Overview of Java Streams Phases

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

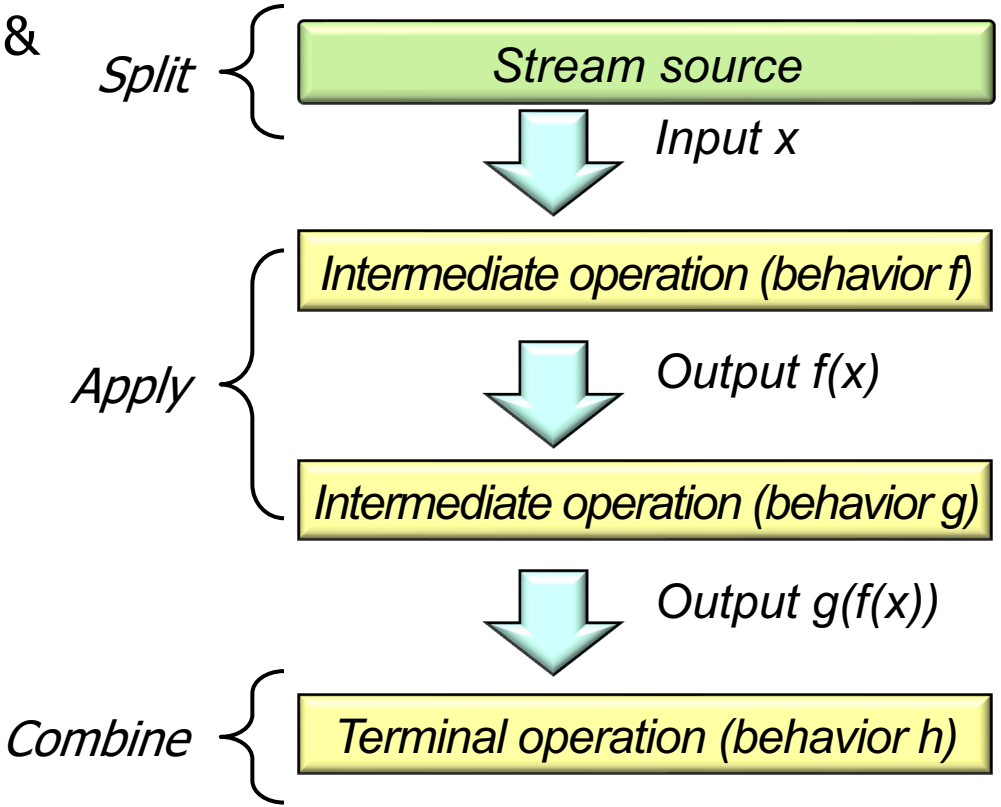
Professor of Computer Science

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



Overview of Stream Phases

Overview of Stream Phases

- Streams usually have three phases

See www.jstatsoft.org/article/view/v040i01/v40i01.pdf

Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data


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

e.g., a Java array, collection, generator function, or input channel

Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data

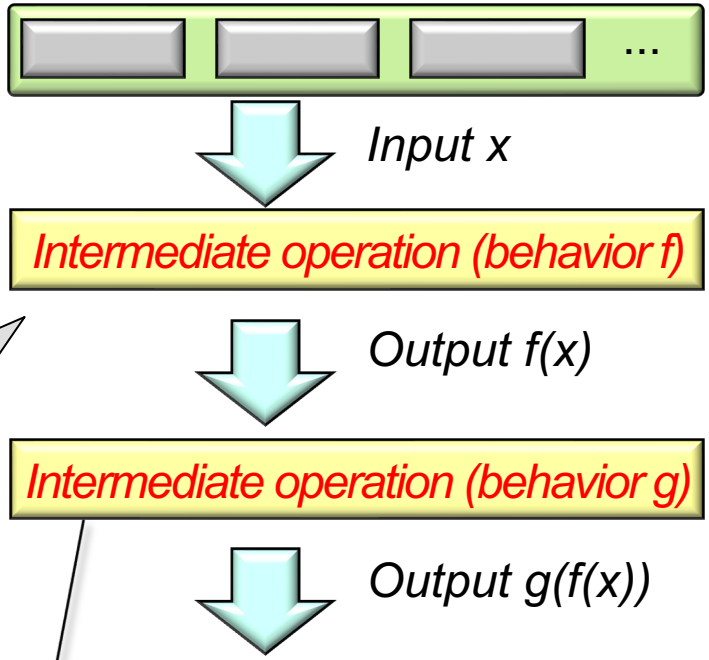

```
List<String> characters =
  List.of("horatio",
          "laertes",
          "Hamlet", ...);

characters
  .stream()
  ...
```

e.g., a Java array, collection, generator function, or input channel

Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data
  - **Apply** – process data through a pipeline of intermediate operations



