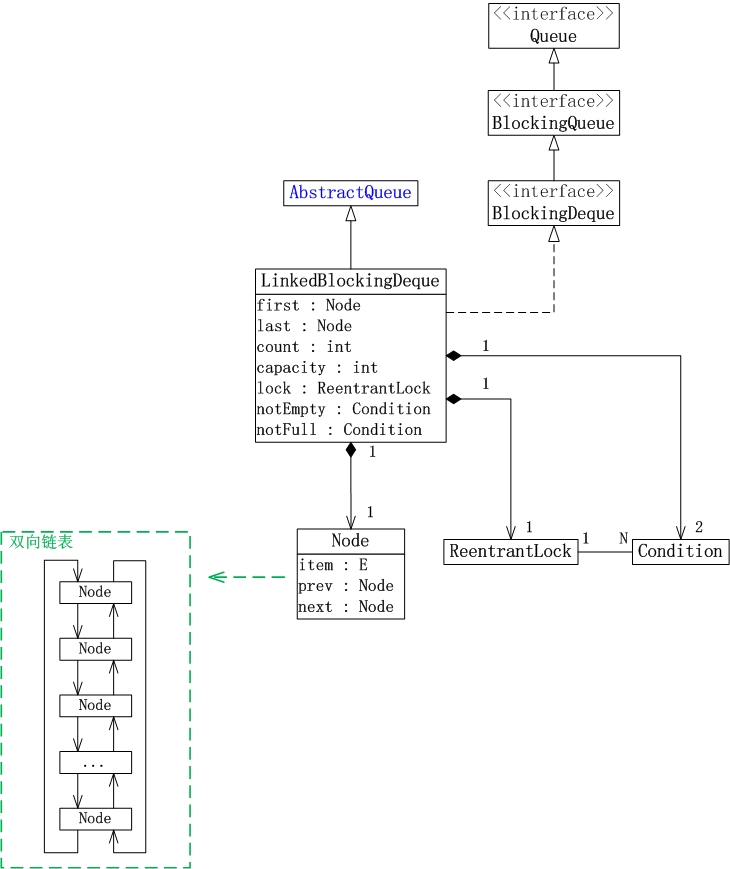
LinkedBlockingDeque是双向链表实现的双向并发阻塞队列。该阻塞队列同时支持FIFO和FILO两种操作方式，即可以从队列的头和尾同时操作(插入/删除)；并且，该阻塞队列是支持线程安全。

此外，LinkedBlockingDeque还是可选容量的(防止过度膨胀)，即可以指定队列的容量。如果不指定，默认容量大小等于Integer.MAX\_VALUE。



**说明**：  
1. LinkedBlockingDeque继承于AbstractQueue，它本质上是一个支持FIFO和FILO的双向的队列。  
2. LinkedBlockingDeque实现了BlockingDeque接口，它支持多线程并发。当多线程竞争同一个资源时，某线程获取到该资源之后，其它线程需要阻塞等待。  
3. LinkedBlockingDeque是通过双向链表实现的。  
3.1 first是双向链表的表头。  
3.2 last是双向链表的表尾。  
3.3 count是LinkedBlockingDeque的实际大小，即双向链表中当前节点个数。  
3.4 capacity是LinkedBlockingDeque的容量，它是在创建LinkedBlockingDeque时指定的。  
3.5 lock是控制对LinkedBlockingDeque的互斥锁，当多个线程竞争同时访问LinkedBlockingDeque时，某线程获取到了互斥锁lock，其它线程则需要阻塞等待，直到该线程释放lock，其它线程才有机会获取lock从而获取cpu执行权。  
3.6 notEmpty和notFull分别是“非空条件”和“未满条件”。通过它们能够更加细腻进行并发控制。

-- 若某线程(线程A)要取出数据时，队列正好为空，则该线程会执行notEmpty.await()进行等待；当其它某个线程(线程B)向队列中插入了数据之后，会调用notEmpty.signal()唤醒“notEmpty上的等待线程”。此时，线程A会被唤醒从而得以继续运行。 此外，线程A在执行取操作前，会获取takeLock，在取操作执行完毕再释放takeLock。

-- 若某线程(线程H)要插入数据时，队列已满，则该线程会它执行notFull.await()进行等待；当其它某个线程(线程I)取出数据之后，会调用notFull.signal()唤醒“notFull上的等待线程”。此时，线程H就会被唤醒从而得以继续运行。 此外，线程H在执行插入操作前，会获取putLock，在插入操作执行完毕才释放putLock。