Paul S. McCarthy and Eric Sunshine -- October 1, 1995

MiscIntList

Inherits From: Object

Declared In: MiscIntList.h

Class Description

An MiscIntList provides a simple interface to an array of integers. The class provides methods sorting the array as well as general list maintenance. Methods are provided for conversion to and from a string representation of the numbers contained in the list.

This class is used by ;MiscTableScroll.rtf;;¬MiscTableScroll as a means to communicate lists of integers with the client. The string conversion methods provide a simple interface for generating data suitable for storage by the NeXT defaults system. For instance, to store the user's column ordering preference in NeXT defaults, the client could ask MiscTableScroll to fill an MiscIntList with the current ordering, convert that to a string representation, and then save it to NeXT defaults.

Instance Variables

Storage* **array**;

array	The storage for the list of integers.

Method Types

Initializing and freeing	;MisχIntList.rtf;init:;← - init ;MiscIntList.rtf;initFromString:;¬ ± initFromString: ;MisχIntList.rtf;free;← - free ;MiscIntList.rtf;copyFromZone:;¬ ± copyFromZone:
Querying and emptying	;MisχIntList.rtf;count;← - count ;MisχIntList.rtf;empty;← - empty
Manipulating by index	;MisχIntList.rtf;intAt:;← - intAt: ;MisχIntList.rtf;addInt:;← - addInt:

	;MiscIntList.rtf;addIntList:;¬ ± addIntList: ;ΜισχΙντΛιστ.ρτφ;ινσερτΙντ:ατ:;← - insertInt:at: ;ΜισχΙντΛιστ.ρτφ;ρεμοωεΙντΑτ:;← - removeIntAt: ;ΜισχΙντΛιστ.ρτφ;ρεπλαχεΙντΑτ:ωιτη:;← - replaceIntAt:with:
Sorting	;ΜισχΙντΛιστ.ρτφ;σορτ;← - sort ;ΜισχΙντΛιστ.ρτφ;σορτΥσινγ:δατα:;← - sortUsing:data:
String conversions	;ΜισχΙντΛιστ.ρτφ;ρεαδΦρομΣτρινγ:;← - readFromString: ;ΜισχΙντΛιστ.ρτφ;ωριτεΤοΣτρινγ;← - writeToString ;ΜισχΙντΛιστ.ρτφ;ωριτεΤοΣτρινγ:σιζε:;← - writeToString:size: ;ΜισχΙντΛιστ.ρτφ;ωριτεΤοΣτρινγ:σιζε:χανΕξπανδ:;← - writeToString:size:canExpand:
Low-level access	;ΜισχΙντΛιστ.ρτφ;ραωΔατα;← - rawData

Instance Methods

addInt:;¬addInt:

- (void)**addInt:**(int)*value*

Appends *value* to the list. Equivalent to: `-insertInt:`*`value`*` at:[self count].`

See also: ;MiscIntList.rtf;addIntList:;¬ ± addIntList:, **;MiscIntList.rtf;insertInt:at:;¬ ±**

insertIntAt:, **;MiscIntList.rtf;removeIntAt:;¬ ±removeIntAt:**

addIntList:;¬addIntList:

- (void)**addIntList:**(MiscIntList*)*list*

Appends the contents of *list* to the end of the receiver.

See also: **;MiscIntList.rtf;addInt:;¬ ± addInt:**

copyFromZone:;¬copyFromZone:

- **copyFromZone:**(NXZone*)*zone*

Allocates, initializes, and returns a copy of the receiver.

See also: - **copy** (Object)

count;¬count

- (unsigned int)**count**

Returns the number of integers in the list.

empty;¬empty

- **empty**

Empties the list. Returns **self**.

See also: ;MiscIntList.rtf;free;¬ ± free

free;¬free

- **free**

Frees the storage used by the list.

See also: ;MiscIntList.rtf;init;¬ ± init, **;MiscIntList.rtf;empty;¬ ± empty**

init;¬init

- **init**

Initializes the storage used by the list. This method is the designated initializer. Returns **self**.

See also: ;MiscIntList.rtf;initFromString:;¬ ± initFromString:, **;MiscIntList.rtf;free;¬ ± free**

initFromString:;¬initFromString:

- **initFromString:**(char const*)*s*

Initializes the list by calling **;MiscIntList.rtf;init;¬-init** and then fills it from the string *s* using **;MiscIntList.rtf;readFromString:;¬-readFromString:**. Returns **self**.

See also: ;MiscIntList.rtf;init;¬ ± init, **;MiscIntList.rtf;free;¬ ± free**, **;MiscIntList.rtf;readFromString:;¬ ± readFromString:**

insertInt:at:;¬insertInt:at:

- (void)**insertInt:**(int)*value*
  at:(unsigned int)*pos*

Inserts *value* at position *pos* in the list. The list is expanded if needed. Intervening slots are initialized to zero.

