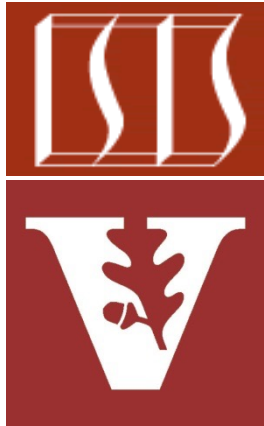


# Overview of the OneShot ExecutorServiceFuture Class



**Douglas C. Schmidt**  
**[d.schmidt@vanderbilt.edu](mailto:d.schmidt@vanderbilt.edu)**  
**[www.dre.vanderbilt.edu/~schmidt](http://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 SearchTaskGang case study
- Recognize the methods that are defined by the TaskGang framework
- Know the subclasses that extends TaskGang (directly or indirectly)
  - SearchTaskGangCommon
  - OneShotThreadPerTask
  - OneShotExecutorService
  - OneShotExecutorServiceFuture

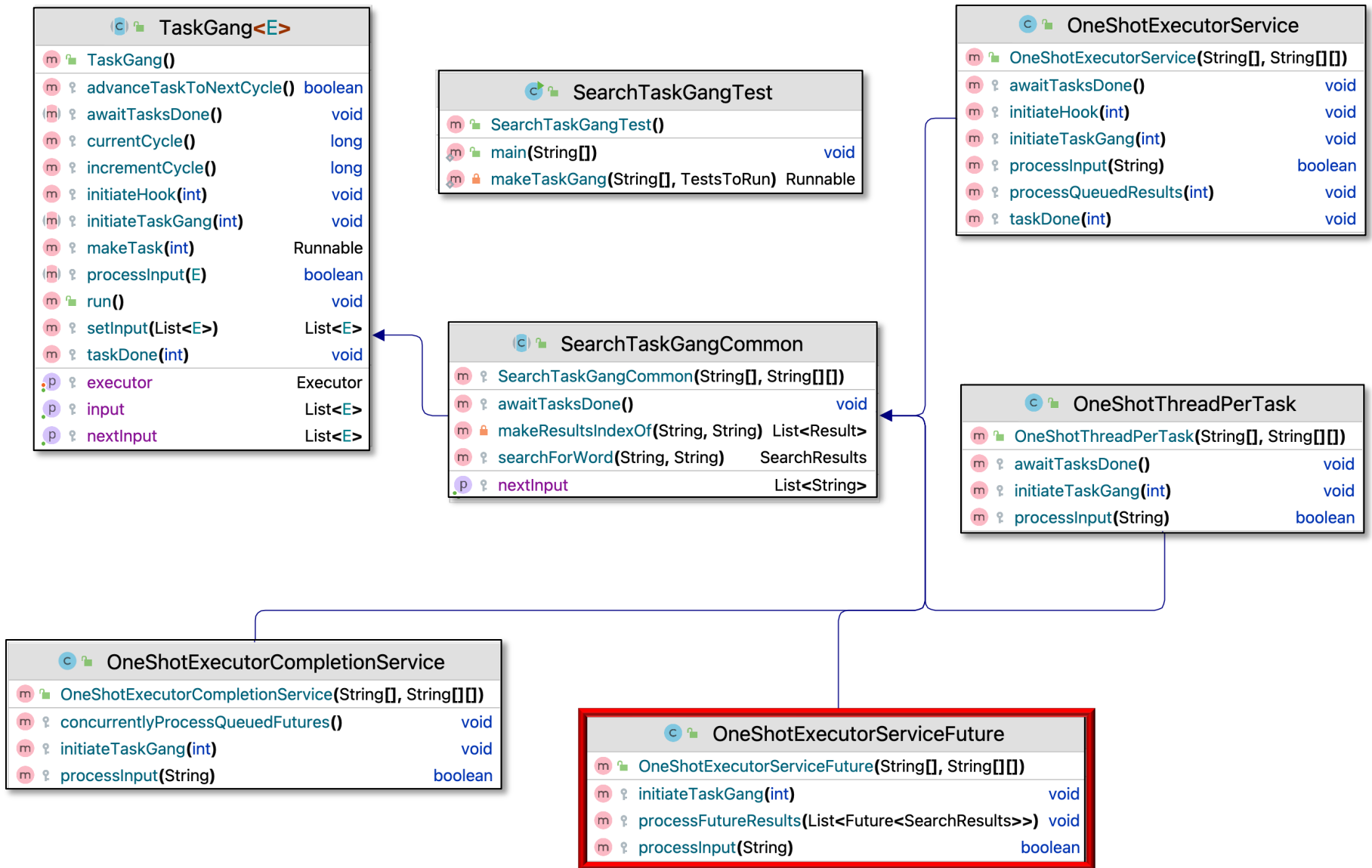
OneShotExecutorServiceFuture	
m	OneShotExecutorServiceFuture(String[], String[][])
f	mResultFutures List<Future<SearchResults>>
m	initiateTaskGang(int) void
m	processFutureResults(List<Future<SearchResults>>) void
m	processInput(String) boolean

