
NSUserDefaults Class Reference

Data Management: Preference Settings





Apple Inc.
© 2009 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

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, Cocoa, Mac, and Mac OS 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

NSUserDefaults Class Reference 5

Overview	5
Persistence of NSURL and file reference URLs	6
Tasks	6
Getting the Shared NSUserDefaults Instance	6
Initializing an NSUserDefaults Object	6
Registering Defaults	7
Getting Default Values	7
Setting Default Values	7
Removing Defaults	8
Maintaining Persistent Domains	8
Accessing Managed Environment Keys	8
Managing the Search List	8
Maintaining Volatile Domains	8
Maintaining Suites	9
Class Methods	9
resetStandardUserDefaults	9
standardUserDefaults	9
Instance Methods	10
addSuiteNamed:	10
arrayForKey:	10
boolForKey:	11
dataForKey:	11
dictionaryForKey:	12
dictionaryRepresentation	13
doubleForKey:	13
floatForKey:	13
init	14
initWithUser:	14
integerForKey:	15
objectForKey:	15
objectIsForcedForKey:	16
objectIsForcedForKey:inDomain:	17
persistentDomainForName:	17
persistentDomainNames	18
registerDefaults:	18
removeObjectForKey:	19
removePersistentDomainForName:	19
removeSuiteNamed:	20
removeVolatileDomainForName:	20
setBool:forKey:	21

- setDouble:forKey: 21
- setFloat:forKey: 21
- setInteger:forKey: 22
- setObject:forKey: 22
- setPersistentDomain:forName: 23
- setURL:forKey: 24
- setVolatileDomain:forName: 24
- stringArrayForKey: 25
- stringForKey: 25
- synchronize 26
- URLForKey: 27
- volatileDomainForName: 27
- volatileDomainNames 28
- Constants 28
 - NSUserDefaults Domains 28
 - Language-Dependent Date/Time Information 29
 - Language-Dependent Numeric Information 34
- Notifications 36
 - NSUserDefaultsDidChangeNotification 36

Document Revision History 37

Index 39

NSUserDefaults Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/Foundation.framework
Availability	Available in Mac OS X v10.0 and later.
Companion guide	User Defaults Programming Topics for Cocoa
Declared in	NSUserDefaults.h
Related sample code	IColorTracking GLUT Quartz Composer QCTV Quartz Composer WWDC 2005 TextEdit Sproing

Overview

The `NSUserDefaults` class provides a programmatic interface for interacting with the defaults system. The defaults system allows an application to customize its behavior to match a user's preferences. For example, you can allow users to determine what units of measurement your application displays or how often documents are automatically saved. Applications record such preferences by assigning values to a set of parameters in a user's defaults database. The parameters are referred to as defaults since they're commonly used to determine an application's default state at startup or the way it acts by default.

At runtime, you use an `NSUserDefaults` object to read the defaults that your application uses from a user's defaults database. `NSUserDefaults` caches the information to avoid having to open the user's defaults database each time you need a default value. The [synchronize](#) (page 26) method, which is automatically invoked at periodic intervals, keeps the in-memory cache in sync with a user's defaults database.

The `NSUserDefaults` class provides convenience methods for accessing common types such as floats, doubles, integers, Booleans, and URLs. A default object must be a property list, that is, an instance of (or for collections a combination of instances of): `NSData`, `NSString`, `NSNumber`, `NSDate`, `NSArray`, or `NSDictionary`. If you want to store any other type of object, you should typically archive it to create an instance of `NSData`. For more details, see *User Defaults Programming Topics for Cocoa*.

Values returned from `NSUserDefaults` are *immutable*, even if you set a mutable object as the value. For example, if you set a mutable string as the value for "MyStringDefault", the string you later retrieve using [stringForKey:](#) (page 25) will be immutable.

A defaults database is created automatically for each user. The `NSUserDefaults` class does not currently support per-host preferences. To do this, you must use the `CFPreferences` API (see *Preferences Utilities Reference*). However, `NSUserDefaults` correctly reads per-host preferences, so you can safely mix `CFPreferences` code with `NSUserDefaults` code.

If your application supports managed environments, you can use an `NSUserDefaults` object to determine which preferences are managed by an administrator for the benefit of the user. Managed environments correspond to computer labs or classrooms where an administrator or teacher may want to configure the systems in a particular way. In these situations, the teacher can establish a set of default preferences and force those preferences on users. If a preference is managed in this manner, applications should prevent users from editing that preference by disabling any appropriate controls.

The `NSUserDefaults` class is thread-safe.

Persistence of NSURL and file reference URLs

When using `NSURL` instances to refer to files within a process, it's important to make the distinction between location-based tracking (file: scheme URLs that are basically paths) versus filesystem identity tracking (file: scheme URLs that are file reference URLs). When persisting an `NSURL`, you should take that behavior into consideration. If your application tracks the resource being located by its identity so that it can be found if the user moves the file, then you should explicitly write the `NSURL`'s bookmark data or encode a file reference URL.

If you want to track a file by reference but you require explicit control over when resolution occurs, you should take care to write out bookmark data to `NSUserDefaults` rather than rely on `+[NSUserDefaults setURL:forKey:]`. This allows you to call `+[NSURL URLByResolvingBookmarkData:options:relativeToURL:bookmarkDataIsStale:error:]` at a time when you know your application will be able to handle the potential I/O or required user interface interactions.

Tasks

Getting the Shared NSUserDefaults Instance

- + [standardUserDefaults](#) (page 9)
Returns the shared defaults object.
- + [resetStandardUserDefaults](#) (page 9)
Synchronizes any changes made to the shared user defaults object and releases it from memory.

Initializing an NSUserDefaults Object

- [init](#) (page 14)
Returns an `NSUserDefaults` object initialized with the defaults for the current user account.
- [initWithUser:](#) (page 14)
Returns an `NSUserDefaults` object initialized with the defaults for the specified user account.

Registering Defaults

- [registerDefaults:](#) (page 18)
Adds the contents the specified dictionary to the registration domain.

