Overview of the Java Completable Futures Framework

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

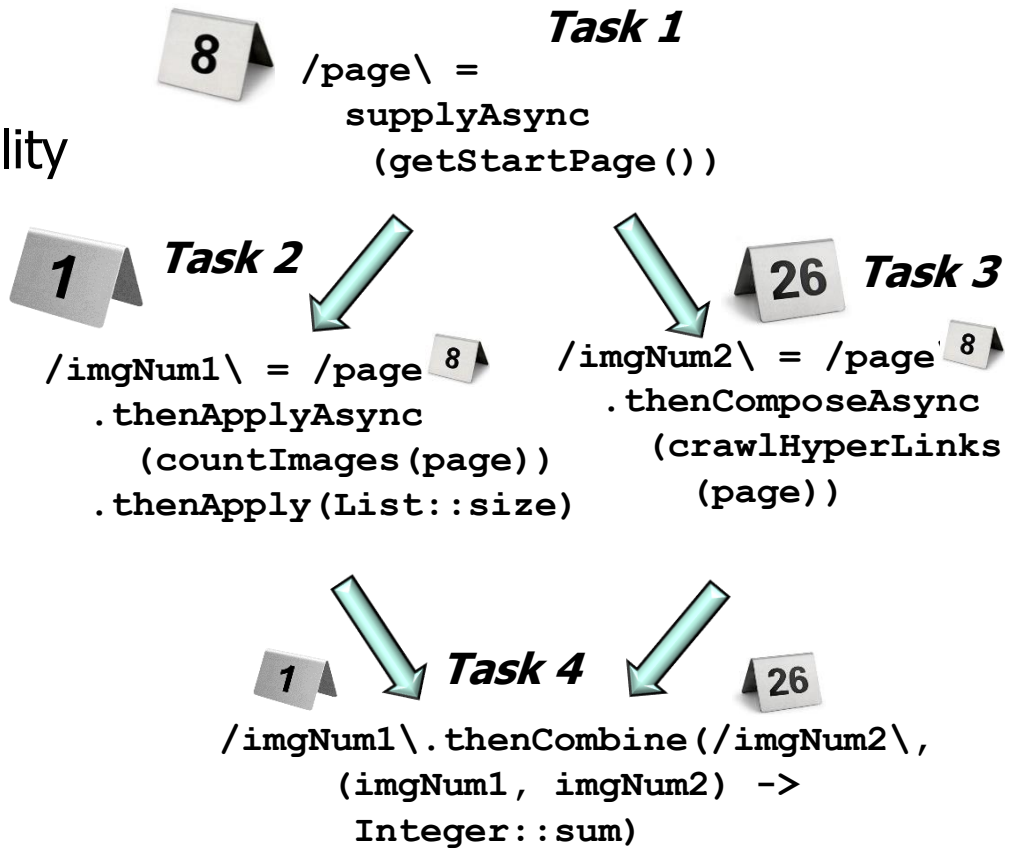
Professor of Computer Science

Institute for Software Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Recognize the key principles underlying reactive programming
- Be aware of structure & functionality of the Java completable futures framework



Overview of the Java Completable Futures Framework

Overview of Completable Futures

- Java's completable future framework provides an asynchronous & reactive concurrent programming model

Class CompletableFuture<T>

java.lang.Object
    java.util.concurrent.CompletableFuture<T>

All Implemented Interfaces:

`CompletionStage<T>, Future<T>`

```
public class CompletableFuture<T>
extends Object
implements Future<T>, CompletionStage<T>
```

A `Future` that may be explicitly completed (setting its value and status), and may be used as a `CompletionStage`, supporting dependent functions and actions that trigger upon its completion.

When two or more threads attempt to `complete`, `completeExceptionally`, or `cancel` a CompletableFuture, only one of them succeeds.

In addition to these and related methods for directly manipulating status and results, CompletableFuture implements interface `CompletionStage` with the following policies:

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletableFuture.html

Overview of Completable Futures

- Java's completable future framework provides an asynchronous & reactive concurrent programming model
  - Supports dependent actions that trigger upon completion of async operations

These dependencies can be modeled via a data flow diagram

***Task 1**: Get start page asynchronously*

***Task 2**: Count images on the page asynchronously*

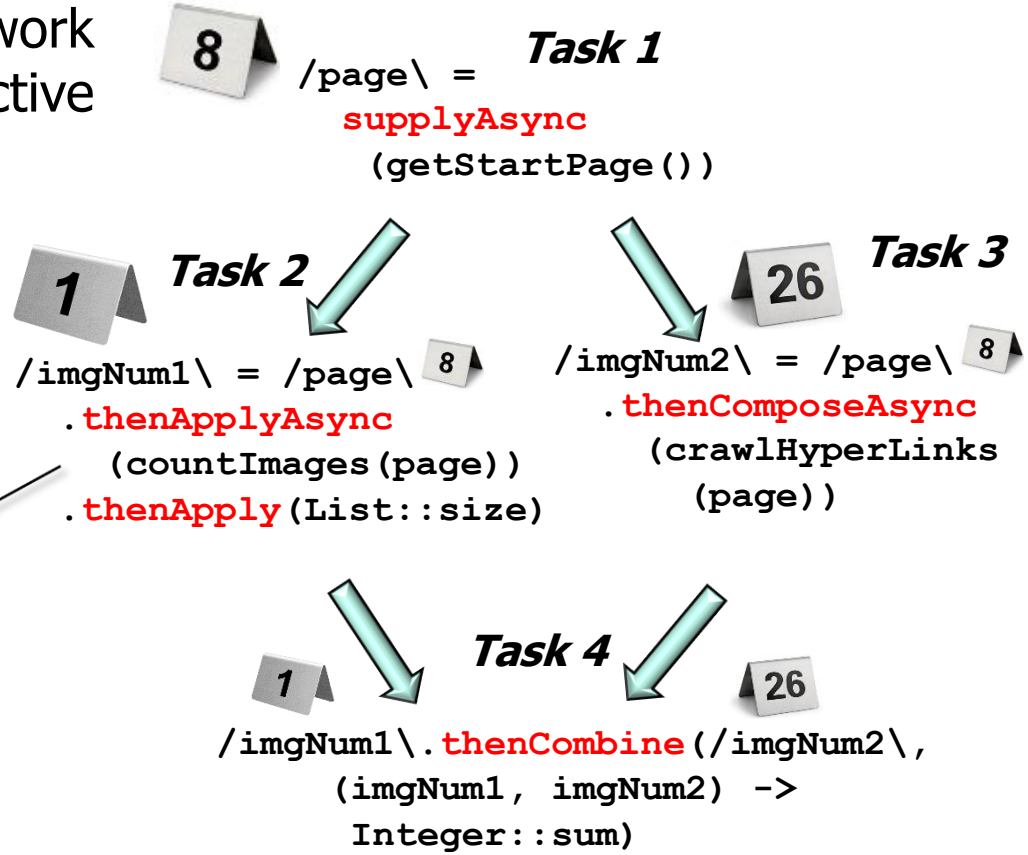
***Task 3**: Count images on all hyperlinked pages asynchronously*

***Task 4**: Combine results to create the total asynchronously*

See en.wikipedia.org/wiki/Web_crawler

Overview of Completable Futures

- Java's completable future framework provides an asynchronous & reactive concurrent programming model
  - Supports dependent actions that trigger upon completion of async operations

