

# Applying Java Structured Concurrency:

## Case Study ex2

**Douglas C. Schmidt**

**[d.schmidt@vanderbilt.edu](mailto:d.schmidt@vanderbilt.edu)**

**[www.dre.vanderbilt.edu/~schmidt](http://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

---

- Case study ex2 demos Java 19 structured concurrency features, which enable a main task to split into several concurrent sub-tasks that run concurrently to completion before the main task can complete
- Java 19 enhances `ExecutorService` to support `AutoCloseable` & updates `Executors` w/new static factory methods that support usage in a structured manner

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

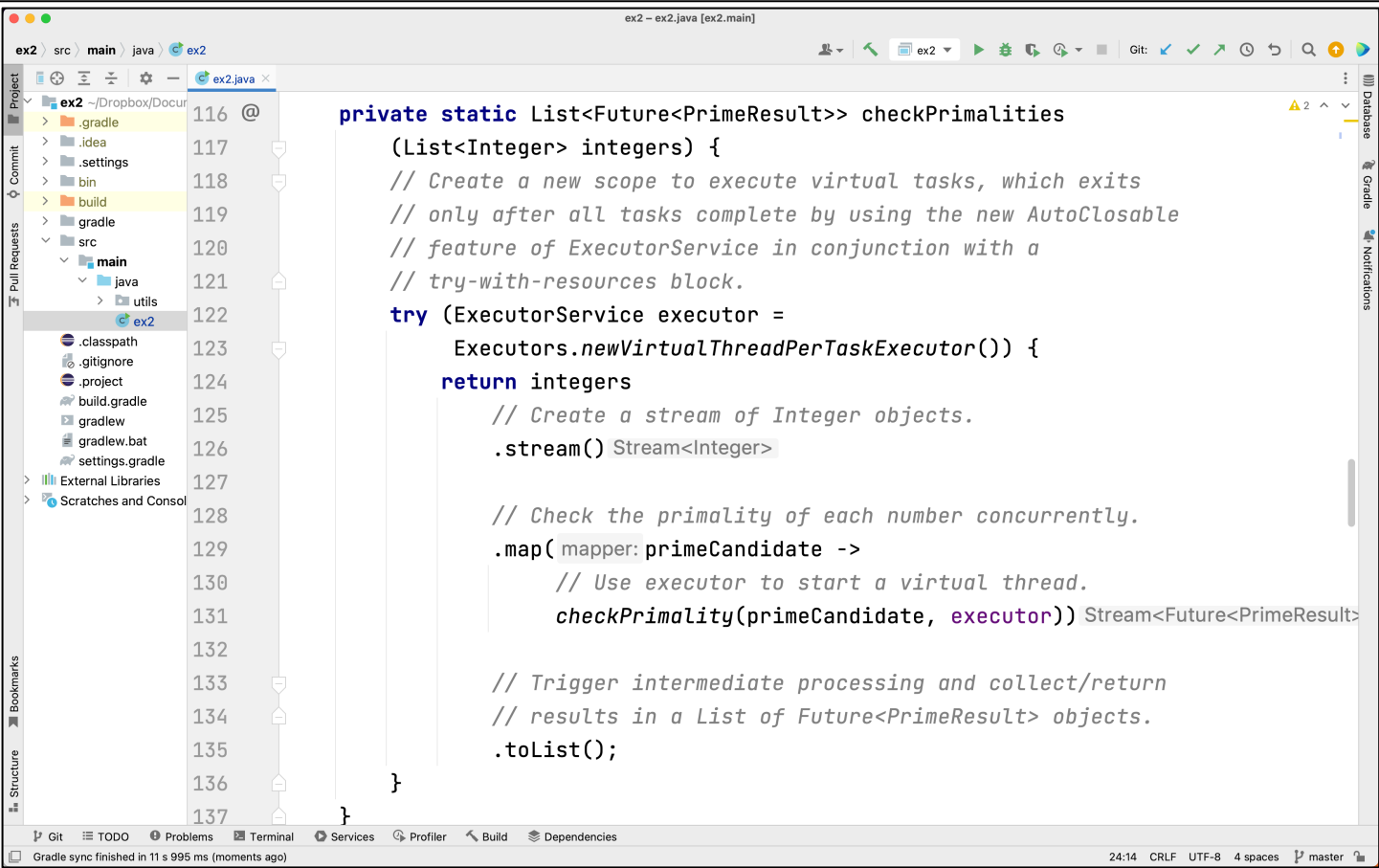
        .map(primeCandidate ->
            checkPrimality
                (primeCandidate,
                 executor))

        .toList();
}
```

---

# Applying Java Structured Concurrency to Case Study ex2

# Applying Java Structured Concurrency to Case Study ex2



The screenshot shows an IDE window titled "ex2 - ex2.java [ex2.main]". The left sidebar displays a project structure for "ex2" located at "~/.Dropbox/Docur". The main editor area shows the following Java code:

```
116 @
117
118
119 // Create a new scope to execute virtual tasks, which exits
120 // only after all tasks complete by using the new AutoClosable
121 // feature of ExecutorService in conjunction with a
122 // try-with-resources block.
123 try (ExecutorService executor =
124     Executors.newVirtualThreadPerTaskExecutor()) {
125     return integers
126         // Create a stream of Integer objects.
127         .stream() Stream<Integer>
128
129         // Check the primality of each number concurrently.
130         .map( mapper: primeCandidate ->
131             // Use executor to start a virtual thread.
132             checkPrimality(primeCandidate, executor)) Stream<Future<PrimeResult>
133
134         // Trigger intermediate processing and collect/return
135         // results in a List of Future<PrimeResult> objects.
136         .toList();
137 }
```

The bottom status bar indicates "Gradle sync finished in 11 s 995 ms (moments ago)" and the current time is 24:14. The encoding is CRLF, UTF-8, with 4 spaces. The master branch is selected.

See [github.com/douglasraigschmidt/LiveLessons/tree/master/Loom/ex2](https://github.com/douglasraigschmidt/LiveLessons/tree/master/Loom/ex2)

---

# End of Applying Java Structured Concurrency: Case Study ex2