The Java Executor Framework: The Java Executors Class

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

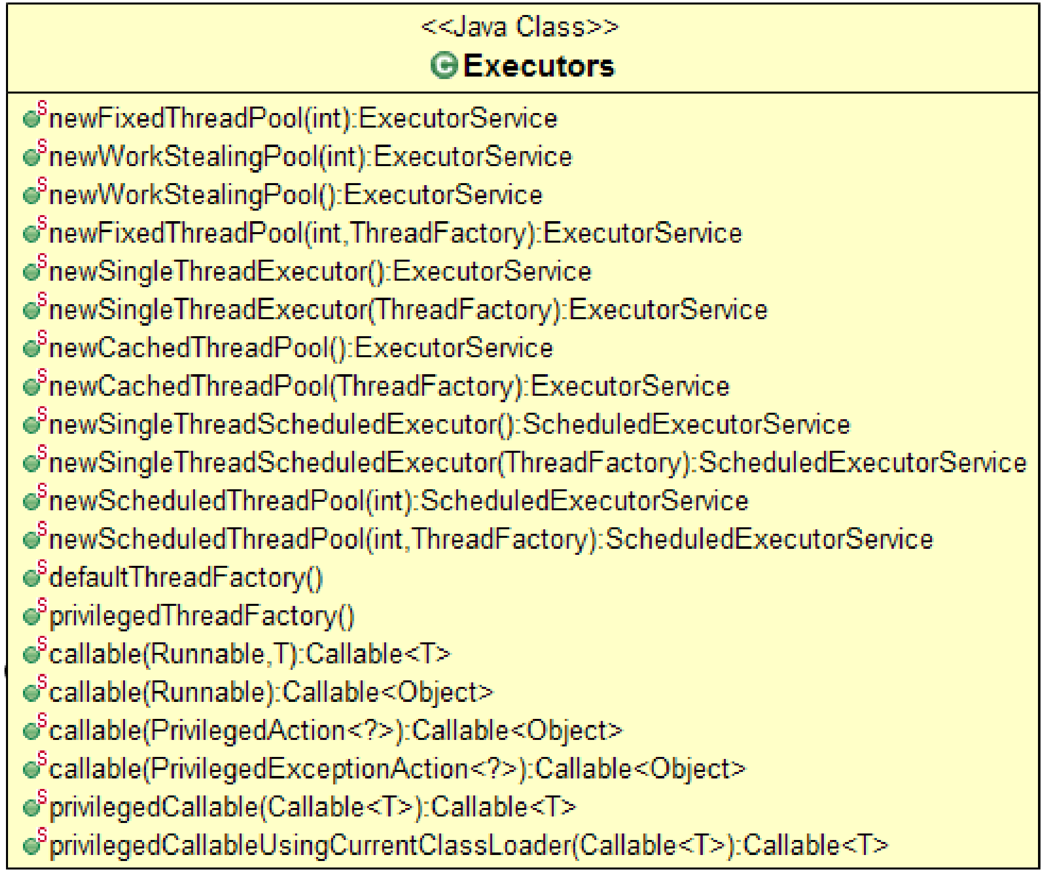
Professor of Computer Science

Institute for Software
Integrated Systems

Vanderbilt University
Nashville, Tennessee, USA

Learning Objectives in this Part of the Lesson

- Understand the purpose of the Java executor framework
- Recognize the features & benefits of thread pools
- Note a human known use of thread pools
- Know the Java Executor framework thread pools
- Learn the key interfaces the framework provides
- Appreciate the factory methods provided by the Java Executors class

