PAStringList

Inherits From: Storage : Object

Declared In: PAStringList.h

Class Description

The PAStringList (a subclass of Storage) is a convenient way to deal with lists of character strings. It contains methods for adding strings, inserting strings, sorting strings and searching for strings.

The PAStringList can also add strings from a delimited string (such as the ones passed to an app by the workspace). The delimiters can be supplied or assumed to be white space.

Finally, the PAStringList implements a browser delegate method so that it can easily display itself if set to be a browser delegate.

To use this object inside of InterfaceBuilder, simply drag its icon from the palettes window into the suitcases window. Then drag in a browser and set its delegate to be the stringlist. Type in some entries into PAStringList's inspector and test interface. The browser fills up!

paste.tiff ¬
Palettes Window

305078_paste.tiff ¬
Example Inspector

918202_paste.tiff ¬
Resulting Browser

Instance Variables

```
BOOL    isSorted;           // Whether or not the list is currently sorted
```

Method Types

```
// Convenience adding methods (defaulting to ifAbsent=NO, noCopy=NO,
//     sorted=NO and at=count)
- addString:(const char *)string;
- addStringIfAbsent:(const char *)string;
- addStringNoCopy:(const char *)string;
- addStringIfAbsentNoCopy:(const char *)string;

- addString:(const char *)string at:(int)at;
- addStringIfAbsent:(const char *)string at:(int)at;
```

```
- addStringNoCopy:(const char *)string at:(int)at;
- addStringIfAbsentNoCopy:(const char *)string at:(int)at;

- addStringSorted:(const char *)string;
- addStringIfAbsentSorted:(const char *)string;
- addStringNoCopySorted:(const char *)string;
- addStringIfAbsentNoCopySorted:(const char *)string;

// The main adding method (all other adds call this)
- addString:(const char *)string ifAbsent:(BOOL)ifAbsent noCopy:(BOOL)noCopy sorted:(BOOL)sorted at:(int)at;

// Adding Multiple strings
- addStrings:(const char *const*)strings;
- addStrings:(const char *const*)strings ifAbsent:(BOOL)ifAbsent noCopy:(BOOL)noCopy sorted:(BOOL)sorted at:(int)at;

// Adding another PAStringList
- addPAStringList:stringListObject;
- addPAStringList:stringListObject ifAbsent:(BOOL)ifAbsent noCopy:(BOOL)noCopy sorted:(BOOL)sorted at:(int)at;

// Adding individual strings from delimited (tab or otherwise) string
- addDelimitedStrings:(const char *)string delimiters:(const char *)dels;
- addDelimitedStrings:(const char *)string delimiters:(const char *)dels ifAbsent:(BOOL)ifAbsent sorted:(BOOL)sorted at:
                                        (int)at;

// Finding strings in list
- (const char *const*)strings;
- (const char *)stringAt:(int)at;
```

```
// Finding the index of a given string and/or whether it is in list
- (BOOL)stringExists:(const char *)string;
- (unsigned)indexOfString:(const char *)string;
- (unsigned)indexOfString:(const char *)string exists:(BOOL *)exists;

// Removing strings from list
- removeString:(const char *)string;
- removeStrings:(const char *const*)strings;
- (char *)removeStringAt:(int)at;

// Sorting list
- (BOOL)isSorted;
- sortStrings:sender;

// Archiving
- write:(NXTypedStream *)stream;
- read:(NXTypedStream *)stream;

// Cleanup
- freeStrings;
- free;

// Bonus NXBrowser support
- (int)browser:sender fillMatrix:matrix inColumn:(int)column;
```

Instance Methods

addString:, addStringIfAbsent: addStringNoCopy: addStringIfAbsentNoCopy:
addString:at:, addStringIfAbsent:at:, addStringIfAbsent:at:
addStringSorted:, addStringIfAbsentSorted:, addStringIfAbsentNoCopySorted:

These methods are convenience methods that simply call the master addString method and provide shorter names for cleaner use assuming the defaults of ifAbsent=NO, noCopy=NO, sorted=NO and at=count. Returns the result of the main addString method (self).

See also: - insertElement:at:

addString:ifAbsent:noCopy:sorted:at:

This method provides a flexible method for adding strings to the list. The ifAbsent flag specifies whether a string should be added if it already exists. The noCopy flag specifies whether the given string should be used (or copied). The sorted flag specifies whether the string should be added alphabetically. The at value specifies where the string should be inserted at if it is not added alphabetically. Returns self.

addStrings, addPAStringList

These methods allow for lists of strings to be added either from a char ** or from a PAStringList object. Returns self.

addDelimitedStrings:delimiters:
addDelimitedStrings:delimiters:ifAbsent:sorted:at:

This method takes a delimited string (like the ones passed from the workspace) and searches for the given delimiters (NULL for general whitespace). It adds each string that it finds between the delimiters using the addString method. Returns

self.

(const char *const*)strings
(const char *)stringAt:(int)at

The strings methods returns all of the strings as an array of char pointers. This array is NOT null terminated.
The stringAt: method returns the string at a particular index.

(BOOL)stringExists
(unsigned int)indexOfString
(unsigned int)indexOfString exists:(BOOL *)exists

stringExists returns whether or not the string is already in the list. indexOfString returns either the strings current index or the index that it should be alphabetically(returning [self count] if not sorted).
indexOfString:stringExists: is a composite method that returns both values in about the same amount of time that it takes to compute one.

removeString:(const char *)string
removeStrings:(const char *const*)strings
removeStringAt:(int)at

These methods allow for the removal of strings by value or index. Returns self.

(BOOL)isSorted
sortStrings:sender

isSorted returns whether or not the list is currently considered to be sorted. Lists are set to be sorted by default and when empty. isSorted is set to NO when a string is added without the 'sorted' flag.

sortStrings: sorts the stringList (ignoring case) and sets the sorted flag. Returns self.

empty, freeStrings, free

These methods empty the list, free the strings and free the list, respectively.

(int)browser:sender fillMatrix:matrix inColumn:(int)column

This method is provided as a delegate method for browser to quickly display string list.