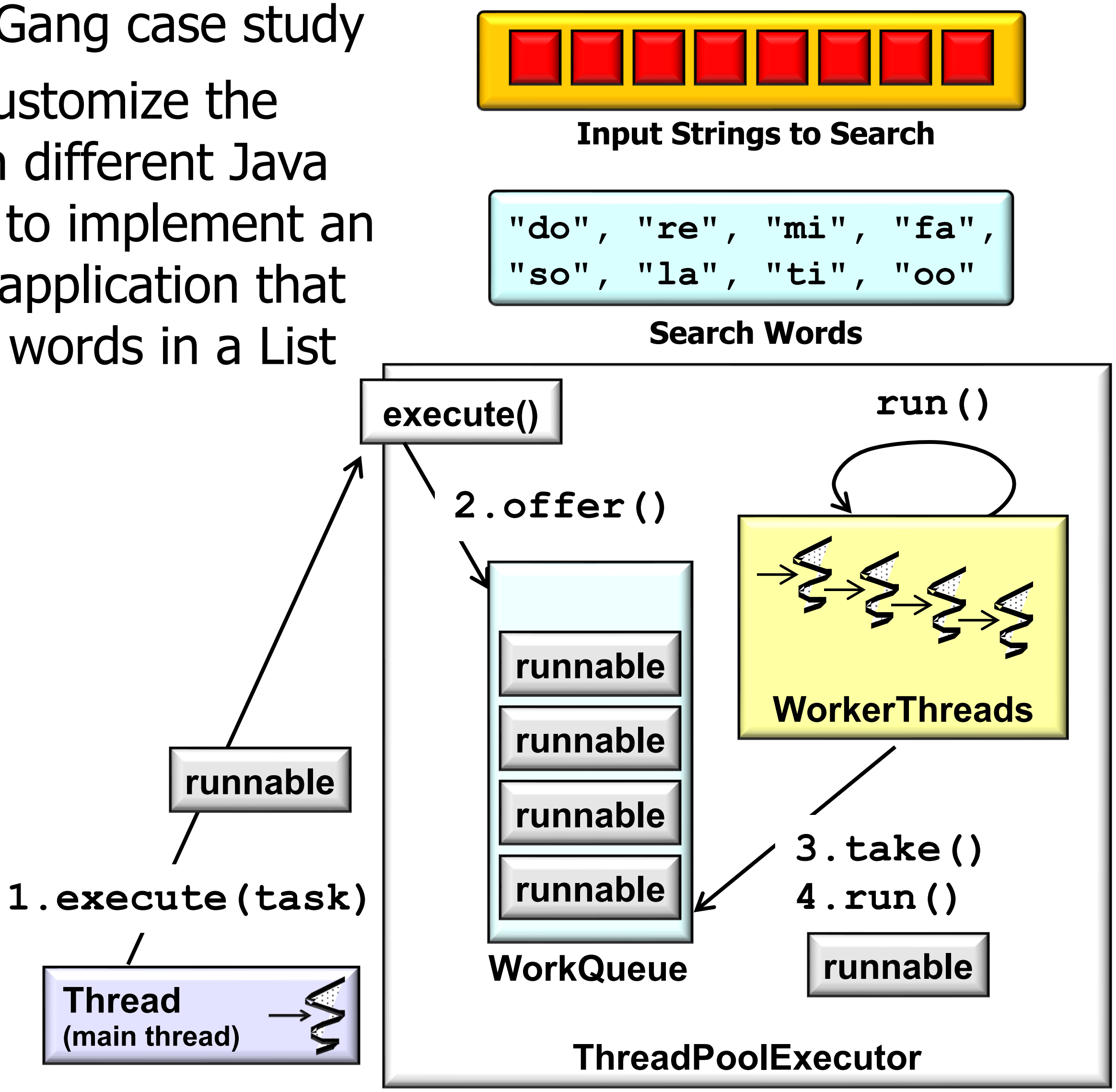
Overview of the Search TaskGang Case Study

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

Institute for Software
Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

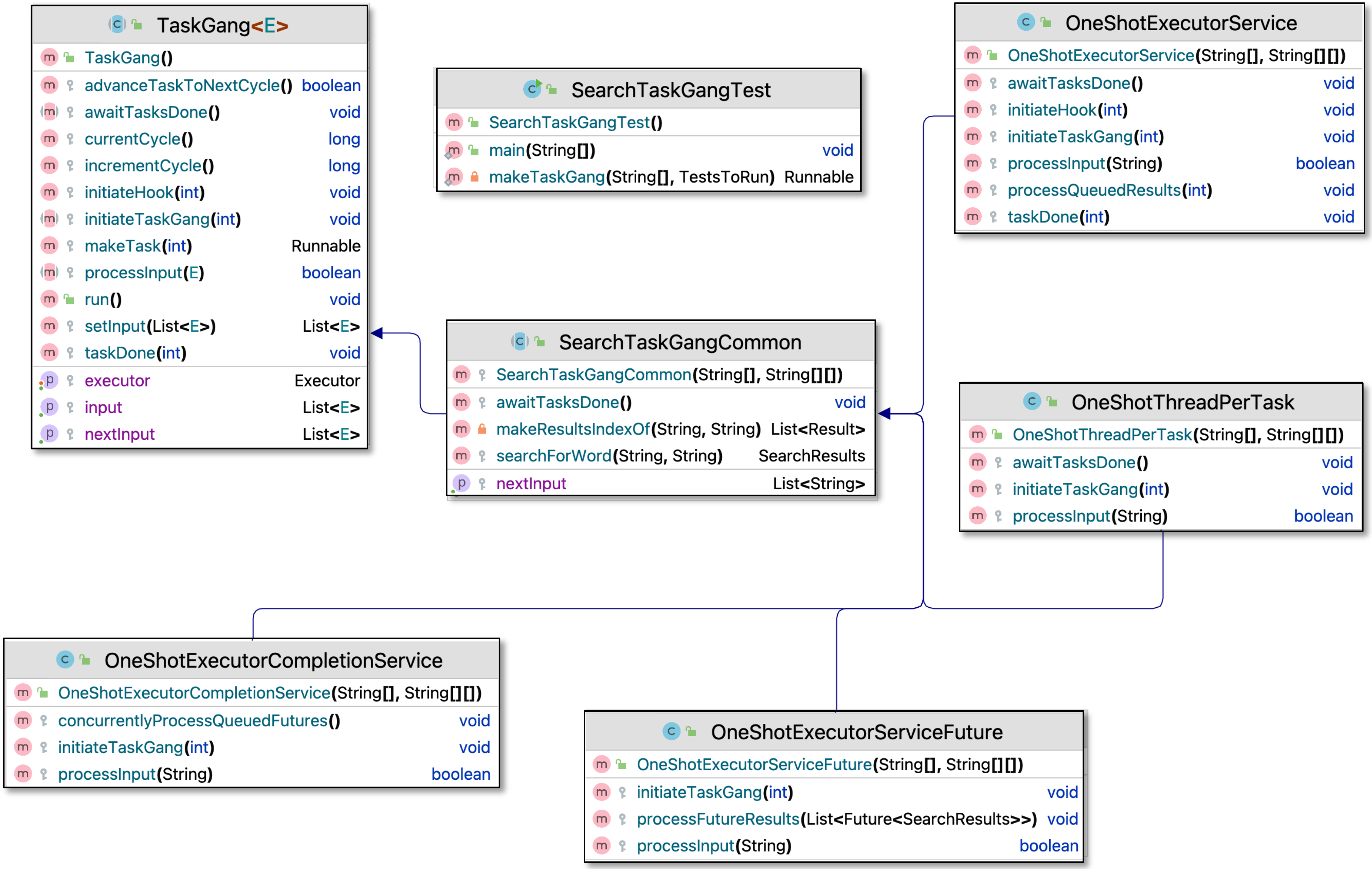
- Understand the SearchTaskGang case study
  - Defines subclasses that customize the TaskGang framework with different Java concurrency mechanisms to implement an "embarrassingly parallel" application that concurrently searches for words in a List of input String objects