Getting Default Values

- [arrayForKey:](#) (page 10)
Returns the array associated with the specified key.
- [boolForKey:](#) (page 11)
Returns the Boolean value associated with the specified key.
- [dataForKey:](#) (page 11)
Returns the data object associated with the specified key.
- [dictionaryForKey:](#) (page 12)
Returns the dictionary object associated with the specified key.
- [floatForKey:](#) (page 13)
Returns the floating-point value associated with the specified key.
- [integerForKey:](#) (page 15)
Returns the integer value associated with the specified key..
- [objectForKey:](#) (page 15)
Returns the object associated with the first occurrence of the specified default.
- [stringArrayForKey:](#) (page 25)
Returns the array of strings associated with the specified key.
- [stringForKey:](#) (page 25)
Returns the string associated with the specified key.
- [doubleForKey:](#) (page 13)
Returns the double value associated with the specified key.
- [URLForKey:](#) (page 27)
Returns the NSURL instance associated with the specified key.

Setting Default Values

- [setBool:forKey:](#) (page 21)
Sets the value of the specified default key to the specified Boolean value.
- [setFloat:forKey:](#) (page 21)
Sets the value of the specified default key to the specified floating-point value.
- [setInteger:forKey:](#) (page 22)
Sets the value of the specified default key to the specified integer value.
- [setObject:forKey:](#) (page 22)
Sets the value of the specified default key in the standard application domain.
- [setDouble:forKey:](#) (page 21)
Sets the value of the specified default key to the double value.

- [setURL:forKey:](#) (page 24)
Sets the value of the specified default key to the specified URL.

Removing Defaults

- [removeObjectForKey:](#) (page 19)
Removes the value of the specified default key in the standard application domain.

Maintaining Persistent Domains

- [synchronize](#) (page 26)
Writes any modifications to the persistent domains to disk and updates all unmodified persistent domains to what is on disk.
- [persistentDomainForName:](#) (page 17)
Returns a dictionary containing the keys and values in the specified persistent domain.
- [persistentDomainNames](#) (page 18)
Returns an array of the current persistent domain names.
- [removePersistentDomainForName:](#) (page 19)
Removes the contents of the specified persistent domain from the user's defaults.
- [setPersistentDomain:forName:](#) (page 23)
Sets the dictionary for the specified persistent domain.

Accessing Managed Environment Keys

- [objectIsForcedForKey:](#) (page 16)
Returns a Boolean value indicating whether the specified key is managed by an administrator.
- [objectIsForcedForKey:inDomain:](#) (page 17)
Returns a Boolean value indicating whether the key in the specified domain is managed by an administrator.

Managing the Search List

- [dictionaryRepresentation](#) (page 13)
Returns a dictionary that contains a union of all key-value pairs in the domains in the search list.

Maintaining Volatile Domains

- [removeVolatileDomainForName:](#) (page 20)
Removes the specified volatile domain from the user's defaults.
- [setVolatileDomain:forName:](#) (page 24)
Sets the dictionary for the specified volatile domain.
- [volatileDomainForName:](#) (page 27)
Returns the dictionary for the specified volatile domain.

- [volatileDomainNames](#) (page 28)
Returns an array of the current volatile domain names.

Maintaining Suites

- [addSuiteNamed:](#) (page 10)
Inserts the specified domain name into the receiver's search list.
- [removeSuiteNamed:](#) (page 20)
Removes the specified domain name from the receiver's search list.

Class Methods

resetStandardUserDefaults

Synchronizes any changes made to the shared user defaults object and releases it from memory.

```
+ (void)resetStandardUserDefaults
```

Discussion

A subsequent invocation of [standardUserDefaults](#) (page 9) creates a new shared user defaults object with the standard search list.

Availability

Available in Mac OS X v10.0 and later.

Declared In

NSUserDefaults.h

standardUserDefaults

Returns the shared defaults object.

```
+ (NSUserDefaults *)standardUserDefaults
```

Return Value

The shared defaults object.

Discussion

If the shared defaults object does not exist yet, it is created with a search list containing the names of the following domains, in this order:

- `NSArgumentDomain`, consisting of defaults parsed from the application's arguments
- A domain identified by the application's bundle identifier
- `NSGlobalDomain`, consisting of defaults meant to be seen by all applications
- Separate domains for each of the user's preferred languages
- `NSRegistrationDomain`, a set of temporary defaults whose values can be set by the application to ensure that searches will always be successful

The defaults are initialized for the current user. Subsequent modifications to the standard search list remain in effect even when this method is invoked again—the search list is guaranteed to be standard only the first time this method is invoked. The shared instance is provided as a convenience—you can create custom instances using `alloc` along with `initWithUser:` (page 14) or `init` (page 14).

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

CIColorTracking

GLUT

IKSlideshowDemo

Quartz Composer QCTV

Quartz Composer WWDC 2005 TextEdit

Declared In

NSUserDefaults.h

Instance Methods

addSuiteNamed:

Inserts the specified domain name into the receiver's search list.

```
- (void)addSuiteNamed:(NSString *)suiteName
```

Parameters

suiteName

The domain name to insert. This domain is inserted after the application domain.

Discussion

The *suiteName* domain is similar to a bundle identifier string, but is not tied to a particular application or bundle. A suite can be used to hold preferences that are shared between multiple applications.

Availability

Available in Mac OS X v10.0 and later.

See Also

+ [standardUserDefaults](#) (page 9)

- [removeSuiteNamed:](#) (page 20)

Declared In

NSUserDefaults.h

arrayForKey:

Returns the array associated with the specified key.

```
- (NSArray *)arrayForKey:(NSString *)defaultName
```

Parameters*defaultName*

A key in the current user's defaults database.

Return ValueThe array associated with the specified key, or `nil` if the key does not exist or its value is not an `NSArray` object.**Special Considerations**

The returned array and its contents are immutable, even if the values you originally set were mutable.

Availability

Available in Mac OS X v10.0 and later.

