

Evaluation of Concurrency & Parallelism Mechanisms in Java

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

- Know which Java mechanism(s) to understand & apply



Which Java Mechanism(s)
to Understand & Apply

Which Java Mechanism(s) to Understand & Apply

- Java's concurrency & parallelism mechanisms span multiple layers in the software stack

Java/JNI

C++/C

C

Applications

Additional Application Frameworks

Concurrency/Parallelism Frameworks
Java Threads & Synchronizers

Execution Environment (JVM, Dalvik/ART, etc.)

System Libraries

Operating System Kernel

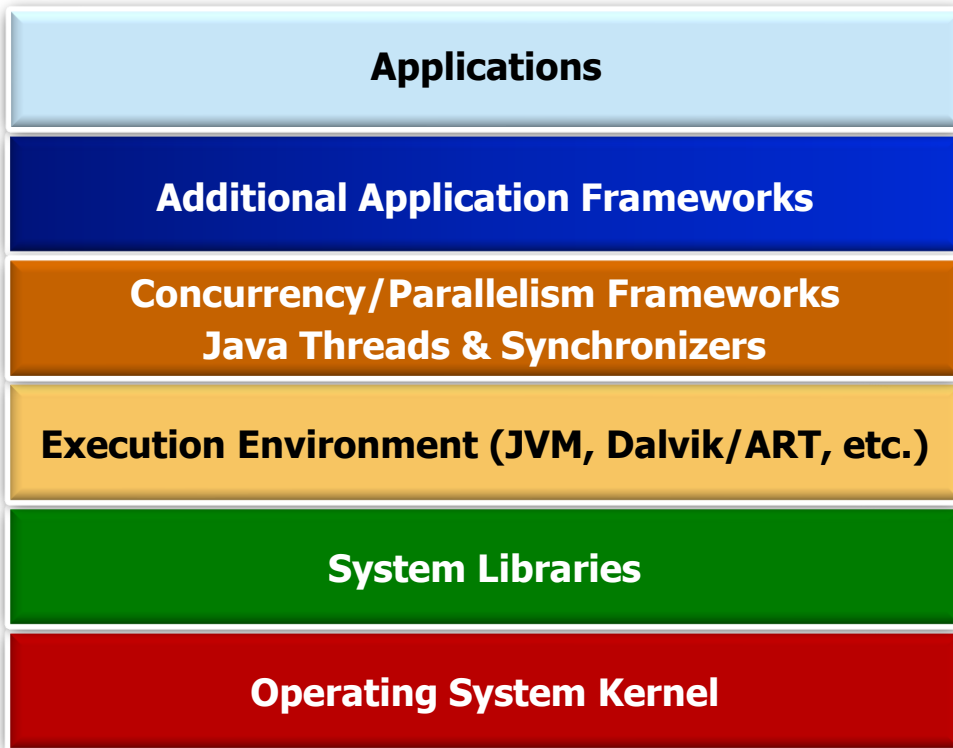


Which Java Mechanism(s) to Understand & Apply

- Java's concurrency & parallelism mechanisms span multiple layers in the software stack
- Choosing best mechanism(s) depend on various factors



