Java 8 Sequential SearchStreamGang Example (Part 2)

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 the SearchStreamGang printPhrases() method

```
void printPhrases(List<List<SearchResults>>
                  listOfListOfSearchResults) {
  Map<String, List<SearchResults>> resultsMap =
    listOfListOfSearchResults
      .stream()
      .flatMap(List::stream)
      .collect(groupingBy(SearchResults::getTitle));

  resultsMap.forEach((key, value) -> {
    System.out.println("Title \"" + key + "\" contained");
                       value.forEach(SearchResults::print);
  });
}
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/SearchStreamGang

Visualizing printPhrases()

Visualizing printPhrases()

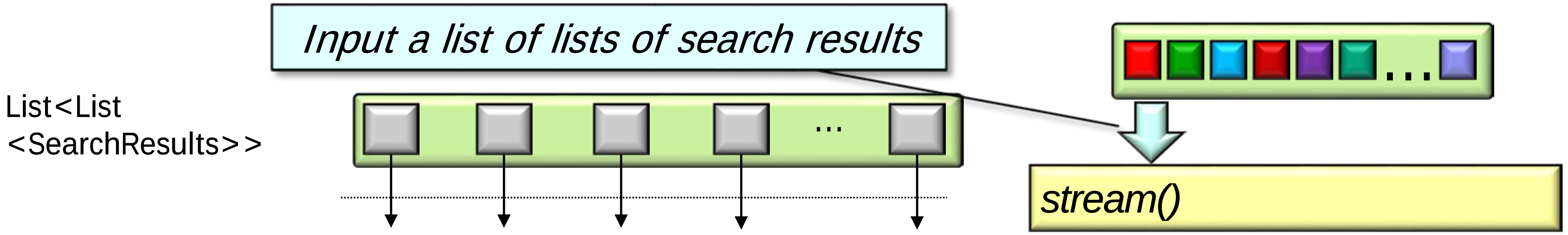
- SearchStreamGang.printPhrases() displays phrases associated with each play

