Calendar Store Framework Reference

Data Management: Calendar Data

2009-07-28

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

Simultaneously published in the United States and Canada.

Contents

Introduction

Framework	/System/Library/Frameworks/CalendarStore.framework
Header file directories	CalendarStore.framework/Headers
Declared in	CalAlarm.h CalAttendee.h CalCalendar.h CalCalendarItem.h CalCalendarStore.h CalEvent.h CalRecurrenceRule.h CalTask.h CalendarStoreErrors.h

This collection of documents describes classes and methods of the Calendar Store framework. The Calendar Store framework provides read and write access to iCal data. Using a shared `CalCalendarStore` object, you can fetch calendars, events, and tasks from the iCal data storage. You can also build queries using `NSPredicate` objects to fetch specific sets of events and tasks. The Calendar Store framework provides a few convenience methods for creating common queries. Typically, you change calendar, event, and task objects by accessing their properties directly, and then invoke the appropriate `save...` method to save your changes to the iCal database. You can also register for change notifications to update previously fetched calendar, event, and task objects when they change.

Concurrency Note: The Calendar Store framework is thread safe. No additional locking or other synchronization is required. The framework handles synchronization for you.

Introduction

Classes

PART I

Classes

CalAlarm Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Declared in	CalendarStore/CalAlarm.h
Availability	Available in Mac OS X v10.5 and later.
Companion guide	Calendar Store Programming Guide

Overview

The `CalAlarm` class represents alarm objects in iCal. Use the `alarm` (page 14) class method to create an alarm and use the properties to set information about an alarm. Use the `triggerDateRelativeTo:` (page 14) method if you need to know how much time is left before an alarm triggers. Use `CalCalendarItem` methods to add alarms to, or remove alarms from, events and tasks.

Tasks

Creating and Initializing Alarms

`+ alarm` (page 14)
Creates and returns a new alarm object.

Getting and Setting Properties

`absoluteTrigger` (page 12) *property*
The date and time to trigger the alarm.

`action` (page 12) *property*
The action to take when triggering the alarm.

`emailAddress` (page 13) *property*
An email address that is the recipient of an email alarmâ alarm that triggers an email message.

`relativeTrigger` (page 13) *property*
The relative date and time to trigger the alarm.

`sound` (page 13) *property*
The sound to play when the alarm triggers.

`url` (page 13) *property*
The URL to open when the alarm triggers.

Getting Relative Dates

`- triggerDateRelativeTo:` (page 14)
Returns a delta value between the specified date and the date that the alarm is scheduled to trigger.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

absoluteTrigger

The date and time to trigger the alarm.

```
@property(copy) NSDate *absoluteTrigger
```

Discussion
When you set the `absoluteTrigger` property, the `relativeTrigger` (page 13) property is set to `nil`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

action

The action to take when triggering the alarm.

```
@property(copy) NSString *action
```

Discussion
The value of this property is one of the constants described in "`Alarm Actions`" (page 15).

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

emailAddress

An email address that is the recipient of an email alarmâ alarm that triggers an email message.

```
@property(copy) NSString *emailAddress
```

Discussion
When you set the `emailAddress` property, the `action` (page 12) property is set to `CalAlarmActionEmail` (page 15), and the `sound` (page 13) and `url` (page 13) properties are set to `nil`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

relativeTrigger

The relative date and time to trigger the alarm.

```
@property NSTimeInterval relativeTrigger
```

Discussion
When you set the `relativeTrigger` property, the `absoluteTrigger` (page 12) property is set to `nil`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

sound

The sound to play when the alarm triggers.

```
@property(copy) NSString *sound
```

Discussion
The value of this property is the name of a system sound that can be used with the `soundNamed:` class method to create an `NSSound` object. When you set the `sound` property, the `action` (page 12) property is set to `CalAlarmActionSound` (page 15), and the `emailAddress` (page 13) and `url` (page 13) properties are set to `nil`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

url

The URL to open when the alarm triggers.

```
@property(copy) NSURL *url
```

Discussion
When you set the `url` property, the `action` (page 12) property is set to `CalAlarmActionProcedure` (page 15), and the `emailAddress` (page 13) and `sound` (page 13) properties are set to `nil`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

Class Methods

alarm

Creates and returns a new alarm object.

```
+ (id)alarm
```

Return Value
Newly initialized `CalAlarm` object.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

Instance Methods

triggerDateRelativeTo:

Returns a delta value between the specified date and the date that the alarm is scheduled to trigger.

```
- (NSDate *)triggerDateRelativeTo:(NSDate *)date
```

Parameters
date
The start date that you want to compute the delta date from.

Return Value
The delta value between *date* and the date the alarm triggers.

Discussion
Use this method if you need to know precisely how long it will be before an alarm triggers. Alarms with relative triggers do not contain this information in the properties. Use this method to compute the delta value. For example, pass the current date and this method returns the time remaining until the alarm triggers.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAlarm.h`

Constants

Alarm Actions

The action to take when an alarm triggers.

```
extern NSString * const CalAlarmActionDisplay;
extern NSString * const CalAlarmActionEmail;
extern NSString * const CalAlarmActionProcedure;
extern NSString * const CalAlarmActionSound;
```

Constants

`CalAlarmActionDisplay`
A message should be displayed when an alarm triggers.
Available in Mac OS X v10.5 and later.
Declared in `CalAlarm.h`.

`CalAlarmActionEmail`
An email message should be sent when an alarm triggers.
Available in Mac OS X v10.5 and later.
Declared in `CalAlarm.h`.

`CalAlarmActionProcedure`
A file should be opened when an alarm triggers.
Available in Mac OS X v10.5 and later.
Declared in `CalAlarm.h`.

`CalAlarmActionSound`
A sound should be played when an alarm triggers.
Available in Mac OS X v10.5 and later.
Declared in `CalAlarm.h`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalendarStore/CalAlarm.h`

CalAttendee Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Declared in	CalendarStore/CalAttendee.h
Availability	Available in Mac OS X v10.5 and later.
Companion guide	Calendar Store Programming Guide

Overview

A `CalAttendee` object represents one attendee of a calendar event.

You do not create `CalAttendee` objects directly. Send `attendees` to a `CalEvent` object to get an array of `CalAttendee` objects.

Use the `status` (page 18) property to get the confirmation status of an attendee. Use the `commonName` (page 18) and `address` (page 18) properties to get more information about an attendee. You cannot modify the properties of a `CalAttendee` object—they are read-only.

Tasks

Getting Properties

`address` (page 18) *property*
An `NSURL` object that can be used to contact the attendee.

`commonName` (page 18) *property*
The user-entered name of the attendee.

`status` (page 18) *property*
The attendee's confirmation status.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

address

An `NSURL` object that can be used to contact the attendee.

```
@property(readonly) NSURL *address;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAttendee.h`

commonName

The user-entered name of the attendee.

```
@property(readonly) NSString *commonName;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAttendee.h`

status

The attendee's confirmation status.

```
@property(readonly) NSString *status;
```

Discussion
The value of this property is one of the constants in "`CalAttendee Status Strings`" (page 19).

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalAttendee.h`

Constants

CalAttendee Status Strings

Describe an attendee's confirmation status of an event.

```
extern NSString * const CalAttendeeStatusNeedsAction;
extern NSString * const CalAttendeeStatusAccepted;
extern NSString * const CalAttendeeStatusDeclined;
extern NSString * const CalAttendeeStatusTentative;
```

Constants

`CalAttendeeStatusNeedsAction`

The status is not set for this attendee.

Available in Mac OS X v10.5 and later.

Declared in `CalAttendee.h`.

`CalAttendeeStatusAccepted`

The attendee accepted the invitation.

Available in Mac OS X v10.5 and later.

Declared in `CalAttendee.h`.

`CalAttendeeStatusDeclined`

The attendee declined the invitation.

Available in Mac OS X v10.5 and later.

Declared in `CalAttendee.h`.

`CalAttendeeStatusTentative`

The attendee's status is tentative.

Available in Mac OS X v10.5 and later.

Declared in `CalAttendee.h`.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`CalendarStore/CalAttendee.h`

CalCalendar Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Declared in	CalendarStore/CalCalendar.h
Availability	Available in Mac OS X v10.5 and later.
Companion guide	Calendar Store Programming Guide
Related sample code	CalendarItems Reminders

Overview

A `CalCalendar` object represents a calendar in iCal.