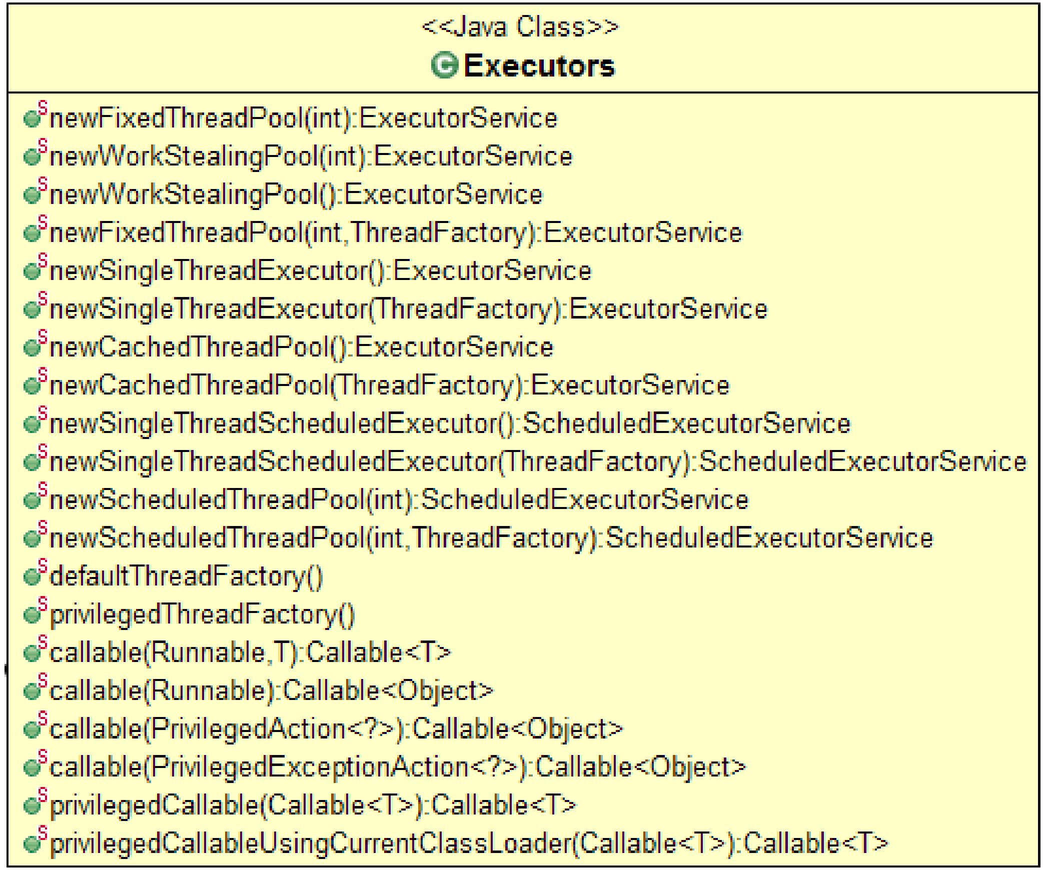


```
<<Java Class>>
Executors
newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>
```

The Java Executors Class

The Java Executors Class

- Executors is a utility class that creates executor implementations



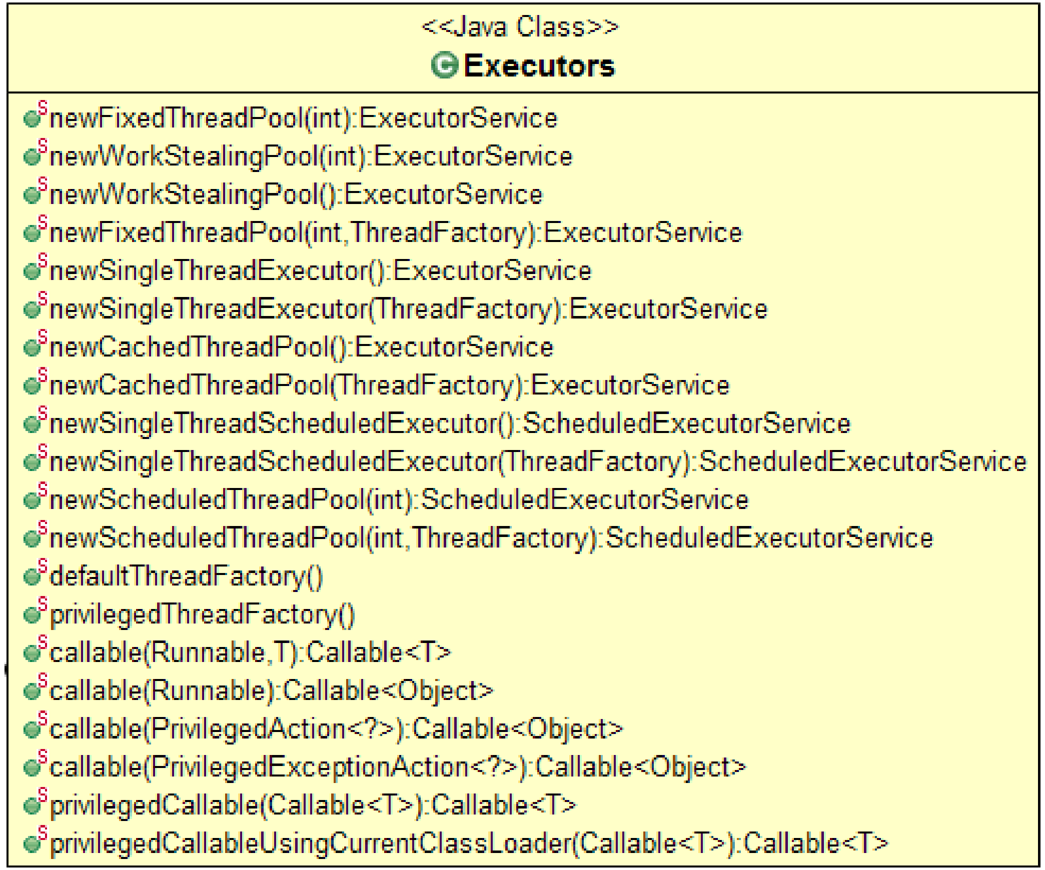
<<Java Class>>
Executors

newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/Executors.html

The Java Executors Class

- Executors is a utility class that creates executor implementations
  - A utility class is a final class having only static methods, no (non-static) state, & a private constructor



