Java Parallel ImageStreamGang

Example: Introduction

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

- Recognize the structure & functionality of the ImageStreamGang app

List of URLs to Download

List of Filters to Apply

Learning Objectives in this Part of the Lesson

- Recognize the structure & functionality of the ImageStreamGang app
- It applies several Java parallelism frameworks
Learning Objectives in this Part of the Lesson

- Recognize the structure & functionality of the ImageStreamGang app
- It applies several Java parallelism frameworks

Parallel streams must use fork-join pool framework

See [docs.oracle.com/javase/tutorial/collectionsStreams/parallelism.html](docs.oracle.com/javase/tutorial/collectionsStreams/parallelism.html)
Learning Objectives in this Part of the Lesson

- Recognize the structure & functionality of the ImageStreamGang app
- It applies several Java parallelism frameworks

Parallel Streams

- filter(not(this::urlCached))
- map(this::downloadImage)
- flatMap(this::applyFilters)
- collect(toList())

Completable Futures may use fork-join pool framework

- map(this::checkUrlCachedAsync)
- map(this::downloadImageAsync)
- flatMap(this::applyFiltersAsync)
- collect(toFuture())
- thenAccept(this::log)

See [www.nurkiewicz.com/2013/05/java-8-definitive-guide-to.html](www.nurkiewicz.com/2013/05/java-8-definitive-guide-to.html)
Learning Objectives in this Part of the Lesson

- Recognize the structure & functionality of the ImageStreamGang app
- It applies several Java parallelism frameworks
- Focus is on integrating object-oriented & functional programming paradigms

Patterns are used to emphasize key roles & responsibilities in the app’s design
Overview of the Pattern-Oriented ImageStreamGang App
Overview of the Pattern-Oriented ImageStreamGang App

- This app combines streams & completable futures with the StreamGang framework to download, transform, store, & display images

**Parallel Streams**

1. filter(not(this::urlCached))
2. map(this::downloadImage)
3. flatMap(this::applyFilters)
4. collect(toList())

**Completable Futures**

1. map(this::checkUrlCachedAsync)
2. map(this::downloadImageAsync)
3. flatMap(this::applyFiltersAsync)
4. collect(toFuture())
5. thenAccept(this::log)

Overview of the Pattern-Oriented ImageStreamGang App

- This app combines streams & completable futures with the StreamGang framework to download, transform, store, & display images, e.g.

Prompt user for list of URLs to download
Overview of the Pattern-Oriented ImageStreamGang App

- This app combines streams & completable futures with the StreamGang framework to download, transform, store, & display images, e.g.

List of URLs to Download

- User supplies the list of URLs to download
Overview of the Pattern-Oriented ImageStreamGang App

- This app combines streams & composable futures with the StreamGang framework to download, transform, store, & display images, e.g.

List of URLs to Download

Download images via one or more threads
Overview of the Pattern-Oriented ImageStreamGang App

- This app combines streams & completable futures with the StreamGang framework to download, transform, store, & display images, e.g.

List of URLs to Download

List of Filters to Apply

Apply filters to transform downloaded images
Overview of the Pattern-Oriented ImageStreamGang App

- This app combines streams & completable futures with the StreamGang framework to download, transform, store, & display images, e.g.
Overview of the Pattern-Oriented ImageStreamGang App

- The ImageStreamGang app applies a range of modern Java features

Overview of the Pattern-Oriented ImageStreamGang App

- The ImageStreamGang app applies a range of modern Java features, e.g.
- Sequential & parallel streams

```java
List<Image> filteredImages =
    getInput()
    .parallelStream()
    .filter(not(this::urlCached))
    .map(this::downloadImage)
    .flatMap(this::applyFilters)
    .collect(toList());
```

We’ll cover parallel streams now
Overview of the Pattern-Oriented ImageStreamGang App

- The ImageStreamGang app applies a range of modern Java features, e.g.
  - Sequential & parallel streams
  - Completable futures

```java
getInput()
  .stream()
  .map(this::checkUrlCachedAsync)
  .map(this::downloadImageAsync)
  .flatMap(this::applyFiltersAsync)
  .collect(toFuture())
  .thenAccept(stream ->
    log(stream.flatMap(Optional :: stream), urls.size()))
.join();
```

We cover completable futures later
Overview of the Pattern-Oriented ImageStreamGang App

• The ImageStreamGang app applies a range of modern Java features, e.g.
  • Sequential & parallel streams
  • Completable futures
  • Lambda expressions & method references

Runnable mCompletionHook =
    () -> MainActivity.this.runOnUiThread(this::goToResultActivity);
Overview of the Pattern-Oriented ImageStreamGang App

