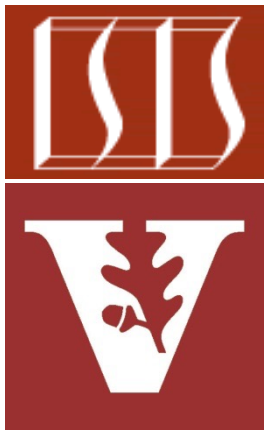
Overview of the ImageTaskGang Application



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

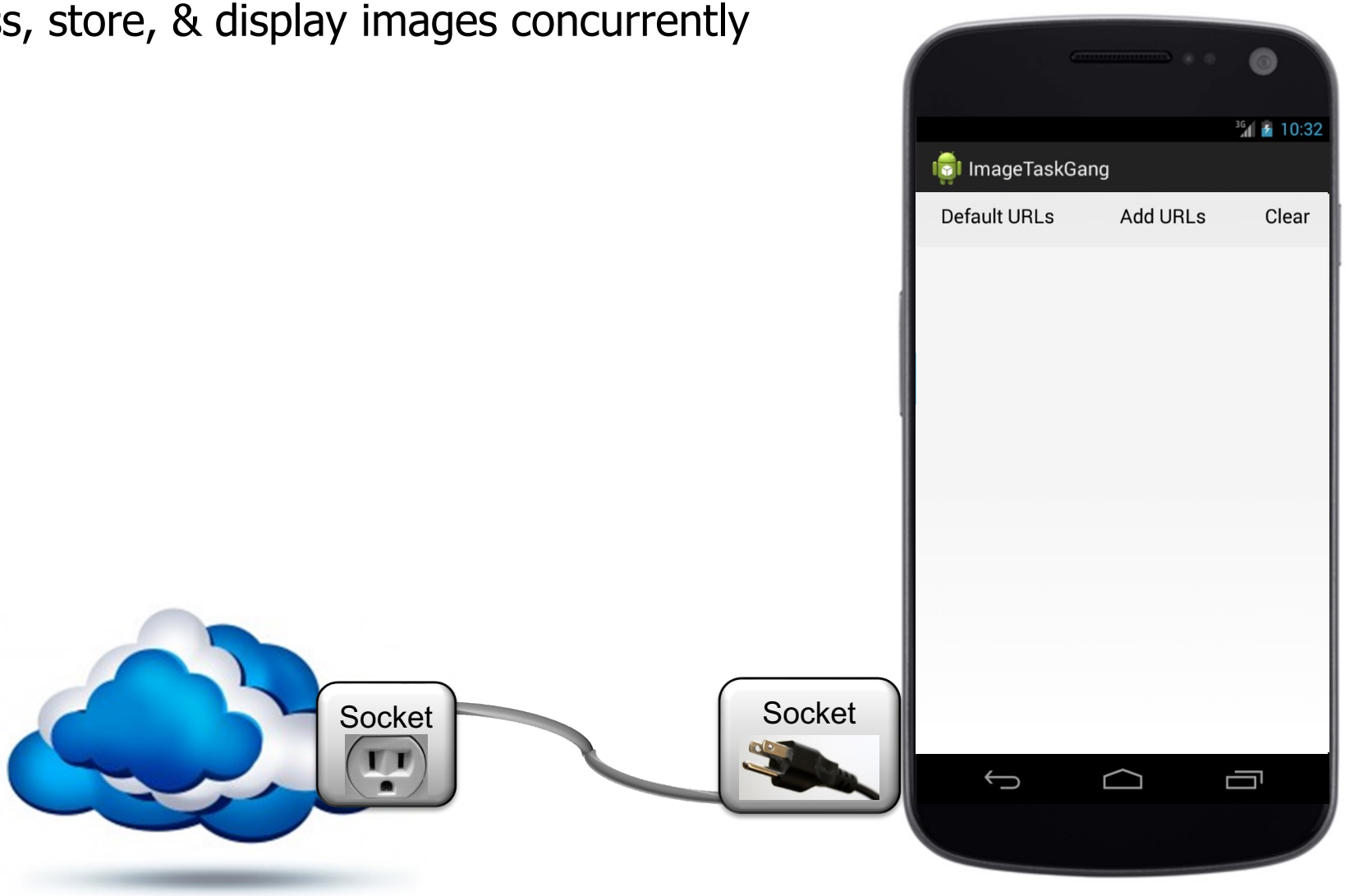
- Understand the ImageTaskGang application(s) & the patterns that are applied

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

Overview of the Pattern-Oriented ImageTaskGang

Overview of the Pattern-Oriented ImageTaskGang

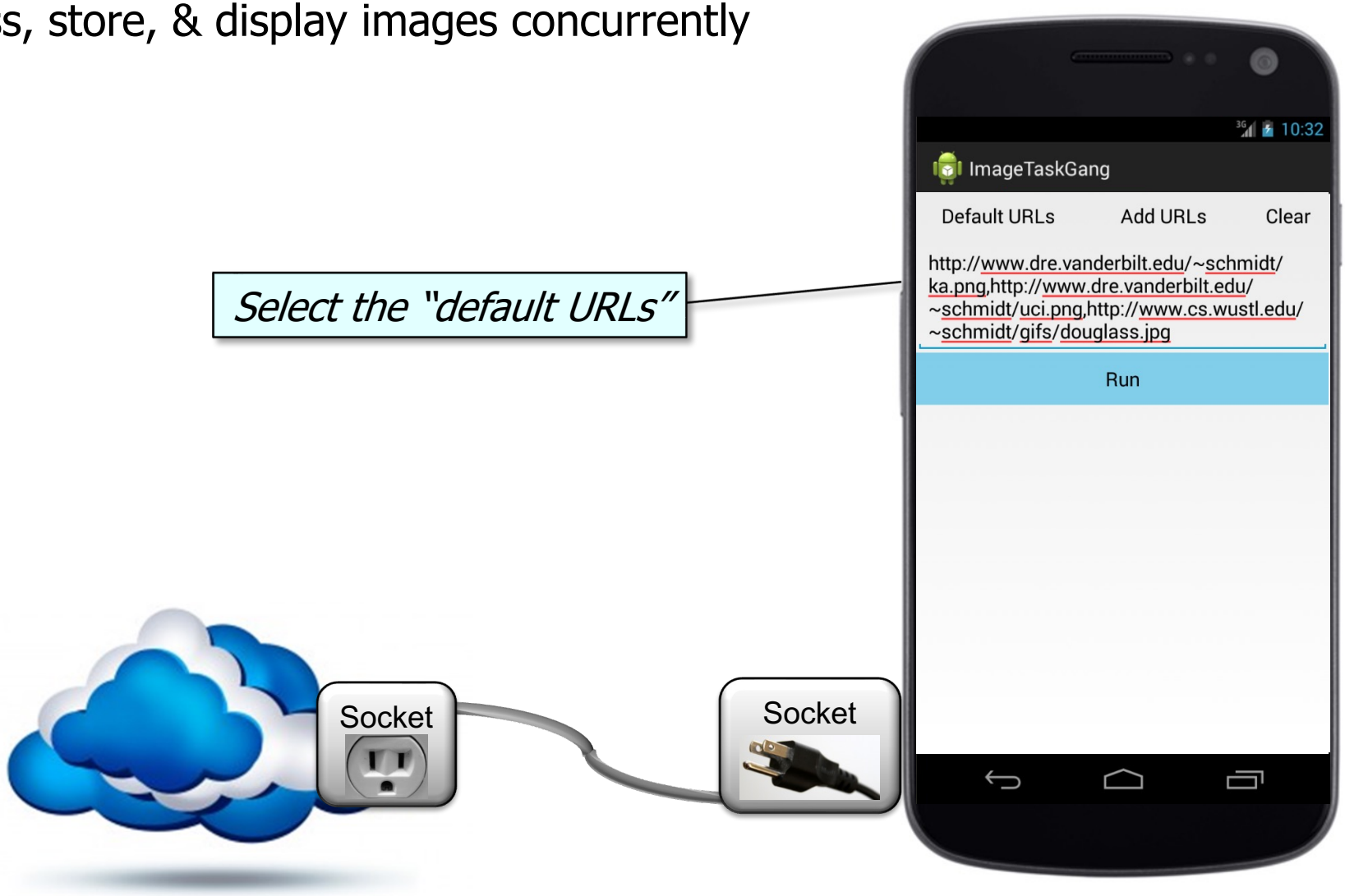
- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently



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

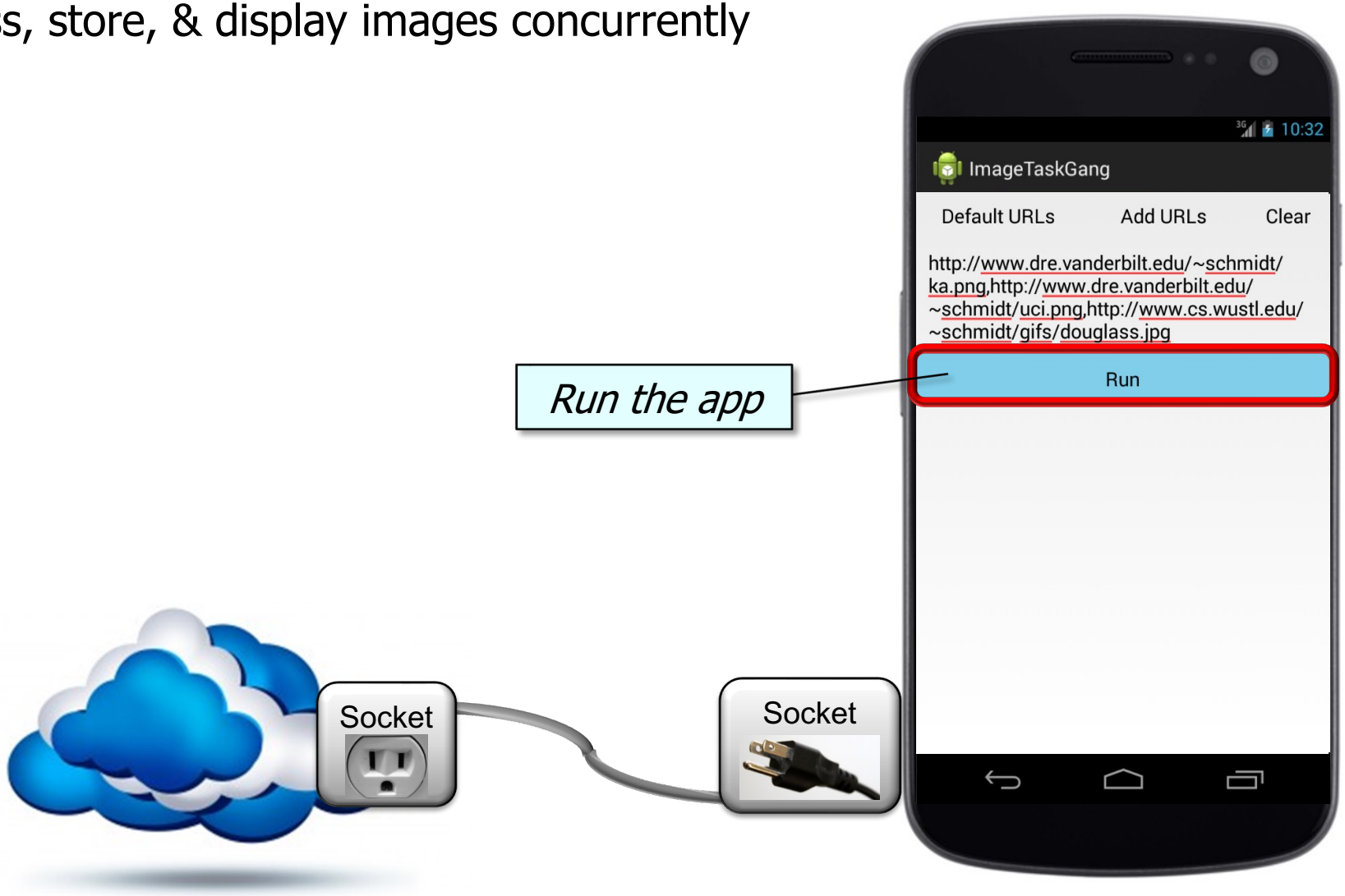
Overview of the Pattern-Oriented ImageTaskGang

- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently



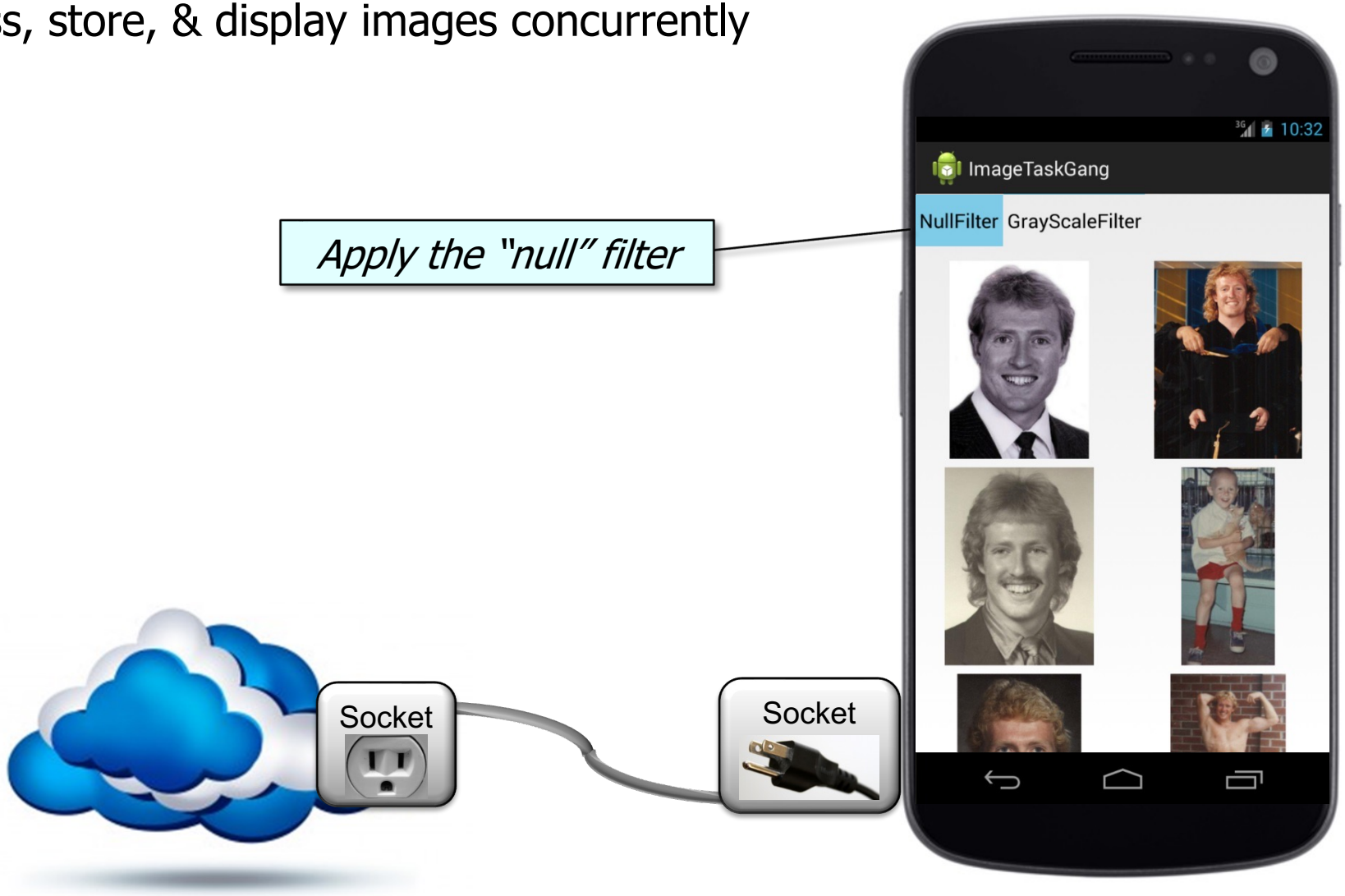
Overview of the Pattern-Oriented ImageTaskGang

- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently



Overview of the Pattern-Oriented ImageTaskGang

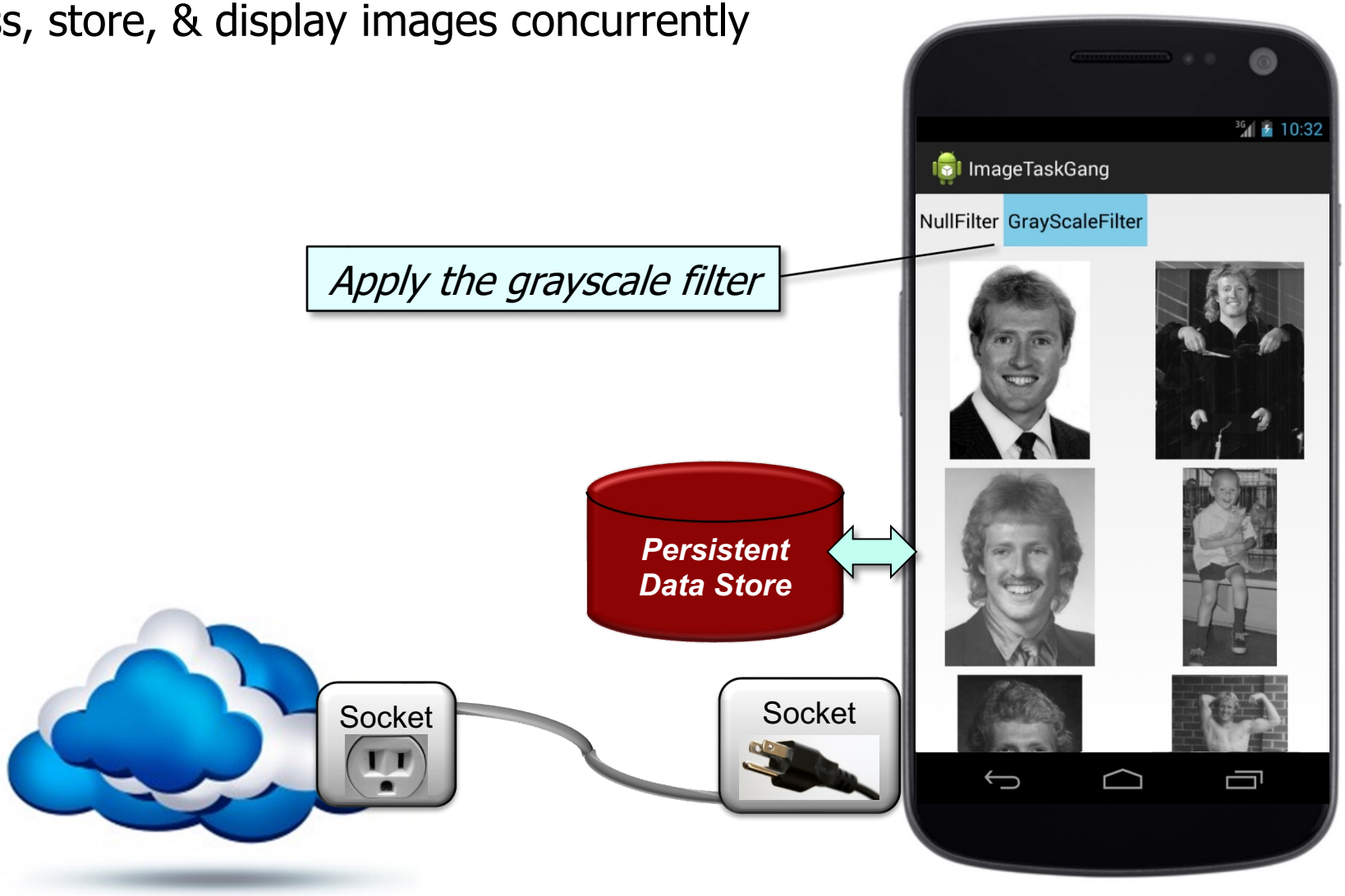
- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently



See en.wikipedia.org/wiki/NOP_(code)

Overview of the Pattern-Oriented ImageTaskGang

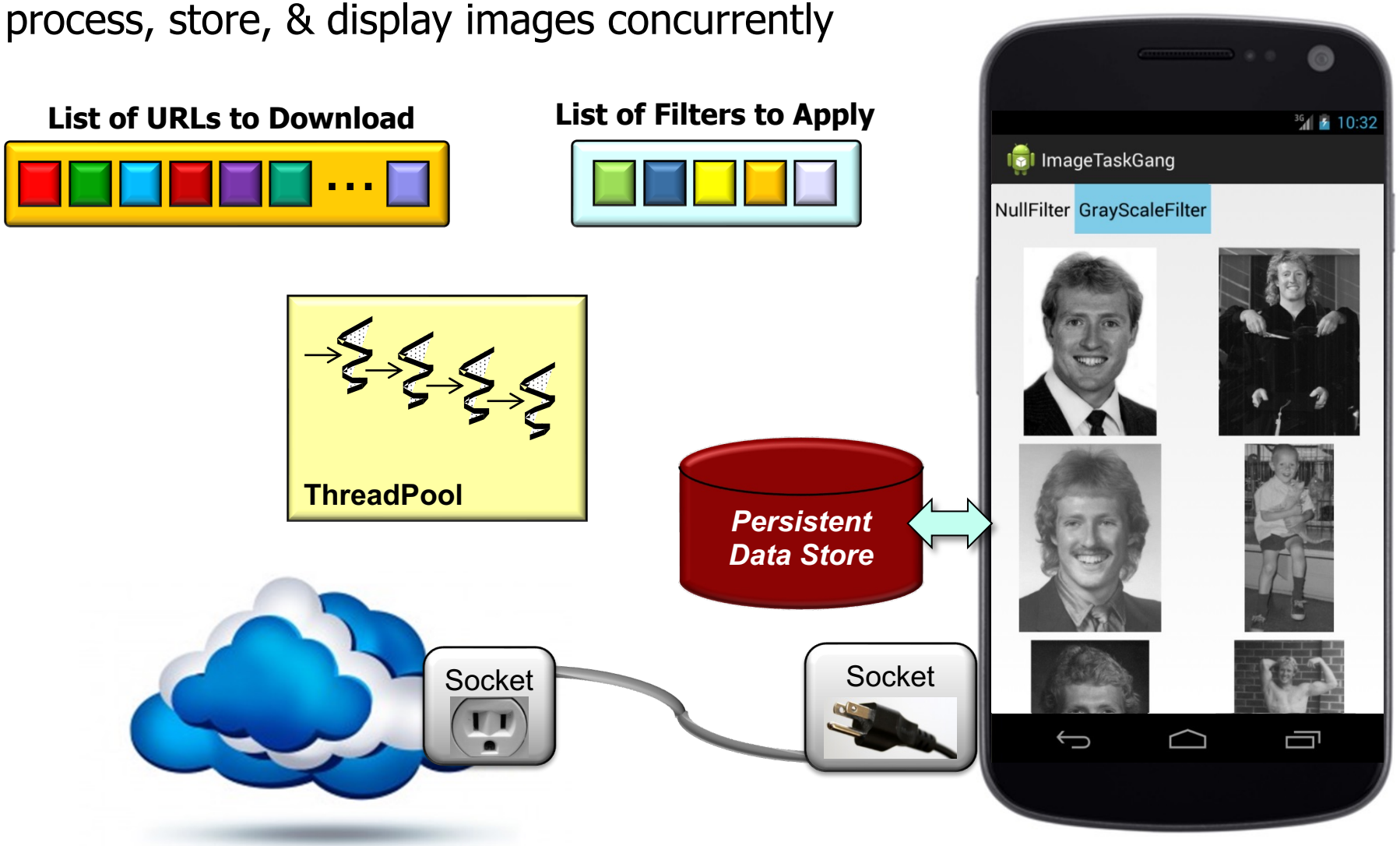
- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently



See en.wikipedia.org/wiki/Grayscale

Overview of the Pattern-Oriented ImageTaskGang

- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently

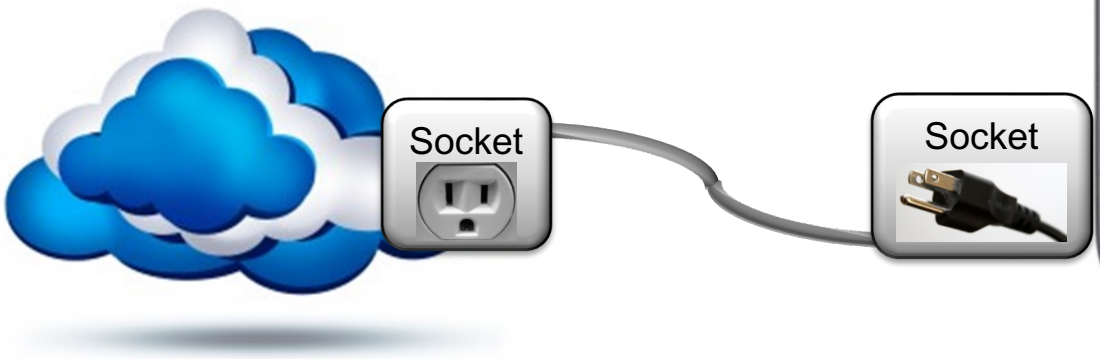


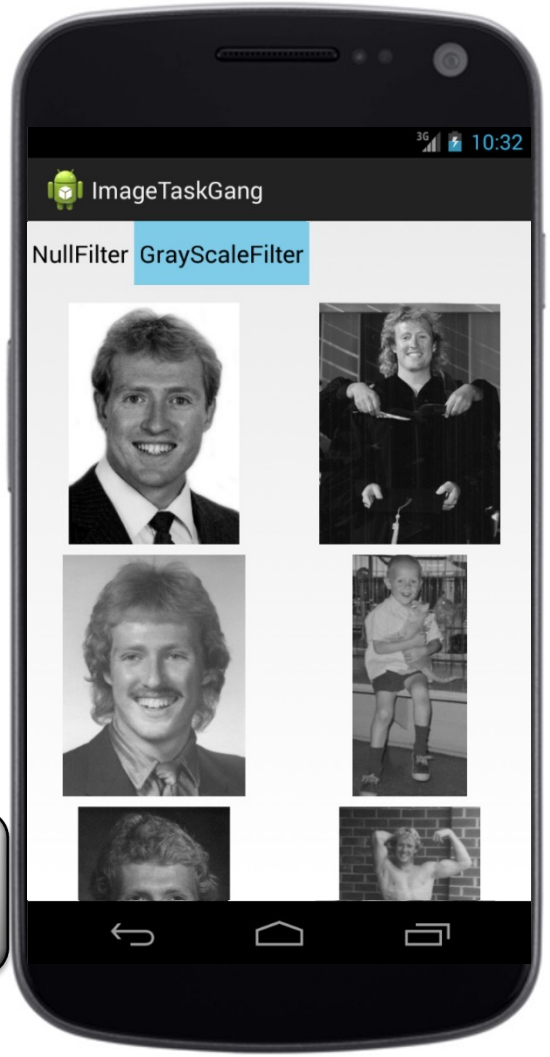
Various Java collections & thread pools are applied

Overview of the Pattern-Oriented ImageTaskGang

- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently

Strategy	Implementation
Executor model	Fixed or variable-size Thread pool
Unit of concurrency	Task per URL to download & image filter to apply
Results processing model	Asynchronous Future model for immediate concurrent processing



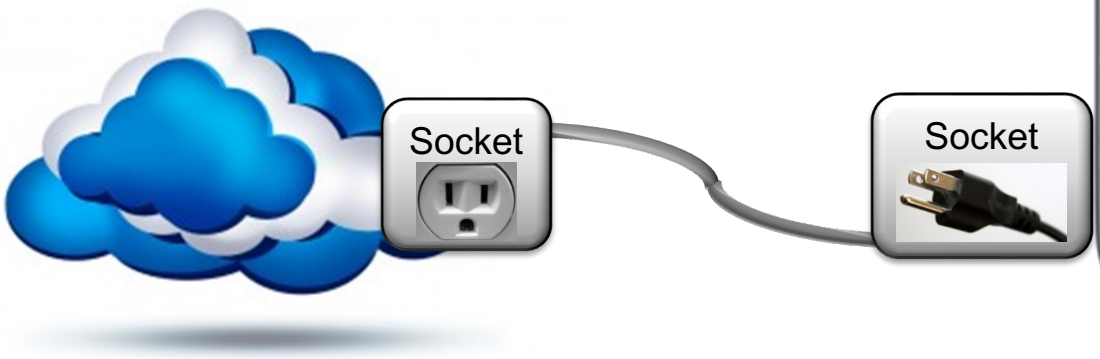


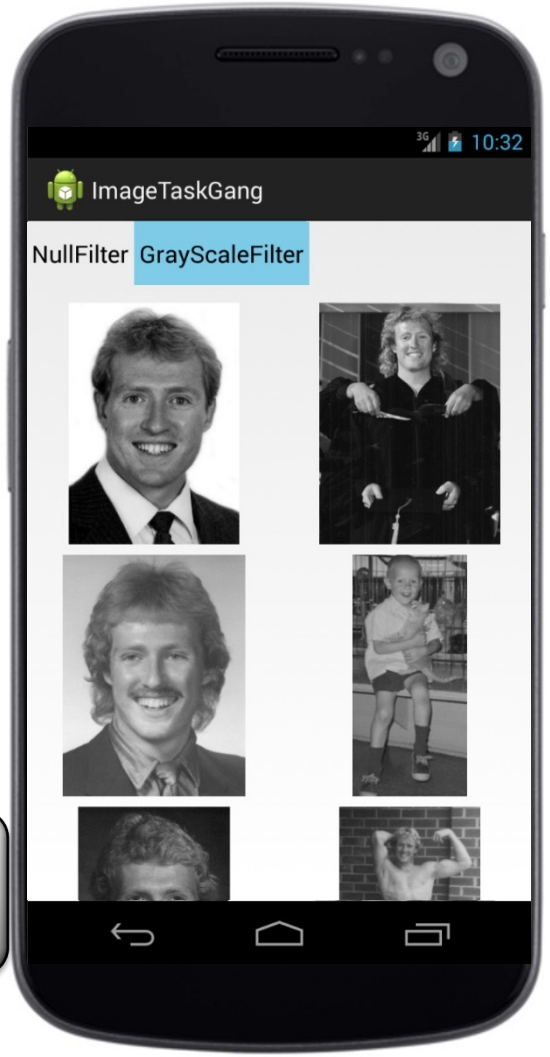
Various concurrency strategies are also applied

Overview of the Pattern-Oriented ImageTaskGang

- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently

Strategy	Implementation
Executor model	Fixed or variable-size Thread pool
Unit of concurrency	Task per URL to download & image filter to apply
Results processing model	Asynchronous Future model for immediate concurrent processing

