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

```
return Single
    .fromCallable(reduceFraction)

    .subscribeOn(Schedulers.single())

    .map(convertToMixedString)

    .doOnSuccess(printResult)

    .ignoreElement();
```

See 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()

```
return Single
  .fromCallable(reduceFraction)

  .subscribeOn(Schedulers.single())

  .map(convertToMixedString)

  .doOnSuccess(printResult)

  .ignoreElement();
```

See github.com/douglascraigschmidt/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

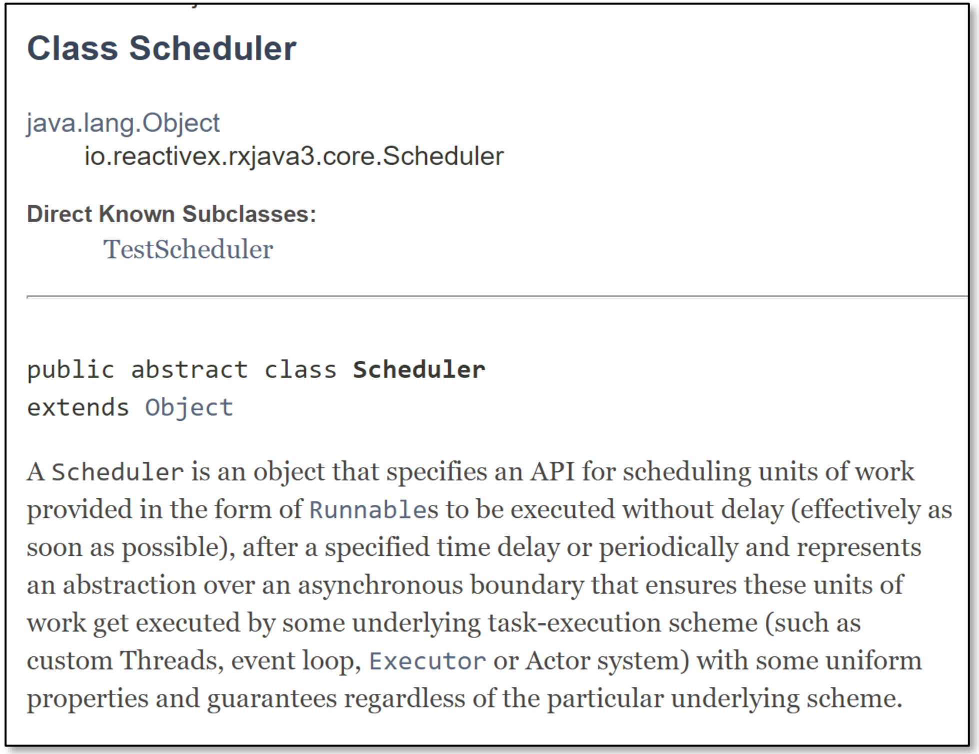
```
Single<T> subscribeOn(Scheduler
                          scheduler)
```

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

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)
```



Class Scheduler

java.lang.Object
    io.reactivex.rxjava3.core.Scheduler

Direct Known Subclasses:
    TestScheduler

