Java 8 Parallel ImageStreamGang Example (Part 1)

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

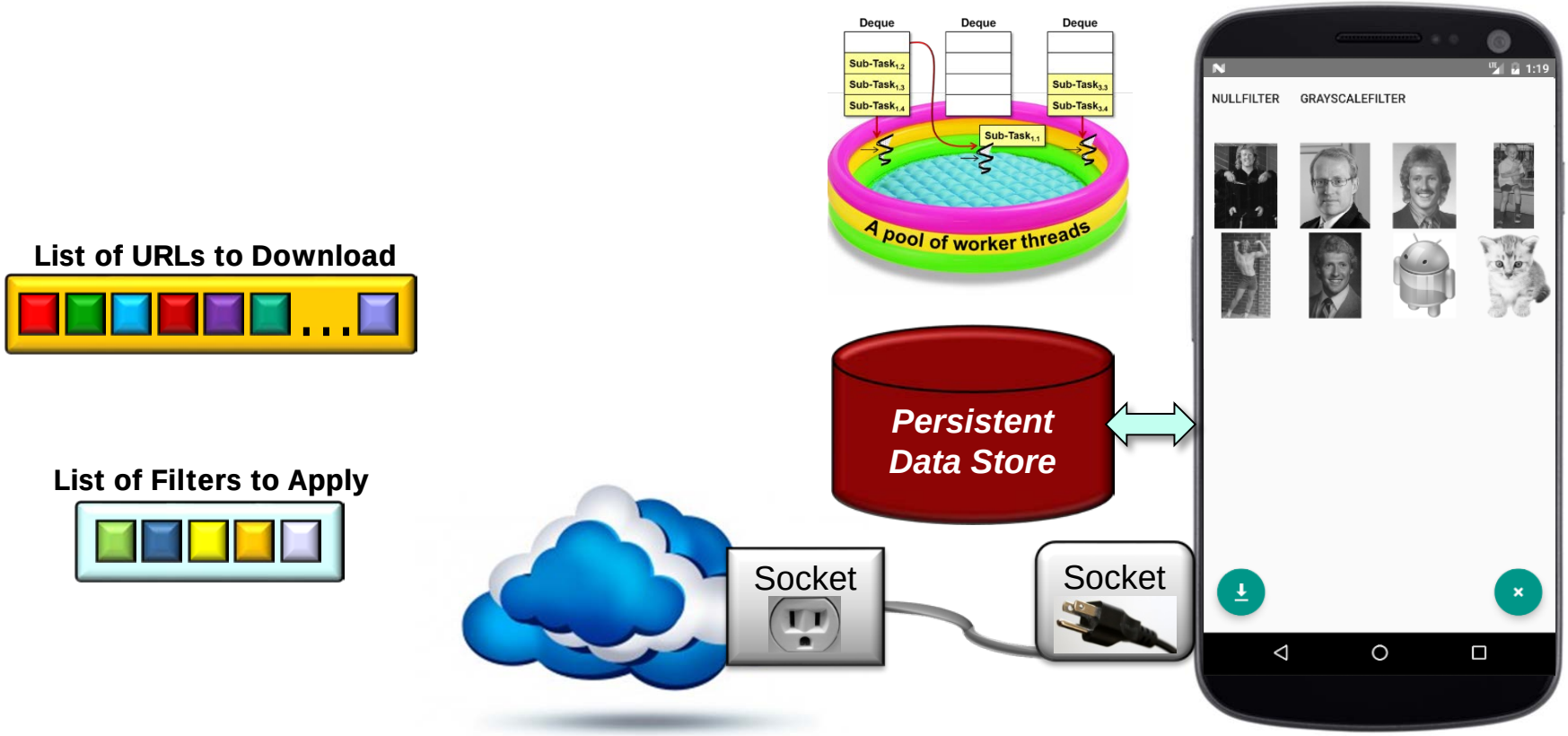
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

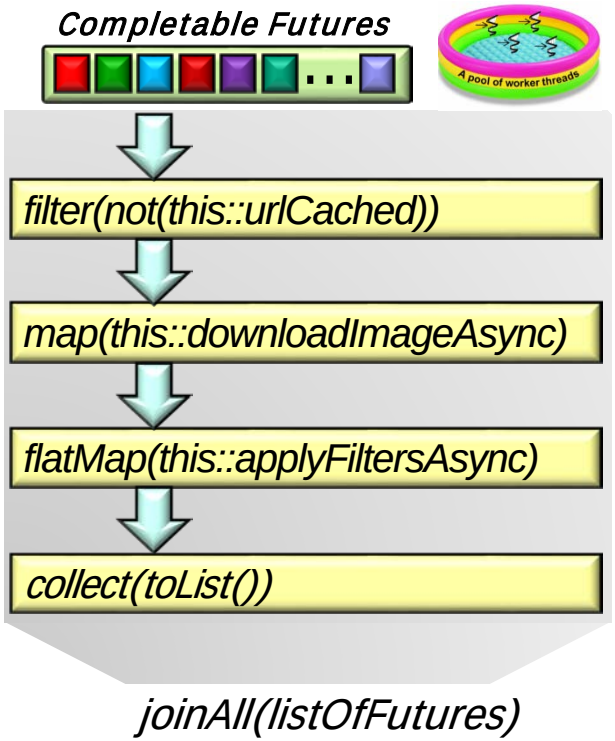
- Recognize the structure & functionality of the ImageStreamGang app

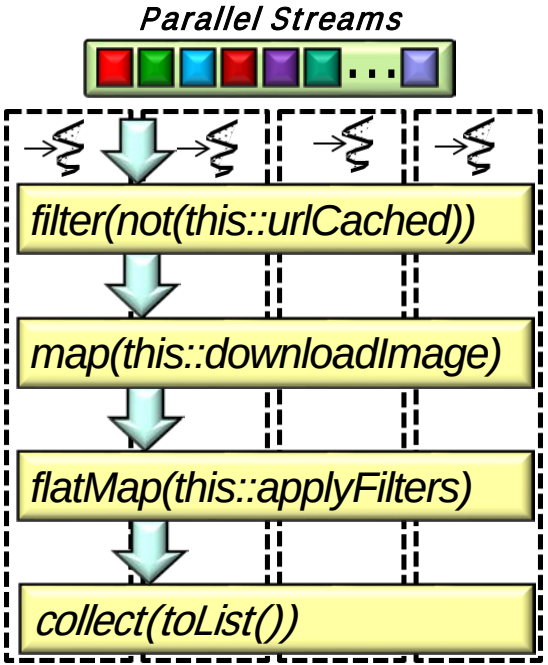


See github.com/douglascraigschmidt/LiveLessons/tree/master/ImageStreamGang

Learning Objectives in this Part of the Lesson

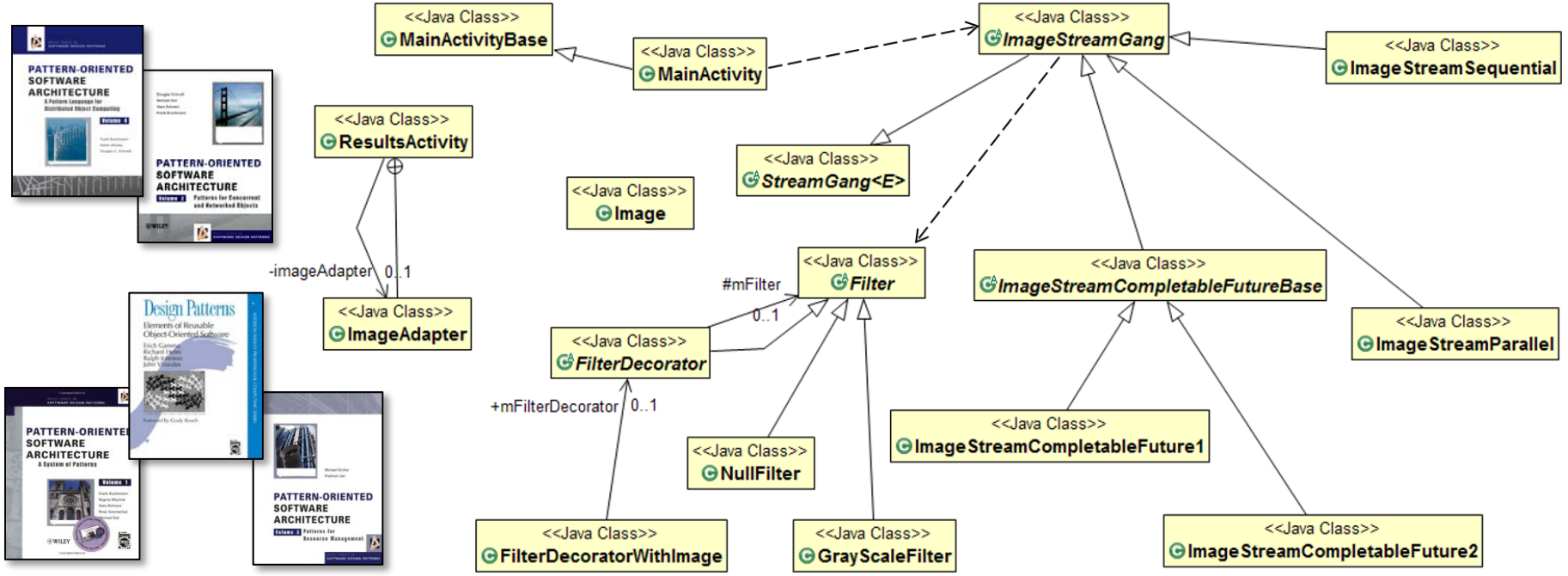
- Recognize the structure & functionality of the ImageStreamGang app
  - It applies several Java 8 concurrency & parallelism frameworks





Learning Objectives in this Part of the Lesson

- Recognize the structure & functionality of the ImageStreamGang app
  - It applies several Java 8 concurrency & parallelism frameworks
  - Focus is on integrating object-oriented & functional programming paradigms

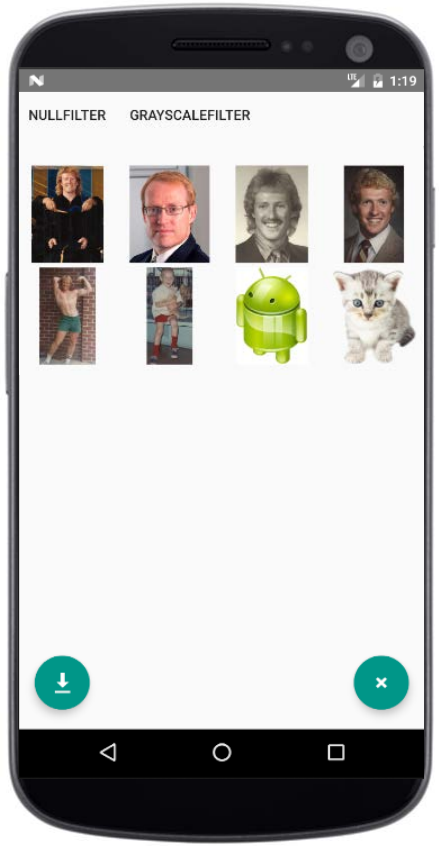


Patterns are used to emphasize key roles & responsibilities in the app's design

Overview of the Pattern-Oriented ImageStreamGang App

Overview of the Pattern-Oriented ImageStreamGang App

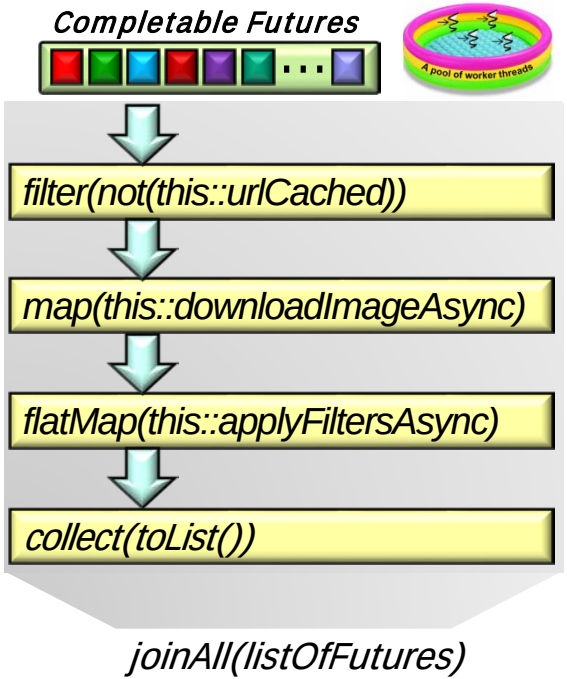
- The UI for the ImageStreamGang app is implemented with Java 8 features

