Key Combining Operators in the Observable Class (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

- Recognize key operators defined in—or used with—Observables
  - Factory method operators
  - Transforming operators
  - Action operators
  - Combining operators
    - These operators create an Observable from multiple iterations or sources
      - e.g., repeat() & mergeWith()

Key Combining Operators in the Observable Class

Key Combining Operators in the Observable Class

- The repeat() operator
  - Returns an Observable that repeats the sequence of items emitted by the given Observable at most `times' # of times

```
Observable<T> repeat
   (long times)
```

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#repeat

Key Combining Operators in the Observable Class

- The repeat() operator
  - Returns an Observable that repeats the sequence of items emitted by the given Observable at most `times' # of times
    - This param results in 'times' subscriptions to the original source

```
Observable<T> repeat
   (long times)
```

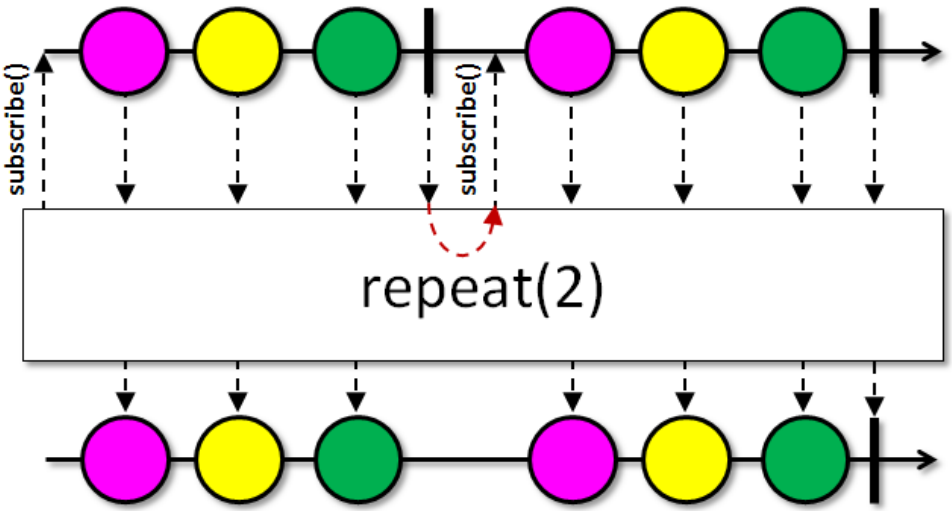
Key Combining Operators in the Observable Class

- The repeat() operator
  - Returns an Observable that repeats the sequence of items emitted by the given Observable at most `times' # of times
    - This param results in 'times' subscriptions to the original source
    - Returns a new Observable instance

```
Observable<T> repeat
   (long times)
```

Key Combining Operators in the Observable Class

- The repeat() operator
  - Returns an Observable that repeats the sequence of items emitted by the given Observable at most `times' # of times
  - This operator doesn't operate by default on a particular Scheduler



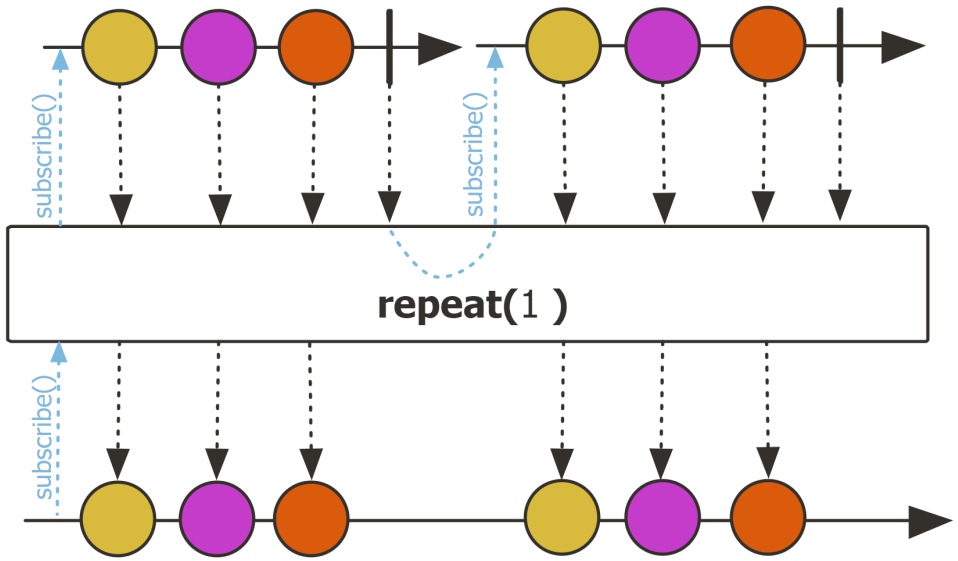

```
Observable
    .fromCallable(() ->
        BigFractionUtils.
            makeBigFraction
              (random, true))
    .repeat(4);
```