```
public abstract class Scheduler
extends Object
```

A `Scheduler` is an object that specifies an API for scheduling units of work provided in the form of `Runnable`s to be executed without delay (effectively as soon as possible), after a specified time delay or periodically and represents an abstraction over an asynchronous boundary that ensures these units of work get executed by some underlying task-execution scheme (such as custom Threads, event loop, `Executor` or Actor system) with some uniform properties and guarantees regardless of the particular underlying scheme.

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/core/Scheduler.html

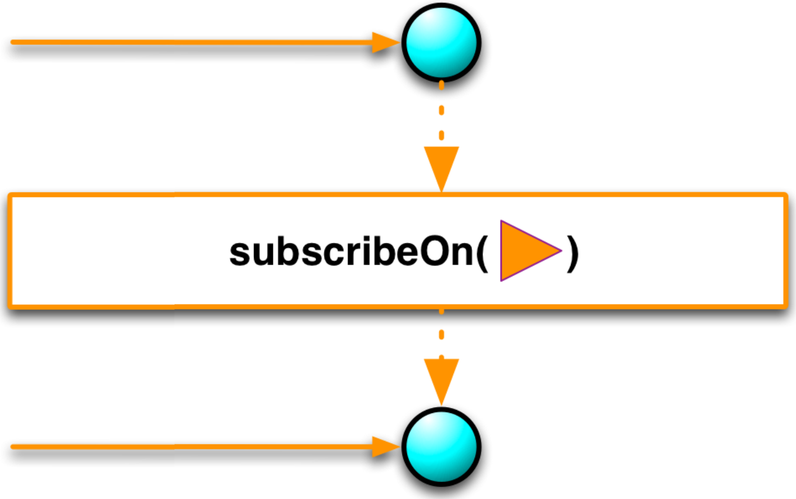
Applying Key Methods in the Single Class in ex2

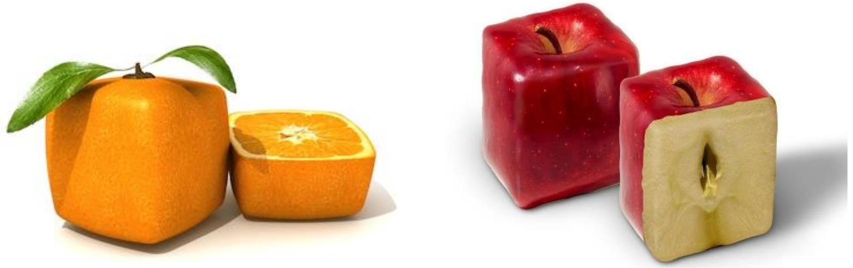
- The subscribeOn() method
  - Run subscribe(), onSubscribe(), & request() on the specified Scheduler worker
    - The scheduler param indicates what thread to perform the operation on
    - Returns the Single requesting async processing

```
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

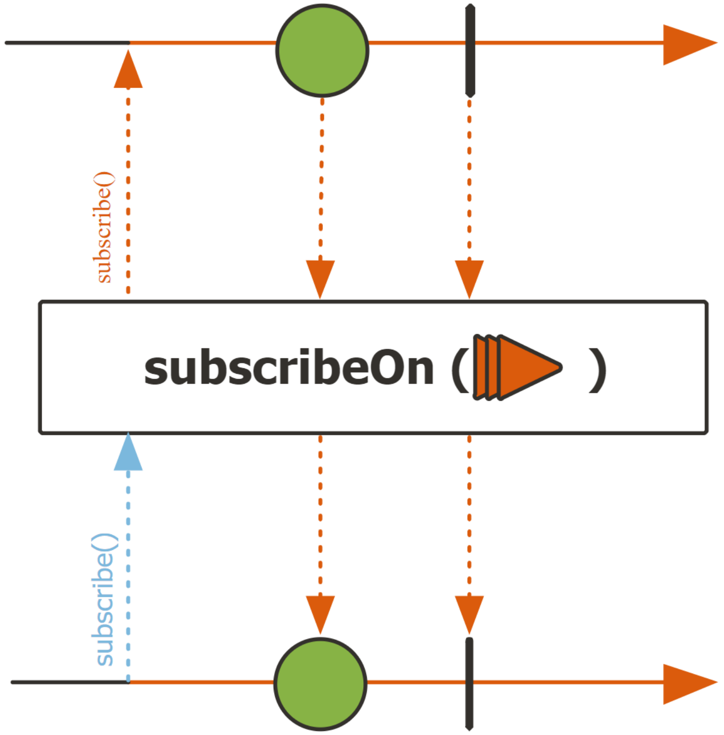
- The subscribeOn() method
  - Run subscribe(), onSubscribe(), & request() on the specified Scheduler worker
  - The semantics of subscribeOn() are a bit unusual
    - Placing this operator in a chain impacts the execution context of the onNext(), onError(), & onComplete() signals
      - i.e., from beginning of chain up to next occurrence of a observeOn()

```
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

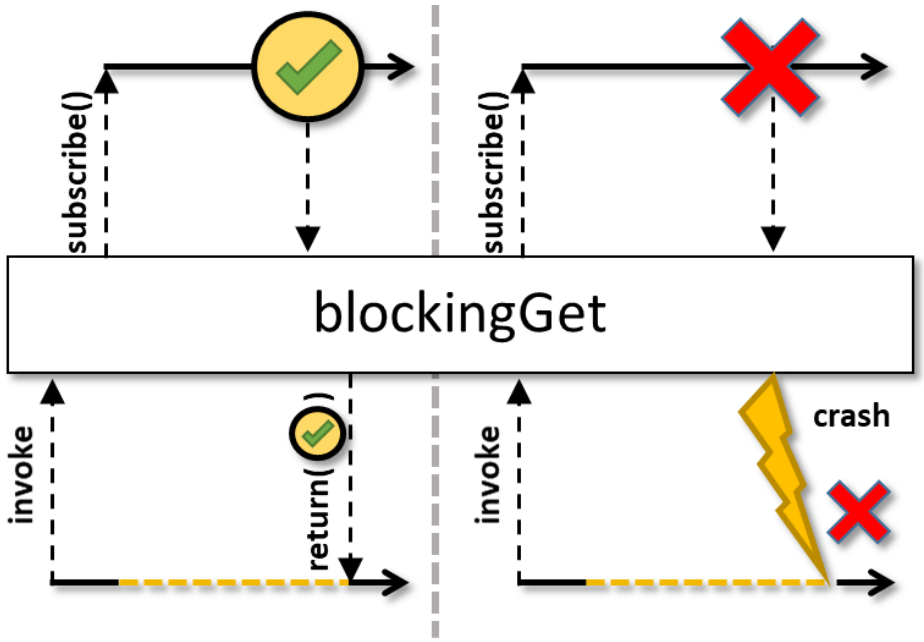
- The blockingGet() method
  - Block until the current Single signals a success value (which is returned) or an exception (which is propagated)