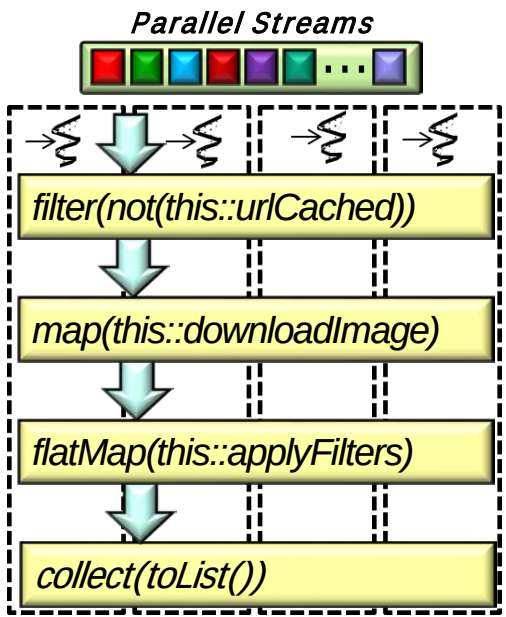


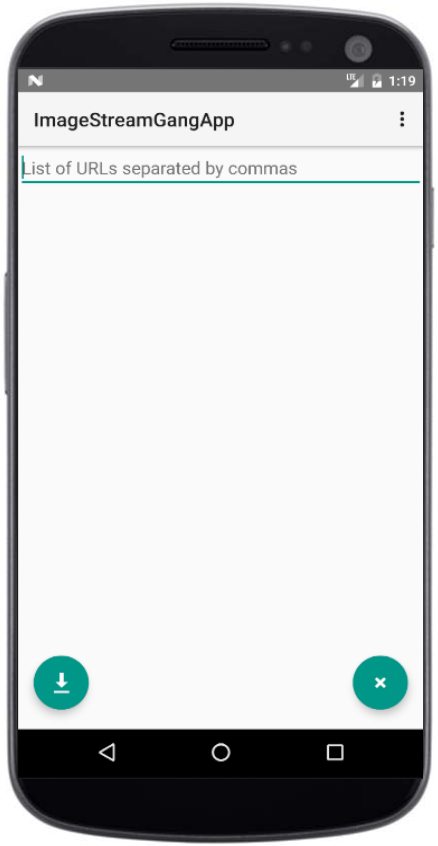
See github.com/douglascraigschmidt/LiveLessons/tree/master/ImageStreamGang

Overview of the Pattern-Oriented ImageStreamGang App

- This app shows how the StreamGang framework can be combined with Java 8 streams & completable futures to download, filter, store, & display images

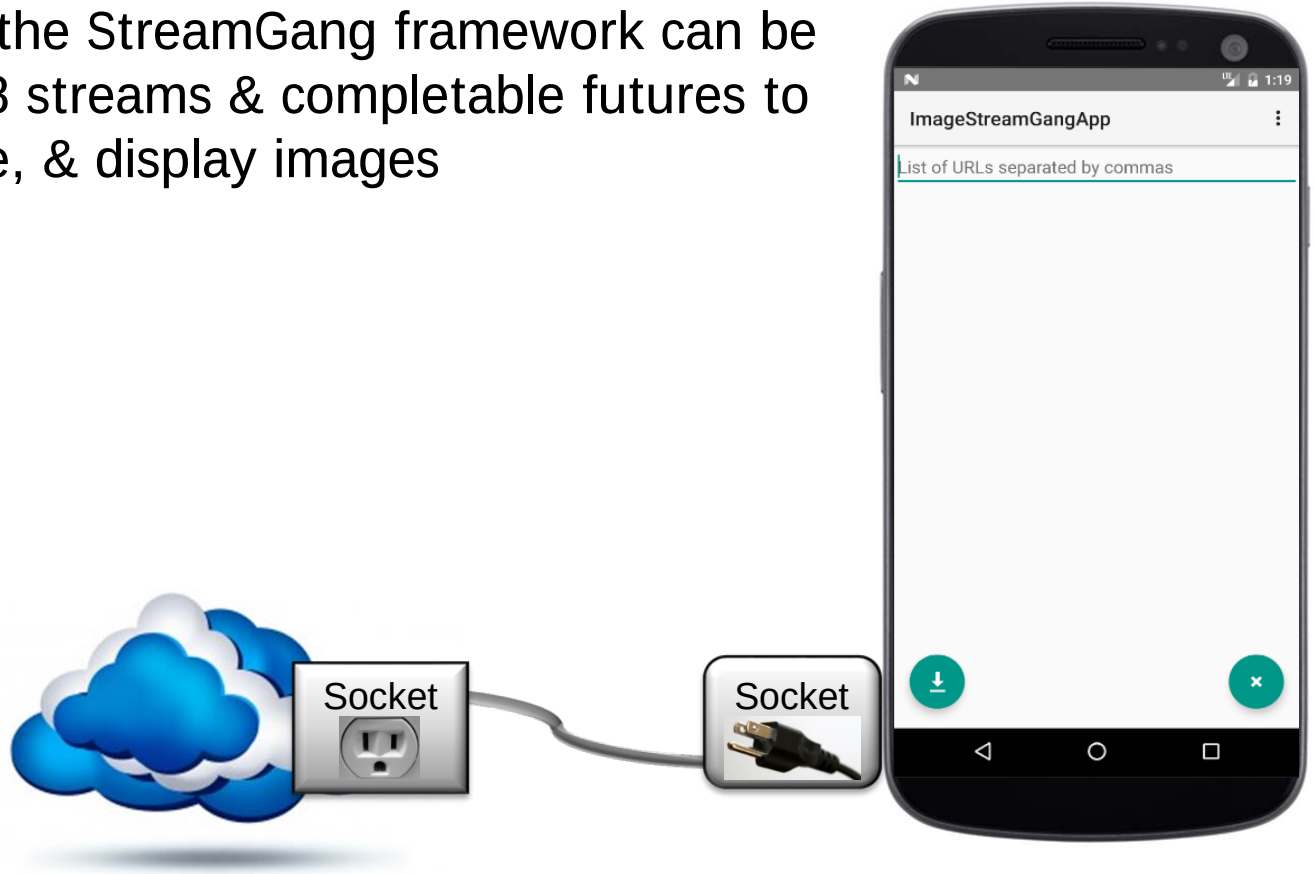






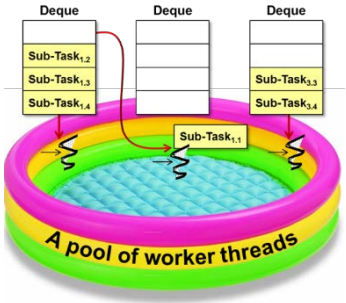
Overview of the Pattern-Oriented ImageStreamGang App

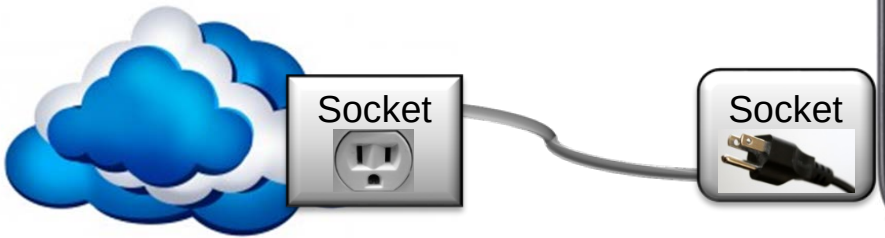
- This app shows how the StreamGang framework can be combined with Java 8 streams & completable futures to download, filter, store, & display images

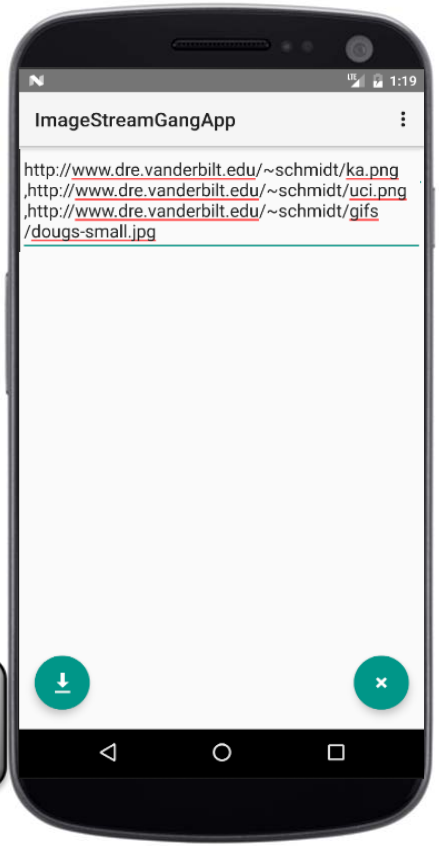


Overview of the Pattern-Oriented ImageStreamGang App

- This app shows how the StreamGang framework can be combined with Java 8 streams & completable futures to download, filter, store, & display images, e.g.



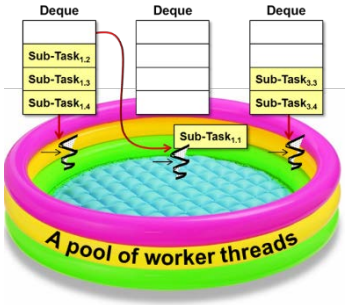


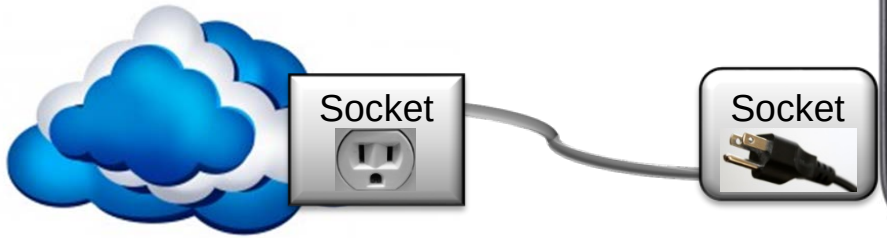


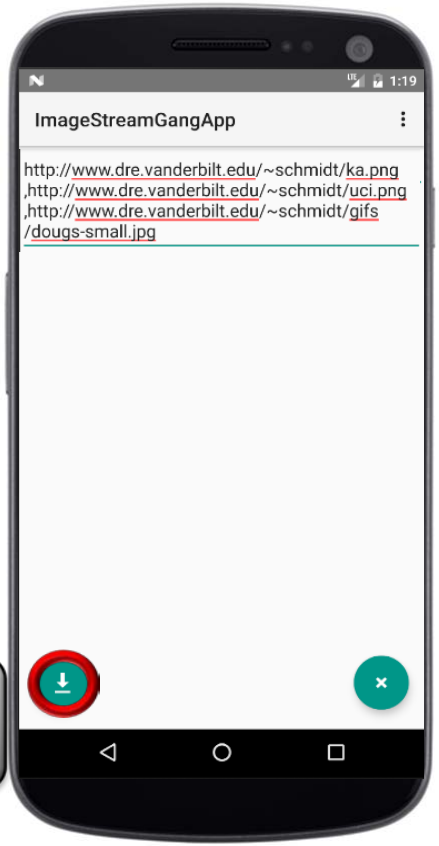
Prompt user for list of URLs to download

Overview of the Pattern-Oriented ImageStreamGang App

- This app shows how the StreamGang framework can be combined with Java 8 streams & completable futures to download, filter, store, & display images, e.g.



