Key Methods in the Java Recursive Action & RecursiveTask Subclasses

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

• Recognize the key methods in the ForkJoinTask class

• Recognize the key methods in the RecursiveAction & RecursiveTask classes
Key Methods in the Java RecursiveAction
RecursiveAction extends ForkJoinTask & does not return a result

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

- RecursiveAction extends ForkJoinTask & does not return a result
- Subclasses override compute() to perform task’s main computation

```
abstract class RecursiveAction
    extends ForkJoinTask<Void> {
    protected abstract Void compute();
    ...
```

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/RecursiveTask.html#compute](http://docs.oracle.com/javase/8/docs/api/java/util/concurrent/RecursiveTask.html#compute)
• RecursiveAction extends ForkJoinTask & does not return a result
• Subclasses override compute() to perform task’s main computation
• If data size is below a certain threshold perform work directly

abstract class RecursiveAction extends ForkJoinTask<Void> {
    protected abstract Void compute();
    ...
}
RecursiveAction extends ForkJoin Task & does not return a result

- Subclasses override compute() to perform task’s main computation
  - If data size is below a certain threshold perform work directly
  - If data size is large, split work into smaller sub-tasks that are fork()’d to run in parallel

```java
class RecursiveAction extends ForkJoinTask<Void> {
    protected abstract Void compute();
    ...
```
RecursiveAction extends ForkJoinTask & does not return a result

- Subclasses override compute() to perform task’s main computation
  - If data size is below a certain threshold perform work directly
  - If data size is large, split work into smaller sub-tasks that are fork()’d to run in parallel
  - These smaller sub-tasks are join()’d, but a result is not returned directly
    - e.g., results may be stored in an array

```
abstract class RecursiveAction extends ForkJoinTask<Void> {
  protected abstract Void compute();
  ...
```
Key Methods in Java RecursiveAction

- RecursiveAction extends ForkJoin Task & does not return a result
  - Subclasses override compute() to perform task’s main computation
- The fork-join framework calls exec() to execute the task

```java
abstract class RecursiveAction extends ForkJoinTask<Void> {
    protected abstract Void compute();

    protected final boolean exec() {
        compute();
        return true;
    }
    ...
```

exec() is an abstract method in ForkJoin Task that’s overridden in RecursiveAction

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/RecursiveTask.html#exec
RecursiveAction extends ForkJoin Task & does not return a result

- Subclasses override compute() to perform task’s main computation

- The fork-join framework calls exec() to execute the task

See en.wikipedia.org/wiki/Template_method_pattern
RecursiveAction extends ForkJoin Task & does not return a result

- Subclasses override compute() to perform task’s main computation
- The fork-join framework calls exec() to execute the task

The result of compute() is not stored for subsequent access
Key Methods in the Java RecursiveTask
RecursiveTask extends ForkJoinTask to return a result.

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

- RecursiveTask extends ForkJoinTask to return a result
- Subclasses override compute() to perform task’s main computation

```java
abstract class RecursiveTask<V>
    extends ForkJoinTask<V> {
    protected abstract V compute();
    ...
}
```

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/RecursiveTask.html#compute
• RecursiveTask extends ForkJoinTask to return a result
  
• Subclasses override compute() to perform task’s main computation

  • If data size is below a certain threshold perform work directly
RecursiveTask extends ForkJoinTask to return a result

Subclasses override compute() to perform task’s main computation

- If data size is below a certain threshold perform work directly
- If data size is large, split work into smaller sub-tasks that are fork()’d to run in parallel

abstract class RecursiveTask<V>
extends ForkJoinTask<V> {
    protected abstract V compute();
    ...
}
RecursiveTask extends ForkJoinTask to return a result

- Subclasses override compute() to perform task’s main computation
  - If data size is below a certain threshold perform work directly
  - If data size is large, split work into smaller sub-tasks that are fork()’d to run in parallel
  - Results of these smaller sub-tasks are join()’d into a merged result

abstract class RecursiveTask<V> extends ForkJoinTask<V> {
    protected abstract V compute();
    ...
}
Key Methods in Java RecursiveTask

- RecursiveTask extends ForkJoinTask to return a result
  - Subclasses override compute() to perform task’s main computation
  - The fork-join framework calls exec() to execute the task

abstract class RecursiveTask<V>
  extends ForkJoinTask<V> {
    protected abstract V compute();

    V result;

    protected final boolean exec() {
      result = compute();
      return true;
    }

    ...

exec() is an abstract (template) method in ForkJoinTask that’s overridden in RecursiveTask

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/RecursiveTask.html#exec
RecursiveTask extends ForkJoinTask to return a result

- Subclasses override compute() to perform task’s main computation
- The fork-join framework calls exec() to execute the task

abstract class RecursiveTask<V> extends ForkJoinTask<V> {
    protected abstract V compute();

    V result;

    protected final boolean exec() {
        result = compute();
        return true;
    }

    ...

The result of compute() is stored for subsequent access
End of Key Methods in the Java RecursiveAction & RecursiveTask Subclasses