```
T blockingGet()
```

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

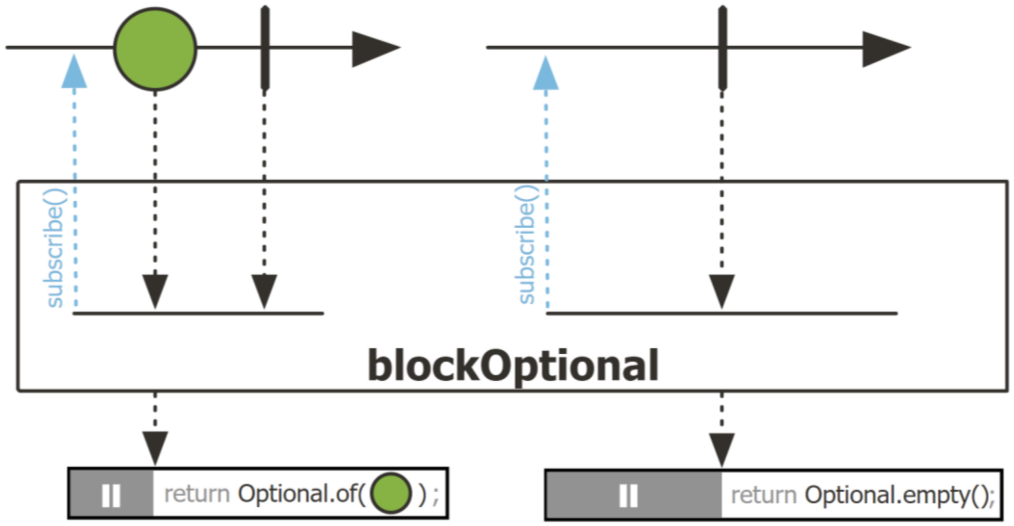
Applying Key Methods in the Single Class in ex2

- 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

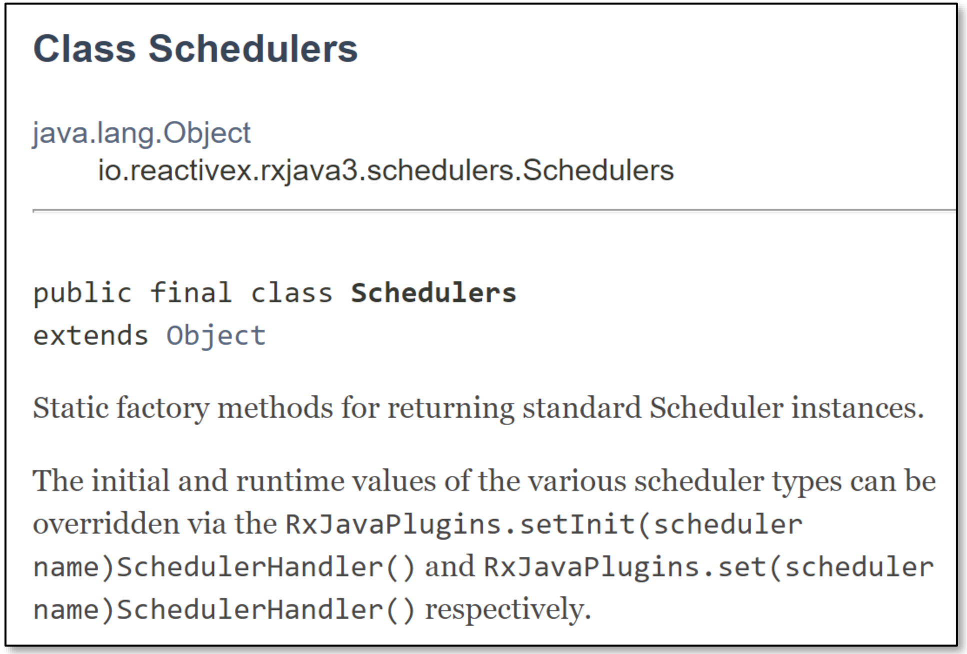
- The Schedulers.single() method
  - Hosts a single-threaded Executor Service-based worker that runs concurrently wrt the caller