Java/JNI
C++/C
C



Which Java Mechanism(s) to Understand & Apply

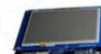
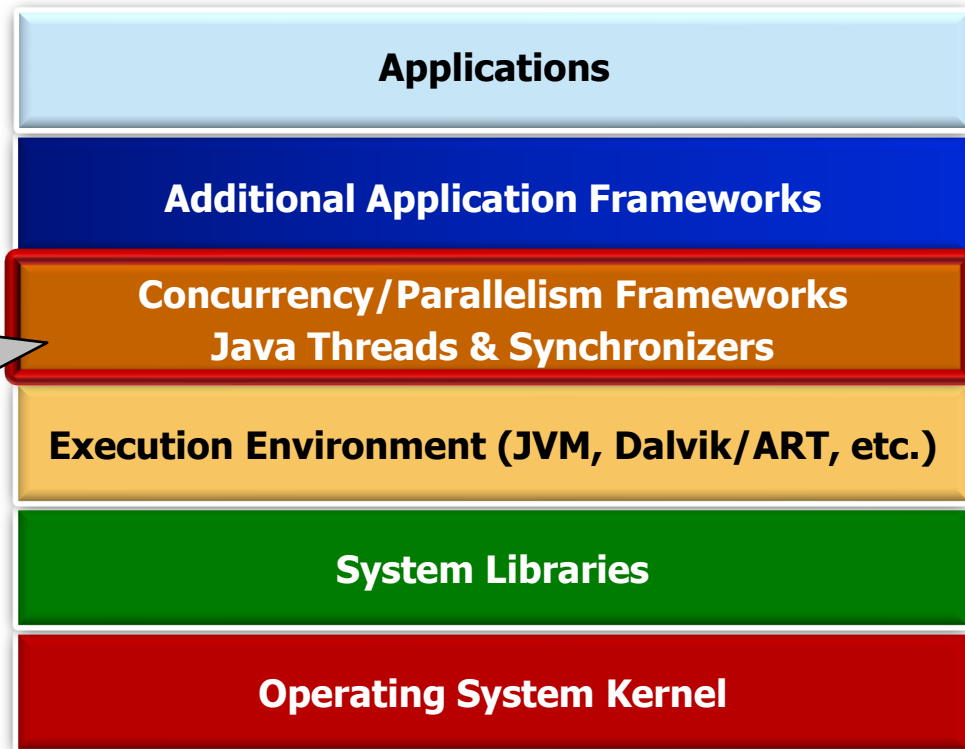
- Developers of low-level classes & performance-sensitive apps may prefer shared object mechanisms

Package `java.util.concurrent` Description

Utility classes commonly useful in concurrent programming. This package includes a few small standardized extensible frameworks, as well as some classes that provide useful functionality and are otherwise tedious or difficult to implement. Here are brief descriptions of the main components. See also the `java.util.concurrent.locks` and `java.util.concurrent.atomic` packages.



va/JNI
C C++/C



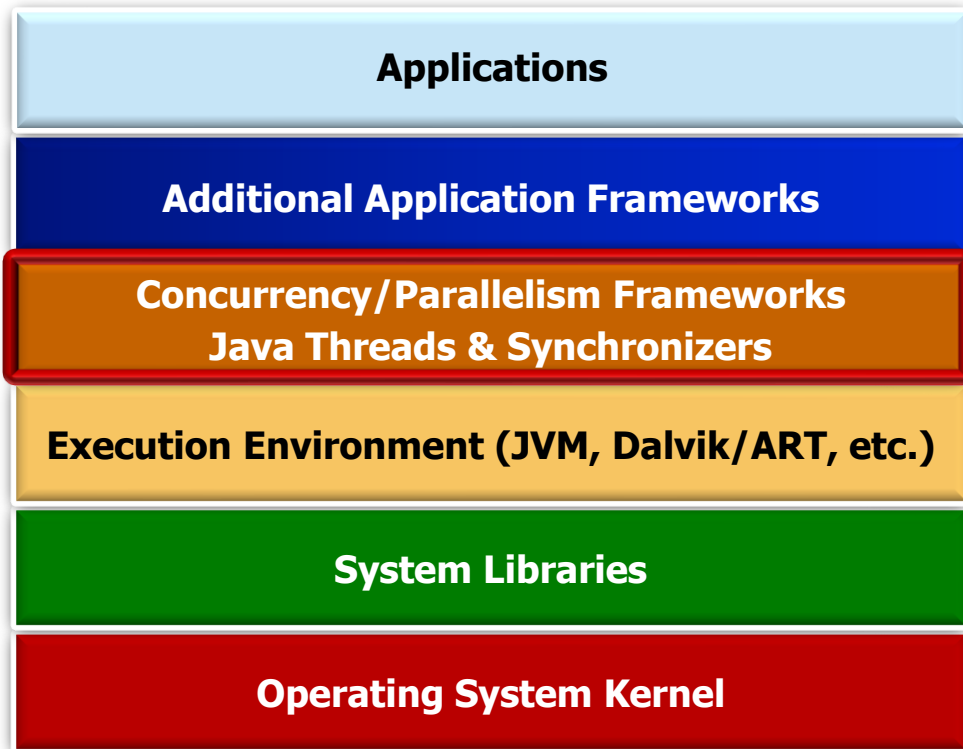
e.g., `java.util.concurrent` as per www.youtube.com/watch?v=sq0MX3fHkro

