Applying Key Methods in the Mono Class: Case Study ex1

Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

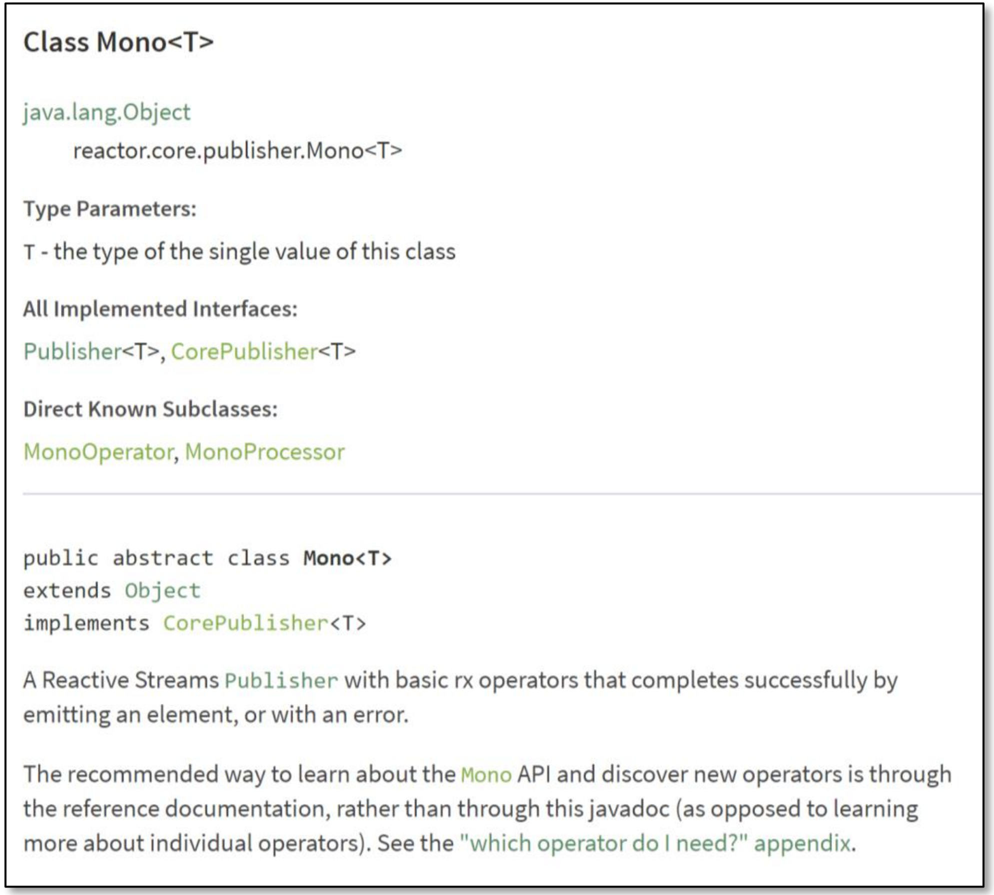
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Recognize key operators in the Mono class & know how they are applied in a detailed case study



Class Mono<T>

java.lang.Object
    reactor.core.publisher.Mono<T>

Type Parameters:

T - the type of the single value of this class

All Implemented Interfaces:

Publisher<T>, CorePublisher<T>

Direct Known Subclasses:

MonoOperator, MonoProcessor

```
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

Learning Objectives in this Part of the Lesson

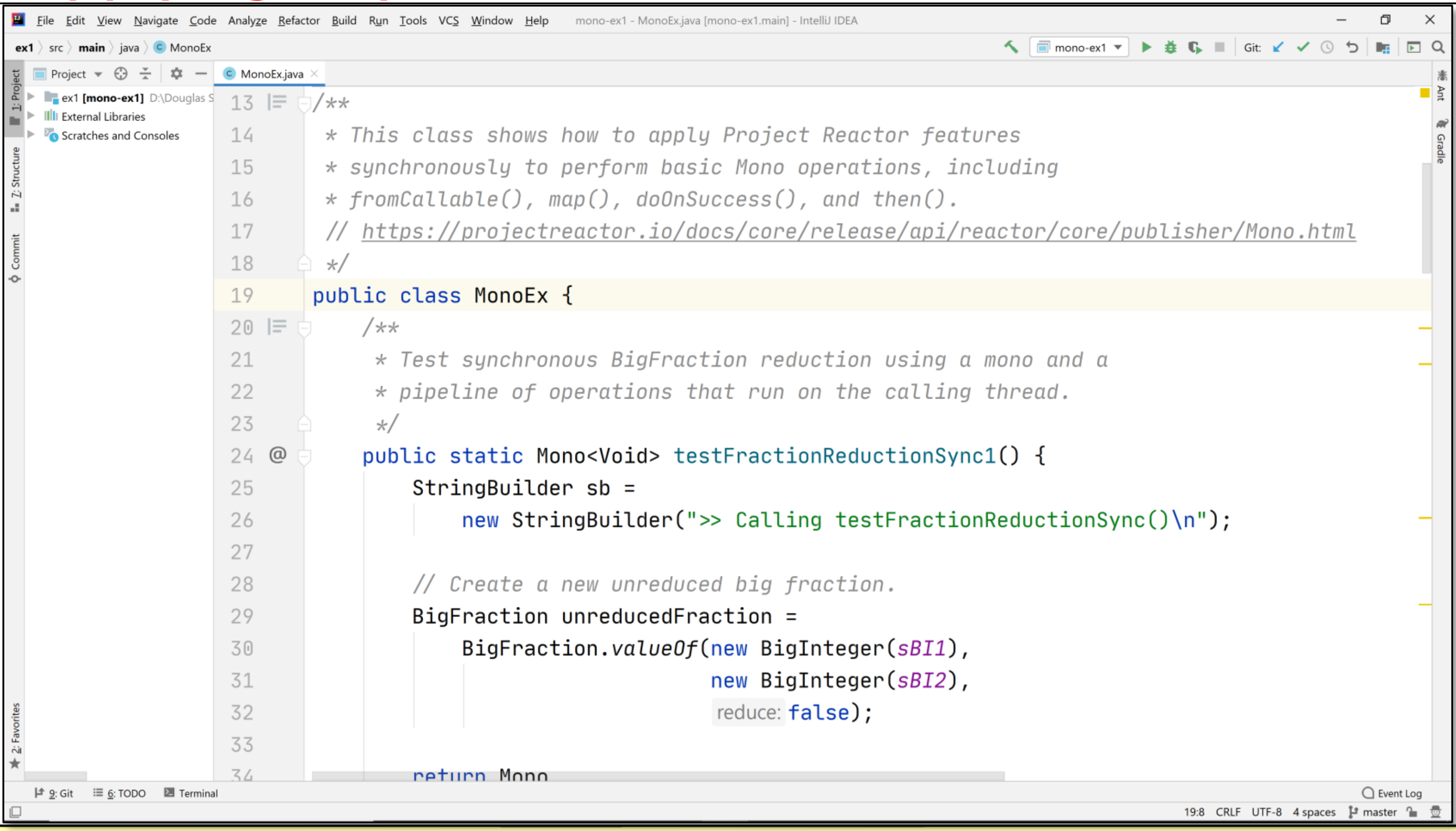
- Recognize key operators in the Mono class & know how they are applied in a detailed case study
  - Case study ex1 shows how to apply the just(), fromCallable(), map(), doOnSuccess(), & then() operators to create, reduce, transform, and display a Big Fraction synchronously

```
return Mono
   .fromCallable(() -> BigFraction
              .reduce(sUnreducedFrac))
   .doOnSuccess(bf ->
                  logBigFraction
                    (sUnreducedFrac,
                     bf, sb))
   .map(BigFraction::toMixedString)
   .doOnSuccess(bf ->
     displayMixedBigFraction(bf,
                              sb))
   .then();
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/mono/ex1

Applying Key Methods in the Mono Class to ex1

Applying Key Methods in the Mono Class to ex1


File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
mono-ex1 - MonoEx.java [mono-ex1.main] - IntelliJ IDEA
ex1 src main java MonoEx
mono-ex1
Git:
Project
ex1 [mono-ex1] D:\Douglas S
External Libraries
Scratches and Consoles
MonoEx.java
9: Git
6: TODO
Terminal
Event Log
19:8 CRLF UTF-8 4 spaces master

```java
/**
 * This class shows how to apply Project Reactor features
 * synchronously to perform basic Mono operations, including
 * fromCallable(), map(), doOnSuccess(), and then().
 // https://projectreactor.io/docs/core/release/api/reactor/core/publisher/Mono.html
 */
public class MonoEx {
    /**
     * Test synchronous BigFraction reduction using a mono and a
     * pipeline of operations that run on the calling thread.
     */
    public static Mono<Void> testFractionReductionSync1() {
        StringBuilder sb =
            new StringBuilder(">> Calling testFractionReductionSync()\n");

        // Create a new unreduced big fraction.
        BigFraction unreducedFraction =
            BigFraction.valueOf(new BigInteger(sBI1),
                                new BigInteger(sBI2),
                                reduce: false);

        return Mono
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Reactive/mono/ex1

End of Applying Key Methods in the Mono Class: Case Study ex1