```
static Scheduler single()
```

See 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



Class Schedulers

java.lang.Object
    io.reactivex.rxjava3.schedulers.Schedulers

```
public final class Schedulers
extends Object
```

Static factory methods for returning standard Scheduler instances.

The initial and runtime values of the various scheduler types can be overridden via the `RxJavaPlugins.setInit(scheduler name)SchedulerHandler()` and `RxJavaPlugins.set(scheduler name)SchedulerHandler()` respectively.

See reactivex.io/RxJava/3.x/javadoc/io/reactivex/rxjava3/schedulers/Schedulers.html

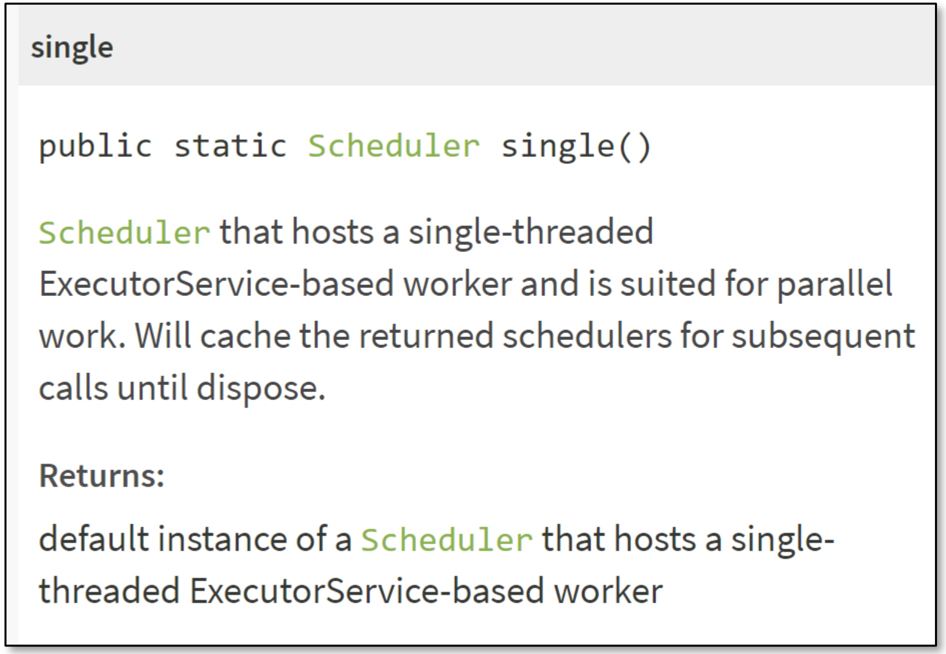
Applying Key Methods in the Single Class in ex2

- 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
    - 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



single

