

Applying Key Operators in the Flux Class: Case Study ex3 (Part 1)

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 1 of case study ex3 shows how to use Flux operators fromIterable(), flatMap(), map(), onErrorResume(), onErrorStop(), collectList(), filter(), onErrorContinue(), & the parallel thread pool to create, reduce, multiply, & display BigFraction objects (a)synchronously

```
return Flux
    .fromIterable(denominators)
    .map(denominator -> BigFraction
        .valueOf(Math.abs
            (sRAND.nextInt()),
            denominator)
    .onErrorResume(errorHandler)
    .onErrorStop()
    .collectList()
    .flatMap(list -> BigFractionUtils
        .sortAndPrintList(list,
            sb));
```

See github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/flux/ex3

Learning Objectives in this Part of the Lesson

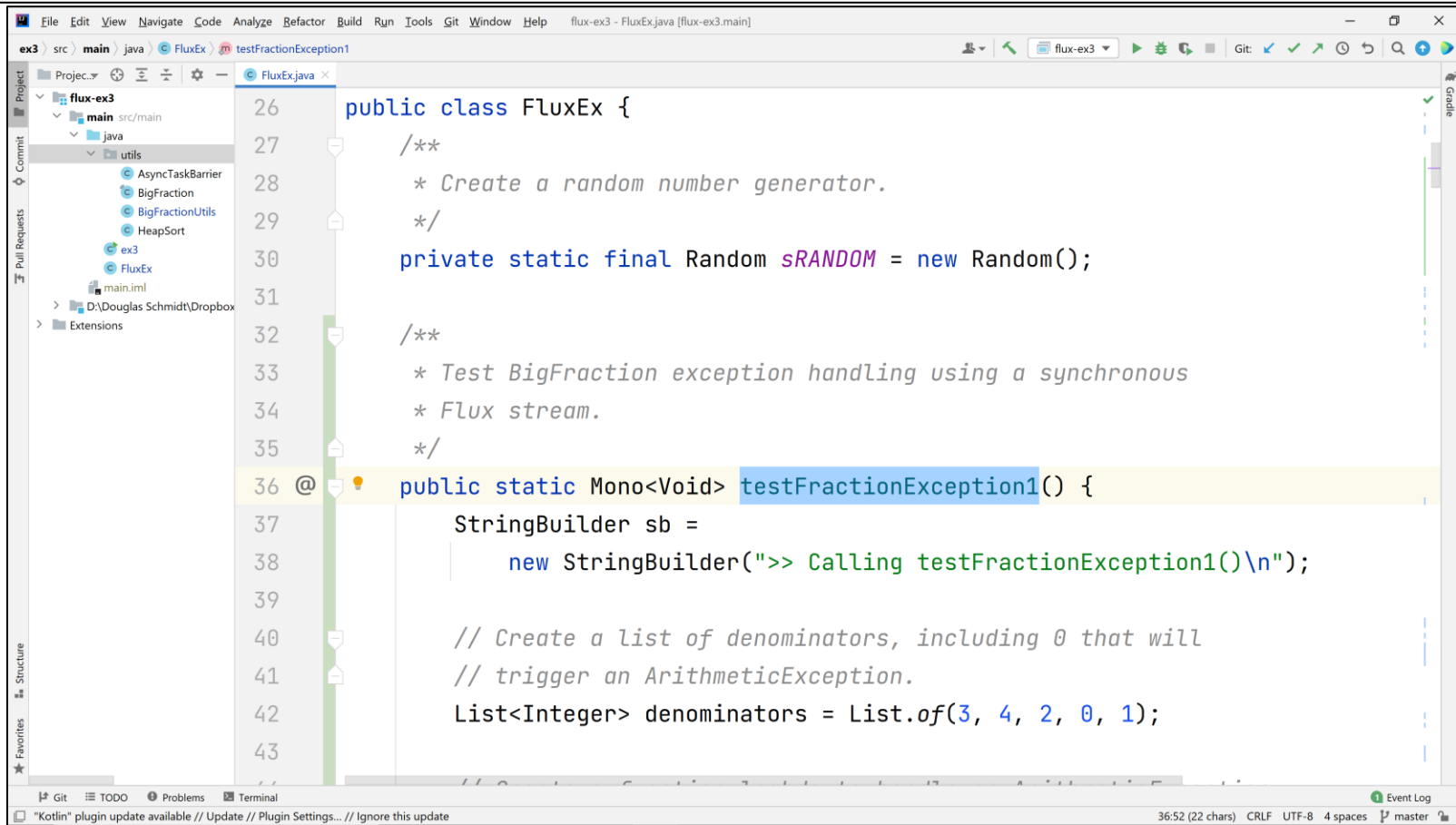
- Part 1 of case study ex3 shows how to use Flux operators fromIterable(), flatMap(), map(), onErrorResume(), onErrorStop(), collectList(), filter(), onErrorContinue(), & the parallel thread pool to create, reduce, multiply, & display BigFraction objects (a)synchronously
- It also shows the use of Mono operators like fromCallable(), subscribeOn(), firstWithSignal(), flatMap(), onErrorResume(), then(), & doOnSuccess()

```
Mono<List<BigFraction>> qSortM =  
    Mono.fromCallable(() ->  
        quickSort(list))  
        .subscribeOn  
          (Schedulers.parallel());  
Mono<List<BigFraction>> hSortM =  
    Mono.fromCallable(() ->  
        heapSort(list))  
        .subscribeOn  
          (Schedulers.parallel());  
  
return Mono.firstWithSignal  
        (qSortM, hSortM)  
        .doOnSuccess(displayList)  
        .then();
```

See github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/flux/ex3

Applying Key Operators in the Flux Class to ex3

Applying Key Operators in the Flux Class to ex3



```
26 public class FluxEx {
27     /**
28      * Create a random number generator.
29      */
30     private static final Random sRANDOM = new Random();
31
32     /**
33      * Test BigFraction exception handling using a synchronous
34      * Flux stream.
35      */
36     @Test public static Mono<Void> testFracException1() {
37         StringBuilder sb =
38             new StringBuilder(">> Calling testFracException1()\n");
39
40         // Create a list of denominators, including 0 that will
41         // trigger an ArithmeticException.
42         List<Integer> denominators = List.of(3, 4, 2, 0, 1);
43     }
```

See github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/flux/ex3

End of Applying Key Methods in the Flux Class: Case Study ex3 (Part 1)