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

Overview of Search TaskGang Case Study

Overview of the SearchTaskGang Case Study



TaskGang<E>

- TaskGang()
- advanceTaskToNextCycle() boolean
- awaitTasksDone() void
- currentCycle() long
- incrementCycle() long
- initiateHook(int) void
- initiateTaskGang(int) void
- makeTask(int) Runnable
- processInput(E) boolean
- run() void
- setInput(List<E>) List<E>
- taskDone(int) void
- executor Executor
- input List<E>
- nextInput List<E>

SearchTaskGangTest

- SearchTaskGangTest()
- main(String[]) void
- makeTaskGang(String[], TestsToRun) Runnable

OneShotExecutorService

- OneShotExecutorService(String[], String[][])
- awaitTasksDone() void
- initiateHook(int) void
- initiateTaskGang(int) void
- processInput(String) boolean
- processQueuedResults(int) void
- taskDone(int) void

SearchTaskGangCommon

- SearchTaskGangCommon(String[], String[][])
- awaitTasksDone() void
- makeResultsIndexOf(String, String) List<Result>
- searchForWord(String, String) SearchResults
- nextInput List<String>

OneShotThreadPerTask

- OneShotThreadPerTask(String[], String[][])
- awaitTasksDone() void
- initiateTaskGang(int) void
- processInput(String) boolean

OneShotExecutorCompletionService

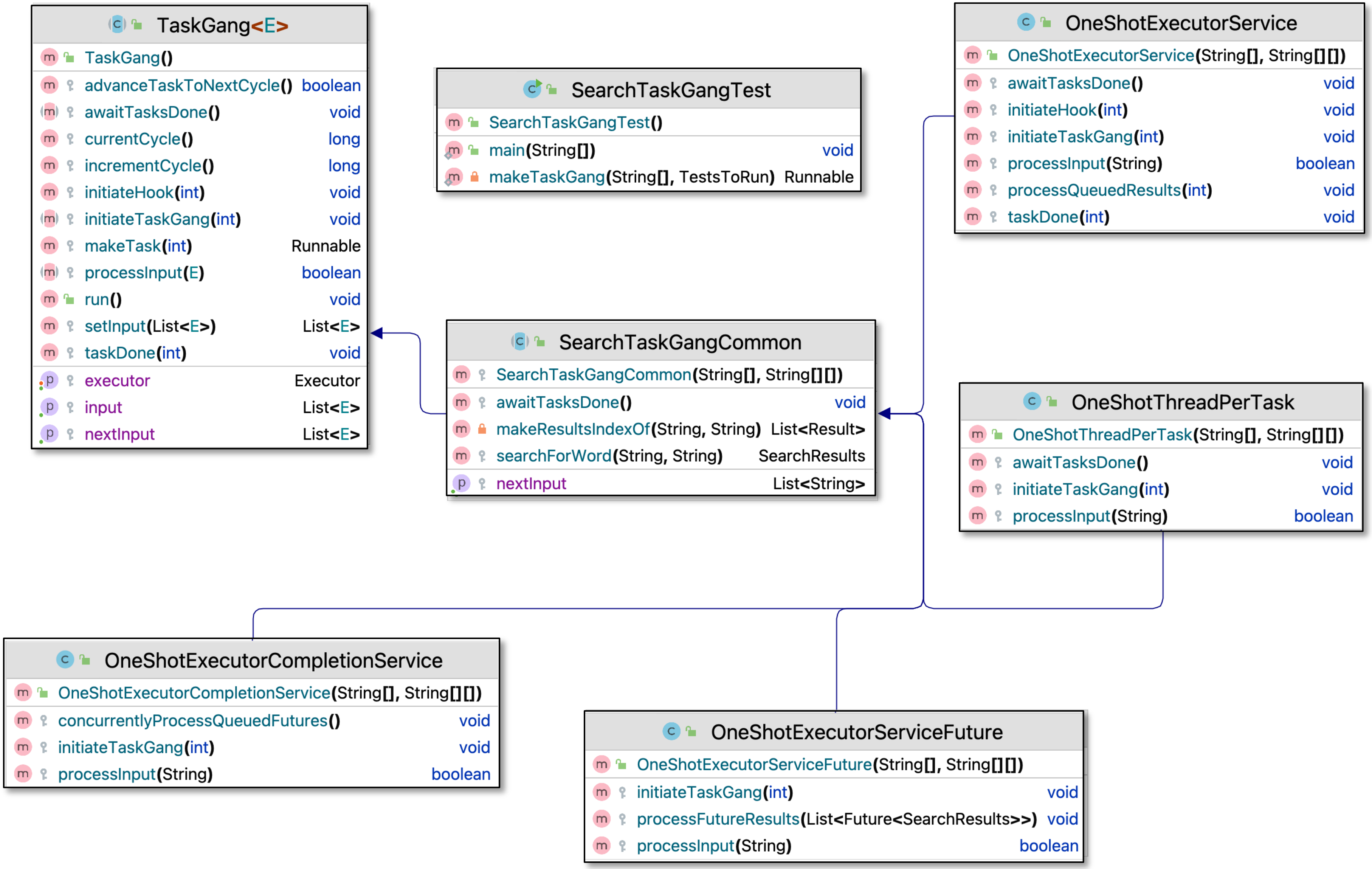
- OneShotExecutorCompletionService(String[], String[][])
- concurrentlyProcessQueuedFutures() void
- initiateTaskGang(int) void
- processInput(String) boolean

OneShotExecutorServiceFuture

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

A set of classes that showcase various Java Executor framework mechanisms

Overview of the SearchTaskGang Case Study



TaskGang<E>	
TaskGang()	
advanceTaskToNextCycle()	boolean
awaitTasksDone()	void
currentCycle()	long
incrementCycle()	long
initiateHook(int)	void
initiateTaskGang(int)	void
makeTask(int)	Runnable
processInput(E)	boolean
run()	void
setInput(List<E>)	List<E>
taskDone(int)	void
executor	Executor
input	List<E>
nextInput	List<E>

SearchTaskGangTest	
SearchTaskGangTest()	
main(String[])	void
makeTaskGang(String[], TestsToRun)	Runnable

OneShotExecutorService	
OneShotExecutorService(String[], String[][])	
awaitTasksDone()	void
initiateHook(int)	void
initiateTaskGang(int)	void
processInput(String)	boolean
processQueuedResults(int)	void
taskDone(int)	void

SearchTaskGangCommon	
SearchTaskGangCommon(String[], String[][])	
awaitTasksDone()	void
makeResultsIndexOf(String, String)	List<Result>
searchForWord(String, String)	SearchResults
nextInput	List<String>