Use the `calendar` (page 24) method to create a `CalCalendar` object directly, or use the `CalCalendarStore calendars` method to get an array of all the calendar objects. If you know the calendar UID, use the `calendarWithUID:` `CalCalendarStore` method to get the associated calendar object.

Use the properties in this class to get attributes about a calendar—for example, use the properties to get the title and color of a calendar. Use the `eventsWithPredicate:` and `tasksWithPredicate:` `CalCalendarStore` methods to fetch associated events and tasks.

If you retain `CalCalendar` objects, then register for the `CalCalendarsChangedNotification` notification.

Tasks

Creating Calendars

`+ calendar` (page 24)
Creates and returns a new `CalCalendar` object.

Getting Properties

`color` (page 22) *property*
The calendar's color.

`isEditable` (page 22) *property*
A Boolean value indicating whether the calendar is editable or not.

`notes` (page 23) *property*
Textual notes about the calendar.

`title` (page 23) *property*
The calendar's title.

`type` (page 23) *property*
The type of calendar.

`uid` (page 23) *property*
A unique identifier for the receiver.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

color

The calendar's color.

```
@property(copy) NSColor * color;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendar.h`

isEditable

A Boolean value indicating whether the calendar is editable or not.

```
@property(readonly) BOOL isEditable;
```

Discussion
Set to `YES` if the calendar is editable; otherwise, `NO`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendar.h`

notes

Textual notes about the calendar.

```
@property(copy) NSString * notes;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendar.h`

title

The calendar's title.

```
@property(copy) NSString * title;
```

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
Reminders

Declared In
`CalCalendar.h`

type

The type of calendar.

```
@property(readonly) NSString * type;
```

Discussion
The value of this property is one of the constants described in "`Calendar Types`" (page 24).

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendar.h`

uid

A unique identifier for the receiver.

```
@property(readonly) NSString * uid;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendar.h`

Class Methods

calendar

Creates and returns a new `CalCalendar` object.

```
+ (id)calendar
```

Return Value
A newly created and initialized `CalCalendar` object.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
Reminders

Declared In
`CalCalendar.h`

Constants

Calendar Types

The type of calendar.

```
extern NSString * const CalCalendarTypeBirthday;
extern NSString * const CalCalendarTypeCalDAV;
extern NSString * const CalCalendarTypeExchange;
extern NSString * const CalCalendarTypeIMAP;
extern NSString * const CalCalendarTypeLocal;
extern NSString * const CalCalendarTypeSubscription;
```

Constants

`CalCalendarTypeBirthday`
An birthday calendar maintained by Address Book.
Available in Mac OS X v10.5 and later.
Declared in `CalCalendar.h`.

`CalCalendarTypeCalDAV`
A CalDAV server calendar.
Available in Mac OS X v10.5 and later.
Declared in `CalCalendar.h`.

`CalCalendarTypeExchange`
An Exchange calendar.
Available in Mac OS X v10.6 and later.
Declared in `CalCalendar.h`.

`CalCalendarTypeIMAP`
: An IMAP calendar.

  Available in Mac OS X v10.5 and later.

  Declared in `CalCalendar.h`.

`CalCalendarTypeLocal`
: A local calendar that may be synced over .Mac.

  Available in Mac OS X v10.5 and later.

  Declared in `CalCalendar.h`.

`CalCalendarTypeSubscription`
: A subscribed calendar.

  Available in Mac OS X v10.5 and later.

  Declared in `CalCalendar.h`.

Declared In
`CalendarStore/CalCalendar.h`

CalCalendarItem Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Declared in	CalendarStore/CalCalendarItem.h
Availability	Available in Mac OS X v10.5 and later.
Companion guide	Calendar Store Programming Guide

Overview

`CalCalendarItem` is an abstract superclass for `CalEvent` and `CalTask` objects. It provides common properties for its subclasses such as the title and associated calendar object of a calendar item. You do not create `CalCalendarItem` objects directly. Use the `CalCalendarStore` methods to fetch events and tasks. When you have a `CalCalendarItem` object you can add and remove alarms using the methods listed in "Setting Alarms" (page 28).

Tasks

Getting and Setting Properties

`calendar` (page 28) *property*
The associated calendar object for the calendar item.

`notes` (page 29) *property*
The notes about the calendar item.

`url` (page 30) *property*
The URL for the calendar item.

`title` (page 29) *property*
The title of the calendar item.

`uid` (page 30) *property*
The calendar item's unique identifier. This property is read-only.

`dateStamp` (page 29) *property*
The date the calendar item was last modified (not the same as the date is was last synced). This property is read-only.

`alarms` (page 28) *property*
An array containing the calendar item's alarm objects—an array of `CalAlarm` objects.

Setting Alarms

- `hasAlarm` (page 31)
Returns whether or not the receiver has an alarm.
- `nextAlarmDate` (page 31)
Returns the date of the next alarm.
- `addAlarm:` (page 30)
Adds an alarm to the receiver.
- `addAlarms:` (page 30)
Adds the alarms contained in an array to the receiver.
- `removeAlarm:` (page 31)
Removes the specified alarm from the receiver.
- `removeAlarms:` (page 32)
Removes the alarms contained in an array from the receiver.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

alarms

An array containing the calendar item's alarm objects—an array of `CalAlarm` objects.

```
@property(copy) NSArray *alarms;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

calendar

The associated calendar object for the calendar item.

```
@property(retain) CalCalendar *calendar;
```

Discussion
An error occurs if you attempt to save a calendar item without first setting the calendar property.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems
Reminders

Declared In
`CalCalendarItem.h`

dateStamp

The date the calendar item was last modified (not the same as the date is was last synced). This property is read-only.

```
@property(readonly) NSDate *dateStamp;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

notes

The notes about the calendar item.

```
@property(copy) NSString *notes;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

title

The title of the calendar item.

```
@property(copy) NSString *title;
```

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems
Reminders

Declared In
`CalCalendarItem.h`

uid

The calendar item's unique identifier. This property is read-only.

```
@property(readonly) NSString *uid;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

url

The URL for the calendar item.

```
@property(copy) NSURL *url;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

Instance Methods

addAlarm:

Adds an alarm to the receiver.

```
- (void)addAlarm:(CalAlarm *)alarm
```

Parameters

alarm
The alarm to add.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

addAlarms:

Adds the alarms contained in an array to the receiver.

```
- (void)addAlarms:(NSArray *)alarms
```

Parameters

alarms

An array of `CalAlarm` objects to add.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

hasAlarm

Returns whether or not the receiver has an alarm.

```
- (BOOL)hasAlarm
```

Return Value
`YES` if the receiver has an alarm; otherwise, `NO`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

nextAlarmDate

Returns the date of the next alarm.

```
- (NSDate *)nextAlarmDate
```

Return Value
The date the next alarm triggers.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

removeAlarm:

Removes the specified alarm from the receiver.

```
- (void)removeAlarm:(CalAlarm *)alarm
```

Parameters

alarm

The alarm to remove.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

removeAlarms:

Removes the alarms contained in an array from the receiver.

```
- (void)removeAlarms:(NSArray *)alarms
```

Parameters

alarms

An array of `CalAlarm` objects to remove.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarItem.h`

CalCalendarStore Class Reference

Inherits from	NSObject
Conforms to	NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	CalendarStore/CalCalendarStore.h
Companion guide	Calendar Store Programming Guide
Related sample code	CalendarItems Reminders SimpleCalendar

Overview

There is only one `CalCalendarStore` object and it represents a connection to the iCal database. You retrieve all other types of calendar objects—including calendars, events, and tasks—using this shared `CalCalendarStore` object.

Use the `defaultCalendarStore` (page 35) class method to get the shared `CalCalendarStore` object. Use the `calendars` (page 39) method to fetch all the `CalCalendar` objects, and the `calendarWithUID:` (page 39) method to fetch individual calendars. Use the `eventsWithPredicate:` (page 40) and `tasksWithPredicate:` (page 45) methods to fetch events and tasks. These methods take an `NSPredicate` object as the argument. Use the `eventPredicate...` and `taskPredicate...` methods to create `NSPredicate` objects for common queries that you can pass to these methods.

Use specific `CalCalendar`, `CalEvent`, and `CalTask` properties and methods to make changes to these types of objects. If you make local changes to calendars, events, and tasks, invoke the corresponding `save...` method; otherwise, your changes do not persist. You might also need to track changes.

Observing Notifications

Calendars, events, and tasks may be added, changed, or deleted after you fetch them. Changes can occur locally or externally where local changes are made by your application and external changes are made by other applications. When these objects change, you might want to take some action, especially if you retain fetched objects—for example, update the display. You can track changes by observing notifications.

