Managing the Java Thread Lifecycle: Introduction

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

- Be aware of the Java thread lifecycle

The primarily focus of this overall lesson is on starting & stopping Java threads.
Overview of the States in the Java Thread Lifecycle
Overview of the Java Thread Lifecycle

• A Java thread is a highly complex entity

See docs.oracle.com/javase/8/docs/api/java/lang/Thread.html
Overview of the Java Thread Lifecycle

- A Java thread is a highly complex entity
- A thread interacts with many other entities

See docs.oracle.com/javase/8/docs/api/java/lang/Thread.html
Overview of the Java Thread Lifecycle

- The lifecycle of a thread must therefore be managed carefully
Overview of the Java Thread Lifecycle

- The lifecycle of a thread must therefore be managed carefully.

A thread transitions through various “states” based on program operations.
Overview of the Java Thread Lifecycle

- The lifecycle of a thread must therefore be managed carefully

Two key parts of a Java thread’s lifecycle involve starting & stopping it
Overview of the Java Thread Lifecycle

- You needn’t understand all the details of Java thread internals to program them effectively
Overview of the Java Thread Lifecycle

- You needn’t understand all the details of Java thread internals to program them effectively
- However, your reputation (& value) as a “full stack” developer will increase if you learn how threads work internally

See [www.laurencegellert.com/2012/08/what-is-a-full-stack-developer](http://www.laurencegellert.com/2012/08/what-is-a-full-stack-developer)
Overview of the Java Thread Lifecycle

• You needn’t understand all the details of Java thread internals to program them effectively

• However, your reputation (& value) as a “full stack” developer will increase if you learn how threads work internally

• This type of “systems” knowledge may help you avoid “commoditization”!!

See softwareengineeringdaily.com/2016/08/07/you-are-not-a-commodity
End of Managing the Java Thread Lifecycle: Introduction