CFMutableArray Reference

Core Foundation

2005-12-06

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Simultaneously published in the United States and Canada.

Contents

CFMutableArray Reference

Derived From:	CFArray : CFPropertyList : CFType
Framework:	CoreFoundation/CoreFoundation.h
Declared in	CFArray.h
Companion guides	Collections Programming Topics for Core Foundation Property List Programming Topics for Core Foundation

Overview

CFMutableArray manages dynamic arrays. The basic interface for managing arrays is provided by CFArray. CFMutableArray adds functions to modify the contents of an array.

You create a mutable array object using either the `CFArrayCreateMutable` (page 7) or `CFArrayCreateMutableCopy` (page 8) function.

CFMutableArray provides several functions for changing the contents of an array, for example the `CFArrayAppendValue` (page 6) and `CFArrayInsertValueAtIndex` (page 9) functions add values to an array and `CFArrayRemoveValueAtIndex` (page 10) removes values from an array. You can also reorder the contents of an array using `CFArrayExchangeValuesAtIndices` (page 9) and `CFArraySortValues` (page 13).

CFMutableArray is "toll-free bridged" with its Cocoa Foundation counterpart, NSMutableArray. This means that the Core Foundation type is interchangeable in function or method calls with the bridged Foundation object. Therefore, in a method where you see an `NSMutableArray *` parameter, you can pass in a `CFMutableArrayRef`, and in a function where you see a `CFMutableArrayRef` parameter, you can pass in an NSMutableArray instance. This fact also applies to concrete subclasses of NSMutableArray. See Interchangeable Data Types for more information on toll-free bridging.

Functions

CFArrayAppendArray

Adds the values from one array to another array.

```
void CFArrayAppendArray (
   CFMutableArrayRef theArray,
   CFArrayRef otherArray,
   CFRange otherRange
);
```

Parameters

theArray

The array to which values from *otherArray* are added. If *theArray* is a limited-capacity array, adding `otherRange.length` values from *otherArray* must not cause the capacity limit of *theArray* to be exceeded.

otherArray

An array providing the values to be added to *theArray*.

otherRange

The range within *otherArray* from which to add the values to *theArray*. The range must not exceed the index space of *otherArray*.

Discussion

The new values are retained by *theArray* using the retain callback provided when *theArray* was created. If the values are not of the type expected by the retain callback, the behavior is undefined. The values are assigned to the indices one larger than the previous largest index in *theArray*, and beyond, and the count of *theArray* is increased by `otherRange.length`. The values are assigned new indices in *theArray* from smallest to largest index in the order in which they appear in *otherArray*.

Availability

Available in CarbonLib v1.1 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

ImageBrowserView

Declared In

`CFArray.h`

CFArrayAppendValue

Adds a value to an array giving it the new largest index.

```
void CFArrayAppendValue (
   CFMutableArrayRef theArray,
   const void *value
);
```

Parameters

theArray

The array to which *value* is to be added. If *theArray* is a limited-capacity array and it is full before this operation, the behavior is undefined.

value

A CFType object or a pointer value to add to *theArray*.

Discussion
The *value* parameter is retained by *theArray* using the retain callback provided when *theArray* was created. If *value* is not of the type expected by the retain callback, the behavior is undefined. The *value* parameter is assigned to the index one larger than the previous largest index and the count of *theArray* is increased by one.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
HID Explorer
ImageBrowserView
MoreIsBetter
MoreSCF
QISA

Declared In
CFArray.h

CFArrayCreateMutable

Creates a new empty mutable array.

```
CFMutableArrayRef CFArrayCreateMutable (
   CFAllocatorRef allocator,
   CFIndex capacity,
   const CFArrayCallBacks *callBacks
);
```

Parameters

allocator

The allocator to use to allocate memory for the new array and its storage for values. Pass NULL or kCFAllocatorDefault to use the current default allocator.

capacity

The maximum number of values that can be contained by the new array. The array starts empty and can grow to this number of values (and it can have less). If this parameter is 0, the array's maximum capacity is not limited. The value must not be negative.

callBacks

A pointer to a CFArrayCallBacks structure initialized with the callbacks for the array to use on each value in the array. A copy of the contents of the callbacks structure is made, so that a pointer to a structure on the stack can be passed in or can be reused for multiple array creations. This parameter may be NULL, which is treated as if a valid structure of version 0 with all fields NULL had been passed in.