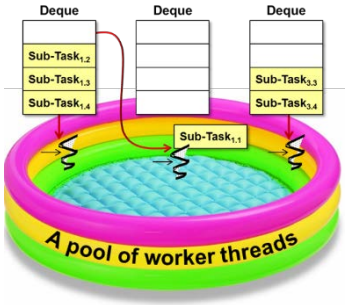


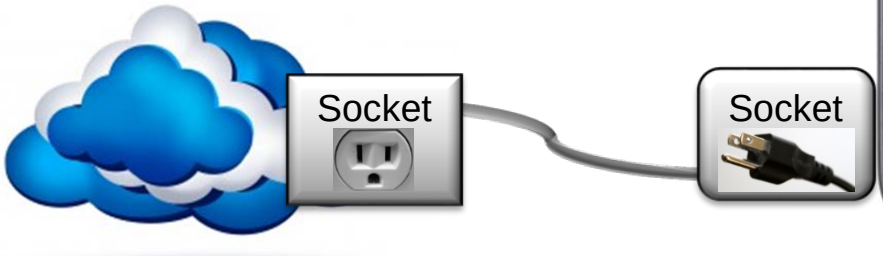


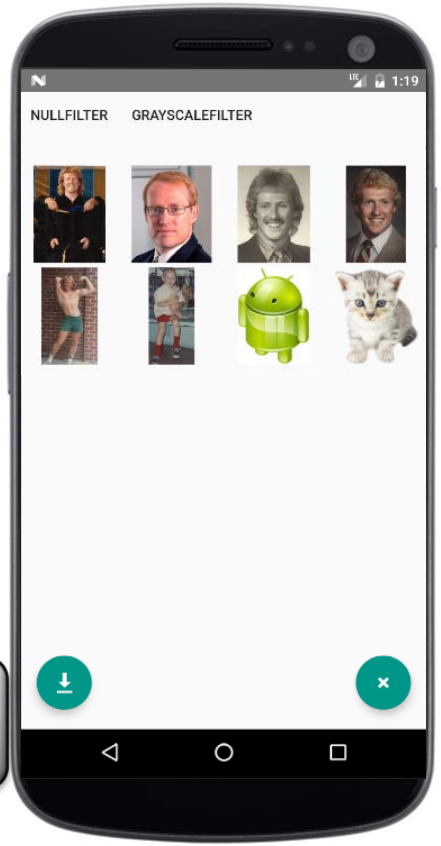
Download images via one or more threads

Overview of the Pattern-Oriented ImageStreamGang App

- This app shows how the StreamGang framework can be combined with Java 8 streams & completable futures to download, filter, store, & display images, e.g.



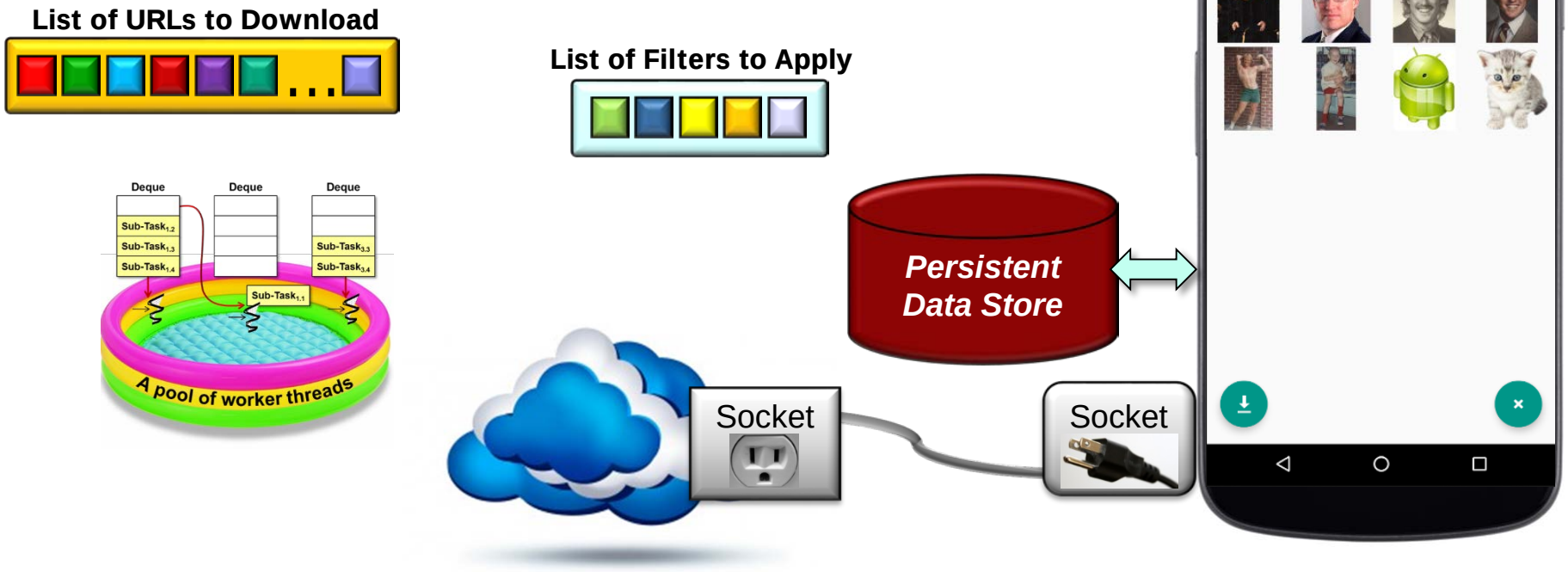




Apply filters to transform downloaded images

Overview of the Pattern-Oriented ImageStreamGang App

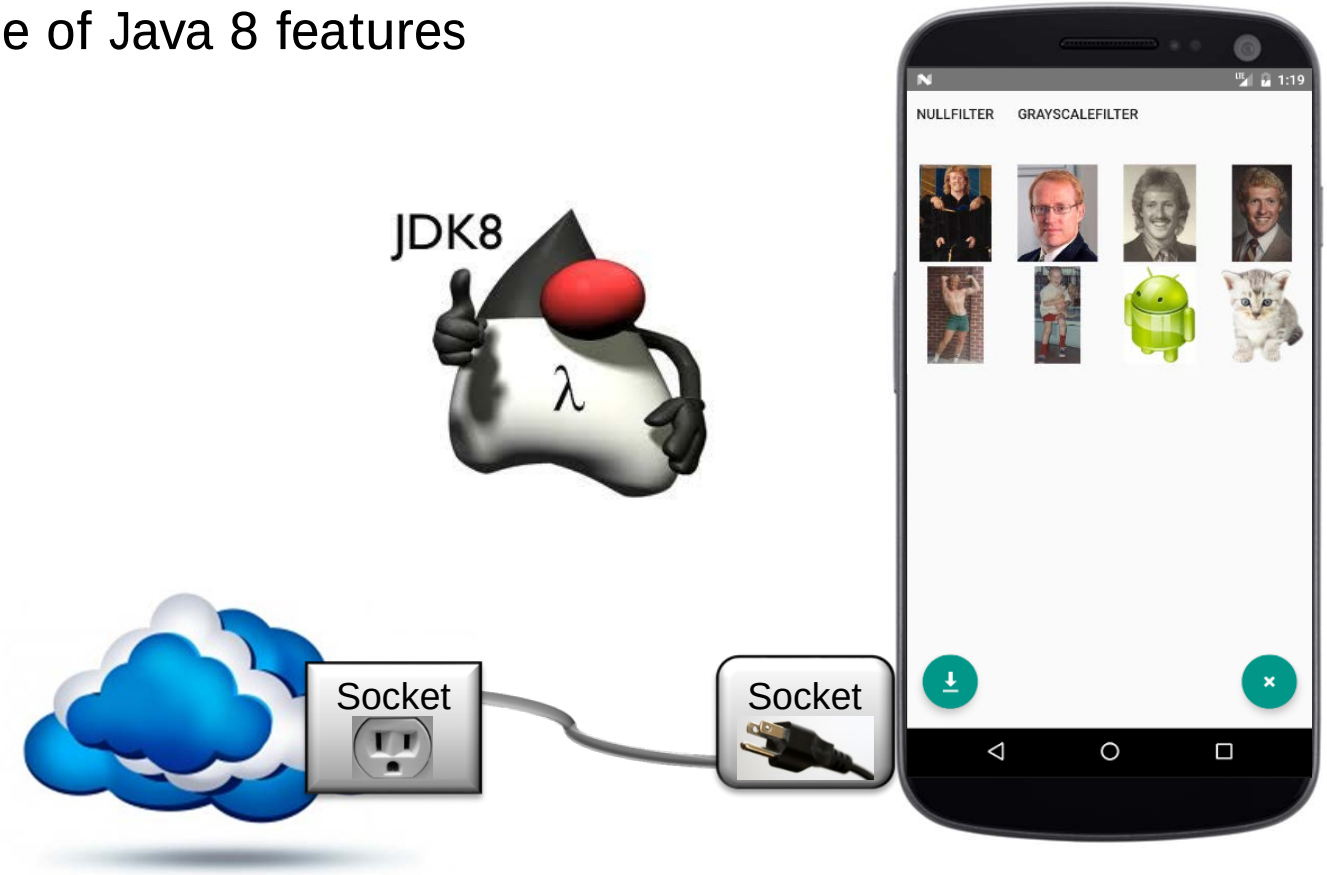
- This app shows how the StreamGang framework can be combined with Java 8 streams & completable futures to download, filter, store, & display images, e.g.



Output filtered images to persistent storage

Overview of the Pattern-Oriented ImageStreamGang App

- This app uses a range of Java 8 features



Overview of the Pattern-Oriented ImageStreamGang App

- This app uses a range of Java 8 features, e.g.
  - Sequential & parallel streams

```
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

- This app uses a range of Java 8 features, e.g.
  - Sequential & parallel streams
  - Completable futures

```
List<CompletableFuture<List<Image>>>
  listOfFutures = getInput()
    .stream()
    .filter(not(this::urlCached))
    .map(this::downloadImageAsync)
    .flatMap(applyFiltersAsync)
    .collect(toList());

CompletableFuture<List<List<Image>>>
    allImagesDone =
      StreamsUtils.joinAll(listOfFutures);
```

We'll cover completable futures later

Overview of the Pattern-Oriented ImageStreamGang App

- This app uses a range of Java 8 features, e.g.
  - Sequential & parallel streams
  - Completable futures
  - Lambda expressions & method references

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

We covered these foundational Java 8 features earlier

Overview of the Pattern-Oriented ImageStreamGang App

- This app uses a range of Java 8 features, e.g.
  - Sequential & parallel streams
  - Completable futures
  - Lambda expressions & method references

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

versus