```
...
Title "The Tragedy of Hamlet, Prince of Denmark" contained
"It shall be so. Madness in great ones must not unwatch'd go." at [89594]
"Give every man thine ear, but few thy voice" at [26207]
"There is nothing either good or bad but thinking makes it so" at [62609]
"To be, or not to be- that is the question" at [83061]
"Neither a borrower nor a lender be" at [26556]
"This above all- to thine own self be true, And it must follow, as the night the day,
Thou canst not then be false to any man" at [26693]
"Frailty, thy name is woman" at [17233]
"The lady doth protest too much, methinks" at [102267]
"Get thee to a nunnery" at [86071|86953]
"Brevity is the soul of wit" at [54747]
...
```

This method shows the flatMap() & collect(groupingBy()) aggregate operations

Visualizing printPhrases()

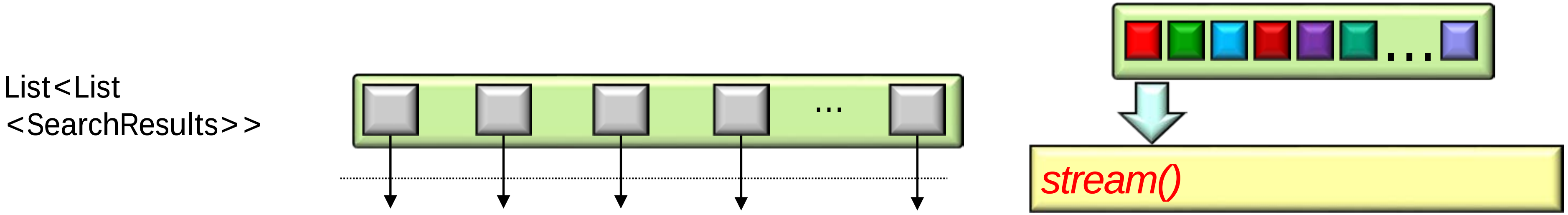
- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



Input a list of lists of search results
List<List <SearchResults>>
...
...
stream()

Visualizing printPhrases()

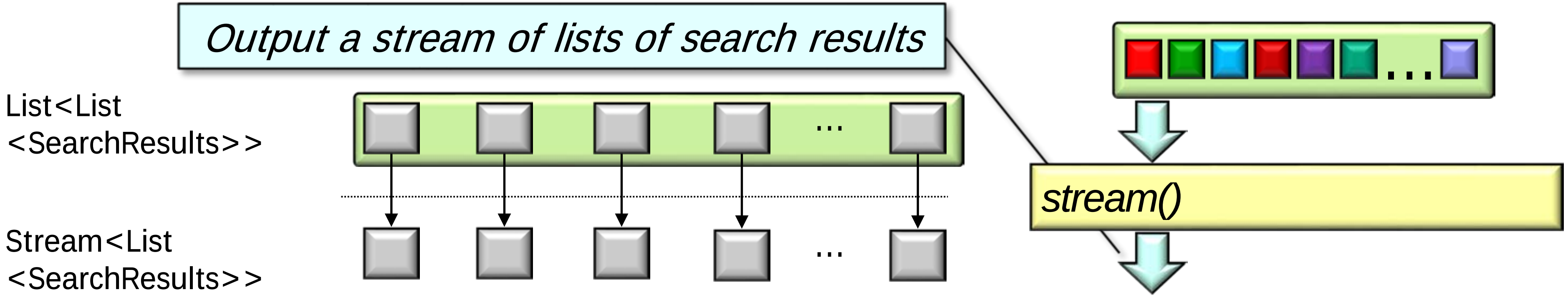
- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



Convert list to a (sequential) stream of lists of search results

Visualizing printPhrases()

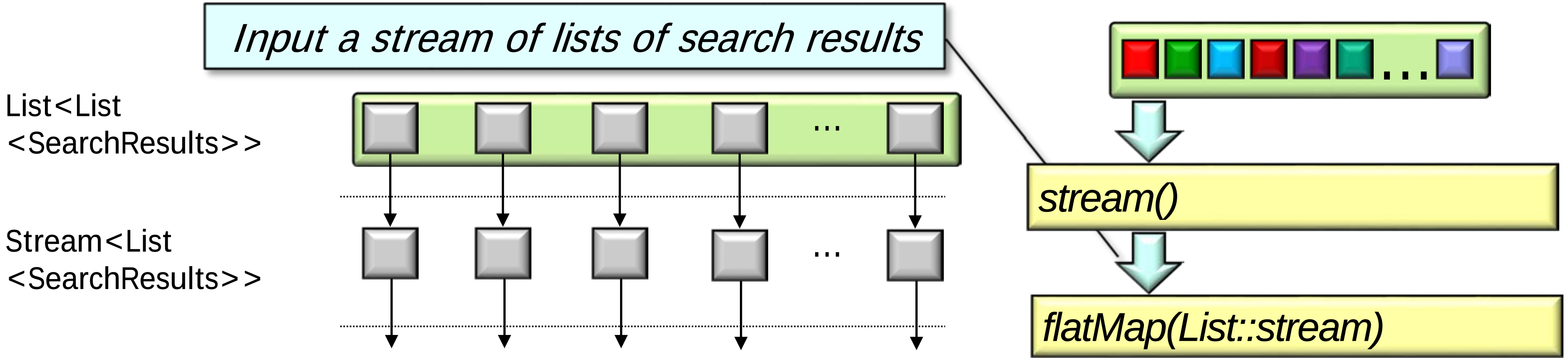
- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



Output a stream of lists of search results
List<List <SearchResults>>
Stream<List <SearchResults>>
…
…
stream()

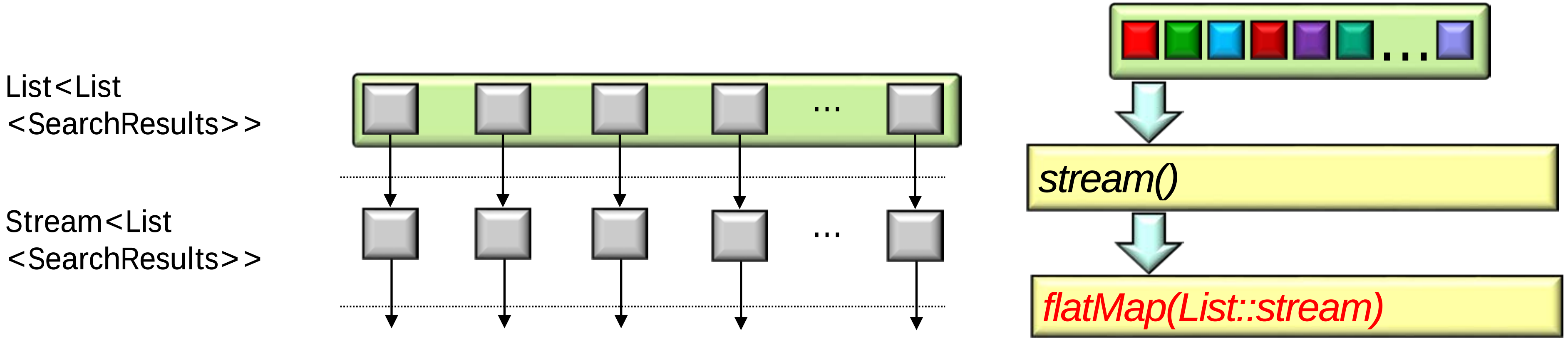
