Key Action Operators in the Flux 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 Flux operators
  - Factory method operators
  - Transforming operators
- Action operators
  - These operators don’t modify a Flux, but instead use it for side effects
    - e.g., doOnNext()
Key Action Operators in the Flux Class
Key Action Operators in the Flux Class

- The doOnNext() operator
- Add a behavior triggered when a Flux emits an item

```java
Flux<T> doOnNext
    (Consumer<? super T> onNext)
```

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Flux.html#doOnNext
Key Action Operators in the Flux Class

- The doOnNext() operator
- Add a behavior triggered when a Flux emits an item
  - The behavior is passed as a consumer param that’s called on successful completion

```
Flux<T> doOnNext
    (Consumer<? super T> onNext)
```

**Interface Consumer<T>**

Type Parameters:
T - the type of the input to the operation

All Known Subinterfaces:
Stream.Builder<T>

Functional Interface:
This is a functional interface and can therefore be used as the assignment target for a lambda expression or method reference.

See [docs.oracle.com/javase/8/docs/api/java/util/function/Consumer.html](docs.oracle.com/javase/8/docs/api/java/util/function/Consumer.html)
Key Action Operators in the Flux Class

- The `doOnNext()` operator
  - Add a behavior triggered when a Flux emits an item
    - The behavior is passed as a consumer param that’s called on successful completion
    - i.e., it is a “callback”

```java
Flux<T> doOnNext
    (Consumer<? super T> onNext)
```

See [en.wikipedia.org/wiki/Callback_(computer_programming)](en.wikipedia.org/wiki/Callback_(computer_programming))
Key Action Operators in the Flux Class

• The doOnNext() operator
  • Add a behavior triggered when a Flux emits an item
    • The behavior is passed as a consumer param that’s called on successful completion
  • Returns a Flux that is not modified at all
    • i.e., the type and/or value of its elements are not changed

\[
\text{Flux}<T> \; \text{doOnNext} \\
(\text{Consumer}<? \; \text{super} \; T> \; \text{onNext})
\]
Key Action Operators in the Flux Class

- The `doOnNext()` operator
- Add a behavior triggered when a Flux emits an item
- Used primarily for debugging, logging, and/or getting visibility into a Flux chain

Log each BigFraction value on success (otherwise skip)

```java
Flux.fromIterable(bigFractionList)
    .doOnNext(bf ->
        logBigFraction(sUnreducedFraction, bf, sb))
    ...
```

See `Reactive/flux/ex1/src/main/java/FluxEx.java`
Key Action Operators in the Flux Class

- The `doOnNext()` operator
  - Add a behavior triggered when a Flux emits an item
  - Used primarily for debugging, logging, and/or getting visibility into a Flux chain
- RxJava’s `Observable.doOnNext()` works the same

```java
Observable.fromIterable(bigFractionList)
  .doOnNext(bf -> logBigFraction(sUnreducedFraction, bf, sb))
...```

Key Action Operators in the Flux Class

- The `doOnNext()` operator
  - Add a behavior triggered when a Flux emits an item
  - Used primarily for debugging, logging, and/or getting visibility into a Flux chain
  - RxJava’s `Observable.doOnNext()` works the same
- Similar to `Stream.peek()` in Java Streams

```java
List<String> collect = List.of("a", "b", "c").stream().peek(System.out::println)
  .map(String::toUpperCase).collect(toList());
```

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