Which Java Mechanism(s) to Understand & Apply

- Developers of low-level classes & performance-sensitive apps may prefer shared object mechanisms
 - **Pros:** Efficient & lightweight
 - **Cons:** Tedious & error-prone



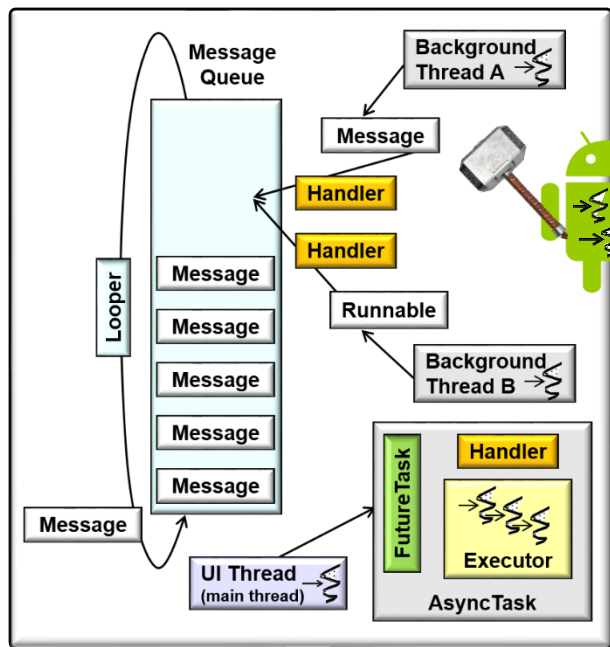
C
C++/C
Java/JNI



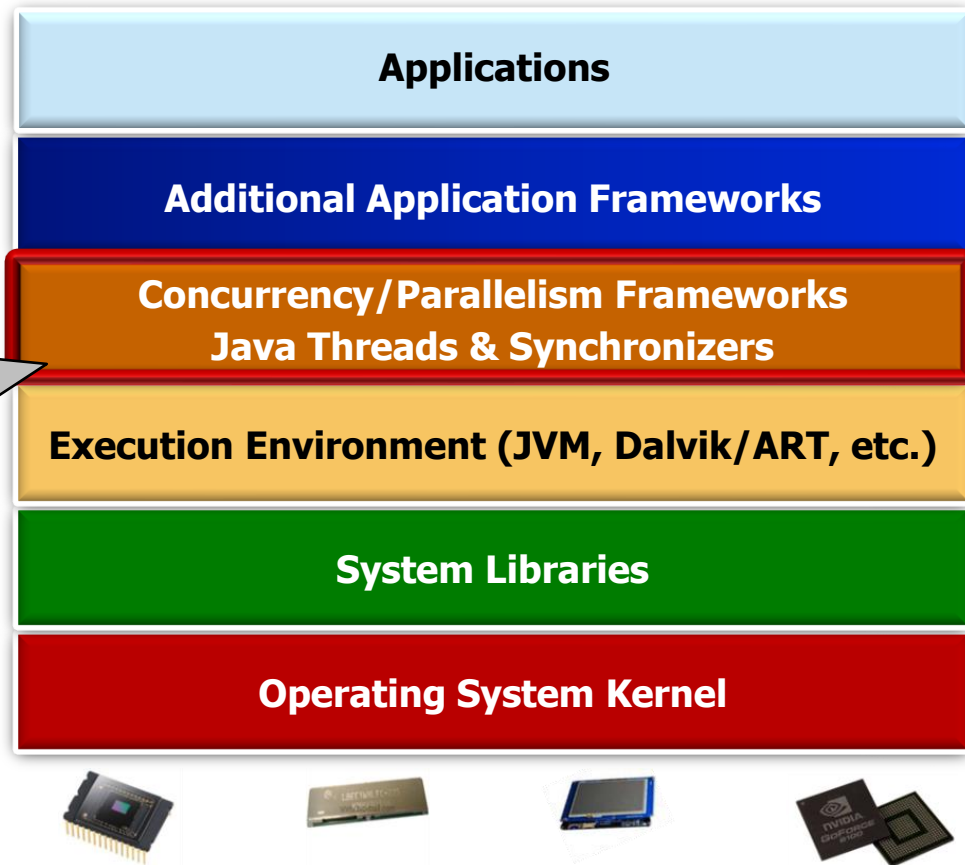
Shared objects are often best used by infrastructure vs. app developers