Visualizing printPhrases()

- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found


Input a stream of lists of search results
List<List <SearchResults>>
Stream<List <SearchResults>>
…
stream()
flatMap(List::stream)

Visualizing printPhrases()

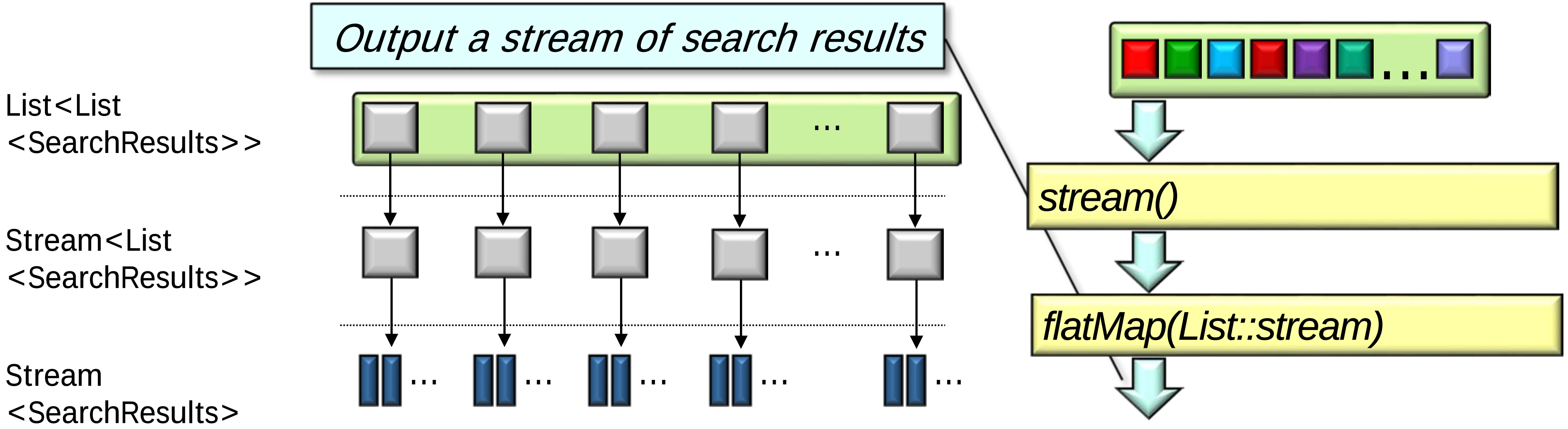
- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



Flatten the stream of lists of search results to a stream of search results

Visualizing printPhrases()

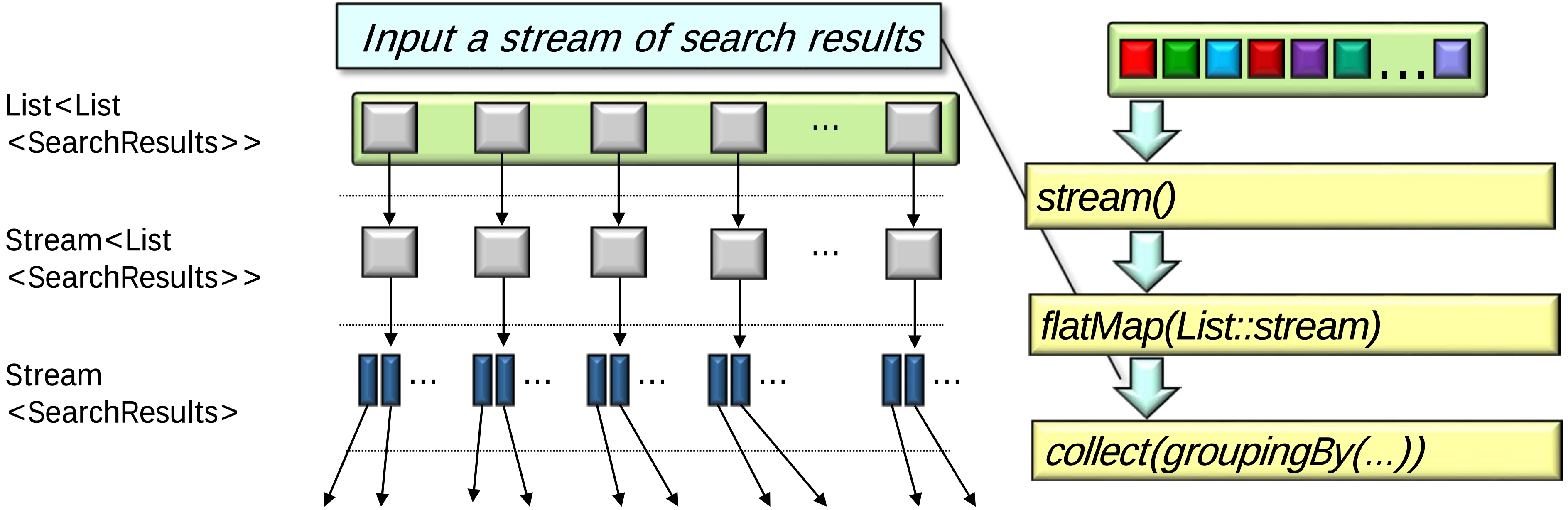
- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



Output a stream of search results
List<List <SearchResults>>
Stream<List <SearchResults>>
Stream <SearchResults>
...
stream()
flatMap(List::stream)

