NSRecursiveLock

Inherits From:	NSObject
Conforms To:	NSLocking NSObject (NSObject)
Declared In:	Foundation/NSLock.h

Class Description

NSRecursiveLock defines a lock that may be acquired multiple times by the same thread without causing a deadlock, a situation where a thread is permanently blocked waiting for itself to relinquish a lock. While the locking thread has one or more locks, all other threads are prevented from accessing the code protected by the lock. Here's an example where a recursive lock functions properly but other lock types would deadlock:

```
NSRecursiveLock  *theLock = [[NSRecursiveLock alloc] init];
...
[theLock lock];
/* lengthy operations involving global data */
[theLock lock];   /* possibly invoked in a subroutine */
...
[theLock unlock]; /* relinquishes most recent lock */
...
[theLock unlock]; /* relinquishes the first lock */
```

Unless **theLock** was an NSRecursiveLock, a deadlock condition would occur at the second **lock** message in the example above.

The NSConditionLock, NSLock, and NSRecursiveLock classes all implement the NSLocking protocol with various features and performance characteristics; see the other class descriptions for more information.

Adopted Protocols

NSLocking	– lock – unlock

Method Types

Acquiring a lock	– lockBeforeDate: – tryLock

Instance Methods

lockBeforeDate

– (BOOL)**lockBeforeDate:**(NSDate *)*limit*

Attempts to acquire a lock before the date represented by *limit*. Returns YES if the lock is acquired within this time limit. Returns NO if the time limit expires before a lock can be acquired.

tryLock

– (BOOL)**tryLock**

Attempts to acquire a lock. Returns immediately with a value of YES if successful and NO otherwise.