

---

# ISyncClient Class Reference

Data Management: Syncing



2009-03-27



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, iCal, iPod, Mac, and Mac OS are trademarks of Apple Inc., registered in the United States and other countries.

MobileMe 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

## **ISyncClient Class Reference 5**

---

Overview	5
Tasks	6
Getting and Setting Attributes	6
Specifying Supported Entities	7
Getting Sync Status	7
Enabling Entities	7
Replacing Records	7
Filtering	7
Alerting Clients	8
Instance Methods	8
canPullChangesForEntityName:	8
canPushChangesForEntityName:	8
clientIdentifier	9
clientType	9
displayName	9
enabledEntityNames	10
filters	10
formatsRelationships	10
imagePath	11
isEnabledForEntityName:	11
lastSyncDateForEntityName:	12
lastSyncStatusForEntityName:	12
objectForKey:	12
setDisplayNames:	13
setEnabled:forEntityNames:	13
setFilters:	14
setFormatsRelationships:	14
setImagePath:	14
setObject:forKey:	15
setShouldReplaceClientRecords:forEntityNames:	15
setShouldSynchronize:withClientsOfType:	16
setSyncAlertHandler:selector:	17
setSyncAlertToolPath:	18
shouldReplaceClientRecordsForEntityName:	18
shouldSynchronizeWithClientsOfType:	19
supportedEntityNames	19
syncAlertToolPath	19
Constants	20
Client Types	20
ISyncStatus	20

**Document Revision History 23**

---

**Index 25**

---

# ISyncClient Class Reference

---

<b>Inherits from</b>	NSObject
<b>Conforms to</b>	NSObject (NSObject)
<b>Framework</b>	/System/Library/Frameworks/SyncServices.framework
<b>Availability</b>	Available in Mac OS X v10.4 and later.
<b>Companion guide</b>	Sync Services Programming Guide
<b>Declared in</b>	ISyncClient.h
<b>Related sample code</b>	People SeeMyFriends StickiesWithCoreData

## Overview

An `ISyncClient` object represents an application, tool, or device that syncs records—for example, Address Book, MobileMe, or a mobile phone.

An `ISyncClient` object encapsulates information that assists the sync engine in identifying your client, determining its capabilities, and maintaining its state. For example, you use an `ISyncClient` object to get the list of entities that a client supports, find out when an entity was last synced, and setup filters. `ISyncClient` also provides some methods for controlling the sync mode.

You create an `ISyncClient` object by registering a unique client identifier with the shared `ISyncManager` object. Send either the `registerClientWithIdentifier:descriptionFilePath:` or `clientWithIdentifier:` message to the shared `ISyncManager` object. You obtain the shared instance by sending `sharedManager` to `ISyncManager` class. You unregister a client, remove all information the sync engine knows about that client, using the `unregisterClient:ISyncManager` method. See *Sync Services Programming Guide* for more information on registering and unregistering clients. You should never subclass or instantiate `ISyncClient` directly.

When you create a client using the `registerClientWithIdentifier:descriptionFilePath:ISyncManager` method, you specify the client's capabilities using a client description file. Some of the `ISyncClient` methods are simply accessors that you can use to get or set the properties of this client description. For example, you use the client description to specify the entities and properties that a client supports, and you use the [supportedEntityNames](#) (page 19) method to get those supported entities. You can also use the [canPushChangesForEntityName:](#) (page 8) and [canPullChangesForEntityName:](#) (page 8) methods to find out which entities your client can push and pull. See *Sync Services Programming Guide* to learn more about the properties of a client description file.

Typically, the user requests that an application or device be resets (so that all the records on the client are replaced by the records in the truth database). The preference panel or configuration tool that receives this user request sends a [setShouldReplaceClientRecords:forEntityNames:](#) (page 15) message to the ISyncClient so that the next time the client syncs the truth is pulled. This is called a **pull the truth** sync mode and must be requested before the sync session enters the negotiation state.

Clients can optionally sync simultaneously. Use the [setShouldSynchronize:withClientsOfType:](#) (page 16) method to specify the type of client your client is interested in syncing simultaneously with. If you want to participate in a sync when your application isn't running, use the [setSyncAlertToolPath:](#) (page 18) method to specify that an alert tool be launched. Otherwise, use the [setSyncAlertHandler:selector:](#) (page 17) method to specify that a target and action be invoked when another client of the specified type syncs. If both a sync tool and sync target-action are registered, only the sync target-action is invoked.

