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 Observable operators
  • 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

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

See [reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#repeat](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
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)`
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.

Generate 4 random, reduced big fractions

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

See [Reactive/Observable/ex1/src/main/java/ObservableEx.java](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

```java
Flux.from(Mono.fromCallable(() -> BigFractionUtils.makeBigFraction(random, true))).repeat(4);
```

Generate 4 random, reduced big fractions

See [projectreactor.io/docs/core/release/api/reactor/core/publisher/Flux.html#repeat](http://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

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

See [reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#mergeWith](http://reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#mergeWith)
• 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

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

See [reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/ObservableSource.html](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
The `mergeWith()` operator

Merges the sequence of items of this Observable with the success value of the other parameter.

This operator combines items emitted by multiple Observable Sources so that they appear as a single ObservableSource.

```java
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 parameter.
  • 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](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 parameter.
  - 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](http://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

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

See [docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#concat](docs.oracle.com/javase/8/docs/api/java/util/stream/Stream.html#concat)
End of Key Combining Operators in the Observable Class (Part 1)