```
Runnable mCompletionHook = new Runnable() {
    public void run() {
    MainActivity.this.runOnUiThread
     (new Runnable() { public void run()
       { goToResultActivity(); } }); }};
```

We covered these foundational Java 8 features earlier

Overview of the Pattern-Oriented 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

Overview of the Pattern-Oriented ImageStreamGang App

- "Gang-of-Four" & POSA patterns are applied to enhance its framework-based architecture
  - Patterns most essential to its design
    - *Pipes and Filters & Future*

See www.hillside.net/plop/2011/papers/B-10-Hanmer.pdf & en.wikipedia.org/wiki/Futures_and_promises

Overview of the Pattern-Oriented ImageStreamGang App

- "Gang-of-Four" & POSA patterns are applied to enhance its framework-based architecture
  - Patterns most essential to its design
    - *Pipes and Filters & Future*
    - *Pooling*

See kircher-schwanninger.de/michael/publications/Pooling.pdf

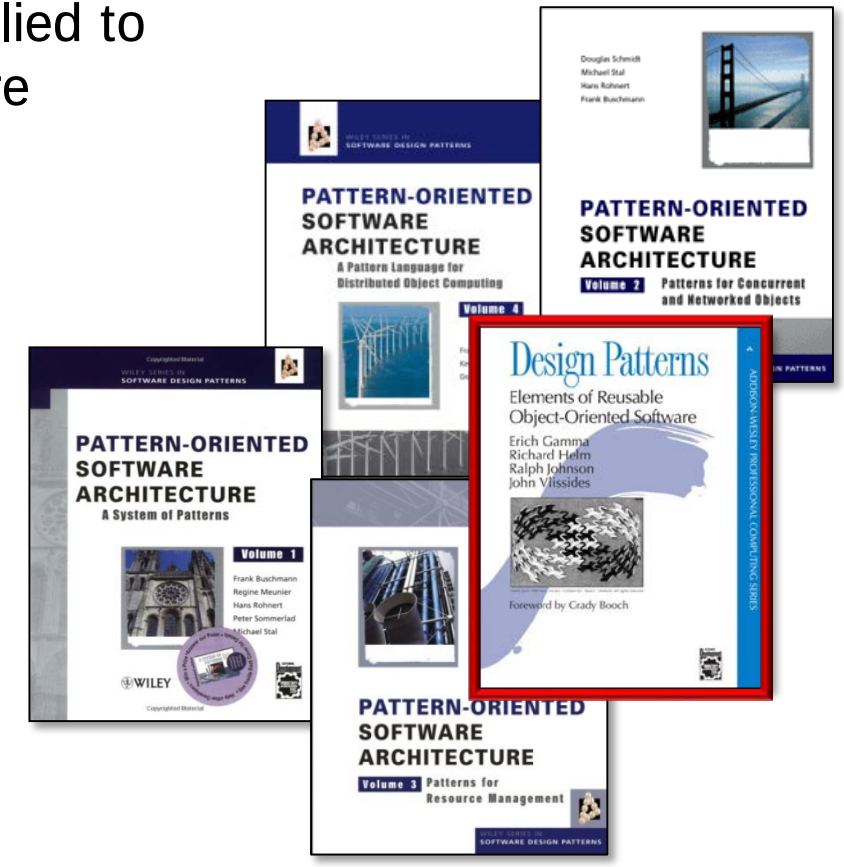
Overview of the Pattern-Oriented ImageStreamGang App

- "Gang-of-Four" & POSA patterns are applied to enhance its framework-based architecture
  - Patterns most essential to its design
    - *Pipes and Filters & Future*
    - *Pooling*
    - *Template Method*

See en.wikipedia.org/wiki/Template_method_pattern

Overview of the Pattern-Oriented ImageStreamGang App