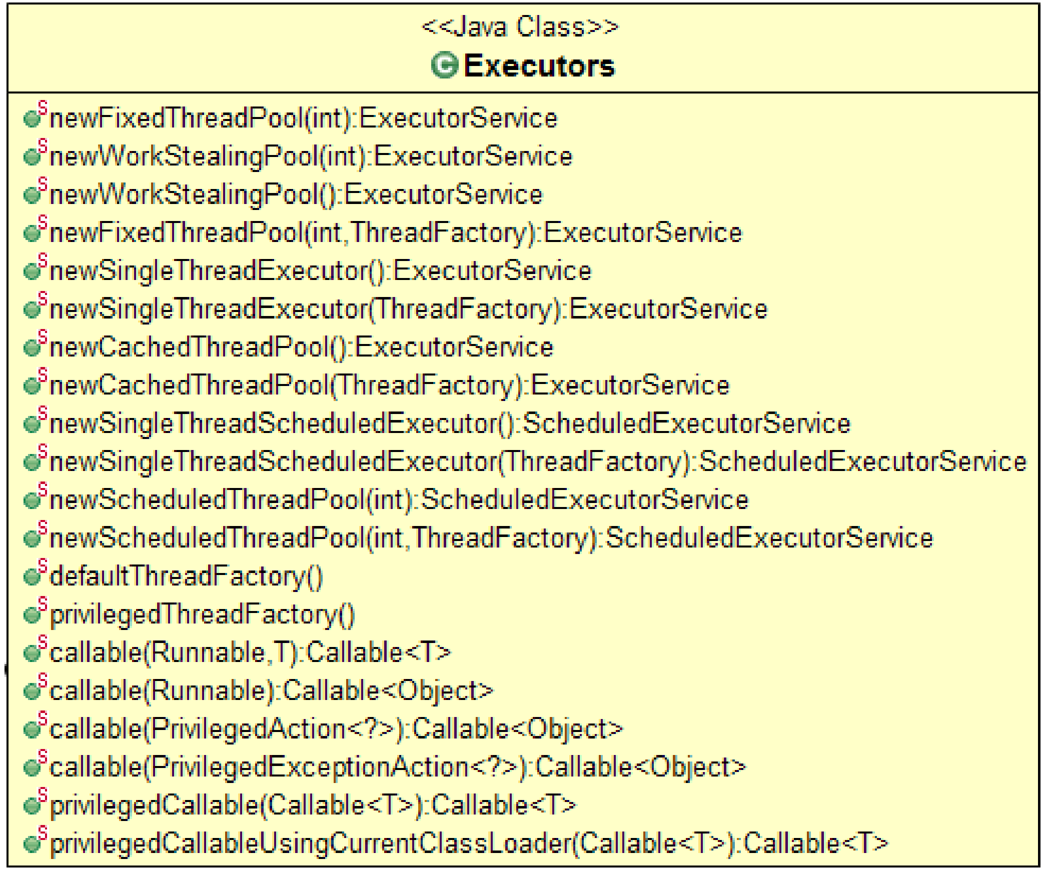
<<Java Class>>
Executors

```
newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>
```

See www.quora.com/What-is-the-best-way-to-write-utility-classes-in-Java/answer/Jon-Harley

The Java Executors Class

- Executors is a utility class that creates executor implementations
  - A utility class is a final class having only static methods, no (non-static) state, & a private constructor
    - Utility classes are largely a vestige of early versions of Java that lacked default methods in interfaces