See [SearchTaskGang/src/main/java/tasks/OneShotExecutorServiceFuture.java](#)

---

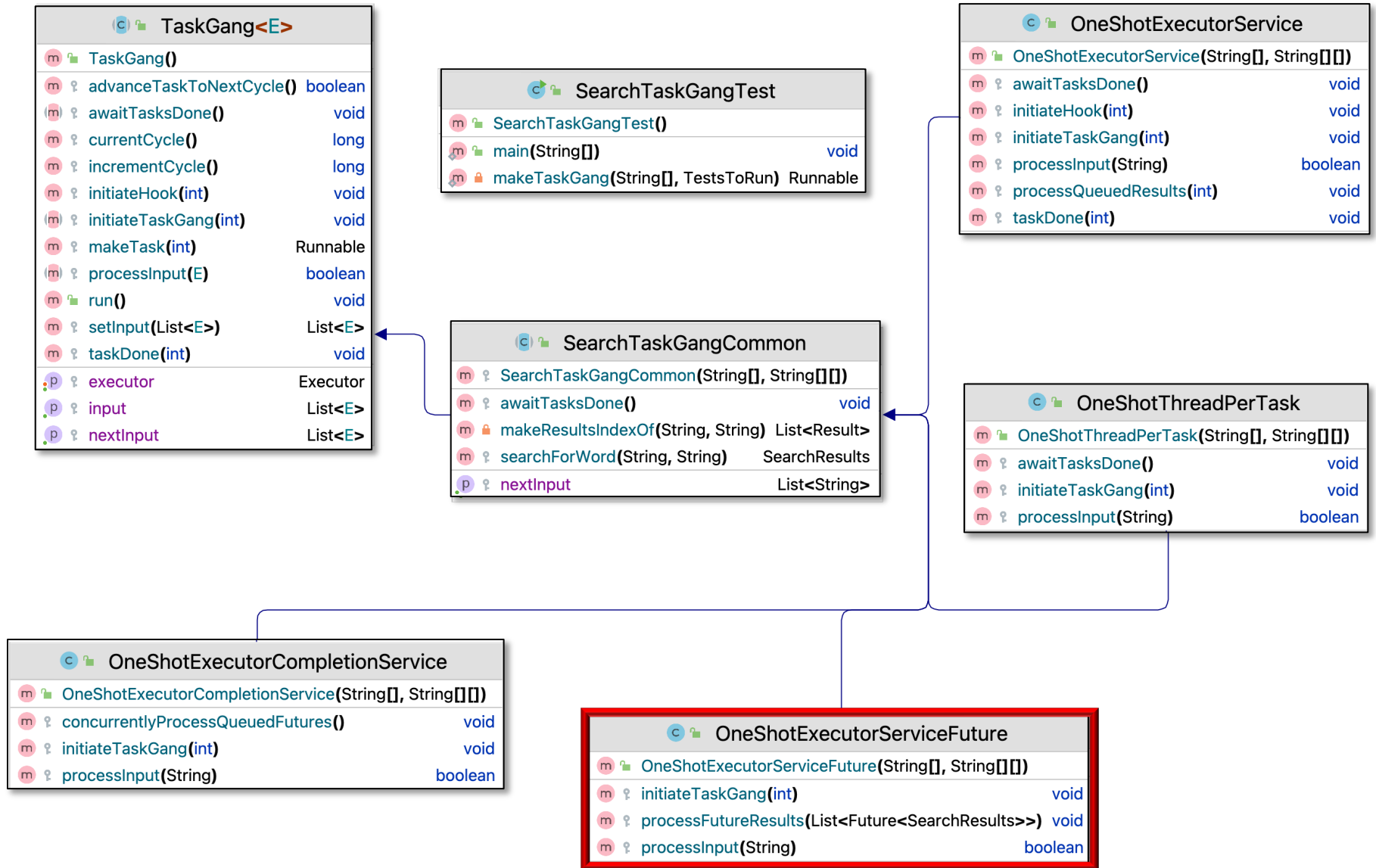
# Overview of Oneshot ExecutorServiceFuture

