Key Methods in Java ForkJoinPool

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

- Recognize the key methods in the ForkJoinPool class

<table>
<thead>
<tr>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>ForkJoinPool()</code></td>
</tr>
<tr>
<td><code>ForkJoinPool(int)</code></td>
</tr>
<tr>
<td><code>ForkJoinPool(int, ForkJoinWorkerThreadFactory, UncaughtExceptionHandler, boolean)</code></td>
</tr>
<tr>
<td><code>commonPool()</code>: ForkJoinPool</td>
</tr>
<tr>
<td><code>invoke(ForkJoinTask&lt;T&gt;)</code></td>
</tr>
<tr>
<td><code>execute(ForkJoinTask&lt;&gt;)</code>: void</td>
</tr>
<tr>
<td><code>execute(Runnable): void</code></td>
</tr>
<tr>
<td><code>submit(ForkJoinTask&lt;&gt;)</code>: ForkJoinTask&lt;T&gt;</td>
</tr>
<tr>
<td><code>submit(Callable&lt;T&gt;): ForkJoinTask&lt;T&gt;</code></td>
</tr>
<tr>
<td><code>submit(Runnable,T): ForkJoinTask&lt;T&gt;</code></td>
</tr>
<tr>
<td><code>submit(Runnable): ForkJoinTask&lt;T&gt;</code></td>
</tr>
<tr>
<td><code>invokeAll(Collection&lt;Callable&lt;T&gt;&gt;,): List&lt;Future&lt;T&gt;&gt;</code></td>
</tr>
<tr>
<td><code>shutdown(): void</code></td>
</tr>
<tr>
<td><code>shutdownNow(): List&lt;Runnable&gt;</code></td>
</tr>
<tr>
<td><code>isTerminated(): boolean</code></td>
</tr>
<tr>
<td><code>isTerminating(): boolean</code></td>
</tr>
<tr>
<td><code>isShutdown(): boolean</code></td>
</tr>
<tr>
<td><code>awaitTermination(long, TimeUnit): boolean</code></td>
</tr>
</tbody>
</table>
Key Methods in Java ForkJoinPool
Key Methods in Java ForkJoinPool

- ForkJoinPool extends AbstractExecutorService

```java
class ForkJoinPool extends AbstractExecutorService {
    ...
    void execute(Runnable cmd){...}

    <T> Future<T> submit
       (Callable<T> task){...}

    <T> List<Future<T>> invokeAll
       (Collection<? extends Callable<T>> tasks){...}

    <T> T invokeAny
       (Collection<? extends Callable<T>> tasks){...}
}
```

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html
ForkJoinPool extends AbstractExecutorService
It therefore implements the ExecutorService methods

```java
class ForkJoinPool extends AbstractExecutorService {
    ...
    void execute(Runnable cmd){...}

    <T> Future<T> submit
    (Callable<T> task){...}

    <T> List<Future<T>> invokeAll
    (Collection<? extends Callable<T>> tasks){...}

    <T> T invokeAny
    (Collection<? extends Callable<T>> tasks){...}
```
ForkJoinPool extends AbstractExecutorService

It therefore implements the ExecutorService methods

Arrange async execution of a one-way task

Key Methods in Java ForkJoinPool

class ForkJoinPool extends AbstractExecutorService {

    ... 

    void execute(Runnable cmd) {...}

    <T> Future<T> submit
        (Callable<T> task) {...}

    <T> List<Future<T>> invokeAll
        (Collection<? extends Callable<T>> tasks) {...}

    <T> T invokeAny
        (Collection<? extends Callable<T>> tasks) {...}

}
ForkJoinPool extends AbstractExecutorService

- It therefore implements the ExecutorService methods
  - Arrange async execution of a one-way task
  - Submit a two-way task for execution, return a future

```java
class ForkJoinPool extends AbstractExecutorService {
    ... 
    void execute(Runnable cmd) {...}

    <T> Future<T> submit (Callable<T> task) {...}

    <T> List<Future<T>> invokeAll (Collection<? extends Callable<T>> tasks) {...}

    <T> T invokeAny (Collection<? extends Callable<T>> tasks) {...}
}
```
ForkJoinPool extends AbstractExecutorService

It therefore implements the ExecutorService methods

• Arrange async execution of a one-way task
• Submit a two-way task for execution, return a future
• Run all tasks in the collection & wait for them all to finish

Key Methods in Java ForkJoinPool

class ForkJoinPool extends AbstractExecutorService {
    ... 
    void execute(Runnable cmd){...}

    <T> Future<T> submit
        (Callable<T> task){...}

    <T> List<Future<T>> invokeAll
        (Collection<? extends Callable<T>> tasks){...}

