Java Sequential SearchStreamGang Example: Helper Methods

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

- Know how to apply sequential streams to the SearchStreamGang program
  - Understand key helper method implementations in SearchStreamGang

```
void main(String[] args) { ...
  List<List<CharSequence>> input =
    new ArrayList<List<CharSequence>>() {{
      add(TestDataFactory.getSharedInput
           (sSHAKESPEARE_DATA_FILE, "@"));
    }};

  List<String> wordsToFind =
    TestDataFactory.getPhraseList(sPHRASE_LIST_FILE);

  runTests(phaseList, input);
  ...
```

These helper methods use Java aggregate operations in sequential streams

Helper Methods in the SearchStreamGang Program

Helper Methods in the SearchStreamGang Program

- The program gets the input & list of phrases from 2 text files
  - Each work within an input file is separated by a '@' character

```
...
@The Tragedy of Hamlet
...
@The Tragedy of Julius Caesar
...
@The Tragedy of Macbeth
...
```

```
void main(String[] args) { ...
  List<List<CharSequence>> input =
    new ArrayList
      <List<CharSequence>>() {{
      add(TestDataFactory
        .getSharedInput
           (sSHAKESPEARE_DATA_FILE,
            "@"));
  }};

  List<String> wordsToFind =
    TestDataFactory.getPhraseList
      (sPHRASE_LIST_FILE);
  ...
```

Helper Methods in the SearchStreamGang Program

- The program gets the input & list of phrases from 2 text files

```
...
Beware the Ides of March
Brevity is the soul of wit
All that glisters is not gold
Sit you down, father; rest you
my kingdom for a horse!
...
```

```
void main(String[] args) { ...
  List<List<CharSequence>> input =
    new ArrayList
      <List<CharSequence>>() {{
      add(TestDataFactory
        .getSharedInput
           (sSHAKESPEARE_DATA_FILE,
            "@"));
    }};

  List<String> wordsToFind =
    TestDataFactory.getPhraseList
      (sPHRASE_LIST_FILE);
  ...
```

Each phrase appears on a separate line

Helper Methods in the SearchStreamGang Program

- Return the input data in the given file as a list of strings

```
static List<CharSequence> getInput(String file, String split) {
  URI uri = ClassLoader.getSystemResource(file).toURI();


  String bytes = new String(Files.readAllBytes
                                 (Paths.get(uri)));

  return Pattern
    .compile(split)
    .splitAsStream(bytes)
    .filter(((Predicate<String>) String::isEmpty).negate())
    .collect(toList());
  ...
```

See SearchStreamGang/src/main/java/utils/TestDataFactory.java

Helper Methods in the SearchStreamGang Program

- Return the input data in the given file as a list of strings

```
static List<CharSequence> getInput(String file, String split) {
  URI uri = ClassLoader.getSystemResource(file).toURI();
```

Convert the file name into a path name

```
  String bytes = new String(Files.readAllBytes
                                (Paths.get(uri)));

  return Pattern
    .compile(split)
    .splitAsStream(bytes)
    .filter(((Predicate<String>) String::isEmpty).negate())
    .collect(toList());
  ...
```

Helper Methods in the SearchStreamGang Program

- Return the input data in the given file as a list of strings

```
static List<CharSequence> getInput(String file, String split) {
  URI uri = ClassLoader.getSystemResource(file).toURI();


  String bytes = new String(Files.readAllBytes
                                   (Paths.get(uri)));

  return Pattern
    .compile(split)
    .splitAsStream(bytes)
    .filter(((Predicate<String>) String::isEmpty).negate())
    .collect(toList());
  ...
```

Open the file & read all the bytes

Helper Methods in the SearchStreamGang Program

- Return the input data in the given file as a list of strings

```
static List<CharSequence> getInput(String file, String split) {
  URI uri = ClassLoader.getSystemResource(file).toURI();


  String bytes = new String(Files.readAllBytes
                                   (Paths.get(uri)));

  return Pattern
    .compile(split)
    .splitAsStream(bytes)
    .filter(((Predicate<String>) String::isEmpty).negate())
    .collect(toList());
  ...
```

Compile a regular expression used to split the file into a list of strings

Helper Methods in the SearchStreamGang Program

- Return the input data in the given file as a list of strings

```
static List<CharSequence> getInput(String file, String split) {
  URI uri = ClassLoader.getSystemResource(file).toURI();


  String bytes = new String(Files.readAllBytes
                                  (Paths.get(uri)));

  return Pattern
    .compile(split)
    .splitAsStream(bytes)
    .filter(((Predicate<String>) String::isEmpty).negate())
    .collect(toList());
  ...
```

Filter out any empty strings in the stream

Helper Methods in the SearchStreamGang Program

- Return the input data in the given file as a list of strings

```
static List<CharSequence> getInput(String file, String split) {
  URI uri = ClassLoader.getSystemResource(file).toURI();



  String bytes = new String(Files.readAllBytes
                                   (Paths.get(uri)));

  return Pattern
    .compile(split)
    .splitAsStream(bytes)
    .filter(((Predicate<String>) String::isEmpty).negate())
    .collect(toList());
  ...
```

Collect results into a list of strings

Helper Methods in the SearchStreamGang Program

- Return the input data in the given file as a list of strings

```
static List<CharSequence> getInput(String file, String split) {
  URI uri = ClassLoader.getSystemResource(file).toURI();


  String bytes = new String(Files.readAllBytes
                                     (Paths.get(uri)));

  return Pattern
    .compile(split)
    .splitAsStream(bytes)
    .filter(((Predicate<String>) String::isEmpty).negate())
    .map(string -> new SharedString(string.toCharArray()))
    .collect(toList());
  ...
```

An optimization could map each string to a SharedString to eliminate copying overhead.

See SearchStreamGang/src/main/java/livelessons/utils/SharedString.java

Helper Methods in the SearchStreamGang Program

- Return the phrase list in the file as a list of non-empty strings

```
static List<String> getPhraseList(String file) {
  return Files
    .lines(Paths
          .get(ClassLoader.getSystemResource(file).toURI()))



    .filter(((Predicate<String>) String::isEmpty).negate())


    .collect(toList());
}
```

See SearchStreamGang/src/main/java/utils/TestDataFactory.java

Helper Methods in the SearchStreamGang Program

- Return the phrase list in the file as a list of non-empty strings

```
static List<String> getPhraseList(String file) {
  return Files
    .lines(Paths
           .get(ClassLoader.getSystemResource(file).toURI()))

    .filter(((Predicate<String>) String::isEmpty).negate())


    .collect(toList());
}
```

Read all lines from file into a stream

Helper Methods in the SearchStreamGang Program

- Return the phrase list in the file as a list of non-empty strings

```
static List<String> getPhraseList(String file) {
  return Files
    .lines(Paths
          .get(ClassLoader.getSystemResource(file).toURI()))
```

Filter out any empty strings in the stream

```
    .filter(((Predicate<String>) String::isEmpty).negate())

    .collect(toList());
}
```

Helper Methods in the SearchStreamGang Program

- Return the phrase list in the file as a list of non-empty strings

```
static List<String> getPhraseList(String file) {
  return Files
    .lines(Paths
          .get(ClassLoader.getSystemResource(file).toURI()))



    .filter(((Predicate<String>) String::isEmpty).negate())


    .collect(toList());
}
```

Collect the results into a list of strings

End of Java Sequential SearchStreamGang Example: Helper Methods