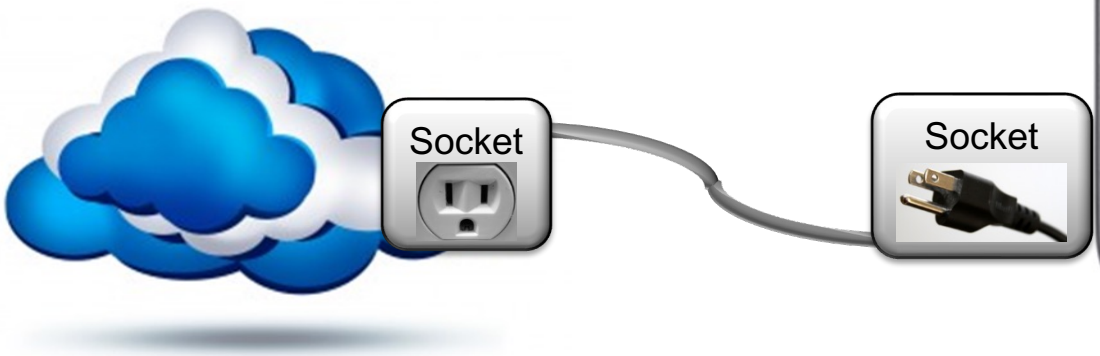


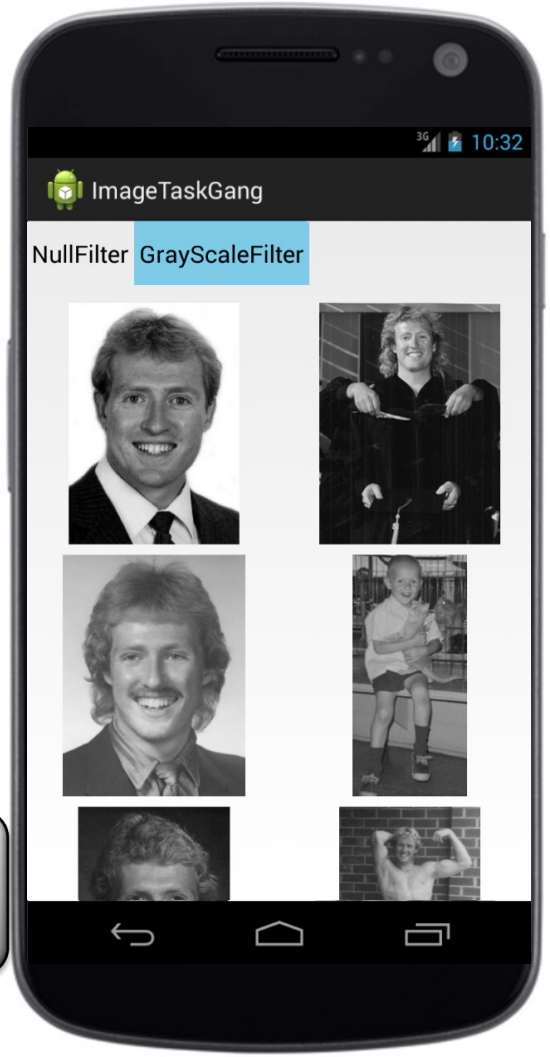


Overview of the Pattern-Oriented ImageTaskGang

- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently

Strategy	Implementation
Executor model	Fixed or variable-size Thread pool
Unit of concurrency	Task per URL to download & image filter to apply
Results processing model	Asynchronous Future model for immediate concurrent processing

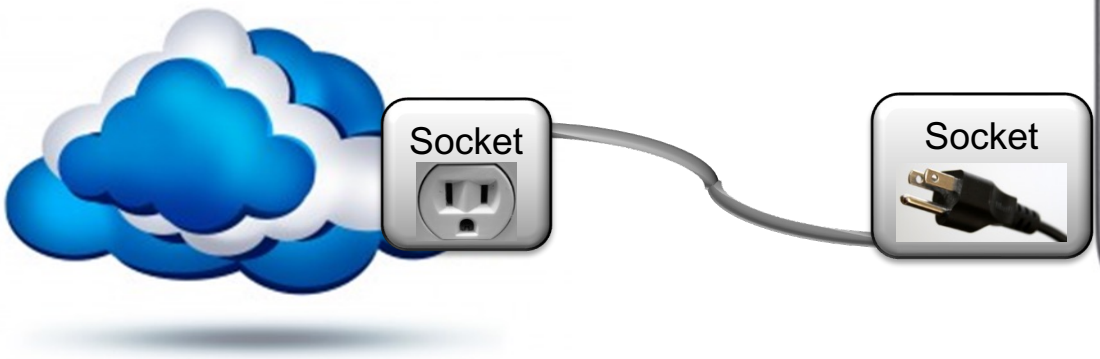


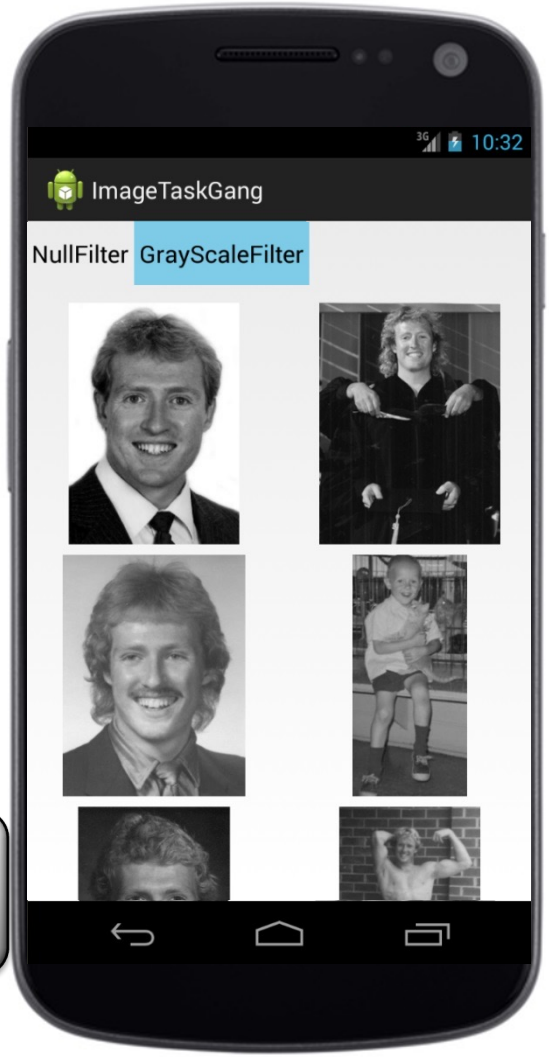


Overview of the Pattern-Oriented ImageTaskGang

- This app shows how the TaskGang framework can be customized to download, process, store, & display images concurrently

