Key Action Operators in the Mono Class

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 Mono operators
  - Factory method operators
  - Transforming operators
  - Action operators
    - These operators don't modify a Mono, but instead uses it for side effects
      - e.g., doOnSuccess()

Key Action Operators in the Mono Class

Key Action Operators in the Mono Class

- The doOnSuccess() operator
  - Add a behavior triggered when the Mono completes successfully

```
Mono<T> doOnSuccess
   (Consumer<? super T>
    onSuccess)
```

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html#doOnSuccess

Key Action Operators in the Mono Class

- The doOnSuccess() operator
  - Add a behavior triggered when the Mono completes successfully
    - The behavior is passed as a consumer param that's called on successful completion

```
Mono<T> doOnSuccess
   (Consumer<? super T>
    onSuccess)
```

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

Key Action Operators in the Mono Class

- The doOnSuccess() operator
  - Add a behavior triggered when the Mono completes successfully
    - The behavior is passed as a consumer param that's called on successful completion
      - i.e., it's a "callback"

```
Mono<T> doOnSuccess
   (Consumer<? super T>
    onSuccess)
```

See en.wikipedia.org/wiki/Callback_(computer_programming)

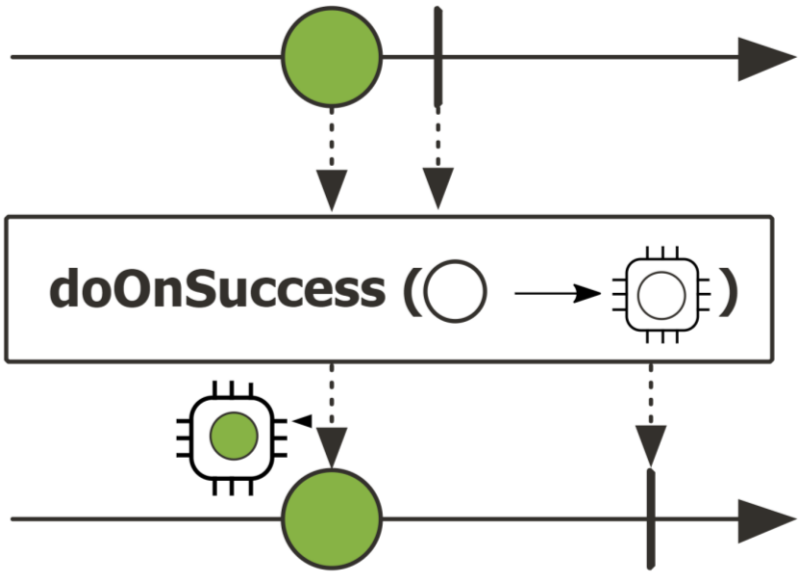
Key Action Operators in the Mono Class

- The doOnSuccess() operator
  - Add a behavior triggered when the Mono completes successfully
    - The behavior is passed as a consumer param that's called on successful completion
  - Returns a Mono that is not modified at all
    - i.e., the type and/or value of its element is not changed

```
Mono<T> doOnSuccess
   (Consumer<? super T>
    onSuccess)
```

Key Action Operators in the Mono Class

- The doOnSuccess() operator
  - Add a behavior triggered when the Mono completes successfully
  - Used primarily for debugging or logging into a Mono chain



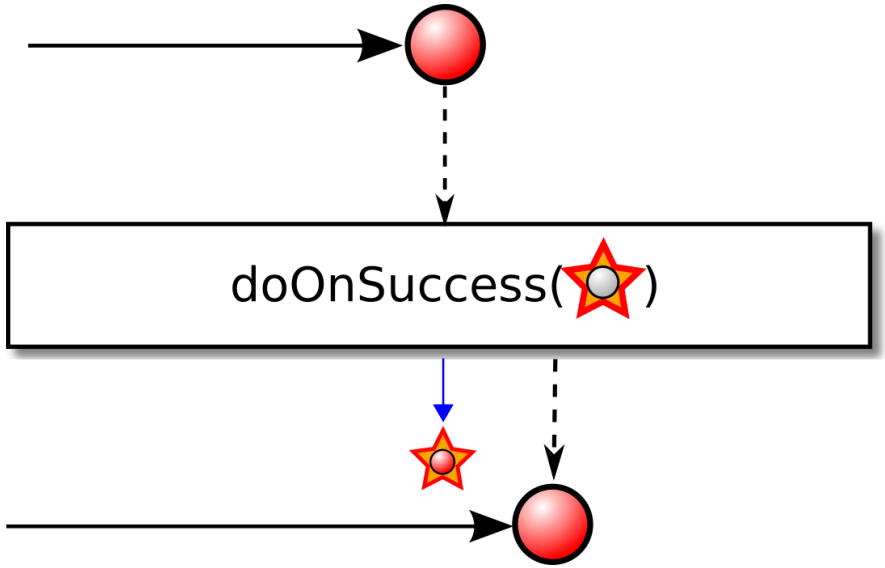
Log the BigFraction value on success (otherwise skip)

```
Mono
   .fromCallable(() -> BigFraction.reduce(sUnreducedFraction))
   .doOnSuccess(bf ->
                logBigFraction(sUnreducedFraction, bf, sb))
   ...
```

See Reactive/mono/ex1/src/main/java/MonoEx.java

Key Action Operators in the Mono Class

- The doOnSuccess() operator
  - Add a behavior triggered when the Mono completes successfully
  - Used primarily for debugging or logging into a Mono chain
  - RxJava's Single.doOnSuccess() works the same way



Log the BigFraction value on success (otherwise skip)

```
Single
   .fromCallable(() -> BigFraction.reduce(sUnreducedFraction))
   .doOnSuccess(bf ->
                logBigFraction(sUnreducedFraction, bf, sb))
   ...
```

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Single.html#doOnSuccess

Key Action Operators in the Mono Class

- The doOnSuccess() operator
  - Add a behavior triggered when the Mono completes successfully
  - Used primarily for debugging or logging into a Mono chain
  - RxJava's Single.doOnSuccess() works the same way
  - Similar to the Java Completable Future thenAccept() operator

thenAccept

```
public CompletableFuture<Void> thenAccept(Consumer<? super T> action)
```

Description copied from interface: CompletionStage

Returns a new CompletionStage that, when this stage completes normally, is executed with this stage's result as the argument to the supplied action. See the CompletionStage documentation for rules covering exceptional completion.

Specified by:

thenAccept in interface CompletionStage<T>

Parameters:

action - the action to perform before completing the returned CompletionStage

Returns:

the new CompletionStage

```
CompletableFuture.supplyAsync(reduce)
                 .thenApply(BigFraction::toMixedString)
                 .thenAccept(System.out::println);
```

Print results to standard output

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletableFuture.html#thenAccept

End of Key Action Operators in the Mono Class