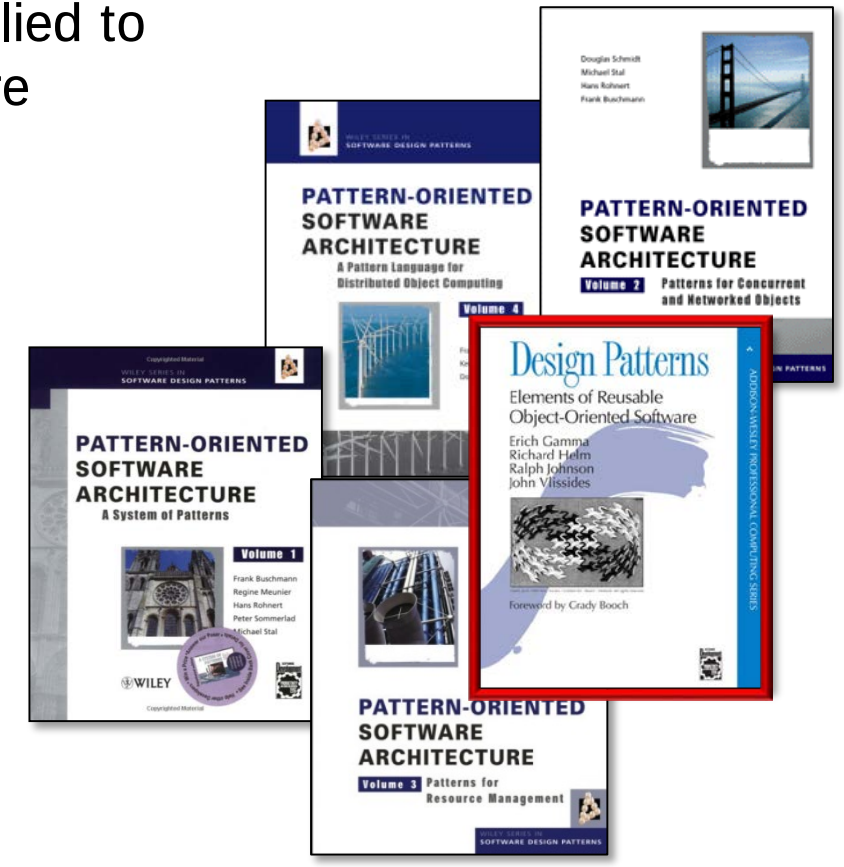
- "Gang-of-Four" & POSA patterns are applied to enhance its framework-based architecture
  - Patterns most essential to its design
    - *Pipes and Filters & Future*
    - *Pooling*
    - *Template Method*
    - *Decorator & Factory Method*



See en.wikipedia.org/wiki/Decorator_pattern & en.wikipedia.org/wiki/Factory_method_pattern

Overview of the Pattern-Oriented ImageStreamGang App

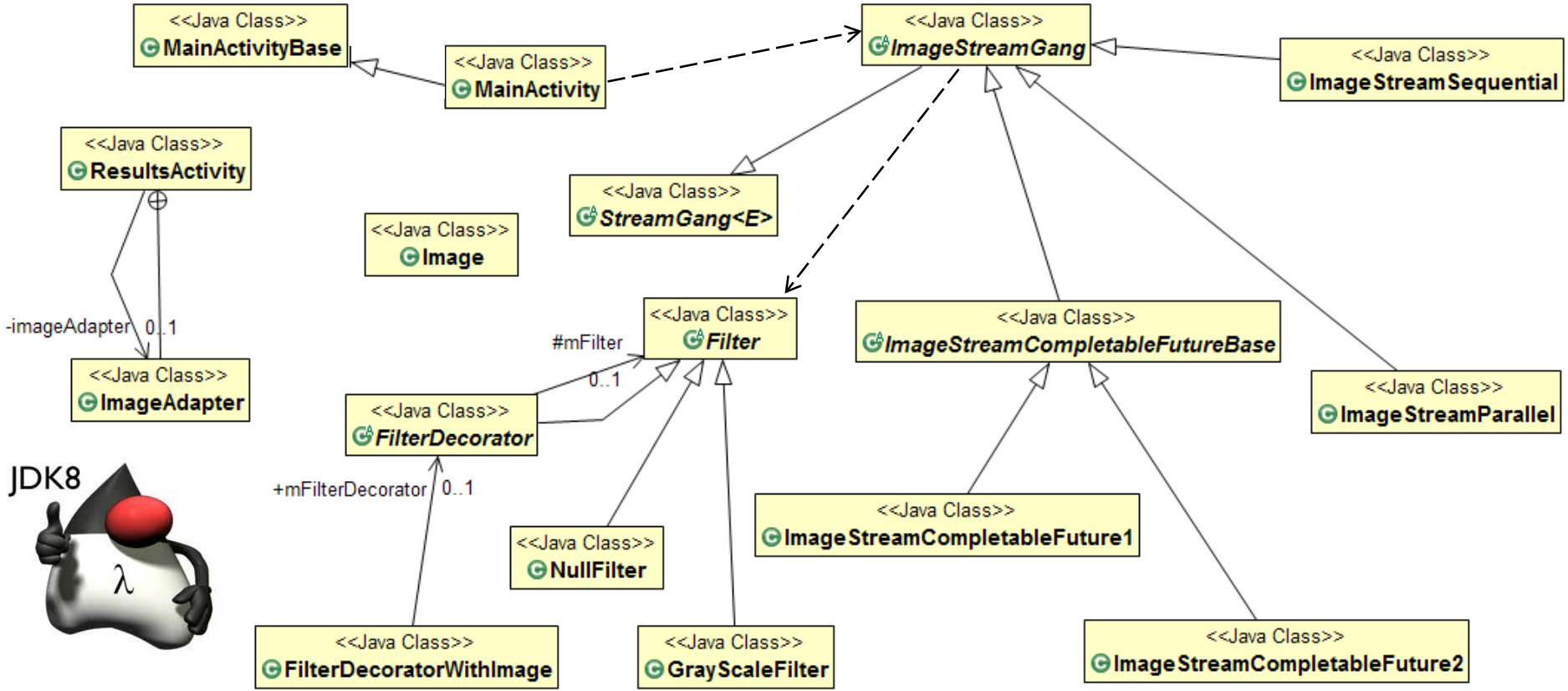
- "Gang-of-Four" & POSA patterns are applied to enhance its framework-based architecture
  - Patterns most essential to its design
    - *Pipes and Filters & Future*
    - *Pooling*
    - *Template Method*
    - *Decorator & Factory Method*
  - The *Singleton* & *Command* patterns are also used in its implementation



See en.wikipedia.org/wiki/Singleton_pattern & en.wikipedia.org/wiki/Command_pattern

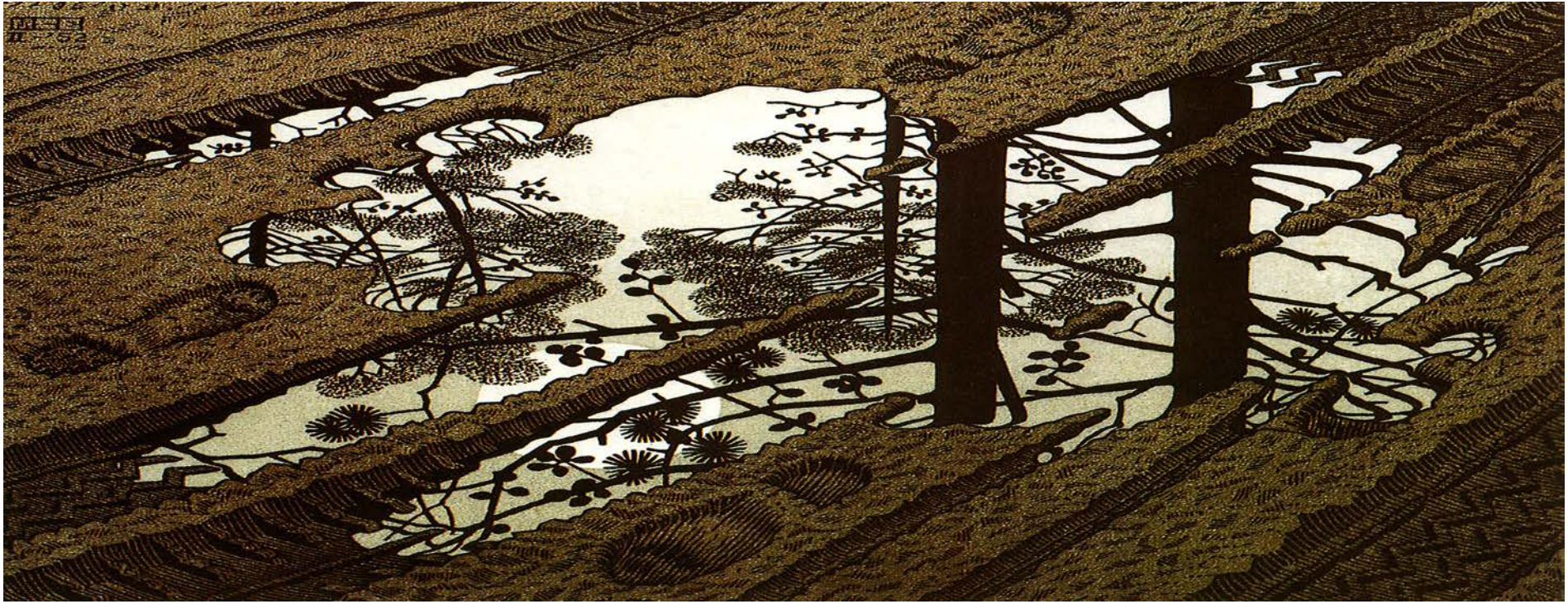
Overview of the Pattern-Oriented ImageStreamGang App

- This app is complicated & contains many classes



Overview of the Pattern-Oriented 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

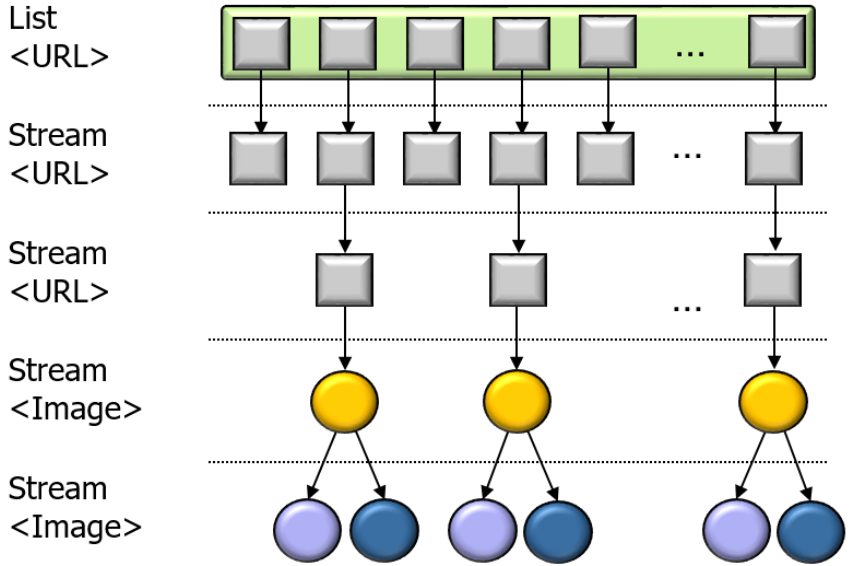
Overview of the Pattern-Oriented ImageStreamGang App

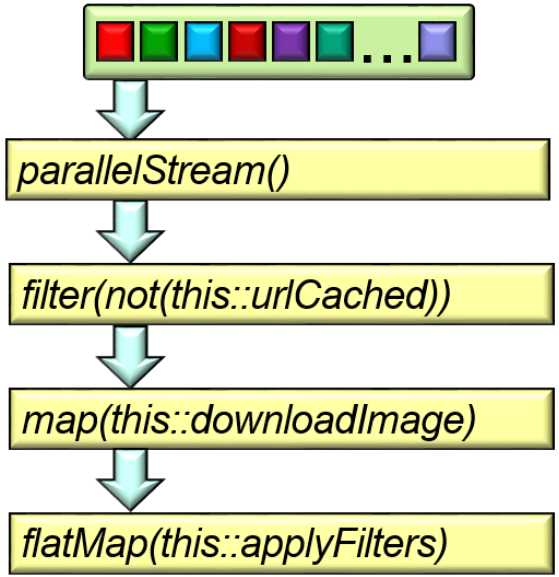
- This app is complicated & contains many classes
  - We therefore analyze it from various perspectives