Visualizing printPhrases()

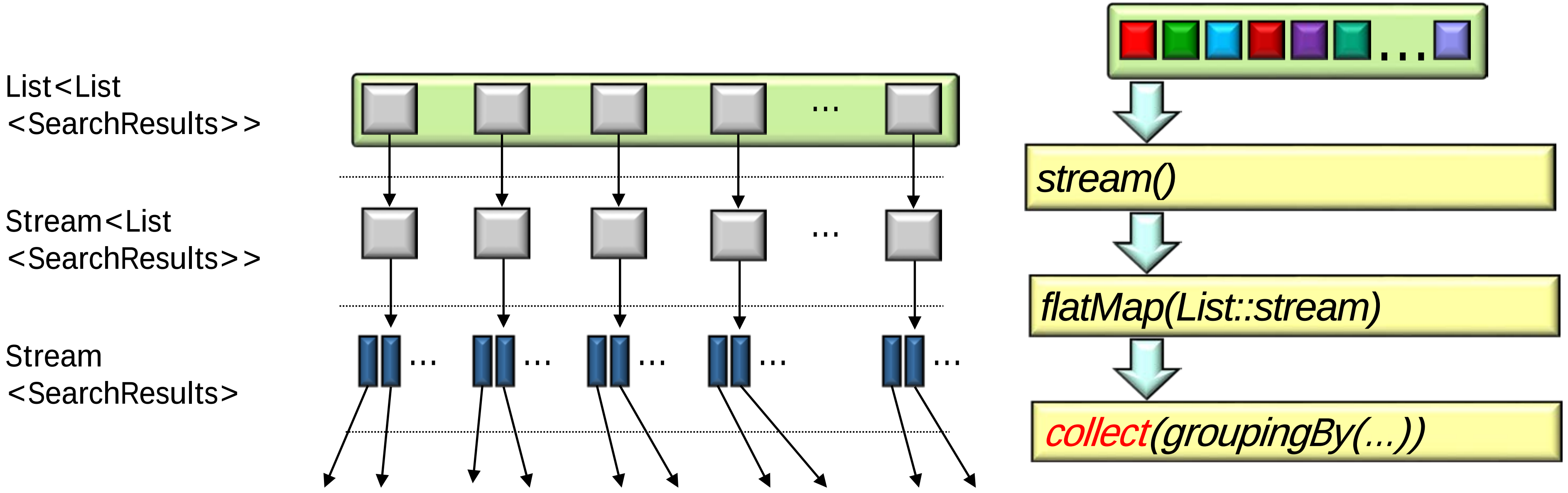
- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



Input a stream of search results
List<List <SearchResults>>
Stream<List <SearchResults>>
Stream <SearchResults>
stream()
flatMap(List::stream)
collect(groupingBy(...))

Visualizing printPhrases()

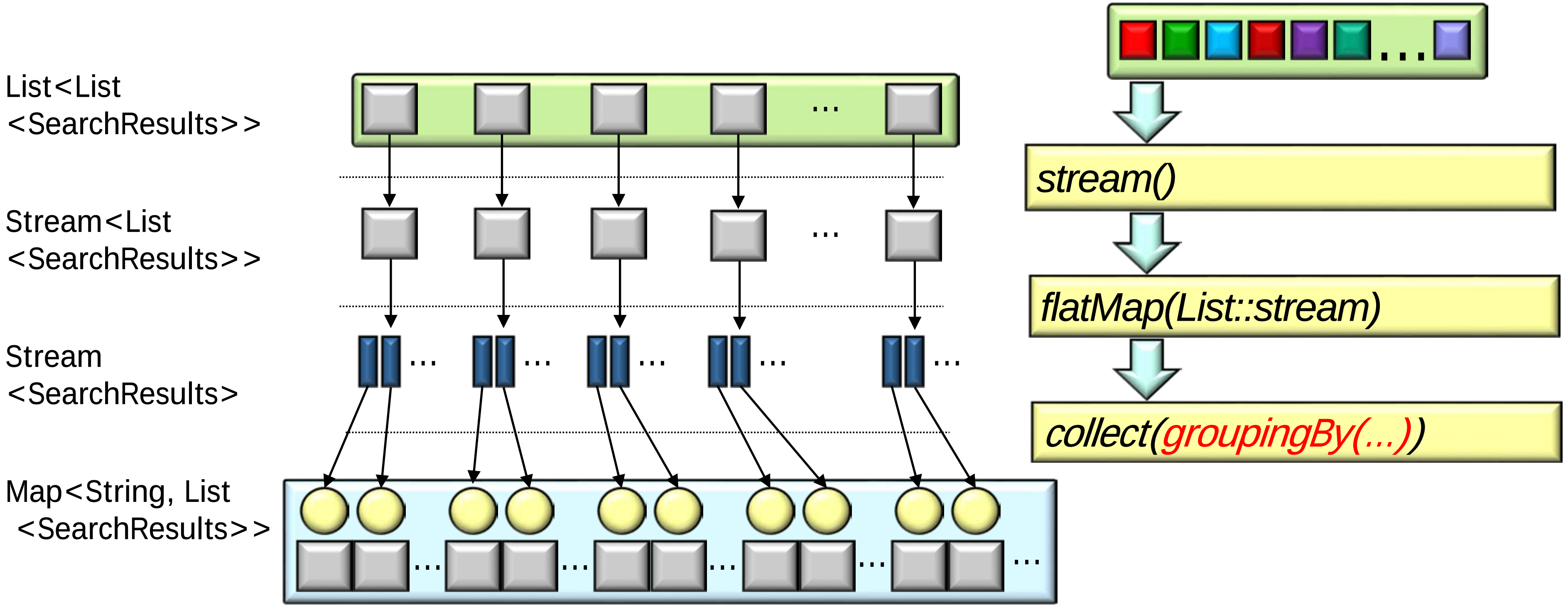
- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



Trigger intermediate operation processing

Visualizing printPhrases()

- printPhrases() uses a stream that converts a list of lists of search results into a map that associates phrases with the plays where they were found



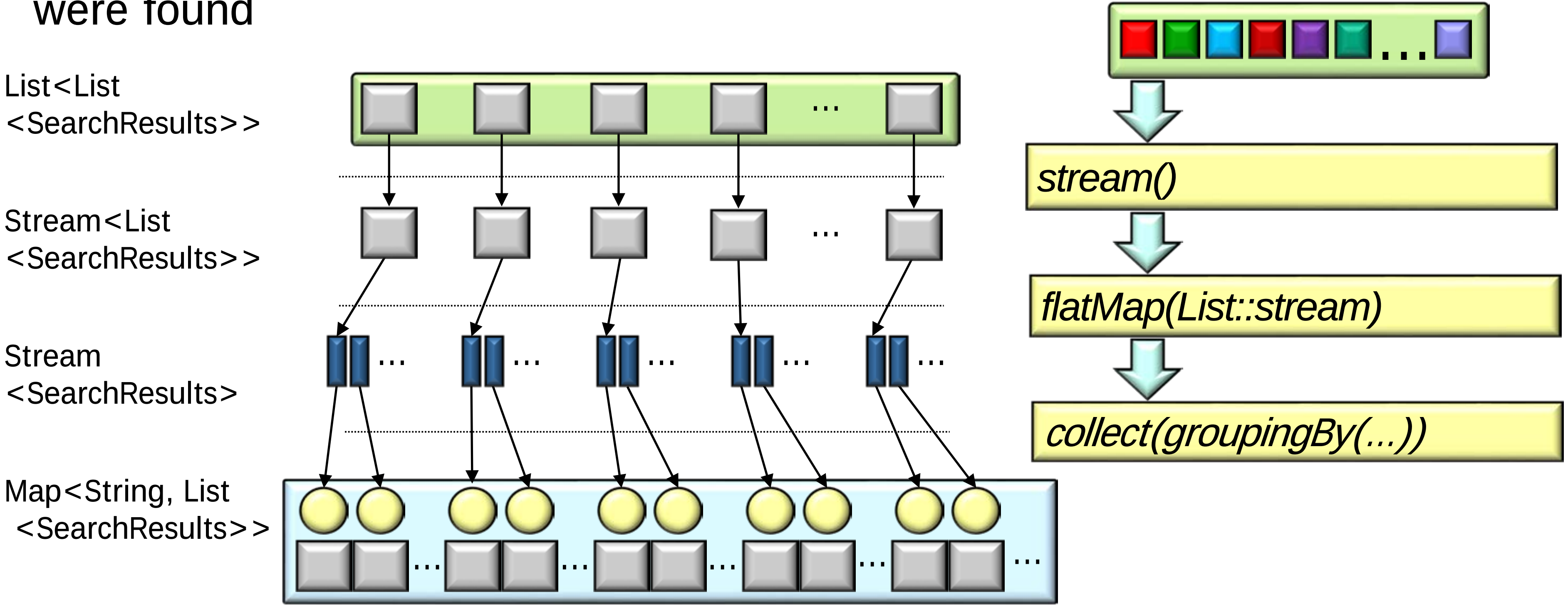
List<List <SearchResults>>
Stream<List <SearchResults>>
Stream <SearchResults>
Map<String, List <SearchResults>>
…
stream()
flatMap(List::stream)
collect(groupingBy(…))

Create a map that groups phrases according to the plays where they appear

Implementing printPhrases() as a Sequential Stream

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to converts a list of lists of search results to a map that associates phrases found in the input with the plays where they were found



List<List <SearchResults>>
Stream<List <SearchResults>>
Stream <SearchResults>
Map<String, List <SearchResults>>
stream()
flatMap(List::stream)
collect(groupingBy(...))

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to display phrases associated with each play

