Pros & Cons of Java 8 Parallel Streams

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

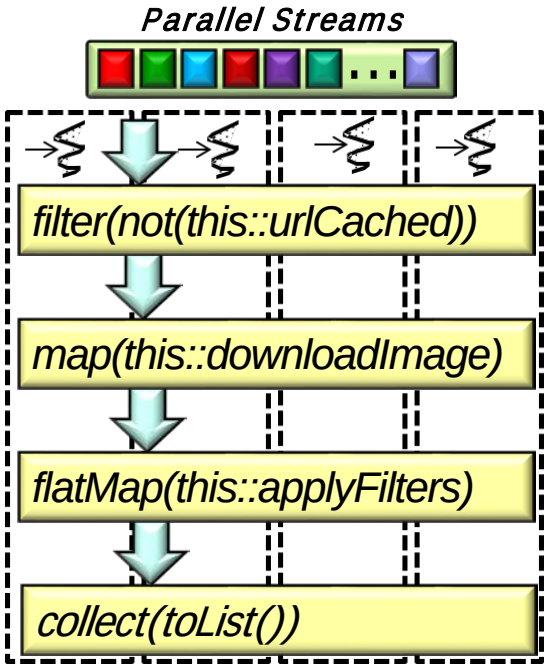
Professor of Computer Science

Institute for Software
Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Lesson

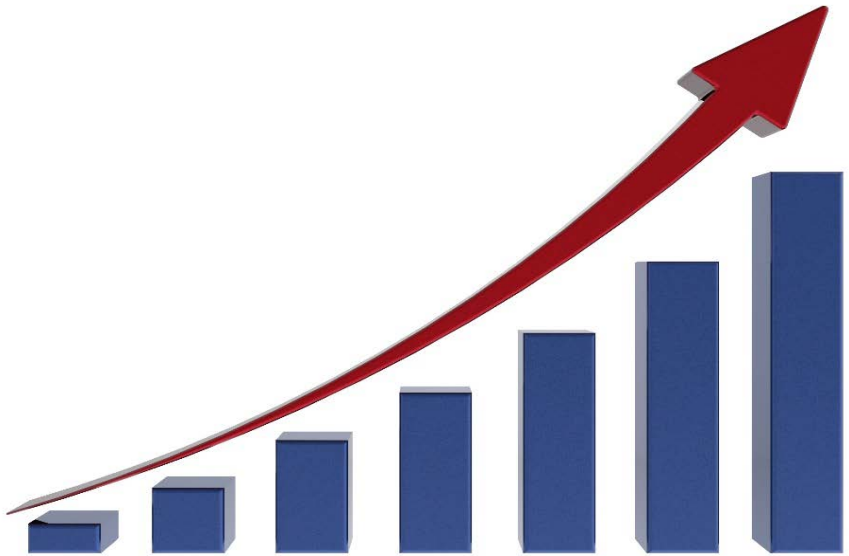
- Evaluate the pros & cons of Java 8 parallel streams

