

Applying Java Platform Threads & Virtual Threads: Case Study ex1



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

**Institute for Software
Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA**



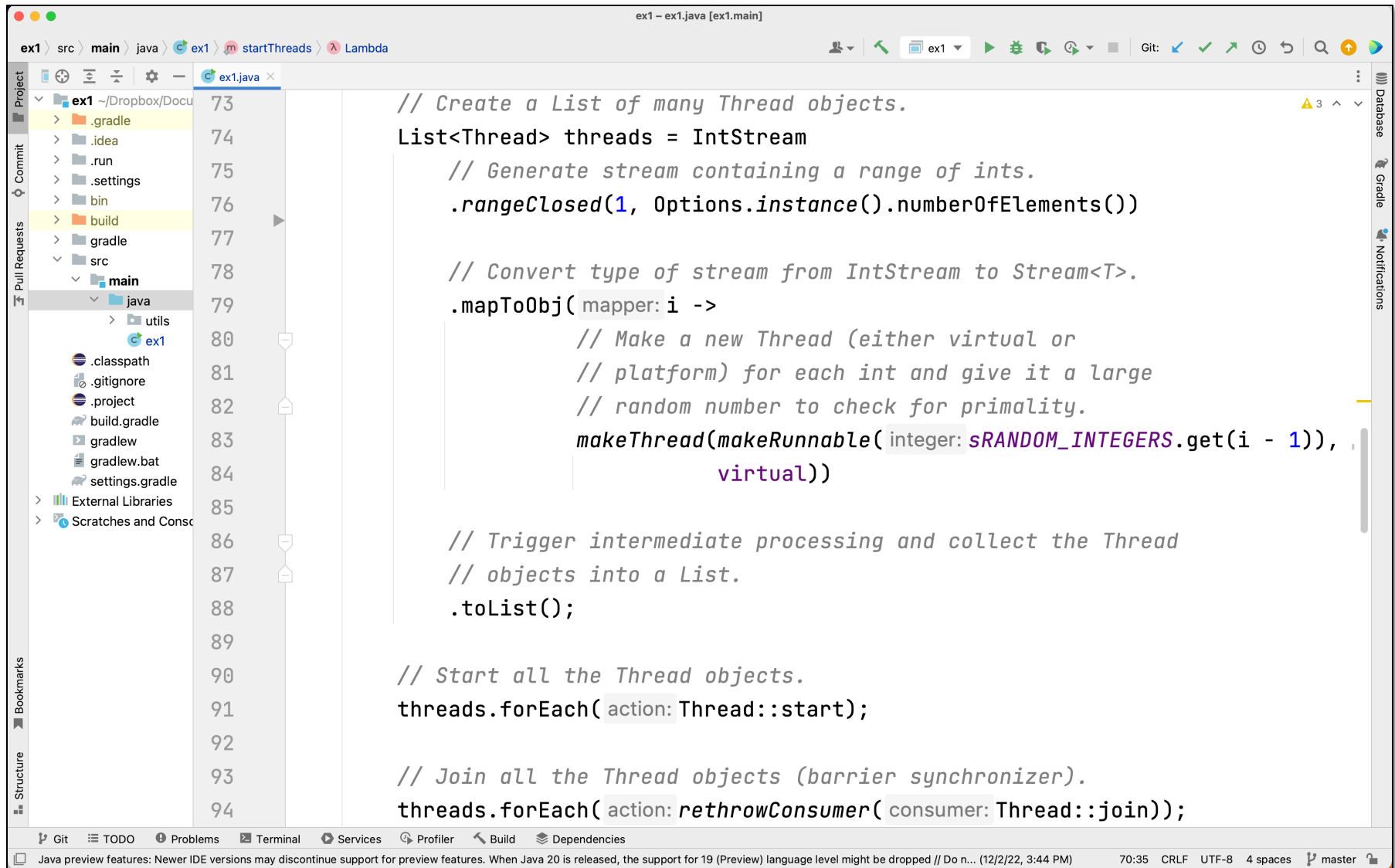
Learning Objectives in this Part of the Lesson

- Understand how Java threads support concurrency
- Learn how our case study app works
- Know alternative ways of giving code to a thread
- Learn how to pass parameters to a Java thread
- Know the differences between Java platform & virtual threads
 - Be aware of how to program Java platform & virtual threads

```
Thread makeThread
    (Runnable runnable,
     boolean virtual) {
    if (virtual)
        return Thread.ofVirtual()
            .unstarted(runnable);
    else
        return Thread.ofPlatform()
            .unstarted(runnable);
    }
```

Applying Java Platform Threads & Virtual Threads

Applying Java Platform Threads & Virtual Threads



The screenshot shows an IDE window titled "ex1 - ex1.java [ex1.main]". The left sidebar displays a project structure with folders like ".gradle", ".idea", ".run", ".settings", "bin", "build", "gradle", "src", "main", "java", and "utils". The main editor area contains the following Java code:

```
// Create a List of many Thread objects.
List<Thread> threads = IntStream
    // Generate stream containing a range of ints.
    .rangeClosed(1, Options.instance().numberOfElements())

    // Convert type of stream from IntStream to Stream<T>.
    .mapToObj( mapper: i ->
        // Make a new Thread (either virtual or
        // platform) for each int and give it a large
        // random number to check for primality.
        makeThread(makeRunnable( integer: sRANDOM_INTEGERS.get(i - 1)),
            virtual))

    // Trigger intermediate processing and collect the Thread
    // objects into a List.
    .toList();

// Start all the Thread objects.
threads.forEach( action: Thread::start);

// Join all the Thread objects (barrier synchronizer).
threads.forEach( action: rethrowConsumer( consumer: Thread::join));
```

The IDE interface includes a top toolbar with icons for navigation and execution, a bottom toolbar with tabs for Git, TODO, Problems, Terminal, Services, Profiler, Build, and Dependencies, and a status bar at the bottom showing "Java preview features: Newer IDE versions may discontinue support for preview features. When Java 20 is released, the support for 19 (Preview) language level might be dropped // Do n... (12/2/22, 3:44 PM) 70:35 CRLF UTF-8 4 spaces P master".

See github.com/douglas-craig-schmidt/LiveLessons/tree/master/Loom/ex1

End of Applying Java Platform Threads & Virtual Threads