

Implementing WordSearcher

`.printSuffixSlice()`

Douglas C. Schmidt

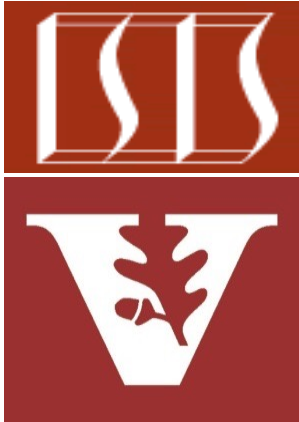
d.schmidt@vanderbilt.edu

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

- Visualize aggregate operations in SimpleSearchStream's WordSearcher .printResults() method
- Understand the implementation of aggregate operations in SimpleSearch Stream's WordSearcher.printSuffixSlice() method

```
void printSuffixSlice(String word, List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                            LinkedHashMap::new,
                            toDownstreamCollector()))
        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

See [SimpleSearchStream/src/main/java/search/WordSearcher.java](https://github.com/oreilind/java8/blob/master/src/main/java/search/WordSearcher.java)

Implementing the Word Searcher.printSuffixSlice() Method

Implementing the `WordSearcher.printSuffixSlice()` Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))

        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

This method shows `collect(groupingBy())`, `dropWhile()`, & `forEach()`

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))

        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

Convert the list param into a stream.

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))
        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

Collect SearchResults into a Map, with word as the key & the list of indices as the value.

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))
        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

LinkedHashMap preserves the insertion order.

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))
        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

This factory method creates a downstream collector that merges results lists together.

See upcoming lesson on "Java Streams: Implementing Custom Non-Concurrent Collectors"

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))
        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

*Get the EntrySet for this map
& convert it into a stream.*

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))
        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

Slice the stream to contain remaining elements after dropping subset of elements that don't match 'word'.

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))

        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

Print out the matching results in the stream.

Implementing the `WordSearcher.printSuffixSlice()` Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))

        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

See earlier lesson on *"Implementing the `WordSearcher.printResults()` Method"*

Implementing the WordSearcher.printSuffixSlice() Method

- Print a slice of the list of results starting at a particular word

```
public void printSuffixSlice(String word,
                             List<SearchResults> results) {
    results
        .stream()
        .collect(groupingBy(SearchResults::getWord,
                             LinkedHashMap::new,
                             toDownstreamCollector()))

        .entrySet()
        .stream()
        .dropWhile(e -> notEqual(e, word))
        .forEach(e -> printResult(e.getKey(), e.getValue()));
}
```

This is Stream's forEach() not Map's forEach()!

See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#forEach

Implementing the WordSearcher.printSuffixSlice() Method

- Returns true if entry.getKey() != to word, else false

```
private boolean notEqual
    (Map.Entry<String, List<SearchResults.Result>> entry,
     String word) {

    return !entry.getKey().equals(word);
}
```

Implementing the WordSearcher.printSuffixSlice() Method

- Returns true if entry.getKey() != to word, else false

```
private boolean notEqual  
    (Map.Entry<String, List<SearchResults.Result>> entry,  
     String word) {
```

Params are the map entry & the word to match

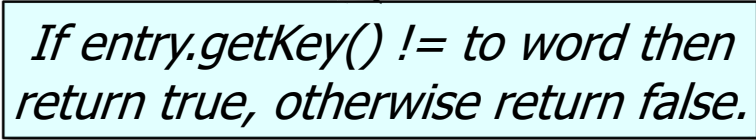
```
    return !entry.getKey().equals(word);  
}
```

Implementing the WordSearcher.printSuffixSlice() Method

- Returns true if entry.getKey() != to word, else false

```
private boolean notEqual
    (Map.Entry<String, List<SearchResults.Result>> entry,
     String word) {

    return !entry.getKey().equals(word);
}
```



If entry.getKey() != to word then return true, otherwise return false.

End of Implementing Word
Searcher.printSuffixSlice()