  - Watch this video carefully to understand how it all works

See Lesson 4.4 at www.dre.vanderbilt.edu/~schmidt/LiveLessons/CPiJava

Overview of the Pattern-Oriented ImageStreamGang App

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





Overview of the Pattern-Oriented ImageStreamGang App

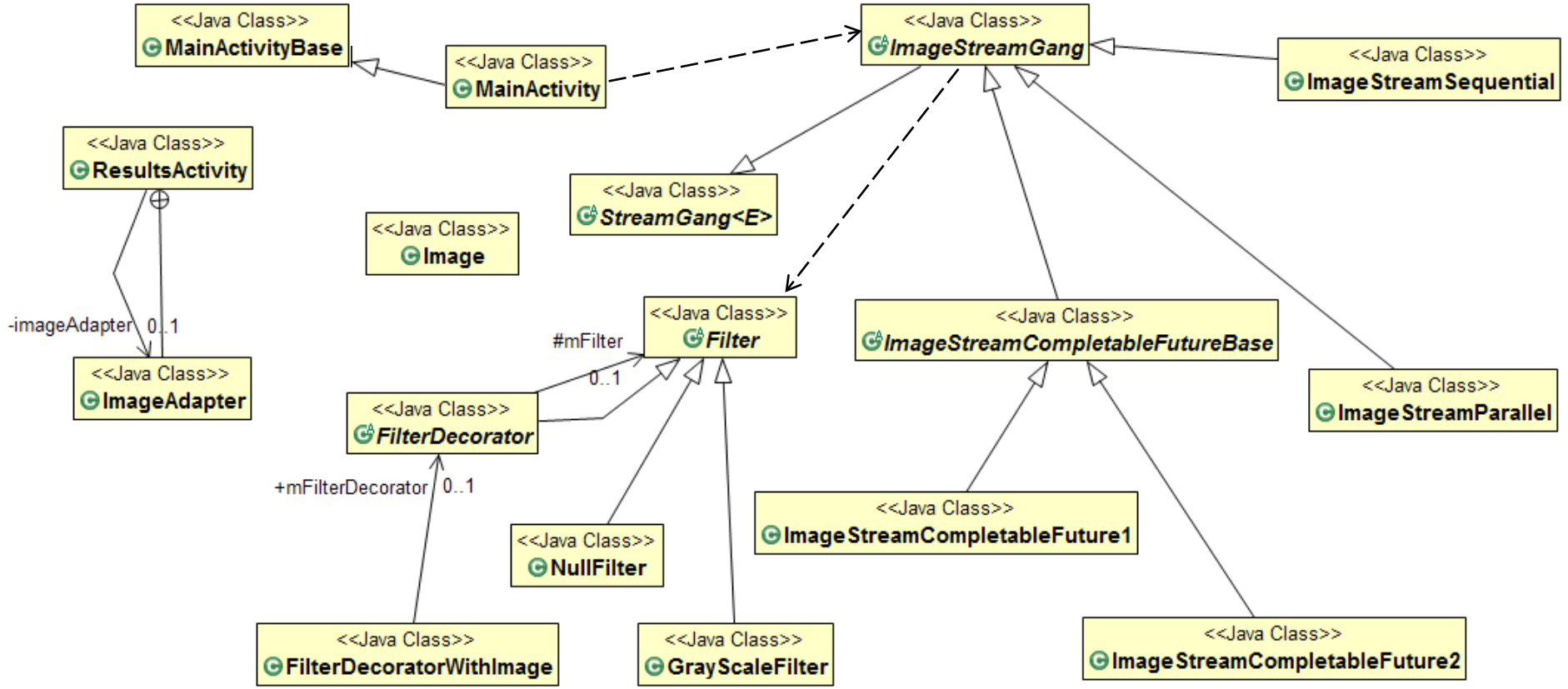
- This app is complicated & contains many classes
  - We therefore analyze it from various perspectives
  - Watch this video carefully to understand how it all works
  - Visualize the data flow in a parallel stream
  - Run/read the code to see how it all works

See github.com/douglascraigschmidt/LiveLessons/tree/master/ImageStreamGang

The Structure of the ImageStreamGang App

The Structure of the ImageStreamGang App

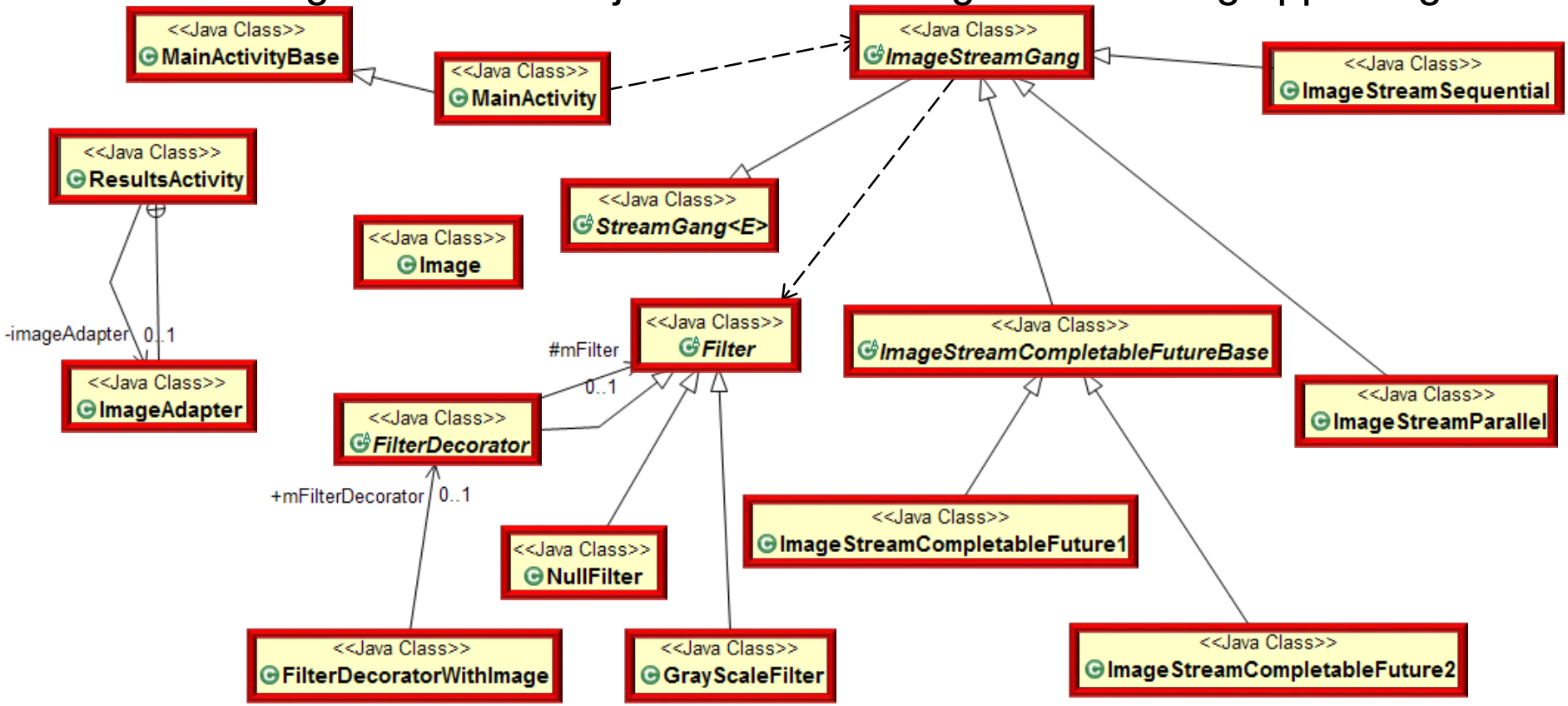
- UML class diagram for the object-oriented ImageStreamGang app design



This design shows the synergy between object-oriented & functional programming

The Structure of the ImageStreamGang App

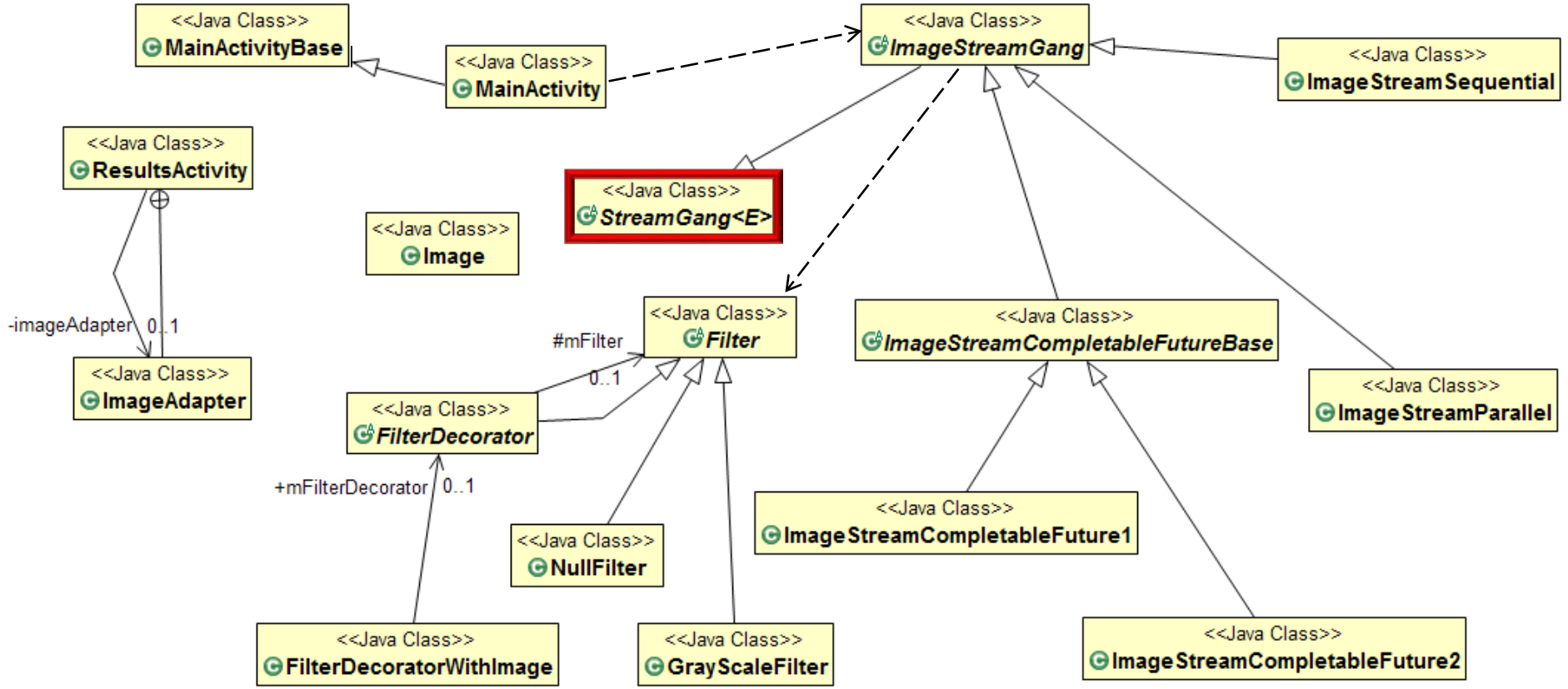
- UML class diagram for the object-oriented ImageStreamGang app design



These classes apply Java 8 features to image downloading & processing

The Structure of the ImageStreamGang App

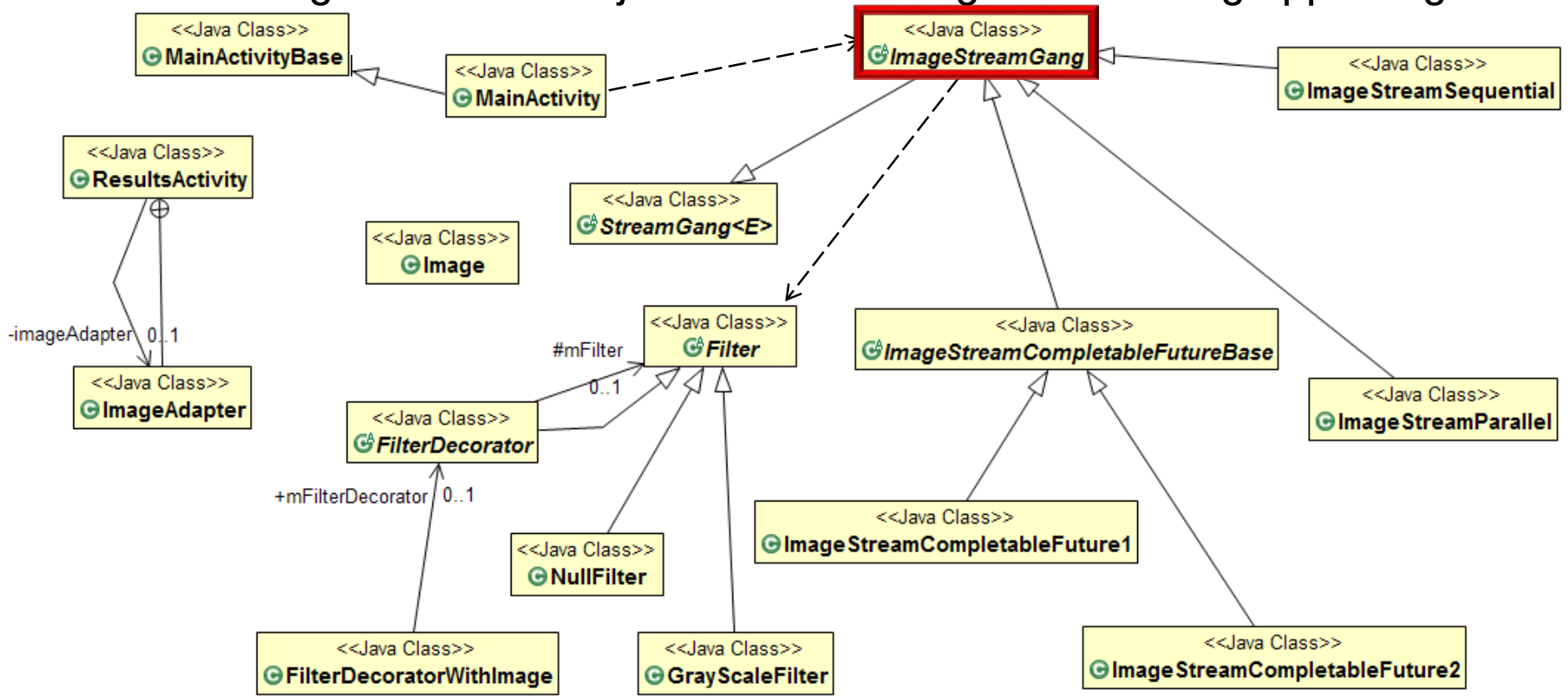
- UML class diagram for the object-oriented ImageStreamGang app design



A framework for initiating streams that process input from a list of elements