If any of the fields are not valid pointers to functions of the correct type, or this parameter is not a valid pointer to a CFArrayCallBacks structure, the behavior is undefined. If any value put into the array is not one understood by one of the callback functions, the behavior when that callback function is used is undefined. If the array contains CFType objects only, then pass kCFTypeArrayCallBacks as this parameter to use the default callback functions.

Return Value
A new mutable array, or NULL if there was a problem creating the object. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
HID Config Save
HID Explorer
ImageBrowserView
ImageClient
QISA

Declared In
`CFArray.h`

CFArrayCreateMutableCopy

Creates a new mutable array with the values from another array.

```
CFMutableArrayRef CFArrayCreateMutableCopy (
   CFAllocatorRef allocator,
   CFIndex capacity,
   CFArrayRef theArray
);
```

Parameters

allocator
The allocator to use to allocate memory for the new array and its storage for values. Pass `NULL` or `kCFAllocatorDefault` to use the current default allocator.

capacity
The maximum number of values that can be contained by the new array. The array starts with the same count as *theArray* and can grow to this number of values (and it can have less). If this parameter is `0`, the array's maximum capacity is not limited. The capacity must not be negative, and must be greater than or equal to the count of *theArray*.

theArray
The array to copy. The pointer values from the array are copied into the new array. However, the values are also retained by the new array.

Return Value
A new mutable array that contains the same values as *theArray*. The new array has the same count as the *theArray* and uses the same callbacks. Ownership follows the Create Rule.

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Related Sample Code
CFPreferences
ImageBrowserView
MoreSCF
QISA
QTCarbonShell

Declared In
CFArray.h

CFArrayExchangeValuesAtIndices

Exchanges the values at two indices of an array.

```
void CFArrayExchangeValuesAtIndices (
   CFMutableArrayRef theArray,
   CFIndex idx1,
   CFIndex idx2
);
```

Parameters

theArray
The array that contains the values to be swapped.

idx1
The index of the value to swap with the value at *idx2*. The index must not exceed the index space of *theArray* (0 to N-1 inclusive, where N is the count of *theArray* before the operation).

idx2
The index of the value to swap with the value at *idx1*. The index must not exceed the index space of *theArray* (0 to N-1 inclusive, where N is the count of *theArray* before the operation).

Availability
Available in CarbonLib v1.0 and later.
Available in Mac OS X v10.0 and later.

Declared In
CFArray.h

CFArrayInsertValueAtIndex

Inserts a value into an array at a given index.

```
void CFArrayInsertValueAtIndex (
   CFMutableArrayRef theArray,
   CFIndex idx,
   const void *value
);
```

Parameters

theArray
The array into which *value* is inserted. If *theArray* is a fixed-capacity array and it is full before this operation, the behavior is undefined.

idx
The index at which to insert *value*. The index must not exceed the index space of *theArray* (0 to N-1 inclusive, where N is the count of *theArray* before the operation. If the index is the same as the count of *theArray*, this function has the same effect as CFArrayAppendValue (page 6).

value

The value to insert into *theArray*. The value is retained by *theArray* using the retain callback provided when *theArray* was created. If *value* is not of the type expected by the retain callback, the behavior is undefined.

Discussion

The *value* parameter is assigned to the index *idx*, and all values in *theArray* with equal and larger indices have their indices increased by one.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

CFPreferences

ComboBoxPrefs

ImageBrowserView

QTCarbonShell

RecentItems

Declared In

`CFArray.h`

CFArrayRemoveAllValues

Removes all the values from an array, making it empty.

```
void CFArrayRemoveAllValues (
   CFMutableArrayRef theArray
);
```

Parameters

theArray

The array from which all of the values are removed.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

ImageClient

Declared In

`CFArray.h`

CFArrayRemoveValueAtIndex

Removes the value at a given index from an array.

```
void CFArrayRemoveValueAtIndex (
   CFMutableArrayRef theArray,
   CFIndex idx
);
```

Parameters

theArray

The array from which the value is to be removed.

idx

The index of the value to remove. The value not lie outside the index space of *theArray* (0 to N-1 inclusive, where N is the count of *theArray* before the operation).

