Java 8 Stream Factory Methods

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 Lesson

- Recognize common stream factory methods
Common Factory Methods for Creating Streams
Common Factory Methods for Creating Streams

• Streams can be obtained various ways

See docs.oracle.com/javase/8/docs/api/java/util/stream/package-summary.html
Streams can be obtained various ways, e.g.,

- From a Java collection

```
List<String> wordsToFind = Arrays.asList("do", "re", "me", ...);

List<SearchResults> results = wordsToFind.stream()
    ...

or

List<SearchResults> results = wordsToFind.parallelStream()
    ...
```
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
- From a Java collection

```java
List<String> wordsToFind = Arrays.asList("do", "re", "me", ...);

List<SearchResults> results = wordsToFind.stream()
    ...;
```

See [docs.oracle.com/javase/tutorial/collectionsstreams](docs.oracle.com/javase/tutorial/collectionsstreams)
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection

```java
List<String> wordsToFind =
    Arrays.asList("do", "re", "me", ...);

List<SearchResults> results =
    wordsToFind.stream();
    ...

or

List<SearchResults> results =
    wordsToFind.parallelStream();
    ...
```

See docs.oracle.com/javase/tutorial/collections/streams/parallelism.html
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
- From a Java collection

```java
List<String> wordsToFind = Arrays.asList("do", "re", "me", ...);

List<SearchResults> results = wordsToFind.stream()
    ...,

or

List<SearchResults> results =
    wordsToFind.stream()
    ...
    .parallel()
```

A call to `parallel()` can appear anywhere in a stream & will have same effect as `parallelStream()`

See `docs.oracle.com/javase/8/docs/api/java/util/stream/BaseStream.html#parallel`
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  ```java
  String[] array = {
      "a", "b", "c", "d", "e"
  };
  Stream<String> stream =
      Arrays.stream(array);
  stream.forEach(s ->
      System.out.println(s));
  or
  stream.forEach(System.out::println);
  ```
  - From an array
    ```java
    Common Factory Methods for Creating Streams
    ```

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  ```java
  String[] array = {
      "a", "b", "c", "d", "e"
  };
  Stream<String> stream =
      Arrays.stream(array);
  stream.forEach(s ->
      System.out.println(s));
  or
  stream.forEach(System.out::println);
  ```
  - From an array
    ```java
    ```
Streams can be obtained various ways, e.g.,

- From a Java collection

```java
String[] array = {
    "a", "b", "c", "d", "e"
};

Stream<String> stream =
    Arrays.stream(array);

stream.forEach(s ->
    System.out.println(s));
```

- Or

```java
stream.forEach(System.out::println);
```

Common Factory Methods for Creating Streams

- Get & print all the elements in an array
Streams can be obtained various ways, e.g.,

- From a Java collection
- From an array

```java
String[] array = {
    "a", "b", "c", "d", "e"
};

Stream<String> stream =
    Arrays.stream(array);

stream.forEach(s ->
    System.out.println(s));

or

stream.forEach(System.out::println);
```

Get & print all the elements in an array

Common Factory Methods for Creating Streams
Streams can be obtained various ways, e.g.,

- From a Java collection
- From an array
- From a static factory method

```java
String[] array = {
    "a", "b", "c", "d", "e"
};

Stream<String> s = Stream.of(array);

s.forEach(s -> System.out.println(s));

or

stream.forEach(System.out::println);
```
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method

```java
String[] array = {
    "a", "b", "c", "d", "e"
};

Stream<String> s = Stream.of(array);

s.forEach(s -> System.out.println(s));
```

Get & print all the elements in an array

or

```java
stream.forEach(System.out::println);
```
Streams can be obtained various ways, e.g.,

- From a Java collection
- From an array
- From a static factory method

```java
String[] array = {
    "a", "b", "c", "d", "e"
};

Stream<String> s = Stream.of(array);

s.forEach(s -> System.out.println(s));

or

stream.forEach(System.out::println);
```