See Also- [setObject:forKey:](#) (page 22)**Related Sample Code**

Quartz Composer WWDC 2005 TextEdit

Declared In

NSUserDefaults.h

boolForKey:

Returns the Boolean value associated with the specified key.

- (BOOL)boolForKey:(NSString *)*defaultName***Parameters***defaultName*

A key in the current user's defaults database.

Return ValueIf a boolean value is associated with *defaultName* in the user defaults, that value is returned. Otherwise, `NO` is returned.**Availability**

Available in Mac OS X v10.0 and later.

See Also- [setBool:forKey:](#) (page 21)**Related Sample Code**

Sproing

Declared In

NSUserDefaults.h

dataForKey:

Returns the data object associated with the specified key.

- (NSData *)dataForKey:(NSString *)*defaultName*

Parameters

defaultName

A key in the current user's defaults database.

Return Value

The data object associated with the specified key, or `nil` if the key does not exist or its value is not an `NSData` object.

Special Considerations

The returned data object is immutable, even if the value you originally set was a mutable data object.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setObject:forKey:](#) (page 22)

Related Sample Code

QTQuartzPlayer

Declared In

NSUserDefaults.h

dictionaryForKey:

Returns the dictionary object associated with the specified key.

- (NSDictionary *)dictionaryForKey:(NSString *)*defaultName*

Parameters

defaultName

A key in the current user's defaults database.

Return Value

The dictionary object associated with the specified key, or `nil` if the key does not exist or its value is not an `NSDictionary` object.

Special Considerations

The returned dictionary and its contents are immutable, even if the values you originally set were mutable.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setObject:forKey:](#) (page 22)

Declared In

NSUserDefaults.h

dictionaryRepresentation

Returns a dictionary that contains a union of all key-value pairs in the domains in the search list.

- (NSDictionary *)dictionaryRepresentation

Return Value

A dictionary containing the keys. The keys are names of defaults and the value corresponding to each key is a property list object (NSData, NSString, NSNumber, NSDate, NSArray, or NSDictionary).

Discussion

As with [objectForKey:](#) (page 15), key-value pairs in domains that are earlier in the search list take precedence. The combined result does not preserve information about which domain each entry came from.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

NewsReader

OpenCL NBody Simulation Example

Declared In

NSUserDefaults.h

doubleForKey:

Returns the double value associated with the specified key.

- (double)doubleForKey:(NSString *)defaultName

Parameters

defaultName

A key in the current user's defaults database.

Return Value

The double value associated with the specified key. If the key does not exist, this method returns 0.

Availability

Available in Mac OS X v10.5 and later.

See Also

- [setDouble:forKey:](#) (page 21)

Declared In

NSUserDefaults.h

floatForKey:

Returns the floating-point value associated with the specified key.

- (float)floatForKey:(NSString *)defaultName

Parameters*defaultName*

A key in the current user's defaults database.

Return Value

The floating-point value associated with the specified key. If the key does not exist, this method returns 0.

Availability

Available in Mac OS X v10.0 and later.

See Also- [setFloat:forKey:](#) (page 21)**Declared In**

NSUserDefaults.h

init

Returns an NSUserDefaults object initialized with the defaults for the current user account.

- (id)init

Return Value

An initialized NSUserDefaults object whose argument and registration domains are already set up.

Discussion

This method does not put anything in the search list. Invoke it only if you've allocated your own NSUserDefaults instance instead of using the shared one.

Availability

Available in Mac OS X v10.0 and later.

See Also+ [standardUserDefaults](#) (page 9)**Declared In**

NSUserDefaults.h

initWithUser:

Returns an NSUserDefaults object initialized with the defaults for the specified user account.

- (id)initWithUser:(NSString *)username

Parameters*username*

The name of the user account.

Return Value

An initialized NSUserDefaults object whose argument and registration domains are already set up. If the current user does not have access to the specified user account, this method returns nil.

Discussion

This method does not put anything in the search list. Invoke it only if you've allocated your own `NSUserDefaults` instance instead of using the shared one.

You do not normally use this method to initialize an instance of `NSUserDefaults`. Applications used by a superuser might use this method to update the defaults databases for a number of users. The user who started the application must have appropriate access (read, write, or both) to the defaults database of the new user, or this method returns `nil`.

Availability

Available in Mac OS X v10.0 and later.

See Also

+ [standardUserDefaults](#) (page 9)

Declared In

`NSUserDefaults.h`

integerForKey:

Returns the integer value associated with the specified key..

```
- (NSInteger)integerForKey:(NSString *)defaultName
```

Parameters

defaultName

A key in the current user's defaults database.

Return Value

The integer value associated with the specified key. If the specified key does not exist, this method returns 0.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setInteger:forKey:](#) (page 22)

Related Sample Code

ClipboardViewer

IKSlideshowDemo

NumberInput_IMKit_Sample

Sproing

Declared In

`NSUserDefaults.h`

objectForKey:

Returns the object associated with the first occurrence of the specified default.

```
- (id)objectForKey:(NSString *)defaultName
```

Parameters*defaultName*

A key in the current user's defaults database.

Return ValueThe object associated with the specified key, or `nil` if the key was not found.**Discussion**

This method searches the domains included in the search list in the order they are listed.

Special Considerations

The returned object is immutable, even if the value you originally set was mutable.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [arrayForKey:](#) (page 10)
- [dataForKey:](#) (page 11)
- [dictionaryForKey:](#) (page 12)
- [stringArrayForKey:](#) (page 25)
- [stringForKey:](#) (page 25)

Related Sample Code

FunHouse

ImageApp

PrefsPane

Quartz Composer QCTV

Quartz Composer WWDC 2005 TextEdit

Declared In

NSUserDefaults.h

objectIsForcedForKey:

Returns a Boolean value indicating whether the specified key is managed by an administrator.

- (BOOL)objectIsForcedForKey:(NSString *)key

Parameters*key*

The key whose status you want to check.

Return Value

YES if the value of the specified key is managed by an administrator, otherwise NO.