If your application uses only a subset of the entities, attributes, and relationships defined in a schema, then you can restrict pulled records to that subset using custom filters. You set filters using the [setFilters:](#) (page 14) method. Each filter is expected to conform to the ISyncFiltering protocol and are used to reject or accept records from the sync engine before they are pulled. Use the [filters](#) (page 10) method to get the filters currently used by a client. See *Sync Services Programming Guide* for more information on using filters.

## Tasks

### Getting and Setting Attributes

- [clientIdentifier](#) (page 9)  
Returns the client's identifier specified when registering the client.
- [clientType](#) (page 9)  
Returns the receiver's client type.
- [displayName](#) (page 9)  
Returns the receiver's display name specified in the client description file when registering the client or by sending [setDisplayDisplayName:](#) (page 13) to the receiver.
- [imagePath](#) (page 11)  
Returns the absolute path to the image representation of the client.
- [setDisplayDisplayName:](#) (page 13)  
Sets the display name for the receiver to *displayName*.
- [setImagePath:](#) (page 14)  
Sets the receiver's absolute image path to *path*.
- [objectForKey:](#) (page 12)  
Returns the object for *key* that was specified using the [setObject:forKey:](#) (page 15) method.
- [setObject:forKey:](#) (page 15)  
Associates arbitrary information specified by a key-value pair to the receiver.
- [formatsRelationships](#) (page 10)  
Returns a Boolean value indicating whether the client may reformat relationships.
- [setFormatsRelationships:](#) (page 14)  
Sets whether the client may reformat relationships.

## Specifying Supported Entities

- [canPushChangesForEntityName:](#) (page 8)  
Returns YES if the client supports pushing changes to entity records specified by *entityName*, NO otherwise.
- [canPullChangesForEntityName:](#) (page 8)  
Returns YES if the client supports pulling changes to entity records specified by *entityName*, NO otherwise.
- [supportedEntityNames](#) (page 19)  
Returns an array of NSString objects containing the names of the entities the client supports.

## Getting Sync Status

- [lastSyncDateForEntityName:](#) (page 12)  
Returns the start date of the last time an entity, specified by *entityName*, was synced.
- [lastSyncStatusForEntityName:](#) (page 12)  
Returns the status of the last time an entity, specified by *entityName* was synced.

## Enabling Entities

- [enabledEntityNames](#) (page 10)  
Returns an array of NSString objects containing the names of the entities that are enabled.
- [isEnabledForEntityName:](#) (page 11)  
Returns YES if the entity specified by *entityName* is enabled, NO otherwise.
- [setEnabled:forEntityNames:](#) (page 13)  
If *flag* is YES, enables the entities specified by *entityNames*, otherwise disables them.

## Replacing Records

- [setShouldReplaceClientRecords:forEntityNames:](#) (page 15)  
Sets whether or not a client should pull the truth—replace all its records for the specified entities on the next sync.
- [shouldReplaceClientRecordsForEntityName:](#) (page 18)  
Returns YES if the client should replace all records for the entity specified by *entityName* during the next sync, NO otherwise.

## Filtering

- [filters](#) (page 10)  
Returns an array of filters that define a subset of the records the client syncs.
- [setFilters:](#) (page 14)  
Sets the receiver's filters used to control the records pulled from the sync engine to *filters*, an array of objects conforming to the ISyncFiltering protocol.

## Alerting Clients

- [shouldSynchronizeWithClientsOfType:](#) (page 19)  
Returns YES if the client is registered to receive alerts when clients of *clientType* sync, NO otherwise.
- [setShouldSynchronize:withClientsOfType:](#) (page 16)  
Adds the receiver as an observer of alerts when clients of the specified type sync.
- [setSyncAlertToolPath:](#) (page 18)  
Specifies the absolute path to a tool that is launched when an observed client creates a session and begins syncing.
- [setSyncAlertHandler:selector:](#) (page 17)  
Sets the target and action to be invoked when an observed client creates a session and begins syncing.
- [syncAlertToolPath](#) (page 19)  
Returns the path to the tool that is launched when an observed client begins syncing.