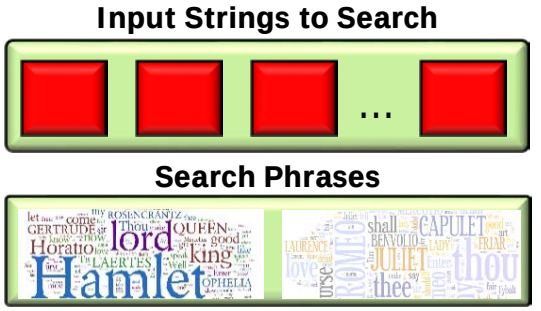


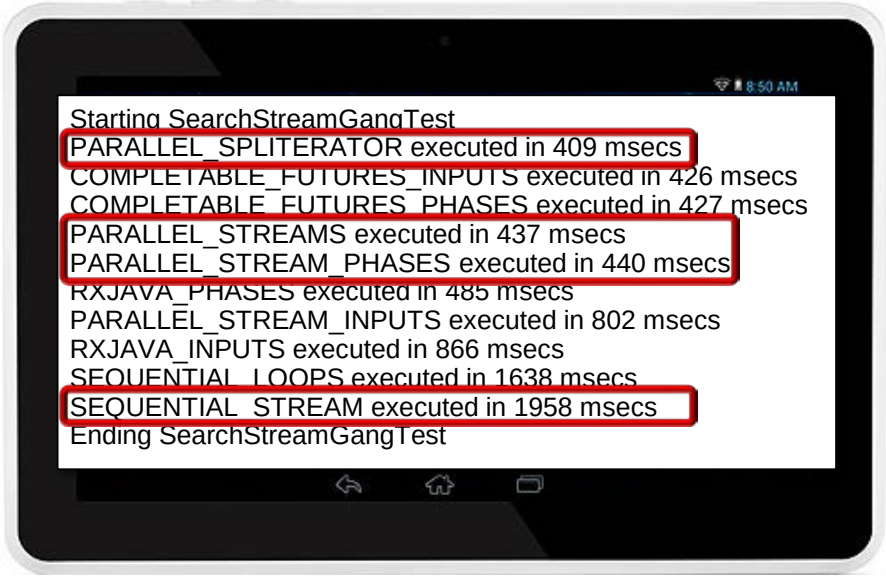
Pros of Java 8 Parallel Streams

Pros of Java 8 Parallel Streams

- The parallel stream implementations we analyzed are faster than the sequential stream implementations

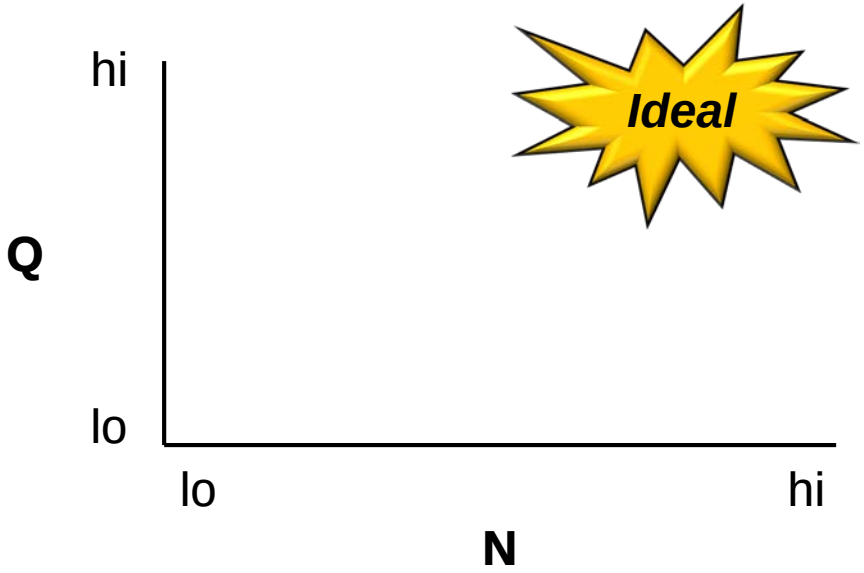






Pros of Java 8 Parallel Streams

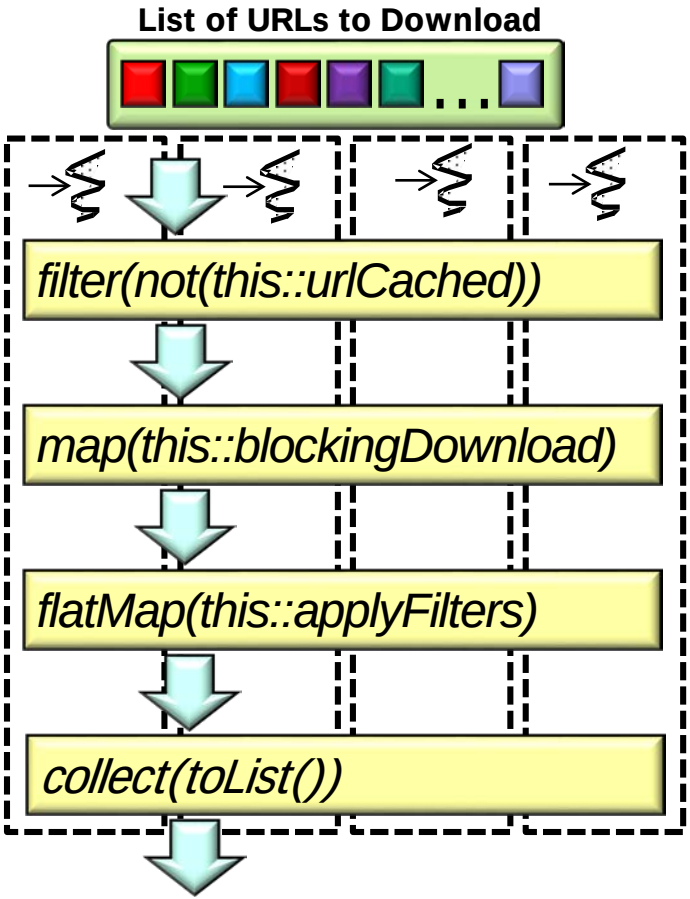
- The performance speedup is a largely a function of the partitioning strategy for the input (N), the amount of work performed (Q), & the # of cores