Discussion

This method assumes that the key is a preference associated with the current user and application. For managed keys, the application should disable any user interface that allows the user to modify the value of *key*.

Availability

Available in Mac OS X v10.2 and later.

See Also

- [objectIsForcedForKey:inDomain:](#) (page 17)

Declared In

NSUserDefaults.h

objectIsForcedForKey:inDomain:

Returns a Boolean value indicating whether the key in the specified domain is managed by an administrator.

```
- (BOOL)objectIsForcedForKey:(NSString *)key inDomain:(NSString *)domain
```

Parameters

key

The key whose status you want to check.

domain

The domain of the key.

Return Value

YES if the key is managed by an administrator in the specified domain, otherwise NO.

Discussion

This method assumes that the key is a preference associated with the current user. For managed keys, the application should disable any user interface that allows the user to modify the value of *key*.

Availability

Available in Mac OS X v10.2 and later.

See Also

- [objectIsForcedForKey:](#) (page 16)

Declared In

NSUserDefaults.h

persistentDomainForName:

Returns a dictionary containing the keys and values in the specified persistent domain.

```
- (NSDictionary *)persistentDomainForName:(NSString *)domainName
```

Parameters

domainName

The domain whose keys and values you want. This value should be equal to your application's bundle identifier.

Return Value

A dictionary containing the keys. The keys are names of defaults and the value corresponding to each key is a property list object (NSData, NSString, NSNumber, NSDate, NSArray, or NSDictionary).

Availability

Available in Mac OS X v10.0 and later.

See Also

- [removePersistentDomainForName:](#) (page 19)
- [setPersistentDomain:forName:](#) (page 23)

Related Sample Code

GLUT

OpenCL NBody Simulation Example

Declared In

NSUserDefaults.h

persistentDomainNames

Returns an array of the current persistent domain names.

- (NSArray *)persistentDomainNames

Return Value

An array of NSString objects containing the domain names.

Discussion

You can get the keys and values for each domain by passing the returned domain names to the [persistentDomainForName:](#) (page 17) method.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [removePersistentDomainForName:](#) (page 19)
- [setPersistentDomain:forName:](#) (page 23)

Declared In

NSUserDefaults.h

registerDefaults:

Adds the contents the specified dictionary to the registration domain.

- (void)registerDefaults:(NSDictionary *)*dictionary*

Parameters*dictionary*

The dictionary of keys and values you want to register.

Discussion

If there is no registration domain, one is created using the specified dictionary, and `NSRegistrationDomain` is added to the end of the search list.

The contents of the registration domain are not written to disk; you need to call this method each time your application starts. You can place a plist file in the application's Resources directory and call `registerDefaults:` with the contents that you read in from that file.

Availability

Available in Mac OS X v10.0 and later.

Related Sample Code

DeskPictAppDockMenu

Dicey

Sproing

TemperatureConverter

TemperatureTester

Declared In

NSUserDefaults.h

removeObjectForKey:

Removes the value of the specified default key in the standard application domain.

```
- (void)removeObjectForKey:(NSString *)defaultName
```

Parameters

defaultName

The key whose value you want to remove.

Discussion

Removing a default has no effect on the value returned by the [objectForKey:](#) (page 15) method if the same key exists in a domain that precedes the standard application domain in the search list.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setObject:forKey:](#) (page 22)

Related Sample Code

Quartz Composer WWDC 2005 TextEdit

UserDefaults

Declared In

NSUserDefaults.h

removePersistentDomainForName:

Removes the contents of the specified persistent domain from the user's defaults.

```
- (void)removePersistentDomainForName:(NSString *)domainName
```

Parameters

domainName

The domain whose keys and values you want. This value should be equal to your application's bundle identifier.

Discussion

When a persistent domain is changed, an [NSUserDefaultsDidChangeNotification](#) (page 36) is posted.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setPersistentDomain:forName:](#) (page 23)

Related Sample Code

OpenCL NBody Simulation Example

Declared In

NSUserDefaults.h

removeSuiteNamed:

Removes the specified domain name from the receiver's search list.

```
- (void)removeSuiteNamed:(NSString *)suiteName
```

Parameters

suiteName

The domain name to remove.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [addSuiteNamed:](#) (page 10)

Declared In

NSUserDefaults.h

removeVolatileDomainForName:

Removes the specified volatile domain from the user's defaults.

```
- (void)removeVolatileDomainForName:(NSString *)domainName
```

Parameters

domainName

The volatile domain you want to remove.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setVolatileDomain:forName:](#) (page 24)

Declared In

NSUserDefaults.h

setBool:forKey:

Sets the value of the specified default key to the specified Boolean value.

```
- (void)setBool:(BOOL) value forKey:(NSString *) defaultName
```

Parameters

value

The Boolean value to store in the defaults database.

defaultName

The key with which to associate with the value.

Discussion

Invokes [setObject:forKey:](#) (page 22) as part of its implementation.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [boolForKey:](#) (page 11)

Related Sample Code

Sproing

Declared In

NSUserDefaults.h

setDouble:forKey:

Sets the value of the specified default key to the double value.

```
- (void)setDouble:(double) value forKey:(NSString *) defaultName
```

Parameters

value

The double value.

defaultName

The key with which to associate with the value.

Availability

Available in Mac OS X v10.5 and later.

Declared In

NSUserDefaults.h

setFloat:forKey:

Sets the value of the specified default key to the specified floating-point value.

```
- (void)setFloat:(float) value forKey:(NSString *) defaultName
```

Parameters*value*

The floating-point value to store in the defaults database.

defaultName

The key with which to associate with the value.

DiscussionInvokes [setObject:forKey:](#) (page 22) as part of its implementation.**Availability**

Available in Mac OS X v10.0 and later.

See Also- [floatForKey:](#) (page 13)**Declared In**

NSUserDefaults.h

setInteger:forKey:

Sets the value of the specified default key to the specified integer value.

