Applying Key Operators in the Observable 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 RxJava Observable operators flatMap(), fromArray(), fromIterable(), onErrorReturn(), fromCallable(), collects(), onErrorResumeNext(), reduce(), filter(), subscribeOn(), map(), & Schedulers .computation() to asynchronously create, multiply, & display BigFraction objects, even in the presence of errors

```
Observable
   .fromCallable(() ->
     BigFraction
       .valueOf(Math.abs
                (sRAND.nextInt()),
                 denominator))
   .subscribeOn
     (Schedulers.computation())

   .onErrorReturn(errorHandler)

   .map(bf -> bf.multiply
        (sBigReducedFraction)))
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/Observable/ex3

Learning Objectives in this Part of the Lesson

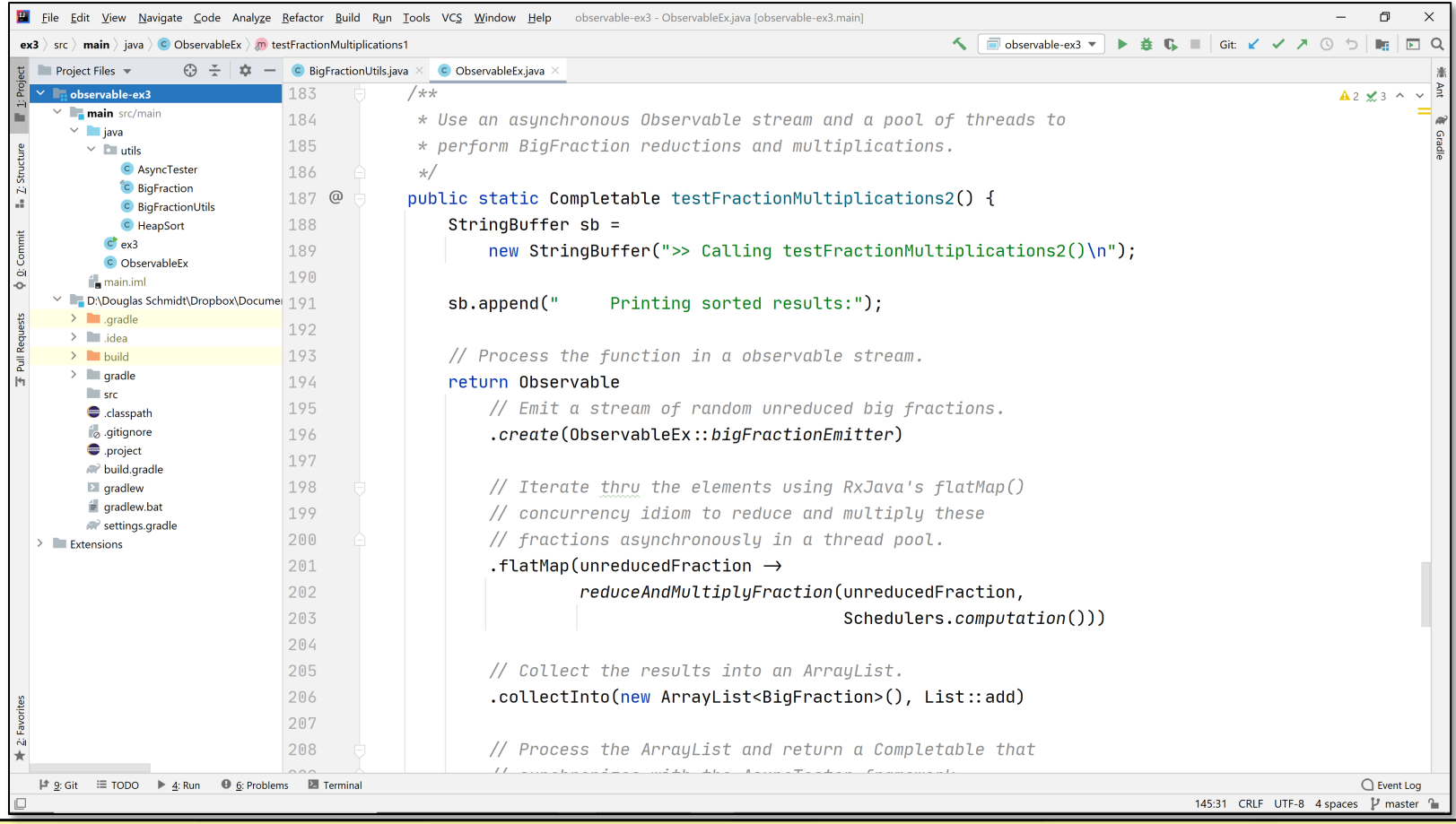
- Part 1 of case study ex3 shows how to use RxJava Observable operators flatMap(), fromArray(), fromIterable(), onErrorReturn(), fromCallable(), collects(), onErrorResumeNext(), reduce(), filter(), subscribeOn(), map(), & Schedulers .computation() to asynchronously create, multiply, & display BigFraction objects, even in the presence of errors
  - It also shows how Single operators doOnSuccess(), ignoreElement(), flatMapCompletable(), & just() can be used with Observable operators

```
return Single
    .just(list)

    .doOnSuccess(displayList)

    .ignoreElement();
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/Observable/ex3

Applying Key Operators in the Observable Class to ex3

Applying Key Operators in the Observable Class to ex3



```
/**
 * Use an asynchronous Observable stream and a pool of threads to
 * perform BigFraction reductions and multiplications.
 */
public static Completable testFractionMultiplications2() {
    StringBuffer sb =
        new StringBuffer(">> Calling testFractionMultiplications2()\n");

    sb.append("     Printing sorted results:");

    // Process the function in a observable stream.
    return Observable
        // Emit a stream of random unreduced big fractions.
        .create(ObservableEx::bigFractionEmitter)

        // Iterate thru the elements using RxJava's flatMap()
        // concurrency idiom to reduce and multiply these
        // fractions asynchronously in a thread pool.
        .flatMap(unreducedFraction ->
                reduceAndMultiplyFraction(unreducedFraction,
                                          Schedulers.computation()))

        // Collect the results into an ArrayList.
        .collectInto(new ArrayList<BigFraction>(), List::add)

        // Process the ArrayList and return a Completable that
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/Observable/ex3

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