- *N is the # of data items to process per thread*
- *Q quantifies how CPU-intensive the processing is*

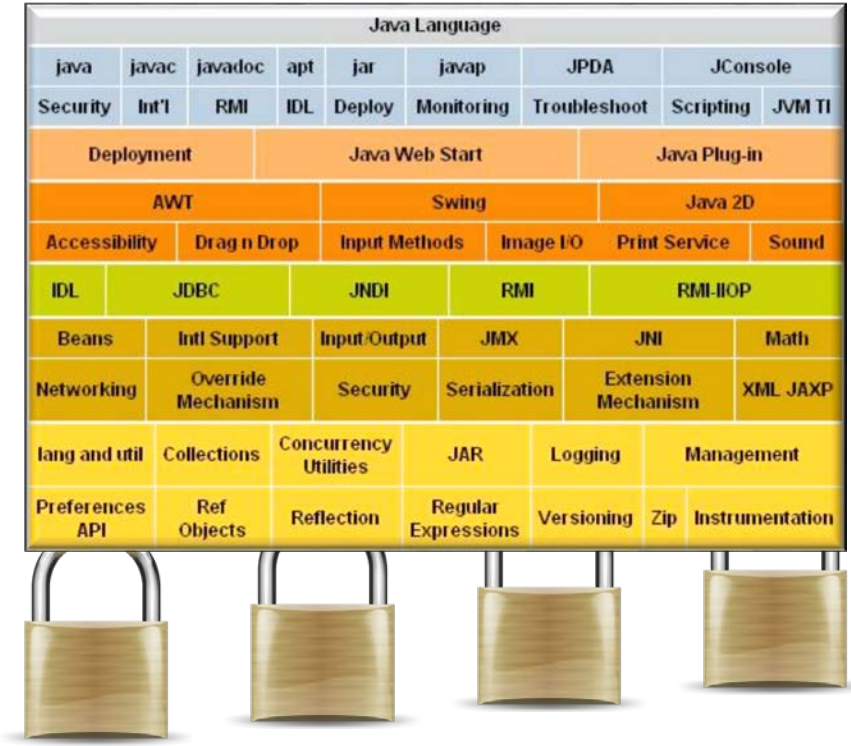
Pros of Java 8 Parallel Streams

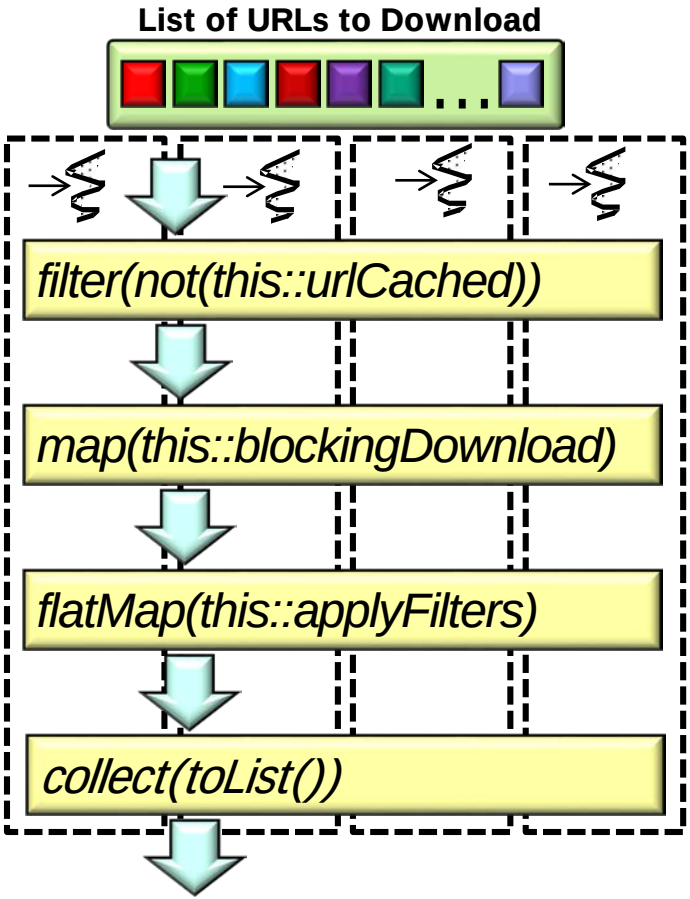
- No explicit synchronization was required in these implementations