- (void)setInteger:(NSInteger)*value* forKey:(NSString *)*defaultName***Parameters***value*

The integer value to store in the defaults database.

defaultName

The key with which to associate with the value.

DiscussionInvokes [setObject:forKey:](#) (page 22) as part of its implementation.**Availability**

Available in Mac OS X v10.0 and later.

See Also- [integerForKey:](#) (page 15)**Related Sample Code**

IKSlideshowDemo

Sproing

Declared In

NSUserDefaults.h

setObject:forKey:

Sets the value of the specified default key in the standard application domain.

- (void)setObject:(id)*value* forKey:(NSString *)*defaultName*

Parameters*value*

The object to store in the defaults database. A default's value can be only property list objects: NSData, NSString, NSNumber, NSDate, NSArray, or NSDictionary.

defaultName

The key with which to associate with the value.

Discussion

Setting a default has no effect on the value returned by the [objectForKey:](#) (page 15) method if the same key exists in a domain that precedes the application domain in the search list.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [removeObjectForKey:](#) (page 19)

Related Sample Code

CIColorTracking

PrefsPane

QTAudioExtractionPanel

Quartz Composer QCTV

UserDefaults

Declared In

NSUserDefaults.h

setPersistentDomain:forName:

Sets the dictionary for the specified persistent domain.

```
- (void)setPersistentDomain:(NSDictionary *)domain forName:(NSString *)domainName
```

Parameters*domain*

The dictionary of keys and values you want to assign to the domain.

domainName

The domain whose keys and values you want to set. This value should be equal to your application's bundle identifier.

Discussion

When a persistent domain is changed, an [NSUserDefaultsDidChangeNotification](#) (page 36) is posted.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [persistentDomainForName:](#) (page 17)

- [persistentDomainNames](#) (page 18)

Related Sample Code

GLUT

OpenCL NBody Simulation Example

Declared In

NSUserDefaults.h

setURL:forKey:

Sets the value of the specified default key to the specified URL.

```
- (void)setURL:(NSURL *)url forKey:(NSString *)defaultName
```

Parameters*url*

The NSURL to store in the defaults database.

defaultName

The key with which to associate with the value.

DiscussionWhen an NSURL is stored using `-[NSUserDefaults setURL:forKey:]`, some adjustments are made:

1. Any non-file URL is written by calling `+[NSKeyedArchiver archivedDataWithRootObject:]` using the NSURL instance as the root object.
2. Any file reference `file:scheme` URL will be treated as a non-file URL, and information which makes this URL compatible with 10.5 systems will also be written as part of the archive as well as its minimal bookmark data.
3. Any path-based `file:scheme` URL is written by first taking the absolute URL, getting the path from that and then determining if the path can be made relative to the user's home directory. If it can, the string is abbreviated by using `stringByAbbreviatingWithTildeInPath` and written out. This allows pre-10.6 clients to read the default and use `-[NSString stringByExpandingTildeInPath]` to use this information.

Availability

Available in Mac OS X v10.6 and later.

See Also- [URLForKey:](#) (page 27)**Declared In**

NSUserDefaults.h

setVolatileDomain:forName:

Sets the dictionary for the specified volatile domain.

```
- (void)setVolatileDomain:(NSDictionary *)domain forName:(NSString *)domainName
```

Parameters*domain*

The dictionary of keys and values you want to assign to the domain.

domainName

The domain whose keys and values you want to set.

Discussion

This method raises an `NSInvalidArgumentException` if a volatile domain with the specified name already exists.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [volatileDomainForName:](#) (page 27)
- [volatileDomainNames](#) (page 28)

Declared In

`NSUserDefaults.h`

stringArrayForKey:

Returns the array of strings associated with the specified key.

```
- (NSArray *)stringArrayForKey:(NSString *)defaultName
```

Parameters

defaultName

A key in the current user's defaults database.

Return Value

The array of `NSString` objects, or `nil` if the specified default does not exist, the default does not contain an array, or the array does not contain `NSString` objects.

Special Considerations

The returned array and its contents are immutable, even if the values you originally set were mutable.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setObject:forKey:](#) (page 22)

Declared In

`NSUserDefaults.h`

stringForKey:

Returns the string associated with the specified key.

```
- (NSString *)stringForKey:(NSString *)defaultName
```

Parameters

defaultName

A key in the current user's defaults database.

Return Value

The string associated with the specified key, or `nil` if the default does not exist or does not contain a string.

Special Considerations

The returned string is immutable, even if the value you originally set was a mutable string.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [setObject:forKey:](#) (page 22)

Related Sample Code

CIColorTracking

CIVideoDemoGL

Core Animation QuickTime Layer

DeskPictAppDockMenu

UserDefaults

Declared In

NSUserDefaults.h

synchronize

Writes any modifications to the persistent domains to disk and updates all unmodified persistent domains to what is on disk.

- (BOOL)synchronize

Return Value

YES if the data was saved successfully to disk, otherwise NO.

Discussion

Because this method is automatically invoked at periodic intervals, use this method only if you cannot wait for the automatic synchronization (for example, if your application is about to exit) or if you want to update the user defaults to what is on disk even though you have not made any changes.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [persistentDomainForName:](#) (page 17)

- [persistentDomainNames](#) (page 18)

- [removePersistentDomainForName:](#) (page 19)

- [setPersistentDomain:forName:](#) (page 23)

Related Sample Code

CIColorTracking

CIVideoDemoGL

QTAudioExtractionPanel

QTKitPlayer

Quartz Composer QCTV

Declared In

NSUserDefaults.h

URLForKey:

Returns the NSURL instance associated with the specified key.

- (NSURL *)URLForKey:(NSString *)*defaultName***Parameters***defaultName*

A key in the current user's defaults database.

Return Value

The NSURL instance value associated with the specified key. If the key does not exist, this method returns nil.

