NXInvalidationNotifier

Inherits From: Object

Conforms To: NXReference

Declared In: machkit/NXInvalidationNotifier.h

Class Description

The NXInvalidationNotifier class is an abstract class that defines reference-counted objects that notify other objects when they become invalid. An NXInvalidationNotifier becomes invalid when no more references to it are held (all references have been given up by sending the object a **free** message). An NXInvalidationNotifier could also become invalid for other reasons; for example, an NXConnection object (which is a subclass of NXInvalidationNotifier) becomes invalid when its connection is broken. An invalid object usually exists for a short time after becoming invalid so it can clean up, but it shouldn't be treated as though it were fully usable.

Examples of NXInvalidationNotifier subclasses include NXConnection and NXPort.

Instance Variables

unsigned int **refcount**;
BOOL **isValid**;
NXLock ***listGate**;
List ***funeralList**;

refcount	The object's reference count
isValid	YES if the object is valid
listGate	A lock to protect data structures
funeralList	A list of objects to be notified upon invalidation

Adopted Protocols

NXReference	– addReference – free – references

Method Types

Initializing a new object	– init
Really freeing an object	– deallocate
Getting and setting validity	– invalidate – isValid
Registering for notification	– registerForInvalidationNotification: – unregisterForInvalidationNotification:

Instance Methods

deallocate

– **deallocate**

Deallocates the object's storage, freeing the object regardless of its reference count. A subclass of NXInvalidationNotifier should generally invoke this method from within its implementation of **free** when no more references are held to ensure normal freeing behavior.

See also: – **free**

free

– **free**

Decrements the reference count of the object, marking the object invalid, sending invalidation notifications, and returning **nil** if no more references to the object are held. If references are still held, this method returns **self**. Unlike the **free** method for most classes, this method *never* deallocates the object's storage. (In other words, it never actually frees the object.) This means that the object still exists to receive messages after it becomes invalid due to freeing, which can be useful for objects that need to do some final housekeeping when

no more references are held. Generally a subclass of NXInvalidationNotifier should implement a version of **free** that deallocates itself when no more references are held, with the result that **free** will properly deallocate the object as expected at the appropriate time. For example:

```
- free
{
    id ret = [super free];
    if (ret) return self;
    // No more references held, do the required cleanup
    return [super deallocate];
}
```

See also: – **deallocate**, – **invalidate**

init

– **init**

Initializes the receiver, a newly allocated NXInvalidationNotifier instance. Returns **self**.

invalidate

– **invalidate**

Marks the object as invalid, which means that though the object exists, it's not completely functional and might not exist for long. Once an object becomes invalid, there is no way provided to make the object valid again, and it would be difficult to implement in a subclass in a thread-safe manner. This method sends a **senderIsInvalid:** message to every object that registered for invalidation notification, frees the **funeralList** (but not the objects in it), and returns **self**.

See also: – **registerForInvalidationNotification:**

isValid

– (BOOL)**isValid**

Returns YES if the object is valid. Generally, invalid objects should be sent only messages that allow other objects to clean up and eliminate their use of the object.

registerForInvalidationNotification:

– **registerForInvalidationNotification:**(id <NXSenderIsInvalid>)*anObject*

Registers *anObject* so that it will receive a **senderIsInvalid:** message when the receiver becomes invalid. An object might become invalid because it is about to be freed, because a Distributed Objects connection is broken, or for some other application-specific reason. Returns **self**.

See also: – **invalidate**, – **unregisterForInvalidationNotification:**

unregisterForInvalidationNotification:

– **unregisterForInvalidationNotification:**(id <NXSenderIsInvalid>)*anObject*

Removes *anObject* from the list of objects that are notified when the receiver becomes invalid; thus *anObject* won't be notified. Returns **self**.

See also: – **registerForInvalidationNotification:**