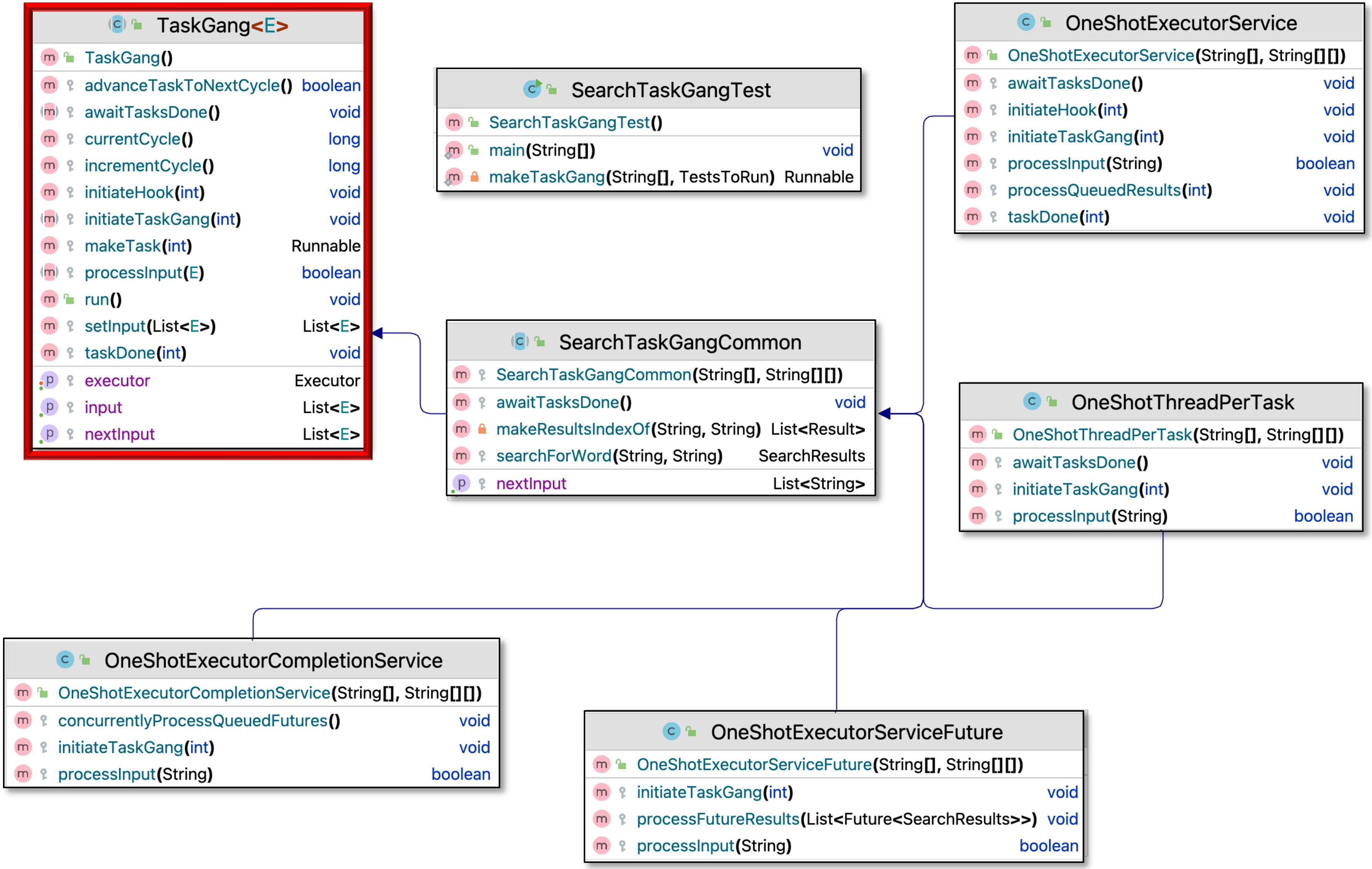
OneShotThreadPerTask	
OneShotThreadPerTask(String[], String[][])	
awaitTasksDone()	void
initiateTaskGang(int)	void
processInput(String)	boolean

OneShotExecutorCompletionService	
OneShotExecutorCompletionService(String[], String[][])	
concurrentlyProcessQueuedFutures()	void
initiateTaskGang(int)	void
processInput(String)	boolean

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

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

Overview of the SearchTaskGang Case Study



TaskGang<E>	
TaskGang()	
advanceTaskToNextCycle()	boolean
awaitTasksDone()	void
currentCycle()	long
incrementCycle()	long
initiateHook(int)	void
initiateTaskGang(int)	void
makeTask(int)	Runnable
processInput(E)	boolean
run()	void
setInput(List<E>)	List<E>
taskDone(int)	void
executor	Executor
input	List<E>
nextInput	List<E>

SearchTaskGangTest	
SearchTaskGangTest()	
main(String[])	void
makeTaskGang(String[], TestsToRun)	Runnable

OneShotExecutorService	
OneShotExecutorService(String[], String[][])	
awaitTasksDone()	void
initiateHook(int)	void
initiateTaskGang(int)	void
processInput(String)	boolean
processQueuedResults(int)	void
taskDone(int)	void

SearchTaskGangCommon	
SearchTaskGangCommon(String[], String[][])	
awaitTasksDone()	void
makeResultsIndexOf(String, String)	List<Result>
searchForWord(String, String)	SearchResults
nextInput	List<String>

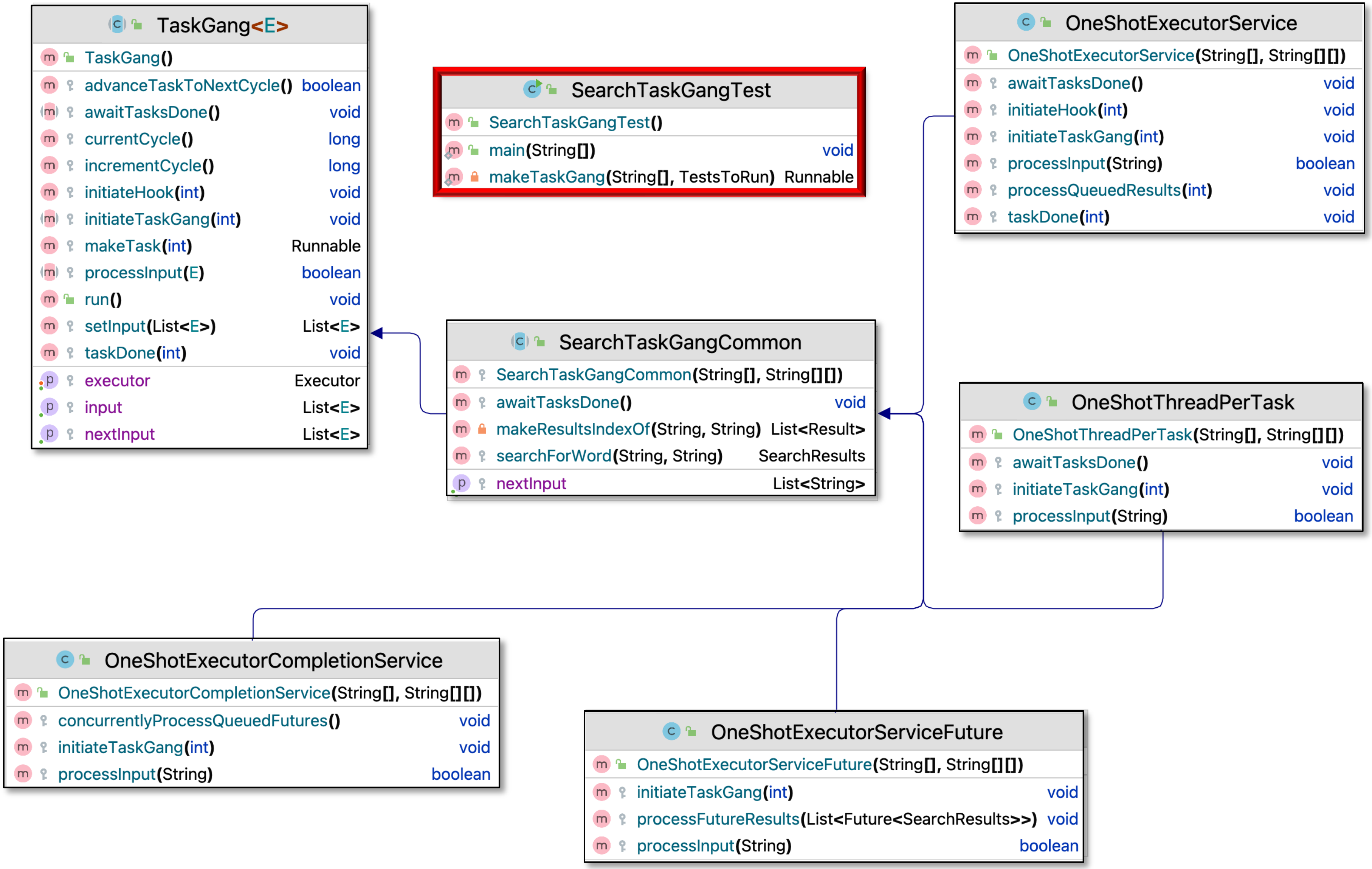
OneShotThreadPerTask	
OneShotThreadPerTask(String[], String[][])	
awaitTasksDone()	void
initiateTaskGang(int)	void
processInput(String)	boolean

