Applying Java Structured Concurrency: Case Study ex4 (Part 1b)

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 classes used to program Java’s structure concurrency model
- Case study ex4 evaluates the design & performance results of various Java concurrency models

```java
try (var executor = Executors.
    newVirtualThreadPerTaskExecutor())
{
    return urlList
        .stream()
        .map(url -> executor
            .submit(() ->
                downloadImage(url)))
        .toList();
}
```
Learning Objectives in this Part of the Lesson

• Understand Java’s structured concurrency model
• Recognize classes used to program Java’s structure concurrency model
• Case study ex4 evaluates the design & performance results of various Java concurrency models
• Part 1b of this case study focuses on the Java structured concurrency Executors.newVirtualThreadPerTaskExecutor()

```java
try (var executor = Executors.newVirtualThreadPerTaskExecutor()) {
    return urlList
        .stream()
        .map(url -> executor.submit(() -> downloadImage(url)))
        .toList();
}
```
The tasks in this case study are largely I/O-bound

Learning Objectives in this Part of the Lesson

- Understand Java’s structured concurrency model
- Recognize classes used to program Java’s structure concurrency model
- Case study ex4 evaluates the design & performance results of various Java concurrency models
- Part 1b of this case study focuses on the Java structured concurrency model

```java
try (var executor = Executors.newVirtualThreadPerTaskExecutor()) {
    return urlList.stream()
        .map(url -> executor.submit(() ->
            downloadImage(url)))
        .toList();
}
```
Learning Objectives in this Part of the Lesson

- Understand Java’s structured concurrency model
- Recognize classes used to program Java’s structure concurrency model
- Case study ex4 evaluates the design & performance results of various Java concurrency models
  - Part 1b of this case study focuses on the Java structured concurrency model
    - This solution uses the Java sequential streams framework

```java
try (var executor = Executors.newVirtualThreadPerTaskExecutor()) {
    return urlList
        .stream()
        .map(url -> executor
            .submit(() ->
                downloadImage(url)))
        .toList();
}
```
Applying Java Structured Concurrency to Case Study ex4
private static List<Future<Image>> transformImages
  (List<Future<Image>> downloadedImages) {
    // Create a new scope to execute virtual tasks, which exits
    // only after all tasks complete.
    try (var scope = new StructuredTaskScope.ShutdownOnFailure()) {
      // A List of Future<Image> objects that complete when the
      // images have been transformed asynchronously.
      var transformedImages = new ArrayList<Future<Image>>();

      // Iterate through the List of imageFutures.
      for (var imageFuture : downloadedImages) {
        transformedImages
          // Append the transforming images at the end
          // of the List.
          .addAll(c::transformImage(
              executor: scope,
              image: rethrowSupplier
                (function: imageFuture::get)
                .get()));

        rethrowRunnable(t::scope::join);
        // Scope doesn't exit until all concurrent tasks complete.
    }
}
End of Applying Java Structured Concurrency: Case Study ex4 (Part 1b)