Recognize How Parallel Programs are Developed in Java

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

• Understand the meaning of key parallel programming concepts
• Know when to apply parallelism
• Recognize how Java supports parallel programming via object-oriented & functional frameworks

See www.dre.vanderbilt.edu/~schmidt/frameworks.html
Learning Objectives in this Part of the Lesson

- Recognize the parallelism frameworks supported by Java, e.g.
  - **Fork-join pools**
    - An object-oriented framework

See [docs.oracle.com/javase/tutorial/essential/concurrency/forkjoin.html](http://docs.oracle.com/javase/tutorial/essential/concurrency/forkjoin.html)
• Recognize the parallelism frameworks supported by Java, e.g.
  • Fork-join pools
  • **Parallel streams**
    • A synchronous functional framework

See [docs.oracle.com/javase/tutorial/collections/streams/parallelism.html](docs.oracle.com/javase/tutorial/collections/streams/parallelism.html)
Learning Objectives in this Part of the Lesson

- Recognize the parallelism frameworks supported by Java, e.g.
  - Fork-join pools
  - Parallel streams
  - **Completable futures**
    - A reactive/asynchronous functional framework

---

**Task 1**

```
Task 1
getPage() = Task 1
supplyAsync
  (getPage())
```

**Task 2**

```
Task 2
/imgNum\ = /page\ 8
  .thenApplyAsync
   (countImages(page))
  .thenApply(List::size)
```

**Task 3**

```
Task 3
/imgNum\ = /page\ 8
  .thenComposeAsync
   (crawlHyperLinks(page))
```

**Task 4**

```
Task 4
/imgNum1\ .thenCombine(/imgNum2\,
  (imgNum1, imgNum2) ->
  Integer::sum)
```

See [www.callicoder.com/java-8-completablefuture-tutorial](http://www.callicoder.com/java-8-completablefuture-tutorial)
• Recognize the parallelism frameworks supported by Java, e.g.
  • Fork-join pools
  • Parallel streams
  • Completable futures
• Reactive streams
  • An async programming paradigm concerned with processing data streams & propagation of changes

See en.wikipedia.org/wiki/Reactive_Streams
Overview of Java Parallelism Frameworks
Overview of Java Parallelism Frameworks

- Java 7 introduced the object-oriented fork-join framework

See [www.infoq.com/interviews/doug-lea-fork-join](http://www.infoq.com/interviews/doug-lea-fork-join)
Overview of Java Parallelism Frameworks

- Java 7 introduced the object-oriented fork-join framework
- Provides high performance, fine-grained task execution for data parallelism

See www.dre.vanderbilt.edu/~schmidt/PDF/DataParallelismInJava.pdf
Overview of Java Parallelism Frameworks

- Java 7 introduced the object-oriented fork-join framework
- Provides high performance, fine-grained task execution for data parallelism
- Supports parallel programming by solving problems via “divide & conquer”

```java
Result solve(Problem problem) {
    if (problem is small)
        directly solve problem
    else {
        a. split problem into independent parts
        b. fork new sub-tasks to solve each part
        c. join all sub-tasks
        d. compose result from sub-results
    }
}
```

See en.wikipedia.org/wiki/Divide_and_conquer_algorithm
Java 7 introduced the object-oriented fork-join framework

- Provides high performance, fine-grained task execution for data parallelism
- Supports parallel programming by solving problems via “divide & conquer”
- Employs *work-stealing* to optimize multi-core processor performance

See gee.cs.oswego.edu/dl/papers/fj.pdf
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks

Java 8 added two new parallel functional programming frameworks

1. Parallel streams

```java
filter(not(this::urlCached))
map(this::downloadImage)
flatMap(this::applyFilters)
collect(toList())
```

See docs.oracle.com/javase/tutorial/collections/streams/parallelism.html
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks

1. Parallel streams
   - Partitions a stream into multiple substreams that run independently & combine into a “reduced” result

```
filter(not(this::urlCached))
map(this::downloadImage)
flatMap(this::applyFilters)
collect(toList())
```
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks

1. Parallel streams
   - Partitions a stream into multiple substreams that run independently & combine into a “reduced” result
   - Chunks of data in the substreams can be mapped to multiple threads (& cores)
Overview of Java Parallelism Frameworks

Java 8 added two new parallel functional programming frameworks

1. Parallel streams
   - Partitions a stream into multiple substreams that run independently & combine into a “reduced” result
   - Chunks of data in the substreams can be mapped to multiple threads (& cores)
   - Leverages the common fork-join pool

See dzone.com/articles/common-fork-join-pool-and-streams
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks

1. Parallel streams
   - Partitions a stream into multiple substreams that run independently & combine into a “reduced” result
   - Chunks of data in the substreams can be mapped to multiple threads (& cores)
   - Leverages the common fork-join pool

Parallel streams provides fine-grained data parallelism functional programming
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks

1. Parallel streams

2. Completable futures

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletableFuture.html
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks
  1. Parallel streams
  2. Completable futures

  - Supports dependent actions that trigger upon completion of async operations

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletionStage.html
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks
  
  1. Parallel streams
  
  2. Completable futures
     
     - Supports dependent actions that trigger upon completion of async operations
     
     - Async operations are a model of concurrent programming where the caller doesn’t block waiting for callee to complete

See en.wikipedia.org/wiki/Asynchrony_(computer_programming)
Overview of Java Parallelism Frameworks

- Java 8 added two new parallel functional programming frameworks
  1. Parallel streams
  2. Completable futures

  - Supports dependent actions that trigger upon completion of async operations
  - Async operations can run in parallel in thread pools

See [www.nurkiewicz.com/2013/05/java-8-definitive-guide-to.html](www.nurkiewicz.com/2013/05/java-8-definitive-guide-to.html)
Overview of Java Parallelism Frameworks

• Java 8 added two new parallel functional programming frameworks

1. Parallel streams

2. Completable futures
   • Supports dependent actions that trigger upon completion of async operations
   • Async operations can run in parallel in thread pools
   • Many types of thread pools can be applied here!
Java 8 added two new parallel functional programming frameworks

1. Parallel streams

2. Completable futures
   - Supports dependent actions that trigger upon completion of async operations
   - Async operations can run in parallel in thread pools

Java completable futures & streams can be combined to good effects!!
Overview of Java Parallelism Frameworks

- Java 9 added support for reactive streams via the "Flow" API

See www.reactive-streams.org
Overview of Java Parallelism Frameworks

- Java 9 added support for reactive streams via the “Flow” API
- Implements a stream-oriented pub/sub framework via two patterns

See javasampleapproach.com/java/java-9/java-9-flow-api-example-publisher-and-subscriber
Overview of Java Parallelism Frameworks

- Java 9 added support for reactive streams via the “Flow” API
- Implements a stream-oriented pub/sub framework via two patterns

- **Iterator**
  - Applies a “pull model” where apps pull items from a publisher source

See [en.wikipedia.org/wiki/Iterator_pattern](en.wikipedia.org/wiki/Iterator_pattern)
Overview of Java Parallelism Frameworks

- Java 9 added support for reactive streams via the “Flow” API
- Implements a stream-oriented pub/sub framework via two patterns
  - **Iterator**
  - **Observer**
    - Applies a “push model” that reacts when a publisher source pushes an item to a subscriber sink

See [en.wikipedia.org/wiki/Observer_pattern](en.wikipedia.org/wiki/Observer_pattern)
Overview of Java Parallelism Frameworks

- Popular reactive streams implementations include RxJava & Project Reactor

See [www.baeldung.com/rx-java](http://www.baeldung.com/rx-java) & [projectreactor.io](http://projectreactor.io)
Overview of Java Parallelism Frameworks

• All these Java frameworks can often eliminate the use of synchronization or explicit threading when developing parallel apps!

Alleviates many accidental & inherent complexities of parallel programming
Overview of Java Parallelism Frameworks

- Java parallel streams & completable future functional frameworks use the object-oriented fork-join framework by default

See [www.oracle.com/technetwork/articles/java/fork-join-422606.html](http://www.oracle.com/technetwork/articles/java/fork-join-422606.html)
End of Recognize How Parallel Programs Are Developed in Java