OneShotExecutorCompletionService	
OneShotExecutorCompletionService(String[], String[][])	
concurrentlyProcessQueuedFutures()	void
initiateTaskGang(int)	void
processInput(String)	boolean

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

This super class defines a framework for spawning & running a "gang" of tasks

Overview of the SearchTaskGang Case Study



TaskGang<E>	
TaskGang()	
advanceTaskToNextCycle()	boolean
awaitTasksDone()	void
currentCycle()	long
incrementCycle()	long
initiateHook(int)	void
initiateTaskGang(int)	void
makeTask(int)	Runnable
processInput(E)	boolean
run()	void
setInput(List<E>)	List<E>
taskDone(int)	void
executor	Executor
input	List<E>
nextInput	List<E>

SearchTaskGangTest	
SearchTaskGangTest()	
main(String[])	void
makeTaskGang(String[], TestsToRun)	Runnable

OneShotExecutorService	
OneShotExecutorService(String[], String[][])	
awaitTasksDone()	void
initiateHook(int)	void
initiateTaskGang(int)	void
processInput(String)	boolean
processQueuedResults(int)	void
taskDone(int)	void

SearchTaskGangCommon	
SearchTaskGangCommon(String[], String[][])	
awaitTasksDone()	void
makeResultsIndexOf(String, String)	List<Result>
searchForWord(String, String)	SearchResults
nextInput	List<String>

OneShotThreadPerTask	
OneShotThreadPerTask(String[], String[][])	
awaitTasksDone()	void
initiateTaskGang(int)	void
processInput(String)	boolean

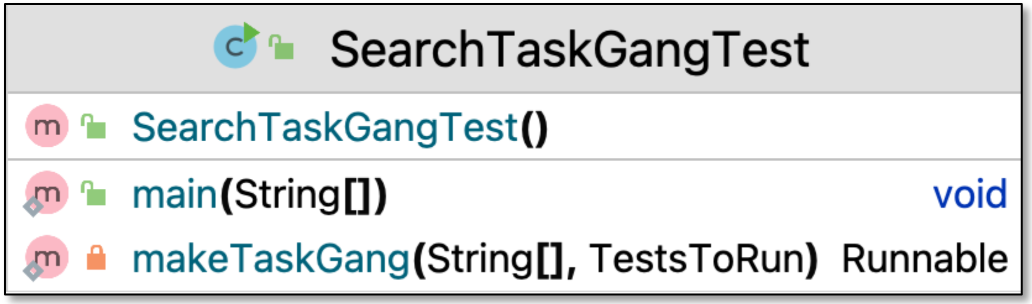
OneShotExecutorCompletionService	
OneShotExecutorCompletionService(String[], String[][])	
concurrentlyProcessQueuedFutures()	void
initiateTaskGang(int)	void
processInput(String)	boolean

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

A driver program that runs various specializations of the TaskGang framework to showcase different Java Executor framework capabilities

Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings



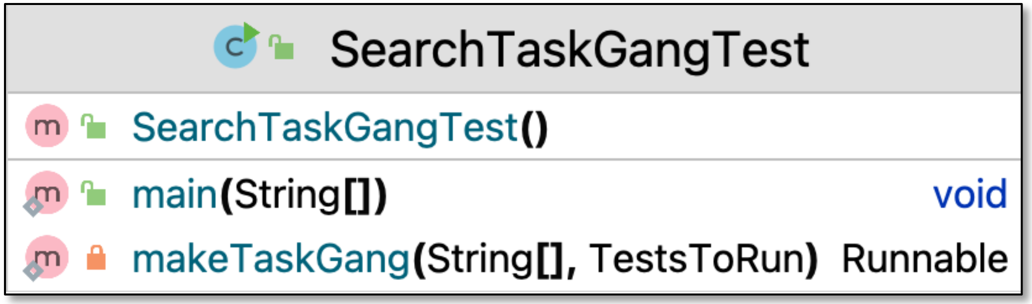
SearchTaskGangTest

SearchTaskGangTest()	
main(String[])	void
makeTaskGang(String[], TestsToRun)	Runnable

See SearchTaskGang/src/main/java/SearchTaskGangTest.java

Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings
  - Defines TestsToRun enum



SearchTaskGangTest

m	SearchTaskGangTest()	
m	main(String[])	void
m	makeTaskGang(String[], TestsToRun)	Runnable

```
enum TestsToRun {
    ONESHOT_THREAD_PER_TASK,
    ONESHOT_EXECUTOR_SERVICE,
    ONESHOT_EXECUTOR_SERVICE_FUTURE,
    ONESHOT_EXECUTOR_COMPLETION_SERVICE
}
```