Pros of Java 8 Parallel Streams

- No explicit synchronization was required in these implementations





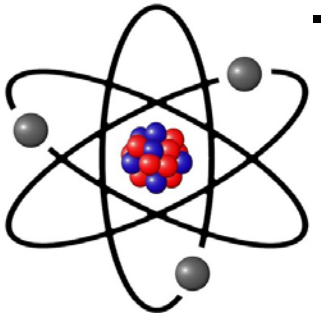
Java libraries handle any locking needed to read/write to files & connections

Pros of Java 8 Parallel Streams

- Converting from sequential to parallel streams required minuscule changes!

```
void processStream() {
  List<Image> filteredImages =
    getInput()
    .stream()
    .filter(not(this::urlCached))
    .map(this::downloadImage)
    .flatMap(this::applyFilters)
    .collect(toList());
    ...
```

```
void processStream() {
  List<Image> filteredImages =
    getInput()
    .parallelStream()
    .filter(not(this::urlCached))
    .map(this::downloadImage)
    .flatMap(this::applyFilters)
    .collect(toList());
    ...
```

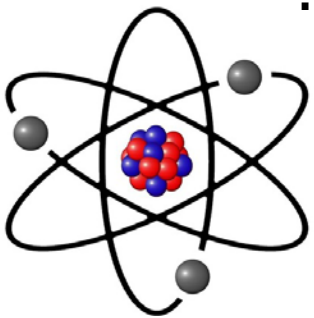


Pros of Java 8 Parallel Streams

- Converting from sequential to parallel streams required minuscule changes!

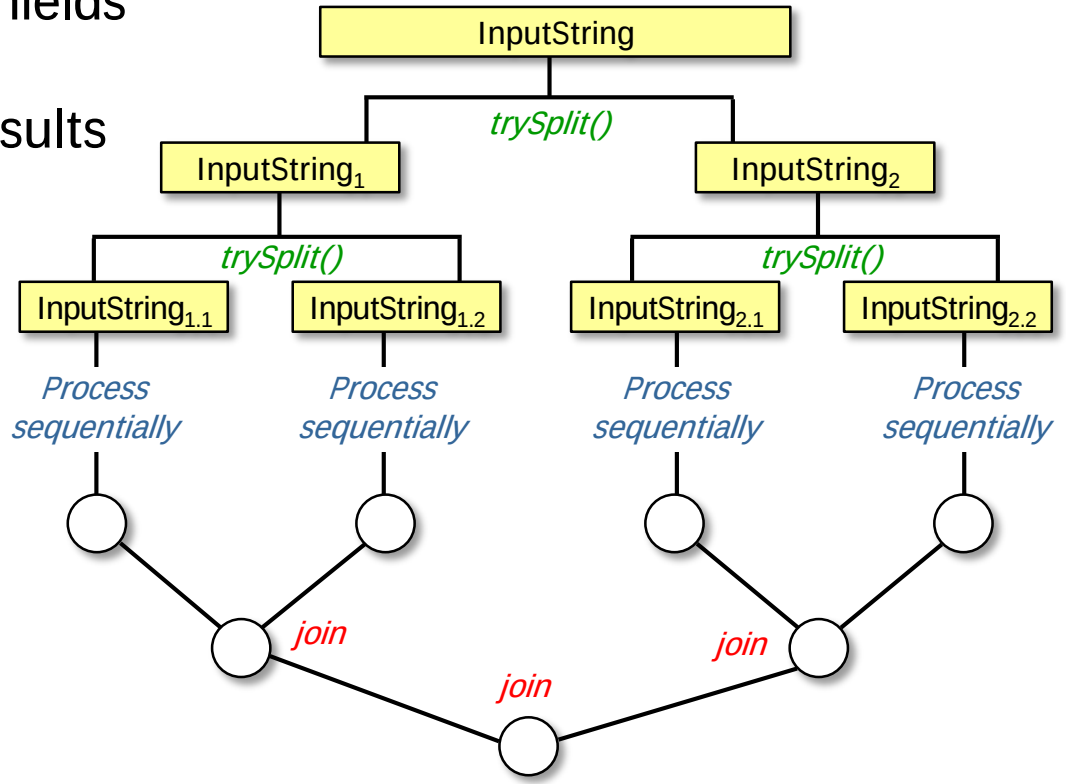
```
List<SearchResults> results =
  mPhrasesToFind
    .parallelStream()
    .map(phase ->
         searchForPhrase(...,
                         false))
    .filter(not(SearchResults
                ::isEmpty))
    .collect(toList());
```

```
List<SearchResults> results =
  mPhrasesToFind
    .parallelStream()
    .map(phase ->
         searchForPhrase(...,
                         true))
    .filter(not(SearchResults
                ::isEmpty))
    .collect(toList());
```



