Understand Java Streams Intermediate Operations filter() & flatMap()

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand the structure & functionality of stream aggregate operations
  - Intermediate operations
    - map() & mapToInt()
    - filter() & flatMap()

Input x

Stream flatMap(Function<…> mapper)

Output f(x)

Stream filter(Predicate<…> pred)

Output g(f(x))

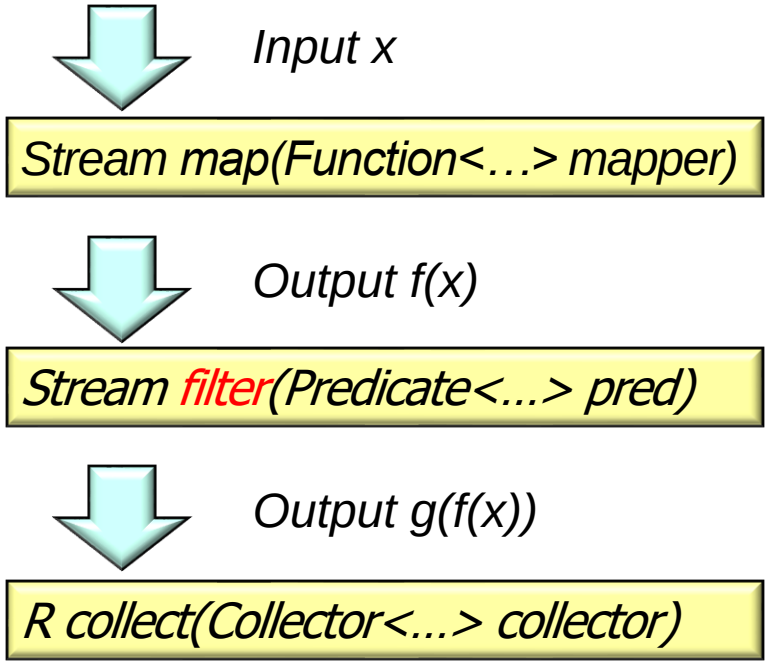
R collect(Collector<…> collector)

These are both stateless, run-to-completion operations

Overview of the filter() Intermediate Operation

Overview of the filter() Intermediate Operation

- Tests a predicate against each element of input stream & returns an output stream containing only elements that match the predicate

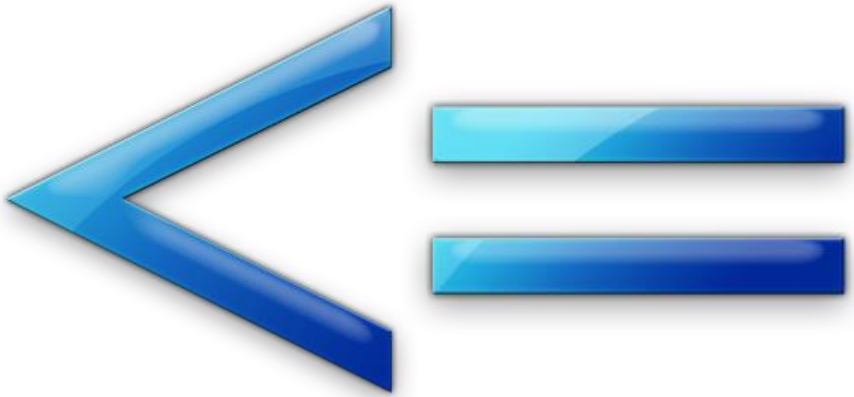


See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#filter

Overview of the filter() Intermediate Operation

- Tests a predicate against each element of input stream & returns an output stream containing only elements that match the predicate

The # of output stream elements may be less than the # of input stream elements.



