methods...

# Common Splitterator Methods

Method	What it does
<code>Splitterator&lt;T&gt; trySplit()</code>	Returns <code>Splitterator</code> containing about half of the data, which is then removed from the original <code>Splitterator</code> (if data is no longer splittable it returns <code>null</code> )
<code>void forEachRemaining(Consumer&lt;T&gt; c)</code>	Processes remaining elements in <code>Splitterator</code>
<code>boolean tryAdvance(Consumer&lt;T&gt; c)</code>	Processes single element from <code>Splitterator</code> (if exists), returns if the element is processed

```
// example #1
```

```
List<String> list = Arrays.asList("One", "Two", "Three", "Four", "Five");
```

```
Stream<String> stream = list.stream();
```

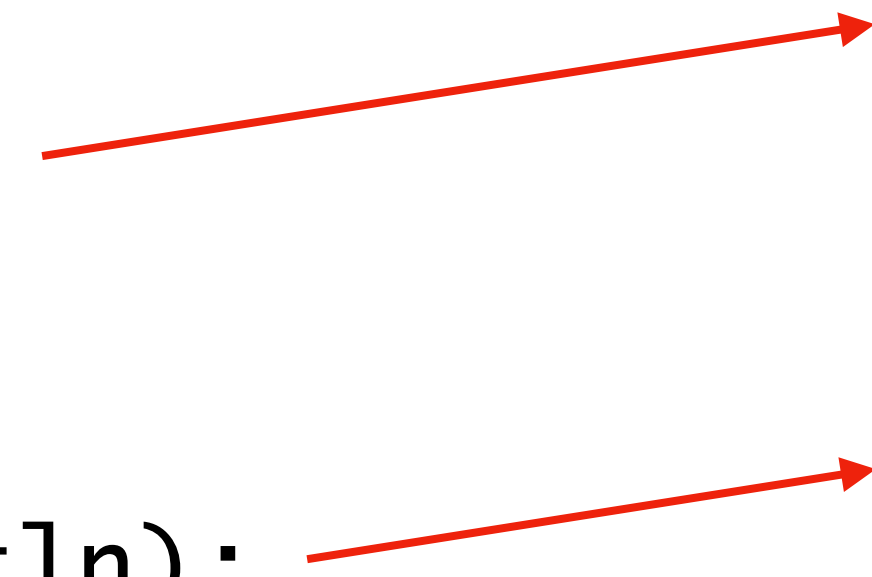
```
Splitter<String> originalSplitter = stream.splitter();
```

```
Splitter<String> newSplitter = originalSplitter.trySplit();
```

```
newSplitter.forEachRemaining(System.out::println);
```

```
System.out.println("---");
```

```
originalSplitter.forEachRemaining(System.out::println);
```



```
One  
Two  
---  
Three  
Four  
Five
```

NB. Once you apply `forEachRemaining()` method on a Splitter, all elements are processed and Splitter is now empty, so if you run this command again:

`originalSplitter.forEachRemaining(System.out::println);` it will return nothing

// example #2

newSplitter

originalSplitter

```
List<String> list = Arrays.asList("One", "Two", "Three", "Four", "Five");
```

```
Stream<String> stream = list.stream();
```

```
Splitter<String> originalSplitter = stream.splitter();
```

```
Splitter<String> newSplitter = originalSplitter.trySplit();
```

```
newSplitter.tryAdvance(System.out::println);
```

```
System.out.println("---");
```

```
newSplitter.forEachRemaining(System.out::println);
```

```
System.out.println("---");
```

```
originalSplitter.tryAdvance(System.out::println);
```

```
System.out.println("---");
```

```
originalSplitter.forEachRemaining(System.out::println);
```

One

---

Two

---

Three

---

Four

Five