Key Concurrency & Scheduler Operators
Associated with the Mono Class (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

• Recognize key Mono operators
• Concurrency & scheduler operators
  • These operators arrange to run other operators in designated threads & thread pools
  • e.g., Schedulers.parallel()
Key Scheduler Operators Associated with the Mono Class
Key Scheduler Operators Associated with the Mono Class

- The Schedulers.parallel() operator
- Returns a Scheduler that hosts a fixed pool of Executor Service-based workers suitable for parallel work

See projectreactor.io/docs/core/release/api/reactor/core/scheduler/Schedulers.html#parallel
Key Scheduler Operators Associated with the Mono Class

- The Schedulers.parallel() operator
- Returns a Scheduler that hosts a fixed pool of Executor Service-based workers suitable for parallel work

```java
static Scheduler parallel()

Mono<BigFraction> multiplyAsync(BigFraction bf1, BigFraction bf2) {
    return Mono
        .fromCallable(() -> bf1.multiply(bf2))
        .subscribeOn(Schedulers.parallel());
}
```

Create a Mono that emits the results of multiplying `bf1` & `bf2` in a thread from the parallel thread pool

See Reactive/mono/ex3/src/main/java/MonoEx.java
Key Scheduler Operators Associated with the Mono Class

- The Schedulers.parallel() operator
  - Returns a Scheduler that hosts a fixed pool of Executor Service-based workers suitable for parallel work
  - Optimized for fast running non-blocking operations
  - i.e., computation-intensive, not I/O-intensive!

See projectreactor.io/docs/core/release/api/reactor/core/scheduler/Schedulers.html
Key Scheduler Operators Associated with the Mono Class

• The Schedulers.parallel() operator
  • Returns a Scheduler that hosts a fixed pool of Executor Service-based workers suitable for parallel work
    • Optimized for fast running non-blocking operations
  • Implemented via “daemon threads”
    • i.e., won’t prevent the app from exiting even if its work isn’t done

See www.baeldung.com/java-daemon-thread
Key Scheduler Operators Associated with the Mono Class

- The Schedulers.parallel() operator
  - Returns a Scheduler that hosts a fixed pool of Executor Service-based workers suitable for parallel work
  - This operator is often used with the flatMap() concurrency idiom

```
return Flux
  .fromIterable(bigFractions)
    .flatMap(bf -> Mono
      .fromCallable(() -> bf)
        .subscribeOn
          (Schedulers
            .parallel())
        .map
          (multiplyFracs)
    )
  .reduce(BigFraction::add)
  ...
```

Multiply many BigFraction objects concurrently

See lesson on “Key Transforming Operators in the Flux Class (Part 2)”
The Schedulers.parallel() operator
- Returns a Scheduler that hosts a fixed pool of Executor Service-based workers suitable for parallel work
- This operator is often used with the flatMap() concurrency idiom
- RxJava’s Schedulers.computation() is similar

See [reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/schedulers/Schedulers.html#computation](reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/schedulers/Schedulers.html#computation)
End of Key Concurrency & Scheduler Operators Associated with the Mono Class (Part 2)