## Instance Methods

### canPullChangesForEntityName:

Returns YES if the client supports pulling changes to entity records specified by *entityName*, NO otherwise.

```
- (BOOL)canPullChangesForEntityName:(NSString *)entityName
```

#### Discussion

Use this method to determine if a client is capable of pulling entity records. For example, an iPod or phone client might pull but never push changes to contacts and calendars. This property is set when registering the client.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- [canPushChangesForEntityName:](#) (page 8) (ISyncManager)
- [registerClientWithIdentifier:descriptionFilePath:](#)

#### Declared In

ISyncClient.h

### canPushChangesForEntityName:

Returns YES if the client supports pushing changes to entity records specified by *entityName*, NO otherwise.

```
- (BOOL)canPushChangesForEntityName:(NSString *)entityName
```

#### Discussion

Use this method to determine if a client is capable of pushing entity records. For example, an iPod or phone client might pull but never push changes to contacts and calendars. This property is set when registering the client.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [canPullChangesForEntityName:](#) (page 8) (ISyncManager)
- `registerClientWithIdentifier:descriptionFilePath:`

### Declared In

ISyncClient.h

## clientIdentifier

Returns the client's identifier specified when registering the client.

- (NSString \*)clientIdentifier

### Discussion

You set the client identifier when registering the client using the `registerClientWithIdentifier:descriptionFilePath:ISyncManager` method. The client identifier is expected to be unique across all clients and is typically a DNS-style name.

### Availability

Available in Mac OS X v10.4 and later.

### Declared In

ISyncClient.h

## clientType

Returns the receiver's client type.

- (NSString \*)clientType

### Discussion

The returned string is expected to be one of the constants described in "[Constants](#)" (page 20). The client type is used to match clients that want to sync simultaneously. You specify the client type in the client description file when registering the client using the `registerClientWithIdentifier:descriptionFilePath:ISyncManager` method.

### Availability

Available in Mac OS X v10.4 and later.

### Declared In

ISyncClient.h

## displayName

Returns the receiver's display name specified in the client description file when registering the client or by sending `setDisplayDisplayName:` (page 13) to the receiver.

- (NSString \*)displayName

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [registerClientWithIdentifier:descriptionFilePath:](#) (ISyncManager)

### Declared In

ISyncClient.h

## enabledEntityNames

Returns an array of NSString objects containing the names of the entities that are enabled.

- (NSArray \*)enabledEntityNames

### Discussion

The enabled entities may be a subset of the supported entities. Use [setEnabled:forEntityNames:](#) (page 13) to enable or disable an entity. You should pass the returned array as the *entityNames* argument to one of the [beginSessionWithClient...](#) ISyncSession class methods when creating a session.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

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

### Declared In

ISyncClient.h

## filters

Returns an array of filters that define a subset of the records the client syncs.

- (NSArray \*)filters

### Discussion

Objects in the returned array are expected to conform to the ISyncFiltering protocol.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setFilters:](#) (page 14)

### Declared In

ISyncClient.h

## formatsRelationships

Returns a Boolean value indicating whether the client may reformat relationships.

- (BOOL)formatsRelationships

**Return Value**

YES if the client reformats relationships; otherwise, NO.

**Discussion**

If the client never reformats the destination records of pulled relationships, then the sync engine can perform some optimizations on behalf of the client. The default value is NO.

**Availability**

Available in Mac OS X v10.6 and later.

**See Also**

- [setFormatsRelationships:](#) (page 14)

**Declared In**

ISyncClient.h

**imagePath**

Returns the absolute path to the image representation of the client.

```
- (NSString *)imagePath
```

**Discussion**

You can specify an image path in the client description file when registering a client or by sending [setImagePath:](#) (page 14) to the receiver.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [registerClientWithIdentifier:descriptionFilePath:](#) (ISyncManager)

**Declared In**

ISyncClient.h

**isEnabledForEntityName:**

Returns YES if the entity specified by *entityName* is enabled, NO otherwise.

```
- (BOOL)isEnabledForEntityName:(NSString *)entityName
```

**Discussion**

If this method returns NO, the sync engine does not allow the client to sync records of type *entityName*.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [enabledEntityNames](#) (page 10)

- [setEnabled:forEntityNames:](#) (page 13)

**Declared In**

ISyncClient.h

## lastSyncDateForEntityName:

Returns the start date of the last time an entity, specified by *entityName*, was synced.

```
- (NSDate *)lastSyncDateForEntityName:(NSString *)entityName
```

### Discussion

Returns a start date of the last sync even if the last sync failed. Returns the start date of the previous sync if the client is currently syncing the entity. Returns `nil` if the client never synced the specified entity or the entity is not supported.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [lastSyncStatusForEntityName:](#) (page 12)

### Declared In

ISyncClient.h

## lastSyncStatusForEntityName:

Returns the status of the last time an entity, specified by *entityName* was synced.

```
- (ISyncStatus)lastSyncStatusForEntityName:(NSString *)entityName
```

### Discussion

For example, the last sync may have succeeded, may have failed, may be in progress, or may have been canceled (see “[ISyncStatus](#)” (page 20) for other possible return values). Returns [ISyncStatusNever](#) (page 21) if the client never synced the specified entity, or the entity is not supported.

The sync engine maintains the last sync information for as long as the client supports *entityName*. When a client stops supporting *entityName*, the last sync information for that entity is removed. If the client starts supporting *entityName* again, this method behaves as if the client never synced the entity.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [lastSyncDateForEntityName:](#) (page 12)

### Declared In

ISyncClient.h

## objectForKey:

Returns the object for *key* that was specified using the [setObject:forKey:](#) (page 15) method.

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

### Availability

Available in Mac OS X v10.4 and later.

**Declared In**

ISyncClient.h

**setDisplayName:**

Sets the display name for the receiver to *displayName*.

```
- (void)setDisplayName:(NSString *)displayName
```

**Discussion**

The display name may be used by GUI applications to graphically identify the client to users. You can also specify a display name when registering the client using the client description file.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [registerClientWithIdentifier:descriptionFilePath: \(ISyncManager\)](#)
- [displayName](#) (page 9)

**Declared In**

ISyncClient.h

**setEnabled:forEntityNames:**

If *flag* is YES, enables the entities specified by *entityNames*, otherwise disables them.

```
- (void)setEnabled:(BOOL)flag forEntityNames:(NSArray *)entityNames
```

**Discussion**

The *entityNames* array of NSString objects is expected to contain names of supported entities, otherwise an exception is raised.

The first time a client syncs, a panel appears asking the user if it's OK to sync entities belonging to a data class (a panel may appear for each data class). If the user declines then the entities are disabled, otherwise they are enabled. If you want to allow the user to enable entities, invoke this method by passing YES as the *flag* argument and all the entity names in the data class as the *entityNames* argument. Then the next time the client syncs, a panel appears again asking the user if it's OK to sync the data class.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [enabledEntityNames](#) (page 10)
- [isEnabledForEntityName:](#) (page 11)

**Declared In**

ISyncClient.h

**setFilters:**

Sets the receiver's filters used to control the records pulled from the sync engine to *filters*, an array of objects conforming to the ISyncFiltering protocol.

```
- (void)setFilters:(NSArray *)filters
```

**Discussion**

You use filters to define a subset of the records that this client syncs.

When pulling changes, the sync engine passes each record to each filter before giving changes to that record to the client. If any one of the filters rejects the record, it is not given to the client. See *ISyncFilter Class Reference* for some default filters.

This method recomputes the records that need to be sent to the client during the next sync operation which can be expensive. Consequently, do not invoke this method frequently.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [filters](#) (page 10)

**Related Sample Code**

People

**Declared In**

ISyncClient.h

**setFormatsRelationships:**

Sets whether the client may reformat relationships.

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

**Parameters**

*flag*

YES if the client reformats relationships; otherwise, NO.

**Availability**

Available in Mac OS X v10.6 and later.

**See Also**

- [formatsRelationships](#) (page 10)

**Declared In**

ISyncClient.h

**setImagePath:**

Sets the receiver's absolute image path to *path*.

```
- (void)setImagePath:(NSString *)path
```

**Discussion**

The image may be used by GUI applications to represent the client. You can also specify an image path when registering the client using the client description file.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [imagePath](#) (page 11) (ISyncManager)
- `registerClientWithIdentifier:descriptionFilePath:`

**Declared In**

ISyncClient.h

**setObject:forKey:**

Associates arbitrary information specified by a key-value pair to the receiver.

```
- (void)setObject:(id < NSCoding >)value forKey:(NSString *)key
```

**Discussion**

This method retains *value* and copy *key*. Pass *nil* for *value* to release a previously retained value. Use [objectForKey:](#) (page 12) to retrieve the value for a given key. The *value* is released when the client is unregistered.

This method is provided as a convenience for developers who have additional data they want to store with an object that is not defined in the schema. For example, use this method to store client-specific configuration information if multiple clients are associated with the same user defaults domain or if you want to store a sync anchor.

