Applying Java Functional Programming Features: the ThreadJoinTest Case Study

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 how Java functional features are applied in an "embar-rassingly parallel" program
- Know how to create, start, process, & join Java Thread objects via functional programming features
- Recognize how to use modern Java functional programming features in conjunction with Java Thread methods
  - i.e., concurrently search for a List of phrases in the works of Shakespeare

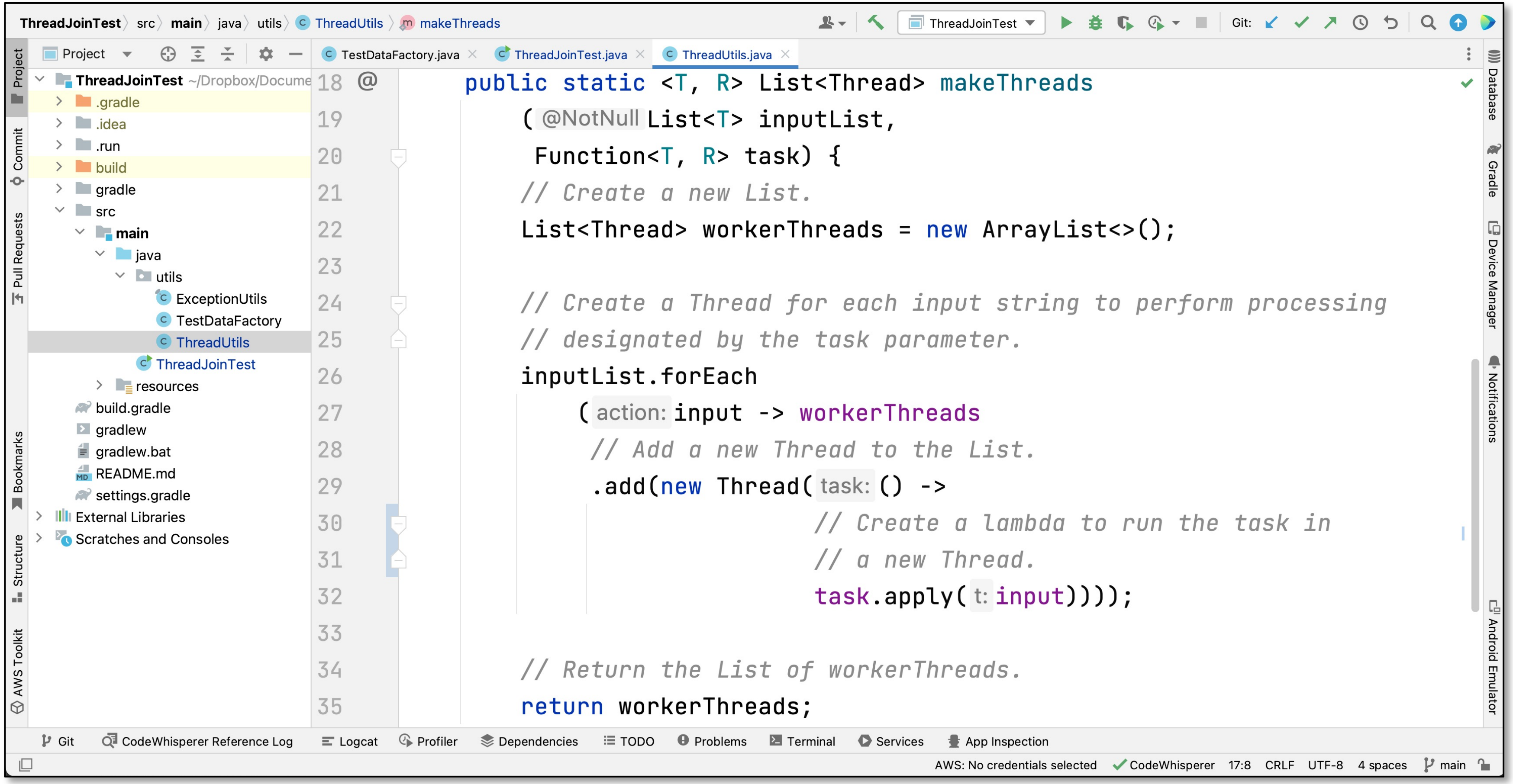
```
<T, R> List<Thread> makeThreads
   (List<T> inputList,
    Function<T, R> task) {
   List<Thread> workerThreads =
     new ArrayList<>();

   inputList.forEach(input ->
     workerThreads
       .add(new Thread
          (() -> task
            .apply(input))));

   return workerThreads;
}
```

Applying Java Function Programming Features & Threads

Applying the Java Functional Programming Features & Threads



ThreadJoinTest 〉 src 〉 main 〉 java 〉 utils 〉 ThreadUtils 〉 makeThreads

TestDataFactory.java | ThreadJoinTest.java | ThreadUtils.java

```java
@ public static <T, R> List<Thread> makeThreads
      ( @NotNull List<T> inputList,
       Function<T, R> task) {
     // Create a new List.
     List<Thread> workerThreads = new ArrayList<>();

     // Create a Thread for each input string to perform processing
     // designated by the task parameter.
     inputList.forEach
         ( action: input -> workerThreads
          // Add a new Thread to the List.
          .add(new Thread( task: () ->
                               // Create a lambda to run the task in
                               // a new Thread.
                               task.apply( t: input)))));

     // Return the List of workerThreads.
     return workerThreads;
```

Project
ThreadJoinTest ~/Dropbox/Docume
.gradle
.idea
.run
build
gradle
src
main
java
utils
ExceptionUtils
TestDataFactory
ThreadUtils
ThreadJoinTest
resources
build.gradle
gradlew
gradlew.bat
README.md
settings.gradle
External Libraries
Scratches and Consoles
Git | CodeWhisperer Reference Log | Logcat | Profiler | Dependencies | TODO | Problems | Terminal | Services | App Inspection
AWS: No credentials selected | CodeWhisperer | 17:8 | CRLF | UTF-8 | 4 spaces | main

See github.com/douglascraigschmidt/ModernJava/tree/main/CS/ThreadJoinTest

End of the Applying Java Functional Programming Features: the ThreadJoin Test Case Study