Which Java Mechanism(s) to Understand & Apply

- Framework developers may want to use the Java message passing mechanisms



ava/JNI
C++/C
C



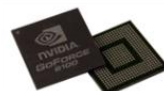
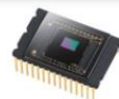
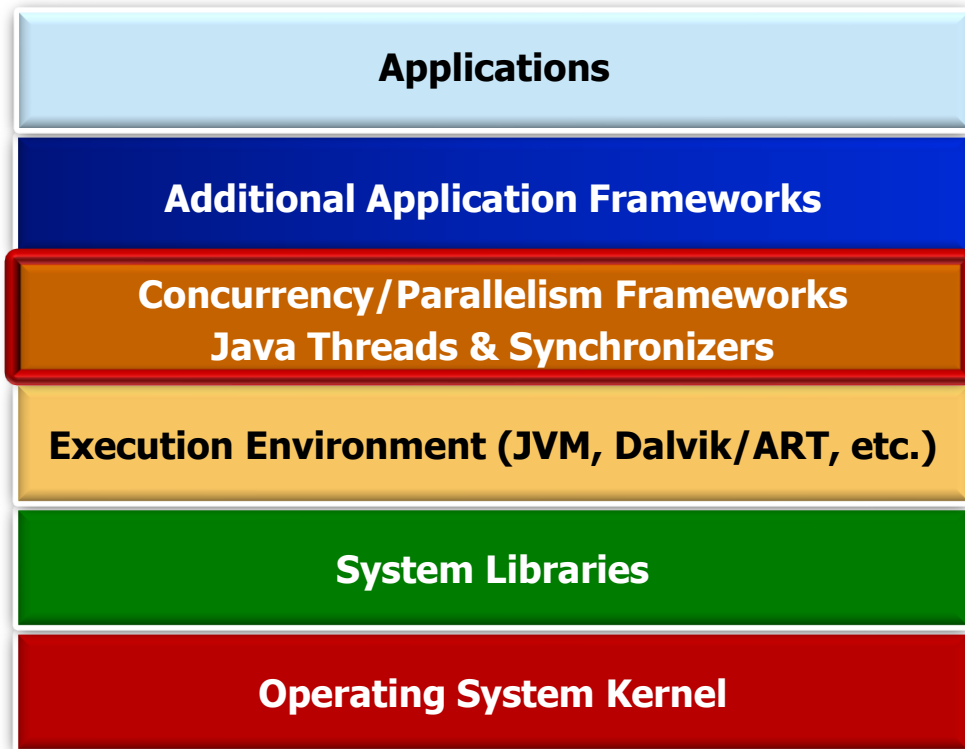
e.g., Android AsyncTask/HaMeR frameworks or Java ExecutorCompetitionService

Which Java Mechanism(s) to Understand & Apply

- Framework developers may want to use the Java message passing mechanisms
 - Pros:** Flexible & decoupled
 - Cons:** Time/space overhead