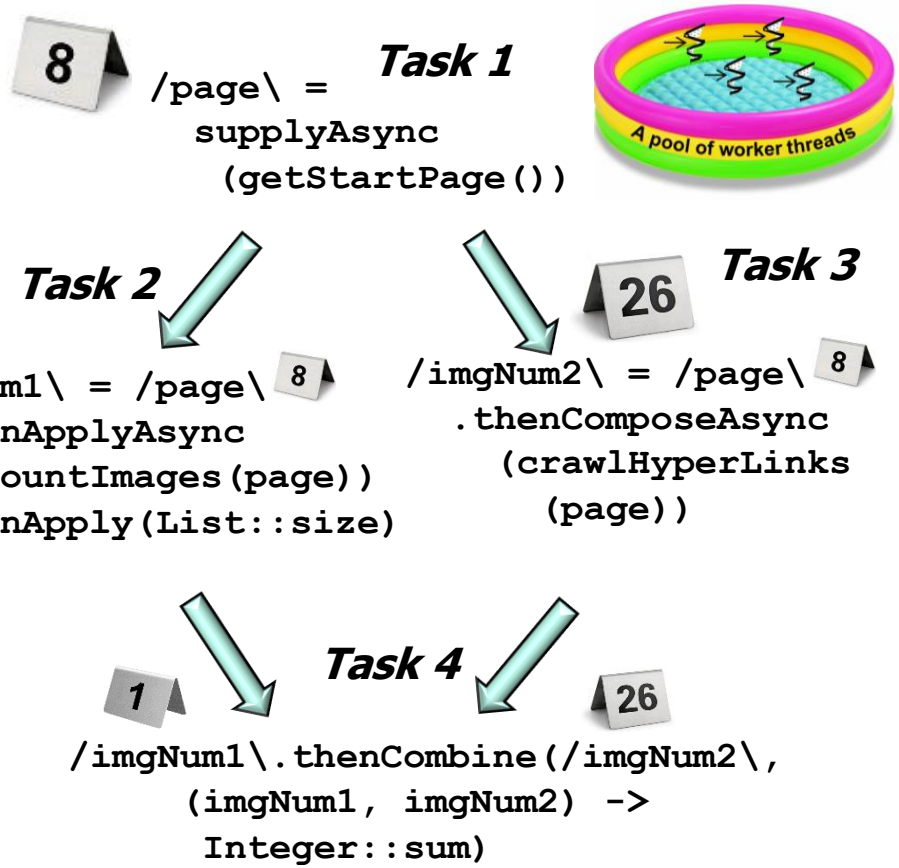


Async operations can be forked, chained, & joined

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletionStage.html

Overview of Completable Futures

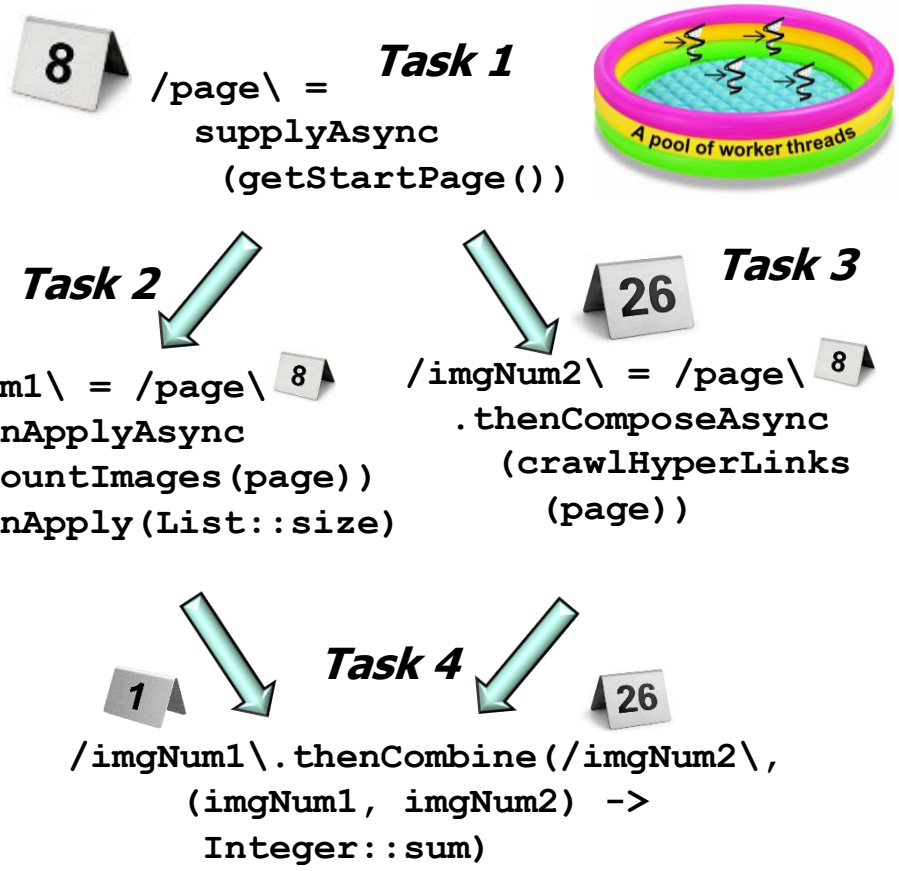
- Java's completable future framework provides an asynchronous & reactive concurrent programming model
  - Supports dependent actions that trigger upon completion of async operations
  - Async operations can run concurrently in thread pools