Get & print all the elements in an array
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method

```java
Stream.iterate(new BigInteger[]{BigInteger.ONE, BigInteger.ONE},
               f -> new BigInteger[]{f[1],
                                     f[0].add(f[1])})
      .map(f -> f[0])
      .limit(100)
      .forEach(System.out::println);
```

Do not use iterate() in a parallel stream! Use range() instead.
Common Factory Methods for Creating Streams

• Streams can be obtained various ways, e.g.,
  • From a Java collection
  • From an array
  • From a static factory method

```java
Stream.iterate(new BigInteger[]{BigInteger.ONE, BigInteger.ONE},
               f -> new BigInteger[]{f[1],
                                      f[0].add(f[1])})
    .map(f -> f[0])
    .limit(100)
    .forEach(System.out::println);
```

Generate & print first 100 Fibonacci numbers
Streams can be obtained various ways, e.g.,

- From a Java collection
- From an array
- From a static factory method

```java
Stream.iterate(new BigInteger[]{BigInteger.ONE, BigInteger.ONE},
    f -> new BigInteger[]{f[1], f[0].add(f[1])})
    .map(f -> f[0])
    .limit(100)
    .forEach(System.out::println);
```

Generate & print first 100 Fibonacci numbers
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method

```java
Stream.iterate(new BigInteger[] {BigInteger.ONE, BigInteger.ONE},
              f -> new BigInteger[] {f[1],
                                      f[0].add(f[1])})
      .map(f -> f[0])
      .limit(100)
      .forEach(System.out::println);
```

Generate & print first 100 Fibonacci numbers
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method

```java
Stream.iterate(new BigInteger[]{BigInteger.ONE, BigInteger.ONE},
    f -> new BigInteger[]{f[1],
                        f[0].add(f[1])})
  .map(f -> f[0])
  .limit(100)
  .forEach(System.out::println);
```

Generate & print first 100 Fibonacci numbers
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method

Generate & print first 100 Fibonacci numbers

```java
Stream.iterate(new BigInteger[]{BigInteger.ONE, BigInteger.ONE},
        f -> new BigInteger[]{f[1],
                      f[0].add(f[1])})
    .map(f -> f[0])
    .limit(100)
    .forEach(System.out::println);
```

Stream.iterate(new BigInteger[]{BigInteger.ONE, BigInteger.ONE},
        f -> new BigInteger[]{f[1],
                      f[0].add(f[1])})
    .map(f -> f[0])
    .limit(100)
    .forEach(System.out::println);
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method

```java
Stream.iterate(new BigInteger[]{BigInteger.ONE, 
  BigInteger.ONE},
  f -> new BigInteger[]{f[1], 
    f[0].add(f[1])})
  .map(f -> f[0])
  .limit(100)
  .forEach(System.out::println);
```

Generate & print first 100 Fibonacci numbers

See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#limit
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method

```java
Stream.iterate(new BigInteger[]{BigInteger.ONE, BigInteger.ONE},
    f -> new BigInteger[]{f[1], f[0].add(f[1])})
    .map(f -> f[0])
    .limit(100)
    .forEach(System.out::println);
```

Generate & print first 100 Fibonacci numbers
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
- From a Java collection
- From an array
- From a static factory method
- BufferedReader.lines() obtains lines of a file

```java
void printFileLines
    (String filename) {
    try (BufferedReader reader =
        Files.newBufferedReader
        (Paths.get(filename))) {
        reader
          .lines ()
          .forEach
            (System.out::println);
    } catch (IOException ex) {...}
}
```

Get & print all lines in a file
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file

```java
void printFileLines(String filename) {
    try (BufferedReader reader = Files.newBufferedReader(Paths.get(filename))) {
        reader.lines().forEach(System.out::println);
    } catch (IOException ex) {...}
}
```

