Applying Key Methods in the Observable Class (Part 5)

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

- Recognize key methods in the Observable class & how they are applied in the case studies

Class Observable<T>

java.lang.Object
  io.reactivex.rxjava3.core.Observable<T>

Type Parameters:
  T - the type of the items emitted by the Observable

All Implemented Interfaces:
  ObservableSource<T>

Direct Known Subclasses:
  ConnectableObservable, GroupedObservable, Subject

public abstract class Observable<T>
extends Object
implements ObservableSource<T>

The Observable class is the non-backpressured, optionally multi-valued base reactive class that offers factory methods, intermediate operators and the ability to consume synchronous and/or asynchronous reactive dataflows.

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html
Learning Objectives in this Part of the Lesson

- Case study ex3 shows how to apply various RxJava operations \textit{asynchronously} to reduce & multiply BigFraction objects
  - e.g., fromIterable(), map(), create(), flatMap(), flatMapCompletable(), filter(), collectInto(), subscribeOn(), onErrorReturn(), & Schedulers.computation()

```java
return Observable
  .create(ObservableEx::bFEmmitter)
  .flatMap(unreducedFraction ->
            reduceAndMultiplyFraction
            (unreducedFraction,
            Schedulers.computation()))
  .collectInto(new ArrayList<BigFraction>(), List::add)
  .flatMapCompletable(list ->
            BigFractionUtils
            .sortAndPrintList(list,
            sb));
```

Applying Key Methods in the Observable Class to ex3
Applying Key Methods in the Observable Class to ex3

- testFractionMultiplications()
- Use an asynchronous Observable stream & a pool of threads to perform BigFraction object reductions & multiplications

```java
return Observable
    .create(ObservableEx::bFEmitter)
    .flatMap(unreducedFraction ->
        reduceAndMultiplyFraction
        (unreducedFraction,
         Schedulers.computation()))
    .collectInto(new ArrayList<BigFraction>(), List::add)
    .flatMapCompletable(list ->
        BigFractionUtils
        .sortAndPrintList(list, sb));
```

See Reactive/Observable/ex3/src/main/java/ObservableEx.java
Applying Key Methods in the Observable Class to ex2

- testFractionMultiplications()
- Use an asynchronous Observable stream & a pool of threads to perform BigFraction object reductions & multiplications
- Demonstrates fromCallable(), create(), flatMap(), collectInto(), filter(), flatMapCompletable(), subscribeOn(), & Schedulers.computation()

```java
return Observable.create(ObservableEx::bFEmitter)
    .flatMap(unreducedFraction ->
        reduceAndMultiplyFraction
            (unreducedFraction, Schedulers.computation()))
    .collectInto(new ArrayList<>(), List::add)
    .flatMapCompletable(list ->
        BigFractionUtils.sortAndPrintList(list, sb));
```
Applying Key Methods in the Observable Class to ex3

```java
public static Completable testFractionMultiplications()
{
    StringBuilder sb =
        new StringBuilder(">> Calling testFractionMultiplications1()\n");

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

    // Process the function in a observable stream.
    return Observable
        .create(ObservableEx::bigFractionEmitter)
        .flatMap(unreducedFraction ->
            reduceAndMultiplyFraction(unreducedFraction,
               Schedulers.computation()))
        .collect(Collectors.toCollection(ArrayList::new));
```

See github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/Observable/ex3
End of Applying Key Methods in the Observable Class (Part 5)