The following notifications are posted when your application changes these types of objects: `CalCalendarsChangedNotification` (page 48), `CalEventsChangedNotification` (page 49), and `CalTasksChangedNotification` (page 49).

The following notifications are posted when another application changes these types of objects: `CalCalendarsChangedExternallyNotification` (page 49), `CalEventsChangedExternallyNotification` (page 50), and `CalTasksChangedExternallyNotification` (page 50) notifications.

Tasks

Creating and Initializing

`+ defaultCalendarStore` (page 35)
Returns the shared `CalCalendarStore` object.

Accessing Calendars

`- calendars` (page 39)
Returns an array of `CalCalendar` objects representing the user's calendars in the order they appear in iCal.

`- calendarWithUID:` (page 39)
Returns a `CalCalendar` object that corresponds to the given UID.

`- saveCalendar:error:` (page 43)
Saves local changes to the specified calendar object to the iCal database.

`- removeCalendar:error:` (page 41)
Removes the specified calendar from the iCal database.

Accessing Events

`- eventsWithPredicate:` (page 40)
Returns an array of `CalEvent` objects that match the specified predicate.

`- eventWithUID:occurrence:` (page 41)
Returns an event that matches the specified unique identifier.

`- saveEvent:span:error:` (page 43)
Saves the specified event to the calendar store.

`- removeEvent:span:error:` (page 42)
Removes the specified event from the calendar store.

Accessing Tasks

`- tasksWithPredicate:` (page 45)
Returns an array of `CalTask` objects that match the specified predicate.

- `taskWithUID:` (page 46)
  Returns a task that matches the specified unique identifier.
- `saveTask:error:` (page 44)
  Saves the specified task to the calendar store.
- `removeTask:error:` (page 42)
  Removes the specified task from the calendar store.

Creating Predicates

+ `eventPredicateWithStartDate:endDate:calendars:` (page 36)
  Returns an `NSPredicate` object that specifies events within a date range that belong to the specified calendars.
+ `eventPredicateWithStartDate:endDate:UID:calendars:` (page 36)
  Returns an `NSPredicate` object that specifies events with a UID and within a date range that belong to the specified calendars.
+ `taskPredicateWithCalendars:` (page 37)
  Returns an `NSPredicate` object that specifies tasks that belong to the specified calendars.
+ `taskPredicateWithUncompletedTasks:` (page 38)
  Returns an `NSPredicate` object that specifies tasks that are incomplete and belong to the specified calendars.
+ `taskPredicateWithUncompletedTasksDueBefore:calendars:` (page 38)
  Returns an `NSPredicate` object that specifies incomplete tasks due before the specified date and that belong to the specified calendars.
+ `taskPredicateWithTasksCompletedSince:calendars:` (page 37)
  Returns an `NSPredicate` object that specifies completed tasks since the specified date and that belong to the specified calendars.

Class Methods

defaultCalendarStore

Returns the shared `CalCalendarStore` object.

```
+ (CalCalendarStore *)defaultCalendarStore
```

Return Value
The shared `CalCalendarStore` object.

The first time an application invokes this method, Calendar Store might migrate the calendar data from a previous Mac OS X file format to the current one. This method returns `nil` if the migration fails.

Discussion
Invoke this method to get the shared instance. Do not create a `CalCalendarStore` object directly.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems
Reminders
SimpleCalendar

Declared In
`CalCalendarStore.h`

eventPredicateWithStartDate:endDate:calendars:

Returns an `NSPredicate` object that specifies events within a date range that belong to the specified calendars.

```
+ (NSPredicate *)eventPredicateWithStartDate:(NSDate *)startDate endDate:(NSDate
    *)endDate calendars:(NSArray *)calendars
```

Parameters

`startDate`
The start date of the date range.

`endDate`
The end date of the date range.

`calendars`
An array of `CalCalendar` objects that the events must belong to.

Return Value
An `NSPredicate` object that specifies events within a date range that belong to the specified calendars.

Discussion
If an event is greater than or equal to *`startDate`* and less than or equal to *`endDate`*, and belongs to a calendar in *`calendars`*, then it is included in the predicate's result.

Availability
Available in Mac OS X v10.5 and later.

See Also
`+ eventPredicateWithStartDate:endDate:UID:calendars:` (page 36)

Related Sample Code
CalendarItems
Reminders

Declared In
`CalCalendarStore.h`

eventPredicateWithStartDate:endDate:UID:calendars:

Returns an `NSPredicate` object that specifies events with a UID and within a date range that belong to the specified calendars.

```
+ (NSPredicate *)eventPredicateWithStartDate:(NSDate *)startDate endDate:(NSDate
    *)endDate UID:(NSString *)UID calendars:(NSArray *)calendars
```

Parameters

startDate

The start date of the date range.

endDate

The end date of the date range.

UID

The UID for the returned events.

calendars

An array of `CalCalendar` objects that the events must belong to.

Return Value

An `NSPredicate` object that specifies events with a UID and within a date range that belong to the specified calendars.

Discussion

Recurring events have the same UID. Therefore, use this method, instead of the `eventPredicateWithStartDate:endDate:calendars:` method, if you want to fetch all events belonging to the same master recurring event.

Availability

Available in Mac OS X v10.5 and later.

See Also

`+ eventPredicateWithStartDate:endDate:calendars:` (page 36)

Declared In

`CalCalendarStore.h`

taskPredicateWithCalendars:

Returns an `NSPredicate` object that specifies tasks that belong to the specified calendars.

```
+ (NSPredicate *)taskPredicateWithCalendars:(NSArray *)calendars
```

Parameters

calendars

An array of `CalCalendar` objects that the tasks must belong to.

Return Value

An `NSPredicate` object that specifies tasks that belong to the specified calendars.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`CalCalendarStore.h`

taskPredicateWithTasksCompletedSince:calendars:

Returns an `NSPredicate` object that specifies completed tasks since the specified date and that belong to the specified calendars.

```
+ (NSPredicate *)taskPredicateWithTasksCompletedSince:(NSDate *)completedSince
    calendars:(NSArray *)calendars
```

Parameters

completedSince

Specifies the completion date.

calendars

An array of `CalCalendar` objects that the tasks must belong to.

Discussion

If a task is completed after *completedSince* and belongs to a calendar in *calendars*, then it is included in the returned predicate's result.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`CalCalendarStore.h`

taskPredicateWithUncompletedTasks:

Returns an `NSPredicate` object that specifies tasks that are incomplete and belong to the specified calendars.

```
+ (NSPredicate *)taskPredicateWithUncompletedTasks:(NSArray *)calendars
```

Parameters

calendars

An array of `CalCalendar` objects that the tasks must belong to.

Return Value

An `NSPredicate` object that specifies tasks that are incomplete and belong to the specified calendars.

Availability

Available in Mac OS X v10.5 and later.

Related Sample Code

CalendarItems

Declared In

`CalCalendarStore.h`

taskPredicateWithUncompletedTasksDueBefore:calendars:

Returns an `NSPredicate` object that specifies incomplete tasks due before the specified date and that belong to the specified calendars.

```
+ (NSPredicate *)taskPredicateWithUncompletedTasksDueBefore:(NSDate *)dueDate
    calendars:(NSArray *)calendars
```

Parameters

dueDate

Specifies the due date of the tasks.

calendars
An array of `CalCalendar` objects that the tasks must belong to.

Discussion
If an incomplete task's due date is before *dueDate* and belongs to a calendar in *calendars*, then it is included in the returned predicate's result.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
Reminders

Declared In
`CalCalendarStore.h`

Instance Methods

calendars

Returns an array of `CalCalendar` objects representing the user's calendars in the order they appear in iCal.

```
- (NSArray *)calendars
```

Return Value
An array of `CalCalendar` objects. Returns an empty array if the user hasn't launched iCal and created any calendars.

Discussion
This method returns an empty array if the user has calendar data from a previous version of Mac OS X but has not launched iCal in Mac OS X v10.5 yet. iCal needs to be launched at least once on Mac OS X v10.5 to migrate the user's calendar data from a previous version. If no calendar data from a previous version exists, then this method creates and returns the default calendars.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- calendarWithUID:` (page 39)

Related Sample Code
CalendarItems
Reminders
SimpleCalendar

Declared In
`CalCalendarStore.h`

calendarWithUID:

Returns a `CalCalendar` object that corresponds to the given UID.

```
- (CalCalendar *)calendarWithUID:(NSString *)UID
```

Parameters

UID

A string that uniquely identifies a calendar.