Get & print all lines in a file
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods

```java
void printFileLines(String filename){
    try(Stream<String> stream = Files.lines(Paths.get(fileName))) {
        stream.forEach(System.out::println);
    } catch (IOException ex) {...}
}
```

Get & print all lines in a file
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()

```java
new Random()
  .ints(0,100)
  .limit(50)
  .forEach(System.out::println);
```
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()

```java
new Random()
    .ints(0, 100)
    .limit(50)
    .forEach(System.out::println);
```

Generate & print 50 random numbers between 0 & 100
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()

```java
new Random()
    .ints(0,100)
    .limit(50)
    .forEach(System.out::println);
```

Generate & print 50 random numbers between 0 & 100
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()

```java
new Random()
  .ints(0,100)
  .limit(50)
  .forEach(System.out::println);
```

Generate & print 50 random numbers between 0 & 100
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()
  - Other JDK stream-bearing methods

```java
Stream<String> getInputData(String filename, String splitter) {
    return Pattern.compile(splitter)
        .splitAsStream
        (new String(Files.readAllBytes(Paths.get(filename).toURI())));
}
```
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()
  - Other JDK stream-bearing methods

```java
Stream<String> getInputData
    (String filename,
     String splitter){
    return Pattern
               .compile(splitter)
               .splitAsStream
               (new String
                (Files.readAllBytes
                 (Paths.get(filename)
                  .toURI())));
```

*Splits a file into a stream of strings*
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()
  - Other JDK stream-bearing methods

```
List<TreeMap<Long, String>>
    listOfTreeMaps =
    Stream.generate
        (TreeMap<Long, String>::new)
        .limit(100)
        .collect(toList());
```

Generate a list of 100 TreeMaps

See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#generate
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()
  - Other JDK stream-bearing methods

```java
List<TreeMap<Long, String>>
listOfTreeMaps =
Stream.generate
    (TreeMap<Long, String>::new)
    .limit(100)
    .collect(toList());
```

_Generate a list of 100 TreeMaps_
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()
  - Other JDK stream-bearing methods

```java
List<TreeMap<Long, String>> listOfTreeMaps = Stream.generate
    (TreeMap<Long, String>::new)
    .limit(100)
    .collect(toList());
```

Generate a list of 100 TreeMaps
Common Factory Methods for Creating Streams

• Streams can be obtained various ways, e.g.,
  • From a Java collection
  • From an array
  • From a static factory method
  • BufferedReader.lines() obtains lines of a file
  • Streams of file paths & lines can be obtained from Files methods
  • A stream of random numbers can be obtained from Random.ints()
  • Other JDK stream-bearing methods

List<TreeMap<Long, String>>
listOfTreeMaps =
  Stream.generate
    (TreeMap<Long, String>::new)
    .limit(100)
    .collect(toList());

Generate a list of 100 TreeMaps
Common Factory Methods for Creating Streams

- Streams can be obtained various ways, e.g.,
  - From a Java collection
  - From an array
  - From a static factory method
  - BufferedReader.lines() obtains lines of a file
  - Streams of file paths & lines can be obtained from Files methods
  - A stream of random numbers can be obtained from Random.ints()
  - Other JDK stream-bearing methods

```java
List<TreeMap<Long, String>>
listOfTreeMaps =
    Stream.generate
        (TreeMap<Long, String>::new)
        .limit(100)
        .collect(toList());
```

**Generate a list of 100 TreeMaps**
Streams can be obtained various ways, e.g.,

- From a Java collection
- From an array
- From a static factory method
- BufferedReader.lines() obtains lines of a file
- Streams of file paths & lines can be obtained from Files methods
- A stream of random numbers can be obtained from Random.ints()
- Other JDK stream-bearing methods

```java
List<TreeMap<Long, String>>
listOfTreeMaps =
    Stream.generate
        (TreeMap<Long, String>::new)
        .limit(100)
        .collect(toList());
```

We’ll use this idiom in Java 8 programs covered later.
End of Java 8 Stream Factory Methods