    <T> T invokeAny
        (Collection<? extends Callable<T>> tasks){...}

ForkJoinPool extends AbstractExecutorService

It therefore implements the ExecutorService methods

- Arrange async execution of a one-way task
- Submit a two-way task for execution, return a future
- Run all tasks in the collection & wait for them all to finish
- Run all tasks in the collection & wait for the first to finish

```java
class ForkJoinPool extends AbstractExecutorService {
    ... 
    void execute(Runnable cmd){...}

    <T> Future<T> submit (Callable<T> task){...}

    <T> List<Future<T>> invokeAll (Collection<? extends Callable<T>> tasks){...}

    <T> T invokeAny (Collection<? extends Callable<T>> tasks){...}
```
Key Methods in Java ForkJoinPool

- ForkJoinPool extends AbstractExecutorService
- It therefore implements the ExecutorService methods

```java
class ForkJoinPool extends AbstractExecutorService {
    ... 
    void execute(Runnable cmd){...}

    <T> Future<T> submit (Callable<T> task){...}

    <T> List<Future<T>> invokeAll (Collection<? extends Callable<T>> tasks){...}

    <T> T invokeAny (Collection<? extends Callable<T>> tasks){...}
```

However, these methods don’t directly leverage powerful fork-join features!
ForkJoinPool extends Abstract ExecutorService

- It therefore implements the ExecutorService methods
- It also implements key methods for non-ForkJoinTask clients

```java
class ForkJoinPool extends AbstractExecutorService {
    ...
    void execute(ForkJoinTask<Ｔ> task) {
        ...
    }

    T invoke(ForkJoinTask<Ｔ> task) {
        ...
    }

    ForkJoinTask<Ｔ> submit(ForkJoinTask<Ｔ> task) {
        ...
    }
}
```
ForkJoinPool extends AbstractExecutorService

- It therefore implements the ExecutorService methods
- It also implements key methods for non-ForkJoinTask clients

```java
class ForkJoinPool extends AbstractExecutorService {
    ...
    void execute(ForkJoinTask<T> task)
    {
        ...
    }

    T invoke(ForkJoinTask<T> task)
    {
        ...
    }

    ForkJoinTask<T> submit(ForkJoinTask<T> task)
    {
        ...
    }
```

These methods *can* directly leverage powerful fork-join features
Key Methods in Java ForkJoinPool

- ForkJoinPool extends AbstractExecutorService
  - It therefore implements the ExecutorService methods
- It also implements key methods for non-ForkJoinTask clients
  - Arrange async execution of one-way task

```java
class ForkJoinPool extends AbstractExecutorService {
    ...
    void execute(ForkJoinTask<T> task)
    { ... }

    T invoke(ForkJoinTask<T> task)
    { ... }

    ForkJoinTask<T> submit(ForkJoinTask<T> task)
    { ... }
}
```

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#execute](docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#execute)
Key Methods in Java ForkJoinPool

- ForkJoinPool extends AbstractExecutorService
  - It therefore implements the ExecutorService methods
- It also implements key methods for non-ForkJoinTask clients
  - Arrange async execution of one-way task
  - Perform the task, blocking until it completes

```
class ForkJoinPool extends AbstractExecutorService {
    ...
    void execute(ForkJoinTask<T> task)
    {
        ...
    }

    T invoke(ForkJoinTask<T> task)
    {
        ...
    }

    ForkJoinTask<T> submit(ForkJoinTask<T> task)
    {
        ...
    }
}
```

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#invoke
Key Methods in Java ForkJoinPool

- ForkJoinPool extends AbstractExecutorService
  - It therefore implements the ExecutorService methods
- It also implements key methods for non-ForkJoinTask clients
  - Arrange async execution of one-way task
  - Perform the task, blocking until it completes
  - Submit a ForkJoinTask for async execution, return a future

```java
class ForkJoinPool extends AbstractExecutorService {
    ... 
    void execute(ForkJoinTask<T> task)
    { ... }

    T invoke(ForkJoinTask<T> task)
    { ... }

    ForkJoinTask<T> submit(ForkJoinTask<T> task)
    { ... }

    // TWO WAY

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#submit
Key Methods in Java ForkJoinPool

- The ForkJoinPool size defaults to # of cores available to Java runtime.

```java
class ForkJoinPool extends AbstractExecutorService {
    public ForkJoinPool() {
        this(Math.min(MAX_CAP, Runtime.getRuntime().availableProcessors()), ...);
    }

    public ForkJoinPool (int parallelism) {
        this(parallelism, ...);
    }

    ...
}
```

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#ForkJoinPool
The ForkJoinPool size defaults to # of cores available to Java runtime

```java
class ForkJoinPool extends AbstractExecutorService {
    public ForkJoinPool() {
        this(Math.min(MAX_CAP,
                     Runtime.getRuntime()
                     .availableProcessors()),
             ...);
    }

    public ForkJoinPool
        (int parallelism) {
        this(parallelism, ...);
    }

    ...}
```

Returns # of processor cores available to the Java execution environment

See docs.oracle.com/javase/8/docs/api/java/lang/Runtime.html#availableProcessors
Key Methods in Java ForkJoinPool

- The ForkJoinPool size defaults to # of cores available to Java runtime
- This size can also be controlled programmatically via the ForkJoinPool constructor