See www.nurkiewicz.com/2013/05/java-8-definitive-guide-to.html

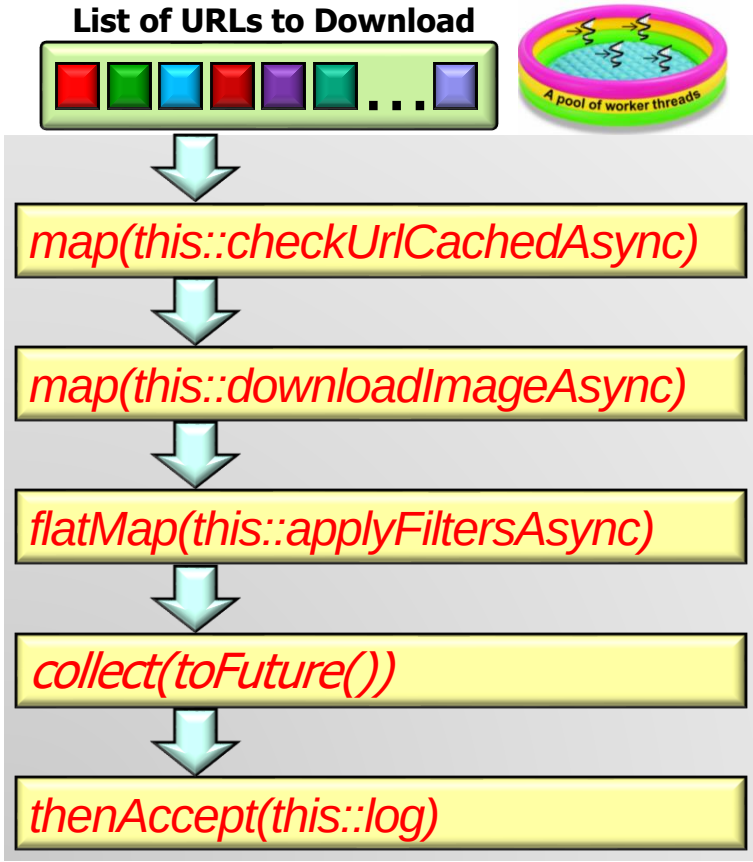
Overview of Completable Futures

- Java's completable future framework provides an asynchronous & reactive concurrent programming model
  - Supports dependent actions that trigger upon completion of async operations
  - Async operations can run concurrently in thread pools
    - Either a (common) fork-join pool or various types of pre- or user-defined thread pools



Overview of Completable Futures

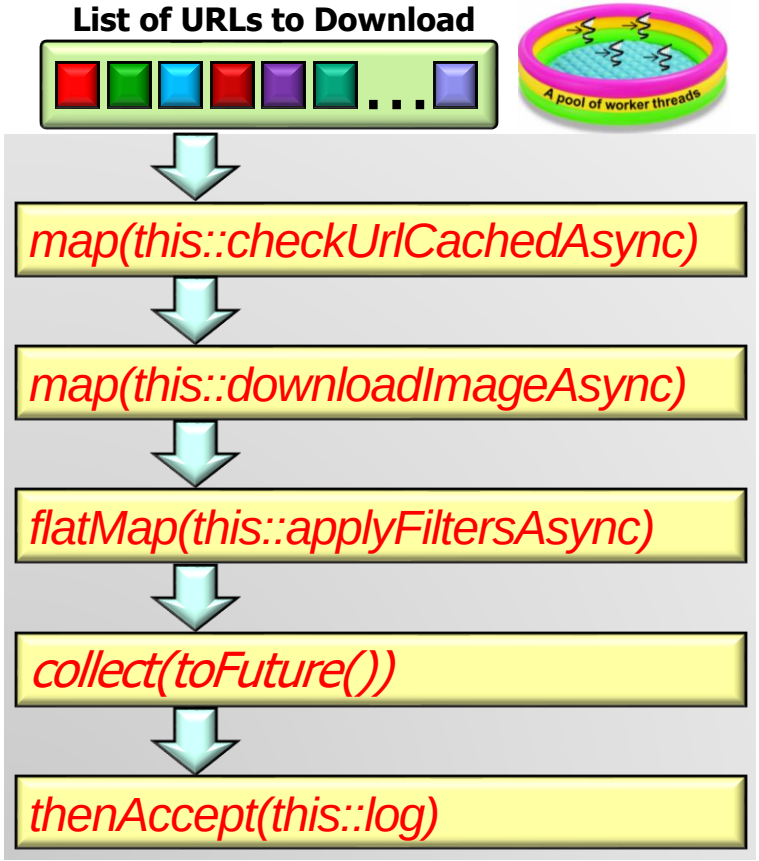
- Java completable futures, sequential streams, & functional programming features can be combined nicely!!