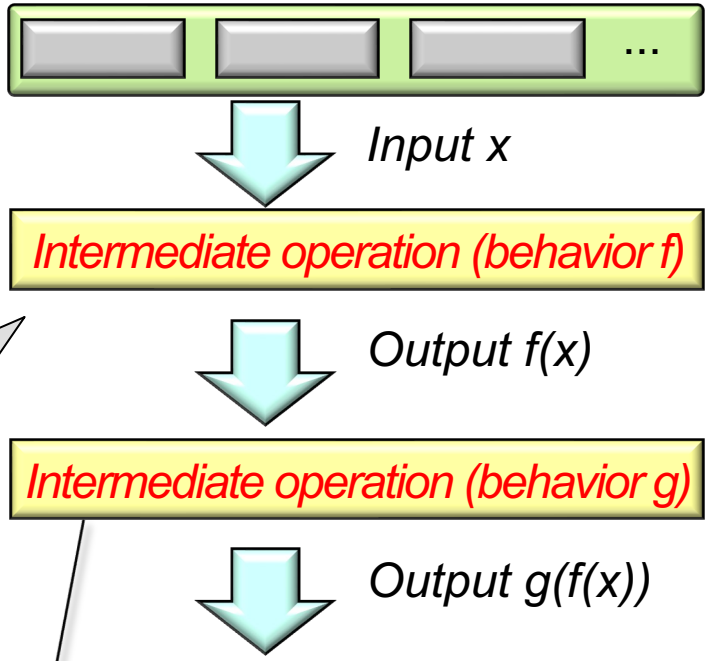

```
Stream
   .of("horatio", "laertes",
       "Hamlet", ...)
   .filter(s -> toLowerCase
               (s.charAt(0)) == 'h')
   .map(this::capitalize)
   .sorted()
   ...
```

Examples of intermediate operations include filter(), map(), & sorted()

Overview of Stream Phases

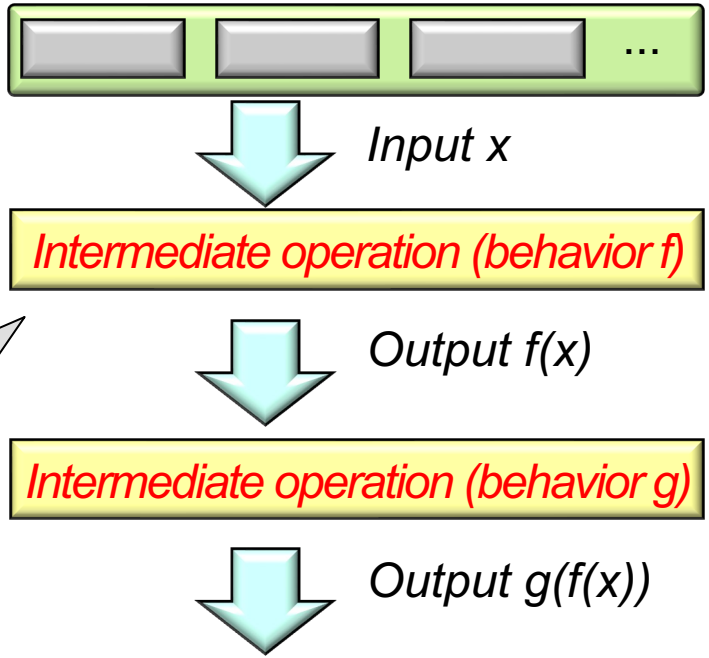
- Streams usually have three phases, i.e.
  - **Split** – start with a source of data
  - **Apply** – process data through a pipeline of intermediate operations
    - Processing often involves transforming

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



Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data
  - **Apply** – process data through a pipeline of intermediate operations
    - Processing often involves transforming



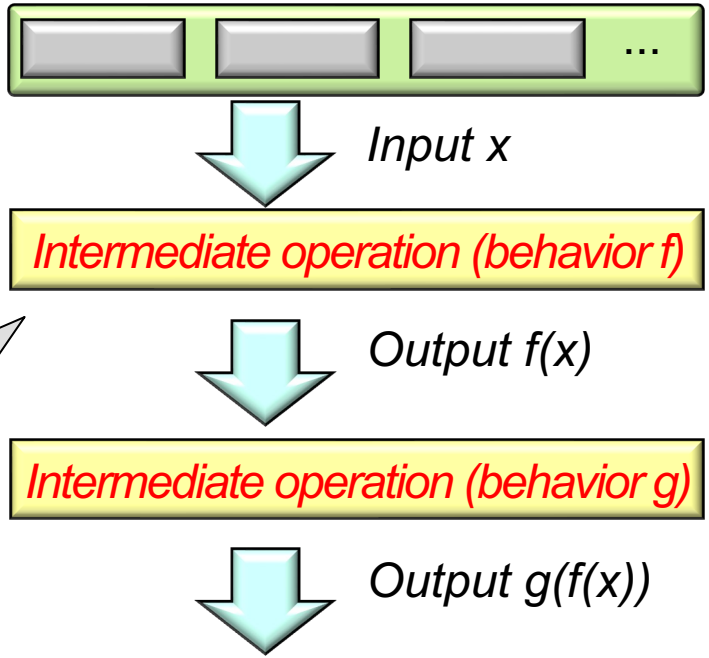

```
Stream
   .of("horatio", "laertes",
       "Hamlet", ...)
   .filter(s -> toLowerCase
               (s.charAt(0)) == 'h')
   .map(this::capitalize)
   .sorted()
   ...
```