DiscussionWhen an NSURL is read using `-[NSUserDefaults URLForKey:]`, the following logic is used:

1. If the value for the key is an NSData, the NSData is used as the argument to `+[NSKeyedUnarchiver unarchiveObjectWithData:]`. If the NSData can be unarchived as an NSURL, the NSURL is returned otherwise nil is returned.
2. If the value for this key was a file reference URL, the file reference URL will be created but its bookmark data will not be resolved until the NSURL instance is later used (e.g. at `-[NSData initWithContentsOfURL:]`).
3. If the value for the key is an NSString which begins with a `~`, the NSString will be expanded using `-[NSString stringByExpandingTildeInPath]` and a file:scheme NSURL will be created from that.

Availability

Available in Mac OS X v10.6 and later.

See Also- [setURL:forKey:](#) (page 24)**Declared In**

NSUserDefaults.h

volatileDomainForName:

Returns the dictionary for the specified volatile domain.

- (NSDictionary *)volatileDomainForName:(NSString *)*domainName***Parameters***domainName*

The domain whose keys and values you want.

Return Value

The dictionary of keys and values belonging to the domain. The keys in the dictionary are names of defaults, and the value corresponding to each key is a property list object (NSData, NSString, NSNumber, NSDate, NSArray, or NSDictionary).

Availability

Available in Mac OS X v10.0 and later.

See Also

- [removeVolatileDomainForName:](#) (page 20)
- [setVolatileDomain:forName:](#) (page 24)

Declared In

NSUserDefaults.h

volatileDomainNames

Returns an array of the current volatile domain names.

- (NSArray *)volatileDomainNames

Return Value

An array of NSString objects with the volatile domain names.

Discussion

You can get the contents of each domain by passing the returned domain names to the [volatileDomainForName:](#) (page 27) method.

Availability

Available in Mac OS X v10.0 and later.

See Also

- [removeVolatileDomainForName:](#) (page 20)
- [setVolatileDomain:forName:](#) (page 24)

Declared In

NSUserDefaults.h

Constants

NSUserDefaults Domains

These constants specify various user defaults domains.

```
extern NSString *NSGlobalDomain;
extern NSString *NSArgumentDomain;
extern NSString *NSRegistrationDomain;
```

Constants

NSGlobalDomain

The domain consisting of defaults meant to be seen by all applications.

Available in Mac OS X v10.0 and later.

Declared in `NSUserDefaults.h`.

NSArgumentDomain

The domain consisting of defaults parsed from the application's arguments. These are one or more pairs of the form *-default value* included in the command-line invocation of the application.

Available in Mac OS X v10.0 and later.

Declared in `NSUserDefaults.h`.

NSRegistrationDomain

The domain consisting of a set of temporary defaults whose values can be set by the application to ensure that searches will always be successful.

Available in Mac OS X v10.0 and later.

Declared in `NSUserDefaults.h`.

Declared In

NSUserDefaults.h

Language-Dependent Date/Time Information

The `NSUserDefaults` class provides the following constants as a convenience. They provide access to values of the keys to the locale dictionary, which is discussed in *User Defaults Programming Topics for Cocoa*.

(Deprecated.) These constants are deprecated in Mac OS X v10.5. Where there are direct replacements, you can find typically them in `NSDateFormatter`—for example, `monthSymbols`, `shortWeekdaySymbols`, and `AMSymbol`—otherwise you should use the patterns described in *Data Formatting Programming Guide for Cocoa*.)

```

extern NSString *NSAMPMDesignation;
extern NSString *NSDateFormatString;
extern NSString *NSDateTimeOrdering;
extern NSString *NSEarlierTimeDesignations;
extern NSString *NSHourNameDesignations;
extern NSString *NSLaterTimeDesignations;
extern NSString *NSMonthNameArray;
extern NSString *NSNextDayDesignations;
extern NSString *NSNextNextDayDesignations;
extern NSString *NSPriorDayDesignations;
extern NSString *NSShortDateFormatString;
extern NSString *NSShortMonthNameArray;
extern NSString *NSShortTimeDateFormatString;
extern NSString *NSShortWeekDayNameArray;
extern NSString *NSThisDayDesignations;
extern NSString *NSTimeDateFormatString;
extern NSString *NSTimeFormatString;
extern NSString *NSWeekDayNameArray;
extern NSString *NSYearMonthWeekDesignations;

```

Constants

`NSAMPMDesignation`

Key for the value that specifies how the morning and afternoon designations are printed, affecting strings that use the %p format specifier. (**Deprecated.** Use `AMSymbol` or `PMsymbol` (`NSDateFormatter`) instead.)

The defaults are “AM” and “PM”.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSDateFormatString`

Key for the format string that specifies how how dates are printed using the date format specifiers. (**Deprecated.** Use the appropriate API from `NSDateFormatter` instead—see *Data Formatting Programming Guide for Cocoa*.)

The default is to use weekday names with full month names and full years, as in “Saturday, March 24, 2001.”

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSDateTimeOrdering`

Key for the string that specifies how to use ambiguous numbers in date strings.

Specify this value as a permutation of the letters M (month), D (day), Y (year), and H (hour). For example, MDYH treats “2/3/01 10” as the 3rd day of February 2001 at 10:00 am, whereas DMYH treats the same value as the 2nd day of March 2001 at 10:00 am. If fewer numbers are specified than are needed, the numbers are prioritized to satisfy day first, then month, and then year. For example, if you supply only the value 12, it means the 12th day of this month in this year because the day must be specified. If you supply “2 12” it means either February 12 or December 2, depending on if the ordering is “MDYH” or “DMYH.”

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSEarlierTimeDesignations`

Key for an array of strings that denote a time in the past. (**Deprecated**. There is no direct replacement. If you need to localize words such as “prior,” you should use a strings file as you would for any other localizable text—see Strings Files.)

These are adjectives that modify values from `NSYearMonthWeekDesignations`. The defaults are “prior,” “last,” “past,” and “ago.”

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSHourNameDesignations`

Key for strings that identify the time of day.

These strings should be bound to an hour. The default is this array of arrays: (0, midnight), (10, morning), (12, noon, lunch), (14, afternoon), (19, dinner).