See also: ;MiscIntList.rtf;addInt:;¬ ± addInt:, **;MiscIntList.rtf;removeIntAt:;¬ ± removeIntAt:**, **;MiscIntList.rtf;replaceIntAt:with:;¬ ± replaceIntAt:with:**

intAt:;¬intAt:

- (int)**intAt:**(unsigned int)*pos*

Returns the integer at position *pos* or 0 if *pos* is out-of-range..

rawData;¬rawData

- (int const*)**rawData**

Returns a pointer to the actual array of integers. This allows low-level, fast access to the data when one needs to avoid the overhead of Objective-C messages. The array should not be modified.

readFromString:;¬readFromString:

- (int)**readFromString:**(char const*)*s*

Fills the list with numbers from the string *s*. This method first empties the list with **;MiscIntList.rtf;empty;¬-empty**, then parses numbers from the string *s* and adds them to the list with **;MiscIntList.rtf;addInt:;¬-addInt:**. The string should be a list of white-space delimited numbers. Returns the number of integers read from the string.

See also: ;MiscIntList.rtf;initFromString:;¬ ± initFromString:, **;MiscIntList.rtf;writeToString;¬ ± writeToString**, **;MiscIntList.rtf;writeToString:size:;¬ ± writeToString:size:**, **;MiscIntList.rtf;writeToString:size:canExpand:;¬ ± writeToString:size:canExpand:**

removeIntAt:;¬removeIntAt:

- (void)**removeIntAt:**(unsigned int)*pos*

Removes the integer at position *pos* from the list. Nothing happens if *pos* is beyond the end of the list.

See also: ;MiscIntList.rtf;addInt:;¬ ± addInt:, **;MiscIntList.rtf;insertInt:at:;¬ ± insertInt:at:**, **;MiscIntList.rtf;replaceIntAt:with:;¬ ± replaceIntAt:with:**

replaceIntAt:with:;¬replaceIntAt:with:

- (void)**replaceIntAt:**(unsigned int)*pos*
  with:(int)*value*

Replaces the integer at position *pos* with *value*. If *pos* is out-of-range then nothing is replaced.

See also: ;MiscIntList.rtf;addInt:;¬ ± addInt:, **;MiscIntList.rtf;insertInt:at:;¬ ± insertInt:at:**, **;MiscIntList.rtf;removeIntAt:;¬ ± removeIntAt:**

sort;¬sort

- (void)**sort**

Sorts the list in ascending numeric order.

See also: ;MiscIntList.rtf;sortUsing:data:;¬ ± sortUsing:data:

sortUsing:data:;¬sortUsing:data:

- (void)**sortUsing:**(MiscIntListCompareFunc)*cmp_func*
  data:(void*)*data*

Sort the list of integers using the comparison function *cmp_func*. If *cmp_func* is NULL then the list is sorted in standard ascending numeric order. The meaning of the *data* argument is unspecified and is determined by the comparision function *cmp_func* to which it is passed unmolested.

`;MiscIntList.rtf;MiscIntListCompareFunc;¬MiscIntListCompareFunc` is a typedef which takes as arguments the *data* variable plus two integers for comparision. It should return a negative value if the first integer is less than the second, a positive value if the first integer is greater than the second, or zero if the integers are equal.

```
typedef int (*MiscIntListCompareFunc)( void* userData, int x, int y );
```

See also: ;MiscIntList.rtf;sort;¬ ± sort

writeToString;¬writeToString

- (char*)**writeToString**

Allocates a string using malloc() and writes a string representation of the contents of the list to it. It is the client's responsibility to free() the returned string. Equivalent to: `-writeToString:0 size:0 canExpand:YES`. Returns the newly allocated buffer.

See also: ;MiscIntList.rtf;readFromString:;¬ ± readFromString:, **;MiscIntList.rtf;writeToString:size:;¬ ± writeToString:size:**, **;MiscIntList.rtf;writeToString:size:canExpand:;¬ ± writeToString:size:canExpand:**

writeToString:size:;¬writeToString:size:

- (char*)**writeToString:**(char*)*buff*
  size:(int)*buff_size*

Writes a string representation of the contents of the list to the buffer *buff* of size *buff_size*. Equivalent to: `-writeToString:`*buff* `size:`*buff_size* `canExpand:NO`. Returns *buff* or 0 if *buff_size* wasn't large enough to hold the string.

See also: ;MiscIntList.rtf;readFromString:;¬ ± readFromString:, **;MiscIntList.rtf;writeToString;¬ ± writeToString**, **;MiscIntList.rtf;writeToString:size:canExpand:;¬ ± writeToString:size:canExpand:**

writeToString:size:canExpand:;¬writeToString:size:canExpand:

- (char*)**writeToString:**(char*)*buff*
  size:(int)*buff_size*
  canExpand:(BOOL)*flag*

Writes a white-space delimited, NULL terminated, printable, string representation of the contents of the list to *buff*. *buff_size* is the maximum number of bytes available in *buff*. If the *canExpand* flag is YES then *buff* is expanded using realloc() if it is otherwise too small. If *canExpand* is NO then the string is truncated to *buff_size* and is not NULL terminated. If *buff* is NULL and *canExpand* is YES then a buffer is allocated with malloc() and returned. Returns 0 if *canExpand* is NO and the string was truncated, *buff* if the string was less than *buff_size* characters, or a buffer allocated with realloc() if the buffer was expanded. It is the caller's responsibility to free()

the returned buffer if *canExpand* was YES.

See also: **;MiscIntList.rtf;readFromString:;¬ ± readFromString:**, **;MiscIntList.rtf;writeToString;¬ ± writeToString**, **;MiscIntList.rtf;writeToString:size:;¬ ± writeToString:size:**

Constants and Defined Types

```
MiscIntListCompareFunc;¬typedef int (*MiscIntListCompareFunc)( void* userData, int x, int y );
```