

CJRConditionLock

Inherits From:	Object
Conforms To:	CJRLock
Declared In:	CJRLock.h
Requires:	ThreadedApp

Class Description

CJRConditionLock is similar to NeXT's NXConditionLock (NS3.3) and NSConditionLock (OpenStep), except that it detects if it being called from the main AppKit thread. If it is, it does not block waiting for a lock, but rather starts a modal event loop looking for EV_UNLOCK events. This keeps the AppKit thread alive and responsive to events.

CJRConditionLocks are used in the same way as NXConditionLocks and NSConditionLocks. However, they cannot be used from fast callback methods (see ThreadedApp).

If they are called from the main thread, the lock methods use the same event-loop strategy as described in CJRLock, except they also check if the lock's condition matches the desired condition. The unlock methods send the required EV_UNLOCK events.

Instance Variables

None declared in this class.

Method Types

Initializing an instance	- <code>init</code> - <code>initWithCondition:</code>
Get the condition of the lock	- <code>condition</code>
Acquire or release the lock	- <code>lock</code> - <code>lockWhenCondition:</code> - <code>unlock</code> - <code>unlockWithCondition:</code>

Instance Methods

condition

- **(int)condition**

Returns the lock's current condition. This condition can be set with the **initWithCondition:** or **unlockWithCondition:** methods.

init

- **init**

Initializes a newly allocated `CJRConditionLock` instance and sets its condition to 0.

initWithCondition:

- **initWithCondition:(int)condition**

Initializes a newly allocated `CJRConditionLock` instance and sets its condition to *condition*. This message should not be sent to an instance that has already been initialized.

lock

- lock

Waits until the lock isn't in use, then grabs the lock. The lock can subsequently be released with either **unlock** or **unlockWithCondition:**.

lockWhenCondition:

- lockWhenCondition:(int)*condition*

Waits until the lock isn't in use and the lock's condition matches *condition*, then grabs the lock. The lock's condition can be set by **initWithCondition:** or **unlockWithCondition:**. The lock can subsequently be released with either **unlock** or **unlockWithCondition:**.

unlock

- unlock

Releases the lock but doesn't change its condition.

See also: - **unlockWithCondition:**

unlockWithCondition:

- unlockWithCondition:(int)*condition*

Sets the lock's condition to *condition* and releases the lock.

See also: - **unlock**