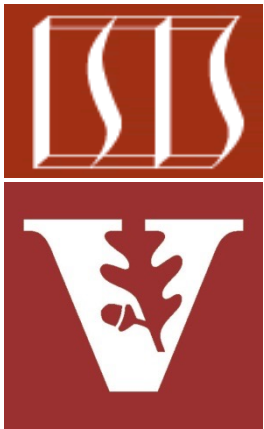


Wrapping Up the ImageTaskGang

Application Analysis



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 the implementation of the ImageTaskGangTest class

ImageTaskGangTest	
m	ImageTaskGangTest()
f	<i>mFilters</i> Filter[]
m	runTests() void
m	makeImageTaskGang(Filter[], List<List<URL>>, TestsToRun) ImageTaskGang
m	warmUpThreadPool() void
m	main(String[]) void

See [ImageTaskGang/src/main/java/livelessons/ImageTaskGangTest.java](https://github.com/leecostello/livelessons/blob/master/src/main/java/livelessons/ImageTaskGangTest.java)

Learning Objectives in this Part of the Lesson

- Show the command-line & Android apps & summarize key points of the lesson

The screenshot displays an IDE with two main windows. The left window shows the command-line output for running the application. The right window shows the Android Studio interface with the project structure and a running device.

```
ImageTaskGang src main java livelessons ImageTaskGangTest runTests
Run: ImageTaskGang [ImageTaskGangTest.main()]
> Task :classes
> Task :ImageTaskGangTest.main()
Starting ImageTaskGangTest
Starting EXECUTOR_COMPLETION_SERVICE_CACHED
EXECUTOR_COMPLETION_SERVICE_CACHED: 28 operations succeeded and 0 operations failed.
Ending EXECUTOR_COMPLETION_SERVICE_CACHED
Starting EXECUTOR_COMPLETION_SERVICE_FIXED
EXECUTOR_COMPLETION_SERVICE_FIXED: 28 operations succeeded and 0 operations failed.
Ending EXECUTOR_COMPLETION_SERVICE_FIXED

Printing 2 results from fastest to slowest
EXECUTOR_COMPLETION_SERVICE_FIXED executed in 36
EXECUTOR_COMPLETION_SERVICE_CACHED executed in 4

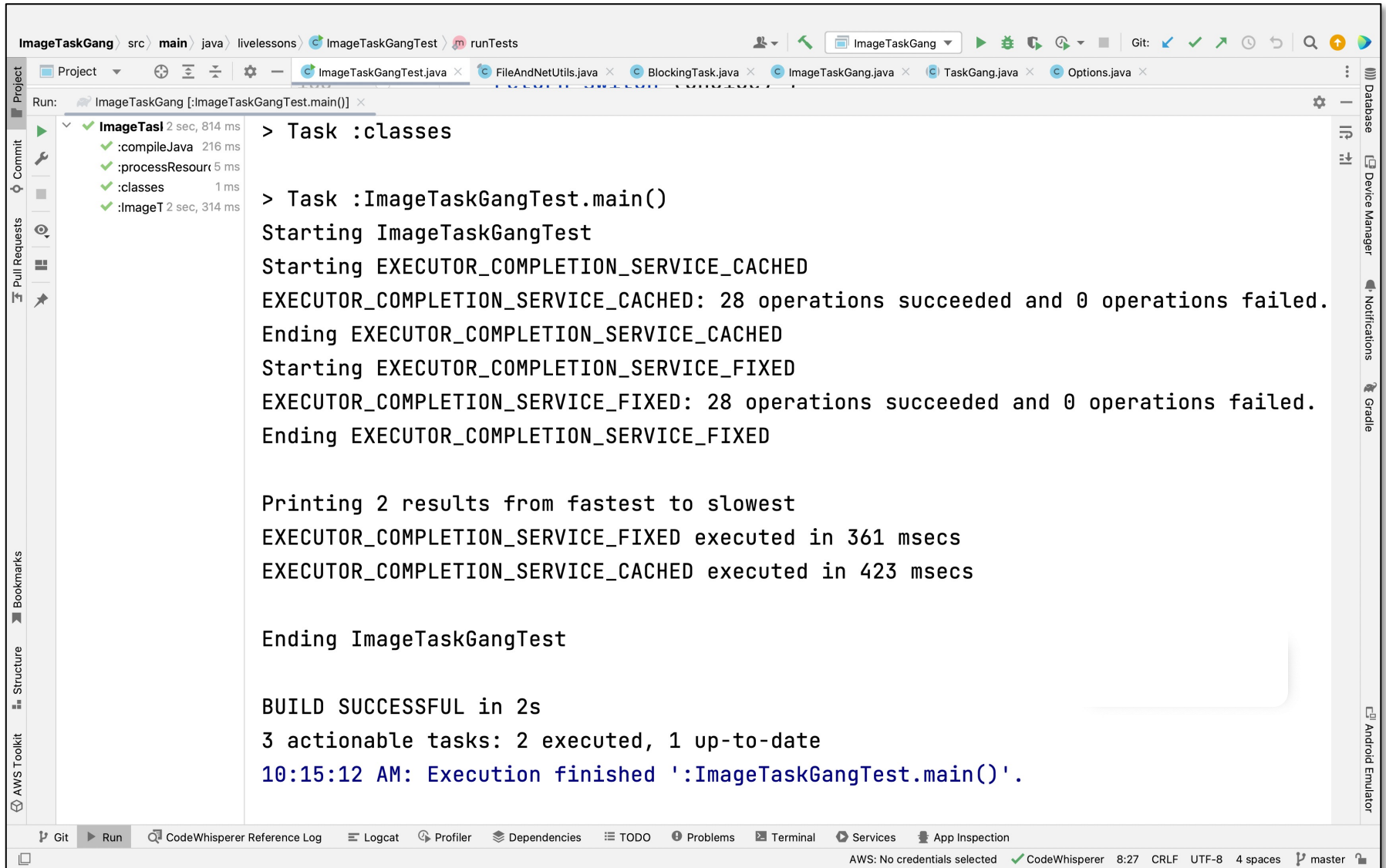
Ending ImageTaskGangTest

BUILD SUCCESSFUL in 2s
3 actionable tasks: 2 executed, 1 up-to-date
10:15:12 AM: Execution finished ':ImageTaskGang'
```