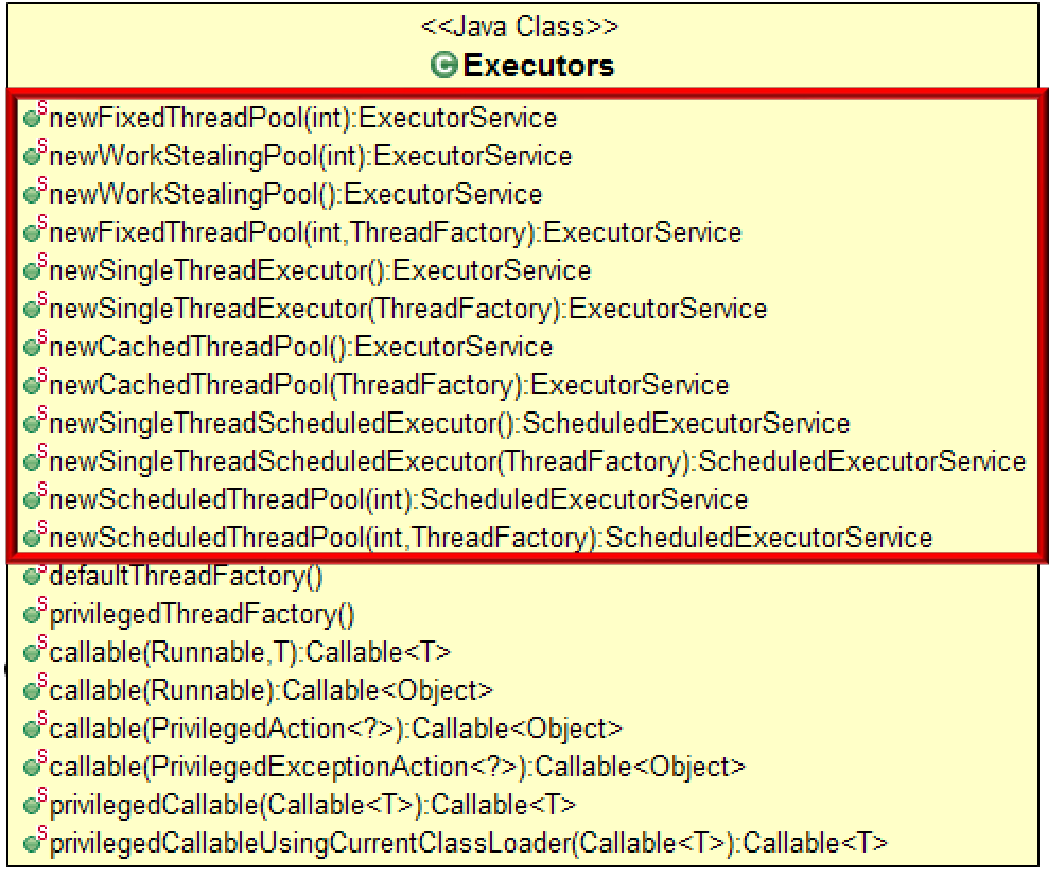
<<Java Class>>
Executors

```
newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>
```

See www.vojtechruzicka.com/avoid-utility-classes

The Java Executors Class

- The Executors utility class has factory methods that create desired executors



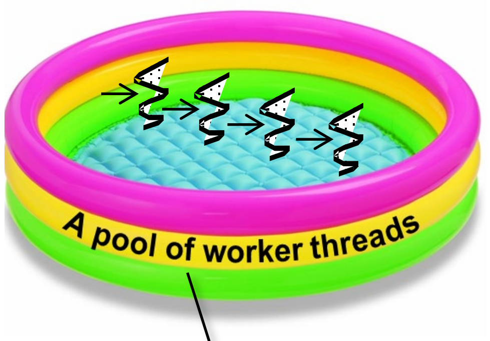
<<Java Class>>
Executors

```
newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>
```

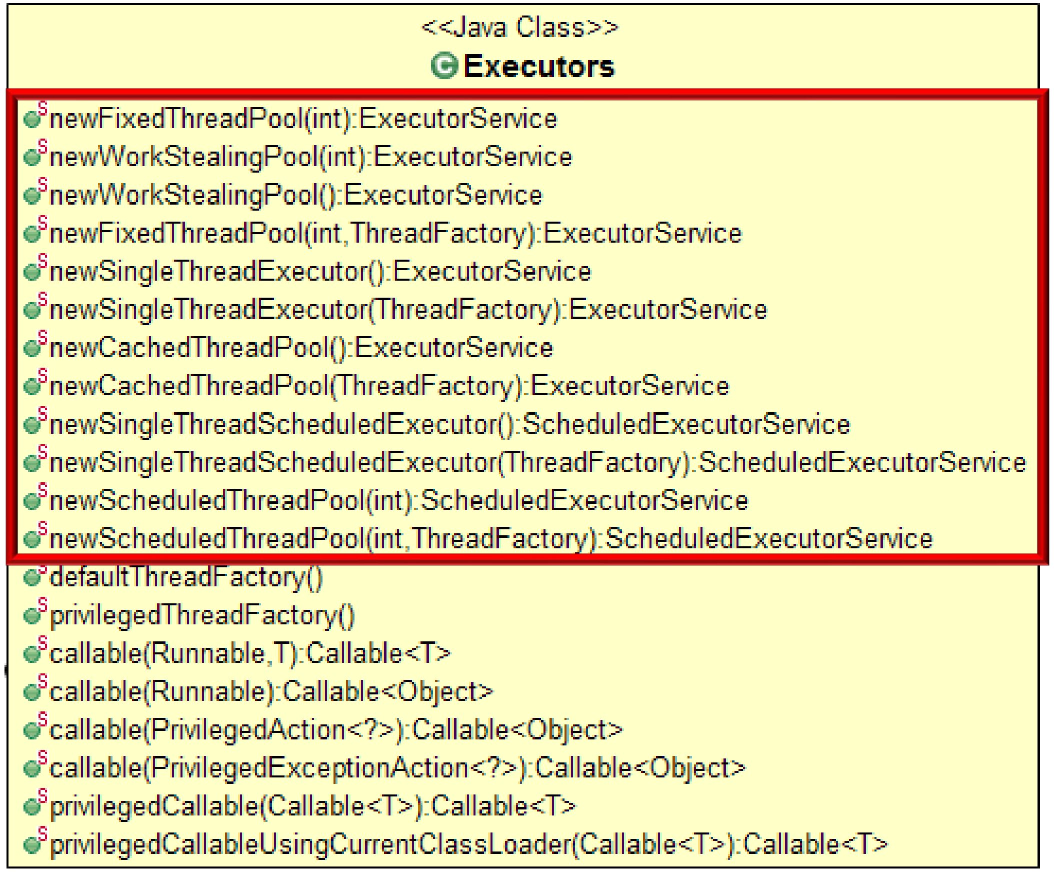
See en.wikipedia.org/wiki/Factory_method_pattern