Generate 4 random, reduced big fractions

See Reactive/Observable/ex1/src/main/java/ObservableEx.java

Key Combining Operators in the Observable Class

- The repeat() operator
  - Returns an Observable that repeats the sequence of items emitted by the given Observable at most `times' # of times
  - This operator doesn't operate by default on a particular Scheduler
- Project Reactor's Flux.repeat() operator works the same



Generate 4 random, reduced big fractions

```
Flux.from
   (Mono.fromCallable(() ->
       BigFractionUtils.makeBigFraction
                          (random, true)))
   .repeat(3);
```

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Flux.html#repeat

Key Combining Operators in the Observable Class

- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other param

```
Observable<T> mergeWith
   (ObservableSource<? extends T>
      other)
```

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#mergeWith

Key Combining Operators in the Observable Class

- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other param
    - The param is the Observable Source to merge with

```
Observable<T> mergeWith
  (ObservableSource<? extends T>
    other)
```

```
@FunctionalInterface
public interface ObservableSource<T>
```

Represents a basic, non-backpressured `Observable` source base interface, consumable via an `Observer`.

Since:
2.0

Method Summary

All Methods	Instance Methods	Abstract Methods

Modifier and Type	Method and Description
void	`subscribe(@NonNull Observer<? super T> observer)` Subscribes the given `Observer` to this `ObservableSource` instance.

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/ObservableSource.html

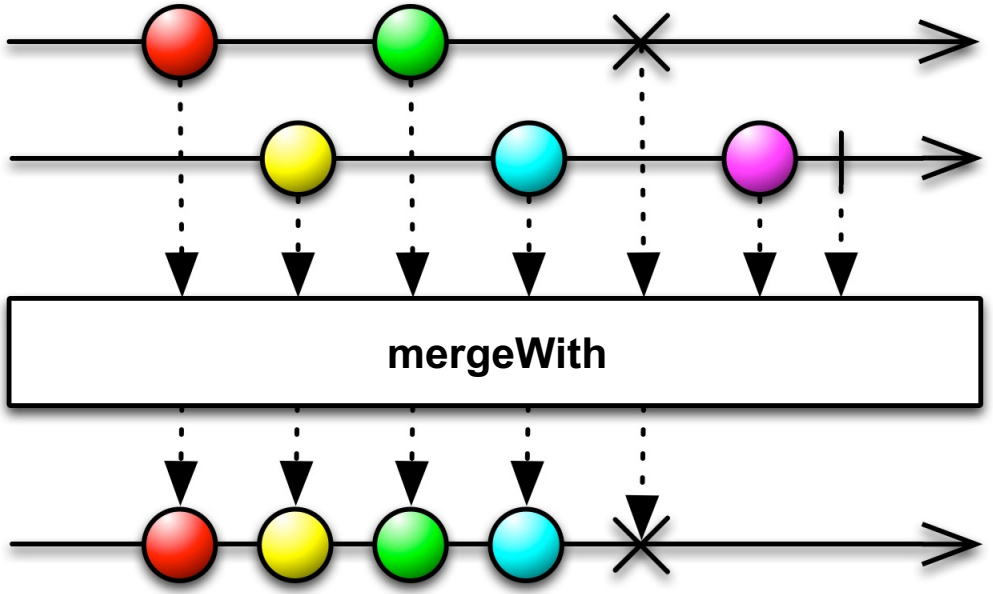
Key Combining Operators in the Observable Class

- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other param
    - The param is the Observable Source to merge with
    - Returns the new merged Observable instance

```
Observable<T> mergeWith
  (ObservableSource<? extends T>
    other)
```

Key Combining Operators in the Observable Class

- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other param
  - This operator combines items emitted by multiple Observable Sources so that they appear as a single ObservableSource

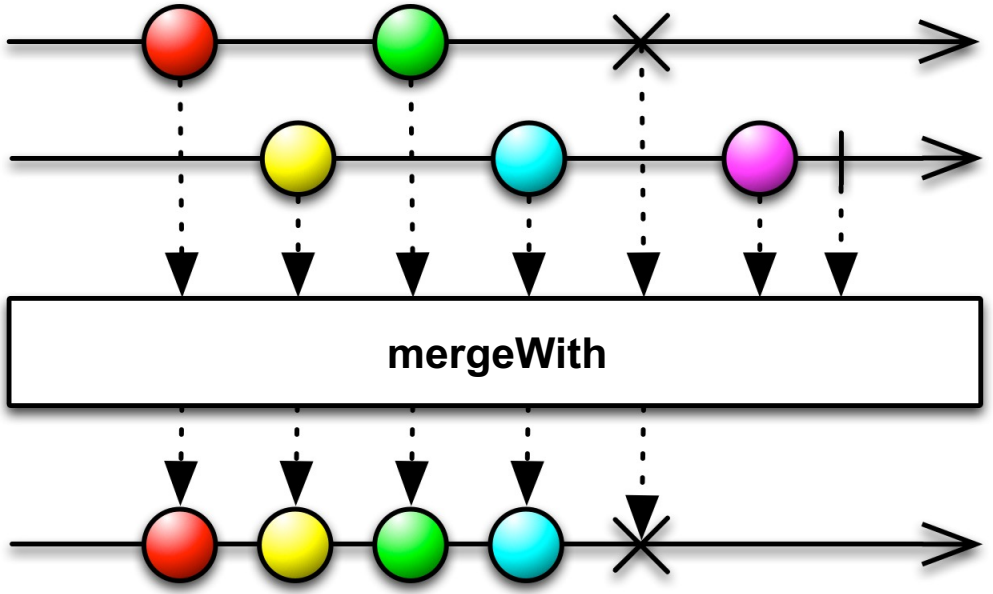



```
Observable<BigFraction> o1 ...
Observable<BigFraction> o2 ...
o1.mergeWith(o2)...
```

See Reactive/Observable/ex1/src/main/java/ObservableEx.java

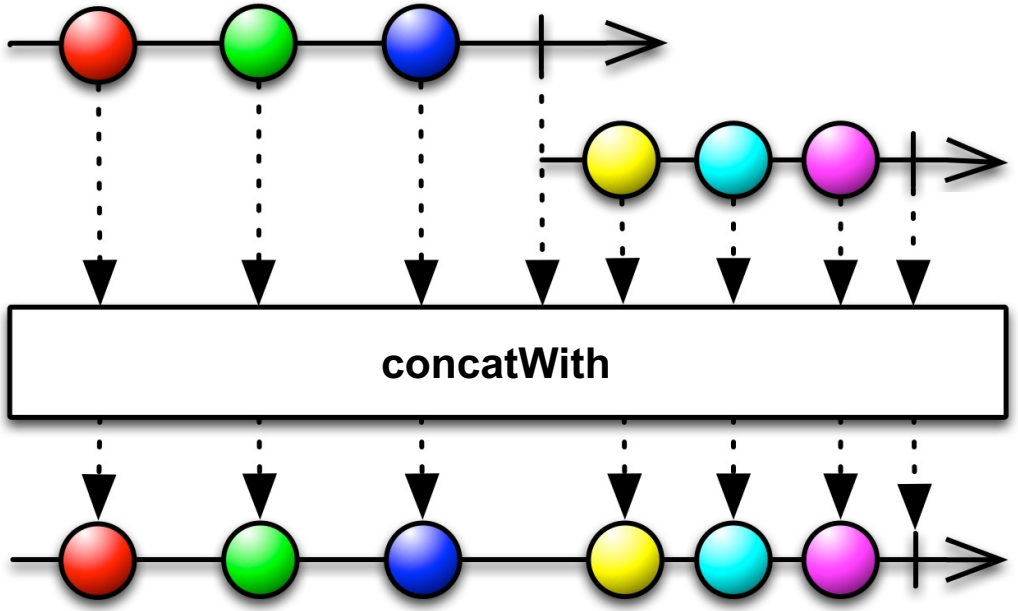
Key Combining Operators in the Observable Class

- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other param
  - This operator combines items emitted by multiple Observable Sources so that they appear as a single ObservableSource
    - This merging may interleave the items



Key Combining Operators in the Observable Class

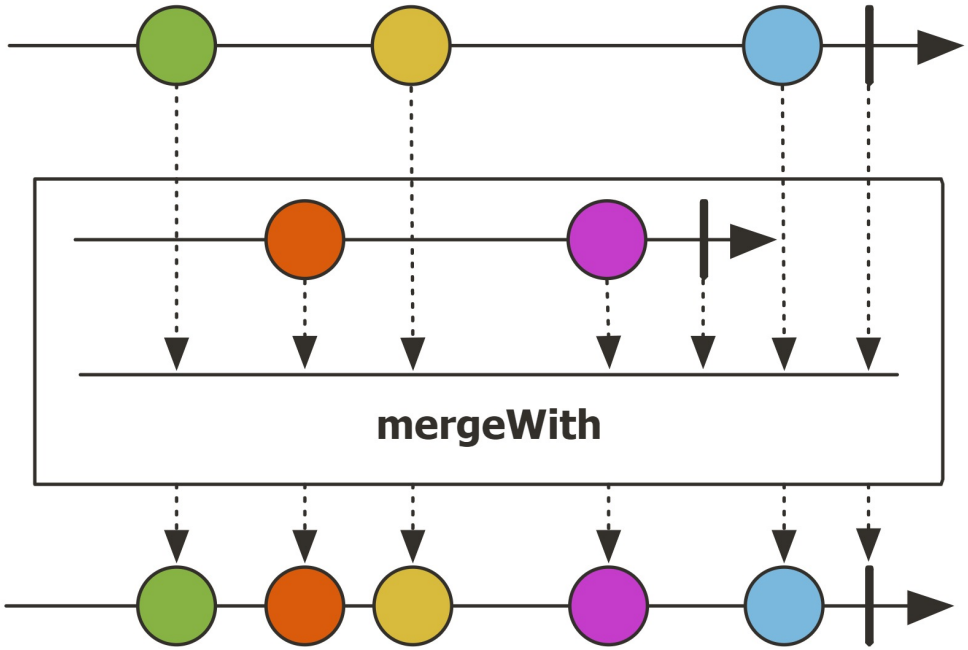
- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other param
  - This operator combines items emitted by multiple Observable Sources so that they appear as a single ObservableSource
    - This merging may interleave the items
    - Use concatWith() to avoid interleaving



See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#concatWith

Key Combining Operators in the Observable Class

- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other paran
  - This operator combines items emitted by multiple Observable Sources so that they appear as a single ObservableSource
  - Project Reactor's operator Flux.mergeWith() works the same




```
Flux<BigFraction> f1 ...
Flux<BigFraction> f2 ...
f1.mergeWith(f2)...
```

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Flux.html#mergeWith

Key Combining Operators in the Observable Class

- The mergeWith() operator
  - Merges the sequence of items of this Observable with the success value of the other param
  - This operator combines items emitted by multiple Observable Sources so that they appear as a single ObservableSource
  - Project Reactor's operator Flux.mergeWith() works the same
- Similar to the Stream.concat() method in Java Streams

concat

```
static <T> Stream<T> concat(Stream<? extends T> a,
                            Stream<? extends T> b)
```

Creates a lazily concatenated stream whose elements are all the elements of the first stream followed by all the elements of the second stream. The resulting stream is ordered if both of the input streams are ordered, and parallel if either of the input streams is parallel. When the resulting stream is closed, the close handlers for both input streams are invoked.

```
List<String> concats
        (List<String> l, int n) {
  Stream<String> s = Stream.empty();
  while (--n >= 0)
    s = Stream.concat(s, l.stream());
  return s.toList();
}
```

See docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#concat

End of Key Combining Operators in the Observable Class (Part 1)