The right window shows the project structure for 'example | imagetaskgang | tasks | ImageTaskGang'. The running device (Pixel 6 Pro API 33) displays the application's UI, which shows a grid of images with filters applied. The filters are 'NULLFILTER' and 'GRAYSCALEFILTER'. The images are arranged in a 2x4 grid, with the last cell containing a circular portrait of a man.

Running the ImageTaskGang Application

Running the ImageTaskGang Command-line App



```
ImageTaskGang [ImageTaskGangTest.main()] x
> Task :classes
> Task :ImageTaskGangTest.main()
Starting ImageTaskGangTest
Starting EXECUTOR_COMPLETION_SERVICE_CACHED
EXECUTOR_COMPLETION_SERVICE_CACHED: 28 operations succeeded and 0 operations failed.
Ending EXECUTOR_COMPLETION_SERVICE_CACHED
Starting EXECUTOR_COMPLETION_SERVICE_FIXED
EXECUTOR_COMPLETION_SERVICE_FIXED: 28 operations succeeded and 0 operations failed.
Ending EXECUTOR_COMPLETION_SERVICE_FIXED

Printing 2 results from fastest to slowest
EXECUTOR_COMPLETION_SERVICE_FIXED executed in 361 msec
EXECUTOR_COMPLETION_SERVICE_CACHED executed in 423 msec

Ending ImageTaskGangTest

BUILD SUCCESSFUL in 2s
3 actionable tasks: 2 executed, 1 up-to-date
10:15:12 AM: Execution finished ':ImageTaskGangTest.main()'.

AWS: No credentials selected ✓ CodeWhisperer 8:27 CRLF UTF-8 4 spaces master
```

See github.com/douglasraigschmidt/LiveLessons/tree/master/ImageTaskGang

Running the ImageTaskGang Android App

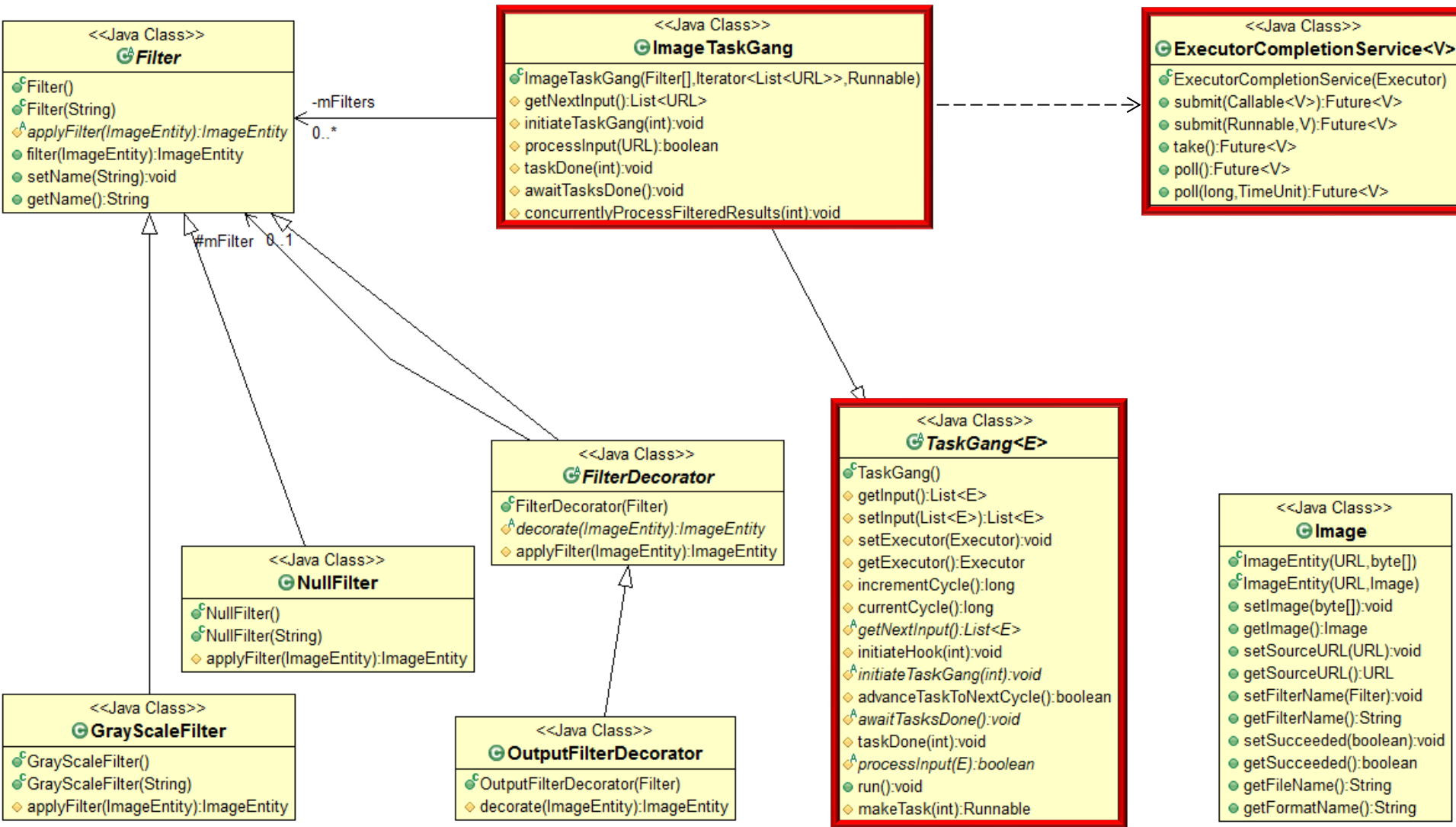
The screenshot displays an IDE interface for running an Android application. On the left, the 'Resource Manager' shows the project structure for 'example.imagetaskgang', including folders for 'filters', 'main', 'platform', 'tasks', and 'utils'. The 'tasks' folder is expanded, highlighting 'ImageTaskGang'. The top toolbar shows the 'Run' button (a green play icon) and the selected device 'Pixel 6 Pro API 33'. The central area shows a virtual device with the app running. The app's UI features two tabs: 'NULLFILTER' and 'GRAYSCALEFILTER'. Below the tabs, there are two rows of image thumbnails. The first row shows four images: a person in a dark coat, a person in a light shirt, a person in a suit, and a person in a suit. The second row shows four images: a person in a white shirt, a person in a white shirt, a person in a suit, and a circular portrait of a person in a suit. The bottom status bar indicates 'Launch succeeded (a minute ago)' and includes various tool icons like Git, Run, Profiler, Logcat, App Quality Insights, Build, TODO, Problems, Terminal, Services, App Inspection, and Upgrade Assistant.