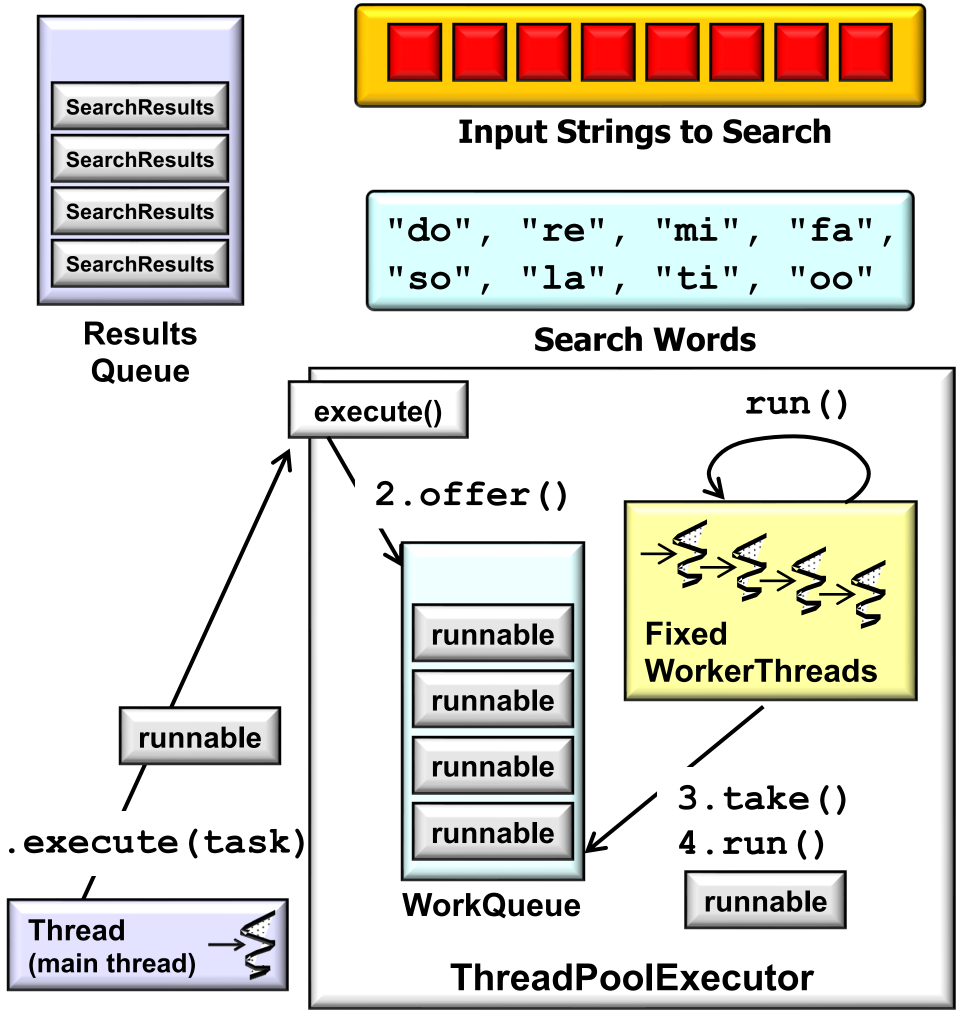
Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings
  - Defines TestsToRun enum

Strategy	**Implementation**
Executor model	Virtual Thread-per-Task
Unit of concurrency	Virtual Thread per input String
Results processing model	Synchronous processing

ONESHOT_THREAD_PER_TASK

Processes a one-shot List of tasks via an Executor that creates a Java virtual Thread for each task



See [SearchTaskGang/src/main/java/tasks/OneShotThreadPerTask.java]

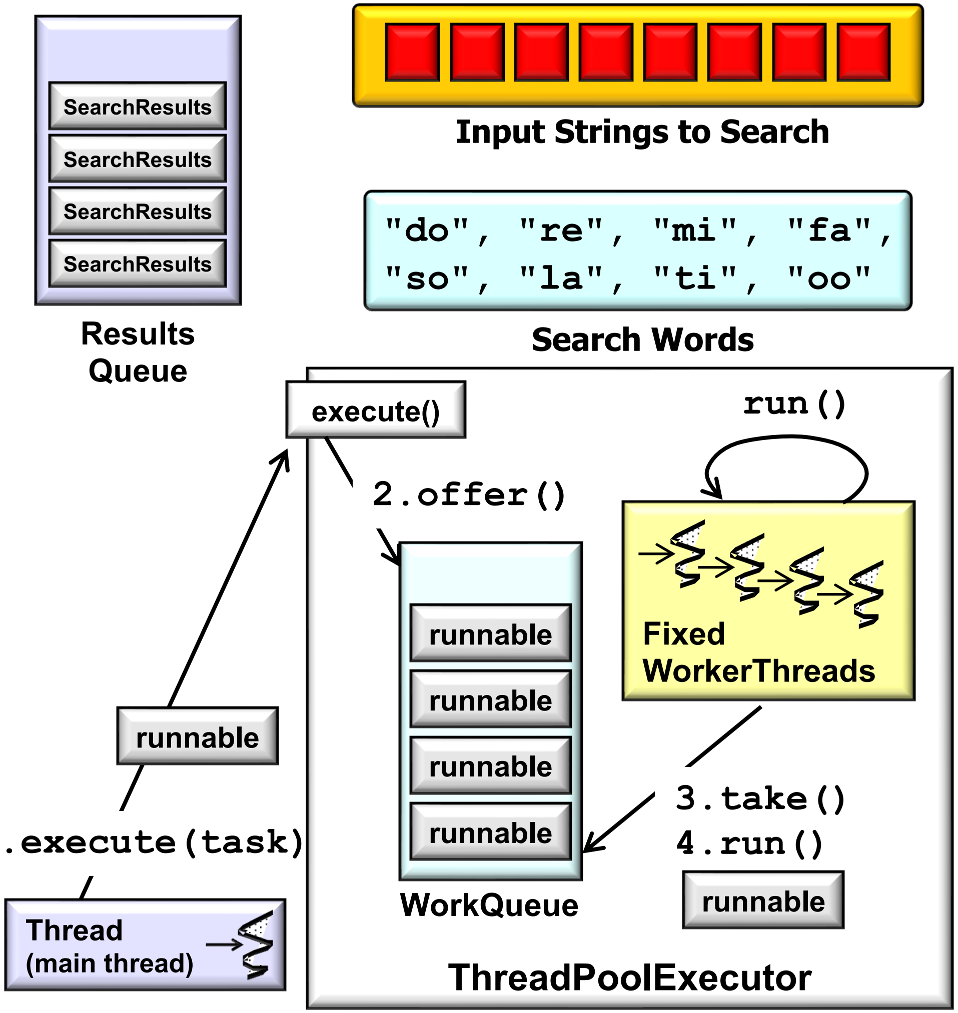
Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings
  - Defines TestsToRun enum

Strategy	Implementation
Executor model	Fixed-size Thread pool
Unit of concurrency	Task per input String
Results processing model	BlockingQueue stores results for immediate concurrent processing

ONESHOT_EXECUTOR_SERVICE

Processes a one-shot List of tasks via a fixed-size pool of Thread objects associated with an ExecutorService that's used as a barrier synchronizer