Discussion

All values in *theArray* with indices larger than *idx* have their indices decreased by one.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

ComboBoxPrefs

ImageClient

MoreSCF

QISA

QTCarbonShell

Declared In

CFArray.h

CFArrayReplaceValues

Replaces a range of values in an array.

```
void CFArrayReplaceValues (
   CFMutableArrayRef theArray,
   CFRange range,
   const void **newValues,
   CFIndex newCount
);
```

Parameters

theArray

The array in which some values are to be replaced. If this parameter is not a valid CFMutableArray object, the behavior is undefined.

range

The range of values within *theArray* to replace. The range location or end point (defined by the location plus length minus 1) must not lie outside the index space of *theArray* (0 to N-1 inclusive, where N is the count of *theArray*). The range length must not be negative. The range may be empty (length 0), in which case the new values are merely inserted at the range location.

newValues

A C array of the pointer-sized values to be placed into *theArray*. The new values in *theArray* are ordered in the same order in which they appear in this C array. This parameter may be NULL if the *newCount* parameter is 0. This C array is not changed or freed by this function. If this parameter is not a valid pointer to a C array of at least *newCount* pointers, the behavior is undefined.

newCount

The number of values to copy from the *newValues* C array into *theArray*. If this parameter is different from the range length, the excess *newCount* values are inserted after the range or the excess range values are deleted. This parameter may be 0, in which case no new values are replaced into *theArray* and the values in the range are simply removed. If this parameter is negative or greater than the number of values actually in the *newValues* C array, the behavior is undefined.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

DockBrowser

Declared In

CFArray.h

CFArraySetValueAtIndex

Changes the value at a given index in an array.

```
void CFArraySetValueAtIndex (
   CFMutableArrayRef theArray,
   CFIndex idx,
   const void *value
);
```

Parameters

theArray

The array in which the value is to be changed.

idx

The index at which to set the new value. The value must not lie outside the index space of *theArray* (0 to N-1 inclusive, where N is the count of the array before the operation).

value

The value to set in *theArray*. The value is retained by *theArray* using the retain callback provided when *theArray* was created and the previous value at *idx* is released. If the value is not of the type expected by the retain callback, the behavior is undefined. The indices of other values are not affected.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

AlbumToSlideshow

Watcher

Declared In

CFArray.h

CFArraySortValues

Sorts the values in an array using a given comparison function.

```
void CFArraySortValues (
   CFMutableArrayRef theArray,
   CFRange range,
   CFComparatorFunction comparator,
   void *context
);
```

Parameters

theArray

The array whose values are sorted.

range

The range of values within *theArray* to sort. The range location or end point (defined by the location plus length minus 1) must not lie outside the index space of *theArray* (`0` to `N-1` inclusive, where `N` is the count of *theArray*). The range length must not be negative. The range may be empty (length 0).

comparator

The function with the comparator function type signature that is used in the sort operation to compare the values in *theArray*. If this parameter is not a pointer to a function of the correct prototype, the behavior is undefined. If there are values in *theArray* that the *comparator* function does not expect or cannot properly compare, the behavior is undefined. The values in the range are sorted from least to greatest according to this function.

context

A pointer-sized program-defined value, which is passed as the third parameter to the *comparator* function, but is otherwise unused by this function. If the context is not what is expected by the *comparator* function, the behavior is undefined.

Availability

Available in CarbonLib v1.0 and later.

Available in Mac OS X v10.0 and later.

Related Sample Code

DockBrowser

HID Explorer

MoreIsBetter

MoreSCF

QISA

Declared In

`CFArray.h`

Data Types

CFMutableArrayRef

A reference to a mutable array object.

```
typedef struct __CFArray *CFMutableArrayRef;
```

Availability
Available in Mac OS X v10.0 and later.

Declared In
`CFArray.h`

Document Revision History

This table describes the changes to *CFMutableArray Reference*.

Date	Notes
2005-12-06	Updated the link to the companion document.
2005-04-29	Moved Introduction to new Introduction page.
2003-08-01	Enhanced description of all the `kCFType*Callbacks` and added link to Carbon-Cocoa integration document.
2003-01-01	First version of this document.

Document Revision History

Index

C