Applying Key Methods in the Single Class (Part 2)

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 methods in the Single class & how they are applied in the case studies
  - Case study ex1
  - Case study ex2

See [github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/Single/ex2](github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/Single/ex2)
Applying Key Methods in the Single Class in ex2
### Applying Key Methods in the Single Class in ex2

- ex2 shows how to apply RxJava features *asynchronously* to perform various Single operations
  - e.g., `fromCallable()`, `doOnSuccess()`, `ignoreElement()`, `subscribeOn()`, `map()`, `blockingGet()`, & `Schedulers.single()`

```java
return Single
    .fromCallable(reduceFraction)
    .subscribeOn(Schedulers.single())
    .map(convertToMixedString)
    .doOnSuccess(printResult)
    .ignoreElement();
```

See [github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/Single/ex2](github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/Single/ex2)
Applying Key Methods in the Single Class in ex2

- The `subscribeOn()` method
- Run `subscribe()`, `onSubscribe()`, & `request()` on the specified Scheduler worker

```java
Single<T> subscribeOn(Scheduler scheduler)
```

Applying Key Methods in the Single Class in ex2

• The subscribeOn() method
  • Run subscribe(), onSubscribe(), & request() on the specified Scheduler worker
  • The scheduler param indicates what thread to perform the operation on

```
Single<T> subscribeOn(Scheduler scheduler)
```

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Scheduler.html
Applying Key Methods in the Single Class in ex2

- The `subscribeOn()` method
  - Run `subscribe()`, `onSubscribe()`, & `request()` on the specified Scheduler worker
    - The scheduler param indicates what thread to perform the operation on
  - Returns the Single requesting async processing

```java
Single<T> subscribeOn(Scheduler scheduler)
```
Applying Key Methods in the Single Class in ex2

• The subscribeOn() method
  • Run subscribe(), onSubscribe(), & request() on the specified Scheduler worker

• The semantics of subscribeOn() are a bit unusual
Applying Key Methods in the Single Class in ex2

• The subscribeOn() method
  • Run subscribe(), onSubscribe(), & request() on the specified Scheduler worker

• The semantics of subscribeOn() are a bit unusual
  • Placing this operator in a chain impacts the execution context of the onNext(),.onError(), & onComplete() signals
  • i.e., from beginning of chain up to next occurrence of a observeOn()

```java
return Single
    .fromCallable(reduceFraction)
    .subscribeOn(Schedulers.single())
    .map(convertToMixedString)
    .doOnSuccess(printResult)
    .ignoreElement();
```

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Single.html#observeOn
Applying Key Methods in the Single Class in ex2

• The `subscribeOn()` method
  • Run `subscribe()`, `onSubscribe()`, & `request()` on the specified Scheduler worker
  • The semantics of `subscribeOn()` are a bit unusual
  • Project Reactor’s method `Mono.subscribeOn()` works the same way

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html#subscribeOn
Applying Key Methods in the Single Class in ex2

- The blockingGet() method

  - Block until the current Single signals a success value (which is returned) or an exception (which is propagated)

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Single.html#blockingGet
Applying Key Methods in the Single Class in ex2

- The blockingGet() method
- Block until the current Single signals a success value (which is returned) or an exception (which is propagated)
- If the source signals an error, the operator wraps a checked exception into a Runtime Exception & throws that
Applying Key Methods in the Single Class in ex2

- The blockingGet() method
  - Block until the current Single signals a success value (which is returned) or an exception (which is propagated)

- Project Reactor’s method Mono. blockOptional() is similar
  - i.e., it block indefinitely until a next signal is received or the Mono completes empty

See projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html#blockOptional
Applying Key Methods in the Single Class in ex2

- The Schedulers.single() method
- Hosts a single-threaded Executor Service-based worker that runs concurrently wrt the caller

See [reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/schedulers/Schedulers.html#single](reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/schedulers/Schedulers.html#single)
Applying Key Methods in the Single Class in ex2

- The Schedulers.single() method
  - Hosts a single-threaded Executor Service-based worker that runs concurrently wrt the caller
  - Optimized for low-latency calls that all run in one (& only one) background thread

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/schedulers/Schedulers.html
The Schedulers.single() method

Hosts a single-threaded Executor Service-based worker that runs concurrently wrt the caller

- Optimized for low-latency calls that all run in one (& only one) background thread

- Implemented via a daemon thread that won’t prevent the app from exiting even if its work isn’t done

See www.baeldung.com/java-daemon-thread
Applying Key Methods in the Single Class in ex2

- The Schedulers.single() method
  - Hosts a single-threaded Executor Service-based worker that runs concurrently wrt the caller
- Project Reactor’s Schedulers. single() method works the same

```java
public static Scheduler single()

Scheduler that hosts a single-threaded ExecutorService-based worker and is suited for parallel work. Will cache the returned schedulers for subsequent calls until dispose.

Returns:
default instance of a Scheduler that hosts a single-threaded ExecutorService-based worker
```

See `projectreactor.io/docs/core/release/api/reactor/core/scheduler/Schedulers.html#single`
Applying Key Methods in the Single Class in ex2

```java
/**
 * This class shows how to apply RxJava to asynchronously reduce,
 * multiply, and display BigFractions via various Single operations,
 * including fromCallable(), subscribeOn(), map(), doOnSuccess(),
 * blockingGet(), array(), ignoreElement(), and the
 * Scheduler.single() thread "pool".
 */

public class SingleEx {

    /**
     * Test asynchronous BigFraction reduction using a Single and a
     * pipeline of operations that run in the background (i.e., off
     * the calling thread).
     */

    public static Completable testFractionReductionAsync() {
        StringBuilder sb =
            new StringBuilder(">> Calling testFractionReductionAsync()\n");

        // Create a new unreduced big fraction.
        BigFraction unreducedFraction = BigFraction
            .valueOf(new BigInteger(sb11)),
```

End of Applying Key Methods in the Single Class (Part 2)