

Advanced Java Completable Future Features: Grouping Completion Stage Methods

Douglas C. Schmidt

d.schmidt@vanderbilt.edu

www.dre.vanderbilt.edu/~schmidt



Professor of Computer Science

**Institute for Software
Integrated Systems**

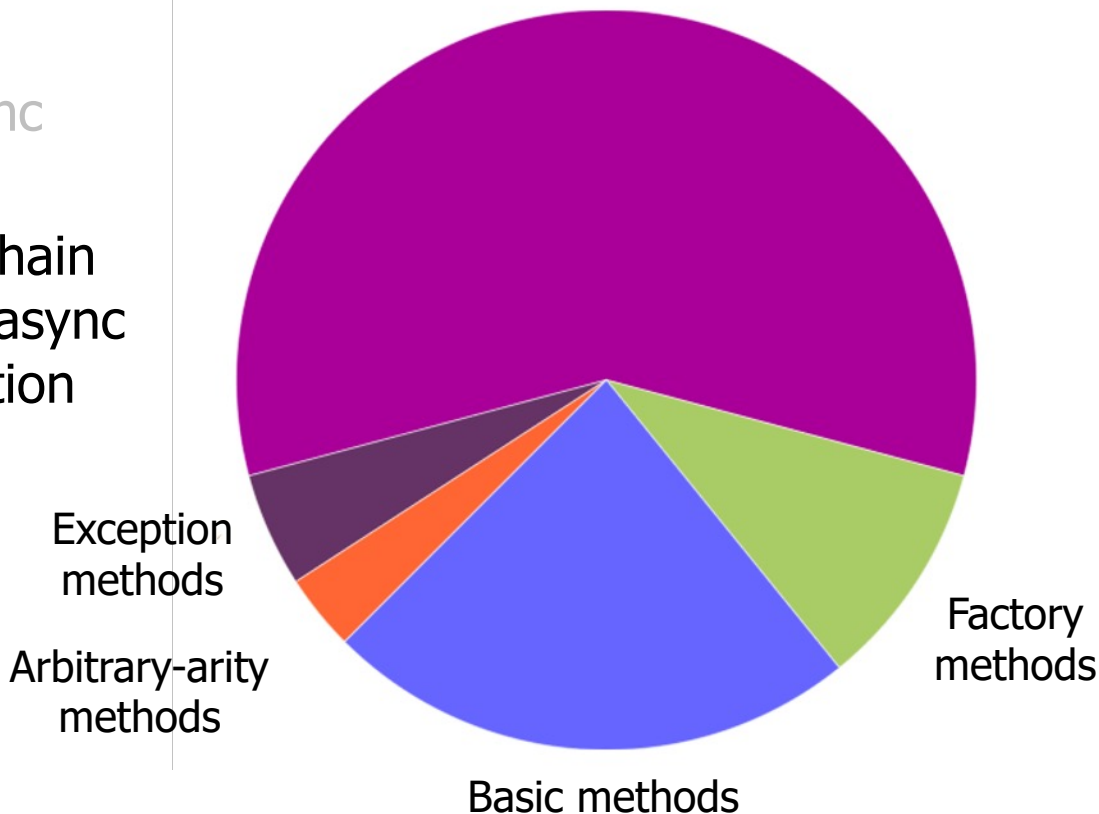
**Vanderbilt University
Nashville, Tennessee, USA**



Learning Objectives in this Part of the Lesson

- Understand advanced features of completable futures, e.g.
 - Factory methods initiate async computations
 - Completion stage methods chain together actions to perform async result processing & composition
 - Method grouping
 - i.e., group methods by how one stage can be triggered by one or more previous stage(s)

