Applying Java Structured Concurrency: Case Study ex2

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

- Understand Java's structured concurrency model
- Recognize the classes used to program Java's structure concurrency model, e.g.
  - ThreadPerTaskExecutor
    - Case study ex2 shows how Java Executors is updated with new factory methods that create a (virtual) Thread per task

```
try (ExecutorService
      executor = Executors
     .newVirtualThreadPerTaskExecutor())
{
  return integers
    .stream()

    .map(primeCandidate ->
         checkPrimality
           (primeCandidate,
            executor))

    .toList();
}
```

The tasks in this case study are all CPU-bound

Learning Objectives in this Part of the Lesson

- Understand Java's structured concurrency model
- Recognize the classes used to program Java's structure concurrency model, e.g.
  - ThreadPerTaskExecutor
    - Case study ex2 shows how Java Executors is updated with new factory methods that create a (virtual) Thread per task
    - It also shows how to combine Java Streams with the new Java Executors features

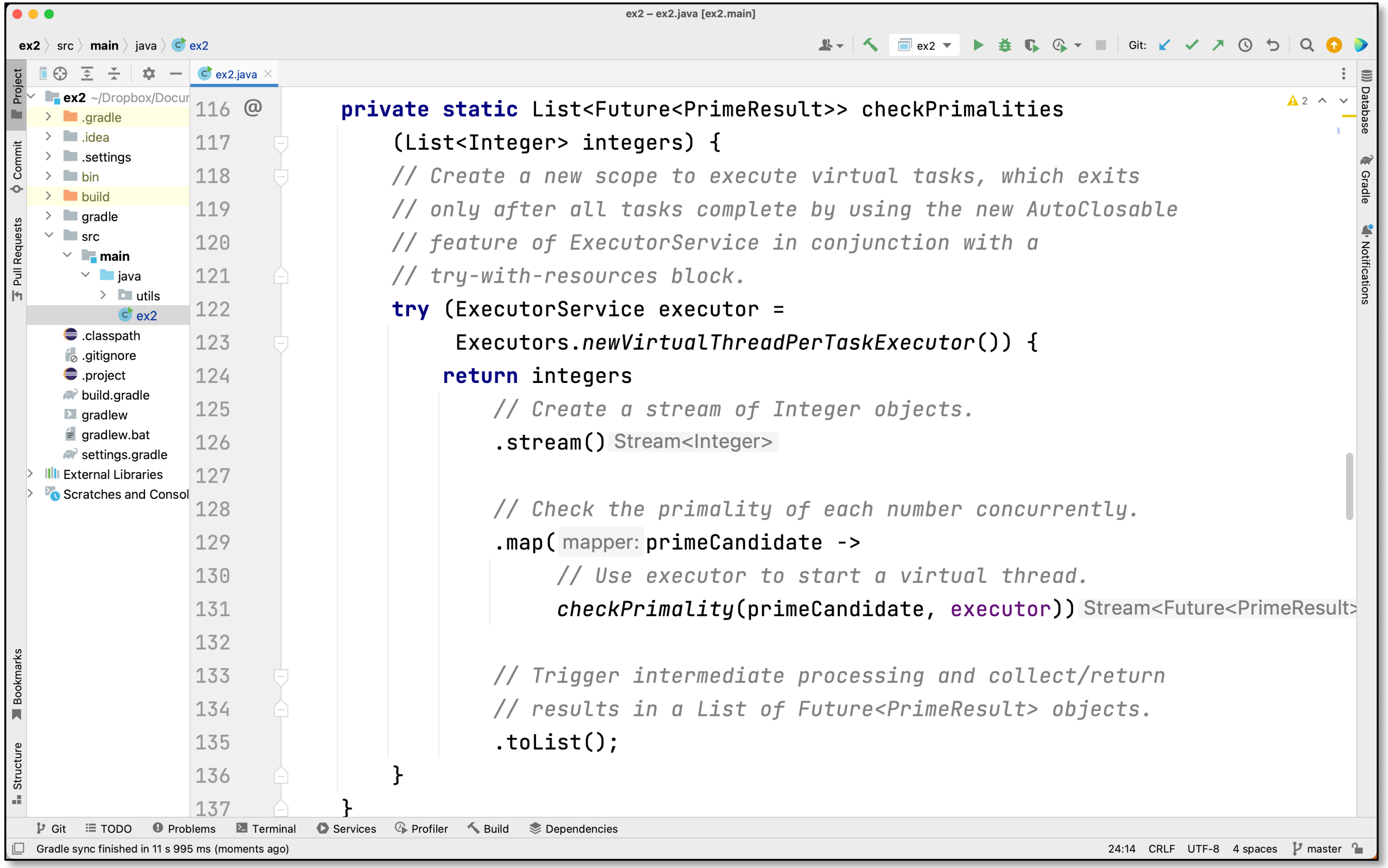
```
try (ExecutorService
      executor = Executors
     .newVirtualThreadPerTaskExecutor())
{
  return integers
     .stream()

     .map(primeCandidate ->
          checkPrimality
            (primeCandidate,
             executor))

     .toList();
}
```

Applying Java Structured Concurrency to Case Study ex2

Applying Java Structured Concurrency to Case Study ex2


ex2 – ex2.java [ex2.main]
ex2 〉 src 〉 main 〉 java 〉 ex2
ex2.java
ex2 ~/Dropbox/Docur
.gradle
.idea
.settings
bin
build
gradle
src
main
java
utils
ex2
.classpath
.gitignore
.project
build.gradle
gradlew
gradlew.bat
settings.gradle
External Libraries
Scratches and Consol
Git TODO Problems Terminal Services Profiler Build Dependencies
Gradle sync finished in 11 s 995 ms (moments ago)
24:14 CRLF UTF-8 4 spaces master

```java
    private static List<Future<PrimeResult>> checkPrimalities
        (List<Integer> integers) {
        // Create a new scope to execute virtual tasks, which exits
        // only after all tasks complete by using the new AutoClosable
        // feature of ExecutorService in conjunction with a
        // try-with-resources block.
        try (ExecutorService executor =
              Executors.newVirtualThreadPerTaskExecutor()) {
            return integers
                // Create a stream of Integer objects.
                .stream() Stream<Integer>

                // Check the primality of each number concurrently.
                .map( mapper: primeCandidate ->
                    // Use executor to start a virtual thread.
                    checkPrimality(primeCandidate, executor)) Stream<Future<PrimeResult>

                // Trigger intermediate processing and collect/return
                // results in a List of Future<PrimeResult> objects.
                .toList();
        }
    }
```

See github.com/douglascraigschmidt/LiveLessons/tree/master/Loom/ex2

End of Applying Java Structured Concurrency: Case Study ex2