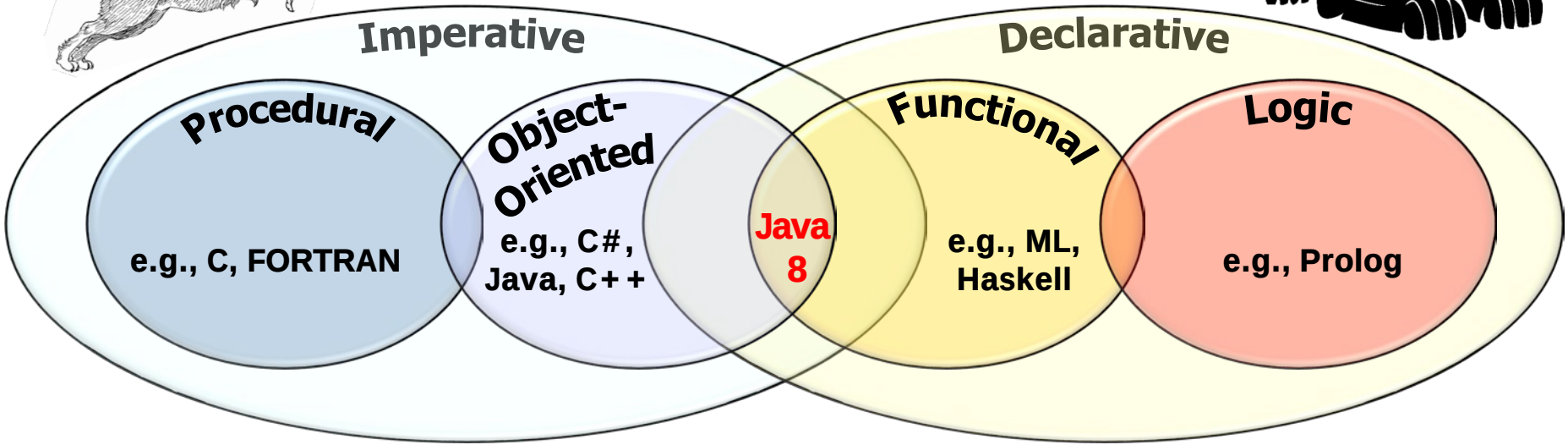
Pros of Java 8 Parallel Streams

- The Java 8 streams framework shields programmers from the details of splitting, processing, & joining results