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSLaterTimeDesignations`

Key for an array of strings that denote a time in the future. (**Deprecated**. There is no direct replacement. If you need to localize words such as “next,” you should use a strings file as you would for any other localizable text—see Strings Files.)

Strings in this array are adjectives that modify a value from `NSYearMonthWeekDesignations`.

The default is an array that contains a single string, “next”.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSMonthNameArray

Key for the value that specifies the names for the months, affecting strings that use the %B format specifier. (**Deprecated.** Use `monthSymbols` or—if you are going to display these in the user interface by themselves—`standaloneMonthSymbols` (`NSDateFormatter`) instead.)

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSNextDayDesignations

Key for an array of strings that denote the day after today. (**Deprecated.** There is no direct replacement. If you need to localize words such as “tomorrow,” you should use a strings file as you would for any other localizable text—see *Strings Files*.)

The default is an array that contains a single string, “tomorrow”.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSNextNextDayDesignations

Key for an array of strings that denote the day after tomorrow. (**Deprecated.** There is no direct replacement. If you need to localize words such as “nextday,” you should use a strings file as you would for any other localizable text—see *Strings Files*.)

The default is an array that contains a single string, “nextday”.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSPriorDayDesignations

Key for an array of strings that denote the day before today. (**Deprecated.** There is no direct replacement. If you need to localize words such as “yesterday,” you should use a strings file as you would for any other localizable text—see *Strings Files*.)

The default is an array that contains a single string, “yesterday”.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSShortDateFormatString

Key for a format string that specifies how dates are abbreviated. (**Deprecated.** Use the appropriate API from `NSDateFormatter` instead—see *Data Formatting Programming Guide for Cocoa*.)

The default is to separate the day month and year with slashes and to put the day first, as in 31/10/01.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSShortWeekDayNameArray

Key for an array of strings that specify the abbreviations for the days of the week, affecting strings that use the `%a` format specifier. (**Deprecated.** Use `shortWeekdaySymbols` or—if you are going to display these in the user interface by themselves—`shortStandardWeekdaySymbols` (`NSDateFormatter`) instead.)

Sunday should be the first day of the week.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSShortMonthNameArray

Key for an array of strings that specify the abbreviations for the months, affecting strings that use the `%b` format specifier. (**Deprecated.** Use `shortMonthSymbols` or—if you are going to display these in the user interface by themselves—`shortStandardMonthSymbols` (`NSDateFormatter`) instead.)

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSShortTimeDateFormatString

Key for a format string that specifies how times and dates are abbreviated. (**Deprecated.** Use the appropriate API from `NSDateFormatter` instead—see *Data Formatting Programming Guide for Cocoa*.)

The default is to use dashes to separate the day, month, and year and to use a 12-hour clock, as in "31-Jan-01 1:30 PM.]"

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSThisDayDesignations

Key for an array of strings that specify what this day is called. (**Deprecated.** There is no direct replacement. If you need to localize words such as "today," you should use a strings file as you would for any other localizable text—see *Strings Files*.)

The default is an array containing two strings, "today" and "now".

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSTimeIntervalFormatString

Key for the value that specifies how dates with times are printed, affecting strings that use the format specifiers `%c`, `%X`, or `%x`. (**Deprecated.** Use the appropriate API from `NSDateFormatter` instead—see *Data Formatting Programming Guide for Cocoa*.)

The default is to use full month names and days with a 24-hour clock, as in "Sunday, January 01, 2001 23:00:00 Pacific Standard Time."

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSTimeInterval

Key for a format string that specifies how dates with times are printed. (**Deprecated.** Use the appropriate API from `NSDateFormatter` instead—see *Data Formatting Programming Guide for Cocoa*.)

The default is to use a 12-hour clock.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSWeekDayNameArray

Key for an array of strings that specify the names for the days of the week, affecting strings that use the `%A` format specifier. (**Deprecated.** Use `weekdaySymbols` or—if you are going to display these in the user interface by themselves—`standaloneWeekdaySymbols` (`NSDateFormatter`) instead.)

Sunday should be the first day of the week.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSYearMonthWeekDesignations

Key for an array of strings that specify the words for year, month, and week in the current locale. (**Deprecated.** There is no direct replacement. If you need to localize words such as “year,” you should use a strings file as you would for any other localizable text—see *Strings Files*.)

The defaults are “year,” “month,” and “week.”

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

Declared In

`NSUserDefaults.h`

Language-Dependent Numeric Information

The `NSUserDefaults` class provides the following constants as a convenience. They provide access to values of the keys to the locale dictionary, which is discussed in *User Defaults Programming Topics for Cocoa*. (**Deprecated.** These constants are deprecated in Mac OS X v10.5. Where there are replacements, you can typically find them in `NSNumberFormatter` or `NSLocale`—for example, `currencySymbol`, `currencyDecimalSeparator`, and `thousandSeparator`—otherwise you should use the patterns described in *Data Formatting Programming Guide for Cocoa*.)

```
extern NSString *NSCurrencySymbol;
extern NSString *NSDecimalDigits;
extern NSString *NSDecimalSeparator;
extern NSString *NSInternationalCurrencyString;
extern NSString *NSNegativeCurrencyFormatString;
extern NSString *NSPositiveCurrencyFormatString;
extern NSString *NSThousandsSeparator;
```

Constants

NSCurrencySymbol

A string that specifies the symbol used to denote currency in this language. (**Deprecated.** Use `currencySymbol` (`NSNumberFormatter`) or retrieve the `NSLocaleCurrencySymbol` from the current locale instead.)

The default is “\$”.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSDecimalDigits

Strings that identify the decimal digits in addition to or instead of the ASCII digits.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSDecimalSeparator

A string that specifies the decimal separator. (**Deprecated.** Use `decimalSeparator` or `currencyDecimalSeparator` (`NSNumberFormatter`) or retrieve the `NSLocaleDecimalSeparator` from the current locale instead.)