See Android version at github.com/douglasraigschmidt/LiveLessons/tree/master/ImageTaskGangApplication

Summary of the ImageTaskGang Application

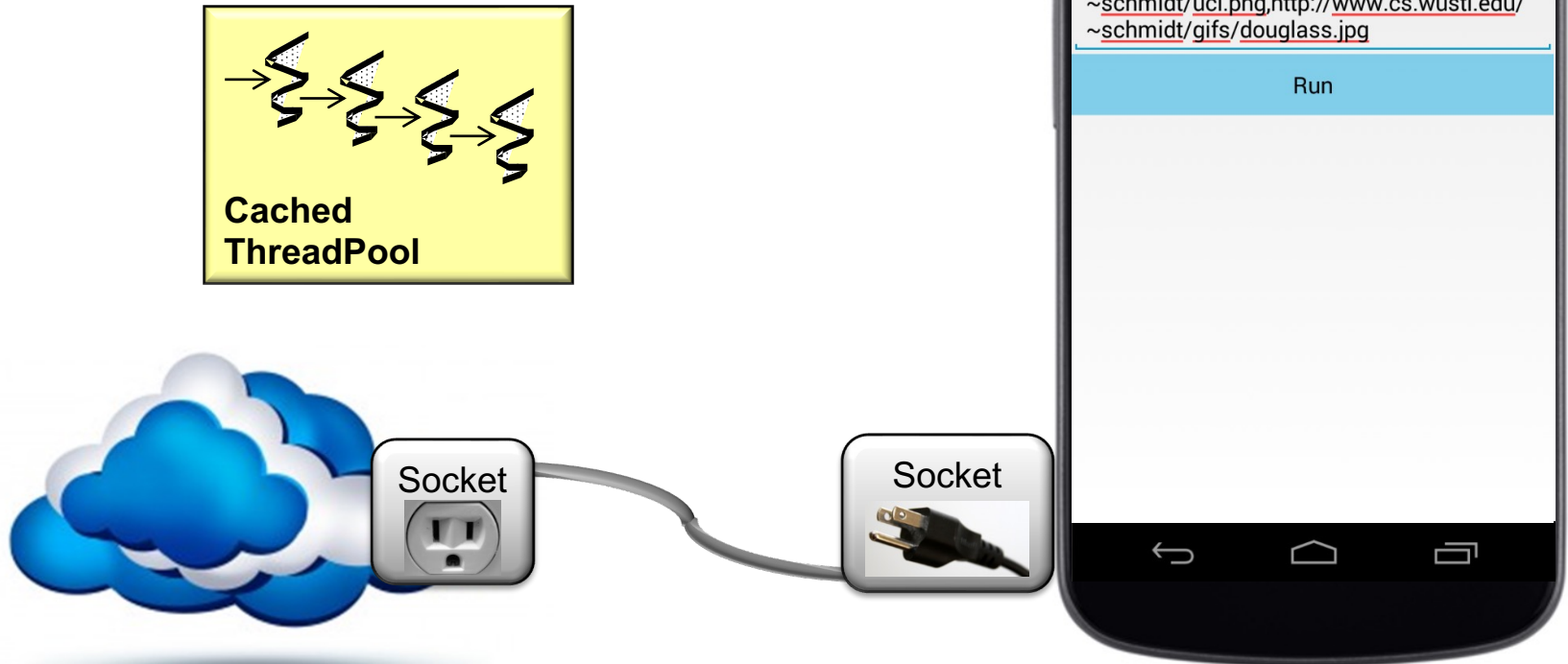
Summary of the ImageTaskGang Application