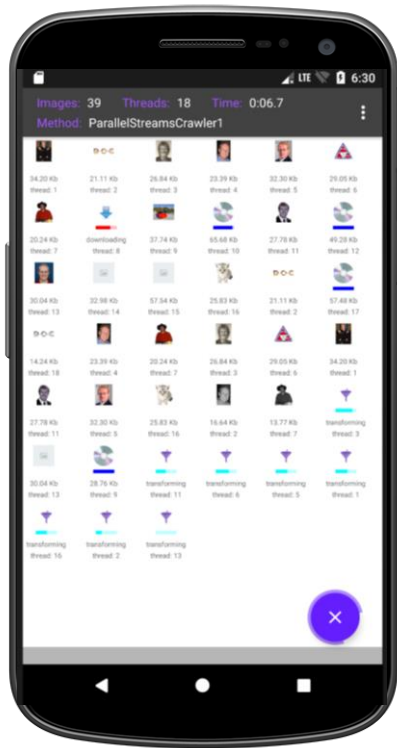
Java/JNI
C++/C
C



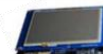
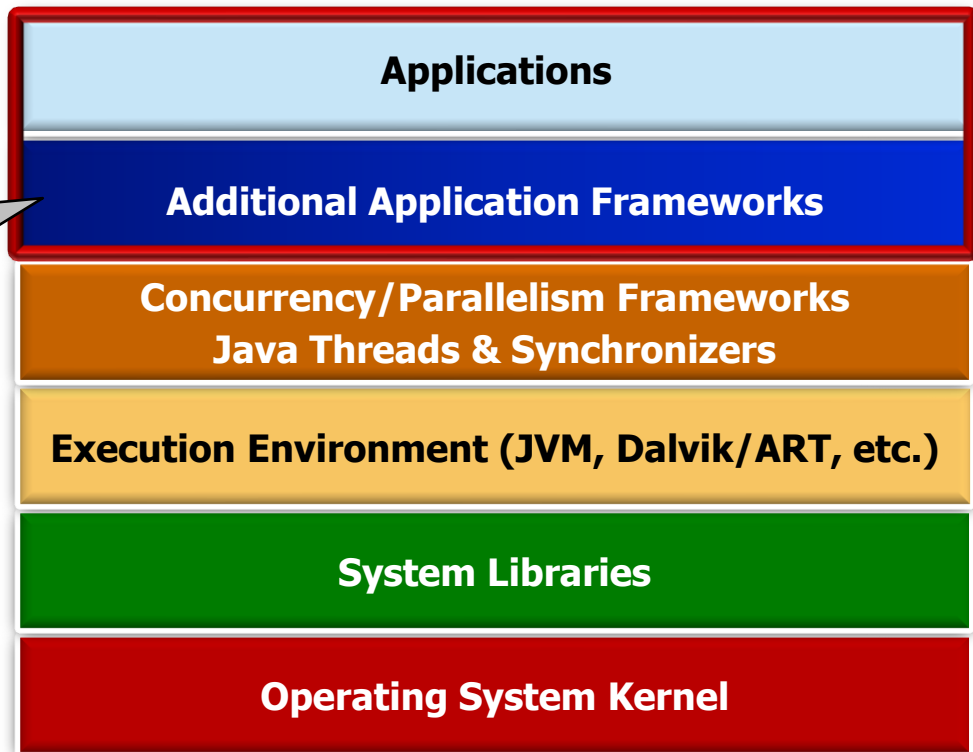
May incur higher context switching, synchronization, & data movement overhead

Which Java Mechanism(s) to Understand & Apply

- Mobile app developers may want to program w/higher-level frameworks



Java/JNI
C++/C
C



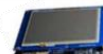
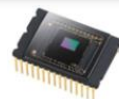
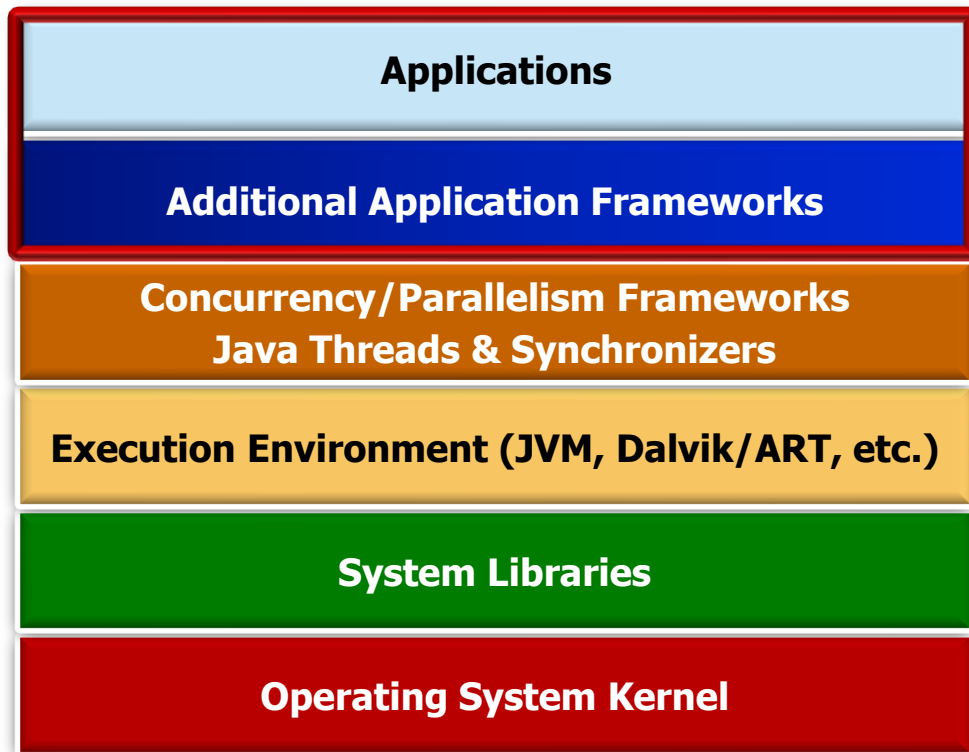
e.g., Java parallel streams & completable futures, RxJava, etc.