```java
class ForkJoinPool extends AbstractExecutorService {
    public ForkJoinPool() {
        this(Math.min(MAX_CAP, Runtime.getRuntime().availableProcessors()), ...);
    }

    public ForkJoinPool (int parallelism) {
        this(parallelism, ...);
    }

    ...
}

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#ForkJoinPool
```
Key Methods in Java ForkJoinPool

- The common fork-join pool can be accessed via a static method

```java
class ForkJoinPool extends AbstractExecutorService {
    // ...
    static final ForkJoinPool common;

    public static ForkJoinPool commonPool() {
        return common;
    }
}
```

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#commonPool](docs.oracle.com/javase/8/docs/api/java/util/concurrent/ForkJoinPool.html#commonPool)
• The common fork-join pool can be accessed via a static method

```java
class ForkJoinPool extends AbstractExecutorService {
    
    static final ForkJoinPool common;
    
    public static ForkJoinPool commonPool() {
        return common;
    }
}
```

This method accesses a static field that can be accessed via all threads in a process
The common fork-join pool can be accessed via a static method.

The common pool is used by any ForkJoinTask that is not explicitly submitted to a specified pool.

```java
class ForkJoinPool extends AbstractExecutorService {
    static final ForkJoinPool common;
    public static ForkJoinPool commonPool() {
        return common;
    }
```
ForkJoinPool also provides various management & monitoring operations

<table>
<thead>
<tr>
<th>Type</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>getParallelism()</code></td>
<td>Returns the targeted parallelism level of this pool</td>
</tr>
<tr>
<td>int</td>
<td><code>getPoolSize()</code></td>
<td>Returns the number of worker threads that have started but not yet terminated</td>
</tr>
<tr>
<td>int</td>
<td><code>getQueuedSubmissionCount()</code></td>
<td>Returns an estimate of the number of tasks submitted to this pool that have not yet begun executing</td>
</tr>
<tr>
<td>long</td>
<td><code>getStealCount()</code></td>
<td>Returns an estimate of the total number of tasks stolen from one thread's work queue by another</td>
</tr>
</tbody>
</table>
• ForkJoinPool also provides various management & monitoring operations

<table>
<thead>
<tr>
<th>Type</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td><code>getParallelism()</code></td>
<td>Returns the targeted parallelism level of this pool</td>
</tr>
<tr>
<td>int</td>
<td><code>getPoolSize()</code></td>
<td>Returns the number of worker threads that have started but not yet terminated</td>
</tr>
<tr>
<td>int</td>
<td><code>getQueuedSubmissionCount()</code></td>
<td>Returns an estimate of the number of tasks submitted to this pool that have not yet begun executing</td>
</tr>
<tr>
<td>long</td>
<td><code>getStealCount()</code></td>
<td>Returns an estimate of the total number of tasks stolen from one thread's work queue by another</td>
</tr>
</tbody>
</table>
ForkJoinPool also provides various management & monitoring operations

<table>
<thead>
<tr>
<th>Type</th>
<th>Method Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>int</td>
<td>getParallelism()</td>
<td>Returns the targeted parallelism level of this pool</td>
</tr>
<tr>
<td>int</td>
<td>getPoolSize()</td>
<td>Returns the number of worker threads that have started but not yet terminated</td>
</tr>
<tr>
<td>int</td>
<td>getQueuedSubmissionCount()</td>
<td>Returns an estimate of the number of tasks submitted to this pool that have not yet begun executing</td>
</tr>
<tr>
<td>long</td>
<td>getStealCount()</td>
<td>Returns an estimate of the total number of tasks stolen from one thread's work queue by another</td>
</tr>
</tbody>
</table>
ForkJoinPool also provides various management & monitoring operations

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>getParallelism()</code></td>
<td>Returns the targeted parallelism level of this pool</td>
</tr>
<tr>
<td><code>getPoolSize()</code></td>
<td>Returns the number of worker threads that have started but not yet terminated</td>
</tr>
<tr>
<td><code>getQueuedSubmissionCount()</code></td>
<td>Returns an estimate of the number of tasks submitted to this pool that have not yet begun executing</td>
</tr>
<tr>
<td><code>getStealCount()</code></td>
<td>Returns an estimate of the total number of tasks stolen from one thread's work queue by another</td>
</tr>
</tbody>
</table>

Key Methods in Java ForkJoinPool
• ForkJoinPool also provides various management & monitoring operations

<table>
<thead>
<tr>
<th>Method</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>int getParallelism()</code></td>
<td>Returns the targeted parallelism level of this pool</td>
</tr>
<tr>
<td><code>int getPoolSize()</code></td>
<td>Returns the number of worker threads that have started but not yet terminated</td>
</tr>
<tr>
<td><code>int getQueuedSubmissionCount()</code></td>
<td>Returns an estimate of the number of tasks submitted to this pool that have not yet begun executing</td>
</tr>
<tr>
<td><code>long getStealCount()</code></td>
<td>Returns an estimate of the total number of tasks stolen from one thread's work queue by another</td>
</tr>
</tbody>
</table>
End of Key Methods in Java ForkJoinPool