A sync anchor is an identifier exchanged between a client and a device, or between two clients running on different computers. Typically, the client that initiates a sync is passed a sync anchor to the device or another client at the end of a successful sync. The next time the client syncs, the recipient of the sync anchor passes the anchor back to the original client to verify that it is in a known state.

**Availability**

Available in Mac OS X v10.4 and later.

**Declared In**

ISyncClient.h

**setShouldReplaceClientRecords:forEntityNames:**

Sets whether or not a client should pull the truth—replace all its records for the specified entities on the next sync.

```
- (void)setShouldReplaceClientRecords:(BOOL)flag forEntityNames:(NSArray *)entityNames
```

**Discussion**

If *flag* is YES, the client should replace all its local records with the records pulled from the sync engine.

After invoking this method, sending `shouldReplaceClientRecordsForEntityName:` (page 18) to any new sessions created for this client returns YES, and sending `shouldPushChangesForEntityName:` or `shouldPushAllRecordsForEntityName:` returns NO.

This request takes effect on the next session created after invoking this method and remains in effect until the client successfully passes through the pull phase of that session. The sync engine needs to know whether a client is going to pull the truth before entering the negotiation phase. This is necessary to detect conflicting push the truth and pull the truth requests.

A client should not remove its local records until after the records are successfully pulled from the sync engine. The local records can be safely removed after `shouldReplaceClientRecordsForEntityName:` (page 18) returns YES.

This method is typically used by a configuration tool that allows the user to revert to the state of the truth.

#### Availability

Available in Mac OS X v10.4 and later.

#### Related Sample Code

People

SeeMyFriends

#### Declared In

ISyncClient.h

## setShouldSynchronize:withClientsOfType:

Adds the receiver as an observer of alerts when clients of the specified type sync.

```
- (void)setShouldSynchronize:(BOOL)flag withClientsOfType:(NSString *)clientType
```

#### Discussion

If *flag* is YES the receiver is added; otherwise the receiver is removed as an observer for alerts of the specified type. Alternatively, you can specify this information when registering the client using the client description file. You can invoke this method multiple times to register additional client types.

Typically, you use this method to setup a dependency between two clients. For example, Address Book might observe all types of clients, and is given an opportunity to join any syncs which synchronize entities defined in the contacts schema. The MobileMe client might observe only device clients, so it can join a Palm or phone sync session. The client is notified only if it has entities in common with the client that initiated the sync.

#### Availability

Available in Mac OS X v10.4 and later.

#### See Also

- `shouldSynchronizeWithClientsOfType:` (page 19) (ISyncManager)
- `setSyncAlertHandler:selector:` (page 17)
- `setSyncAlertToolPath:` (page 18)
- `registerClientWithIdentifier:descriptionFilePath:`

#### Related Sample Code

People

StickiesWithCoreData

**Declared In**

ISyncClient.h

**setSyncAlertHandler:selector:**

Sets the target and action to be invoked when an observed client creates a session and begins syncing.

```
- (void)setSyncAlertHandler:(id)handler selector:(SEL)selector
```

**Discussion**

When *selector* is sent to *handler*, your client has the opportunity to join the sync session.

The *selector* method is expected to take the receiver (an ISyncClient object) as the first argument and an array of entity names (an NSArray object) as the second argument. The method signature for *selector* should look like:

```
- (void)client:(ISyncClient *)client willSyncEntityNames:(NSArray *)entityNames
```

If *selector* returns without creating a session, the sync engine assumes the client will not join the session. If this client already has another handler registered—for example, from another client process—this method raises an exception. An observer is automatically removed when the client terminates.

When you create a session using the `beginSessionWithClient:entityNames:beforeDate:ISyncSession` class method, you specify how long you are willing to wait for the sync session. This is the length of time you are willing to wait for all the other clients to join the session. If a client takes too long to join a session, the sync engine may proceed without it.

Use this method instead of [setSyncAlertToolPath:](#) (page 18) if you want to notify a running application only. Use [setShouldSynchronize:withClientsOfType:](#) (page 16) to specify the types of clients the receiver wishes to observe. If both a tool and an observer are registered, only the observer is notified.

**Note:** If your client is multithreaded, the thread that registers the alert handler has to exist and have a run loop running, otherwise the client does not receive the alert.

**Availability**

Available in Mac OS X v10.4 and later.