The Structure of the ImageStreamGang App

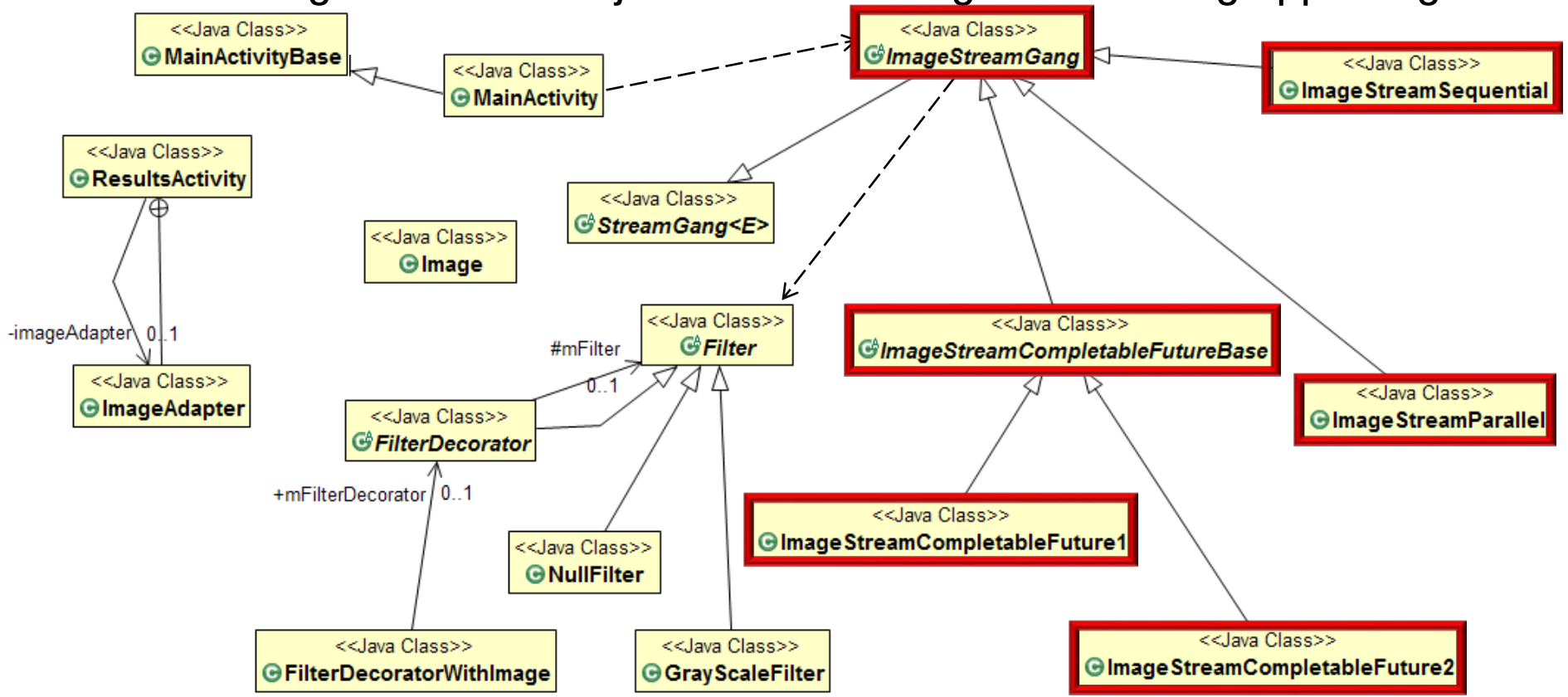
- UML class diagram for the object-oriented ImageStreamGang app design



Customizes the StreamGang framework to download & process images ...

The Structure of the ImageStreamGang App

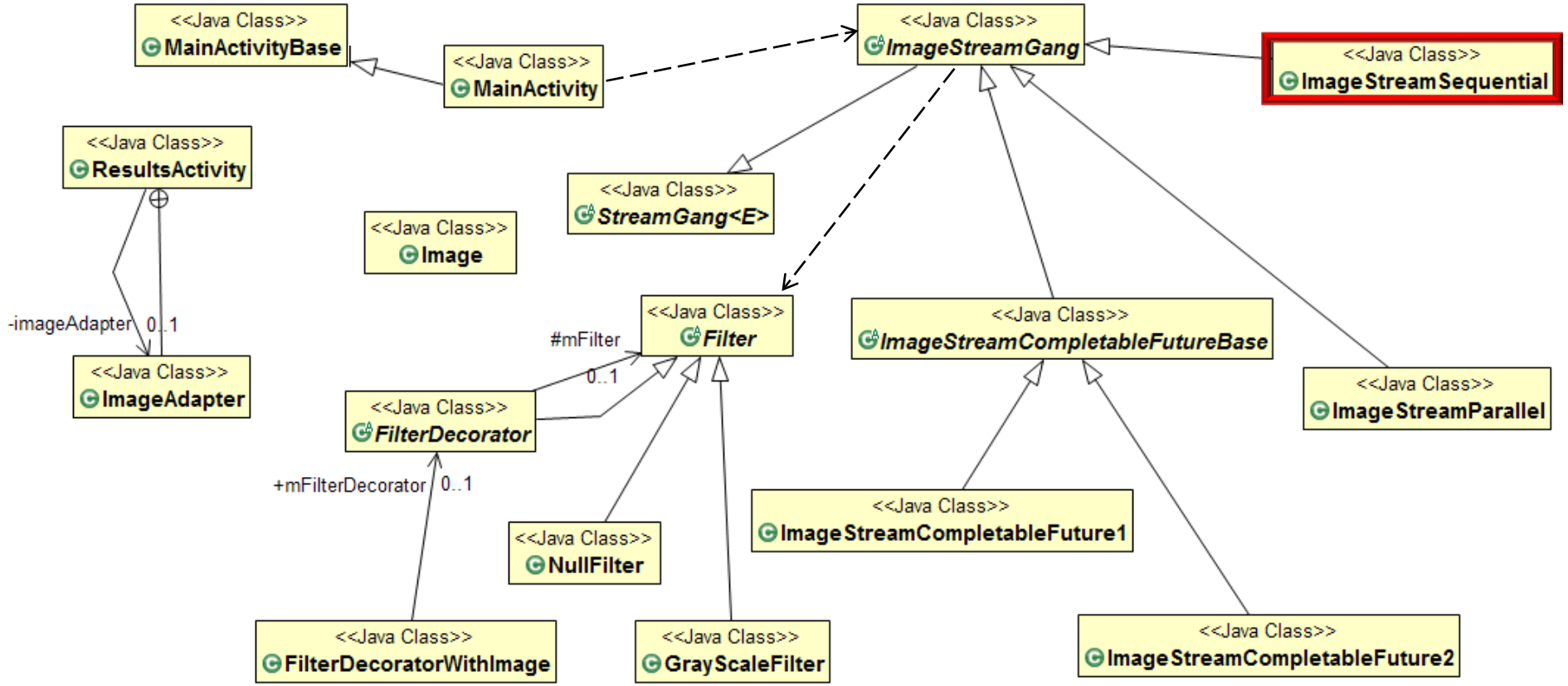
- UML class diagram for the object-oriented ImageStreamGang app design



... based on different Java 8 concurrency & parallelism frameworks

The Structure of the ImageStreamGang App

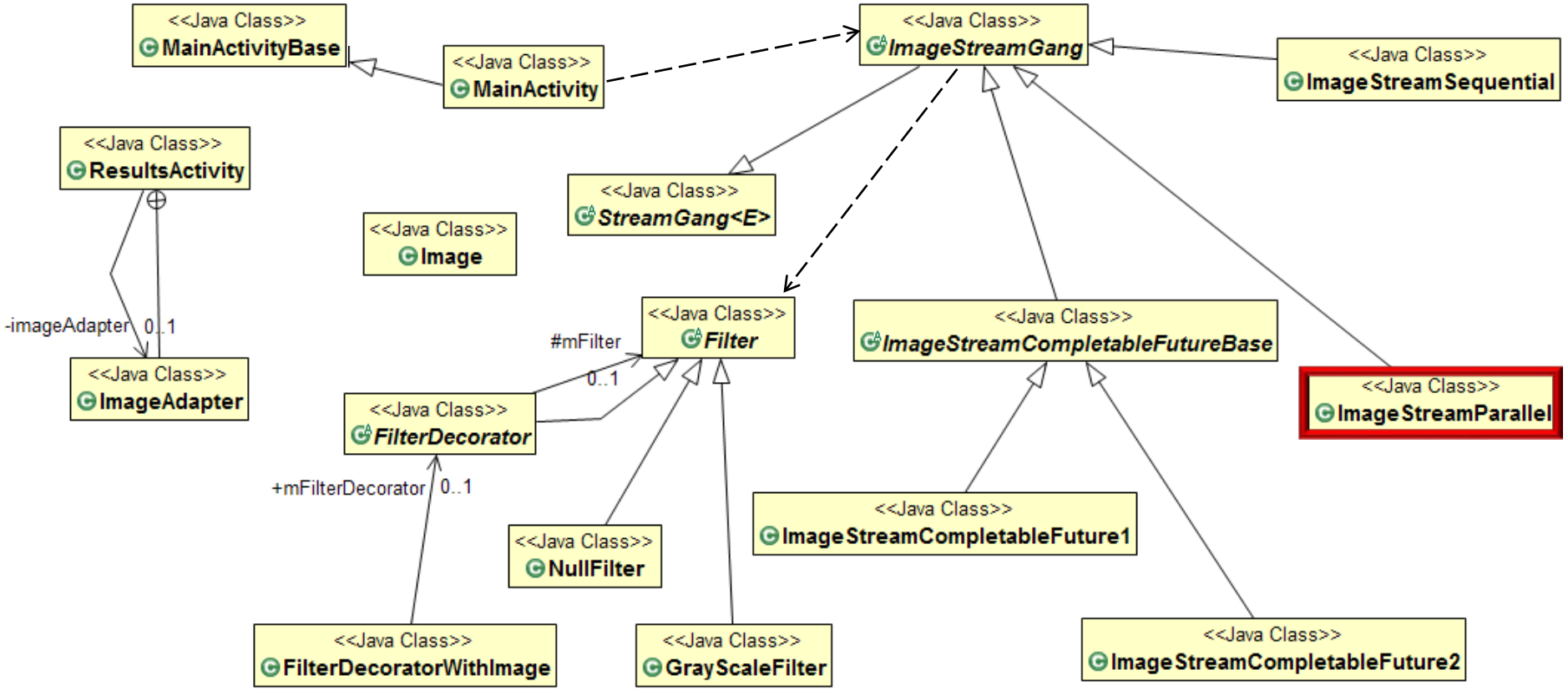
- UML class diagram for the object-oriented ImageStreamGang app design



Uses Java 8 streams to download & filter images sequentially

The Structure of the ImageStreamGang App

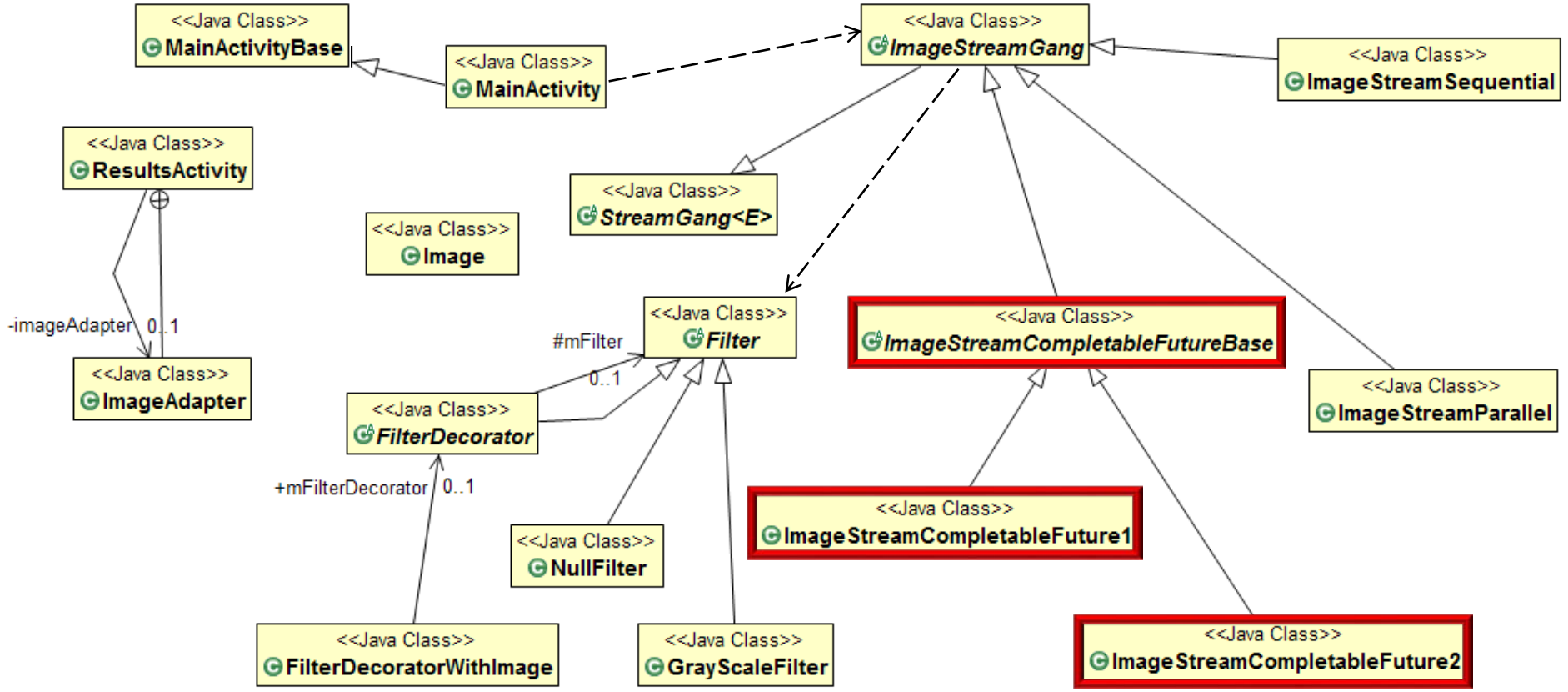
- UML class diagram for the object-oriented ImageStreamGang app design



Uses Java 8 parallel streams to download & filter images concurrently

The Structure of the ImageStreamGang App

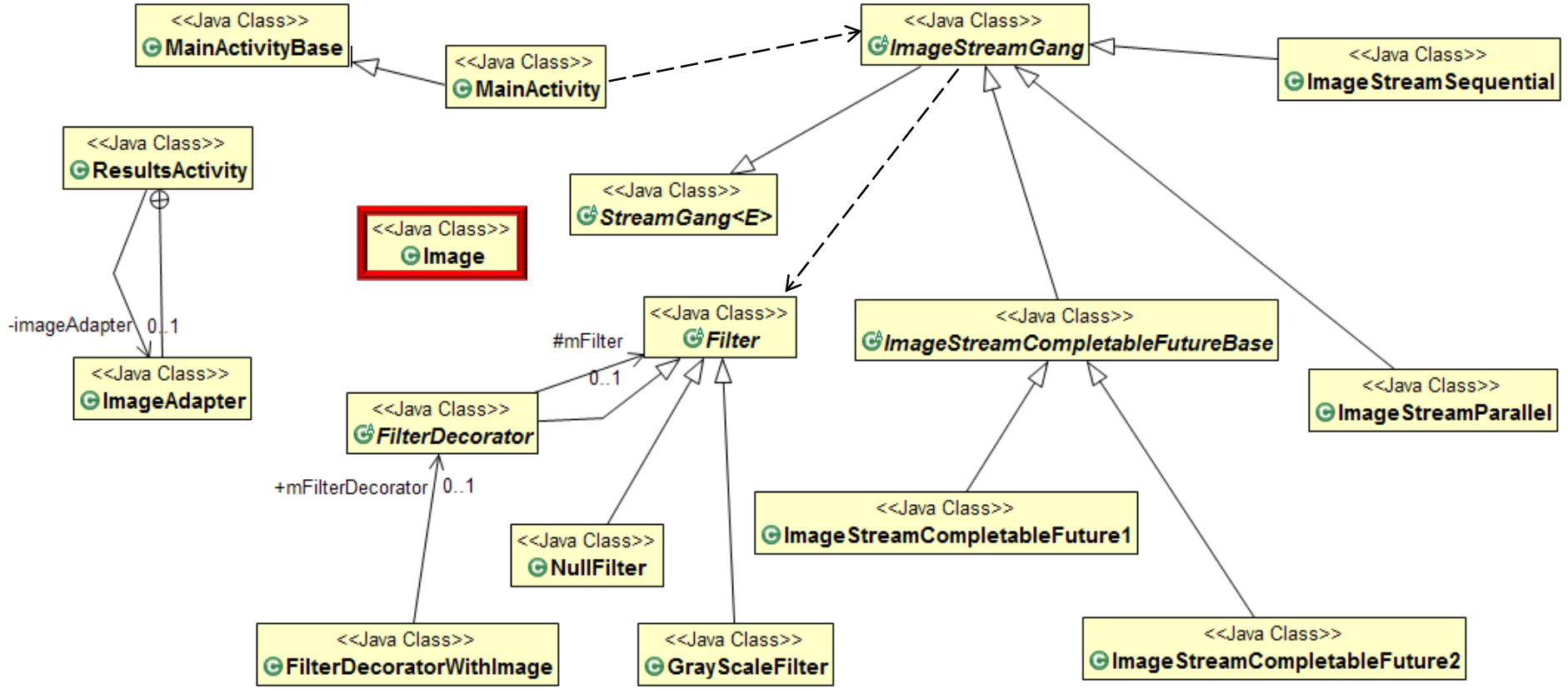
- UML class diagram for the object-oriented ImageStreamGang app design



Uses Java 8 CompletableFutures to download & filter images asynchronously

The Structure of the ImageStreamGang App

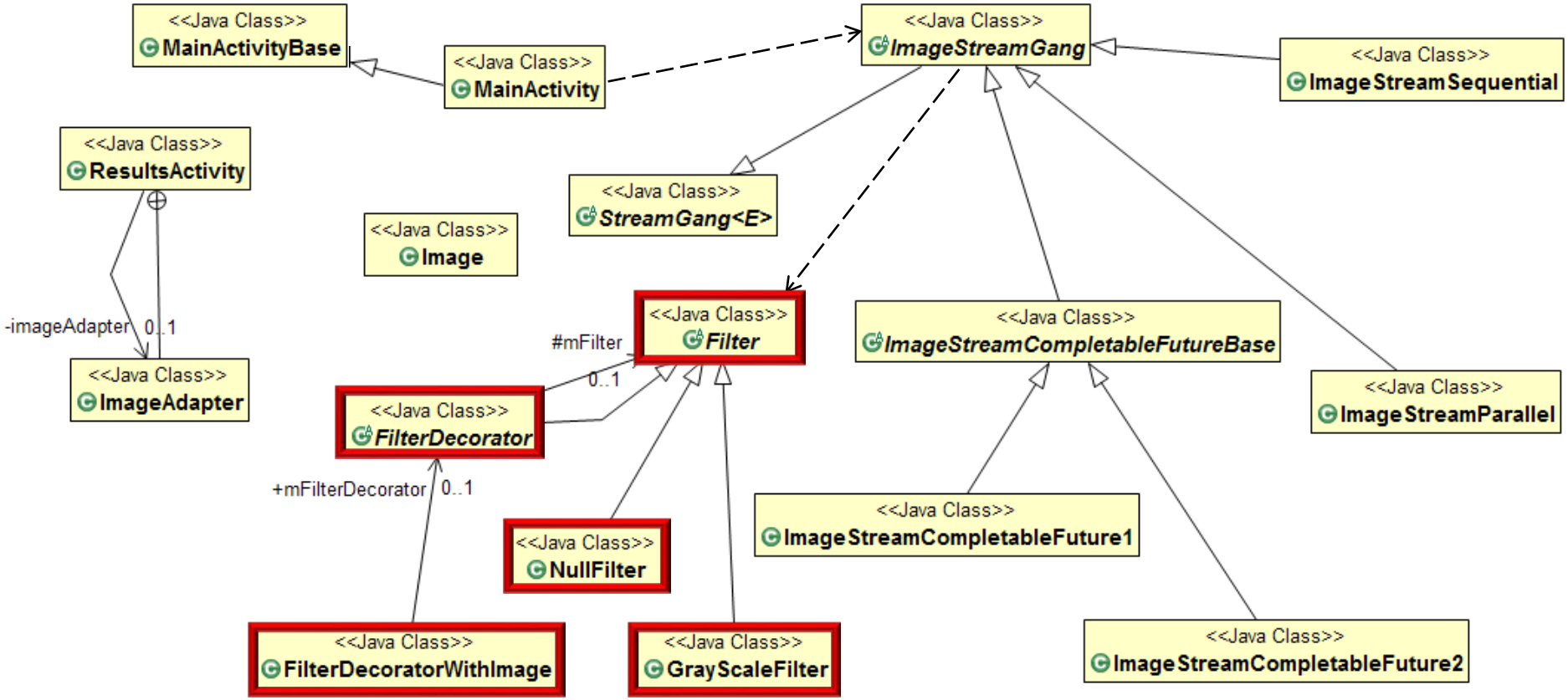
- UML class diagram for the object-oriented ImageStreamGang app design



Stores image meta-data & provides methods for common image-/file-related tasks