Some operation transformations are "stateless"

See hajsoftutorial.com/java-stateless-stateful-operations

Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data
  - **Apply** – process data through a pipeline of intermediate operations
    - Processing often involves transforming



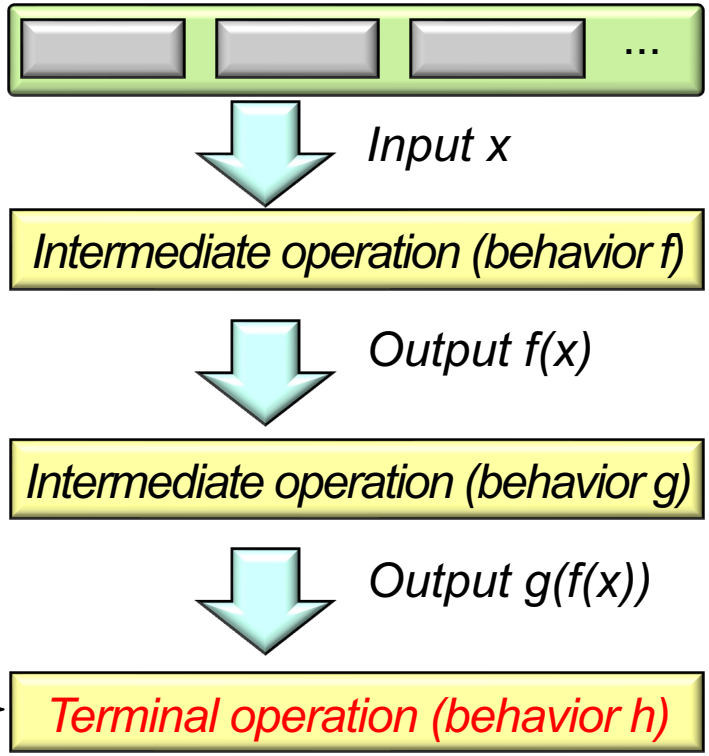

```
Stream
   .of("horatio", "laertes",
       "Hamlet", ...)
   .filter(s -> toLowerCase
               (s.charAt(0)) == 'h')
   .map(this::capitalize)
   .sorted()
   ...
```

Other operation transformations are "stateful"

See hajsoftutorial.com/java-stateless-stateful-operations

Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data
  - **Apply** – process data through a pipeline of intermediate operations
  - **Combine** – finish with a terminal operation that yields a non-stream result

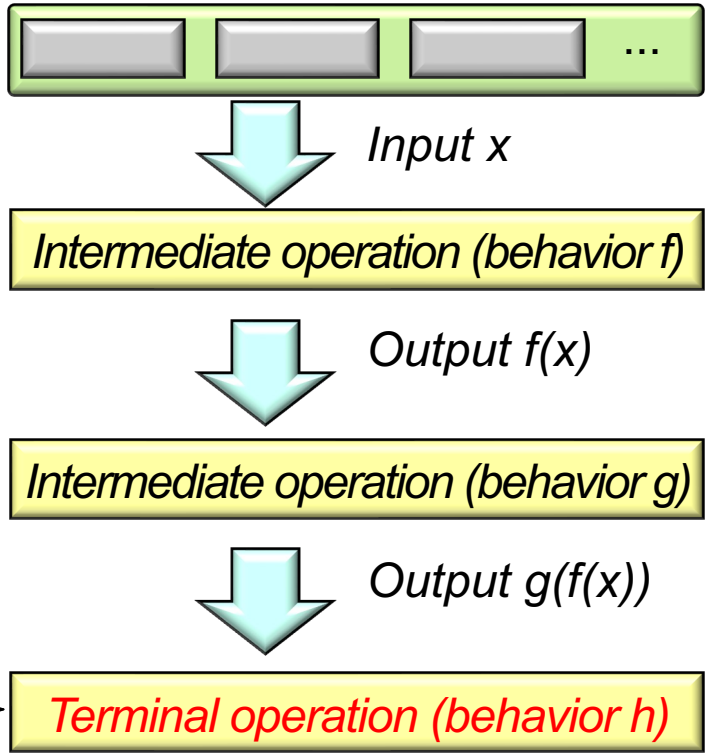



```
...
.filter(s -> toLowerCase
            (s.charAt(0)) == 'h')
.map(this::capitalize)
.sorted()
.forEach(System.out::println);
```

See jenkov.com/tutorials/java-functional-programming/streams.html#terminal-operations

Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data
  - **Apply** – process data through a pipeline of intermediate operations
  - **Combine** – finish with a terminal operation that yields a non-stream result




```
...
.filter(s -> toLowerCase
            (s.charAt(0)) == 'h')
.map(this::capitalize)
.sorted()
.forEach(System.out::println);
```

A terminal operation triggers processing of intermediate operations in a stream

Overview of Stream Phases

- Streams usually have three phases, i.e.
  - **Split** – start with a source of data
  - **Apply** – process data through a pipeline of intermediate operations
  - **Combine** – finish with a terminal operation that yields a non-stream result

A stream only runs if it has one (& only one) terminal operation

End of Overview of Java Streams Phases