# Overview of OneShotExecutorServiceFuture



Customizes SearchTaskGang Common to do a one-shot search for words in a List of Strings with a thread pool & futures

# Overview of OneShotExecutorServiceFuture

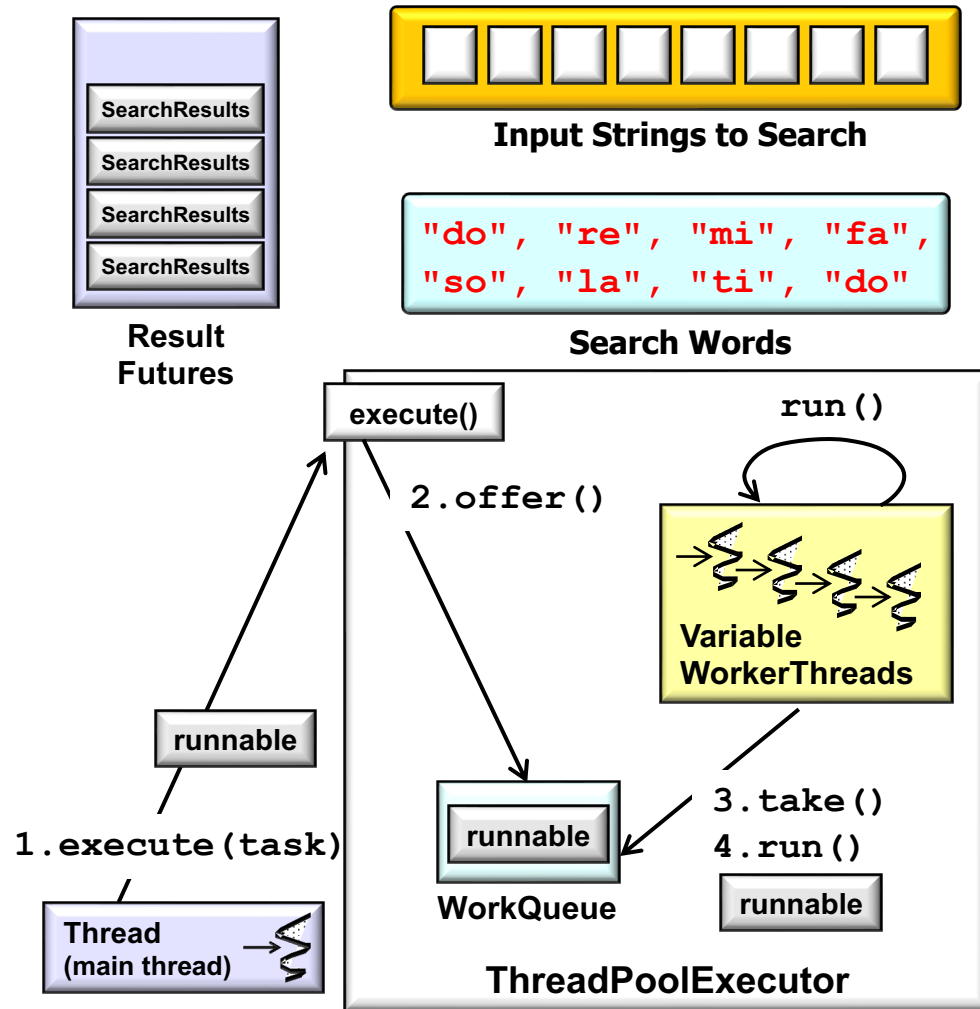


See [SearchTaskGang/src/main/java/tasks/OneShotExecutorServiceFuture.java](https://github.com/OneShot/OneShotExecutorServiceFuture/blob/master/src/main/java/tasks/OneShotExecutorServiceFuture.java)

# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words  
in a List of String objects

Strategy	Implementation
<i>Executor model</i>	Variable-size Thread pool
<i>Unit of concurrency</i>	Task per search word
<i>Results processing model</i>	Synchronous Future model that defers results processing



# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words  
in a List of String objects

OneShotExecutorServiceFuture	
m	<code>OneShotExecutorServiceFuture(String[], String[][])</code>
f	<code>mResultFutures</code> List<Future<SearchResults>>
m	<code>initiateTaskGang(int)</code> void
m	<code>processFutureResults(List&lt;Future&lt;SearchResults&gt;&gt;)</code> void
m	<code>processInput(String)</code> boolean

# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words  
in a List of String objects
- Uses a cached Thread pool
  - Created by an Executors  
factory method

OneShotExecutorServiceFuture	
m	<b>OneShotExecutorServiceFuture(String[], String[][])</b>
f	mResultFutures List<Future<SearchResults>>
m	initiateTaskGang(int) void
m	processFutureResults(List<Future<SearchResults>>) void
m	processInput(String) boolean

```
setExecutor  
    (Executors.newCachedThreadPool ( ) ) ;
```



# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words  
in a List of String objects
  - Uses a cached Thread pool
  - Each input data element is  
*not* processed concurrently

OneShotExecutorServiceFuture		
m	OneShotExecutorServiceFuture(String[], String[][])	
f	mResultFutures	List<Future<SearchResults>>
m	<b>initiateTaskGang(int)</b>	void