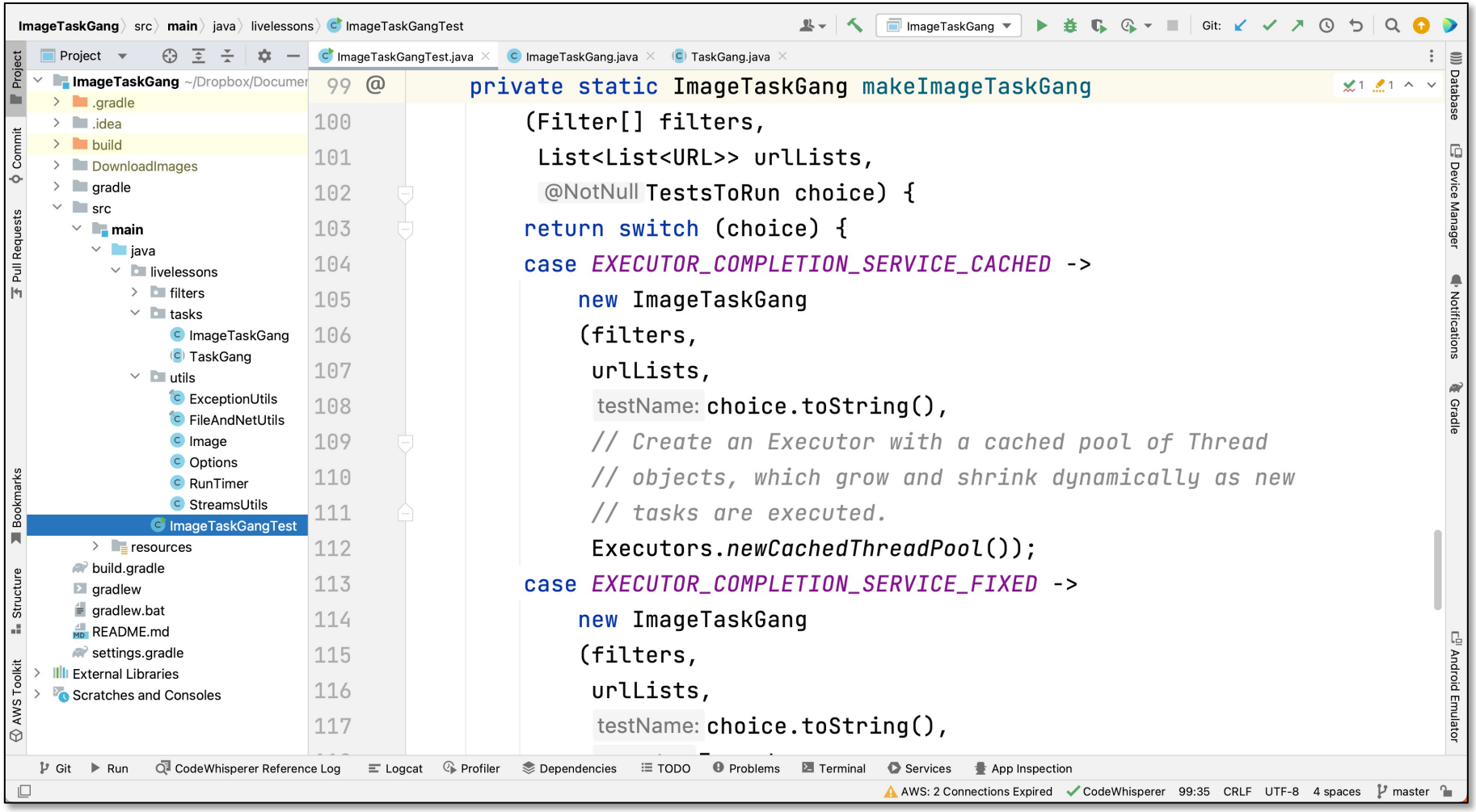
Strategy	Implementation
Executor model	Fixed or variable-size Thread pool
Unit of concurrency	Task per URL to download & image filter to apply
Results processing model	Asynchronous Future model for immediate concurrent processing





Overview of the Pattern-Oriented ImageTaskGang

- We're going to focus largely on the command-line version of ImageTaskGang since it can use more modern Java features than the Android version



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

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation

See en.wikipedia.org/wiki/Design_Patterns & www.dre.vanderbilt.edu/~schmidt/POSA

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*

See en.wikipedia.org/wiki/Proactor_pattern, en.wikipedia.org/wiki/Futures_and_promises, & kircher-schwanninger.de/michael/publications/Pooling.pdf

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*

The Proactor pattern separates the initiation of asynchronous operations from their handling, enabling efficient & scalable event-driven processing

See en.wikipedia.org/wiki/Proactor_pattern

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*

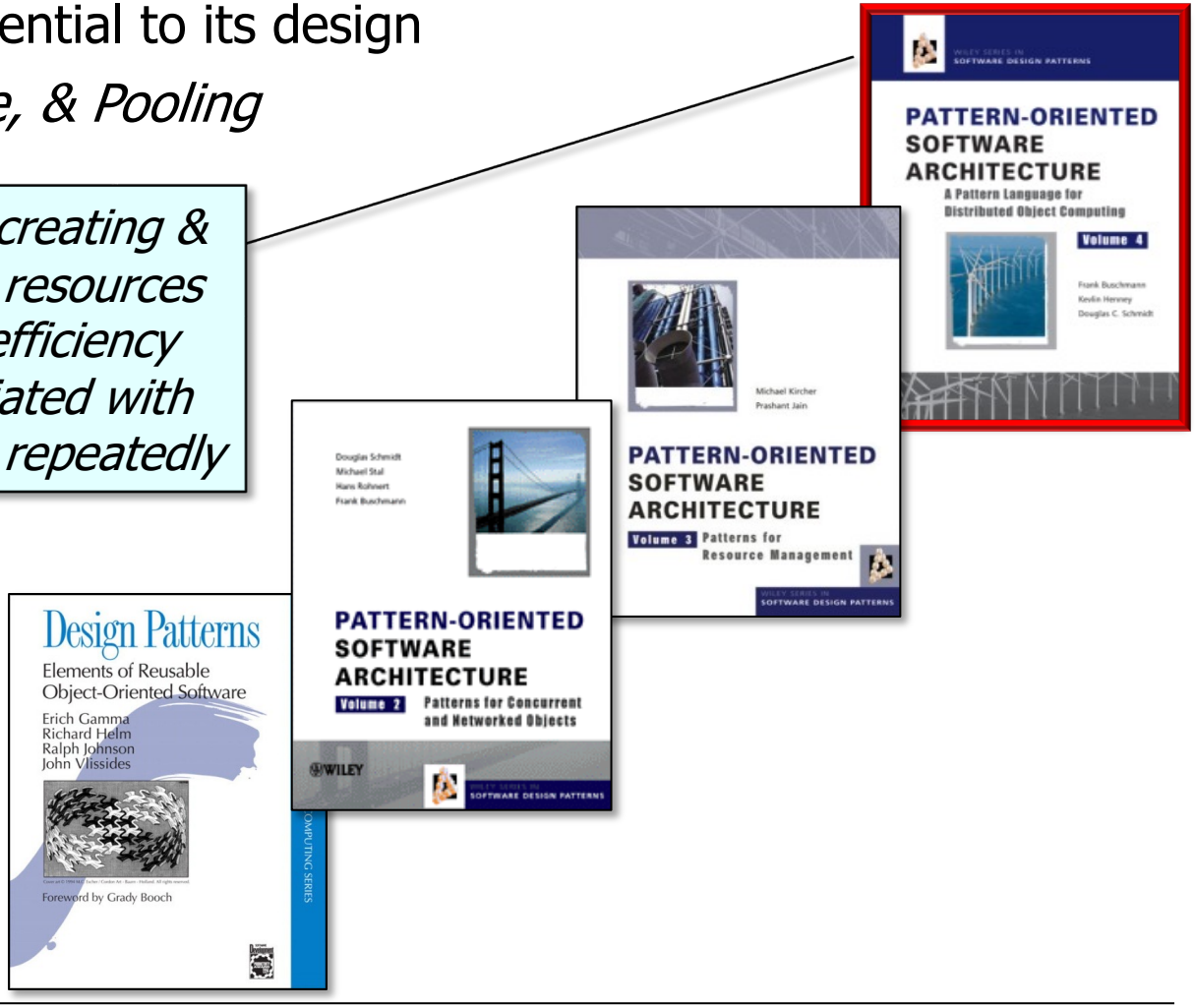
The Future pattern provides a proxy for a result that may not be available yet, allowing a program to continue execution without blocking until the result is needed

