INTRODUCING "JAVA 8"

Concurrency Updates

Concurrency Updates

- scalable Update variables
- fork and join improvement
- Stamped Lock class

Scalable Update variables

- Maintaining a single count, sum, etc. that is updated by possibly many threads is a common scalability problem.
- Following new classes were introduced to support scalable update variables.
  - DoubleAccumulator
  - DoubleAdder
  - LongAccumulator
  - LongAdder

Striped64

- is a new class in JDK 8 that is the base for a whole family of new classes in java.util.concurrency.atomic
- The basic concept of a Striped64 is that it holds a hash table of Cells (think of each Cell as an AtomicLong). When two threads try to add something to a LongAdder – which is a Striped64 – then there is a good chance that the threads will try to add their value to different Cells in that hash table. This reduces the contention to a near minimum.

Fork and Join improvements

- The fork/join framework is an implementation of the ExecutorService interface that helps you take advantage of multiple processors.
- It is designed for work that can be broken into smaller pieces recursively.
- The goal is to use all the available processing power to enhance the performance of your application.
- A static commonPool() method is now available and appropriate for most applications.
- The common pool is used by any ForkJoinTask that is not explicitly submitted to a specified pool.
- The common pool is used by any ForkJoinTask that is not explicitly submitted to a specified pool.
- Using the common pool normally reduces resource usage (its threads are slowly reclaimed during periods of non-use, and reinstated upon subsequent use).

StampedLock

- A new StampedLock class adds a capability-based lock with three modes for controlling read/write access (writing, reading, and optimistic reading).
- They employ a concept of stamps that are *long* values that serve as tickets used by any lock / unlock operation.
- This means that to unlock a R/W operation you need to pass it its correlating lock stamp.

StampedLock example:


```
public class BankAccountWithStampedLock {
   private final StampedLock lock = new StampedLock();
   private double balance;
   public void deposit( double amount) {
   long stamp = lock.writeLock();
   try
   {
   balance = balance + amount;
   }finally{
   lock.unlockWrite(stamp);
   }}
   public double getBalance() {
   long stamp = lock.readLock();
   try{
   return balance;
   } finally{
   lock.unlockRead(stamp);
   }}}
```