#	Source Code		Simplified Code	
	Original	Modified	Original	Modified
| 1 | ```java
LeafQueue leafQueue = ...;
-synchronized (leafQueue) {
   57 LOC
 }
``` | ```java
LeafQueue leafQueue = ...;
+try { leafQueue.getReadLock().lock();
   57 LOC
+} finally {
+  leafQueue.getReadLock().unlock();}
``` | ```java
synchronized (obj)
{
  ...
}
``` | ```java
try {obj.lock();
  ...
} finally {
  obj.unlock();
}
``` |
| 2 | ```java
-Lock readlock =
-  classLoaderContainerMapLock.readLock();
-try { readlock.lock();
-  result = classLoaderContainerMap.get(tccl);
-} finally {readlock.unlock();}
-if (result == null) { Lock writelock =
-  classLoaderContainerMapLock.writeLock();
-  try { writeLock.lock();
    result = classLoaderContainerMap.get(tccl);
    if (result == null) {
      result = new ServerContainerImpl();
      classLoaderContainerMap.put(tccl,result);}
-  } finally {writeLock.unlock();}
``` | ```java
+synchronized (classLoaderContainerMapLock) {



    result = classLoaderContainerMap.get(tccl);
    if (result == null) {
      result = new ServerContainerImpl();
      classLoaderContainerMap.put(tccl,result);}
}
``` | ```java
try {
  readLock.lock();
  read operations
} finally {
  readLock.unlock();
}
try {
  writeLock.lock();
  write operations
} finally {
 writeLock.unlock();
}
``` | ```java
synchronized {
  all operations
}
``` |
| 3 | ```java
 static final Object lock = new Object();
 Map<...> count = new HashMap<>();
-synchronized (count) {
- Pair<Job, String> key =
-   new ImmutablePair<>(jobID, name);

- if (count.containsKey(key)) {
-   count.put(key, count.get(key) + 1);
- } else {count.put(key, 1);}}
``` | ```java
 static final Object lock = new Object();
 Map<...> count = new HashMap<>();
+synchronized(lock)
+  if (!jobCounts.containsKey(jobID)) {
+    jobCounts.put(jobID, new HashMap<>());}
+  Map<...> count = jobCounts.get(jobID);
+  if (count.containsKey(name)) {
+    count.put(name, count.get(name) + 1);
+  } else {count.put(name, 1);}}
``` | ```java
synchronized (obj1)
{
  ...
}
``` | ```java
synchronized
(obj2) {
  ...
}
``` |
| 4 | ```java
-public boolean isAccessed() {
   return this.accessed;}
``` | ```java
+public synchronized boolean isAccessed() {
   return this.accessed;}
``` | ```java
void foo() {...}
``` | ```java
synchronized void
foo() {...}
``` |
| 5 | ```java
-synchronized (this.channelLookup) { try {
   lookupResponse = AkkaUtils.
<...>ask(channelLookup, new
JobManagerMessages.LookupConnectionInformation(c
onnectionInfo, jobID, sourceChannelID),
timeout).response();
-  } catch(IOException ioe) {throw ioe;}}
``` | ```java
   lookupResponse = AkkaUtils.
<...>ask(channelLookup, new
JobManagerMessages.LookupConnectionInformation
(  connectionInfo, jobID, sourceChannelID),
timeout).response();
``` | ```java
synchronized (obj)
{
  ...
}
``` | ```java
...
``` |
| 6 | ```java
 synchronized (buffers) { if (...) {

-    if (spillWriter != null) {
-      spillWriter.close();}
     isFinished = true;}}
``` | ```java
 synchronized (buffers) { if (...) {
     isFinished = true;}}
+if (spillWriter != null) {
+  spillWriter.close();}
``` | ```java
synchronized(obj)
{
  statements1
  statements2 }
``` | ```java
synchronized(obj)
{
  statements2 }
Statements1
``` |
| 7 | ```java
-synchronized void reset() {

   map.clear();
   members = EMPTY_MEMBERS;}
``` | ```java
+final Object membersLock = new Object();
+void reset() { synchronized (membersLock) {
   map.clear();
   members = EMPTY_MEMBERS;}}
``` | ```java
synchronized void
foo() {
  ... }
``` | ```java
void foo() {
  synchronized
(obj) {
    ... }}
``` |
| 8 | ```java
-synchronized void enqueue(final long seqno,
   final boolean lastPacketInBlock,
   final long offsetInBlock) {
- if (running) {
   final Packet p = new Packet(...);
   LOG.debug(...);

   ackQueue.addLast(p); notifyAll();}}
``` | ```java
+void enqueue(final long seqno,
   final boolean lastPacketInBlock,
   final long offsetInBlock) {

   final Packet p = new Packet(...);
   LOG.debug(...);

+  synchronized (this) { if (running) {
     ackQueue.addLast(p); notifyAll();}}}
``` | ```java
synchronized void
foo(...) {
  statements1
  statements2
}
``` | ```java
statements1
synchronized (obj)
{
  statements2
}
``` |
| 9 | ```java
-Membership membership = null;
 public boolean hasMembers() {
   if (membership == null) setupMembership();
   return membership.hasMembers();}
 synchronized void setupMembership() {
   if ( membership == null ) {
     membership = new Membership(...);}}
``` | ```java
+volatile Membership membership = null;
 public boolean hasMembers() {
   if (membership == null) setupMembership();
   return membership.hasMembers();}
 synchronized void setupMembership() {
   if ( membership == null ) {
     membership = new Membership(...);}}
``` | ```java
T foo;
``` | ```java
volatile T foo;
``` |
| 10 | ```java
-volatile int requestCount;

-  requestCount++;
``` | ```java
+final AtomicInteger requestCount =
+  new AtomicInteger(0);
+  requestCount.incrementAndGet();
``` | ```java
volatile T foo;
``` | ```java
TT foo;
``` |