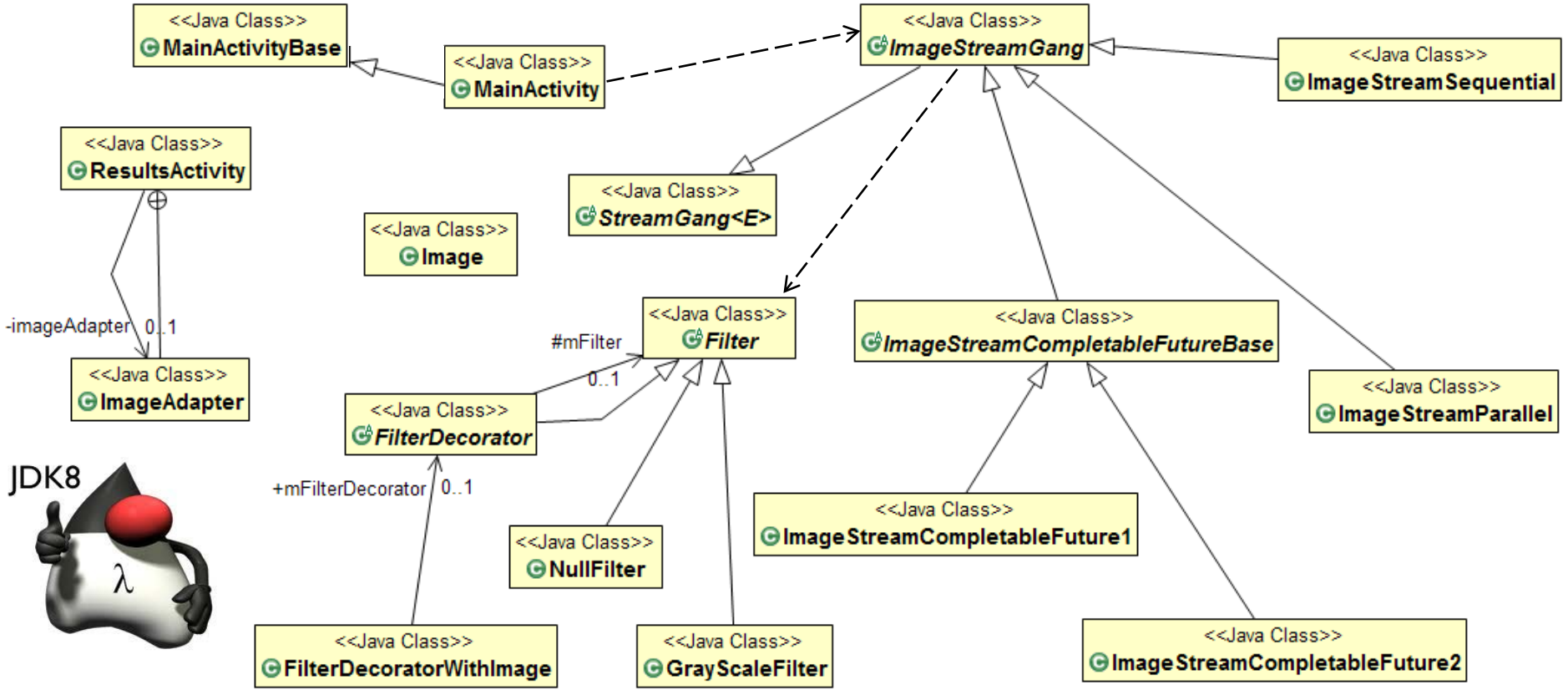
Pros of Java 8 Parallel Streams

- Examples show synergies between functional & object-oriented programming



Pros of Java 8 Parallel Streams

- The overall object-oriented design is easy to understand, reuse, & extend



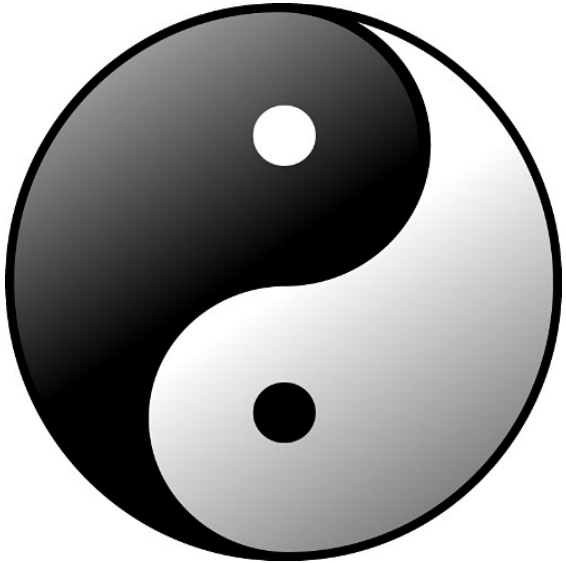
Pros of Java 8 Parallel Streams

- Concrete hook methods helped close the gap between domain intent & computations performed


```
void processStream() {
  List<Image> filteredImages =
    getInput()
      .stream()
      .filter(not(this::urlCached))
      .map(this::downloadImage)
      .flatMap(this::applyFilters)
      .collect(toList());
...
```

Pros of Java 8 Parallel Streams

- Concrete hook methods helped close the gap between domain intent & computations performed



```
void processStream() {
  List<Image> filteredImages =
    getInput()
      .stream()
      .filter(not(this::urlCached))
      .map(this::downloadImage)
      .flatMap(this::applyFilters)
      .collect(toList());
...
```