The decimal separator separates the ones place from the tenths place. The default is “.”.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

NSInternationalCurrencyString

A string containing a three-letter abbreviation for currency, following the ISO 4217 standard. (**Deprecated.** Retrieve the `NSLocaleCurrencySymbol` from the current locale instead.)

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSNegativeCurrencyFormatString`

A format string that specifies how negative numbers are printed when representing a currency value. **(Deprecated.** Use the appropriate API from `NSNumberFormatter` instead—see *Data Formatting Programming Guide for Cocoa*.)

The default is `-$9,999.00`.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSPositiveCurrencyFormatString`

A format string that specifies how positive numbers are printed when representing a currency value. **(Deprecated.** Use the appropriate API from `NSNumberFormatter` instead—see *Data Formatting Programming Guide for Cocoa*.)

The default is `$9,999.00`.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

`NSThousandsSeparator`

A string that specifies the separator character for the thousands place of a decimal number. **(Deprecated.** Retrieve the `NSLocaleGroupingSeparator` from the current locale instead.)

The default is a comma.

Available in Mac OS X v10.0 and later.

Deprecated in Mac OS X v10.5.

Declared in `NSUserDefaults.h`.

Declared In

`NSUserDefaults.h`

Notifications

NSUserDefaultsDidChangeNotification

This notification is posted when a change is made to defaults in a persistent domain.

The notification object is the `NSUserDefaults` object. This notification does not contain a *userInfo* dictionary.

Availability

Available in Mac OS X v10.0 and later.

Declared In

`NSUserDefaults.h`

Document Revision History

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

Date	Notes
2009-08-28	Added to the <code>setURL:</code> and <code>URLForKey:</code> descriptions.
2009-03-27	Updated for Mac OS X v10.6
2009-04-08	Updated property list storage details.
2007-12-11	Corrected discussion of notification posting behavior.
2007-10-31	Clarified the migration options for language-dependent numeric and date and time symbols.
2006-08-29	Made bug fixes.
	Clarified the description of <code>NSArgumentDomain</code> (page 29).
	Expanded the description of <code>registerDefaults:</code> (page 18).
2006-05-23	First publication of this content as a separate document.

REVISION HISTORY

Document Revision History

Index

A

`addSuiteNamed:` instance method [10](#)
`arrayForKey:` instance method [10](#)

B

`boolForKey:` instance method [11](#)

D

`dataForKey:` instance method [11](#)
`dictionaryForKey:` instance method [12](#)
`dictionaryRepresentation` instance method [13](#)
`doubleForKey:` instance method [13](#)

F

`floatForKey:` instance method [13](#)

I

`init` instance method [14](#)
`initWithUser:` instance method [14](#)
`integerForKey:` instance method [15](#)

L

Language-Dependent Date/Time Information [29](#)
Language-Dependent Numeric Information [34](#)

N

`NSAMPMDesignation` constant [30](#)
`NSArgumentDomain` constant [29](#)
`NSCurrencySymbol` constant [35](#)
`NSDateFormatString` constant [30](#)
`NSDateTimeOrdering` constant [31](#)
`NSDecimalDigits` constant (Deprecated in Mac OS X v10.5) [35](#)
`NSDecimalSeparator` constant [35](#)
`NSEarlierTimeDesignations` constant [31](#)
`NSGlobalDomain` constant [29](#)
`NSHourNameDesignations` constant [31](#)
`NSInternationalCurrencyString` constant (Deprecated in Mac OS X v10.5) [35](#)
`NSLaterTimeDesignations` constant [31](#)
`NSMonthNameArray` constant (Deprecated in Mac OS X v10.5) [32](#)
`NSNegativeCurrencyFormatString` constant [36](#)
`NSNextDayDesignations` constant [32](#)
`NSNextNextDayDesignations` constant [32](#)
`NSPositiveCurrencyFormatString` constant [36](#)
`NSPriorDayDesignations` constant [32](#)
`NSRegistrationDomain` constant [29](#)
`NSShortDateFormatString` constant [32](#)
`NSShortMonthNameArray` constant (Deprecated in Mac OS X v10.5) [33](#)
`NSShortTimeDateFormatString` constant [33](#)
`NSShortWeekDayNameArray` constant [33](#)
`NSThisDayDesignations` constant [33](#)
`NSThousandsSeparator` constant [36](#)
`NSTimeDateFormatString` constant [33](#)
`NSTimeFormatString` constant [34](#)
`NSUserDefaults Domains` [28](#)
`NSUserDefaultsDidChangeNotification` notification [36](#)
`NSWeekDayNameArray` constant [34](#)
`NSYearMonthWeekDesignations` constant [34](#)

O

`objectForKey:` [instance method 15](#)
`objectIsForcedForKey:` [instance method 16](#)
`objectIsForcedForKey:inDomain:` [instance method 17](#)

P

`persistentDomainForName:` [instance method 17](#)
`persistentDomainNames` [instance method 18](#)

R

`registerDefaults:` [instance method 18](#)
`removeObjectForKey:` [instance method 19](#)
`removePersistentDomainForName:` [instance method 19](#)
`removeSuiteNamed:` [instance method 20](#)
`removeVolatileDomainForName:` [instance method 20](#)
`resetStandardUserDefaults` [class method 9](#)

S

`setBool:forKey:` [instance method 21](#)
`setDouble:forKey:` [instance method 21](#)
`setFloat:forKey:` [instance method 21](#)
`setInteger:forKey:` [instance method 22](#)
`setObject:forKey:` [instance method 22](#)
`setPersistentDomain:forName:` [instance method 23](#)
`setURL:forKey:` [instance method 24](#)
`setVolatileDomain:forName:` [instance method 24](#)
`standardUserDefaults` [class method 9](#)
`stringArrayForKey:` [instance method 25](#)
`stringForKey:` [instance method 25](#)
`synchronize` [instance method 26](#)

U

`URLForKey:` [instance method 27](#)

V

`volatileDomainForName:` [instance method 27](#)
`volatileDomainNames` [instance method 28](#)