NXLock

Adopted By:	NXConditionLock NXLock NXSpinLock NXRecursiveLock
Declared In:	machkit/NXLock.h

Protocol Description

This protocol is used by classes that provide lock objects. The lock objects provided by NeXTSTEP are used only for protecting critical sections—they contain no useful data.

Although an object that isn't a lock could adopt the NXLock protocol, it may be more desirable to design the object so that all locking is handled internally, through normal use rather than requiring that the object be explicitly locked and unlocked.

Four classes conform to the NXLock protocol:

Class	**Usage**
NXLock	Use NXLock objects to protect regions of code that can consume long periods of time, such as disk I/O or heavy computation.
NXConditionLock	Protects critical sections of code, but can also be used to postpone entry to a critical section until a condition is met. This class is functionally a superset of the NXLock class, though unlocking is slightly more expensive.
NXSpinLock	Use NXSpinLock objects to protect short regions of critical code. Useful in the implementation of drivers or more complex locks. A spin lock may be acquired more quickly than the other locks, but isn't appropriate for long sections of code since blocked spin locks busy-wait.
NXRecursiveLock	Protects critical sections from access by multiple threads, but allows a single thread to acquire a lock several times without deadlocking.

Of these classes, only NXSpinLock busy-waits while the lock is unavailable. The other classes may all be efficiently used for long sections of atomic code. See the class specifications for these classes for further information on their behavior and usage.

Instance Methods

lock

– lock

Acquires a lock. Applications generally do this when entering a critical section of their code.

unlock

– unlock

Releases a lock. Applications generally do this when exiting a critical section of their code.