Input x

Stream map(Function<…> mapper)

Output f(x)

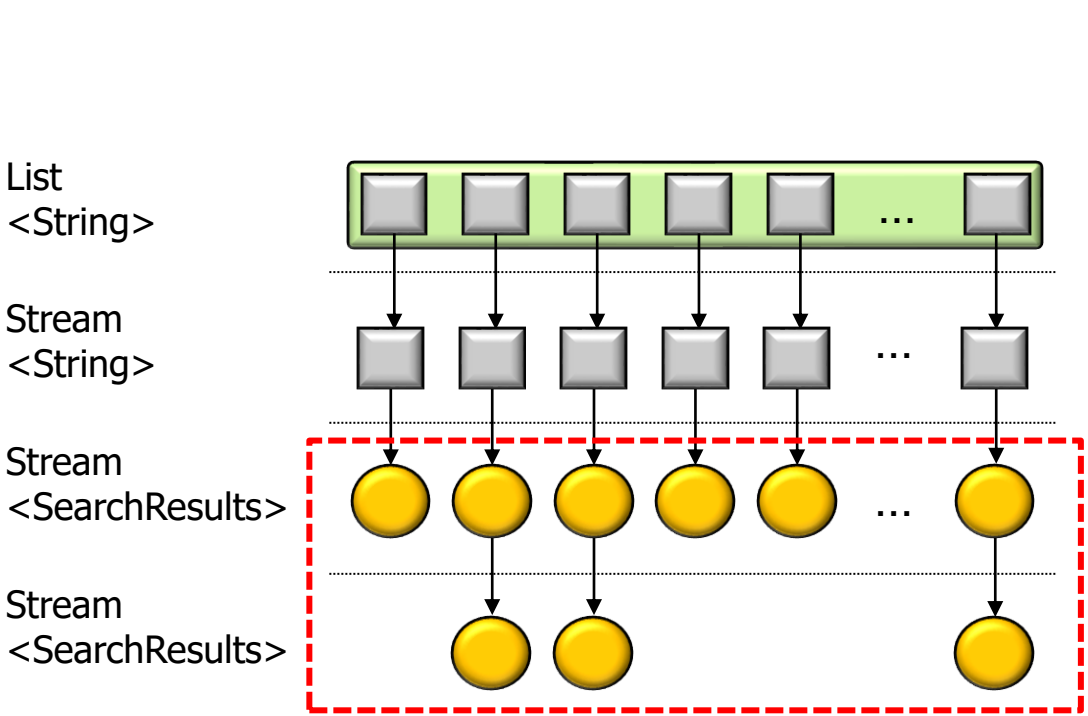
Stream filter(Predicate<…> pred)

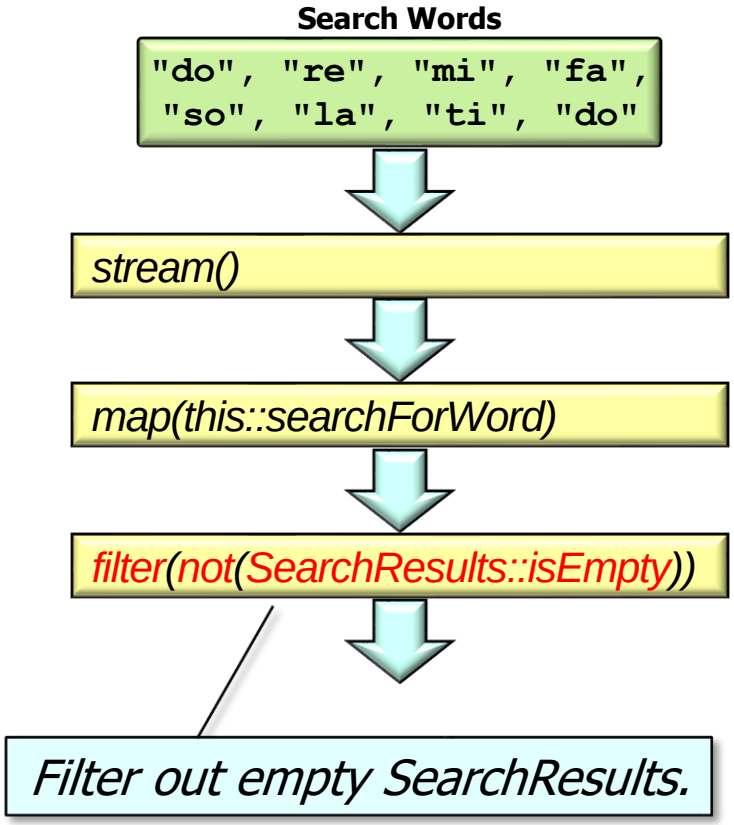
Output g(f(x))

R collect(Collector<…> collector)

Overview of the filter() Intermediate Operation

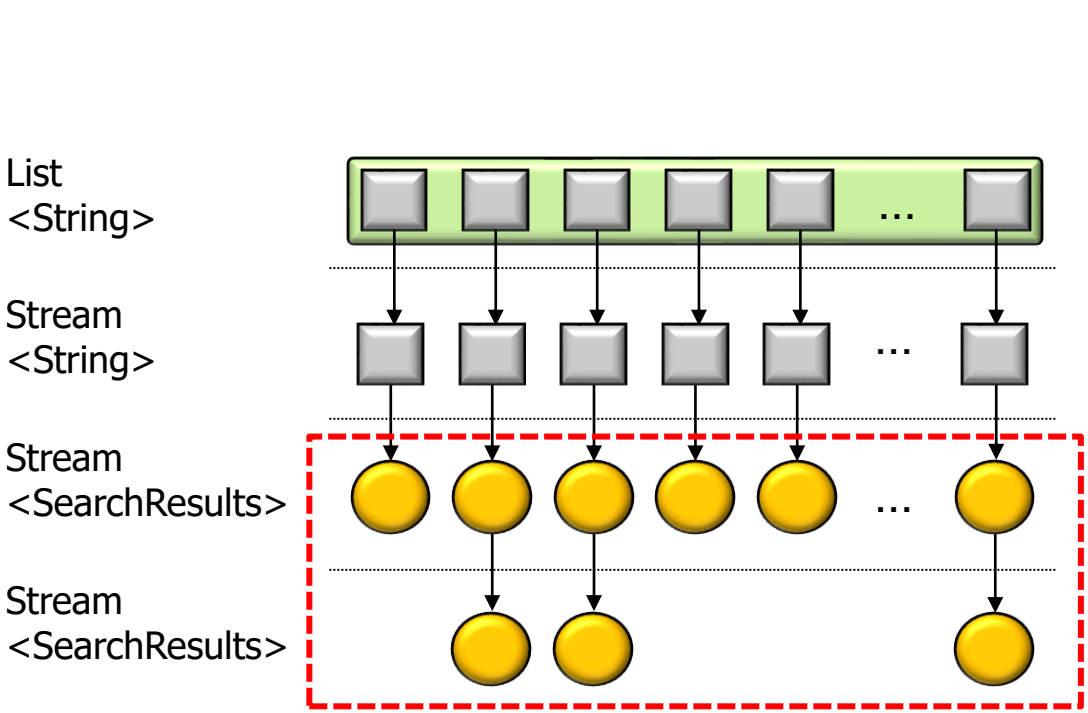
- Example of applying filter() & a predicate in the SimpleSearchStream program

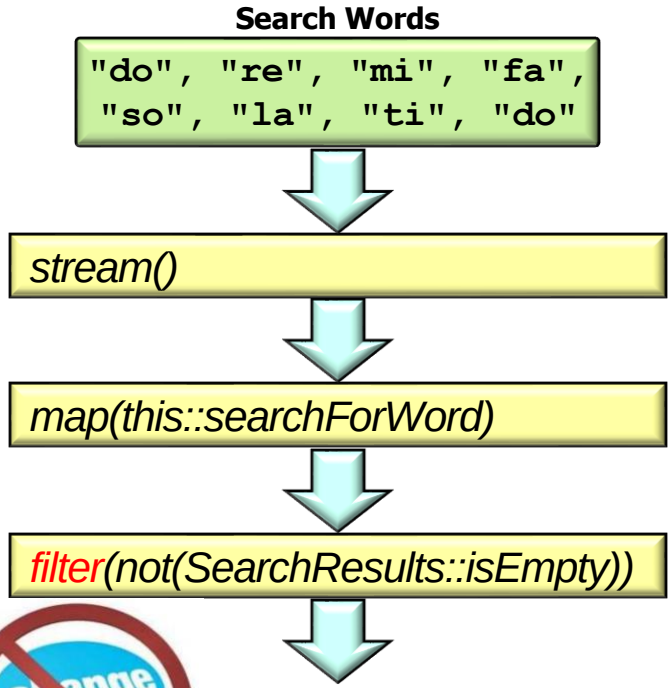




Overview of the filter() Intermediate Operation

- Example of applying filter() & a predicate in the SimpleSearchStream program



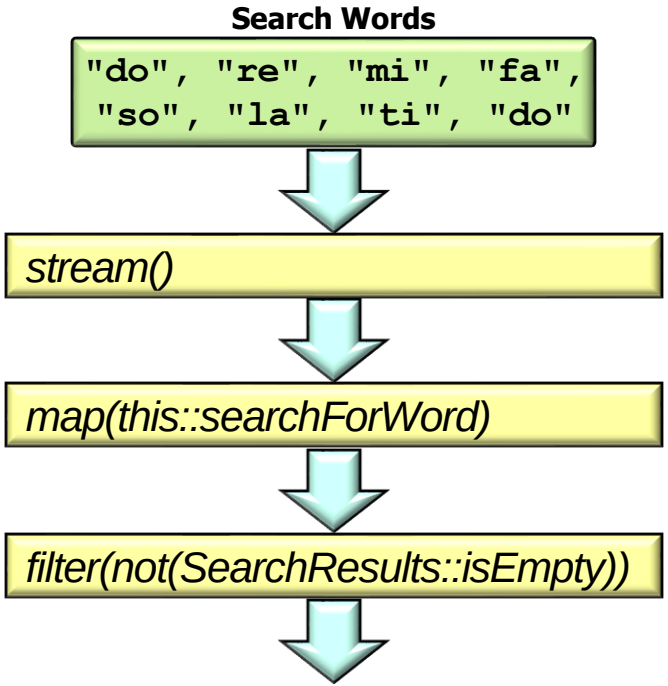


filter() *can't* change the type or value of elements it processes

Overview of the filter() Intermediate Operation

- Example of applying filter() & a predicate in the SimpleSearchStream program

```
List<SearchResults> results =
  wordsToFind
     .stream()
     .map(this::searchForWord)
     .filter(not
        (SearchResults::isEmpty))
     .collect(toList());
```



Again, note the fluent interface style.

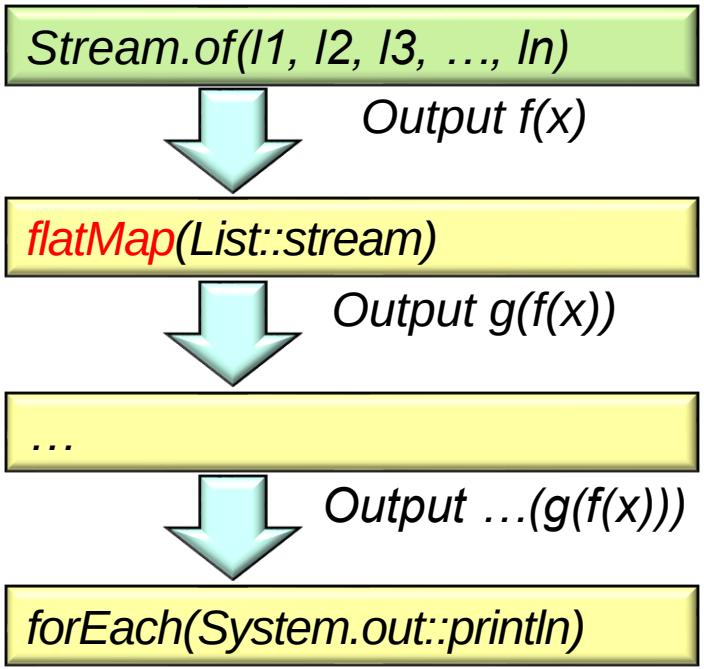
See en.wikipedia.org/wiki/Fluent_interface

