Java Streams: The forEach() Terminal Operation

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 common terminal operations, e.g.
• `forEach()`

```java
void runForEach() {
    ...

    Stream.of("horatio", "laertes", "Hamlet", ...)
        .filter(s -> toLowerCase(s.charAt(0)) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEach(System.out::println);
    ...
}
```

We showcase `forEach()` using the Hamlet program

See github.com/douglascraigschmidt/LiveLessons/tree/master/Java8/ex12
Stream Terminal Operations That Return No Value
Several terminal operations return no value at all

```java
void runForEach() {
    ...

    Stream
        .of("horatio", "laertes", "Hamlet", ...)
        .filter(s -> toLowerCase(s.charAt(0)) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEach(System.out::println);
    ...
```

See [howtodoinjava.com/java8/foreach-method-example](http://howtodoinjava.com/java8/foreach-method-example)
Several terminal operations return no value at all

Several variants of `forEach()` are showcased in this example.

```java
void runForEach() {
...
Stream.of("horatio", "laertes", "Hamlet", ...)
    .filter(s -> toLowerCase(s.charAt(0)) == 'h')
    .map(this::capitalize)
    .sorted()
    .forEach(System.out::println);
...
}

See [github.com/douglasraigschmidt/LiveLessons/tree/master/Java8/ex12](github.com/douglasraigschmidt/LiveLessons/tree/master/Java8/ex12)
Stream Terminal Operations That Return No Value

- Several terminal operations return no value at all

```java
void runForEach() {
    ...

    Stream
        .of("horatio", "laertes", "Hamlet", ...)
        .filter(s -> toLowerCase(s.charAt(0)) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEach(System.out::println);
    ...
}
```
Several terminal operations return no value at all

```java
void runForEach() {
  ...
  Stream.of("horatio", "laertes", "Hamlet", ...)
    .filter(s -> toLowerCase(s.charAt(0)) == 'h')
    .map(this::capitalize)
    .sorted()
    .forEach(System.out::println);
  ...
}
```

See [docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#forEach](docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#forEach)
Stream Terminal Operations That Return No Value

- Several terminal operations return no value at all

```java
void runForEach() {
    List<String> results =
        new ArrayList<>();

    Stream.of("horatio", "laertes", "Hamlet", ...)
        .filter(s -> toLowerCase(s.charAt(0)) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEach(results::add);

    ...
}
```

The method reference passed to forEach() can have side-effects

Stream Terminal Operations That Return No Value

- Several terminal operations return no value at all

```java
void runForEach() {
    List<String> results =
        new ArrayList<>();

    Stream.of("horatio", "laertes", "Hamlet", ...)
        .filter(s -> toLowerCase(s.charAt(0)) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEach(results::add);
    ...
}
```

*Using forEach() in this manner is tricky, however, since programmers must remember to initialize the results object!*
Stream Terminal Operations That Return No Value

- Several terminal operations return no value at all

```java
void runForEach() {
    List<String> results =
    new ArrayList<>();

    Stream.of("horatio", "laertes", "Hamlet", ...)
        .parallel()
        .filter(s -> s.toLowerCase().charAt(0) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEach(results::add);
    ...
}
```

Likewise, using `forEach()` with side-effects in a parallel stream can incur nasty race conditions!!

See [docs.oracle.com/javase/tutorial/collections/stream/parallelism.html#side_effect](docs.oracle.com/javase/tutorial/collections/stream/parallelism.html#side_effect)
Several terminal operations return no value at all.

```java
void runForEach() {
    Queue<String> results = new ConcurrentLinkedQueue<>();
    Stream.of("horatio", "laertes", "Hamlet", ...)
        .parallel()
        .filter(s -> toLowerCase(s.charAt(0)) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEach(results::add);
    ...
}
```

`ConcurrentLinkedQueue` could be used here, but it’s still error-prone & inefficient.

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/ConcurrentLinkedQueue.html](http://docs.oracle.com/javase/8/docs/api/java/util/concurrent/ConcurrentLinkedQueue.html)
Several terminal operations return no value at all.

```java
void runForEach() {
    ...

    Stream.of("horatio", "laertes", "Hamlet", ...)
        .parallel()
        .filter(s -> toLowerCase(s.charAt(0)) == 'h')
        .map(this::capitalize)
        .sorted()
        .forEachOrdered(System.out::println);
    ...
}
```

Use `forEachOrdered()` if presenting the results in "encounter order" is important.

See [docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#forEachOrdered](http://docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#forEachOrdered)
Stream Terminal Operations That Return No Value

- Several terminal operations return no value at all

```java
void runForEach() {
  ...

  Stream.of("horatio", "laertes", "Hamlet", ...)
    .parallel()
    .filter(s -> toLowerCase(s.charAt(0)) == 'h')
    .map(this::capitalize)
    .sorted()
    .forEachOrdered(System.out::println);
  ...
```

forEachOrdered() is only really relevant for parallel streams.

See [docs.oracle.com/javase/8/docs/api/java/util/stream/BaseStream.html#parallel](http://docs.oracle.com/javase/8/docs/api/java/util/stream/BaseStream.html#parallel)
End of Java Streams: the forEach() Terminal Operation