Java Streams: Visualizing the WordSearcher.findWords() Method

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

• Recognize the structure & functionality of the SimpleSearchStream example

• Visualize aggregate operations in SimpleSearch Stream’s WordSearcher.findWords() method

```java
inputString

wordsToFind
"do","re","mi","fa","so","la","ti","do"

stream()

map(this::searchForWord)

filter(not(SearchResults::isEmpty))

collect(toList())
```

Let's start at the very beginning..

See [github.com/douglascraigschmidt/LiveLessons/tree/master/SimpleSearchStream](http://github.com/douglascraigschmidt/LiveLessons/tree/master/SimpleSearchStream)
Visualizing the Word Searcher.findWords() Method
Visualizing the `WordSearcher.findWords()` Method

- `WordSearcher.findWords()` searches for words in an input string

```
List<String>...
```

**Input a list of words to find**

```
"do", "re", "mi", "fa", "so", "la", "ti", "do"
```

WordSearcher.findWords() searches for words in an input string.

List `<String>`

wordsToFind

"do", "re", "mi", "fa", "so", "la", "ti", "do"

stream()
Visualizing the `WordSearcher.findWords()` Method

- `WordSearcher.findWords()` searches for words in an input string

Output a stream of words to find

```
List <String>
```

```
Stream <String>
```

```
"do", "re", "mi", "fa", "so", "la", "ti", "do"
```

```
wordsToFind
```

```
stream()
```

```
"do", "re", "mi", "fa", "so", "la", "ti", "do"
```
Visualizing the `WordSearcher.findWords()` Method

- `WordSearcher.findWords()` searches for words in an input string

Input a stream of words to find

```
List<String> stream() // ... wordsToFind
```

```
Stream<String> stream() // ... wordsToFind
```

```
map(this::searchForWord)
```

"do", "re", "mi", "fa", "so", "la", "ti", "do"
Visualizing the WordSearcher.findWords() Method

- WordSearcher.findWords() searches for words in an input string

```java
List<String>
Stream<String>
```

wordsToFind

"do", "re", "mi", "fa", "so", "la", "ti", "do"

Search for the word in the input string

stream()

map(this::searchForWord)
Visualizing the `WordSearcher.findWords()` Method

- `WordSearcher.findWords()` searches for words in an input string

```
output a stream of search results
```

- List `<String>`
- Stream `<String>`
- Stream `<SearchResults>`

```
wordsToFind = "do", "re", "mi", "fa", "so", "la", "ti", "do"
```

```
stream()
map(this::searchForWord)
```

```
<<Java Class>>
SearchResults

- SearchResults()
- SearchResults(long,long,String,String)
- getTitle():String
- headerToString():String
- add(int):void
- isEmpty():boolean
- size():int
- toString():String
- print():SearchResults
```

SearchResults stores # of times a word appeared in the input string
Visualizing the `WordSearcher.findWords()` Method

- `WordSearcher.findWords()` searches for words in an input string

```
Output a stream of search results
```

```
List <String>
Stream <String>
Stream <SearchResults>
```

```
"do", "re", "mi", "fa", "so", "la", "ti", "do"
```

```
stream() -> map(this::searchForWord)
```

Note the transformation of types at this point in the stream!
WordSearcher.findWords() searches for words in an input string.

```java
List<String>
Stream<String>
Stream<SearchResults>
```

Input a stream of search results

wordsToFind

"do", "re", "mi", "fa", "so", "la", "ti", "do"

Visualizing the WordSearcher.findWords() Method
WordSearcher.findWords() searches for words in an input string

```java
Stream<List<String>>
Stream<String>
List<String>

wordsToFind
"do", "re", "mi", "fa", "so", "la", "ti", "do"

remove empty search results from the stream
```
WordSearcher.findWords() searches for words in an input string

Output a stream of non-empty search results

List
<String>

Stream
<String>

Stream
<SearchResults>

Stream
<SearchResults>

wordsToFind
"do", "re", "mi", "fa", "so", "la", "ti", "do"

stream()

map(this::searchForWord)

filter(not(SearchResults::isEmpty))
Visualizing the WordSearcher.findWords() Method

- WordSearcher.findWords() searches for words in an input string

```
Input a stream of non-empty search results
```
WordSearcher.findWords() searches for words in an input string:

```java
Stream<String> wordsToFind = ...
.stream()
.map(this::searchForWord)
.filter(not(SearchResults::isEmpty))
.collect(toList());
```

Trigger intermediate operation processing.
Visualizing the WordSearcher.findWords() Method

- WordSearcher.findWords() searches for words in an input string

```
Stream<String>
map(this::searchForWord)
filter(not(SearchResults::isEmpty))
collect(toList())
```

List<String>

List<SearchResults>

List<SearchResults>

```
"do", "re", "mi", "fa", 
"so", "la", "ti", "do"
```

Return a list of non-empty search results
Visualizing the WordSearcher.findWords() Method

- The “physical” processing of a stream differs from the “logical” model

It may appear that each “row” of data is processed from “left to right”

List
<String>

Stream
<String>

Stream
<SearchResults>

Stream
<SearchResults>

List
<SearchResults>

wordsToFind
"do", "re", "mi", "fa", "so", "la", "ti", "do"

stream()

map(this::searchForWord)

filter(not(SearchResults::isEmpty))

collect(toList())

The “physical” processing of a stream differs from the “logical” model.

However, each element is actually “pulled” from the source through each aggregate operation.

This implementation is much more efficient & supports “short-circuit” operations.

Visualizing the WordSearcher.findWords() Method

wordsToFind
"do", "re", "mi", "fa", "so", "la", "ti", "do"

stream()
map(this::searchForWord)
filter(not(SearchResults::isEmpty))
collect(toList())
End of Java Streams: Visualizing the Word Searcher.findWords() Method