Cons of Java 8 Parallel Streams

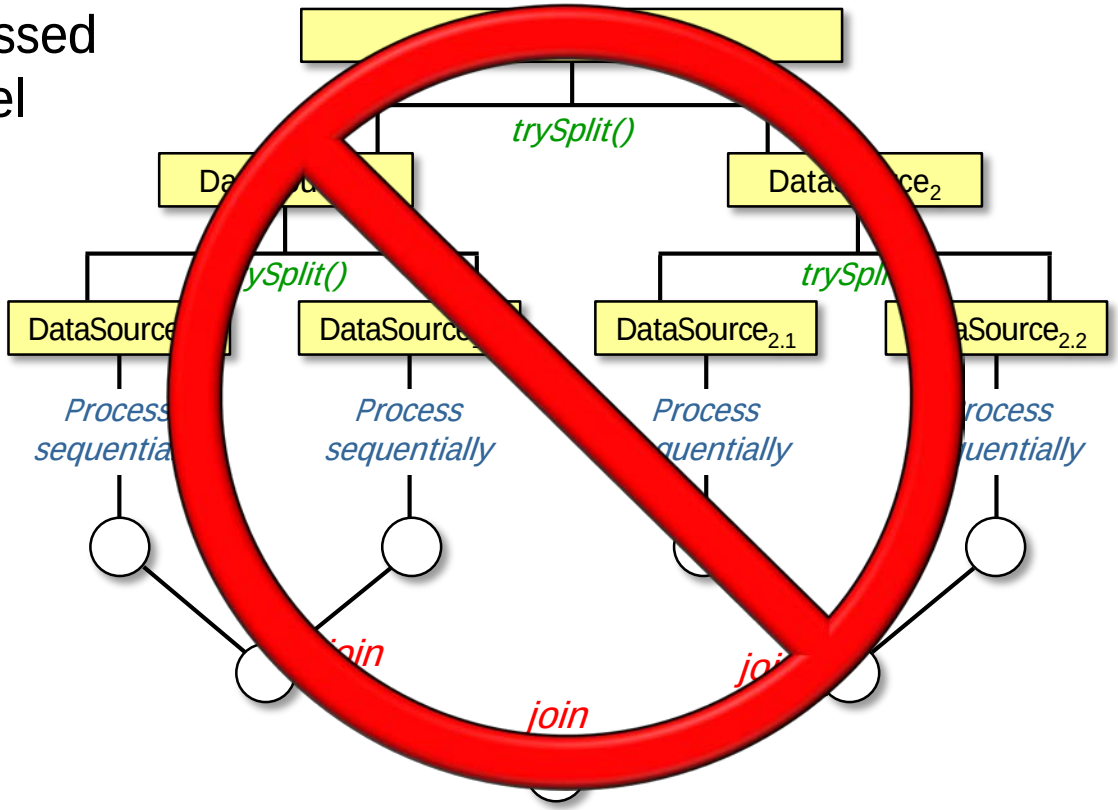
Cons of Java 8 Parallel Streams

- There are some limitations with Java 8 parallel streams

The Java 8 parallel streams framework is not all unicorns & rainbows!!

Cons of Java 8 Parallel Streams

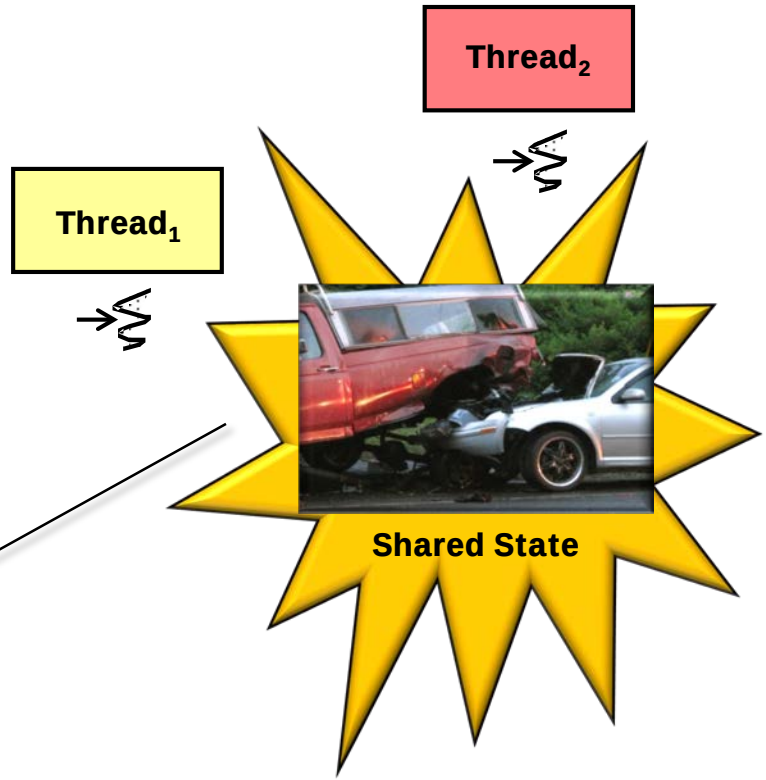
- There are some limitations with Java 8 parallel streams, e.g.
  - Some problems can't expressed using the map/reduce model



