MiscSparseArray

Inherits From:	NSObject
Conforms To:	NSCoding, NSCopying NSObject (NSObject)
Declared In:	MiscSparseArray.h

Class Description

MiscSparseArray is an abstract class which provides the interface for storing objects in a sparse array data structure.   A sparse array is an array, indexed by unsigned integers, which is used most efficiently when the array contains few elements, or are heavily grouped together in clusters, with large barren sections of the index space.

Each index in the index space of unsigned integers (0 .. 2^{32}-1) logically exists; the indices of elements in the array do not need to be contiguous (unlike NSArray). If no object is stored at an index, it contains the value **nil**. The value **nil** cannot be stored in a sparse array (**nil** is not an object).

A sparse array is useful when there is a unique index that can be associated with (or computed from) objects that are to be stored in it. Both of the operations of accessing the value at an index and putting a value at an index occur in constant time. Iterating through all objects in the array also occurs in constant time (but, the constant is extremely large). (Ironically, this constant time is the worst case. In practice, iteration will require time on the order of the number of elements stored in the array, which will generally be better, but is "theoretically" worse). The space usage of a sparse array, however, can be rather high, particularly if the index space is filled rather uniformly (for instance, if object are stored at indices 0..10,000 and 55,122,540..55,143,102 rather than at all indices divisible by 14, storage requirements will be much less). The more uniform the distribution over the index space, the more storage is required, until very quickly more memory is required than is available (physical or virtual).

The default concrete implementation of MiscSparseArray has features reminiscent of a btree, but unlike a tree, the value of each index is stored at a fixed location computed from the index. Other concrete subclasses of MiscSparseArray may have different operating statistics than those quoted above (but may be more space efficient). The Objective C runtime in the GNU C compiler uses a sparse array to store information associated with method selectors. A hash table is another type of sparse array.

Subclassing MiscSparseArray

Subclasses of MiscSparseArray must implement the primitive methods **-count**, **-objectAtIndex:**, and **-putObject:atIndex:** to have the semantics described below.

Creating a MiscSparseArray Object

+ (MiscSparseArray *)**sparseArray**	Creates and returns an empty (and autoreleased) MiscSparseArray.
- (id)**init**	Initializes a newly allocated MiscSparseArray to be empty. This method is the designated initializer of MiscSparseArray instances.

Getting Information About a MiscSparseArray

- (BOOL)**containsObject:**(id)*anObject*	Returns YES if the sparse array contains *anObject*. **-isEqual:** is used

in the comparison.

- (unsigned int)**count** Returns the number of objects stored in the sparse array. This is a primitive method that all subclasses must implement.

- (BOOL)**isEqualToSparseArray:**(MiscSparseArray *)*otherArray*
Returns YES if *otherArray* has the same objects stored in it, and at the same locations, as the receiver.

Accessing Objects and Indices

- (unsigned int)**indexOfObject:**(id)*anObject* Returns the lowest index of an object in the array which is **-isEqual:** to *anObject*. Returns NSNotFound if the object isn't in the array.

- (unsigned int)**indexOfObjectIdenticalTo:**(id)*anObject*
Returns the lowest index of an object in the array which is pointer-equal to *anObject*. Returns NSNotFound if the object isn't in the array.

- (void)**makeObjectsPerform:**(SEL)*aSelector* Sends a **-perform:** message to each object in the array.

- (void)**makeObjectsPerform:**(SEL)*aSelector* **withObject:**(id)*anObject*
Sends a **-perform:withObject:** message to each object in the array.

- (id)**objectAtIndex:**(unsigned int)*anIndex*
Returns the object previously stored at *anIndex* or **nil** if there is no object stored there. This is a primitive method that all subclasses must implement.

- (void)**putObject:**(id)*anObjectOrNil* **atIndex:**(unsigned int)*anIndex*
Puts *anObjectOrNil* at *anIndex* in the array. The first parameter may be **nil**, which has the effect of removing the object (if any) at *anIndex* from the array. This is a primitive method that all subclasses must implement.

- (void)**removeAllObjects**
Removes all objects from the array.

- (void)**removeObject:**(id)*anObject*
Removes all objects which are **-isEqual:** to *anObject* from the array.

- (void)**removeObjectAtIndex:**(unsigned int)*anIndex*
Removes the object (if any) stored at the index *anIndex*.

- (void)**removeObjectIdenticalTo:**(id)*anObject* Removes all objects which are pointer-equal to *anObject* from the array.

Enumerating MiscSparseArrays

- (NSEnumerator *)**objectEnumerator** Returns an enumerator which enumerates the objects in the array from lowest index to highest index.

- (NSEnumerator *)**reverseObjectEnumerator** Returns an enumerator which enumerates the objects in the array from highest index to lowest index.

Describing MiscSparseArrays

- (NSString *)**description** Returns a string descibing the contents of the receiving sparse array.

- (NSString *)**descriptionWithLocale:**(NSDictionary *)*localeDictionary*
Returns a string representation of receiver (see **description**). Included are the key and values that represent the locale data from

localeDictionary.