The Java Executors Class

- The Executors utility class has factory methods that create desired executors



e.g., cached, fixed, work-stealing thread pools, etc.



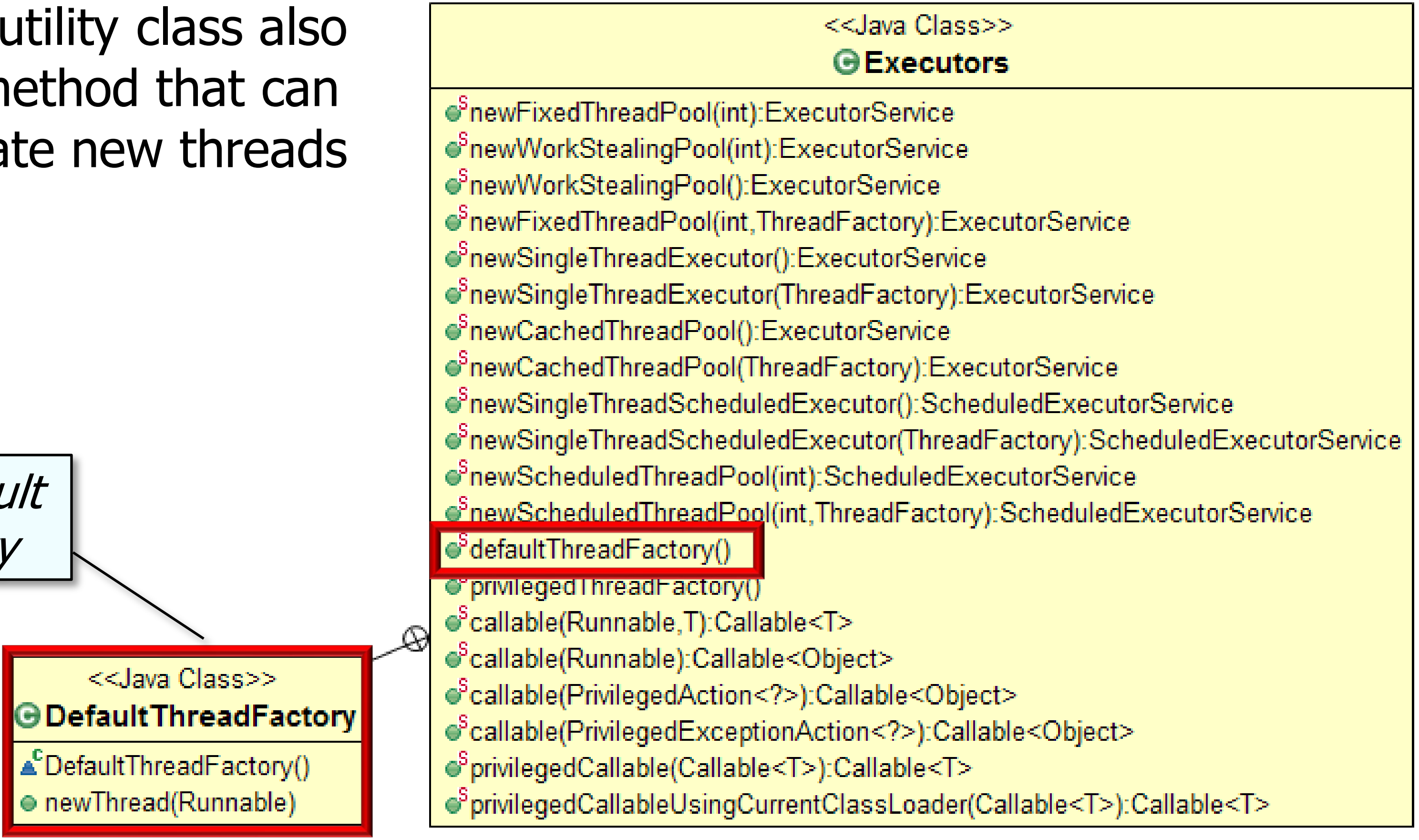
<<Java Class>>
Executors

```
newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>
```

