Java Streams: Overview of the SimpleSearchStream Program

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 program

We use this program to showcase key Java sequential streams capabilities.

See [github.com/douglascraigschmidt/LiveLessons/tree/master/SimpleSearchStream](github.com/douglascraigschmidt/LiveLessons/tree/master/SimpleSearchStream)
Visualizing the Simple SearchStream program
Visualizing the SimpleSearchStream Program

- This program finds words in an input string

inputString

wordsToFind

stream()

map(this::searchForWord)

filter(not(SearchResults::isEmpty))

collect(toList())

See [github.com/douglascraigschmidt/LiveLessons/tree/master/SimpleSearchStream](github.com/douglascraigschmidt/LiveLessons/tree/master/SimpleSearchStream)
This program finds words in an input string.

Let's start at the very beginning…

- **inputString**
- **wordsToFind**
  - "do", "re", "mi", "fa", "so", "la", "ti", "do"

1. `stream()`
2. `map(this::searchForWord)`
3. `filter(not(SearchResults::isEmpty))`
4. `collect(toList())`

See [en.wikipedia.org/wiki/Do-Re-Mi](en.wikipedia.org/wiki/Do-Re-Mi)
Visualizing the SimpleSearchStream Program

- This program finds words in an input string

```
See SimpleSearchStream/src/main/java/search/WordSearcher.java
```

```java
inputString

map(this::searchForWord)

filter(not(SearchResults::isEmpty))

collect(toList())

wordsToFind

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

Let's start at the very beginning…

It showcases Java functional programming features (lambda expressions and method references) along with Java sequential streams.
This program finds words in an input string.

Stepping through the code:

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

inputString = "Let's start at the very beginning..."

Starting SimpleSearchStream
... Word "Re" matched at index [131, 141, 151, 202, 212, 222, 979, 1025, 1219, 1259, 1278, 1300, 1351, 1370, 1835, 1875, 1899, 1939, 2266, 2295]


Word "La" matched at index [234, 417, 658, 886, 991, 1207, 1247, 1269, 1291, 1339, 1361, 1742, 1847, 1863, 1909, 1949, 2161, 2254, 2276, 2283]...

Ending SimpleSearchStream
```

The program produces nicely formatted output.
Visualizing the SimpleSearchStream Program

- Also prints a slice of the search results starting at a particular word, e.g., “La”

Starting SimpleSearchStream...
Word "La" appeared at indices [234|417|658|886|991|1207|1247|1269|1291|1339|1361|1742|1847|1863|1909|1949|2161|2254|2276|2283]
Word "Ti" appeared at indices [237|994|1272|1294|1364|1850|1860|1912|1915|1952|1955|2299]
... Ending SimpleSearchStream

Print out results of each map entry (key = word & value = list of search results).
Entry Point Into the Simple SearchStream Program
Entry Point Into the SimpleSearchStream Program

- It searches sequentially for words in a string containing the contents of a file

```java
static public void main(String[] args) { ... 
    String input = TestDataFactory
        .getInput(sINPUT_FILE, "@").get(0);

    List<String> wordsToFind = TestDataFactory
        .getWordList(sWORD_LIST_FILE);

    WordSearcher searcher =
        new WordSearcher(input);

    List<SearchResults> results =
        searcher.findWords(wordsToFind);

    searcher.printResults(results); ... 
}
```

See SimpleSearchStream/src/main/java/Main.java
Entry Point Into the SimpleSearchStream Program

- It searches sequentially for words in a string containing the contents of a file.

```java
static public void main(String[] args) {
    String input = TestDataFactory.getInput(sINPUT_FILE, "@").get(0);

    List<String> wordsToFind = TestDataFactory.getWordList(sWORD_LIST_FILE);

    WordSearcher searcher = new WordSearcher(input);

    List<SearchResults> results = searcher.findWords(wordsToFind);

    searcher.printResults(results); ...
}
```

Create an input string containing the lyrics to the do-re-mi song.

See SimpleSearchStream/src/main/java/utils/TestDataFactory.java
Entry Point Into the SimpleSearchStream Program

- It searches sequentially for words in a string containing the contents of a file.

```java
static public void main(String[] args) { ...
    String input = TestDataFactory
        .getInput(sINPUT_FILE, "@").get(0);

    List<String> wordsToFind = TestDataFactory
        .getWordList(sWORD_LIST_FILE);

    WordSearcher searcher =
        new WordSearcher(input);

    List<SearchResults> results =
        searcher.findWords(wordsToFind);

    searcher.printResults(results); ...
```

See SimpleSearchStream/src/main/java/utils/TestDataFactory.java
Entry Point Into the SimpleSearchStream Program

It searches sequentially for words in a string containing the contents of a file.

```java
static public void main(String[] args) {
    String input = TestDataFactory
        .getInput(sINPUT_FILE, "@").get(0);

    List<String> wordsToFind = TestDataFactory
        .getWordList(sWORD_LIST_FILE);

    WordSearcher searcher =
        new WordSearcher(input);

    List<SearchResults> results =
        searcher.findWords(wordsToFind);

    searcher.printResults(results); ...
```

Create an object used to search for words in the input string.

Entry Point Into the SimpleSearchStream Program

- It searches sequentially for words in a string containing the contents of a file

```java
static public void main(String[] args) { ...
    String input = TestDataFactory
        .getInput(sINPUT_FILE, "@").get(0);

    List<String> wordsToFind = TestDataFactory
        .getWordList(sWORD_LIST_FILE);

    WordSearcher searcher =
        new WordSearcher(input);

    List<SearchResults> results =
        searcher.findWords(wordsToFind);

    searcher.printResults(results); ...
```
Entry Point Into the SimpleSearchStream Program

- It searches sequentially for words in a string containing the contents of a file

```java
static public void main(String[] args) {
    String input = TestDataFactory.getInput(sINPUT_FILE, "@").get(0);

    List<String> wordsToFind = TestDataFactory.getWordList(sWORD_LIST_FILE);

    WordSearcher searcher =
        new WordSearcher(input);

    List<SearchResults> results =
        searcher.findWords(wordsToFind);

    searcher.printResults(results); ...
```

Print all matching words.
End of Java Streams: Overview of the Simple SearchStream Program