Return Value

The calendar that corresponds to the UID. Returns `nil` if no calendar with the specified UID exists.

Availability

Available in Mac OS X v10.5 and later.

See Also

- `calendars` (page 39)

Related Sample Code

CalendarItems

Reminders

Declared In

`CalCalendarStore.h`

eventsWithPredicate:

Returns an array of `CalEvent` objects that match the specified predicate.

```
- (NSArray *)eventsWithPredicate:(NSPredicate *)predicate
```

Parameters

predicate

Describes the set of records that should be returned. If `nil`, an exception is raised.

Return Value

An array of `CalEvent` objects that match the specified predicate. Returns `nil` if the predicate is invalid.

Discussion

The predicate passed to this method must be valid—it must be created using one of the class methods defined in this class to create predicates, the `eventPredicateWith...` or `taskPredicateWith...` methods.

For performance reasons, this method only returns events that fall within a specific four year timespan. If the date range between the predicate's start date and end date is greater than four years, then the timespan containing events is always the first four years of the date range

If you observe the `CalEventsChangedNotification` notification, you can retain the predicate and send it `evaluateWithObject:`, passing the changed event to determine whether the notification affects the event objects you previously fetched.

Availability

Available in Mac OS X v10.5 and later.

See Also

- `eventWithUID:occurrence:` (page 41)

Related Sample Code
CalendarItems
Reminders
SimpleCalendar

Declared In
`CalCalendarStore.h`

eventWithUID:occurrence:

Returns an event that matches the specified unique identifier.

```
- (CalEvent *)eventWithUID:(NSString *)uid occurrence:(NSDate *)date
```

Parameters

uid
The unique identifier of an event.

date
The date of a recurring event. Pass `nil` if the event is not recurring.

Return Value
A `CalEvent` object that matches the specified unique identifier and date. Returns `nil` if the event is not found, or the event is recurring and *date* is not specified.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- eventsWithPredicate:` (page 40)

Declared In
`CalCalendarStore.h`

removeCalendar:error:

Removes the specified calendar from the iCal database.

```
- (BOOL)removeCalendar:(CalCalendar *)calendar error:(NSError **)error
```

Parameters

calendar
The calendar object to remove. Must be a local calendar not a subscribed, birthday, or CalDAV calendar.

error
If this method returns `NO`, an `NSError` object describing the error. See *Calendar Store Constants Reference* for description of error codes.

Return Value
If successful, returns `YES`; otherwise, returns `NO` and sets the *error* argument to an `NSError` object describing the error.

Availability
Available in Mac OS X v10.5 and later.

See Also
- saveCalendar:error: (page 43)

Declared In
CalCalendarStore.h

removeEvent:span:error:

Removes the specified event from the calendar store.

```
- (BOOL)removeEvent:(CalEvent *)event span:(CalSpan)span error:(NSError **)error
```

Parameters

event

The event to remove.

span

Specifies the span of a recurring event—whether the change should be applied to future occurrences, all occurrences, or just this instance. Applying changes to future occurrences or all occurrences may cause the UID or occurrence date of the event to change.

error

If this method returns NO, an NSError object describing the error. See *Calendar Store Constants Reference* for description of error codes.

Return Value
If successful, returns YES; otherwise, returns NO and sets the *error* argument to an NSError object describing the error.

Availability
Available in Mac OS X v10.5 and later.

See Also
- saveEvent:span:error: (page 43)

Declared In
CalCalendarStore.h

removeTask:error:

Removes the specified task from the calendar store.

```
- (BOOL)removeTask:(CalTask *)task error:(NSError **)error
```

Parameters

task

The task to remove.

error

If this method returns NO, an NSError object describing the error. See *InstantMessage Constants Reference* for description of error codes.

Return Value
If successful, returns YES; otherwise, returns NO and sets the *error* argument to an NSError object describing the error.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `saveTask:error:` (page 44)

Declared In
`CalCalendarStore.h`

saveCalendar:error:

Saves local changes to the specified calendar object to the iCal database.

```
- (BOOL)saveCalendar:(CalCalendar *)calendar error:(NSError **)error
```

Parameters

calendar
The calendar object to save. Must be a local calendar not a subscribed, birthday, or CalDAV calendar.

error
If this method returns `NO`, an `NSError` object describing the error. See *InstantMessage Constants Reference* for description of error codes.

Return Value
If successful, returns `YES`; otherwise, returns `NO` and sets the *error* argument to an `NSError` object describing the error.

Discussion
Use this method to both create a new calendar and save changes to an existing calendar. Changes you make to calendars do not persist unless you invoke this method.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `removeCalendar:error:` (page 41)

Related Sample Code
Reminders

Declared In
`CalCalendarStore.h`

saveEvent:span:error:

Saves the specified event to the calendar store.

```
- (BOOL)saveEvent:(CalEvent *)event span:(CalSpan)span error:(NSError **)error
```

Parameters

event
The event to save.

span

Specifies the span of a recurring event—if the change should be applied to future occurrences, all occurrences, or just this instance. Applying changes to future occurrences or all occurrences may cause the UID or occurrence date of the event to change.

error

If this method returns `NO`, an `NSError` object describing the error. See *InstantMessage Constants Reference* for description of error codes.

Return Value
If successful, returns `YES`; otherwise, returns `NO` and sets the *error* argument to an `NSError` object describing the error.

Discussion
Use this method to save new event objects and modifications to existing event objects. Changes to event objects are not persistent until this method is invoked. The `calendar` (page 28) property needs to be set before attempting to save an event.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- removeEvent:span:error:` (page 42)

Related Sample Code
Reminders

Declared In
`CalCalendarStore.h`

saveTask:error:

Saves the specified task to the calendar store.

```
- (BOOL)saveTask:(CalTask *)task error:(NSError **)error
```

Parameters

task

The task to save.

error

If this method returns `NO`, an `NSError` object describing the error. See *Calendar Store Constants Reference* for description of error codes.

Return Value
If successful, returns `YES`; otherwise, returns `NO` and sets the *error* argument to an `NSError` object describing the error.

Discussion
Use this method to save new task objects and modifications to existing task objects. Changes to task objects are not persistent until this method is invoked. The `calendar` (page 28) property needs to be set before attempting to save a task.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `removeTask:error:` (page 42)

Related Sample Code
Reminders

Declared In
`CalCalendarStore.h`

tasksWithPredicate:

Returns an array of `CalTask` objects that match the specified predicate.

```
- (NSArray *)tasksWithPredicate:(NSPredicate *)predicate
```

Parameters

predicate

Describes the set of records that should be returned.

Return Value
An array of `CalTask` objects. Returns `nil` if the predicate is invalid.

Discussion
The predicate passed to this method must be valid. You create valid predicates using the predicate methods described in this class. The predicate must specify an array of calendars to query and can filter the tasks by due date and completion date. All subpredicates must be joined with an `AND`, and all dates and calendars must be arrays and added via variable substitution. Calendars are specified using the `IN` operator.

Date ranges may be specified using the `BETWEEN`, greater than, less than, greater than or equal, and less than or equal operators. Use a subpredicate that tests equality to `null` to return tasks that do not have a completion or due date.

For example, the following code fragment creates a predicate that returns all incomplete tasks due before March 1, 2006.

```
dueDatePredicate = [NSPredicate predicateWithFormat:@"dueDate < %@"
   argumentArray:[NSArray arrayWithObject:[NSDate
dateWithNaturalLanguageString:@"12am March 1, 2006"]]];
completionDatePredicate = [NSPredicate predicateWithFormat:@"completedDate =
<null>"];
calendarPredicate = [NSPredicate predicateWithFormat:@"calendar IN %@))"
   argumentArray:[NSArray arrayWithObject:[CalCalendarStore calendars]]];
finalPredicate = [NSPredicate andPredicateWithSubpredicates:
   [NSArray arrayWithObjects:dueDatePredicate, completionDatePredicate,
calendarPredicate, nil]];
NSArray *tasks = [[CalCalendarStore defaultCalendarStore]
tasksWithPredicate:finalPredicate];
```

If you observe the `CalTasksChangedNotification` notification, you can retain the predicate and send it `evaluateWithObject:`, passing the changed task to determine whether the notification affects the task objects you previously fetched.

Availability
Available in Mac OS X v10.5 and later.

