Concurreny Library

9-14-2011

Opening Discussion

- Minute essay comments
  - Cores and stressing machines.
- IcP solutions

Motivation

- The future is parallel.
- Core counts are growing but clock speed isn't and neither is single thread performance.
- Software developers are behind the curve on this.

wait/notifyAll

- Allows synchronization between threads. A thread can wait and it won't restart until another thread notifies it.
- Put wait in while loop that checks a Boolean flag.
- Always use notifyAll instead of notify. Failure to do so leads to deadlocks.
- These must be called from inside of a synchronized block.

java.util.concurrent

- Java 5 added the java.util.concurrent package and others below it.
- Provides better ways to do common tasks for parallel.

Executors

- Use the proper one of these to start threads instead of making them manually.
- Allows Callable[A] and Future[A] which return a value.

Parallel Data Structures

- BlockingQueue
- ConcurrentMap
- CountDownLatch
- CyclicBarrier
- Exchanger
- PriorityBlockingQueue
- Semaphore
- Scala provides some support for basic collections.

Locks

- More flexible than synchronized.
- Provides extra power when needed. Particularly for locking across method calls.

Atomics

- Data values with atomic access.
- Faster and easier than doing your own synchronization.

Code

- I want to get commands working so that we can play with some of this in the drawing program.

Minute Essay

- How might you break parts of your project code into different threads to take advantage of many cores?