NSMutableCharacterSet Class Reference

Cocoa > Data Management

2006-05-23

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

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

Apple, the Apple logo, and Cocoa are trademarks of Apple Inc., registered in the United States and other countries.

iPhone is a trademark of Apple Inc.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

NSMutableCharacterSet Class Reference 5

Overview 5
Tasks 5
  Adding and Removing Characters 5
  Combining Character Sets 6
  Inverting a Character Set 6
Instance Methods 6
  addCharactersInRange: 6
  addCharactersInString: 7
  formIntersectionWithCharacterSet: 7
  formUnionWithCharacterSet: 8
  invert 8
  removeCharactersInRange: 8
  removeCharactersInString: 9

Document Revision History 11

Index 13

NSMutableCharacterSet Class Reference

Inherits from	NSCharacterSet : NSObject
Conforms to	NSCopying NSMutableCopying NSCoding (NSCharacterSet) NSCopying (NSCharacterSet) NSMutableCopying (NSCharacterSet) NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	String Programming Guide for Cocoa
Declared in	NSCharacterSet.h
Related sample code	ImageMapExample

Overview

The `NSMutableCharacterSet` class declares the programmatic interface to objects that manage a modifiable set of Unicode characters. You can add or remove characters from a mutable character set as numeric values in `NSRange` structures or as character values in strings, combine character sets by union or intersection, and invert a character set.

Mutable character sets are less efficient to use than immutable character sets. If you don't need to change a character set after creating it, create an immutable copy with `copy` and use that.

`NSMutableCharacterSet` defines no primitive methods. Subclasses must implement all methods declared by this class in addition to the primitives of `NSCharacterSet`. They must also implement `mutableCopyWithZone:`.

Tasks

Adding and Removing Characters

`– addCharactersInRange:` (page 6)

Adds to the receiver the characters whose Unicode values are in a given range.

- `removeCharactersInRange:` (page 8)
  Removes from the receiver the characters whose Unicode values are in a given range.
- `addCharactersInString:` (page 7)
  Adds to the receiver the characters in a given string.
- `removeCharactersInString:` (page 9)
  Removes from the receiver the characters in a given string.

Combining Character Sets

- `formIntersectionWithCharacterSet:` (page 7)
  Modifies the receiver so it contains only characters that exist in both the receiver and *otherSet*.
- `formUnionWithCharacterSet:` (page 8)
  Modifies the receiver so it contains all characters that exist in either the receiver or *otherSet*.

Inverting a Character Set

- `invert` (page 8)
  Replaces all the characters in the receiver with all the characters it didn't previously contain.

Instance Methods

addCharactersInRange:

Adds to the receiver the characters whose Unicode values are in a given range.

```
- (void)addCharactersInRange:(NSRange)aRange
```

Parameters

aRange

The range of characters to add.

aRange.location is the value of the first character to add; *aRange*.location + *aRange*.length-1 is the value of the last. If *aRange*.length is 0, this method has no effect.

Discussion

This code excerpt adds to a character set the lowercase English alphabetic characters:

```
NSMutableCharacterSet *aCharacterSet = [[NSMutableCharacterSet alloc] init];
NSRange lcEnglishRange;

lcEnglishRange.location = (unsigned int)'a';
lcEnglishRange.length = 26;
[aCharacterSet addCharactersInRange:lcEnglishRange];
```

Availability

Available in Mac OS X v10.0 and later.

See Also

- `removeCharactersInRange:` (page 8)
- `addCharactersInString:` (page 7)

Declared In

`NSCharacterSet.h`

addCharactersInString:

Adds to the receiver the characters in a given string.

```
- (void)addCharactersInString:(NSString *)aString
```

Parameters

`aString`

The characters to add to the receiver.

Discussion

This method has no effect if *`aString`* is empty.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `removeCharactersInString:` (page 9)
- `addCharactersInRange:` (page 6)

Related Sample Code

ImageMapExample

Declared In

`NSCharacterSet.h`

formIntersectionWithCharacterSet:

Modifies the receiver so it contains only characters that exist in both the receiver and *`otherSet`*.

```
- (void)formIntersectionWithCharacterSet:(NSCharacterSet *)otherSet
```

Parameters

`otherSet`

The character set with which to perform the intersection.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `formUnionWithCharacterSet:` (page 8)

Declared In

`NSCharacterSet.h`

formUnionWithCharacterSet:

Modifies the receiver so it contains all characters that exist in either the receiver or `otherSet`.

```
- (void)formUnionWithCharacterSet:(NSCharacterSet *)otherSet
```

Availability

Available in Mac OS X v10.0 and later.

See Also

`- formIntersectionWithCharacterSet:` (page 7)

Declared In

`NSCharacterSet.h`

invert

Replaces all the characters in the receiver with all the characters it didn't previously contain.

```
- (void)invert
```

Discussion

Inverting a mutable character set, whether by `invert` or by `invertedSet`, is much less efficient than inverting an immutable character set with `invertedSet`.

Availability

Available in Mac OS X v10.0 and later.

See Also

`- invertedSet` (NSCharacterSet)

Declared In

`NSCharacterSet.h`

removeCharactersInRange:

Removes from the receiver the characters whose Unicode values are in a given range.

```
- (void)removeCharactersInRange:(NSRange)aRange
```

Parameters

aRange

The range of characters to remove.

aRange`.location` is the value of the first character to remove; *aRange*`.location +` *aRange*`.length- 1` is the value of the last. If *aRange*`.length` is `0`, this method has no effect.

Availability

Available in Mac OS X v10.0 and later.

See Also

`- addCharactersInRange:` (page 6)

`- removeCharactersInString:` (page 9)

Declared In

`NSCharacterSet.h`

removeCharactersInString:

Removes from the receiver the characters in a given string.

```
- (void)removeCharactersInString:(NSString *)aString
```

Parameters

aString

The characters to remove from the receiver.

Discussion

This method has no effect if *aString* is empty.

Availability

Available in Mac OS X v10.0 and later.

See Also

- `- addCharactersInString:` (page 7)
- `- removeCharactersInRange:` (page 8)

Declared In

`NSCharacterSet.h`

Document Revision History

This table describes the changes to *NSMutableCharacterSet Class Reference*.

Date	Notes
2006-05-23	First publication of this content as a separate document.

Index

A

addCharactersInRange: instance method 6
addCharactersInString: instance method 7

F

formIntersectionWithCharacterSet: instance method 7
formUnionWithCharacterSet: instance method 8

I

invert instance method 8

R

removeCharactersInRange: instance method 8
removeCharactersInString: instance method 9