Which Java Mechanism(s) to Understand & Apply

- Mobile app developers may want to program w/higher-level frameworks
 - **Pros:** Productivity & robustness
 - **Cons:** Time/space overhead & overly prescriptive



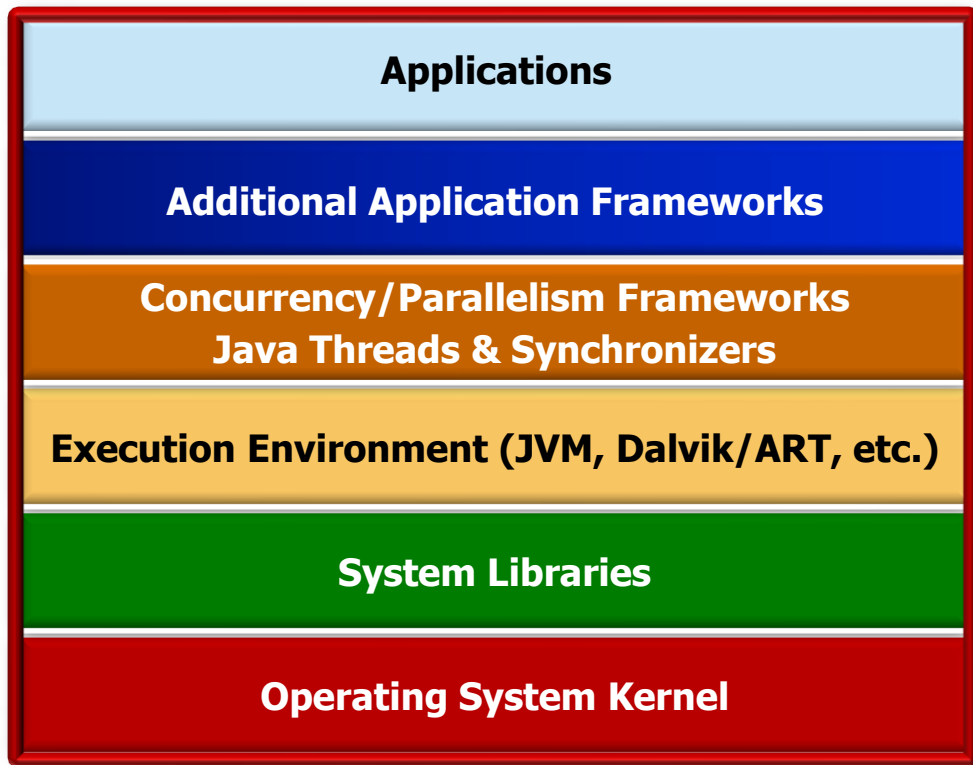
C
C++/C
Java/JNI



Which Java Mechanism(s) to Understand & Apply



C
C++/C
Java/JNI



“Full stack” developers should understand concepts & mechanisms at each layer

End of Evaluation of Concurrency & Parallelism in Java