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

Overview of Completable Futures

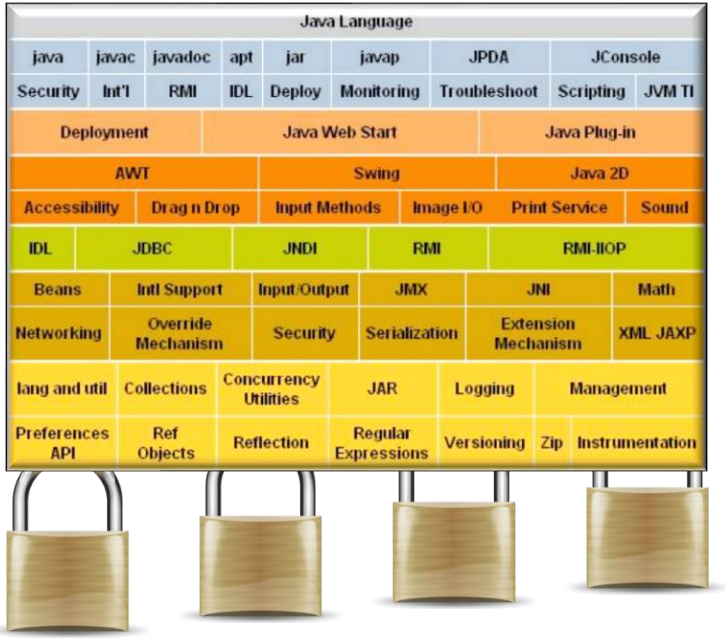
- Java completable futures often need no explicit synchronization or threading when developing concurrent apps!

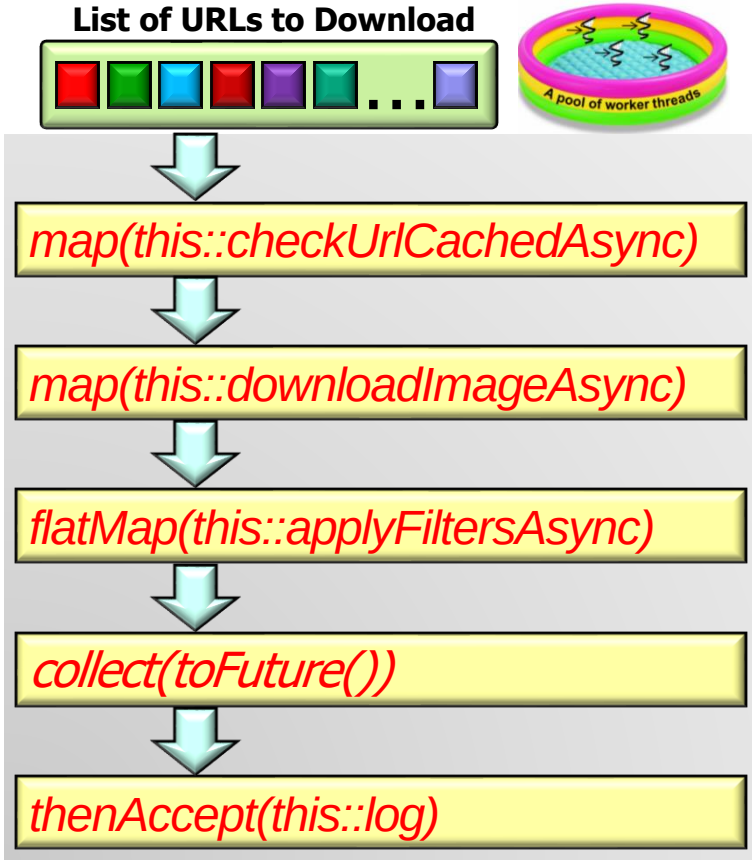


Alleviates many accidental & inherent complexities of concurrent programming

Overview of Completable Futures

- Java completable futures often need no explicit synchronization or threading when developing concurrent apps!





Java class libraries handle locking needed to protect shared mutable state

End of Overview of the Java Completable Futures Framework