See SearchTaskGang/src/main/java/tasks/OneShotExecutorService.java

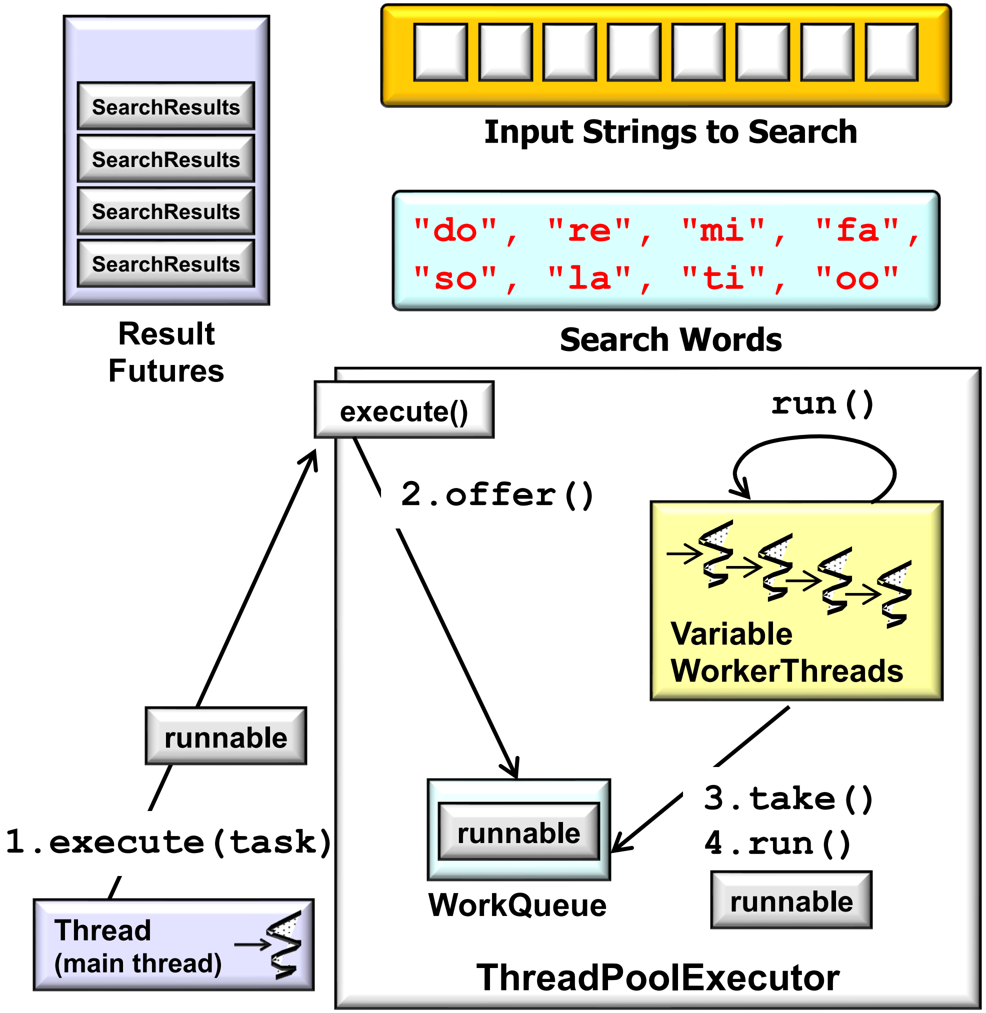
Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings
  - Defines TestsToRun enum

Strategy	Implementation
Executor model	Variable-size Thread pool
Unit of concurrency	Task per search word
Results processing model	Synchronous Future model for deferred concurrent processing

ONESHOT_EXECUTOR _SERVICE_FUTURE

Processes a one-shot List of tasks via a variable-size pool of Thread objects associated with an ExecutorService



See SearchTaskGang/src/main/java/tasks/OneShotExecutorServiceFuture.java

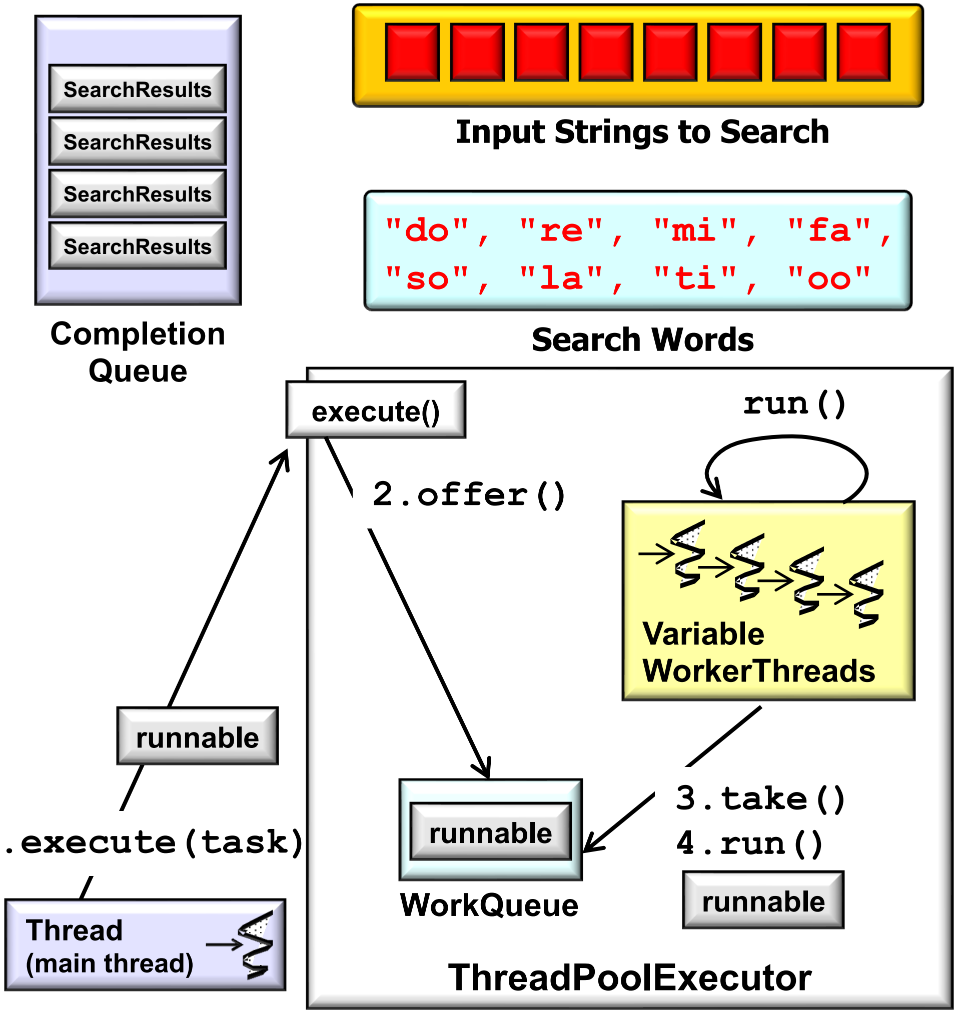
Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings
  - Defines TestsToRun enum

Strategy	Implementation
Executor model	"Work-stealing" Thread pool
Unit of concurrency	Task per search word & input String
Results processing model	Asynchronous Future model for immediate concurrent processing

ONESHOT_EXECUTOR _COMPLETION_SERVICE

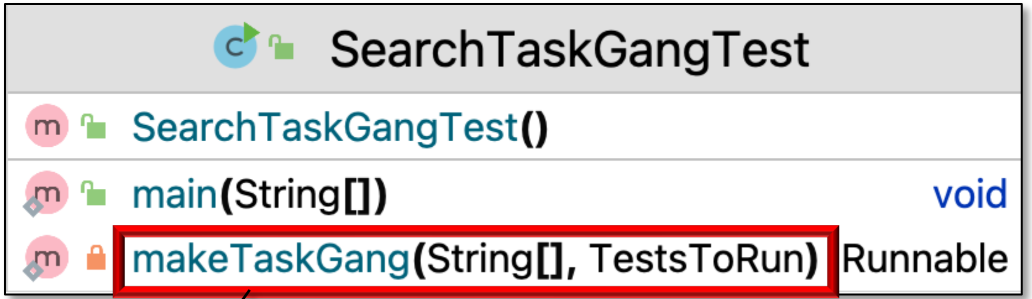
Processes a one-shot List of tasks via a pool of "work-stealing" Thread objects associated with an ExecutorService



See SearchTaskGang/src/main/java/tasks/OneShotExecutorCompletionService.java

Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings
  - Defines TestsToRun enum
  - Defines a factory method to create the tests



SearchTaskGangTest

SearchTaskGangTest()

