Overview of Key Classes in the RxJava API

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

• Understand key classes in the RxJava API
• Understand key classes in the RxJava API

Flowable & Observable
Key Classes in the RxJava API
Key Classes in the RxJava API

- There are three key classes in the RxJava API
Key Classes in the RxJava API

- There are three key classes in the RxJava API
- **Single** – Completes successfully or with failure, may or may not emit a single value

---

**Class Single<T>**

```java
public abstract class Single<T>
    extends Object
    implements SingleSource<T>

The Single class implements the Reactive Pattern for a single value response.

Single behaves similarly to Observable except that it can only emit either a single successful value or an error (there is no onComplete notification as there is for an Observable).

The Single class implements the SingleSource base interface and the default consumer type it interacts with is the SingleObserver via the subscribe(SingleObserver) method.
```

Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - Similar to a Java Completable Future or an async Optional<T>

```java
BigFraction unreducedFraction = makeBigFraction(...);

Single
  .fromCallable(() -> BigFraction
    .reduce(unreducedFraction))
  .subscribeOn
    (Schedulers.single())
  .map(result ->
    result.toMixedString())
  .doOnSuccess(result ->
    System.out.println
      ("big fraction = " + result + "\n"));
```
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
    - Similar to a Java Completable Future or an async Optional\(<T>\)
    - Can be documented via a “marble diagram”

See [medium.com/@jshvarts/read-marble-diagrams-like-a-pro-3d72934d3ef5](medium.com/@jshvarts/read-marble-diagrams-like-a-pro-3d72934d3ef5)
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
    - Similar to a Java Completable Future or an async Optional<T>
    - Can be documented via a “marble diagram”
Key Classes in the RxJava API

- There are three key classes in the RxJava API

  - **Single** – Completes successfully or with failure, may or may not emit a single value
    - Similar to a Java Completable Future or an async Optional<T>
    - Can be documented via a “marble diagram”

This is the optional item emitted by the Single
Key Classes in the RxJava API

• There are three key classes in the RxJava API
  • **Single** – Completes successfully or with failure, may or may not emit a single value
    • Similar to a Java Completable Future or an async Optional\(<T>\)
    • Can be documented via a “marble diagram”
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
    - Similar to a Java Completable Future or an async Optional<T>
    - Can be documented via a “marble diagram”
There are three key classes in the RxJava API

**Single** – Completes successfully or with failure, may or may not emit a single value
- Similar to a Java Completable Future or an async Optional<T>
- Can be documented via a “marble diagram”
There are three key classes in the RxJava API:

- **Single** – Completes successfully or with failure, may or may not emit a single value.
  - Similar to a Java Completable Future or an async Optional<T>.
  - Can be documented via a “marble diagram”.

If the Single terminates abnormally the vertical line is replaced by an X.
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite # of events (zero to infinite) & may complete successfully or with a failure

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Flowable.html
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite number of events (zero to infinite) & may complete successfully or with a failure
    - Similar to an async Java stream
      - i.e., completable futures used with a Java stream

```java
def reduce
fromCallable(() -> BigFraction.reduce(unreducedFraction))
.subscribeOn(scheduler)
.flatMap(reducedFraction ->
  Observable.fromCallable(() ->
    reducedFraction.multiply(sBigReducedFraction))
.subscribeOn(scheduler));
```
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite # of events (zero to infinite) & may complete successfully or with a failure
    - Similar to an async Java stream
    - Can also be documented via a marble diagram

See medium.com/@jshvarts/read-marble-diagrams-like-a-pro-3d72934d3ef5
There are three key classes in the RxJava API:

- **Single** – Completes successfully or with failure, may or may not emit a single value.
- **Observable** – Emits an indefinite # of events (zero to infinite) & may complete successfully or with a failure.
  - Similar to an async Java stream.
  - Can also be documented via a marble diagram.

This is the timeline of an Observable, where time flows from left to right.
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite number of events (zero to infinite) & may complete successfully or with a failure
    - Similar to an async Java stream
    - Can also be documented via a marble diagram

These are the items emitted by the Observable
There are three key classes in the RxJava API:

- **Single** – Completes successfully or with failure, may or may not emit a single value.
- **Observable** – Emits an indefinite number of events (zero to infinite) & may complete successfully or with a failure.
  - Similar to an async Java stream.
  - Can also be documented via a marble diagram.

These dotted lines & this box indicate that a transformation is being applied to the Observable.

The text inside the box indicates the type of transformation.
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite # of events (zero to infinite) & may complete successfully or with a failure
    - Similar to an async Java stream
    - Can also be documented via a marble diagram

This Observable is the result of the transformation
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite # of events (zero to infinite) & may complete successfully or with a failure
    - Similar to an async Java stream
    - Can also be documented via a marble diagram

This vertical line indicates the Observable completed successfully
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite number of events (zero to infinite) & may complete successfully or with a failure
    - Similar to an async Java stream
    - Can also be documented via a marble diagram

If an Observable terminates abnormally the vertical line is replaced by an X
Key Classes in the RxJava API

- There are three key classes in the RxJava API
  - **Single** – Completes successfully or with failure, may or may not emit a single value
  - **Observable** – Emits an indefinite # of events (zero to infinite) & may complete successfully or with a failure
  - **Flowable** – Generalizes Observable to support backpressure

---

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Flowable.html
There are three key classes in the RxJava API:

- **Single** – Completes successfully or with failure, may or may not emit a single value.
- **Observable** – Emits an indefinite number of events (zero to infinite) & may complete successfully or with a failure.
- **Flowable** – Generalizes Observable to support backpressure.
  - i.e., the publisher can be slowed down when the consumer can’t keep up.

See [www.baeldung.com/rxjava-backpressure](http://www.baeldung.com/rxjava-backpressure)
End of Overview of Key Classes in the RxJava API