m	processFutureResults(List<Future<SearchResults>>)	void
m	processInput(String)	boolean

```
mResultFutures = new ArrayList<Future<SearchResults>>  
                (inputSize * mWordsToFind.length);  
  
for (var inputData : getInput())  
    processInput(inputData);  
  
processFutureResults(mResultFutures);
```

# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words in a List of String objects
  - Uses a cached Thread pool
  - Each input data element is *not* processed concurrently
  - Instead, the ExecutorService searches each word in the input concurrently

OneShotExecutorServiceFuture		
m	OneShotExecutorServiceFuture(String[], String[][])	
f	mResultFutures	List<Future<SearchResults>>
m	initiateTaskGang(int)	void
m	processFutureResults(List<Future<SearchResults>>)	void
m	processInput(String)	boolean

```
for (var word : mWordsToFind) {  
    var resultFuture = getExecutor().submit  
        (( ) -> searchForWord(word, inputData));  
  
    mResultFutures.add(resultFuture);  
}
```

# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words in a List of String objects
  - Uses a cached Thread pool
  - Each input data element is *not* processed concurrently
  - Instead, the ExecutorService searches each word in the input concurrently
    - Results stored in Futures List

OneShotExecutorServiceFuture		
m	OneShotExecutorServiceFuture(String[], String[][])	
f	mResultFutures	List<Future<SearchResults>>
m	initiateTaskGang(int)	void
m	processFutureResults(List<Future<SearchResults>>)	void
m	processInput(String)	boolean

```
for (var word : mWordsToFind) {  
    var resultFuture = getExecutor().submit  
        (( ) -> searchForWord(word, inputData));  
  
    mResultFutures.add(resultFuture);  
}
```

# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words in a List of String objects
  - Uses a cached Thread pool
  - Each input data element is *not* processed concurrently
  - Instead, the ExecutorService searches each word in the input concurrently
- Results processing doesn't start until all Futures added

