Key Factory Method 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

```java
public abstract class Mono<T>
    extends Object
    implements CorePublisher<T>

A Reactive Streams Publisher with basic rx operators that completes successfully by emitting an element, or with an error.

The recommended way to learn about the Mono API and discover new operators is through the reference documentation, rather than through this javadoc (as opposed to learning more about individual operators). See the “which operator do I need?” appendix.
```

See [projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html](projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html)
Learning Objectives in this Part of the Lesson

• Recognize key Mono operators
  • Factory method operators
    • These operators create Mono objects in various ways
      • e.g., just() & fromCallable()

See en.wikipedia.org/wiki/Factory_method_pattern
Key Factory Method
Operators in the Mono Class
Key Factory Method Operators in the Mono Class

- The just() method
  - Create a new Mono that emits the specified item

static <T> Mono<T> just(T data)

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html#just
Key Factory Method Operators in the Mono Class

• The just() method
  • Create a new Mono that emits the specified item
    • The param is the one & only item emitted to onNext()
Key Factory Method Operators in the Mono Class

- The just() method

  - Create a new Mono that emits the specified item
    - The param is the one & only item emitted to onNext()
    - Returns the Mono that’s captured at instantiation time
      - i.e., just() is “eager” & it therefore always runs in the thread where the “assembly” is performed

Keep these semantics in mind when we talk about the subscribeOn() operator..
Key Factory Method Operators in the Mono Class

- The just() method
  - Create a new Mono that emits the specified item
  - This factory method adapts non-reactive input sources into the reactive model

Create a Mono that reduces a big fraction

```
Mono
  .just(BigFraction
    .reduce(unreducedFraction))
...```

See Reactive/mono/ex1/src/main/java/MonoEx.java
The just() method

- Create a new Mono that emits the specified item
- This factory method adapts non-reactive input sources into the reactive model

```
Mono.just(BigFraction.reduce(unreducedFraction))
```

This value is captured “eagerly” at instantiation time & is the value returned for all subscribers
Key Factory Method Operators in the Mono Class

- The just() method
  - Create a new Mono that emits the specified item
  - This factory method adapts non-reactive input sources into the reactive model
  - RxJava’s Single.just() works the same way

Create a Single that reduces a big fraction

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Single.html#just
Key Factory Method Operators in the Mono Class

• The fromCallable() operator
• This factory method creates a Mono of type T

static <T> Mono<T> fromCallable
(Callable<? extends T> supplier)

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html#fromCallable
Key Factory Method Operators in the Mono Class

- The `fromCallable()` operator
- This factory method creates a Mono of type T
- The Mono’s value is produced via the provided Callable supplier

```java
static <T> Mono<T> fromCallable
    (Callable<? extends T> supplier)
```

**Interface Callable<V>**

- **Type Parameters:**
  - V - the result type of method call

- **All Known Subinterfaces:**

- **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/concurrent/Callable.html](docs.oracle.com/javase/8/docs/api/java/util/concurrent/Callable.html)
Key Factory Method Operators in the Mono Class

• The `fromCallable()` operator
  • This factory method creates a Mono of type `T`
    • The Mono’s value is produced via the provided Callable supplier
    • Returns the Mono emitted by the supplier at runtime
      • i.e., it’s “lazy”

```
static <T> Mono<T> fromCallable (Callable<? extends T> supplier)
```

Again, keep these semantics in mind when we talk about `subscribeOn()` later..
Key Factory Method Operators in the Mono Class

- The fromCallable() operator
  - This factory method creates a Mono of type T
  - This factory method adapts non-reactive input sources into the reactive model

Create a Mono that reduces a big fraction

See Reactive/mono/ex1/src/main/java/MonoEx.java
Key Factory Method Operators in the Mono Class

- The `fromCallable()` operator
  - This factory method creates a Mono of type T
  - This factory method adapts non-reactive input sources into the reactive model
  - Unlike `just()`, `fromCallable()` is "lazy"

It invokes the callable param at the time of subscription & separately for each subscriber

See earlier discussion of the Mono.`just()` operator
**Key Factory Method Operators in the Mono Class**

- The `fromCallable()` operator
  - This factory method creates a Mono of type T
  - This factory method adapts non-reactive input sources into the reactive model

- RxJava’s `Single.fromCallable()` works the same way

*It invokes the callable param at the time of subscription & separately for each subscriber*

See [reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Single.html#fromCallable](reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Single.html#fromCallable)
End of Key Factory Method
Operators in the Mono Class