See Also
- taskWithUID: (page 46)

Related Sample Code
CalendarItems
Reminders

Declared In
CalCalendarStore.h

taskWithUID:

Returns a task that matches the specified unique identifier.

```
- (CalTask *)taskWithUID:(NSString *)uid
```

Parameters
uid
A unique identifier for a task.

Return Value
A task that matches the specified unique identifier.

Availability
Available in Mac OS X v10.5 and later.

See Also
- tasksWithPredicate: (page 45)

Declared In
CalCalendarStore.h

Constants

CalSpan

The range of events to apply changes to for a recurring event.

```
typedef enum {
    CalSpanThisEvent,
    CalSpanFutureEvents,
    CalSpanAllEvents
} CalSpan;
```

Constants
CalSpanThisEvent
Apply changes to just this instance of a recurring event.
Available in Mac OS X v10.5 and later.
Declared in CalCalendarStore.h.

`CalSpanFutureEvents`
Apply changes to all future events of a recurring event.

Available in Mac OS X v10.5 and later.

Declared in `CalCalendarStore.h`.

`CalSpanAllEvents`
Apply changes to all events of a recurring event.

Available in Mac OS X v10.5 and later.

Declared in `CalCalendarStore.h`.

Discussion
Use these constants when invoking the `saveEvent:span:error:` (page 43) and `removeEvent:span:error:` (page 42) methods.

Declared In
`CalendarStore/CalCalendarStore.h`

Changed Externally Notification Keys

Specifies the records that changed.

```
extern NSString * const CalInsertedRecordsKey;
extern NSString * const CalUpdatedRecordsKey;
extern NSString * const CalDeletedRecordsKey;
```

Constants
`CalInsertedRecordsKey`
An array of record UIDs that were inserted.

Available in Mac OS X v10.5 and later.

Declared in `CalCalendarStore.h`.

`CalUpdatedRecordsKey`
An array of record UIDs whose properties were changed.

Available in Mac OS X v10.5 and later.

Declared in `CalCalendarStore.h`.

`CalDeletedRecordsKey`
An array of record UIDs that were deleted.

Available in Mac OS X v10.5 and later.

Declared in `CalCalendarStore.h`.

Discussion
These keys are used in the user information dictionary of `CalCalendarStore` change notifications.

Declared In
`CalendarStore/CalNotificationCenter.h`

User Information Dictionary Keys

Keys in the user information dictionary of a notification sent by an instance of this class.

```
extern NSString * const CalSenderProcessIDKey;
extern NSString * const CalUserUIDKey;
```

Constants

`CalSenderProcessIDKey`

The key for a process ID that identifies the application that changed either a calendar, event, or task object.

This key is used in the user information dictionary for these notifications:

`CalCalendarsChangedNotification`
`CalEventsChangedNotification`
`CalTasksChangedNotification`
`CalCalendarsChangedExternallyNotification`
`CalEventsChangedExternallyNotification`
`CalTasksChangedExternallyNotification`

Available in Mac OS X v10.5 and later.

Declared in `CalCalendarStore.h`.

`CalUserUIDKey`

The key for a user UID that identifies the user that changed either a calendar, event, or task object.

This key is used in the user information dictionary for these notifications:

`CalCalendarsChangedNotification`
`CalEventsChangedNotification`
`CalTasksChangedNotification`
`CalCalendarsChangedExternallyNotification`
`CalEventsChangedExternallyNotification`
`CalTasksChangedExternallyNotification`

Available in Mac OS X v10.5 and later.

Declared in `CalCalendarStore.h`.

Declared In
`CalCalendarStore.h`

Notifications

CalCalendarsChangedNotification

Posted by the shared `NSNotificationCenter` object when this application process changes calendar objects. The notification object is the shared `CalCalendarStore` object. The user information dictionary contains the `CalSenderProcessIDKey` and `CalUserUIDKey` keys which identify the process and user that made the change to the calendars. In addition, the user information dictionary may contain one or more of the following keys which indicate the type of change: `CalInsertedRecordsKey`, `CalUpdatedRecordsKey`, and `CalDeletedRecordsKey`. The value for these keys is an array containing the UIDs for the events that were inserted, updated, or deleted. Use the `calendarWithUID:` (page 39) method to get the changed calendar object.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarStore.h`

CalEventsChangedNotification

Posted by the shared `NSNotificationCenter` object when this application process changes event objects. The notification object is the shared `CalCalendarStore` object. The user information dictionary contains the `CalSenderProcessIDKey` and `CalUserUIDKey` keys which identify the process and user that made the change to the calendars. In addition, the user information dictionary may contain one or more of the following keys which indicate the type of change: `CalInsertedRecordsKey`, `CalUpdatedRecordsKey`, and `CalDeletedRecordsKey`. The value for these keys is an array containing the UIDs for the events that were inserted, updated, or deleted. Use the `eventsWithPredicate:` (page 40) or the `eventWithUID:occurrence:` (page 41) method to get the changed event object. If the event is recurring, read Fetching Objects in *Calendar Store Programming Guide* for how to fetch recurring events within a date range.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarStore.h`

CalTasksChangedNotification

Posted by the shared `NSNotificationCenter` object when this application process changes task objects. The notification object is the shared `CalCalendarStore` object. The user information dictionary contains the `CalSenderProcessIDKey` and `CalUserUIDKey` keys which identify the process and user that made the change to the calendars. In addition, the user information dictionary may contain one or more of the following keys which indicate the type of change: `CalInsertedRecordsKey`, `CalUpdatedRecordsKey`, and `CalDeletedRecordsKey`. The value for these keys is an array containing the UIDS for the tasks that were inserted, updated, or deleted. Use the `tasksWithPredicate:` (page 45) or the `taskWithUID:` (page 46) method to get the changed task object.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarStore.h`

CalCalendarsChangedExternallyNotification

Posted by the shared `NSNotificationCenter` object when another application process changes calendar objects. The notification object is the shared `CalCalendarStore` object. The user information dictionary contains the `CalSenderProcessIDKey` and `CalUserUIDKey` keys which identify the process and user that made the change to the calendars. In addition, the user information dictionary may contain one or more of the following keys which indicate the type of change: `CalInsertedRecordsKey`, `CalUpdatedRecordsKey`, and `CalDeletedRecordsKey`. The value for these keys is an array containing the UIDs for the calendars that were inserted, updated, or deleted. Use the `calendarWithUID:` (page 39) method to get the changed calendar object.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarStore.h`

CalEventsChangedExternallyNotification

Posted by the shared `NSNotificationCenter` object when another application process changes event objects. This notification is identical to the `CalEventsChangedNotification` (page 49) notification except that it is triggered by an external process.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarStore.h`

CalTasksChangedExternallyNotification

Posted by the shared `NSNotificationCenter` object when another application process changes task objects. This notification is identical to the `CalTasksChangedNotification` (page 49) notification except that it is triggered by an external process.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalCalendarStore.h`

CalEvent Class Reference

Inherits from	CalCalendarItem : NSObject
Conforms to	NSCopying (CalCalendarItem) NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Declared in	CalendarStore/CalEvent.h
Availability	Available in Mac OS X v10.5 and later.
Companion guide	Calendar Store Programming Guide
Related sample code	CalendarItems Reminders SimpleCalendar

Overview

A `CalEvent` object represents an event added to a calendar in iCal.

Use the `event` (page 54) method to create a new event or use the `CalCalendarStore eventsWithPredicate:` (page 40) method to fetch existing events. This method takes an `NSPredicate` object as the argument so you can build your own queries. Use the `CalCalendarStore eventPredicateWithStartDate:endDate:calendars:` method to create an `NSPredicate` object for common queries that you can pass to the `eventsWithPredicate:` method.

Use the properties in this class to get information about an event. For example, the `attendees` property is an array of `CalAttendee` objects representing the people who are invited to this event. The `attendees` property is read-only. Use the `startDate` and `endDate` properties to access the start and end date and time for an event.

If you retain event objects, you can observe the `CalEventsChangedNotification` notification to update event objects when they change. Event objects can be added, changed, or deleted locally and externally after you fetch them.

Tasks

Creating Events

`+ event` (page 54)
Creates and returns a newly allocated `CalEvent` object.

Getting Properties

`isAllDay` (page 53) *property*
`YES` if this is an all day event; otherwise, `NO`.

`location` (page 53) *property*
A description of the location of this event.

`recurrenceRule` (page 54) *property*
The recurrence rule of an event.

`startDate` (page 54) *property*
The start date and time for this event.

