Understanding Java Streams
Common Creation Operations

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 Java streams structure & functionality, e.g.
- Fundamentals of streams
- Three streams phases
- Operations that create a stream
Operations that Create a Java Stream
The GoF *Factory Method* pattern defines an interface for creating an object, but shields implementation details from clients of this interface.

See [en.wikipedia.org/wiki/Factory_method_pattern](en.wikipedia.org/wiki/Factory_method_pattern)
Operations that Create a Java Stream

- Java Streams use factory methods to create a stream from some source

```
Stream.of("horatio",
         "laertes",
         "Hamlet",
         ...)
...
```

See [docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#of](docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#of)
Operations that Create a Java Stream

- Java Streams use factory methods to create a stream from some source

```
Stream.of("horatio", "laertes", "Hamlet", ...)
```

The `Stream.of()` factory method converts an array of T into a stream of T

See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#of
Operations that Create a Java Stream

- Many factory methods create streams

  ```java
  collection.stream()
  collection.parallelStream()
  ```

  These are the most common factory methods used in Streams

  ```java
  Pattern.compile(...)
  .splitAsStream()
  Stream.of(value1, ..., valueN)
  StreamSupport.stream(iterable.spliterator(), ...)
  ...```

  ```java
  Arrays.stream(array)
  Arrays.stream(array, start, end)
  Files.lines(file_path)
  "string".chars()
  Stream.iterate(init_value,
                 generate_expression)
  Stream.builder().add(...).build()
  Stream.generate(supplier)
  Files.list(file_path)
  Files.find(file_path, max_depth,
              matcher)
  ...
  ```

  These methods are inherited from the Java Collection interface
Many factory methods create streams

```java
collection.stream()
collection.parallelStream()

Pattern.compile(...) .splitAsStream()
Stream.of(value1, ..., valueN)
StreamSupport.stream (iterable.spliterator(), ...)

Arrays.stream(array)
Arrays.stream(array, start, end)
Files.lines(file_path) "string".chars()
Stream.iterate(init_value, generate_expression)
Stream.builder().add(...).build()
Stream.generate(supplier)
Files.list(file_path)
Files.find(file_path, max_depth, matcher)
```

We also show several examples of these factory methods throughout the course

See the upcoming lesson on "Java Streams: Common Factory Methods"
Operations that Create a Java Stream

- Many factory methods create streams

```java
collection.stream()
collection.parallelStream()
Pattern.compile(...)
    .splitAsStream()
Stream.of(value1, ..., valueN)
StreamSupport.stream
    (iterable.spliterator(), ...)
...
```

```java
Arrays.stream(array)
Arrays.stream(array, start, end)
Files.lines(file_path)
"string".chars()
Stream.iterate(init_value,
    generate_expression)
Stream.builder().add(...).build()
Stream.generate(supplier)
Files.list(file_path)
Files.find(file_path, max_depth,
    matcher)
...
```

There are also many other factory methods that create Java streams

See the upcoming lesson on "Java Streams: Other Factory Methods"
End of Understanding Java Streams Common Creation Operations