```
void printPhrases(List<List<SearchResults>> listOfListOfResults){
  Map<String, List<SearchResults>> map = listOfListOfResults
    .stream()

    .flatMap(List::stream)

    .collect(groupingBy(SearchResults::getTitle));

  map.forEach((key, value) -> {
                System.out.println("Title \""
                                   + key
                                   + "\" contained");
                value.forEach(SearchResults::print);});
}
```

See SearchStreamGang/src/main/java/livelessons/streamgangs/SearchStreamGang.java

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to display phrases associated with each play

```
void printPhrases(List<List<SearchResults>> listOfListOfResults){
  Map<String, List<SearchResults>> map = listOfListOfResults
    .stream()

    .flatMap(List::stream)

    .collect(groupingBy(SearchResults::getTitle));

  map.forEach((key, value) -> {
               System.out.println("Title \""
                                  + key
                                  + "\" contained");
               value.forEach(SearchResults::print);});
}
```

Converts the list of lists of search results into a stream of lists of search results

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to display phrases associated with each play

```
void printPhrases(List<List<SearchResults>> listOfListOfResults){
  Map<String, List<SearchResults>> map = listOfListOfResults
    .stream()

    .flatMap(List::stream)

    .collect(groupingBy(SearchResults::getTitle));

  map.forEach((key, value) -> {
    System.out.println("Title \""
                       + key
                       + "\" contained");
    value.forEach(SearchResults::print);});
}
```

Return an output stream containing the results of flattening the stream of lists into a stream of search results

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

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to display phrases associated with each play

```
void printPhrases(List<List<SearchResults>> listOfListOfResults){
  Map<String, List<SearchResults>> map = listOfListOfResults
    .stream()

    .flatMap(List::stream)

    .collect(groupingBy(SearchResults::getTitle));

  map.forEach((key, value) -> {
    System.out.println("Title \""
                       + key
                       + "\" contained");
    value.forEach(SearchResults::print);});
}
```

# of output stream elements may differ from the # of input stream elements

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to display phrases associated with each play

```
void printPhrases(List<List<SearchResults>> listOfListOfResults){
  Map<String, List<SearchResults>> map = listOfListOfResults
    .stream()

    .flatMap(List::stream)

    .collect(groupingBy(SearchResults::getTitle));

  map.forEach((key, value) -> {
              System.out.println("Title \""
                                 + key
                                 + "\" contained");
              value.forEach(SearchResults::print);});
}
```

Groups elements via to a classification function & return results in a Map

See docs.oracle.com/javase/8/docs/api/java/util/stream/Collectors.html#groupingBy

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to display phrases associated with each play

```
void printPhrases(List<List<SearchResults>> listOfListOfResults){
  Map<String, List<SearchResults>> map = listOfListOfResults
    .stream()

    .flatMap(List::stream)

    .collect(groupingBy(SearchResults::getTitle));

  map.forEach((key, value) -> {
                System.out.println("Title \""
                                   + key
                                   + "\" contained");
                value.forEach(SearchResults::print);});
}
```

Associates phrases found in input with titles where they were found

See docs.oracle.com/javase/8/docs/api/java/util/Map.html

Implementing printPhrases() as a Sequential Stream

- printPhrases() uses a stream to display phrases associated with each play

```
void printPhrases(List<List<SearchResults>> listOfListOfResults){
  Map<String, List<SearchResults>> map = listOfListOfResults
    .stream()

    .flatMap(List::stream)

    .collect(groupingBy(SearchResults::getTitle));

  map.forEach((key, value) -> {
              System.out.println("Title \""
                                 + key
                                 + "\" contained");
              value.forEach(SearchResults::print);});
}
```

Displays titles (keys) & phrases (values) in map

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

End of Java 8 Sequential SearchStreamGang Example (Part 2)