`endDate` (page 53) *property*
The end date and time for this event.

`attendees` (page 52) *property*
An array of invited guests or an empty array if there are no attendees. This property is read-only. (read-only)

`isDetached` (page 53) *property*
Returns whether or not an event is detached. This property is read-only. (read-only)

`occurrence` (page 54) *property*
The occurrence date of an event. This property is read-only. (read-only)

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

attendees

An array of invited guests or an empty array if there are no attendees. This property is read-only. (read-only)

```
@property(readonly) NSArray *attendees
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalEvent.h`

endDate

The end date and time for this event.

```
@property(copy) NSDate *endDate
```

Discussion
An error occurs if you attempt to save an event whose start date occurs after the end date.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems

Declared In
`CalEvent.h`

isAllDay

`YES` if this is an all day event; otherwise, `NO`.

```
@property BOOL isAllDay
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalEvent.h`

isDetached

Returns whether or not an event is detached. This property is read-only. (read-only)

```
@property(readonly) BOOL isDetached
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalEvent.h`

location

A description of the location of this event.

```
@property(copy) NSString *location
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalEvent.h`

occurrence

The occurrence date of an event. This property is read-only. (read-only)

```
@property(readonly) NSDate *occurrence
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalEvent.h`

recurrenceRule

The recurrence rule of an event.

```
@property(copy) CalRecurrenceRule *recurrenceRule
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalEvent.h`

startDate

The start date and time for this event.

```
@property(copy) NSDate *startDate
```

Discussion
An error occurs if you attempt to save an event whose start date occurs after the end date.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems

Declared In
`CalEvent.h`

Class Methods

event

Creates and returns a newly allocated `CalEvent` object.

```
+ (id)event
```

Discussion
Use the `saveEvent:span:error:` (page 43) method to save a new event to the iCal database.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- saveEvent:span:error:` (page 43)

Related Sample Code
CalendarItems
Reminders

Declared In
`CalEvent.h`

CalNthWeekDay Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	CalendarStore/CalRecurrenceRule.h
Companion guide	Calendar Store Programming Guide

Overview

`CalNthWeekDay` objects are used to describe the `nthWeekDaysOfTheMonth` property of an `CalRecurrenceRule` object which specifies the nth instance of a particular day of the week—for example, the third Tuesday of every month. The properties defined in this class are read-only.

Tasks

Getting Properties

`dayOfTheWeek` (page 58) *property*
The day of the week, a number from `1` to `7` with Sunday equal to `1`.

`weekNumber` (page 58) *property*
The week of the month, a number that is either `1`, `2`, `3`, `4`, or `-1` where `-1` indicates the last week of the month.

Properties

For more about Objective-C properties, see “Properties” in *The Objective-C Programming Language*.

dayOfTheWeek

The day of the week, a number from `1` to `7` with Sunday equal to `1`.

```
@property(readonly) NSUInteger dayOfTheWeek;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

weekNumber

The week of the month, a number that is either `1`, `2`, `3`, `4`, or `-1` where `-1` indicates the last week of the month.

```
@property(readonly) NSInteger weekNumber;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

CalRecurrenceEnd Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Availability	Available in Mac OS X v10.5 and later.
Declared in	CalendarStore/CalRecurrenceRule.h
Companion guide	Calendar Store Programming Guide

Overview

`CalRecurrenceEnd` objects are used to describe a property of `CalRecurrenceRule` objects that defines how long a recurrence is scheduled to repeat.

You can create a recurrence end either using the `recurrenceEndWithEndDate:` (page 61) method specifying the date after which the event no longer repeats, or using the `recurrenceEndWithOccurrenceCount:` (page 61) method specifying the number of times it should repeat.

Tasks

Creating

`+ recurrenceEndWithEndDate:` (page 61)
Creates and returns a newly allocated `CalRecurrenceEnd` object with the specified end date.

`+ recurrenceEndWithOccurrenceCount:` (page 61)
Creates and returns a newly allocated `CalRecurrenceEnd` object with the specified recurrence count.

Getting Properties

`usesEndDate` (page 60) *property*
`YES` if the recurrence uses an end date; `NO` if it uses a occurrence count.

`endDate` (page 60) *property*
The end date if the recurrence uses an end date value; `nil` if it uses an occurrence count.

`occurrenceCount` (page 60) *property*
The occurrence count—number of times an event repeats—if the recurrence uses a number value; `0` if it uses an end date.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

endDate

The end date if the recurrence uses an end date value; `nil` if it uses an occurrence count.

```
@property(readonly) NSDate *endDate;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

occurrenceCount

The occurrence count—number of times an event repeats—if the recurrence uses a number value; `0` if it uses an end date.

```
@property(readonly) NSUInteger occurrenceCount;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

usesEndDate

`YES` if the recurrence uses an end date; `NO` if it uses a occurrence count.

```
@property(readonly) BOOL usesEndDate;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

Class Methods

recurrenceEndWithEndDate:

Creates and returns a newly allocated `CalRecurrenceEnd` object with the specified end date.

```
+ (id)recurrenceEndWithEndDate:(NSDate *)endDate
```

Parameters

endDate

The date a recurring event should end. Raises an exception if you pass `nil`.

Return Value

A newly allocated `CalRecurrenceEnd` object.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`CalRecurrenceRule.h`

recurrenceEndWithOccurrenceCount:

Creates and returns a newly allocated `CalRecurrenceEnd` object with the specified recurrence count.