- The ImageStreamGang app applies a range of modern Java features, e.g.
  - Sequential & parallel streams
  - Completable futures
  - Lambda expressions & method references

```java
Runnable mCompletionHook = () -> MainActivity.this.runOnUiThread(
    this::goToResultActivity);
```

versus

```java
Runnable mCompletionHook = new Runnable() {
    public void run() {
        MainActivity.this.runOnUiThread(
            new Runnable() { public void run() {
                goToResultActivity(); }
            });
    }
};
```
Overview of Patterns Applied in the ImageStreamGang App
Overview of Patterns Applied in the ImageStreamGang App

• “Gang-of-Four” & POSA patterns are applied to enhance its framework-based architecture

See en.wikipedia.org/wiki/Design_Patterns & www.dre.vanderbilt.edu/~schmidt/POSA
Some patterns are essential to its design
Overview of Patterns Applied in the ImageStreamGang App

• Some patterns are essential to its design
• **Pipes and Filters**
  • Divide application’s tasks into several self-contained data processing steps & connect these steps via intermediate data buffers to a data processing pipeline

Overview of Patterns Applied in the ImageStreamGang App

- Some patterns are essential to its design
- **Future**
  - Provides a proxy to a client when it invokes a service to keep track of the state of the service’s concurrent computation & only returns a value to the client when the computation completes

See [en.wikipedia.org/wiki/Futures_and_promises](en.wikipedia.org/wiki/Futures_and_promises)
Overview of Patterns Applied in the ImageStreamGang App

- Some patterns are essential to its design
  - **Resource Pool**
  - Prevents expensive acquisition & release of resources by recycling resources no longer needed

See kircher-schwanninger.de/michael/publications/Pooling.pdf
Some patterns are essential to its design

**Template Method**

- Defines the overall structure of a method, while allowing subclasses to refine, or redefine, certain steps

See [en.wikipedia.org/wiki/Template_method_pattern](en.wikipedia.org/wiki/Template_method_pattern)
Some patterns are essential to its design

**Factory Method**

- Encapsulate the concrete details of object creation inside a factory method, rather than letting clients create the object themselves

See [en.wikipedia.org/wiki/Factory_method_pattern](en.wikipedia.org/wiki/Factory_method_pattern)
Overview of Patterns Applied in the ImageStreamGang App

- Some patterns are essential to its design
- **Decorator**
  - Allows behavior to be added to an individual object, dynamically, without affecting the behavior of other objects from the same class

Overview of Patterns Applied in the ImageStreamGang App

• Other patterns are also applied
  
• **Singleton**
  
• Ensure a class has only one instance & provide a global point of access to it

---

**Singleton**

<table>
<thead>
<tr>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>static Instance()</td>
</tr>
<tr>
<td>SingletonOperation()</td>
</tr>
<tr>
<td>GetSingletonData()</td>
</tr>
<tr>
<td>static uniqueInstance singletonData</td>
</tr>
</tbody>
</table>

return uniqueInstance

---

See [en.wikipedia.org/wiki/Singleton_pattern](en.wikipedia.org/wiki/Singleton_pattern)
Overview of Patterns Applied in the ImageStreamGang App

- Other patterns are also applied
- **Command Processor**
  - Packages a piece of application functionality—as well as its parameterization in an object—to make it usable in another context, such as later in time or in a different thread

See [www.dre.vanderbilt.edu/~schmidt/CommandProcessor.pdf](http://www.dre.vanderbilt.edu/~schmidt/CommandProcessor.pdf)
Strategy for Understanding the ImageStreamGang App
Strategy for Understanding the ImageStreamGang App

- This app is complicated & contains many classes

[Diagram showing class relationships]
Strategy for Understanding the ImageStreamGang App

- This app is complicated & contains many classes
- We therefore analyze it from various perspectives

Including pattern-oriented design, data flows, & detailed code walkthroughs
Strategy for Understanding the ImageStreamGang App

• This app is complicated & contains many classes
  • We therefore analyze it from various perspectives

• Watch this entire lesson carefully to understand how it all works
Strategy for Understanding the ImageStreamGang App

- This app is complicated & contains many classes
  - We therefore analyze it from various perspectives
  - Watch this entire lesson carefully to understand how it all works
- Visualize the data flow in a parallel stream

```plaintext
<table>
<thead>
<tr>
<th>List</th>
<th>[URL]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream</td>
<td>[URL]</td>
</tr>
<tr>
<td>Stream</td>
<td>[URL]</td>
</tr>
<tr>
<td>Stream</td>
<td>[Image]</td>
</tr>
<tr>
<td>Stream</td>
<td>[Image]</td>
</tr>
</tbody>
</table>
```

```
parallelStream()
filter(not(this::urlCached))
map(this::downloadImage)
flatMap(this::applyFilters)
```
Strategy for Understanding the ImageStreamGang App

- This app is complicated & contains many classes
  - We therefore analyze it from various perspectives
  - Watch this entire lesson carefully to understand how it all works
  - Visualize the data flow in a parallel stream
  - Run/read the code to see how it all works

End of Java Parallel ImageStreamGang Example: Introduction