When we use lock api. Thread inter-communication is achieved through Condition.

Synchronized--- lock

Wait --- Get condition from lock, lock.await

Notify --- Get condition from lock, lock.signal.

Below code is an example code of BlockingQueue, which uses Condition to intercommunicate between threads.

**public** **class** BlockingQueueImpl<T> {

Queue<T> queue;

**int** capacity;

Lock lock = **new** ReentrantLock();

Condition full = lock.newCondition();

Condition empty = lock.newCondition();

**public** BlockingQueueImpl(**int** size) {

**this**.capacity = size;

queue = **new** LinkedList<T>();

}

**public** **void** insert(T element) {

**try** {

lock.lock();

**if** (capacity == queue.size()) {

full.await();

}

full.signal();

queue.add(element);

notifyAll();

} **catch** (InterruptedException e) {

e.printStackTrace();

} **finally** {

lock.unlock();

}

}

**public** T fetchAndRemove() {

**try** {

lock.lock();

**if** (0 == capacity) {

empty.await();

}

T element = queue.remove();

empty.signal();

**return** element;

} **catch** (InterruptedException e) {

e.printStackTrace();

} **finally** {

lock.unlock();

}

**return** **null**;

}

**public** **static** **void** main(String[] args) {

BlockingQueueImpl b = **new** BlockingQueueImpl<Integer>(10);

}

}