The Java Executors Class

- The Executors utility class also has a factory method that can be used to create new threads

There's a default thread factory

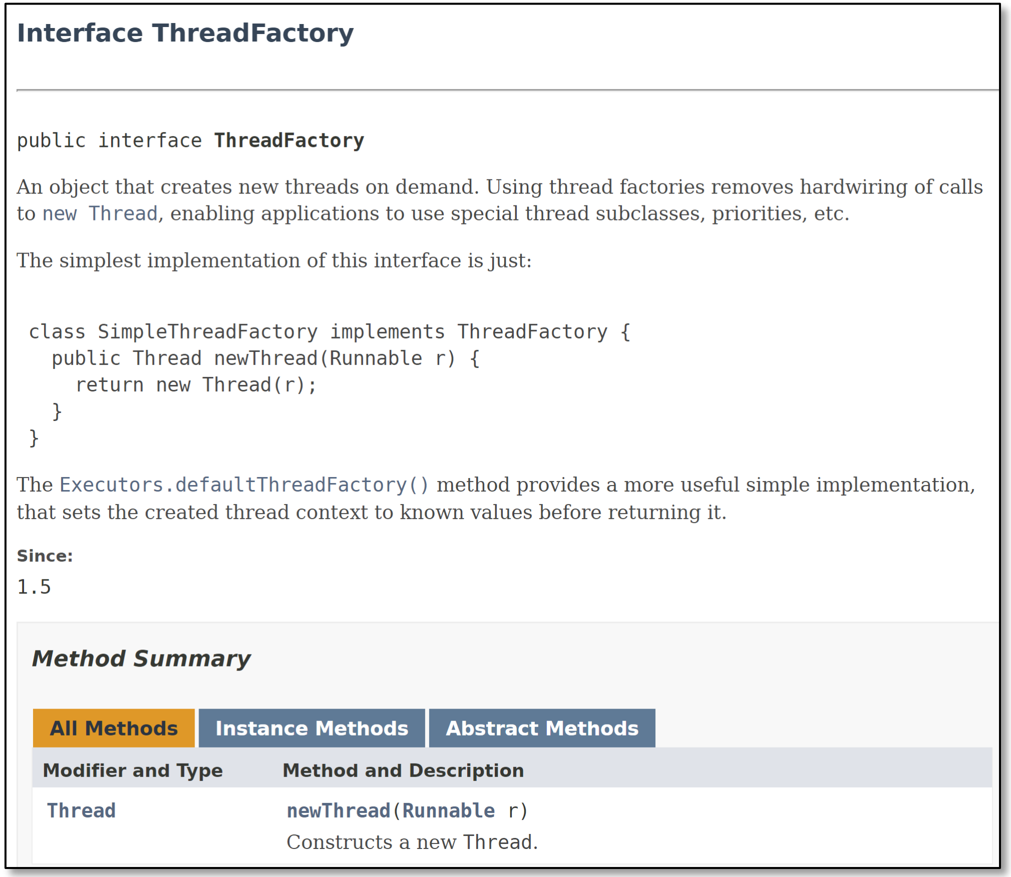


<<Java Class>>
Executors
newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>
<<Java Class>>
DefaultThreadFactory
DefaultThreadFactory()
newThread(Runnable)

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/Executors.html#defaultThreadFactory

The Java Executors Class

- The Executors utility class also has a factory method that can be used to create new threads
  - The DefaultThreadFactory implements the Thread Factory interface



Interface ThreadFactory

`public interface ThreadFactory`

An object that creates new threads on demand. Using thread factories removes hardwiring of calls to `new Thread`, enabling applications to use special thread subclasses, priorities, etc.

The simplest implementation of this interface is just:

```
 class SimpleThreadFactory implements ThreadFactory {
   public Thread newThread(Runnable r) {
     return new Thread(r);
   }
 }
```

The `Executors.defaultThreadFactory()` method provides a more useful simple implementation, that sets the created thread context to known values before returning it.