```
public static Scheduler single()
```

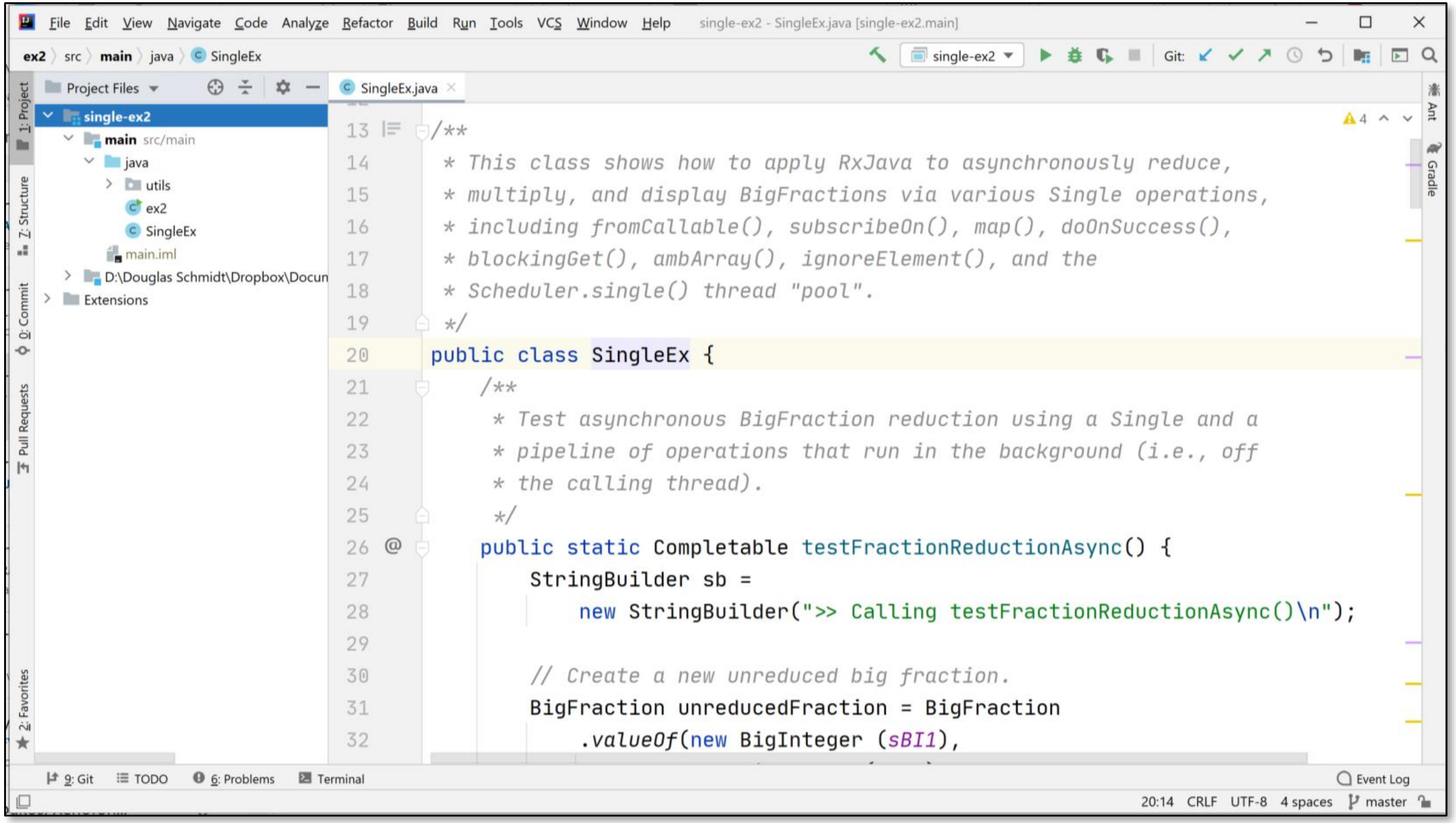
Scheduler that hosts a single-threaded ExecutorService-based worker and is suited for parallel work. Will cache the returned schedulers for subsequent calls until dispose.

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


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
single-ex2 - SingleEx.java [single-ex2.main]
ex2 src main java SingleEx
single-ex2
Git:
Project Files
single-ex2
main src/main
java
utils
ex2
SingleEx
main.iml
D:\Douglas Schmidt\Dropbox\Docun
Extensions
SingleEx.java
1: Project
7: Structure
0: Commit
Pull Requests
2: Favorites
Ant
Gradle
9: Git
TODO
6: Problems
Terminal
Event Log
20:14 CRLF UTF-8 4 spaces master

```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(), ambArray(), 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 (sBI1),
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/Single/ex2

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