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
  - Arbitrary-arity methods that process futures in bulk

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletableFuture.html
Arbitrary-Arity Methods
Process Futures in Bulk
### Arbitrary-Arity Methods Process Futures in Bulk

- Arbitrary-arity methods return futures that are triggered after completion of any/all futures

<table>
<thead>
<tr>
<th>Methods</th>
<th>Params</th>
<th>Returns</th>
<th>Behavior</th>
</tr>
</thead>
<tbody>
<tr>
<td>allOf</td>
<td>Varargs</td>
<td>Completable Future(&lt;\text{Void})&gt;</td>
<td>Return a future that completes when all futures in params complete</td>
</tr>
<tr>
<td>anyOf</td>
<td>Varargs</td>
<td>Completable Future(&lt;\text{Void})&gt;</td>
<td>Return a future that completes when any future in params complete</td>
</tr>
</tbody>
</table>

See [en.wikipedia.org/wiki/Arity](en.wikipedia.org/wiki/Arity)
Arbitrary-arity methods return futures that are triggered after completion of any/all futures.

- The returned future can be used to wait for any or all of \( N \) completable futures in an array to complete.

```java
class CompletetableFuture<T>

- CompletableFuture()
- cancel(boolean): boolean
- isCancelled(): boolean
- isDone(): boolean
- get()
- get(long, TimeUnit)
- join()
- complete(T): boolean
- supplyAsync(Supplier<U>): CompletableFuture<U>
- supplyAsync(Supplier<U>, Executor): CompletableFuture<U>
- runAsync(Runnable): CompletableFuture<Void>
- runAsync(Runnable, Executor): CompletableFuture<Void>
- completedFuture(U): CompletableFuture<U>
- thenApply(Function<?>): CompletableFuture<U>
- thenAccept(Consumer<? super T>): CompletableFuture<Void>
- thenCombine(CompletionStage<? extends U>, BiFunction<>::): CompletableFuture<V>
- thenCompose(Function<>::): CompletableFuture<U>
- whenComplete(BiConsumer<>::): CompletableFuture<T>

- allOf(CompletableFuture[]): CompletableFuture<Void>
- anyOf(CompletableFuture[]): CompletableFuture<Object>
```
Arbitrary-Arity Methods Process Futures in Bulk

- Arbitrary-arity methods return futures that are triggered after completion of any/all futures
- The returned future can be used to wait for any or all of $N$ completable futures in an array to complete

These “arbitrary-arity” methods are hard to program without using wrappers
Arbitrary-Arity Methods Process Futures in Bulk

- Arbitrary-arity methods return futures that are triggered after completion of any/all futures
- The returned future can be used to wait for any or all of $N$ completable futures in an array to complete

We focus on `allOf()`, which is like `thenCombine()` on steroids!

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/CompletableFuture.html#allOf
End of Advanced Java CompletableFuture Features: Arbitrary-Arity Methods