Since:
1.5

Method Summary

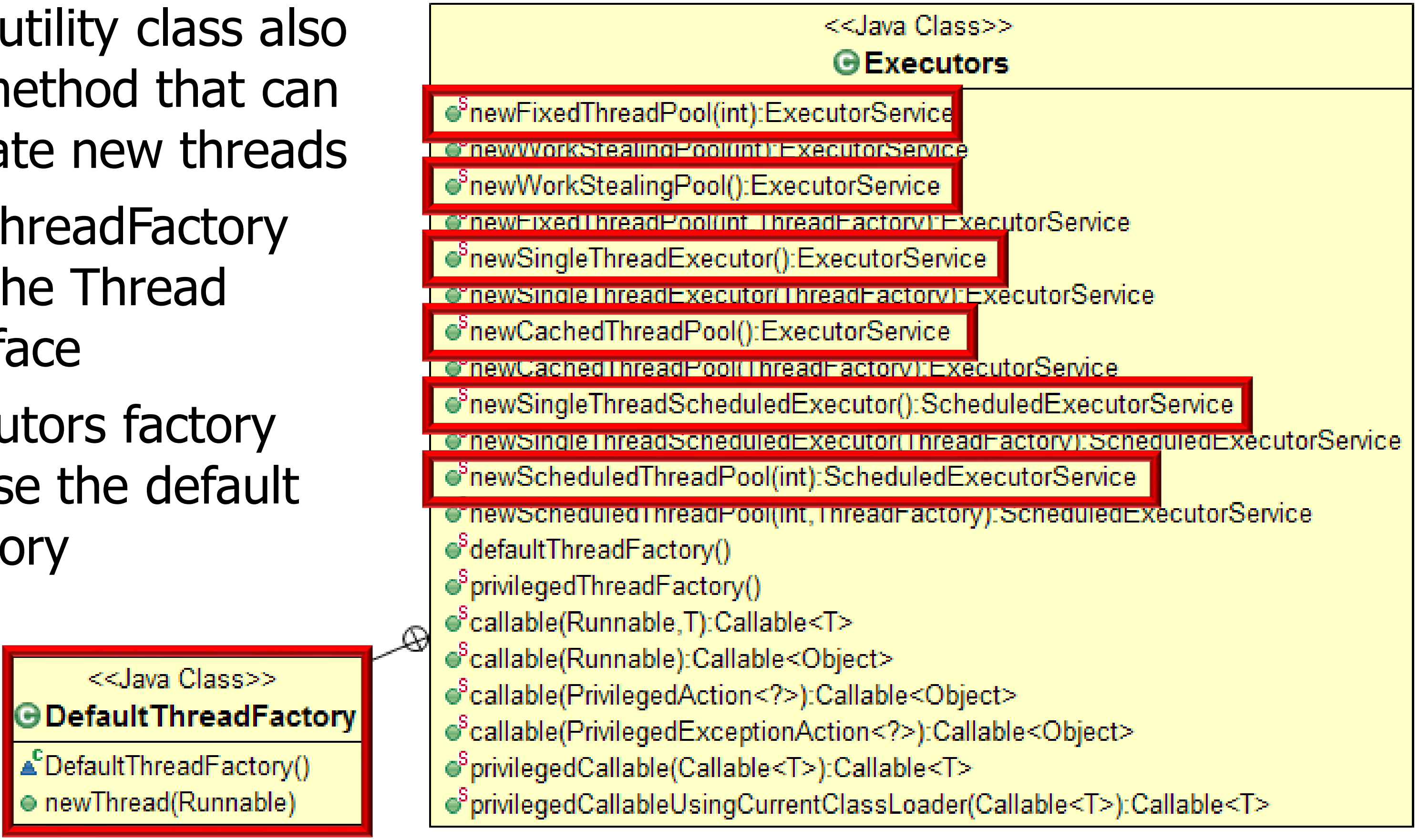
All Methods **Instance Methods** **Abstract Methods**

Modifier and Type	Method and Description
Thread	newThread(Runnable r) Constructs a new Thread.

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ThreadFactory.html

The Java Executors Class

- The Executors utility class also has a factory method that can be used to create new threads
  - The DefaultThreadFactory implements the Thread Factory interface
    - Many Executors factory methods use the default thread factory


<<Java Class>>
Executors

- newFixedThreadPool(int):ExecutorService
- newWorkStealingPool(int):ExecutorService
- newWorkStealingPool():ExecutorService
- newFixedThreadPool(int, ThreadFactory):ExecutorService
- newSingleThreadExecutor():ExecutorService
- newSingleThreadExecutor(ThreadFactory):ExecutorService
- newCachedThreadPool():ExecutorService
- newCachedThreadPool(ThreadFactory):ExecutorService
- newSingleThreadScheduledExecutor():ScheduledExecutorService
- newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
- newScheduledThreadPool(int):ScheduledExecutorService
- newScheduledThreadPool(int, ThreadFactory):ScheduledExecutorService
- defaultThreadFactory()
- privilegedThreadFactory()
- callable(Runnable,T):Callable<T>
- callable(Runnable):Callable<Object>
- callable(PrivilegedAction<?>):Callable<Object>
- callable(PrivilegedExceptionAction<?>):Callable<Object>
- privilegedCallable(Callable<T>):Callable<T>
- privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>

