

Applying Key Operators in the Observable

Class: Case Study ex2 (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

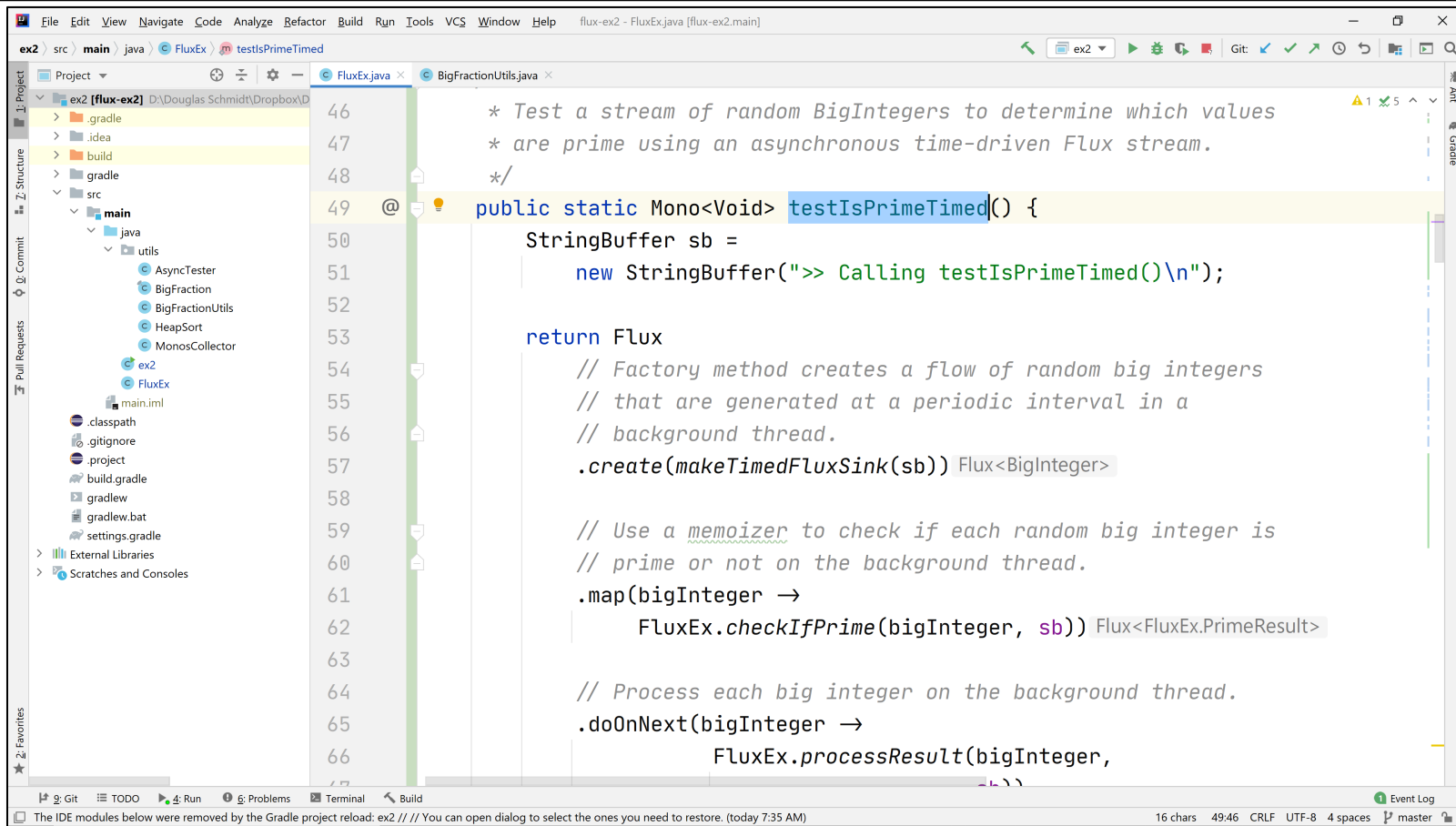
- Part 2 of case study ex2 explores how to use the RxJava Observable operators `create()`, `rangeLong()`, `map()`, `take()`, `filter()`, `subscribe()`, `doFinally()`, `subscribeOn()`, `ignoreElements()`, `observeOn()`, & `doOnNext()` to create large random `BigInteger` objects in one thread & asynchronously check if they are prime in another thread

Observable

```
.create
    (ObservableEx::emitAsync)
...
.observeOn (Schedulers
            .newThread())
.map (bigInteger ->
      ObservableEx.checkIfPrime
        (bigInteger, sb))
...
.doFinally (() ->
             BigFractionUtils.
               display (sb.toString()))
.ignoreElements ();
```

Applying Key Operators in the Observable Class to ex2

Applying Key Operators in the Flux Class to ex2



```
46      * Test a stream of random BigIntegers to determine which values
47      * are prime using an asynchronous time-driven Flux stream.
48      */
49      @Test
50      public static Mono<Void> testIsPrimeTimed() {
51          StringBuffer sb =
52              new StringBuffer(">> Calling testIsPrimeTimed()\n");
53
54          return Flux
55              // Factory method creates a flow of random big integers
56              // that are generated at a periodic interval in a
57              // background thread.
58              .create(makeTimedFluxSink(sb)) Flux<BigInteger>
59
60              // Use a memoizer to check if each random big integer is
61              // prime or not on the background thread.
62              .map(bigInteger →
63                  FluxEx.checkIfPrime(bigInteger, sb)) Flux<FluxEx.PrimeResult>
64
65              // Process each big integer on the background thread.
66              .doOnNext(bigInteger →
67                  FluxEx.processResult(bigInteger,
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/Observable/ex2

End of Applying Key Methods in the Observable Class: Case Study ex2 (Part 2)