Understanding Method Groupings in the Java Completable Futures API

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 how Java completable futures overcome limitations with Java futures
- Understand how methods are grouped in the Java completable future API
Grouping the Java Completable Future API
The entire completable future framework resides in 1 public class with 60+ methods!!!
This framework implements the Façade pattern. It provides a unified interface to a set of interfaces in a sub-system that makes the subsystem easier to use. See [en.wikipedia.org/wiki/Facade_pattern](en.wikipedia.org/wiki/Facade_pattern)
Given the large # of methods in this API it helps to have a “birds-eye” view

See en.wikipedia.org/wiki/Earthrise
• Some completable future features are basic
Some completable future features are basic
- e.g., the Java Future API + some simple enhancements

Only slightly better than the conventional Future interface
Grouping the Java Completable Future API

- Other completable future features are more advanced

Completion stage methods

Exception methods

Arbitrary-arity methods

Factory methods

Basic methods
Grouping the Java Completable Future API

- Other completable future features are more advanced
- Factory methods

See [en.wikipedia.org/wiki/Factory_method_pattern](en.wikipedia.org/wiki/Factory_method_pattern)
Other completable future features are more advanced

Factory methods

- Initiate async two-way or one-way computations without using threads explicitly
Grouping the Java Completable Future API

- Other completable future features are more advanced
- Factory methods
  - Initiate async two-way or one-way computations without using threads explicitly

Help make programs more *elastic* by leveraging a pool of worker threads
Grouping the Java Completable Future API

- Other completable future features are more advanced
  - Factory methods
  - Completion stage methods

Completion stage methods

- Exception methods
- Arbitrary-arity methods
- Factory methods
- Basic methods
Grouping the Java Completable Future API

- Other completable future features are more advanced
  - Factory methods
  - Completion stage methods
    - Chain together actions that perform async result processing & composition

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletionStage.html
Grouping the Java Completable Future API

- Other completable future features are more advanced
  - Factory methods
  - Completion stage methods
    - Chain together actions that perform async result processing & composition

Help make programs more **responsive** by not blocking caller code
Other completable future features are more advanced
- Factory methods
- Completion stage methods
- “Arbitrary-arity” methods

See [en.wikipedia.org/wiki/Arity](en.wikipedia.org/wiki/Arity)
Grouping the Java Completable Future API

- Other completable future features are more advanced
  - Factory methods
  - Completion stage methods
  - "Arbitrary-arity" methods
- Process futures in bulk by combine multiple futures into a single future
Grouping the Java Completable Future API

- Other completable future features are more advanced
  - Factory methods
  - Completion stage methods
  - “Arbitrary-arity” methods
  - Process futures in bulk by combine multiple futures into a single future

Help make programs more *responsive* by not blocking caller code
Grouping the Java Completable Future API

- Other completable future features are more advanced
  - Factory methods
  - Completion stage methods
  - “Arbitrary-arity” methods
  - Exception methods
Other completable future features are more advanced:
- Factory methods
- Completion stage methods
- “Arbitrary-arity” methods
- Exception methods
  - Handle exceptional conditions at runtime
Grouping the Java Completable Future API

- Other completable future features are more advanced
  - Factory methods
  - Completion stage methods
  - "Arbitrary-arity" methods
- Exception methods
  - Handle exceptional conditions at runtime

Help make programs more *resilient* by handling erroneous computations gracefully
All methods are implemented internally via message-passing that’s ultimately connected to Java-based thread pools.

Ensures loose coupling, isolation, & location transparency between components.
End of Understanding Method Groupings in the Java Completable Futures API