Applying Key Operators in the Flux Class: Case Study ex1 (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

- Part 2 of case study ex1 shows how to use Flux operators fromIterable(), just(), doOnNext(), doOnError(), map(), mapNotNull(), & subscribe() to create, divide, & display Big Fraction objects synchronously.

```java
Flux
  .fromIterable(BigFractionList)
    .map(fraction -> fraction
           .divide(BigFraction.ZERO))
    .doOnError(ex -> logError(ex))
  .subscribe(fraction -> sb.append(" = 
                      + fraction.toMixedString() 
                      + "\n"),
            error -> sb.append("error"),
            () -> BigFractionUtils
                .display(sb.toString()));
```
Applying Key Operators in the Flux Class to ex1
Applying Key Operators in the Flux Class to ex1

See github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/flux/ex1

```java
public class FluxEx {

    public static Mono<Void> testFractionMultiplication() {
        StringBuilder sb =
            new StringBuilder("\n\n>>> Calling testFractionMultiplication()\n");

        Flux
            .just(BigFraction.valueOf(numerator: 100, denominator: 3),
                 BigFraction.valueOf(numerator: 100, denominator: 4),
                 BigFraction.valueOf(numerator: 100, denominator: 2),
                 BigFraction.valueOf(numerator: 100, denominator: 1))
            .map(BigFraction::multiply)
            .subscribe(sb::append);

        return sb.toString()
            .toMono()
            .thenReturn(Void::valueOf);
    }
```

End of Applying Key Operators in the Flux Class: Case Study ex1 (Part 2)