```
+ (id)recurrenceEndWithOccurrenceCount:(NSUInteger)occurrenceCount
```

Parameters

occurrenceCount

The number of times a recurring event should repeat. Raises an exception if you pass `0` or a negative number.

Return Value

A newly allocated `CalRecurrenceEnd` object.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`CalRecurrenceRule.h`

CalRecurrenceRule Class Reference

Inherits from	NSObject
Conforms to	NSCopying NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Declared in	CalendarStore/CalRecurrenceRule.h
Availability	Available in Mac OS X v10.5 and later.
Companion guide	Calendar Store Programming Guide

Overview

The `CalRecurrenceRule` class is used to describe the recurrence pattern for a recurring event. The recurrence rules that you can create are restricted to the recurrence patterns that you have access to in iCal. It is not possible to directly modify a `CalRecurrenceRule` object or any of its propertiesâ properties defined in this class are read-only.

Use one of the `init...` methods to create the desired recurrence rule. These `init...` methods set the receiverâ properties. For example, if you invoke one of the `initMonthly...` methods, the `recurrenceType` property is set to `CalRecurrenceMonthly` (page 74). All recurrence rules have a value set for the `recurrenceType` and `recurrenceInterval` properties. If the `recurrenceEnd` property is `nil`, an event repeats forever. The rest of the properties defined in this class have values depending on the type of recurrence rule and initializer method used to create the recurring event pattern. If a property is not needed to describe a recurrence pattern, then itâ value is `nil`.

Every initializer method has an interval and a `CalRecurrenceEnd` object as arguments. The interval is a value greater than `0` that is the number of units between occurrences of an event. For example, if the `recurrenceType` property is `CalRecurrenceMonthly` (page 74), then the unit of measurement is one month. If the interval is `1`, the event occurs every month, but if it is `2`, it occurs every other month. The `CalRecurrenceEnd` object specifies when a recurring event ends. This can be specified using a counter or an end date (see *CalRecurrenceEnd Class Reference* for details). Pass `nil` as the recurrence end argument if the event never ends.

In addition, there are some initializers for more sophisticated recurrence patterns. However, these initializers are restricted to patterns that can be represented in iCal and include custom repeating patterns. For example, you can specify a pattern where an event occurs on the Friday of the 2nd week of every month, or the Monday of the 3rd week of every fourth month of a year. You can create custom patterns for weekly, monthly, and yearly recurrence rules.

After you create a recurrence rule, use the `recurrenceRule` (page 54) `CalEvent` property to set the recurrence rule for an event. Use the `saveEvent:span:error:` (page 43) method to save your changes to the iCal database.

Tasks

Initializing Recurrence Rules

`- initDailyRecurrenceWithInterval:end:` (page 67)
Initializes and returns a daily recurrence rule with the specified interval and ending rule.

`- initWeeklyRecurrenceWithInterval:end:` (page 70)
Initializes and returns a weekly recurrence rule with the specified interval and ending rule.

`- initWeeklyRecurrenceWithInterval:forDaysOfTheWeek:end:` (page 70)
Initializes and returns a weekly recurrence rule with the specified interval, ending rule, and specific days of the week.

`- initMonthlyRecurrenceWithInterval:end:` (page 68)
Initializes and returns a monthly recurrence rule with the specified interval and end rule.

`- initMonthlyRecurrenceWithInterval:forDaysOfTheMonth:end:` (page 69)
Initializes and returns a monthly recurrence rule that represents an event that occurs more than once a month in a monthly patternâ pattern repeats at the specified interval.

`- initMonthlyRecurrenceWithInterval:forDayOfTheWeek:forWeekOfTheMonth:end:` (page 68)
Initializes and returns a monthly recurrence rule that represents an event that occurs on a specific day of the week and week of the month patternâ pattern repeats at the specified monthly interval.

`- initYearlyRecurrenceWithInterval:end:` (page 71)
Initializes and returns a yearly recurrence rule with the specified yearly interval and end rule.

`- initYearlyRecurrenceWithInterval:forMonthsOfTheYear:end:` (page 72)
Initializes and returns a yearly recurrence rule representing an event that occurs multiple months within a year at a specified yearly interval.

`- initYearlyRecurrenceWithInterval:forDayOfTheWeek:forWeekOfTheMonth:forMonthsOfTheYear:end:` (page 72)
Initializes and returns a yearly recurrence rule that represents an event that has a weekly and monthly pattern that repeats at the specified yearly interval.

Getting Recurrence Properties

`firstDayOfTheWeek` (page 66) *property*
An integer value of `0` or `1` to `7` where `0` indicates no value is set, and `1` to `7` indicates the first day of the week where Sunday is represented by `1`. (read-only)

`recurrenceEnd` (page 66) *property*
An object that describes how a recurring event ends by specifying an end date or a counter. (read-only)

`recurrenceType` (page 67) *property*
The unit of time between intervals. See â `CalRecurrenceTypeâ` (page 73) for possible values. (read-only)

`recurrenceInterval` (page 67) *property*
The number of intervals between the specified pattern of a recurring event. The actual time between a pattern depends on the value of the `recurrenceType` property. (read-only)

`daysOfTheWeek` (page 65) *property*
If the `recurrenceType` property is `CalRecurrenceWeekly`, an integer value ranging from `1` to `7` representing the day of the week in the pattern where Sunday is equal to `1`; otherwise, `nil`. (read-only)

`daysOfTheMonth` (page 65) *property*
If the `recurrenceType` property is `CalRecurrenceMonthly`, an array of integer values ranging from `1` to `31` representing days of the month in the pattern; otherwise, `nil`. (read-only)

`nthWeekDaysOfTheMonth` (page 66) *property*
If the `recurrenceType` property is `CalRecurrenceMonthly` or `CalRecurrenceYearly`, an array of `CalNthWeekDay` objects representing the days within the weeks of a month; otherwise, `nil`. (read-only)

`monthsOfTheYear` (page 66) *property*
If the `recurrenceType` property is `CalRecurrenceYearly`, an array of the months' integer values ranging from `1` to `12` representing the month of a year in the event occurs; otherwise, `nil`. (read-only)

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

daysOfTheMonth

If the `recurrenceType` property is `CalRecurrenceMonthly`, an array of integer values ranging from `1` to `31` representing days of the month in the pattern; otherwise, `nil`. (read-only)

```
@property(readonly) NSArray *daysOfTheMonth
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

daysOfTheWeek

If the `recurrenceType` property is `CalRecurrenceWeekly`, an integer value ranging from `1` to `7` representing the day of the week in the pattern where Sunday is equal to `1`; otherwise, `nil`. (read-only)

```
@property(readonly) NSArray *daysOfTheWeek
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

firstDayOfTheWeek

An integer value of `0` or `1` to `7` where `0` indicates no value is set, and `1` to `7` indicates the first day of the week where Sunday is represented by `1`. (read-only)

```
@property(readonly) NSUInteger firstDayOfTheWeek
```

Discussion
This property only affects the way the recurrence is expanded for weekly recurrence rules with an interval greater than `1`. For those types of recurrence rules, Calendar Store sets this property to `2` (Monday). For all other recurrence rules, this property defaults to `0`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

monthsOfTheYear

If the `recurrenceType` property is `CalRecurrenceYearly`, an array of the months' integer values ranging from `1` to `12` representing the month of a year in the event occurs; otherwise, `nil`. (read-only)

```
@property(readonly) NSArray *monthsOfTheYear
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

nthWeekDaysOfTheMonth

If the `recurrenceType` property is `CalRecurrenceMonthly` or `CalRecurrenceYearly`, an array of `CalNthWeekDay` objects representing the days within the weeks of a month; otherwise, `nil`. (read-only)

```
@property(readonly) NSArray *nthWeekDaysOfTheMonth
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

recurrenceEnd

An object that describes how a recurring event ends by specifying an end date or a counter. (read-only)

```
@property(readonly) CalRecurrenceEnd *recurrenceEnd
```

Discussion
If `nil`, the event never ends.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

recurrenceInterval

The number of intervals between the specified pattern of a recurring event. The actual time between a pattern depends on the value of the `recurrenceType` property. (read-only)

```
@property(readonly) NSUInteger recurrenceInterval
```

Discussion
For example, if the `recurrenceType` property is `CalRecurrenceMonthly` and the `recurrenceInterval` property is `1`, then the event occurs every month. If the `recurrenceInterval` property is `2`, then the event occurs every other month.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

recurrenceType

The unit of time between intervals. See `â CalRecurrenceTypeâ` (page 73) for possible values. (read-only)

```
@property(readonly) CalRecurrenceType recurrenceType
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalRecurrenceRule.h`

Instance Methods

initDailyRecurrenceWithInterval:end:

Initializes and returns a daily recurrence rule with the specified interval and ending rule.

```
- (id)initDailyRecurrenceWithInterval:(NSUInteger)interval end:(CalRecurrenceEnd
    *)end
```

Parameters

interval

The interval in days between occurrences of a recurring event. Must be a value greater than `0`.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value

An initialized daily recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceDaily`. Returns `nil` if *interval* is `0` or a negative number.

Availability

Available in Mac OS X v10.5 and later.

Declared In

`CalRecurrenceRule.h`

initMonthlyRecurrenceWithInterval:end:

Initializes and returns a monthly recurrence rule with the specified interval and end rule.

```
- (id)initMonthlyRecurrenceWithInterval:(NSUInteger)interval end:(CalRecurrenceEnd
    *)end
```

Parameters

interval

The interval in months between occurrences of a recurring event. Must be a value greater than `0`.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value

A newly initialized monthly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceMonthly`. Returns `nil` if *interval* is `0` or a negative number.

Discussion

The returned recurrence rule defaults to the first day of the month.

Availability

Available in Mac OS X v10.5 and later.

See Also

- `initMonthlyRecurrenceWithInterval:forDaysOfTheMonth:end:` (page 69)
- `initMonthlyRecurrenceWithInterval:forDayOfWeek:forWeekOfMonth:end:` (page 68)

Declared In

`CalRecurrenceRule.h`

initMonthlyRecurrenceWithInterval:forDayOfTheWeek:forWeekOfTheMonth:end:

Initializes and returns a monthly recurrence rule that represents an event that occurs on a specific day of the week and week of the month patternâ pattern repeats at the specified monthly interval.

```
- (id)initMonthlyRecurrenceWithInterval:(NSUInteger)interval
    forDayOfTheWeek:(NSUInteger)weekDay forWeekOfTheMonth:(NSInteger)monthWeek
    end:(CalRecurrenceEnd *)end
```

Parameters

interval

The interval in months between occurrences of a recurring event. Must be a value greater than `0`.

weekDay

The day of the week that the event occurs. An integer value ranging from `1` to `7` representing the day of the week where Sunday is equal to `1`.

monthWeek

The week of the month that the event occurs. An integer value that is `1`, `2`, `3`, `4`, or `-1` representing the week of a month where `-1` is the last week of the month.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value

A newly initialized monthly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceMonthly`. Returns `nil` if *interval* is `0` or a negative number.

Discussion

For example, use this method to create a recurrence rule that represents an event that occurs on the first Monday of every month.

Availability

Available in Mac OS X v10.5 and later.