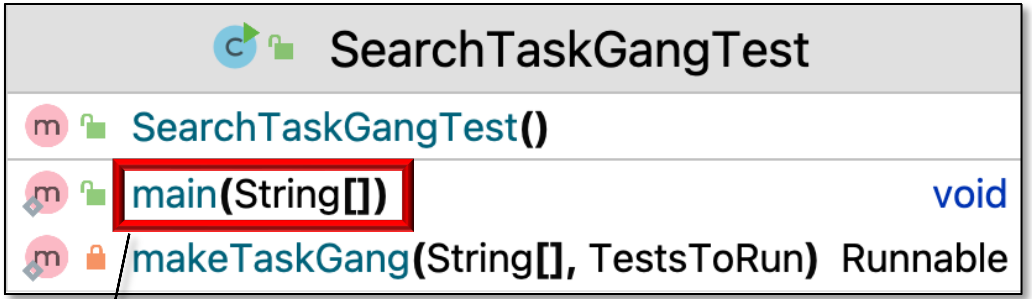
main(String[]) void

makeTaskGang(String[], TestsToRun) Runnable

```
return switch (choice) {
case ONESHOT_THREAD_PER_TASK ->
  new OneShotThreadPerTask
      (sWordList, sOneShotInputStrings);
case ONESHOT_EXECUTOR_SERVICE ->
  new OneShotExecutorService
      (sWordList, sOneShotInputStrings);
  ...
}
```

Overview of the SearchTaskGangTest Class

- Customizes TaskGang framework to search for words in a List of Strings
  - Defines TestsToRun enum
  - Defines a factory method to create the tests
  - Define a main() method to iterate through all the tests & create/run them



SearchTaskGangTest

SearchTaskGangTest()

main(String[]) void

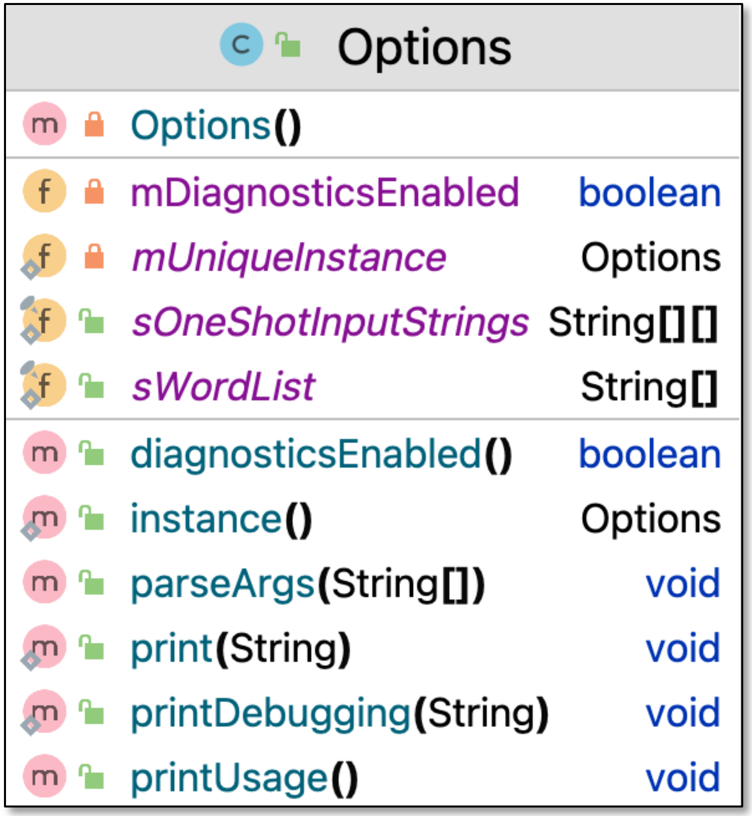
makeTaskGang(String[], TestsToRun) Runnable

```
for (var test : TestsToRun.values())
  makeTaskGang(mWordList, test).run()
```

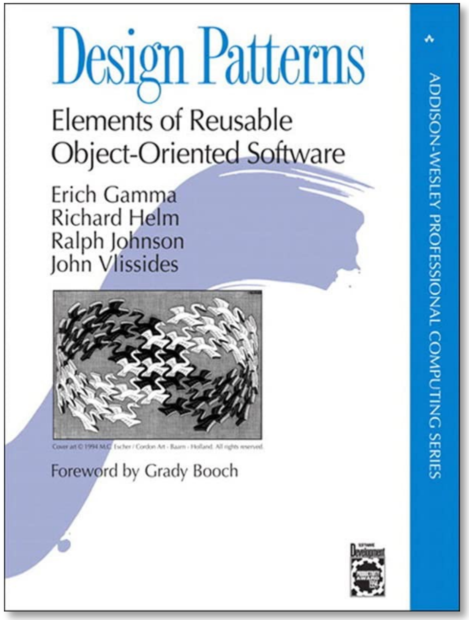
Overview of the Options Singleton

Overview of the Options Singleton

- This class implements the *Singleton* pattern to handle command-line option processing



	Options	
m	Options()	
f	mDiagnosticsEnabled	boolean
f	*mUniqueInstance*	Options
f	*sOneShotInputStrings*	String[][]
f	*sWordList*	String[]
m	diagnosticsEnabled()	boolean
m	instance()	Options
m	parseArgs(String[])	void
m	print(String)	void
m	printDebugging(String)	void
m	printUsage()	void

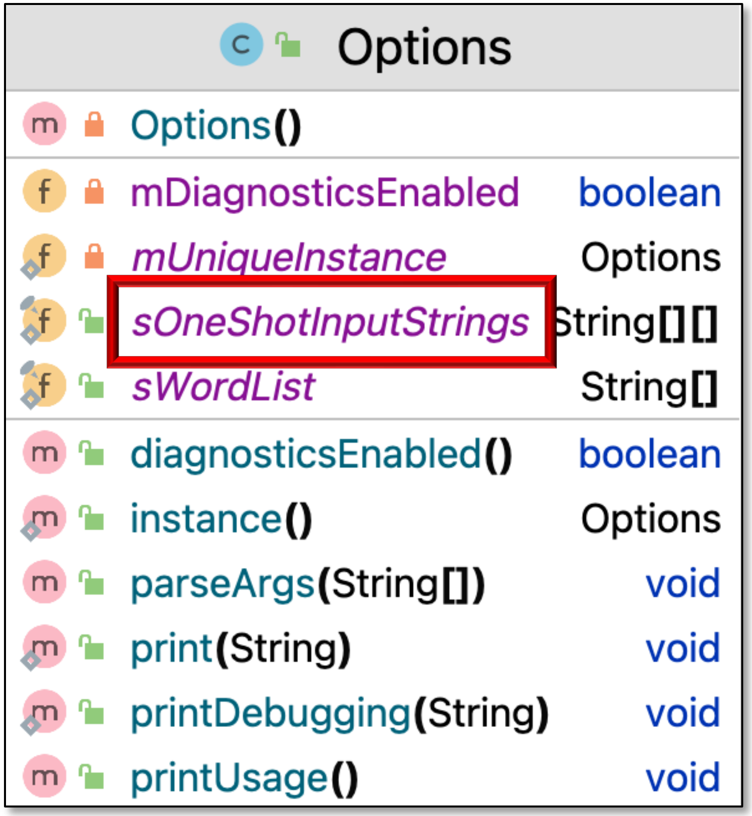

Design Patterns
Elements of Reusable Object-Oriented Software
Erich Gamma
Richard Helm
Ralph Johnson
John Vlissides
Foreword by Grady Booch
ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES

See en.wikipedia.org/wiki/Singleton_pattern

Overview of the Options Singleton

- This class implements the *Singleton* pattern to handle command-line option processing
  - Defines one-shot input String objects

```
public final static String[][]
mOneShotInputStrings = {
    {"xreo", "xfao",
     "xmiomio", "xlao",
     "xtiotio", "xsoosoo",
     "xdoo", "xdoodoo"}
  };
```



