Applying Key Operators in Project Reactor: Case Study ex4 (Part 3)

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



Institute for Software Integrated Systems

Vanderbilt University Nashville, Tennessee, USA





 Part 3 of case study ex4 applies Flux operators flatMap() & subscribe() to create, multiply, & display BigFraction objects asynchronously

```
Mono
.fromSupplier(() ->
    makeBigFraction
    (sRANDOM, true))
.repeat(sMAX_FRACTIONS - 1)
```

(sRANDOM, true))
.repeat(sMAX_FRACTIONS - 1)
.flatMap(bf1 ->
 multiplyFraction(bf1,
 sBigReducedFraction,
 Schedulers.parallel(),

```
(backpressureSubscriber);
```

sb))

subscribe

 Part 3 of case study ex4 applies Flux Mono .fromSupplier(() -> operators flatMap() & subscribe() to

```
create, multiply, & display BigFraction
                                              (sRANDOM, true))
objects asynchronously
                                        .repeat(sMAX FRACTIONS - 1)
                                        .flatMap(bf1 ->
                                           multiplyFraction(bf1,
                                              sBigReducedFraction,
                                              sb))
```

Schedulers.parallel(),

This example *does* apply backpressure via the registered subscriber

(backpressureSubscriber);

makeBigFraction

.subscribe

Mono

- Part 3 of case study ex4 applies Flux operators flatMap() & subscribe() to create, multiply, & display BigFraction objects asynchronously
 - It also shows how to use Mono operators fromSupplier(), repeat(), & subscribeOn()

```
.fromSupplier(() ->
  makeBigFraction
     (sRANDOM, true))
.repeat(sMAX FRACTIONS - 1)
.flatMap(bf1 ->
  multiplyFraction(bf1,
      sBigReducedFraction,
      Schedulers.parallel(),
      sb))
.subscribe
   (backpressureSubscriber);
```

- Part 3 of case study ex4 applies Flux operators flatMap() & subscribe() to create, multiply, & display BigFraction objects asynchronously
 - It also shows how to use Mono operators fromSupplier(), repeat(), & subscribeOn()
 - In addition, it shows how to use the generic blocking Subscriber
 - This subscriber is "backpressure aware"

```
class BackpressureSubscriber<T>
  implements CoreSubscriber<T> {
  @Override
 public void onSubscribe
  (Subscription subscription) {
    mSubscription =
      subscription;
    subscription
      .request(mRequestSize);
```

Applying Key Operators in Project Reactor to ex4

Applying Key Operators in Project Reactor to ex4

```
L → A S E S C → S G S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C S C 
                  FluxEx ) m testFractionMultiplicationsBlockingSubscriber1
Pr... 🔂 😇 🛨 🗯 — 🦸 FluxEx.java
 ex4 ~/Dropbox/Documents
                                                       public static Mono<Void> testFractionMultiplicationsBlockingSubscriber2() {
                                 94
> aradle
 > idea
                                                                 StringBuffer sb =
> ____run
> build
                                                                           new StringBuffer(">> Calling testFractionMultiplicationsBlockingSubscriber2
 > aradle

∨ Imain

      iava
         > utils
                                 98
                                                                 // Create a blocking subscriber that processes various
            ex4.iava
            FluxEx.iava
                                                                 // types of signals and is "backpressure aware."
    agitignore.
    build.gradle
                                                                 BackpressureSubscriber<BigFraction> backpressureSubscriber =
                              100
    gradlew
    gradlew.bat
                                                                           makeBlockingSubscriber(sb,
                              101
    e settings.gradle
III External Libraries
                              102
Scratches and Consoles
                                                                                                                                     true):
                              103
                              104
                                                                 Mono
                              105
                              106
                                                                            // Generate a random large BigFraction.
                                                                            .fromSupplier(() -> BigFractionUtils
                              107
                              108
                                                                                                               .makeBigFraction(sRANDOM, true))
                              109
                                                                            // Generate a total of sMAX_FRACTIONS BigFraction objects.
                              110
                                                                            .repeat(sMAX_FRACTIONS - 1)
                              111
                               Terminal
                                            Services Profiler
Shared indexes for JDK "corretto-11" are downloaded (38.15 MB in 4 min, 52 sec) (today 3:28 PM)
                                                                                                                                                                                                                         66:30 Ø CRLF UTF-8 4 spaces P master a
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/flux/ex4

End of Applying Key Methods in Project Reactor: Case Study ex4 (Part 3)