OneShotExecutorServiceFuture	
m	OneShotExecutorServiceFuture(String[], String[][])
f	mResultFutures List<Future<SearchResults>>
m	<b>initiateTaskGang(int)</b> void
m	processFutureResults(List<Future<SearchResults>>) void
m	processInput(String) boolean

```
mResultFutures = new ArrayList<Future<SearchResults>>  
                (inputSize * mWordsToFind.length);  
  
for (var inputData : getInput()) processInput(inputData);  
  
processFutureResults (mResultFutures);
```

# Overview of OneShotExecutorServiceFuture

- Customizes SearchTaskGang  
Common to search for words in a List of String objects
  - Uses a cached Thread pool
  - Each input data element is *not* processed concurrently
  - Instead, the ExecutorService searches each word in the input concurrently
  - Results processing doesn't start until all Futures added
  - Future results are processed concurrently wrt submit() calls

OneShotExecutorServiceFuture		
m	OneShotExecutorServiceFuture(String[], String[][])	
f	mResultFutures	List<Future<SearchResults>>
m	initiateTaskGang(int)	void
m	processFutureResults(List<Future<SearchResults>>)	void
m	processInput(String)	boolean

```
for (var resultFuture
     : resultFutures) {
    ...
    rethrowSupplier
        (resultFuture::get)
        .get().print();
    ...
}
```

# Overview of OneShotExecutorServiceFuture

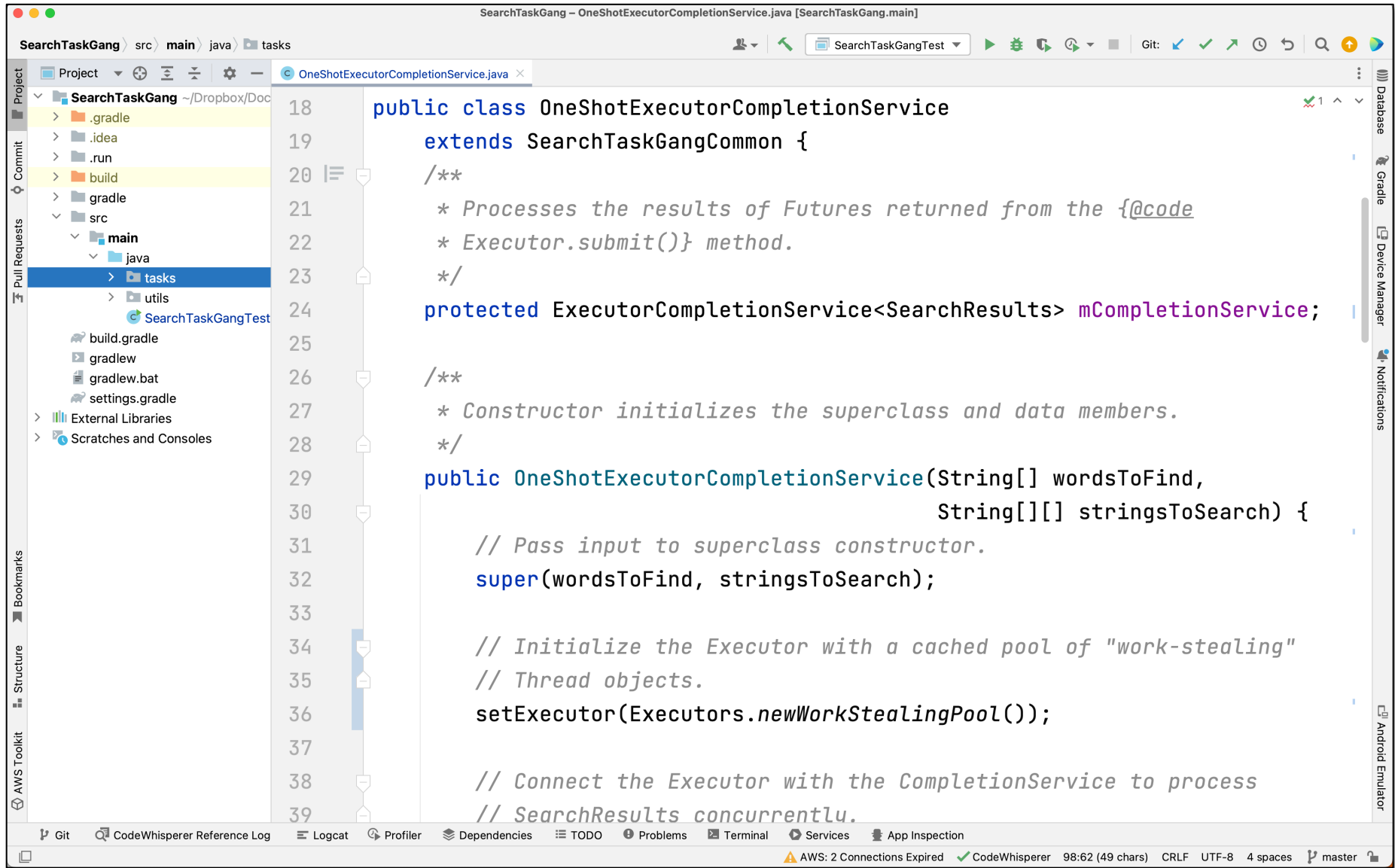
- Customizes SearchTaskGang
  - Common to search for words in a List of String objects
    - Uses a cached Thread pool
    - Each input data element is *not* processed concurrently
    - Instead, the ExecutorService searches each word in the input concurrently
    - Results processing doesn't start until all Futures added
    - Future results are processed concurrently wrt submit() calls
      - However, get() will block if results aren't done