- Customizes the TaskGang framework & uses the Java Executor framework



Summary of the ImageTaskGang Application

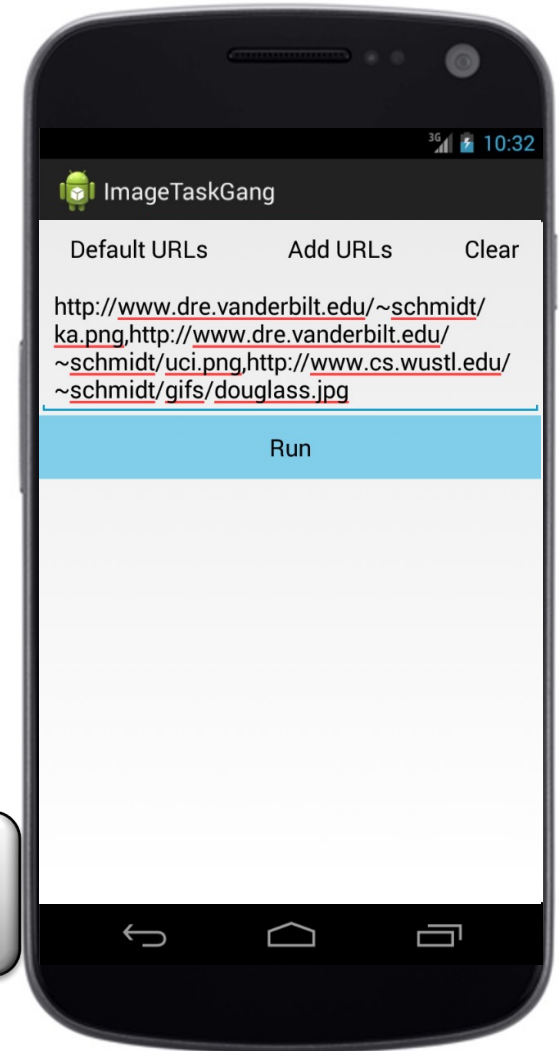
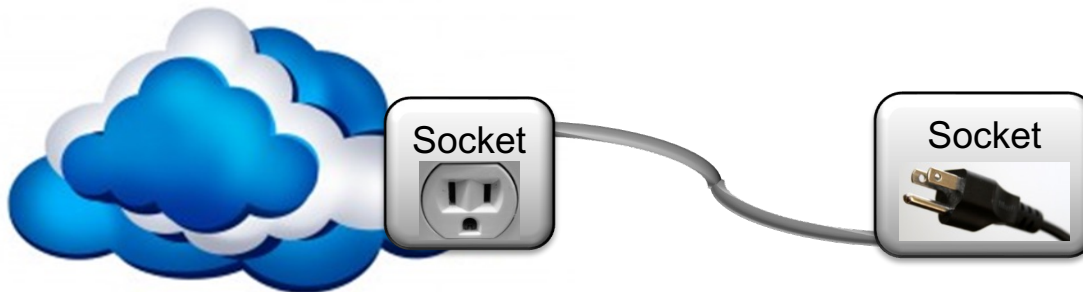
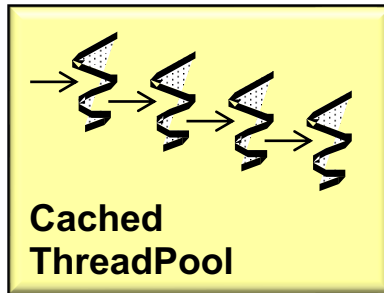
- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService



Summary of the ImageTaskGang Application

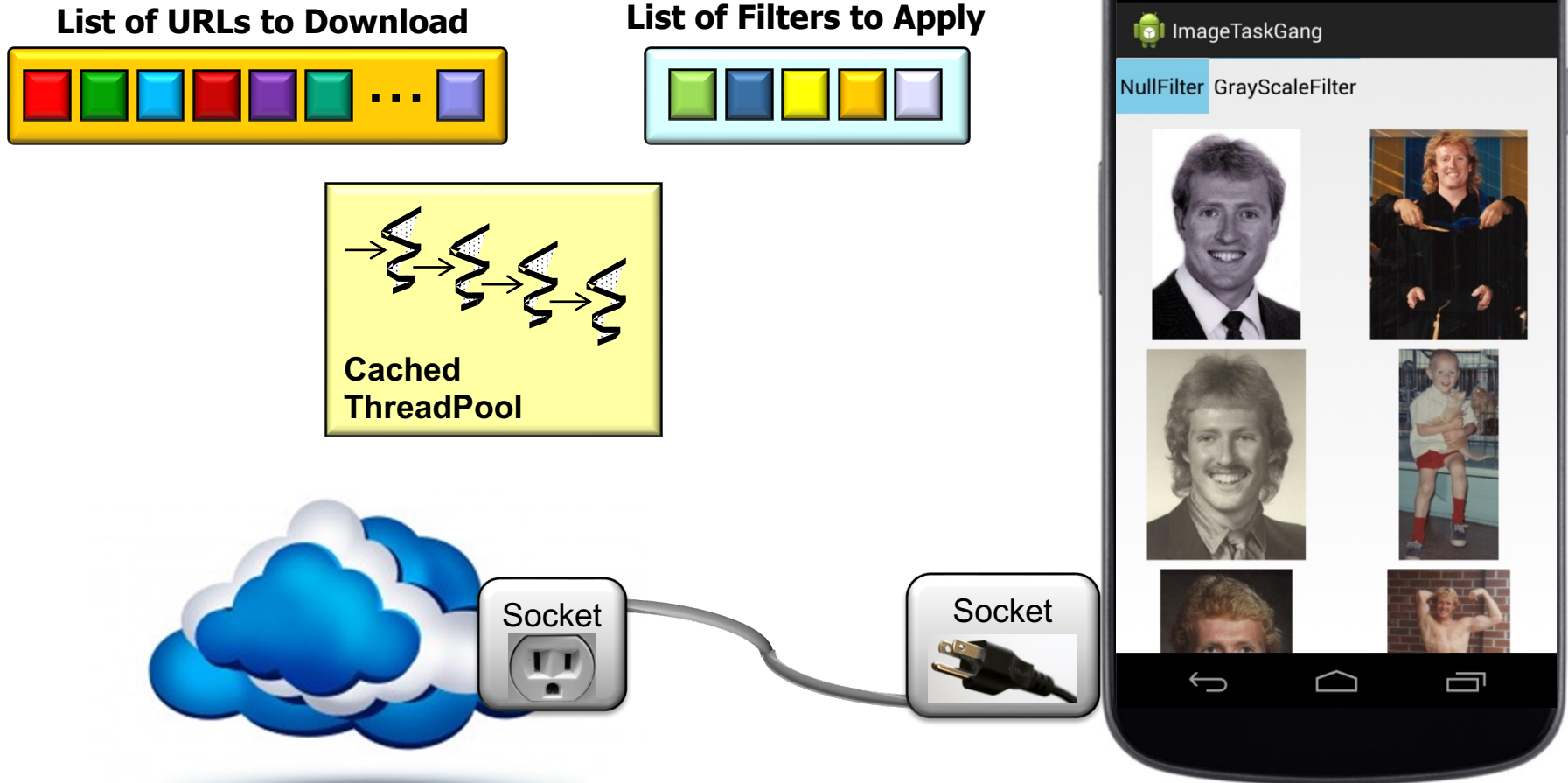
- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService

List of URLs to Download



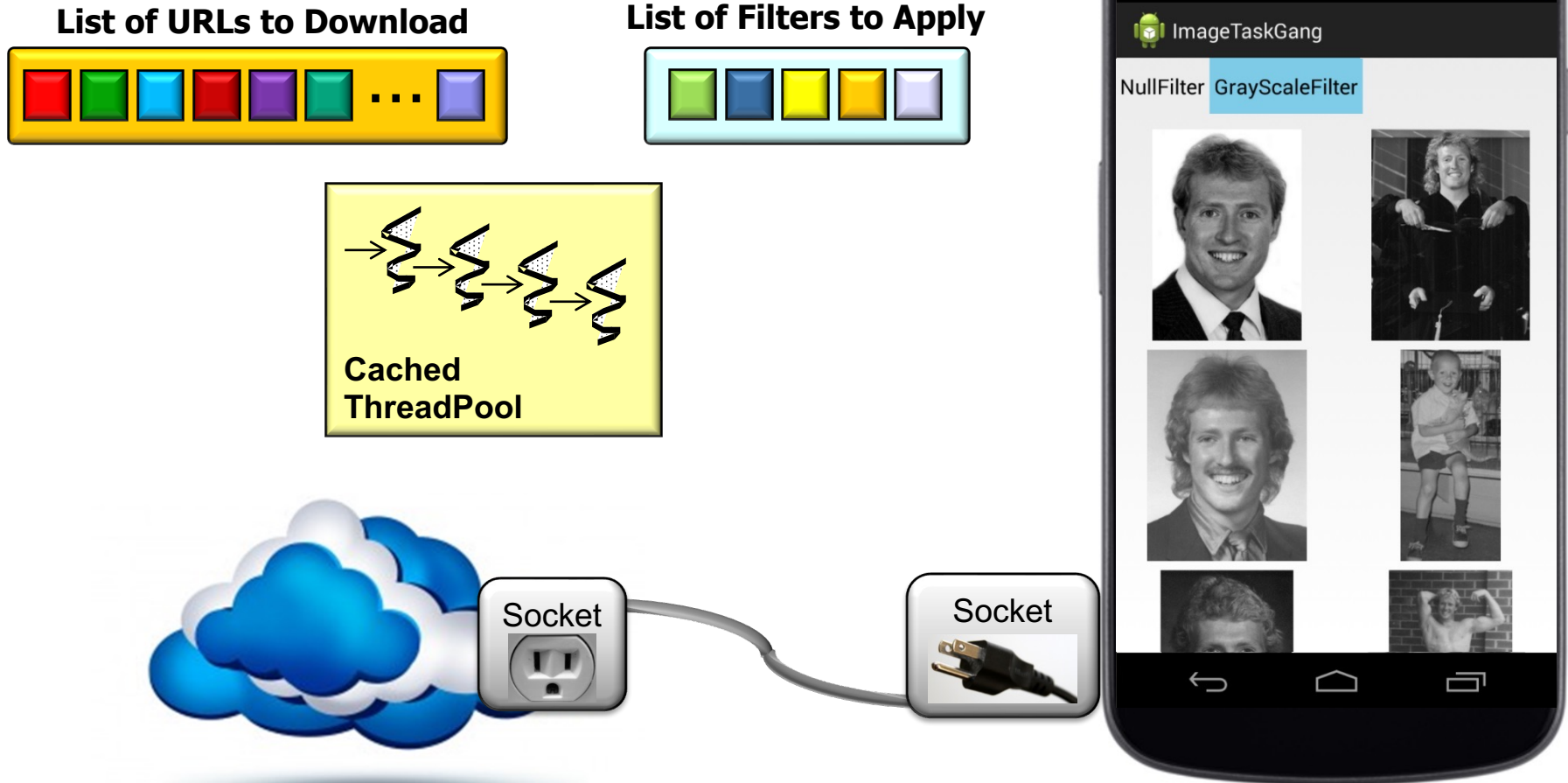
Summary of the ImageTaskGang Application

- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService



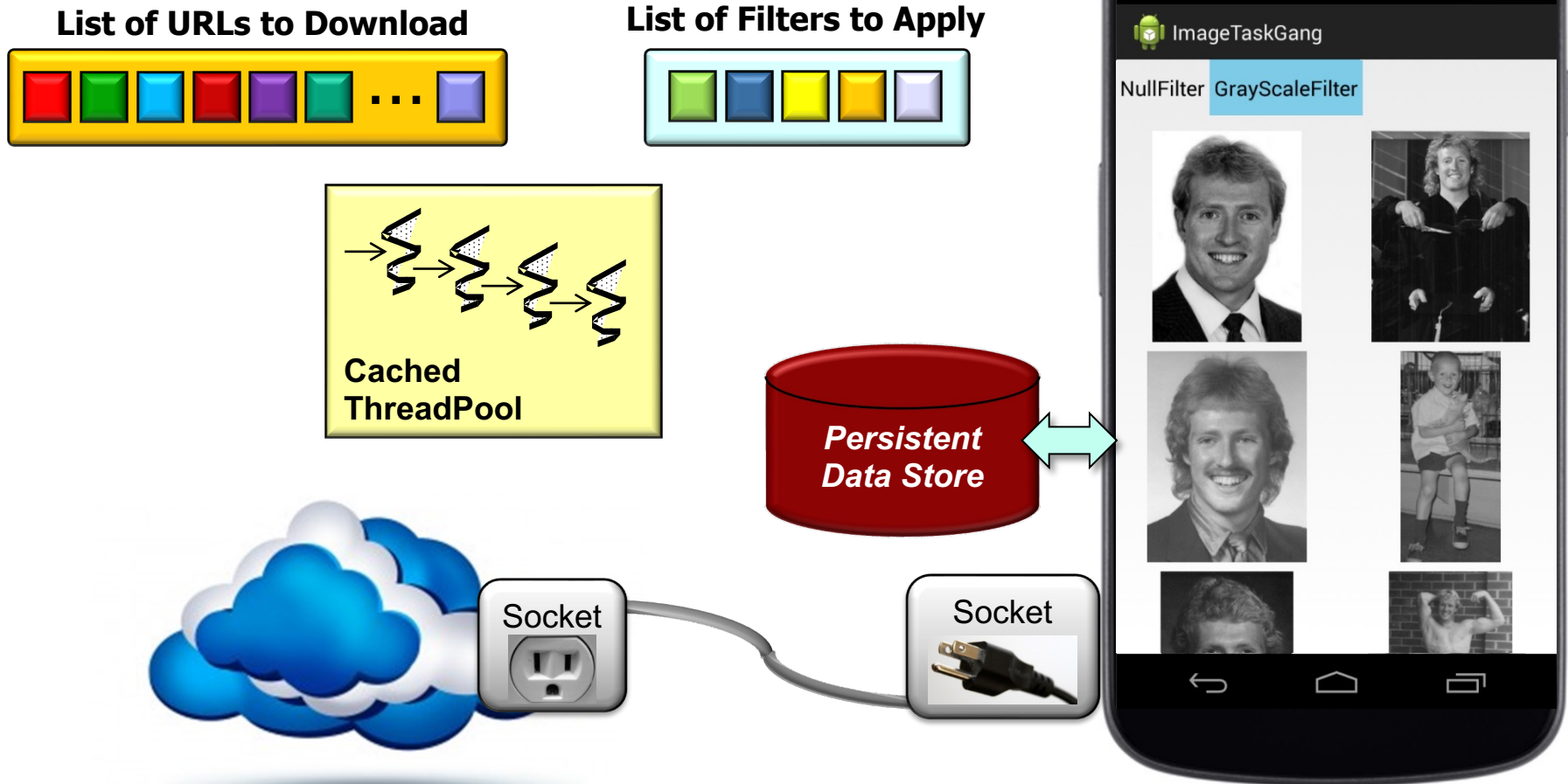
Summary of the ImageTaskGang Application

- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService



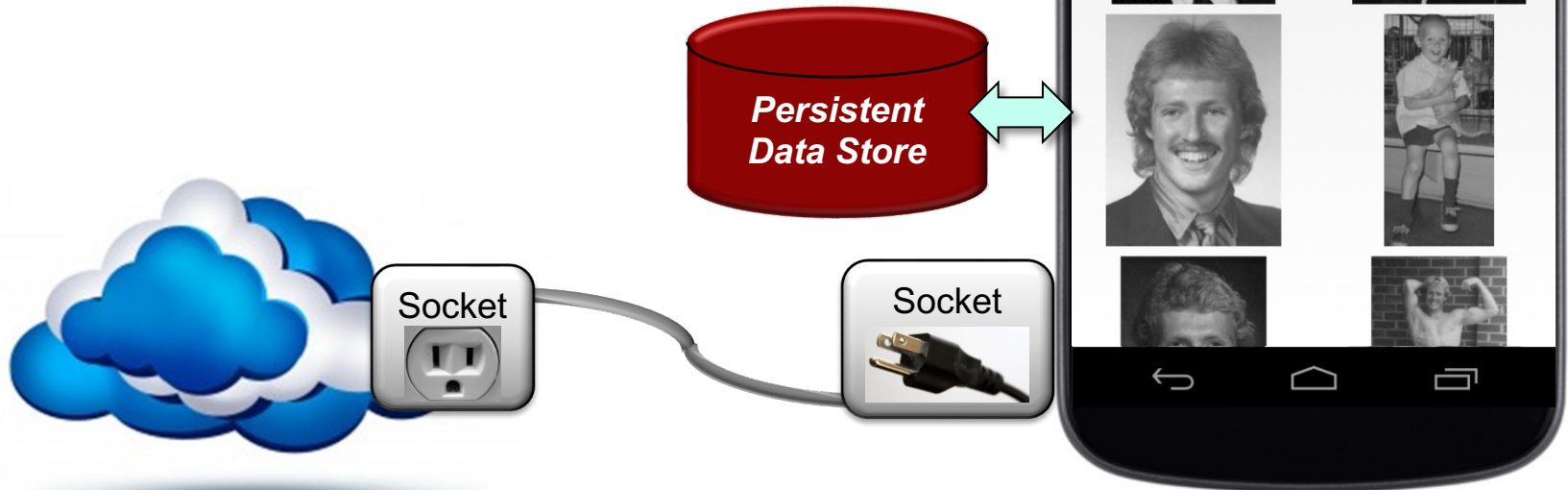
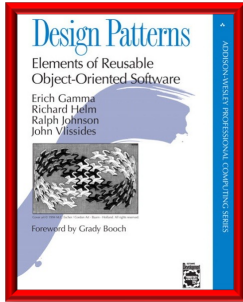
Summary of the ImageTaskGang Application

- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService



Summary of the ImageTaskGang Application

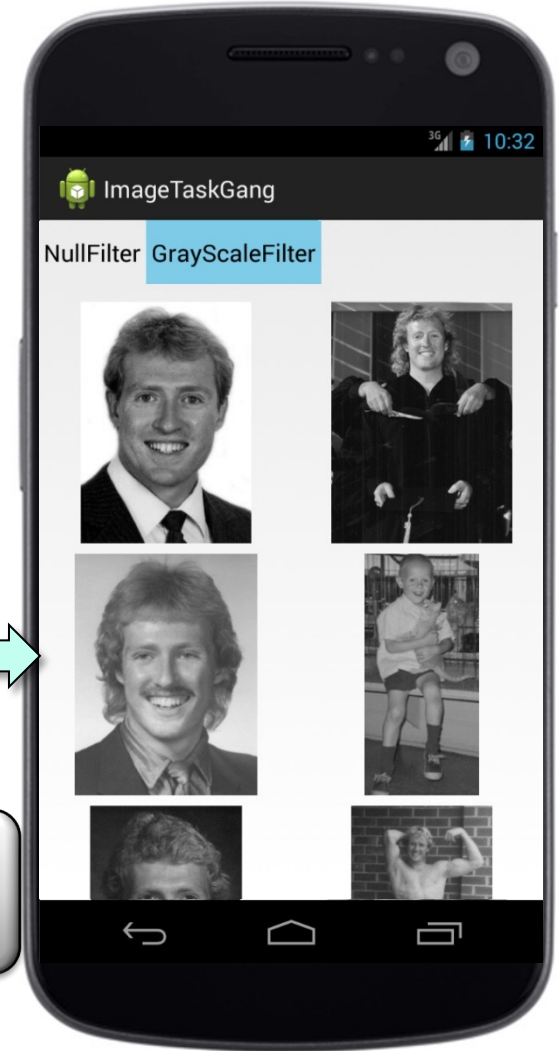
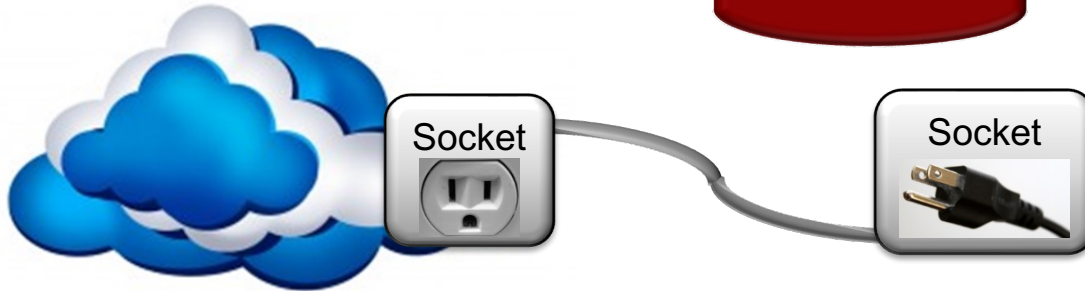
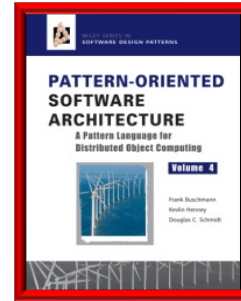
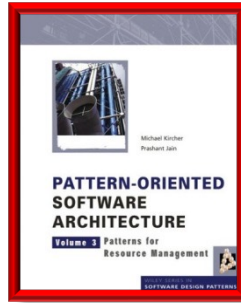
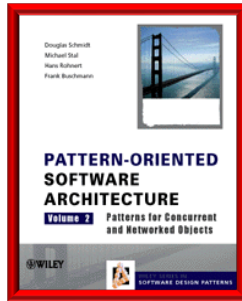
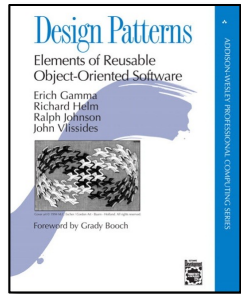
- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService



