

Applying Key Operators in the Parallel Flux Class: Case Study ex5 (Part 2)

Douglas C. Schmidt

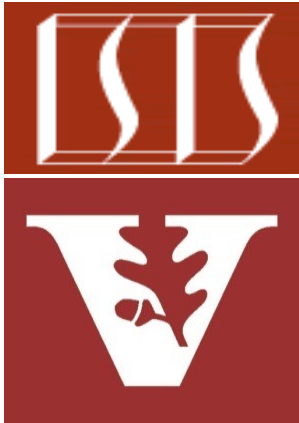
d.schmidt@vanderbilt.edu

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

- Part 2 of case study ex5 shows how to apply Project Reactor features to download & store images from remote web servers by showcasing Flux operators (e.g., fromIterator(), parallel(), & collect()), ParallelFlux operators (e.g., runOn(), map(), & sequential()), & Mono operators (e.g., doOnSuccess() & then()), as well as the Schedulers.boundedElastic() thread pool

```
return Flux
    .fromIterable(getUrlList)

    .parallel()

    .runOn
        (Schedulers
            .boundedElastic())

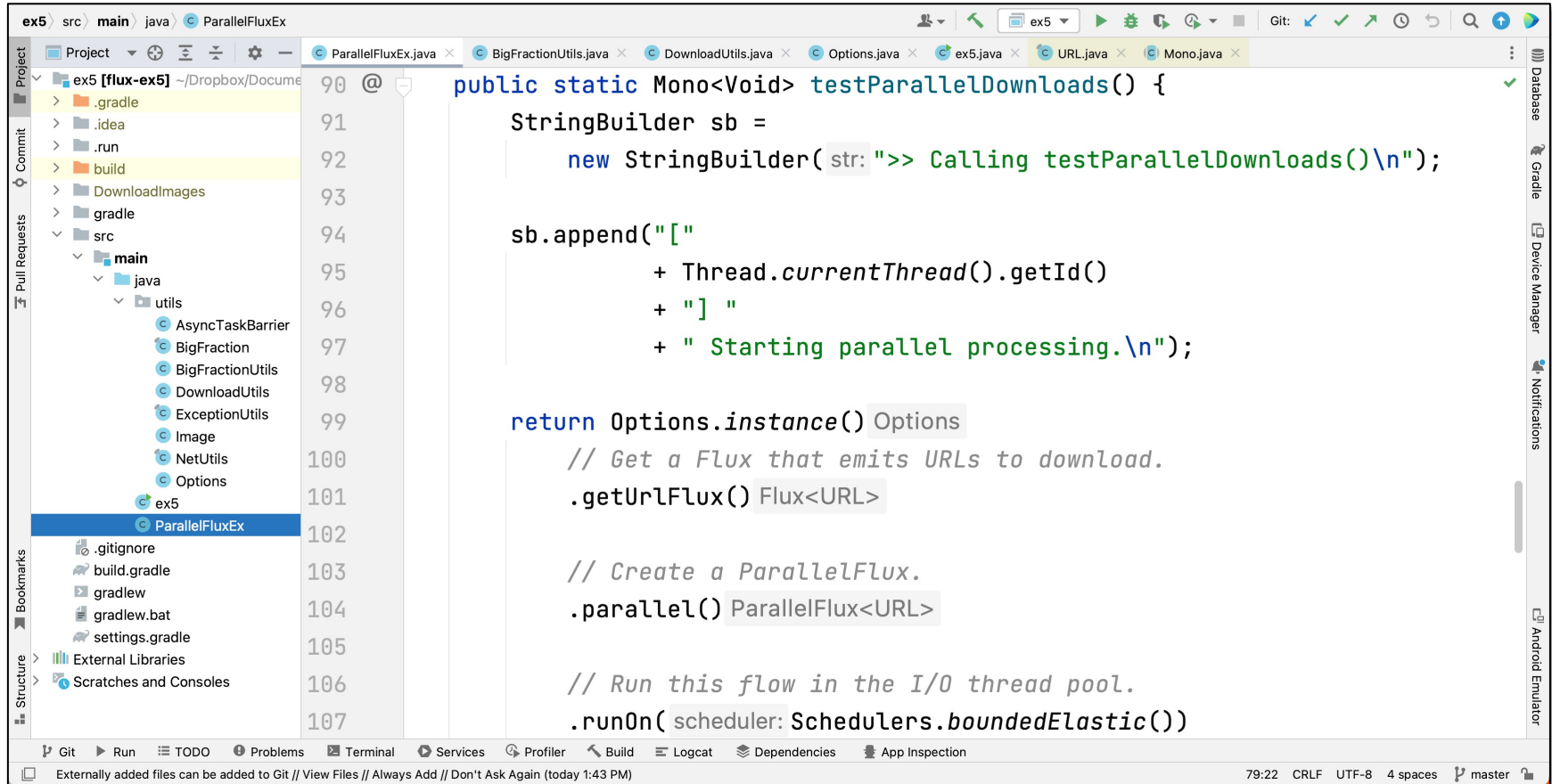
    .map(downloadAndStoreImage)

    .sequential()

    .collect(toList())
```

Applying Key Operators in the ParallelFlux Class to ex5

Applying Key Operators in the ParallelFlux Class to ex5



```
90 @ public static Mono<Void> testParallelDownloads() {
91     StringBuilder sb =
92         new StringBuilder(str: ">> Calling testParallelDownloads()\n");
93
94     sb.append("[ "
95         + Thread.currentThread().getId()
96         + " ] "
97         + " Starting parallel processing.\n");
98
99     return Options.instance() Options
100         // Get a Flux that emits URLs to download.
101         .getUriFlux() Flux<URL>
102
103         // Create a ParallelFlux.
104         .parallel() ParallelFlux<URL>
105
106         // Run this flow in the I/O thread pool.
107         .runOn( scheduler: Schedulers.boundedElastic())
```

See github.com/douglasraigschmidt/LiveLessons/tree/master/Reactive/flux/ex5

End of Applying Key Operators in the ParallelFlux Class: Case Study ex5 (Part 2)