Overview of the flatMap() Intermediate Operation

Overview of the flatMap() Intermediate Operation

- Returns a stream that replaces each element of this stream w/contents of a mapped stream produced by applying the provided mapping function to each element

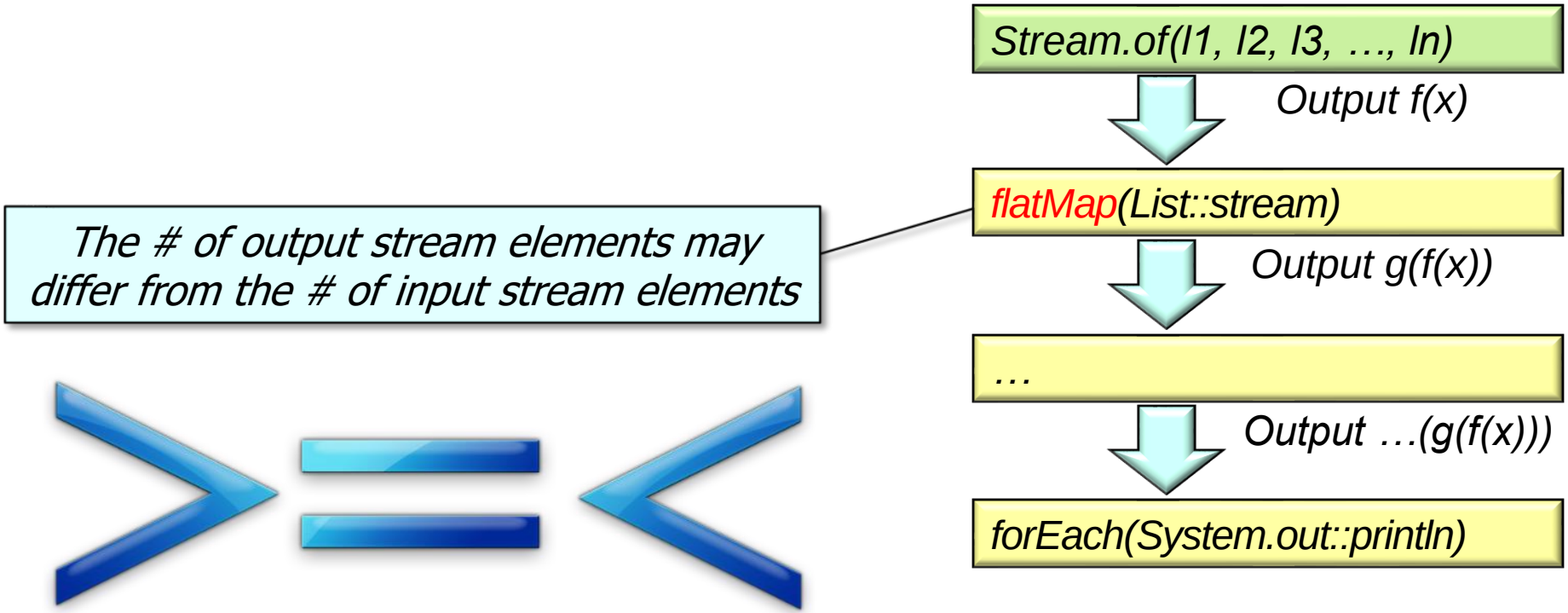
This definition sounds like map() at first glance, but there are important differences!