GoF & POSA patterns enhance app & framework reusability, flexibility, portability, & performance

Summary of the ImageTaskGang Application

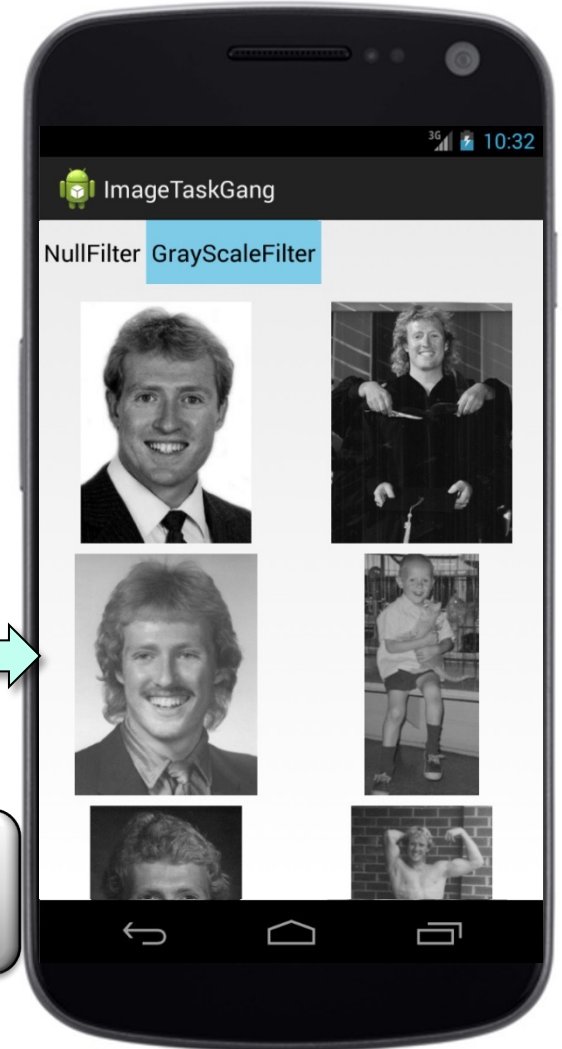
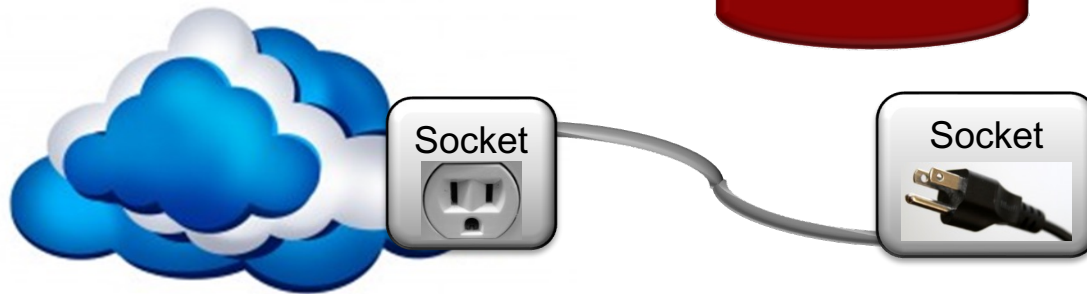
- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService



GoF & POSA patterns enhance app & framework reusability, flexibility, portability, & performance

Summary of the ImageTaskGang Application

- Shows how TaskGang framework can be applied to download, process, store, & display images concurrently via the Java ExecutorCompletionService



The knowledge of patterns simplifies the understanding of the ImageTaskGang app & framework & how to extend it

End of Wrapping Up the ImageTaskGang Application Analysis