**See Also**

- [shouldSynchronizeWithClientsOfType:](#) (page 19)
- [syncAlertToolPath](#) (page 19)

**Related Sample Code**

People

SeeMyFriends

**Declared In**

ISyncClient.h

## setSyncAlertToolPath:

Specifies the absolute path to a tool that is launched when an observed client creates a session and begins syncing.

```
- (void)setSyncAlertToolPath:(NSString *)path
```

### Discussion

The sync engine retains this path until the client is unregistered or you explicitly change the path using this method. Pass `nil` if you want to disable the sync alert tool.

When the tool is launched it passes the following command-line arguments:

```
--sync <clientIdentifier> --entitynames <entityNames>
```

The *clientIdentifier* argument is the identifier of the observed client. The *entityNames* argument is a single string containing the entity names delimited by commas that is synced. You can send `componentsSeparatedByString:` to the string with `@", "` as the argument to convert it to an array of entity names. The order of the key-value pairs, where `--sync` and `--entitynames` are keys, is arbitrary. If the tool terminates without creating a sync session, the sync engine assumes the client will not join the session.

When you create a session using the `beginSessionWithClient:entityNames:beforeDate:` `ISyncSession` class method, you specify how long you are willing to wait for the sync session. This is the time you are willing to wait for all the other clients to join the session. If a client takes too long to join a session, the sync engine may proceed without it.

Use this method instead of `setSyncAlertHandler:selector:` (page 17) if you want to notify an application or tool that may not be running. Use `setShouldSynchronize:withClientsOfType:` (page 16) to specify the types of clients the receiver wishes to observe. If both a tool and a handler are registered, only the handler is notified.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [shouldSynchronizeWithClientsOfType:](#) (page 19)
- [syncAlertToolPath](#) (page 19)

### Declared In

`ISyncClient.h`

## shouldReplaceClientRecordsForEntityName:

Returns YES if the client should replace all records for the entity specified by `entityName` during the next sync, NO otherwise.

```
- (BOOL)shouldReplaceClientRecordsForEntityName:(NSString *)entityName
```

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setShouldReplaceClientRecords:forEntityNames:](#) (page 15)

### Declared In

ISyncClient.h

## shouldSynchronizeWithClientsOfType:

Returns YES if the client is registered to receive alerts when clients of *clientType* sync, NO otherwise.

- (BOOL)shouldSynchronizeWithClientsOfType:(NSString \*)*clientType*

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [setShouldSynchronize:withClientsOfType:](#) (page 16)
- [setSyncAlertHandler:selector:](#) (page 17)
- [setSyncAlertToolPath:](#) (page 18)
- [syncAlertToolPath](#) (page 19)

### Declared In

ISyncClient.h

## supportedEntityNames

Returns an array of NSString objects containing the names of the entities the client supports.

- (NSArray \*)supportedEntityNames

### Discussion

This property is set when registering the client.

### Availability

Available in Mac OS X v10.4 and later.

### See Also

- [registerClientWithIdentifier:descriptionFilePath:](#) (ISyncManager)

### Related Sample Code

SeeMyFriends

### Declared In

ISyncClient.h

## syncAlertToolPath

Returns the path to the tool that is launched when an observed client begins syncing.

- (NSString \*)syncAlertToolPath

### Availability

Available in Mac OS X v10.4 and later.

**See Also**

- [setShouldSynchronize:withClientsOfTypes:](#) (page 16)
- [setSyncAlertHandler:selector:](#) (page 17)
- [setSyncAlertToolPath:](#) (page 18)
- [shouldSynchronizeWithClientsOfTypes:](#) (page 19)

**Declared In**

ISyncClient.h

## Constants

### Client Types

Specifies the type of client you might want to sync simultaneously with using the [setShouldSynchronize:withClientsOfTypes:](#) (page 16) method. The [clientType](#) (page 9) method also returns one of these constants.

```
extern NSString * const ISyncClientTypeApplication;
extern NSString * const ISyncClientTypeDevice;
extern NSString * const ISyncClientTypeServer;
extern NSString * const ISyncClientTypePeer;
```

**Constants**

ISyncClientTypeApplication

Indicates the client is an application, such as Mail or iCal.

Available in Mac OS X v10.4 and later.

Declared in ISyncClient.h.

ISyncClientTypeDevice

Indicates the client is used to sync a device such as a phone or an iPod.

Available in Mac OS X v10.4 and later.

Declared in ISyncClient.h.