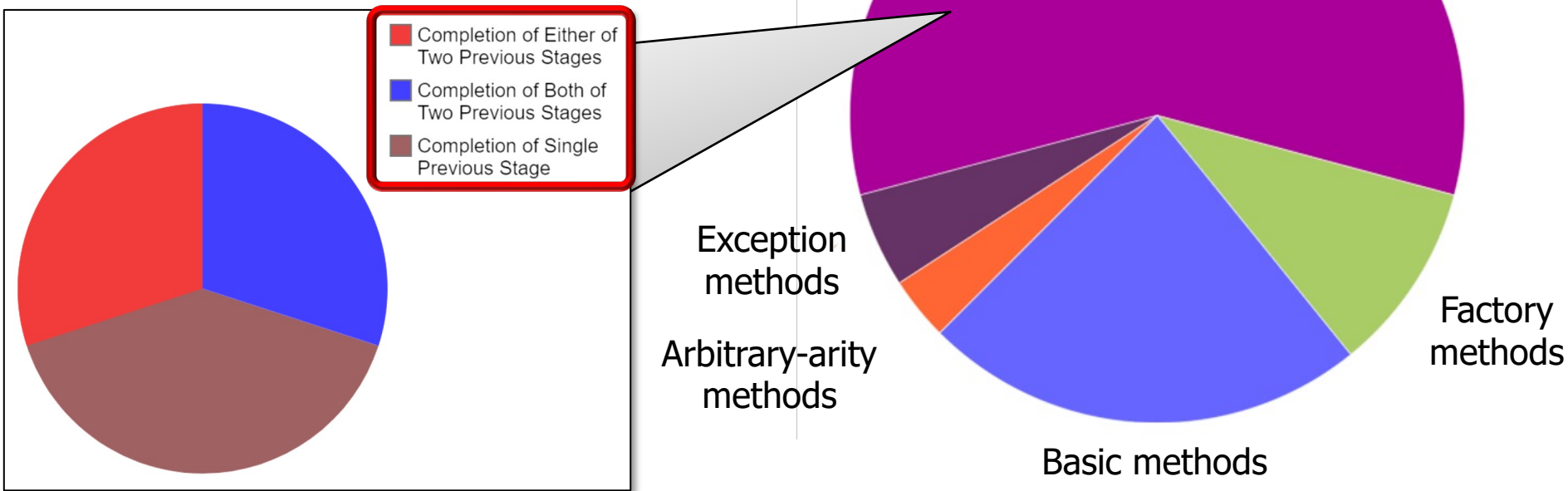
Completion stage methods



Grouping CompletableFuture Completion Stage Methods

Grouping CompletableFuture Completion Stage Methods

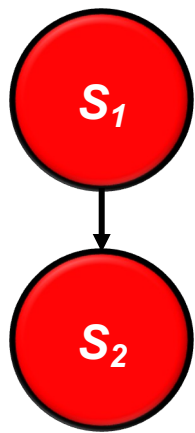
- Completion stage methods are grouped based on how a stage is triggered by one or more previous stage(s)



See www.jesperdj.com/2015/09/26/the-future-is-completable-in-java-8

Grouping CompletableFuture Completion Stage Methods

- Completion stage methods are grouped based on how a stage is triggered by one or more previous stage(s)
- Completion of a single previous stage



Methods	Params	Returns	Behavior
<code>thenApply</code> (<code>Async</code>)	<code>Function</code>	<code>CompletableFuture</code> with <code>Function</code> result	Apply function to result of the previous stage
<code>thenCompose</code> (<code>Async</code>)	<code>Function</code>	<code>CompletableFuture</code> with <code>Function</code> result directly, <i>not</i> a nested future	Apply function to result of the previous stage
<code>thenAccept</code> (<code>Async</code>)	<code>Consumer</code>	<code>CompletableFuture<Void></code>	Consumer handles result of previous stage
<code>thenRun</code> (<code>Async</code>)	<code>Runnable</code>	<code>CompletableFuture<Void></code>	Run action w/out returning value

Grouping CompletableFuture Completion Stage Methods

- Completion stage methods are grouped based on how a stage is triggered by one or more previous stage(s)
 - Completion of a single previous stage

These methods run in the invoking thread or the same thread as previous stage

Methods	Params	Returns	Behavior
<code>thenApply</code> (<code>Async</code>)	<code>Function</code>	<code>CompletableFuture</code> with <code>Function</code> result	Apply function to result of the previous stage
<code>thenCompose</code> (<code>Async</code>)	<code>Function</code>	<code>CompletableFuture</code> with <code>Function</code> result directly, <i>not</i> a nested future	Apply function to result of the previous stage
<code>thenAccept</code> (<code>Async</code>)	<code>Consumer</code>	<code>CompletableFuture<Void></code>	Consumer handles result of previous stage
<code>thenRun</code> (<code>Async</code>)	<code>Runnable</code>	<code>CompletableFuture<Void></code>	Run action w/out returning value

The thread that executes these methods depends on various runtime factors

Grouping CompletableFuture Completion Stage Methods

- Completion stage methods are grouped based on how a stage is triggered by one or more previous stage(s)
- Completion of a single previous stage

**Async() variants run in some thread pool*

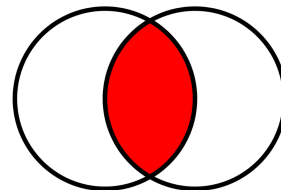
Methods	Params	Returns	Behavior
<code>thenApply</code> (<code>Async</code>)	Function	Completable Future with Function result	Apply function to result of the previous stage
<code>thenCompose</code> (<code>Async</code>)	Function	Completable Future with Function result directly, <i>not</i> a nested future	Apply function to result of the previous stage
<code>thenAccept</code> (<code>Async</code>)	Consumer	Completable Future<Void>	Consumer handles result of previous stage
<code>thenRun</code> (<code>Async</code>)	Runnable	Completable Future<Void>	Run action w/out returning value

See blog.krean.net/2013/12/25/completablefutures-why-to-use-async-methods

Grouping CompletableFuture Completion Stage Methods

- Completion stage methods are grouped based on how a stage is triggered by one or more previous stage(s)
 - Completion of a single previous stage
 - Completion of both of two previous stages
 - i.e., an “and”

Methods	Params	Returns	Behavior
<code>thenCombine</code> (Async)	Bi Function	Completable Future with Bi Function result	Apply bifunction to results of both previous stages
<code>thenAcceptBoth</code> (Async)	Bi Consumer	Completable Future<Void>	BiConsumer handles results of both previous stages
<code>runAfterBoth</code> (Async)	Runnable	Completable Future<Void>	Run action when both previous stages complete

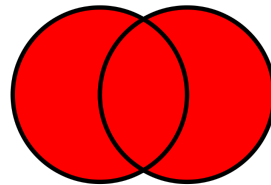


See en.wikipedia.org/wiki/Logical_conjunction

Grouping CompletableFuture Completion Stage Methods

- Completion stage methods are grouped based on how a stage is triggered by one or more previous stage(s)
 - Completion of a single previous stage
 - Completion of both of two previous stages
 - Completion of either of two previous stages
 - i.e., an “or”

Methods	Params	Returns	Behavior
<code>applyToEither</code> (Async)	Function	Completable Future with Function result	Apply function to results of either previous stage
<code>acceptEither</code> (Async)	Consumer	Completable Future<Void>	Consumer handles results of either previous stage
<code>runAfterEither</code> (Async)	Runnable	Completable Future<Void>	Run action when either previous stage completes



See en.wikipedia.org/wiki/Logical_disjunction

End of Advanced Java

Completable Future Features:

Grouping Completion

Stage Methods
