

Enhancing Java Completable Futures: Framework Extensibility (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**



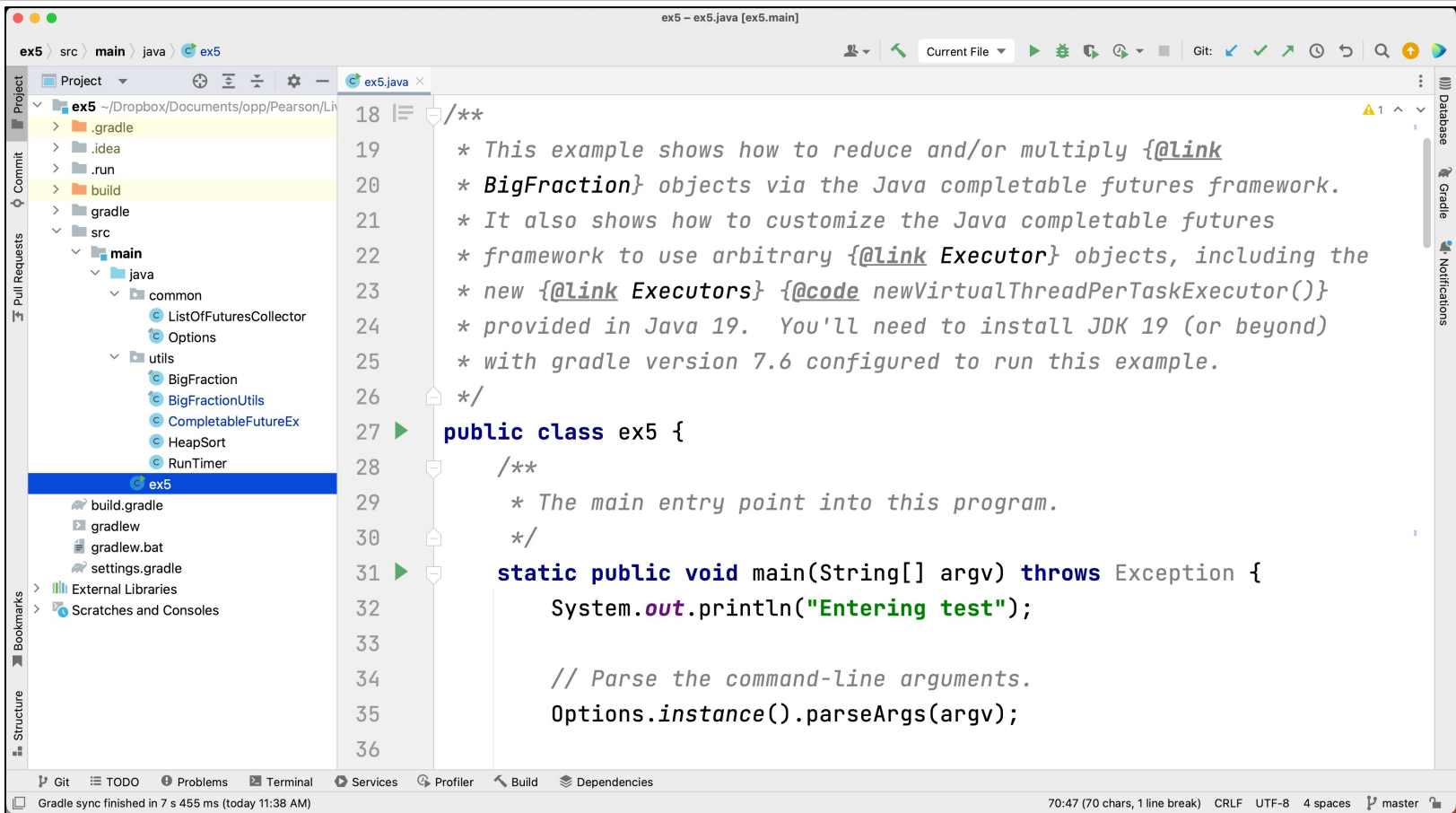
Learning Objectives in this Part of the Lesson

- Evaluate the pros of using the Java completable futures framework
- Evaluate the cons of using the Java completable futures framework
- Understand enhancements to the Java completable futures framework
 - Enhanced timeout handling
 - Enhancing extensibility
 - Analyzing case study ex6 that shows how to apply `CompletableFutureEx` to reduce & multiply `BigFraction` objects via Java virtual threads

```
Function<BigFraction,  
        CompletableFuture  
        <BigFraction>>  
reduceAndMultiplyFrac =  
    unreducedFrac ->  
        CompletableFutureEx  
        .supplyAsync( () ->  
            BigFraction.reduce  
                (unreducedFrac) )  
        .thenApplyAsync  
        (reducedFrac ->  
            reducedFrac  
                .multiply  
                (sBigReducedFrac) );
```

Walkthrough of Case Study ex6

Walkthrough of Case Study ex6



```
ex5 - ex5.java [ex5.main]
ex5 src > main > java > ex5
Project ex5 ~/Dropbox/Documents/opp/Pearson/Li
  > .gradle
  > .idea
  > .run
  > build
  > gradle
  > src
    > main
      > java
        > common
          > ListOfFuturesCollector
          > Options
        > utils
          > BigFraction
          > BigFractionUtils
          > CompletableFutureEx
          > HeapSort
          > RunTimer
          > ex5
          > build.gradle
          > gradlew
          > gradlew.bat
          > settings.gradle
        > External Libraries
        > Scratches and Consoles
Structure
Commit
Pull Requests
Database
Gradle
Notifications
18 /**
19  * This example shows how to reduce and/or multiply {@link
20  * BigFraction} objects via the Java completable futures framework.
21  * It also shows how to customize the Java completable futures
22  * framework to use arbitrary {@link Executor} objects, including the
23  * new {@link Executors} {@code newVirtualThreadPerTaskExecutor()}
24  * provided in Java 19. You'll need to install JDK 19 (or beyond)
25  * with gradle version 7.6 configured to run this example.
26  */
27 public class ex5 {
28     /**
29     * The main entry point into this program.
30     */
31     static public void main(String[] argv) throws Exception {
32         System.out.println("Entering test");
33
34         // Parse the command-line arguments.
35         Options.instance().parseArgs(argv);
36
```

Git sync finished in 7 s 455 ms (today 11:38 AM) 70:47 (70 chars, 1 line break) CRLF UTF-8 4 spaces master

See github.com/douglasraigschmidt/LiveLessons/tree/master/Loom/ex6

End of Enhancing Java Completable Futures: Framework Extensibility (Part 2)