ISyncClientTypeServer

Indicates the client is used to sync a remote server such as MobileMe.

Available in Mac OS X v10.4 and later.

Declared in ISyncClient.h.

ISyncClientTypePeer

Indicates the client is a peer, such as another computer.

Available in Mac OS X v10.4 and later.

Declared in ISyncClient.h.

### ISyncStatus

The following constants are returned by the [lastSyncStatusForEntityName:](#) (page 12) method to indicate the state of the last sync session.

```
typedef SInt32 ISyncStatus;  
enum __ISyncStatus {  
    ISyncStatusRunning=1,  
    ISyncStatusSuccess,  
    ISyncStatusWarnings,  
    ISyncStatusErrors,  
    ISyncStatusCancelled,  
    ISyncStatusFailed,  
    ISyncStatusNever  
};
```

### Constants

`ISyncStatusRunning`

Indicates the client is syncing.

Available in Mac OS X v10.4 and later.

Declared in `ISyncClient.h`.

`ISyncStatusSuccess`

Indicates the last sync was successful.

Available in Mac OS X v10.4 and later.

Declared in `ISyncClient.h`.

`ISyncStatusWarnings`

Indicates the last sync resulted in warnings.

Available in Mac OS X v10.4 and later.

Declared in `ISyncClient.h`.

`ISyncStatusErrors`

Indicates the last sync resulted in errors.

Available in Mac OS X v10.4 and later.

Declared in `ISyncClient.h`.

`ISyncStatusCancelled`

Indicates the last sync was canceled.

Available in Mac OS X v10.4 and later.

Declared in `ISyncClient.h`.

`ISyncStatusFailed`

Indicates the last sync failed to complete (for example, the client crashed).

Available in Mac OS X v10.4 and later.

Declared in `ISyncClient.h`.

`ISyncStatusNever`

Indicates the client has never synced.

Available in Mac OS X v10.4 and later.

Declared in `ISyncClient.h`.



# Document Revision History

---

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

Date	Notes
2009-03-27	Updated for Mac OS X v10.6 by adding the <code>formatsRelationships</code> and <code>setFormatsRelationships:</code> methods.
2007-07-11	First publication of this content as a separate document.

## REVISION HISTORY

### Document Revision History

# Index

---

## C

---

canPullChangesForEntityName: [instance method 8](#)  
canPushChangesForEntityName: [instance method 8](#)  
Client Types [20](#)  
clientIdentifier [instance method 9](#)  
clientType [instance method 9](#)

## D

---

displayName [instance method 9](#)

## E

---

enabledEntityNames [instance method 10](#)

## F

---

filters [instance method 10](#)  
formatsRelationships [instance method 10](#)

## I

---

imagePath [instance method 11](#)  
isEnabledForEntityName: [instance method 11](#)  
ISyncClientTypeApplication [constant 20](#)  
ISyncClientTypeDevice [constant 20](#)  
ISyncClientTypePeer [constant 20](#)  
ISyncClientTypeServer [constant 20](#)  
ISyncStatus [20](#)  
ISyncStatusCancelled [constant 21](#)  
ISyncStatusErrors [constant 21](#)  
ISyncStatusFailed [constant 21](#)  
ISyncStatusNever [constant 21](#)  
ISyncStatusRunning [constant 21](#)

ISyncStatusSuccess [constant 21](#)  
ISyncStatusWarnings [constant 21](#)

## L

---

lastSyncDateForEntityName: [instance method 12](#)  
lastSyncStatusForEntityName: [instance method 12](#)

## O

---

objectForKey: [instance method 12](#)

## S

---

setDisplayname: [instance method 13](#)  
setEnabled:forEntityNames: [instance method 13](#)  
setFilters: [instance method 14](#)  
setFormatsRelationships: [instance method 14](#)  
setImagePath: [instance method 14](#)  
setObject:forKey: [instance method 15](#)  
setShouldReplaceClientRecords:forEntityNames:  
[instance method 15](#)  
setShouldSynchronize:withClientsOfType:  
[instance method 16](#)  
setSyncAlertHandler:selector: [instance method 17](#)  
setSyncAlertToolPath: [instance method 18](#)  
shouldReplaceClientRecordsForEntityName:  
[instance method 18](#)  
shouldSynchronizeWithClientsOfType: [instance method 19](#)  
supportedEntityNames [instance method 19](#)  
syncAlertToolPath [instance method 19](#)