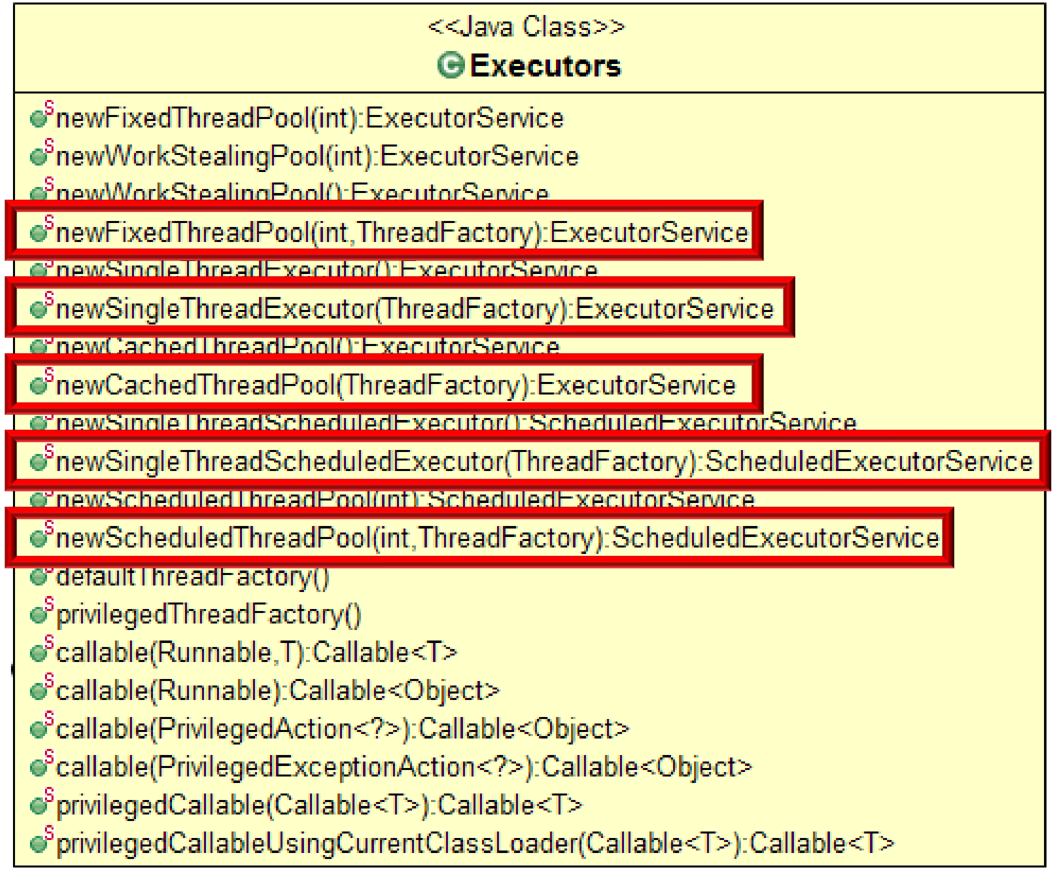
<<Java Class>>
DefaultThreadFactory

- DefaultThreadFactory()
- newThread(Runnable)

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/Executors.html#defaultThreadFactory

The Java Executors Class

- The Executors utility class also has a factory method that can be used to create new threads
  - The DefaultThreadFactory implements the Thread Factory interface
  - You can also define custom thread factories & pass them to factory methods



<<Java Class>>
Executors

```
newFixedThreadPool(int):ExecutorService
newWorkStealingPool(int):ExecutorService
newWorkStealingPool():ExecutorService
newFixedThreadPool(int,ThreadFactory):ExecutorService
newSingleThreadExecutor():ExecutorService
newSingleThreadExecutor(ThreadFactory):ExecutorService
newCachedThreadPool():ExecutorService
newCachedThreadPool(ThreadFactory):ExecutorService
newSingleThreadScheduledExecutor():ScheduledExecutorService
newSingleThreadScheduledExecutor(ThreadFactory):ScheduledExecutorService
newScheduledThreadPool(int):ScheduledExecutorService
newScheduledThreadPool(int,ThreadFactory):ScheduledExecutorService
defaultThreadFactory()
privilegedThreadFactory()
callable(Runnable,T):Callable<T>
callable(Runnable):Callable<Object>
callable(PrivilegedAction<?>):Callable<Object>
callable(PrivilegedExceptionAction<?>):Callable<Object>
privilegedCallable(Callable<T>):Callable<T>
privilegedCallableUsingCurrentClassLoader(Callable<T>):Callable<T>
```

See howtodoinjava.com/java/multi-threading/creating-threads-using-java-util-concurrent-threadfactory

End of the Java Executors Framework: The Java Executors Interface