See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#flatMap

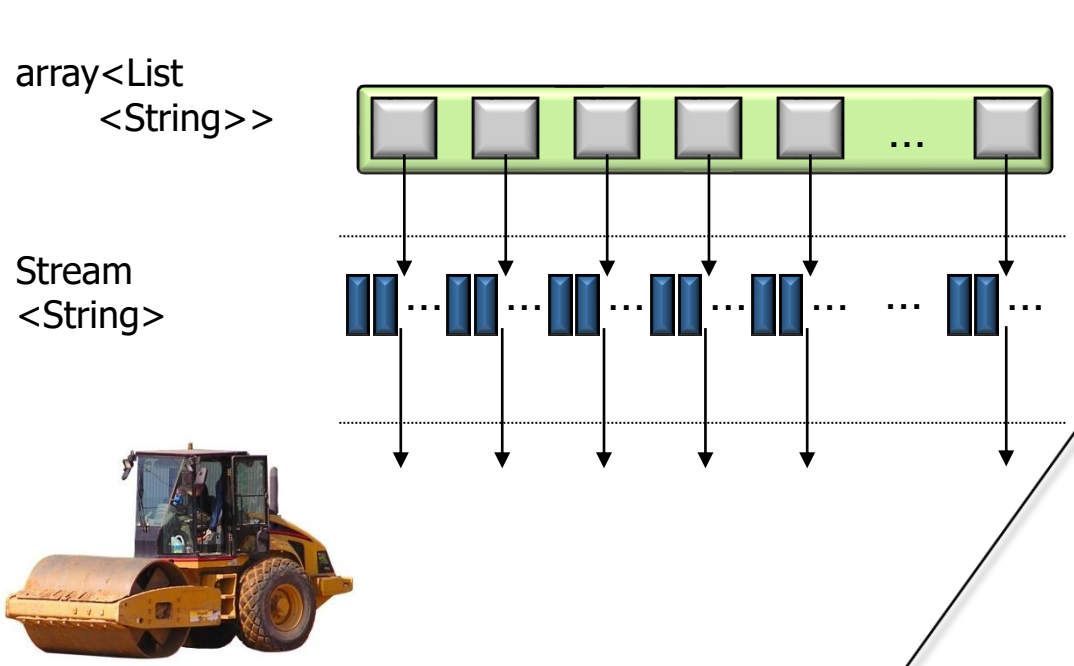
Overview of the flatMap() Intermediate Operation

- Returns a stream that replaces each element of this stream w/contents of a mapped stream produced by applying the provided mapping function to each element



Overview of the flatMap() Intermediate Operation

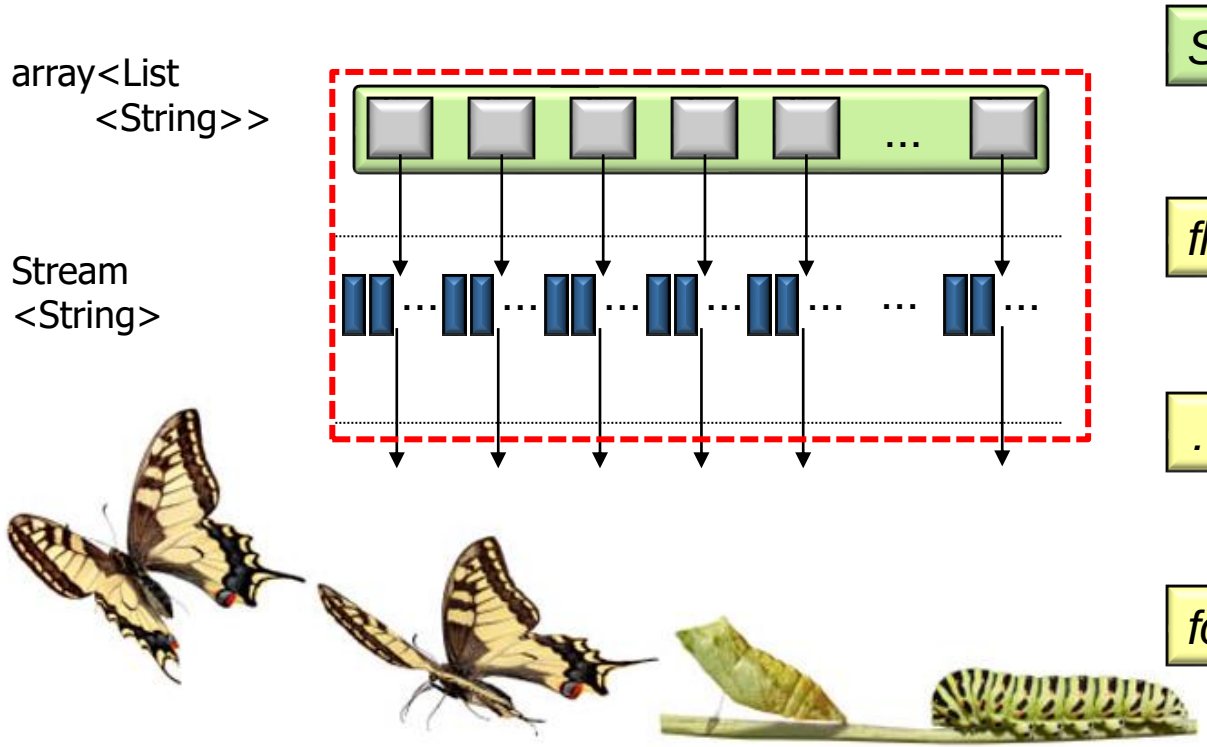
- Returns a stream that replaces each element of this stream w/contents of a mapped stream produced by applying the provided mapping function to each element



