init
initCount:elementSize:description:

Copying and freeing Storage objects

copyFromZone:
free

Getting, adding, and removing elements

addElement:
insertElement:at:
removeElementAt:

setNumSlots:

Archiving read:

write:

addElement:(void *)anElement

Adds anElement at the end of the Storage array and returns self. The size of the array is increased

insertElement:at:

copyFromZone:(NXZone *)zone

Returns a new Storage object containing the same data as the receiver. The data and the object are
memory for both is taken from zone. However, the description string is not copied the two objects

copy (Object)

(unsigned int)count

Returns the number of elements currently in the Storage array.

setNumSlots:

(const char *)description

Returns the string encoding the data type of elements in the Storage array.

initCount:elementSize:description:

(void *)elementAt:(unsigned int)index

Returns a pointer to the element at index in the Storage array. If no element is stored at index (ind
the array), a NULL pointer is returned.

Before using the pointer that's returned, you must convert it into the appropriate type by a cast. Th
either to read the element at index or to alter it.

replaceElementAt:with:, insertElement:at:

empty

Empties the Storage array of all its elements and returns self. The current capacity of the array ren
nothing is deallocated.

empty

init

Initializes the Storage object so that it's ready to store object ids. The initial capacity of the array i
better to store object ids in a List object. Returns self.

initCount:elementSize:description:, initCount: (List)

initCount:(unsigned int)count
elementSize:(unsigned int)sizeInBytes
description:(const char *)string

Initializes the Storage object so that it has count elements. Each element is of size sizeInBytes and
by string. Memory for all the elements is set to 0. Returns self.

If string is NULL, the object won't be archivable. Once set, the description string should never be

This method is the designated initializer for the class. It's used to initialize Storage objects immed
been allocated it should never be used to reinitialize a Storage object that's already been placed in

insertElement:(void *)anElement at:(unsigned int)index

Puts anElement in the Storage array at index. All elements between index and the last element are
The size of the array is increased if necessary. Returns self.

addElement:, setNumSlots:

(BOOL)isEqual:anObject

Compares the receiver with anObject, and returns YES if they're the same and NO if they're not.
are considered to be the same if they have the same number of elements and the elements at each p
match.

read:(NXTypedStream *)stream

Reads the Storage object and the data it stores from the typed stream stream. Where an archived st
`%' descriptor, the NXUniqueString() function is called to make sure that the string is unique with

write:

removeElementAt:(unsigned int)index

Removes the element located at index from the Storage array and returns self. All elements betwe
element are shifted to close the gap.

replaceElementAt:(unsigned int)index with:(void *)anElement

Replaces the data at index with the data pointed to by anElement. However, if no element is stored beyond the end of the array), nothing is replaced. Returns self.

elementAt:, insertElement:at:

setAvailableCapacity:(unsigned int)numSlots

Sets the storage capacity of the array to at least numSlots elements and returns self. If the array al than numSlots elements, its capacity is left unchanged and nil is returned.

setNumSlots:, count

setNumSlots:(unsigned int)numSlots

Sets the number of elements in the Storage array to numSlots and returns self. If numSlots is great number of elements in the array (the value returned by count), the new slots will be filled with zero than the current number of elements in the array, access to all elements with indices equal to or gre be lost.

If necessary, this method increases the capacity of the storage array so there's room for at least nu

setAvailableCapacity:, count

write:(NXTypedStream *)stream

Writes the Storage object and its data to the typed stream stream.

read: