

---

# Collaboration Framework Reference

[Networking](#) > Cocoa



2007-05-14



Apple Inc.  
© 2007 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

.Mac is a registered service mark of Apple Inc.

Apple, the Apple logo, Cocoa, Mac, Mac OS, and Objective-C are trademarks of Apple Inc., registered in the United States and other countries.

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

**Introduction**      **Introduction** 5

---

**Part I**              **Classes** 7

---

**Chapter 1**          **CBGroupIdentity Class Reference** 9

---

Overview 9  
Tasks 9  
Class Methods 10  
Instance Methods 10

**Chapter 2**          **CBIdentity Class Reference** 13

---

Overview 13  
Tasks 13  
Class Methods 14  
Instance Methods 16

**Chapter 3**          **CBIdentityAuthority Class Reference** 21

---

Overview 21  
Tasks 21  
Class Methods 22  
Instance Methods 23

**Chapter 4**          **CBIdentityPicker Class Reference** 25

---

Overview 25  
Tasks 25  
Instance Methods 26

**Chapter 5**          **CBUserIdentity Class Reference** 29

---

Overview 29  
Tasks 29  
Class Methods 30  
Instance Methods 30

**Document Revision History 33**

---

**Index 35**

---

# Introduction

---

<b>Framework</b>	/System/Library/Frameworks/Collaboration.framework
<b>Header file directories</b>	/System/Library/Frameworks/Collaboration.framework/Headers
<b>Declared in</b>	CBIdentity.h CBIdentityAuthority.h CBIdentityPicker.h
<b>Companion guides</b>	Identity Services Programming Guide Core Services Identity Reference

The Collaboration framework is a set of Objective-C classes that allows developers to monitor identities and their attributes. Identities reside in an identity authority, which can be either local to a user's system, or on a network directory. The Collaboration framework also manages a sheet, known as the identity picker, to allow applications to select identities.

The Collaboration framework works closely with the Core Services Identity APIs to form the Identity Services technology. If you need the ability to create and manipulate identities read *Core Services Identity Reference*.



# Classes

---





# CBGroupIdentity Class Reference

---

<b>Inherits from</b>	CBIdentity : NSObject
<b>Conforms to</b>	NSCoding (CBIdentity) NSCopying (CBIdentity) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Collaboration.framework
<b>Availability</b>	Available in Mac OS X v10.5 and later.
<b>Declared in</b>	Collaboration/CBIdentity.h
<b>Companion guide</b>	Identity Services Programming Guide

## Overview

An object of the `CBGroupIdentity` class represents a group identity and is used for viewing the attributes of group identities from an identity authority. The principal attributes of a `CBGroupIdentity` object are a POSIX group identifier (GID) and a list of members.

## Tasks

### Finding Group Identities

+ [groupIdentityWithPosixGID:authority:](#) (page 10)

Returns the group identity with the given POSIX GID in the specified identity authority.

### Group Identity Attributes

- [posixGID](#) (page 11)

Returns the POSIX GID of the identity.

- [members](#) (page 10)

Returns the members of the group.

## Class Methods

### **groupIdentityWithPosixGID:authority:**

Returns the group identity with the given POSIX GID in the specified identity authority.

```
+ (CBGroupIdentity *)groupIdentityWithPosixGID:(gid_t)gid
  authority:(CBIdentityAuthority *)authority
```

#### **Parameters**

*gid*

The GID of the group identity you are searching for.

*authority*

An identity authority in which to search for the group identity.

#### **Return Value**

The group identity object with the given GID in the specified identity authority, or `nil` if no identity exists with the specified GID.

#### **Availability**

Available in Mac OS X v10.5 and later.

#### **Declared In**

`CBIdentity.h`

## Instance Methods

### **members**

Returns the members of the group.

```
- (NSArray *)members
```

#### **Return Value**

An array of `CBIdentity` objects each representing a member of the group identity.

#### **Discussion**

This method only returns direct members of a group, it does not return members of members. Both user and group identities can be members of a group, but a group cannot be a member of itself. You also cannot have “circular” membership, i.e. a group be a member of another group that is a member of the first group.

#### **Availability**

Available in Mac OS X v10.5 and later.

#### **Declared In**

`CBIdentity.h`

## posixGID

Returns the POSIX GID of the identity.

- (gid\_t)posixGID

### Return Value

The POSIX GID of the group identity.

### Discussion

The POSIX GID is an integer that can identify a group within an identity authority. GIDs are not guaranteed to be unique within an identity authority.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentity.h



# CBIdentity Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSCoding NSCopying NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Collaboration.framework
<b>Availability</b>	Available in Mac OS X v10.5 and later.
<b>Declared in</b>	Collaboration/CBIdentity.h
<b>Companion guide</b>	Identity Services Programming Guide

## Overview

A `CBIdentity` object is used for accessing the attributes of an identity stored in an identity authority. You can use an identity object for finding identities, and storing them in an access control list (ACL). If you need to edit these attributes, take advantage of the `CSIdentity` class in Core Services.

You can obtain a `CBIdentity` object from one of the following class factory methods:

`identityWithName:authority:`, `identityWithUUIDString:authority:`,  
`identityWithPersistentReference:`, or `identityWithCSIdentity:`.

A `CBIdentity` object has methods to support for interoperability with the Core Services Identity API. Send `CSIdentity` to your `CBIdentity` object to return an opaque object for use in the Core Services Identity API. Similarly, call `identityWithCSIdentity:` to use an Core Services Identity opaque object in the Collaboration framework.

There are two subclasses of `CBIdentity`: `CBGroupIdentity` and `CBUserIdentity`. If you are working specifically with a group identity, use `CBGroupIdentity`. Similarly, if you are working with a user identity, use `CBUserIdentity`.

## Tasks

### Finding Identities

+ [identityWithCSIdentity:](#) (page 14)

Returns an identity object created from the specified Core Services Identity opaque object.

- + `identityWithName:authority:` (page 15)  
Returns the identity object with the given name from the specified identity authority.
- + `identityWithPersistentReference:` (page 15)  
Returns the identity object matching the persistent reference data.
- + `identityWithUUIDString:authority:` (page 16)  
Returns the identity object with the given UUID from the specified identity authority.

## Getting Identity Attributes

- `aliases` (page 16)  
Returns an array of aliases (alternate names) for the identity.
- `authority` (page 17)  
Returns the identity authority where the identity is stored.
- `emailAddress` (page 17)  
Returns the email address of an identity.
- `fullName` (page 17)  
Returns the full name of the identity.
- `image` (page 18)  
Returns the image associated with an identity.
- `isHidden` (page 18)  
Returns a Boolean value indicating the state of the identity's hidden property.
- `isMemberOfGroup:` (page 18)  
Returns a Boolean value indicating whether the identity is a member of the specified group.
- `posixName` (page 19)  
Returns the POSIX name of the identity.
- `UUIDString` (page 20)  
Returns the UUID of the identity as a string.

## Storing Identities

- `CSIdentity` (page 17)  
Returns an opaque object for use with the Core Services Identity API.
- `persistentReference` (page 19)  
Returns a persistent reference to store a reference to an identity.

## Class Methods

### **identityWithCSIdentity:**

Returns an identity object created from the specified Core Services Identity opaque object.

```
+ (CBIdentity *)identityWithCSIdentity:(CSIdentityRef)csIdentity
```

**Parameters***csIdentity*

The Core Services Identity opaque object.

**Return Value**

The identity object for use with the Collaboration framework.

**Discussion**

This method is used for interoperability with the Core Services Identity API.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

**identityWithName:authority:**

Returns the identity object with the given name from the specified identity authority.

```
+ (CBIdentity *)identityWithName:(NSString *)name authority:(CBIdentityAuthority
*)authority
```

**Parameters***name*

The name of the identity.

*authority*

The identity authority to search.

**Return Value**The identity object, or `nil` if no identity is found with the specified name.**Discussion**

The name is compared against all valid identity names, including full names, short names, email addresses, and aliases.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

**identityWithPersistentReference:**

Returns the identity object matching the persistent reference data.

```
+ (CBIdentity *)identityWithPersistentReference:(NSData *)data
```

**Parameters***data*

The persistent data object that refers to an identity.

**Return Value**The identity object matching the persistent data object, or `nil` if the identity is not found.

**Discussion**

A persistent reference is an opaque data object suitable for persistent storage.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

`CBIdentity.h`

**identityWithUUIDString:authority:**

Returns the identity object with the given UUID from the specified identity authority.

```
+ (CBIdentity *)identityWithUUIDString:(NSString *)uuid
  authority:(CBIdentityAuthority *)authority
```

**Parameters**

*uuid*

The UUID of the identity you are searching for.

*authority*

The identity authority to search.

**Return Value**

The identity object, or `nil` if no identity is found with the matching criteria.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

`CBIdentity.h`

## Instance Methods

**aliases**

Returns an array of aliases (alternate names) for the identity.

```
- (NSArray *)aliases
```

**Return Value**

An array of `NSString` objects containing the alternate names for the identity.

**Discussion**

An identity can have zero or more aliases. Like the full and short names, two identities cannot share an alias.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

`CBIdentity.h`



## authority

Returns the identity authority where the identity is stored.

```
- (CBIdentityAuthority *)authority
```

### Return Value

The identity authority where the identity is stored.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentity.h

## CSIdentity

Returns an opaque object for use with the Core Services Identity API.

```
- (CSIdentityRef)CSIdentity
```

### Return Value

The opaque object for use with the Core Services Identity API.

### Discussion

This method, along with `identityWithCSIdentity:`, is used for interoperability with the Core Services Identity API.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentity.h

## emailAddress

Returns the email address of an identity.

```
- (NSString *)emailAddress
```

### Return Value

The email address of an identity or `nil` if none exists.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentity.h

## fullName

Returns the full name of the identity.

- (NSString \*)fullName

**Return Value**

The full name for the identity.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

## image

Returns the image associated with an identity.

- (NSImage \*)image

**Return Value**

The image associated with an identity, or `nil` if none exists.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

## isHidden

Returns a Boolean value indicating the state of the identity's hidden property.

- (BOOL)isHidden

**Return Value**

YES if the identity is hidden; NO if it is not.

**Discussion**

A hidden identity does not show up in the Identity Picker. A hidden identity refers to system identities such as `root`, `www`, and `wheel`.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

## isMemberOfGroup:

Returns a Boolean value indicating whether the identity is a member of the specified group.

- (BOOL)isMemberOfGroup:(CBGroupIdentity \*)group

**Parameters***group*

The group to check for membership.

**Return Value**

YES if the identity is a member of the group; NO if it is not.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

**persistentReference**

Returns a persistent reference to store a reference to an identity.

- (NSData \*)persistentReference

**Return Value**

A data object that uniquely references an identity.

**Discussion**

A persistent reference data object is an object generated from an identity. Persistent data objects can be written to and read from a file, making them extremely useful for storing identities in an ACL.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

**posixName**

Returns the POSIX name of the identity.

- (NSString \*)posixName

**Return Value**

The POSIX name of the identity.

**Discussion**

The POSIX name is also referred to as the “short name” for an identity. It can only contain the characters A-Z, a-z, 0-9, -, \_, ., and @.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h

## UUIDString

Returns the UUID of the identity as a string.

```
- (NSString *)UUIDString
```

### Return Value

The UUID string of the identity.

### Discussion

The UUID string is generated so it is unique across all identity authorities. When storing ACLs, one method is to store the UUID of each identity. However, it is recommended that you use a persistent data object instead (see `persistentReference`).

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`CBIdentity.h`

# CBIdentityAuthority Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Collaboration.framework
<b>Availability</b>	Available in Mac OS X v10.5 and later.
<b>Declared in</b>	Collaboration/CBIdentityAuthority.h
<b>Companion guide</b>	Identity Services Programming Guide

## Overview

An identity authority is a database that stores information about identities. The `CBIdentityAuthority` class defines one or more identity authorities. This database can be searched for identities in conjunction with the `CBIdentity` class factory methods.

## Tasks

### Interacting with the Core Services Identity API

- [CSIdentityAuthority](#) (page 23)  
Returns an identity authority for use with the Core Services Identity API.
- + [identityAuthorityWithCSIdentityAuthority:](#) (page 22)  
Returns an identity authority specified by a given Core Services Identity authority object.

### Accessing Identity Authorities

- [localizedName](#) (page 24)  
Returns the localized name of the identity authority.
- + [localIdentityAuthority](#) (page 23)  
Returns the identity authority on the local system.
- + [managedIdentityAuthority](#) (page 23)  
Returns the identity authority that contains all the identities in bound network directory servers.

+ [defaultIdentityAuthority](#) (page 22)

Returns an identity authority that contains the identities in both the local and the network-bound authorities.

## Class Methods

### defaultIdentityAuthority

Returns an identity authority that contains the identities in both the local and the network-bound authorities.

```
+ (CBIdentityAuthority *)defaultIdentityAuthority
```

#### Return Value

The local and network-bound identity authorities.

#### Discussion

The default identity authority is the logical union of the identities in the local and managed authorities.

#### Availability

Available in Mac OS X v10.5 and later.

#### Declared In

CBIdentityAuthority.h

### identityAuthorityWithCSIdentityAuthority:

Returns an identity authority specified by a given Core Services Identity authority object.

```
+ (CBIdentityAuthority *)identityAuthorityWithCSIdentityAuthority:(CSIdentityAuthorityRef)CSIdentityAuthority
```

#### Parameters

*CSIdentityAuthority*

The Core Services Identity opaque object.

#### Return Value

The identity authority object for use with the Collaboration framework.

#### Discussion

This method, along with `CSIdentityAuthority`, is used for interoperability with the Core Services Identity API.

#### Availability

Available in Mac OS X v10.5 and later.

#### Declared In

CBIdentityAuthority.h

## localIdentityAuthority

Returns the identity authority on the local system.

```
+ (CBIdentityAuthority *)localIdentityAuthority
```

### Return Value

The identity authority on the local system.

### Discussion

Any identities stored on the local system are contained within this identity authority.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentityAuthority.h

## managedIdentityAuthority

Returns the identity authority that contains all the identities in bound network directory servers.

```
+ (CBIdentityAuthority *)managedIdentityAuthority
```

### Return Value

The identity authorities in bound network directory servers.

### Discussion

If you are bound to a network directory server (such as an LDAP server) that has an identity authority, use this method to search those authorities.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentityAuthority.h

## Instance Methods

## CSIdentityAuthority

Returns an identity authority for use with the Core Services Identity API.

```
- (CSIdentityAuthorityRef)CSIdentityAuthority
```

### Return Value

The opaque authority object for use with the Core Services Identity API.

### Discussion

This method, along with `identityAuthorityWithCSIdentityAuthority:`, is used for interoperability with the Core Services Identity API.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentityAuthority.h

**localizedName**

Returns the localized name of the identity authority.

- (NSString \*)localizedName

**Return Value**

The computer's name if the authority is local, or Managed Network Directory if the authority is managed.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentityAuthority.h



# CBIdentityPicker Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Collaboration.framework
<b>Availability</b>	Available in Mac OS X v10.5 and later.
<b>Declared in</b>	Collaboration/CBIdentityPicker.h
<b>Companion guide</b>	Identity Services Programming Guide

## Overview

A `CBIdentityPicker` object allows a user to select identities—for example, user or group objects—that it wants one or more services or shared resources to have access to. An identity picker can be displayed either as an application-modal dialog or as a sheet attached to a document window. An identity picker returns the selected records to be added to access control lists using Collaboration. If a selected record is not a user or group identity, then an identity picker prompts the end user for additional information—such as a password—to promote that record to a sharing account.

## Tasks

### Running an Identity Picker

- [runModalForWindow:modalDelegate:didEndSelector:contextInfo:](#) (page 27)  
Runs the receiver modally as a sheet attached to a specified window.
- [runModal](#) (page 27)  
Runs the receiver as an application-modal dialog.

### Retrieving Identities

- [identities](#) (page 26)  
Returns an array of the identities selected using the identity picker.

## Setting and Getting Properties

- `setTitle:` (page 28)  
Sets the title of the identity picker.
- `title` (page 28)  
Returns the title of the identity picker.
- `setAllowsMultipleSelection:` (page 28)  
Allows a user to make select multiple identities.
- `allowsMultipleSelection` (page 26)  
Returns a Boolean value indicating whether the user is allowed to select multiple identities.

## Instance Methods

### **allowsMultipleSelection**

Returns a Boolean value indicating whether the user is allowed to select multiple identities.

- (BOOL)allowsMultipleSelection

#### **Return Value**

YES if the user can select multiple records; otherwise, NO.

#### **Availability**

Available in Mac OS X v10.5 and later.

#### **Declared In**

CBIdentityPicker.h

### **identities**

Returns an array of the identities selected using the identity picker.

- (NSArray \*)identities

#### **Return Value**

An array of the selected identities.

#### **Discussion**

The array contains `CBIdentity` objects.

#### **Availability**

Available in Mac OS X v10.5 and later.

#### **Declared In**

CBIdentityPicker.h

## runModal

Runs the receiver as an application-modal dialog.

```
- (NSInteger)runModal
```

### Return Value

NSOKButton if the user selected OK; otherwise, NSCancelButton.

### Discussion

The receiver may create identities for selected records if necessary.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentityPicker.h

## runModalForWindow:modalDelegate:didEndSelector:contextInfo:

Runs the receiver modally as a sheet attached to a specified window.

```
- (void)runModalForWindow:(NSWindow *)window modalDelegate:(id)delegate
    didEndSelector:(SEL)didEndSelector contextInfo:(void *)contextInfo
```

### Parameters

*window*

The parent window for the sheet.

*delegate*

The delegate for the modal session.

*didEndSelector*

A message sent to the delegate after the user responds but before the sheet is dismissed.

*contextInfo*

Contextual data passed to the delegate in the `didEndSelector` message.

### Discussion

The `didEndSelector` parameter is a selector that takes three arguments. The corresponding method should have a declaration modeled on the following example:

```
- (void)identityPickerDidEnd:(CBIdentityPicker *)identityPicker
    identities:(NSArray *)identities contextInfo:(void *)contextInfo;
```

where the `identityPicker` argument is the identity picker object, the `identities` argument is an array containing `CBIdentity` objects, and `contextInfo` is the same `contextInfo` argument that was passed in the original message.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentityPicker.h

## setAllowsMultipleSelection:

Allows a user to make select multiple identities.

```
- (void)setAllowsMultipleSelection:(BOOL)flag
```

### Parameters

*flag*

YES if you can select multiple records; otherwise, NO.

### Discussion

By default, you cannot select multiple records.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentityPicker.h

## setTitle:

Sets the title of the identity picker.

```
- (void)setTitle:(NSString *)title
```

### Parameters

*title*

The title of the identity picker.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentityPicker.h

## title

Returns the title of the identity picker.

```
- (NSString *)title
```

### Return Value

The title of the identity picker.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

CBIdentityPicker.h

# CBUserIdentity Class Reference

---

<b>Inherits from</b>	CBIdentity : NSObject
<b>Conforms to</b>	NSCoding NSCopying NSCoding (CBIdentity) NSCopying (CBIdentity) NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/Collaboration.framework
<b>Availability</b>	Available in Mac OS X v10.5 and later.
<b>Declared in</b>	Collaboration/CBIdentity.h
<b>Companion guide</b>	Identity Services Programming Guide

## Overview

An object of the `CBUserIdentity` class represents a user identity and is used for accessing the attributes of a user identity from an identity authority. The principal attributes of `CBUserIdentity` are a POSIX user identifier (UID), password, and certificate.

## Tasks

### Password Authentication

- [authenticateWithPassword:](#) (page 30)  
Returns a Boolean value indicating whether the given password is correct for the identity.
- [certificate](#) (page 31)  
Returns the public authentication certificate associated with a user identity.
- [isEnabled](#) (page 31)  
Returns a Boolean value indicating whether the identity is allowed to authenticate.

### Using UIDs

- [posixUID](#) (page 31)  
Returns the POSIX UID of the identity.

+ [userIdentityWithPosixUID:authority:](#) (page 30)

Returns the user identity with the given POSIX UID in the specified identity authority.

## Class Methods

### **userIdentityWithPosixUID:authority:**

Returns the user identity with the given POSIX UID in the specified identity authority.

```
+ (CUserIdentity *)userIdentityWithPosixUID:(uid_t)uid
  authority:(CIdentityAuthority *)authority
```

#### **Parameters**

*uid*

The UID of the identity you are searching for.

*authority*

The identity authority to search.

#### **Return Value**

The user identity with the given UID in the specified identity authority, or `nil` if no identity exists with the specified UID.

#### **Availability**

Available in Mac OS X v10.5 and later.

#### **Declared In**

`CIdentity.h`

## Instance Methods

### **authenticateWithPassword:**

Returns a Boolean value indicating whether the given password is correct for the identity.

```
- (BOOL)authenticateWithPassword:(NSString *)password
```

#### **Parameters**

*password*

The password to test for the identity.

#### **Return Value**

TRUE if the password is correct; otherwise, FALSE.

#### **Availability**

Available in Mac OS X v10.5 and later.

#### **Declared In**

`CIdentity.h`

## certificate

Returns the public authentication certificate associated with a user identity.

- (SecCertificateRef)certificate

### Return Value

The public authentication certificate, or `nil` if none exists.

### Discussion

The Collaboration framework supports certificate-based authentication in addition to passwords. If a certificate is stored for a user identity, it will be the default method of authentication.

When a .Mac account is associated with a user identity, the authentication certificate is automatically downloaded from the .Mac servers.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`CIdentity.h`

## isEnabled

Returns a Boolean value indicating whether the identity is allowed to authenticate.

- (BOOL)isEnabled

### Return Value

TRUE if the identity can authenticate; otherwise, FALSE.

### Discussion

If the identity does not have authentication credentials (a password or certificate), it is not able to log in. However, an identity with authentication credentials does not ensure that it is enabled. Any identity can be disabled.

### Availability

Available in Mac OS X v10.5 and later.

### Declared In

`CIdentity.h`

## posixUID

Returns the POSIX UID of the identity.

- (uid\_t)posixUID

### Return Value

The POSIX UID of the identity.

### Discussion

The POSIX UID is a integer that can identify a user within an identity authority. UIDs are not guaranteed to be unique within an identity authority.

**Availability**

Available in Mac OS X v10.5 and later.

**Declared In**

CBIdentity.h



# Document Revision History

---

This table describes the changes to *Collaboration Framework Reference*.

Date	Notes
2007-05-14	New document that describes the Objective-C API for monitoring and editing identities.

## REVISION HISTORY

### Document Revision History

# Index

---

## A

---

aliases **instance method** [16](#)  
allowsMultipleSelection **instance method** [26](#)  
authenticateWithPassword: **instance method** [30](#)  
authority **instance method** [17](#)

## C

---

certificate **instance method** [31](#)  
CSIdentity **instance method** [17](#)  
CSIdentityAuthority **instance method** [23](#)

## D

---

defaultIdentityAuthority **class method** [22](#)

## E

---

emailAddress **instance method** [17](#)

## F

---

fullName **instance method** [17](#)

## G

---

groupIdentityWithPosixGID:authority: **class method** [10](#)

## I

---

identities **instance method** [26](#)  
identityAuthorityWithCSIdentityAuthority:  
    **class method** [22](#)  
identityWithCSIdentity: **class method** [14](#)  
identityWithName:authority: **class method** [15](#)  
identityWithPersistentReference: **class method**  
    [15](#)  
identityWithUUIDString:authority: **class method**  
    [16](#)  
image **instance method** [18](#)  
isEnabled **instance method** [31](#)  
isHidden **instance method** [18](#)  
isMemberOfGroup: **instance method** [18](#)

## L

---

localIdentityAuthority **class method** [23](#)  
localizedName **instance method** [24](#)

## M

---

managedIdentityAuthority **class method** [23](#)  
members **instance method** [10](#)

## P

---

persistentReference **instance method** [19](#)  
posixGID **instance method** [11](#)  
posixName **instance method** [19](#)  
posixUID **instance method** [31](#)

## R

---

runModal **instance method** [27](#)

runModalForWindow:modalDelegate:didEndSelector:  
contextInfo: **instance method** [27](#)

## S

---

setAllowsMultipleSelection: **instance method** [28](#)  
setTitle: **instance method** [28](#)

## T

---

title **instance method** [28](#)

## U

---

userIdentityWithPosixUID:authority: **class**  
**method** [30](#)  
UUIDString **instance method** [20](#)