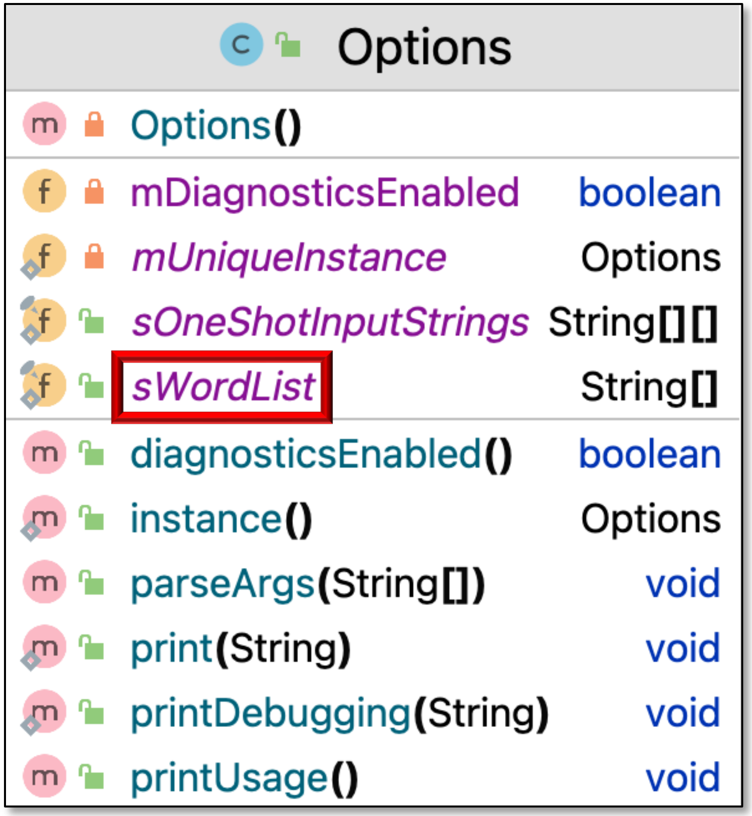
Options

Options()	
mDiagnosticsEnabled	boolean
mUniqueInstance	Options
sOneShotInputStrings	String[][]
sWordList	String[]
diagnosticsEnabled()	boolean
instance()	Options
parseArgs(String[])	void
print(String)	void
printDebugging(String)	void
printUsage()	void

Overview of the Options Singleton

- This class implements the *Singleton* pattern to handle command-line option processing
  - Defines one-shot input String objects
  - Defines array of search words to locate in the input String objects

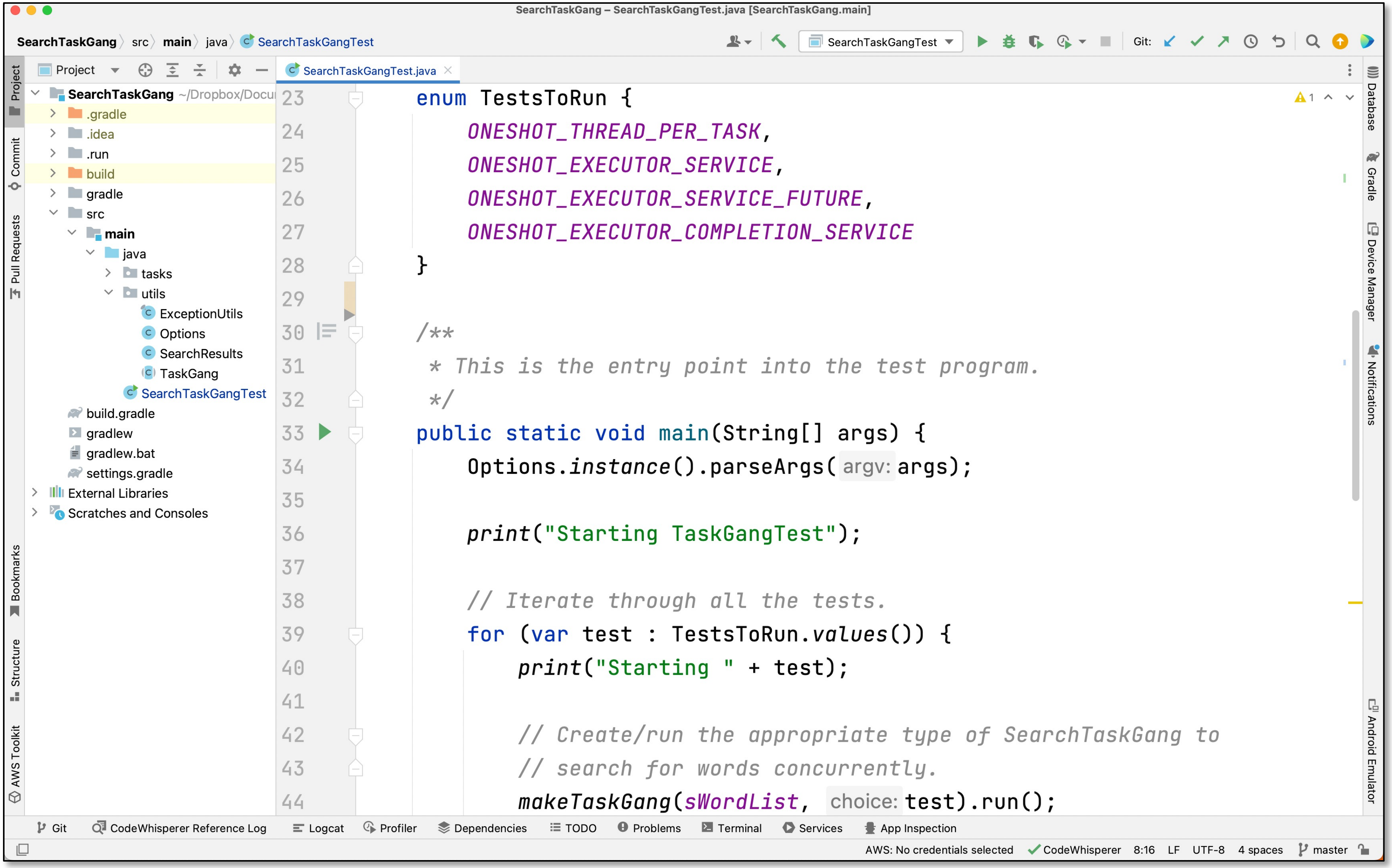
```
public String[] mWordList = {
    "do", "re", "mi",
    "fa", "so", "la",
    "ti", "oo"
}
```



Options

Options()	
mDiagnosticsEnabled	boolean
mUniqueInstance	Options
sOneShotInputStrings	String[][]
sWordList	String[]
diagnosticsEnabled()	boolean
instance()	Options
parseArgs(String[])	void
print(String)	void
printDebugging(String)	void
printUsage()	void

Walkthrough of the SearchTaskGangTest Class



```java
    enum TestsToRun {
        ONESHOT_THREAD_PER_TASK,
        ONESHOT_EXECUTOR_SERVICE,
        ONESHOT_EXECUTOR_SERVICE_FUTURE,
        ONESHOT_EXECUTOR_COMPLETION_SERVICE
    }

    /**
     * This is the entry point into the test program.
     */
    public static void main(String[] args) {
        Options.instance().parseArgs( argv: args);

        print("Starting TaskGangTest");

        // Iterate through all the tests.
        for (var test : TestsToRun.values()) {
            print("Starting " + test);

            // Create/run the appropriate type of SearchTaskGang to
            // search for words concurrently.
            makeTaskGang(sWordList,  choice: test).run();
```

See SearchTaskGang/src/main/java/SearchTaskGangTest.java

End of Overview of Search TaskGang Case Study