Java ExecutorService: Introduction

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 powerful features defined in the Java ExecutorService interface
Overview of the ExecutorService Interface
Overview of the ExecutorService Interface

- Extends Executor

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ExecutorService.html
Overview of the ExecutorService Interface

- Extends Executor
- Submit 1+ tasks & return futures for these tasks

See docs.oracle.com/javase/8/docs/api/java/util/concurrent/ExecutorService.html
Overview of the ExecutorService Interface

- Extends Executor
  - Submit 1+ tasks & return futures for these tasks
- Manage lifecycle of tasks & executor service itself
  - e.g., interrupts worker threads in a pool

See [docs.oracle.com/javase/8/docs/api/java/util/concurrent/ExecutorService.html](docs.oracle.com/javase/8/docs/api/java/util/concurrent/ExecutorService.html)
Overview of the ExecutorService Interface

- A task is a unit of computation that (ideally) does not depend on the state, result, or side effects of other tasks

Overview of the ExecutorService Interface

• A task is a unit of computation that (ideally) does not depend on the state, result, or side effects of other tasks

• A task has four phases in its lifecycle

1. Created

2. Submitted

3. Started

4. Completed
Overview of the ExecutorService Interface

- A task is a unit of computation that (ideally) does not depend on the state, result, or side effects of other tasks
- A task has four phases in its lifecycle

1. Created
   - A new task is instantiated

2. Submitted

3. Started

4. Completed
Overview of the ExecutorService Interface

• A task is a unit of computation that (ideally) does not depend on the state, result, or side effects of other tasks

• A task has four phases in its lifecycle

1. Created

2. Submitted

  • A task is given to an executor service to run

3. Started

4. Completed
A task is a unit of computation that (ideally) does not depend on the state, result, or side effects of other tasks.

A task has four phases in its lifecycle:

1. Created
2. Submitted
3. Started

- A task is executed by a worker thread in the executor service.

4. Completed
Overview of the ExecutorService Interface

- A task is a unit of computation that (ideally) does not depend on the state, result, or side effects of other tasks
- A task has four phases in its lifecycle
  1. Created
  2. Submitted
  3. Started
  4. Completed

- A task is finished (un)successfully or cancelled
End of Java Executor Service: Introduction