Understand the Java Streams
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/douglasraigschmidt/LiveLessons/tree/master/Java8/ex12](https://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);
    // ...
}
```

See howtodoinjava.com/java8/foreach-method-example
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 variants of forEach() are showcased in this example.*

See [github.com/douglascraigschmidt/LiveLessons/tree/master/Java8/ex12](https://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);
    ...
```

Create & process a stream consisting of characters from the play “Hamlet”
Stream Terminal Operations That Return No Value

• Several terminal operations return no value at all

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

  Stream
    .of("horatio", "laertes", "Hamlet", ...) // Performs the designated action on each element of this stream
    .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](http://docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#forEach)
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

See www.logicbig.com/tutorials/core-java-tutorial/java-util-stream/side-effects
• Several terminal operations return no value at all

Using `forEach()` in this manner is tricky, however, since programmers must remember to initialize the results object!

```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);
    ...
}
```
Several terminal operations return no value at all.

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

```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);
    ...
}
```

See docs.oracle.com/javase/tutorial/collections/streams/parallelism.html#side_effects
Stream Terminal Operations That Return No Value

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
Several terminal operations return no value at all. Use `forEachOrdered()` if presenting the results in "encounter order" is important.

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

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

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