Key Terminal Operators in the Flux 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 Flux operators
  - Factory method operators
  - Transforming operators
  - Action operators
  - Combining operators
- Terminal operators
  - Terminate a Flux stream & trigger all the processing of operators in the stream
    - e.g., subscribe()
Key Terminal Operators in the Flux Class
Key Terminal Operators in the Flux Class

- The subscribe() operator
- Subscribe a Consumer to this Flux

Disposable `subscribe`
(Consumer<? super T> consumer,
Consumer<? super Throwable> errorConsumer,
Runnable completeConsumer)
Key Terminal Operators in the Flux Class

- The subscribe() operator
- Subscribe a Consumer to this Flux
- This operator consumes all elements in the sequence, handles errors, & reacts to completion

```java
def disposableSubscribe:
    Disposable subscribe(
        Consumer<? super T> consumer,
        Consumer<? super Throwable> errorConsumer,
        Runnable completeConsumer
    )
```

**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](http://docs.oracle.com/javase/8/docs/api/java/util/function/Consumer.html)
Key Terminal Operators in the Flux Class

- The subscribe() operator
  - Subscribe a Consumer to this Flux
  - This operator consumes all elements in the sequence, handles errors, & reacts to completion
  - This subscription requests unbounded demand
  - i.e., Long.MAX_VALUE

```java
Disposable subscribe(
    Consumer<? super T> consumer,
    Consumer<? super Throwable> errorConsumer,
    Runnable completeConsumer)
```
Key Terminal Operators in the Flux Class

• The subscribe() operator
• Subscribe a Consumer to this Flux
  • This operator consumes all elements in the sequence, handles errors, & reacts to completion
  • This subscription requests unbounded demand
• Signals emitted to this method are represented by the following regular expression:
  onNext()*(onComplete()|onError())?
Key Terminal Operators in the Flux Class

- The `subscribe()` operator
  - Subscribe a Consumer to this Flux
  - This operator consumes all elements in the sequence, handles errors, & reacts to completion
  - A `Disposable` is returned, which indicates a task or resource that can be cancelled/disposed

```java
Disposable subscribe
(Consumer<? super T> consumer,
 Consumer<? super Throwable>
 errorConsumer,
 Runnable completeConsumer)
```

**Interface Disposable**

All Known Subinterfaces:
Disposible.Composite, Disposable.Swap, Scheduler, Scheduler.Worker

All Known Implementing Classes:
BaseSubscriber, DirectProcessor, EmitterProcessor, FluxProcessor,
MonoProcessor, ReplayProcessor, Schedulers.Snapshot,
UnicastProcessor

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

See [projectreactor.io/docs/core/release/api/reactor/core/Disposable.html](http://projectreactor.io/docs/core/release/api/reactor/core/Disposable.html)
Key Terminal Operators in the Flux Class

- The subscribe() operator
- Subscribe a Consumer to this Flux
  - This operator consumes all elements in the sequence, handles errors, & reacts to completion
- A Disposable is returned, which indicates a task or resource that can be cancelled/disposed
  - Disposables can be accumulated & disposed in one fell swoop!

```java
Disposable.Composite mDisposables;

...

mDisposables = Disposables.composite
    (mPublisherScheduler,
     mSubscriberScheduler,
     mSubscriber);

...

mDisposables.dispose();
```

See projectreactor.io/docs/core/release/api/reactor/core/Disposable.Composite.html
Key Terminal Operators in the Flux Class

- The subscribe() operator
  - Subscribe a Consumer to this Flux
  - This operator triggers all the processing in a chain

```java
Flux
  .fromIterable
    (bigFractionList)
  .map(fraction -> fraction
       .multiply(sBigReducedFraction))
  .subscribe(fraction -> sb
       .append(" = " + fraction
       .toMixedString() + "\n"),
  error -> { sb
       .append("error"); ... },
  () -> BigFractionUtils.display(sb.toString()));
```

Initiate processing & handle outputs

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

- The subscribe() operator
- Subscribe a Consumer to this Flux
- This operator triggers all the processing in a chain

Flux
  .fromIterable
    (bigFractionList)
  .map(fraction -> fraction
        .multiply(sBigReducedFraction))
  .subscribe(fraction -> sb
        .append(" = "
        + fraction
        .toMixedString() + "\n"),
    error -> { sb
        .append("error");...
    },
  () -> BigFractionUtils.display(sb.toString()));
Key Terminal Operators in the Flux Class

- The subscribe() operator
  - Subscribe a Consumer to this Flux
  - This operator triggers all the processing in a chain

```java
Flux
  .fromIterable(bigFractionList)
  .map(fraction -> fraction
       .multiply(sBigReducedFraction))
  .subscribe(fraction -> sb
       .append(" = " + fraction
       .toMixedString() + "\n"),
        error -> { sb
                  .append("error"); ... },
  () -> BigFractionUtils.display(sb.toString()));
```

Error Processing
Key Terminal Operators in the Flux Class

• The subscribe() operator
  • Subscribe a Consumer to this Flux
  • This operator triggers all the processing in a chain

Flux
  .fromIterable
    (bigFractionList)
  .map(fraction -> fraction
    .multiply(sBigReducedFraction))
  .subscribe(fraction -> sb
    .append(" = "
    + fraction
    .toMixedString()
    + "\n"),
  error -> { sb
    .append("error"); ...
  },
  () -> BigFractionUtils
  .display(sb.toString()));
Key Terminal Operators in the Flux Class

- The subscribe() operator
  - Subscribe a Consumer to this Flux
  - This operator triggers all the processing in a chain
  - Calling this method will *not* block the caller thread
  - For async streams this method returns & processing continues until the upstream terminates normally or with an error
Key Terminal Operators in the Flux Class

- The subscribe() operator
  - Subscribe a Consumer to this Flux
  - This operator triggers all the processing in a chain
  - Calling this method will not block the caller thread
    - For async streams this method returns & processing continues until the upstream terminates normally or with an error
  - These semantics motivate the need for the AsyncTaskBarrier framework!

See Reactive/mono/ex1/src/main/java/utils/AsyncTaskBarrier.java
Key Terminal Operators in the Flux Class

• The subscribe() operator
  • Subscribe a Consumer to this Flux
  • This operator triggers all the processing in a chain
  • Calling this method will not block the caller thread
  • Other versions of subscribe() support different capabilities

Disposable subscribe
(Consumer<? super T> consumer,
Consumer<? super Throwable> errorConsumer,
Runnable completeConsumer,
Consumer<? super Subscription> subscriptionConsumer)

Pass a custom Consumer called on initial subscribe() signal that can apply backpressure & other features

See [projectreactor.io/docs/core/release/api/reactor/core/publisher/Flux.html#subscribe](http://projectreactor.io/docs/core/release/api/reactor/core/publisher/Flux.html#subscribe)
Key Terminal Operators in the Flux Class

- The subscribe() operator
  - Subscribe a Consumer to this Flux
  - This operator triggers all the processing in a chain
  - Calling this method **not** block the caller thread
  - Other versions of subscribe() support different capabilities
  - RxJava’s Observable.subscribe() works the same

See [reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#subscribe](reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Observable.html#subscribe)
End of Key Terminal Operators in the Flux Class