See dzone.com/articles/whats-wrong-java-8-part-iii

Cons of Java 8 Parallel Streams

- There are some limitations with Java 8 parallel streams, e.g.
  - Some problems can't expressed using the map/reduce model
  - Race conditions may occur if behaviors called by aggregate operations aren't thread-safe



Race conditions occur when a program depends on the sequence or timing of threads for it to operate properly

See en.wikipedia.org/wiki/Race_condition#Software

Cons of Java 8 Parallel Streams

- There are some limitations with Java 8 parallel streams, e.g.
  - Some problems can't expressed using the map/reduce model
  - Race conditions may occur if behaviors called by aggregate operations aren't thread-safe
  - All parallel streams share a common fork-join pool

See dzone.com/articles/think-twice-using-java-8

Cons of Java 8 Parallel Streams

- There are some limitations with Java 8 parallel streams, e.g.
  - Some problems can't expressed using the map/reduce model
  - Race conditions may occur if behaviors called by aggregate operations aren't thread-safe
  - All parallel streams share a common fork-join pool
    - Java 8 completable futures don't have this limitation

See dzone.com/articles/think-twice-using-java-8

Cons of Java 8 Parallel Streams

- There are some limitations with Java 8 parallel streams, e.g.
  - Some problems can't expressed using the map/reduce model
  - Race conditions may occur if behaviors called by aggregate operations aren't thread-safe
  - All parallel streams share a common fork-join pool
  - The parallel streams framework incurs some overhead due to its use of the fork-join pool framework

See coopsoft.com/dl/Blunder.pdf

Cons of Java 8 Parallel Streams

- There are some limitations with Java 8 parallel streams, e.g.
  - Some problems can't expressed using the map/reduce model
  - Race conditions may occur if behaviors called by aggregate operations aren't thread-safe
  - All parallel streams share a common fork-join pool
  - The parallel streams framework incurs some overhead due to its use of the fork-join pool framework
  - Writing parallel spliterators can be tricky...

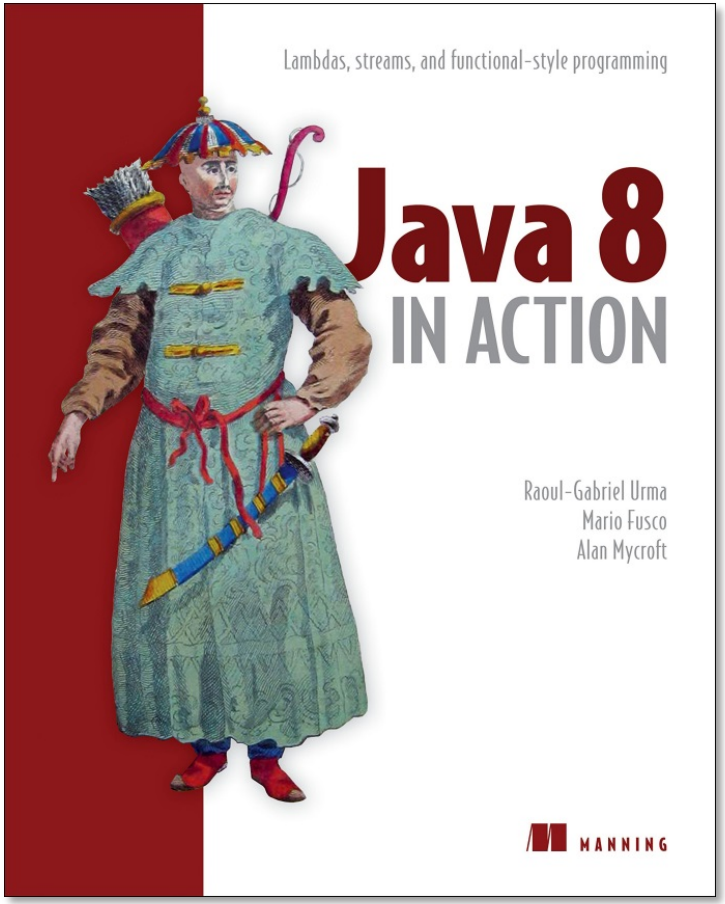
Cons of Java 8 Parallel Streams

- In general, however, the pros of Java 8 parallel streams outweigh the cons in many common use cases!!

See www.ibm.com/developerworks/library/j-jvmc2

Cons of Java 8 Parallel Streams

- Additional material on Java 8 parallel streams appears in the book “Java 8 in Action”



See www.manning.com/books/java-8-in-action

End of Pros & Cons of Java 8 Parallel Streams