OneShotExecutorServiceFuture		
m	OneShotExecutorServiceFuture(String[], String[][])	
f	mResultFutures	List<Future<SearchResults>>
m	initiateTaskGang(int)	void
m	processFutureResults(List<Future<SearchResults>>)	void
m	processInput(String)	boolean

```
for (var resultFuture
     : resultFutures) {
    ...
    rethrowSupplier
        (resultFuture::get)
        .get().print();
    ...
}
```

This "synchronous" Future processing model has limitations..

# Walkthrough of OneShotExecutorServiceFuture



```
SearchTaskGang - OneShotExecutorCompletionService.java [SearchTaskGang.main]
SearchTaskGang > src > main > java > tasks
Project SearchTaskGang - ~/Dropbox/Doc
  .gradle
  .idea
  .run
  build
  gradle
  src
    main
      java
        tasks
          utils
            SearchTaskGangTest
  build.gradle
  gradlew
  gradlew.bat
  settings.gradle
  External Libraries
  Scratches and Consoles
  Database
  Gradle
  Device Manager
  Notifications
  Android Emulator

18 public class OneShotExecutorCompletionService
19 extends SearchTaskGangCommon {
20 /**
21  * Processes the results of Futures returned from the {@code
22  * Executor.submit()} method.
23  */
24 protected ExecutorCompletionService<SearchResults> mCompletionService;
25
26 /**
27  * Constructor initializes the superclass and data members.
28  */
29 public OneShotExecutorCompletionService(String[] wordsToFind,
30                                         String[][] stringsToSearch) {
31     // Pass input to superclass constructor.
32     super(wordsToFind, stringsToSearch);
33
34     // Initialize the Executor with a cached pool of "work-stealing"
35     // Thread objects.
36     setExecutor(Executors.newWorkStealingPool());
37
38     // Connect the Executor with the CompletionService to process
39     // SearchResults concurrently.
```

Git CodeWhisperer Reference Log Logcat Profiler Dependencies TODO Problems Terminal Services App Inspection  
AWS: 2 Connections Expired CodeWhisperer 98:62 (49 chars) CRLF UTF-8 4 spaces master

See [SearchTaskGang/src/main/java/tasks/OneShotExecutorServiceFuture.java](https://github.com/OneShot/OneShotExecutorServiceFuture.java)

---

End of Overview of One  
ShotExecutorServiceFuture