"Flatten" an array of lists of strings into a stream of strings

Overview of the flatMap() Intermediate Operation

- Returns a stream that replaces each element of this stream w/contents of a mapped stream produced by applying the provided mapping function to each element



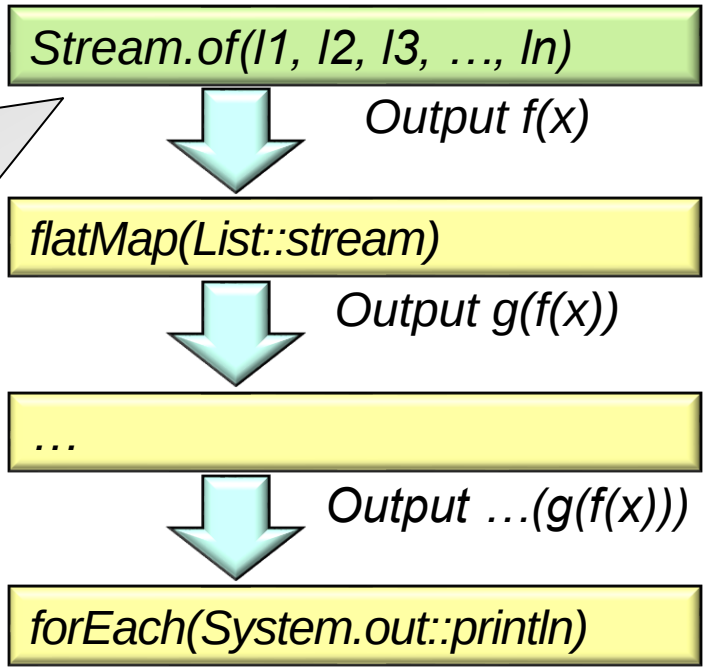
flatMap() *may* transform the type of elements it processes

Overview of the flatMap() Intermediate Operation

- Returns a stream that replaces each element of this stream w/contents of a mapped stream produced by applying the provided mapping function to each element

```
List<String> l1 = ...;
List<String> l2 = ...;
List<String> l3 = ...;
...
List<String> ln = ...;

Stream
   .of(l1, l2, l3, ..., ln)
   .flatMap(List::stream)
   ...
   .forEach(System.out::println);
```



See github.com/douglascraigschmidt/LiveLessons/tree/master/Java8/ex12

End of Understand Java Streams Intermediate Operations filter() & flatMap()