See en.wikipedia.org/wiki/Futures_and_promises

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*

The Pooling pattern involves creating & managing a pool of reusable resources to improve performance & efficiency by reducing overhead associated with creating/destroying resources repeatedly



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

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*
    - GoF: *Decorator*, *Factory Method*, & *Template Method*

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

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*
    - GoF: *Decorator*, *Factory Method*, & *Template Method*

The Decorator pattern allows behavior to be added to an object dynamically, providing flexible & alternative combinations of features without modifying the underlying object

See en.wikipedia.org/wiki/Decorator_pattern

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*
    - GoF: *Decorator*, *Factory Method*, & *Template Method*

The Factory Method pattern defines an interface for creating objects but allows implementations to decide which concrete class to instantiate, promoting loose coupling & extensibility

See en.wikipedia.org/wiki/Factory_method_pattern

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
    - POSA: *Proactor, Future, & Pooling*
    - GoF: *Decorator*, *Factory Method*, & *Template Method*

The Template Method pattern defines the skeleton of an algorithm in a base class, allowing subclasses to override certain steps while keeping the overall structure of the code intact, enabling more reuse & customization

See en.wikipedia.org/wiki/Template_method_pattern

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
  - 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 ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
  - The *Singleton* & *Command* patterns are also used in its implementation

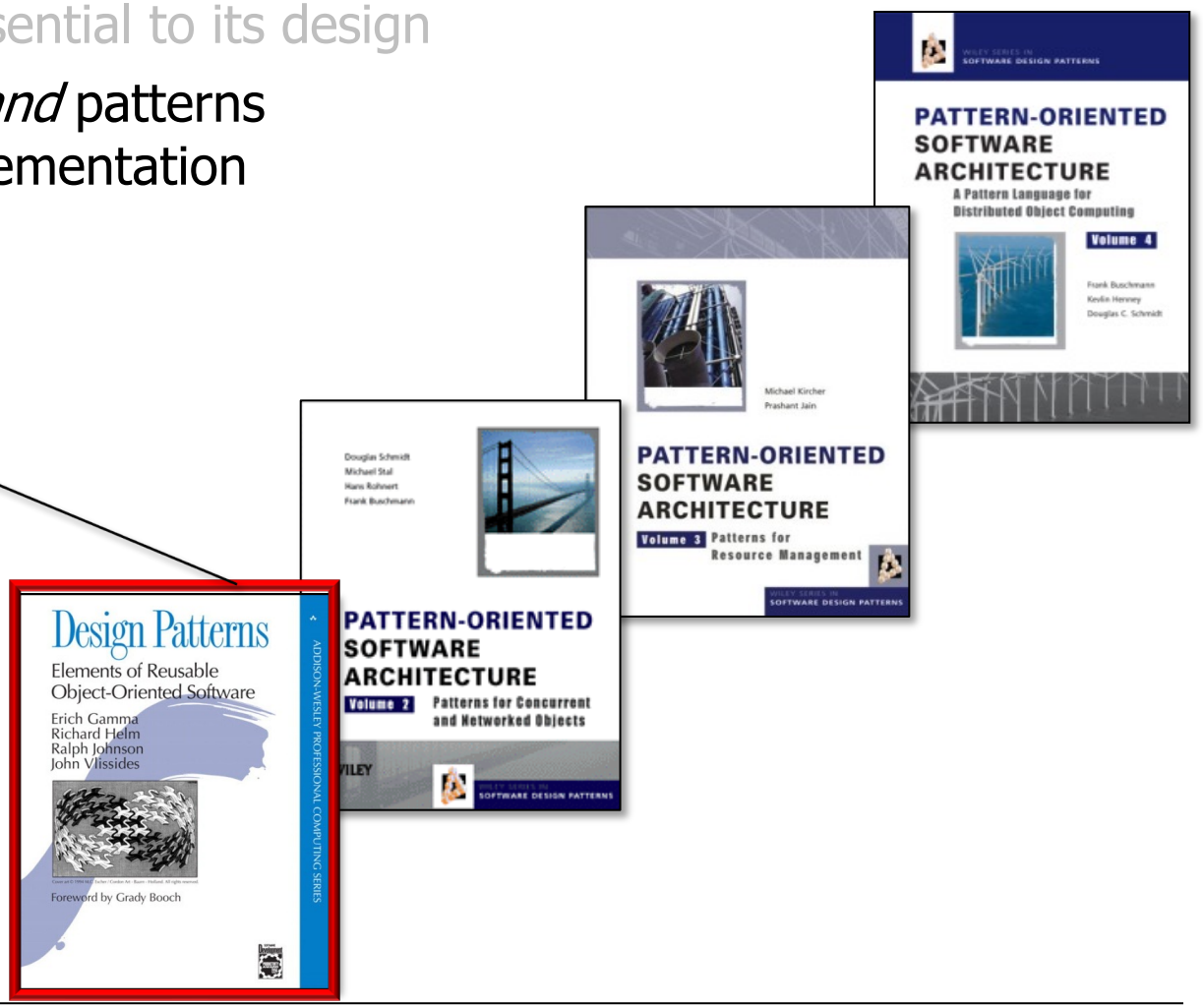
The Singleton pattern ensures that only one instance of a class is created & provides a global point of access to it, commonly used for scenarios where a single, shared resource needs to be managed

See en.wikipedia.org/wiki/Singleton_pattern

Overview of the Pattern-Oriented ImageTaskGang

- Several "Gang-of-Four" & POSA patterns are applied to enhance the framework-based ImageTaskGang implementation
  - These patterns most essential to its design
  - The *Singleton* & *Command* patterns are also used in its implementation

The Command pattern encapsulates a request as an object, allowing the parameterization of clients with different requests, queuing or logging requests, & supporting operations like undo & redo



See en.wikipedia.org/wiki/Command_pattern

End of the Overview of the ImageTaskGang Application