The Structure of the ImageStreamGang App

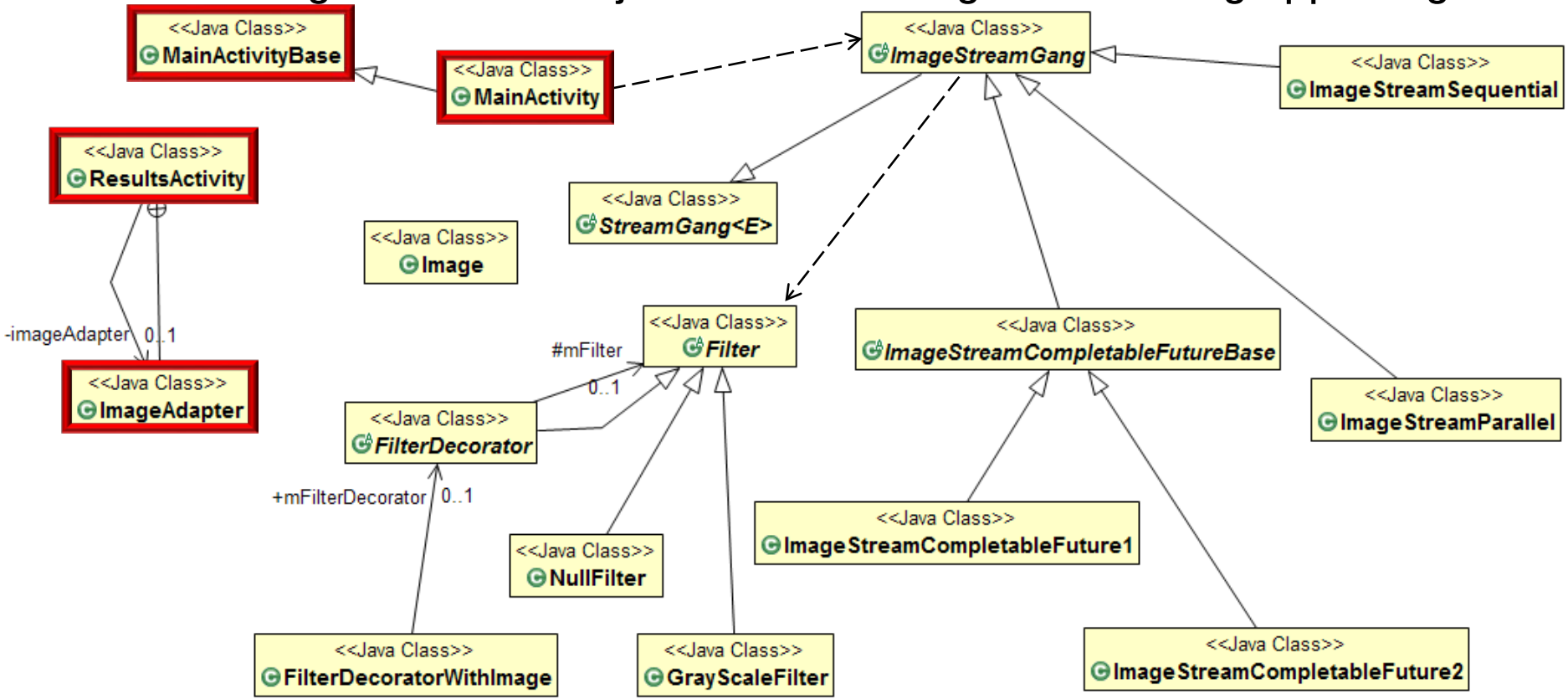
- UML class diagram for the object-oriented ImageStreamGang app design



This class hierarchy applies operations to filter & store images

The Structure of the ImageStreamGang App

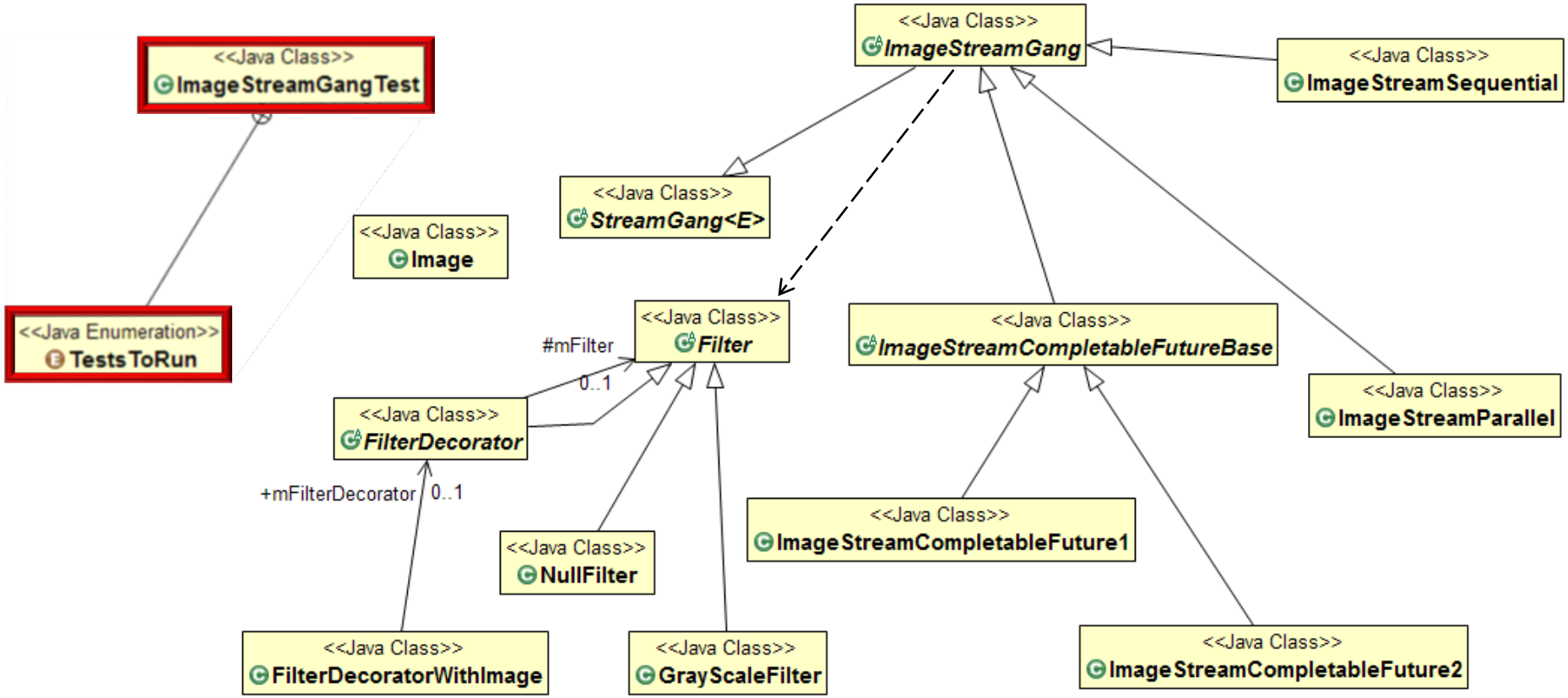
- UML class diagram for the object-oriented ImageStreamGang app design



Provides the user interface for an Android app

The Structure of the ImageStreamGang App

- UML class diagram for the object-oriented ImageStreamGang app design



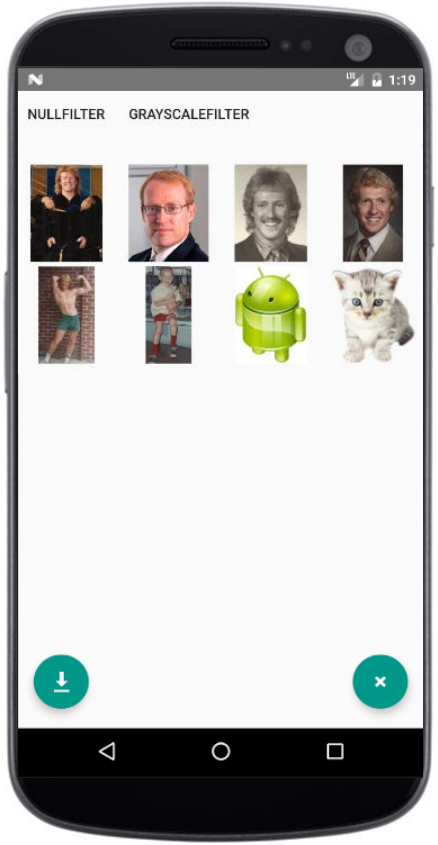
There's a Java console version of ImageStreamGang that shares most of the code

Running the Image StreamGang App

Running the ImageStreamGang App

Printing 4 results for input file 1 from fastest to slowest
COMPLETABLE_FUTURES_2 executed in 1128 msecs
COMPLETABLE_FUTURES_1 executed in 1146 msecs
PARALLEL_STREAM executed in 1415 msecs
SEQUENTIAL_STREAM executed in 2973 msecs

Printing 4 results for input file 2 from fastest to slowest
COMPLETABLE_FUTURES_2 executed in 204 msecs
COMPLETABLE_FUTURES_1 executed in 244 msecs
PARALLEL_STREAM executed in 281 msecs
SEQUENTIAL_STREAM executed in 786 msecs
Ending ImageStreamGangTest



See Lesson 4.4 at www.safaribooksonline.com/library/view/java-concurrency-2e/9780134510644

End of Java 8 Parallel ImageStreamGang Example (Part 1)