See Also

- `initMonthlyRecurrenceWithInterval:end:` (page 68)
- `initMonthlyRecurrenceWithInterval:forDaysOfTheMonth:end:` (page 69)

Declared In

`CalRecurrenceRule.h`

initMonthlyRecurrenceWithInterval:forDaysOfTheMonth:end:

Initializes and returns a monthly recurrence rule that represents an event that occurs more than once a month in a monthly patternâ pattern repeats at the specified interval.

```
- (id)initMonthlyRecurrenceWithInterval:(NSUInteger)interval
    forDaysOfTheMonth:(NSArray *)monthDays end:(CalRecurrenceEnd *)end
```

Parameters

interval

The interval in months between occurrences of a recurring event. Must be a value greater than `0`.

monthDays

An array of numbers specifying the days of the month pattern that repeats. The integer values can range from `1` to `31` representing the days of the month.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value
A newly initialized monthly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceMonthly`. Returns `nil` if *`interval`* is `0` or a negative number.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `initMonthlyRecurrenceWithInterval:end:` (page 68)
- `initMonthlyRecurrenceWithInterval:forDayOfWeek:forWeekOfMonth:end:` (page 68)

Declared In
`CalRecurrenceRule.h`

initWeeklyRecurrenceWithInterval:end:

Initializes and returns a weekly recurrence rule with the specified interval and ending rule.

```
- (id)initWeeklyRecurrenceWithInterval:(NSUInteger)interval end:(CalRecurrenceEnd
    *)end
```

Parameters

`interval`

The interval in weeks between occurrences of a recurring event. Must be a value greater than `0`.

`end`

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value
An initialized weekly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceWeekly`. Returns `nil` if *`interval`* is `0` or a negative number.

Availability
Available in Mac OS X v10.5 and later.

See Also
- `initWeeklyRecurrenceWithInterval:forDaysOfTheWeek:end:` (page 70)

Declared In
`CalRecurrenceRule.h`

initWeeklyRecurrenceWithInterval:forDaysOfTheWeek:end:

Initializes and returns a weekly recurrence rule with the specified interval, ending rule, and specific days of the week.

```
- (id)initWeeklyRecurrenceWithInterval:(NSUInteger)interval forDaysOfTheWeek:(NSArray
    *)days end:(CalRecurrenceEnd *)end
```

Parameters

`interval`

The interval in weeks between occurrences of a recurring event. Must be a value greater than `0`.

days

An array of numbers specifying the weekly pattern that repeats. The integer values can range from `1-7` where Sunday is equal to `1`.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value
A newly initialized weekly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceWeekly`. Returns `nil` if *interval* is `0` or a negative number, or any of the other arguments are invalid.

Discussion
This initializer allows you to specify a weekly pattern that repeats. For example, if *days* contains the numbers `2` and `4`, and interval is `2`, then an event occurs biweekly on Mondays and Wednesdays.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- initWeeklyRecurrenceWithInterval:end:` (page 70)

Declared In
`CalRecurrenceRule.h`

initYearlyRecurrenceWithInterval:end:

Initializes and returns a yearly recurrence rule with the specified yearly interval and end rule.

```
- (id)initYearlyRecurrenceWithInterval:(NSUInteger)interval end:(CalRecurrenceEnd
    *)end
```

Parameters

interval

The interval in years between occurrences of a recurring event. Must be a value greater than `0`.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value
A newly initialized yearly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceYearly`. Returns `nil` if *interval* is `0` or a negative number.

Availability
Available in Mac OS X v10.5 and later.

See Also
`- initYearlyRecurrenceWithInterval:forMonthsOfTheYear:end:` (page 72)
`- initYearlyRecurrenceWithInterval:forDayOfWeek:forWeekOfMonth:forMonthsOfTheYear:end:` (page 72)

Declared In
`CalRecurrenceRule.h`

initYearlyRecurrenceWithInterval:forDayOfTheWeek:forWeekOfTheMonth:forMonthsOfTheYear:end:

Initializes and returns a yearly recurrence rule that represents an event that has a weekly and monthly pattern that repeats at the specified yearly interval.

```
- (id)initYearlyRecurrenceWithInterval:(NSUInteger)interval
    forDayOfTheWeek:(NSUInteger)weekDay forWeekOfTheMonth:(NSInteger)monthWeek
    forMonthsOfTheYear:(NSArray *)months end:(CalRecurrenceEnd *)end
```

Parameters

interval

The interval in years between this pattern of a recurring event. Must be a value greater than `0`.

weekDay

The day of the week that the event occurs. An integer value ranging from `1-7` representing the day of the week where Sunday is equal to `1`.

monthWeek

The week of the month that the event occurs. An integer value that is either `1`, `2`, `3`, `4`, or `-1` representing the week of a month where `-1` is the last week of the month.

months

The months of the year that the event occurs. An array of numbers where each number is an integer value ranging from `1-12` representing the month of a year.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value

A newly initialized yearly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceYearly`. Returns `nil` if *interval* is `0` or a negative number.

Discussion

This method allows you to create a pattern for an entire year that repeatsâ where event occurs on the same day of the week, in the same week of a month, and possibly more than one month of a year. For example, use this method to represent an event that occurs every year on the first Friday of the sixth and twelfth months.

Availability

Available in Mac OS X v10.5 and later.

See Also

- `initYearlyRecurrenceWithInterval:end:` (page 71)
- `initYearlyRecurrenceWithInterval:forMonthsOfTheYear:end:` (page 72)

Declared In

`CalRecurrenceRule.h`

initYearlyRecurrenceWithInterval:forMonthsOfTheYear:end:

Initializes and returns a yearly recurrence rule representing an event that occurs multiple months within a year at a specified yearly interval.

```
- (id)initYearlyRecurrenceWithInterval:(NSUInteger)interval
    forMonthsOfTheYear:(NSArray *)months end:(CalRecurrenceEnd *)end
```

Parameters

interval

The interval in years between occurrences of a recurring event. Must be a value greater than `0`.

end

Describes how a recurring event ends. Pass `nil` if the event never ends.

Return Value

A newly initialized yearly recurrence rule objectâ with `recurrenceType` property set to `CalRecurrenceYearly`. Returns `nil` if *interval* is `0` or a negative number.

Discussion

For example, use this method if you want to represent an event that occurs every year on the first day of the first and seventh month of the year. If you need to specify the day of the month that the event occurs, use the `initYearlyRecurrenceWithInterval:forDayOfWeek:forWeekOfMonth:forMonthsOfTheYear:end:` (page 72) method.

Availability

Available in Mac OS X v10.5 and later.

See Also

`- initYearlyRecurrenceWithInterval:end:` (page 71)

`- initYearlyRecurrenceWithInterval:forDayOfWeek:forWeekOfMonth:forMonthsOfTheYear:end:` (page 72)

Declared In

`CalRecurrenceRule.h`

Constants

Recurrence Rule Constants

Constants used by this class.

```
extern NSUInteger const CalDefaultRecurrenceInterval;
```

Constants

`CalDefaultRecurrenceInterval`

The default recurrence interval. The default value of `1` indicates that the event repeats daily, weekly, monthly, or yearly depending on the recurrence type.

Available in Mac OS X v10.5 and later.

Declared in `CalRecurrenceRule.h`.

Declared In

`CalendarStore/CalRecurrenceRule.h`

CalRecurrenceType

The unit of measurement between occurrences of a recurring event.

```
typedef enum {
    CalRecurrenceDaily,
    CalRecurrenceWeekly,
    CalRecurrenceMonthly,
    CalRecurrenceYearly
} CalRecurrenceType;
```

Constants

`CalRecurrenceDaily`

Indicates a daily unit of measurement.

Available in Mac OS X v10.5 and later.

Declared in `CalRecurrenceRule.h`.

`CalRecurrenceWeekly`

Indicates a weekly unit of measurement.

Available in Mac OS X v10.5 and later.

Declared in `CalRecurrenceRule.h`.

`CalRecurrenceMonthly`

Indicates a monthly unit of measurement.

Available in Mac OS X v10.5 and later.

Declared in `CalRecurrenceRule.h`.

`CalRecurrenceYearly`

Indicates a yearly unit of measurement.

Available in Mac OS X v10.5 and later.

Declared in `CalRecurrenceRule.h`.

Discussion

The four types of units are days, weeks, months, and years which correspond to the types of recurrence rules. For example, if the recurrence rule is `CalRecurrenceWeekly`, then an interval of `1` indicates the event repeats weekly. If the interval is `2`, the event repeats biweekly.

Declared In

`CalendarStore/CalRecurrenceRule.h`

CalTask Class Reference

Inherits from	CalCalendarItem : NSObject
Conforms to	NSCopying (CalCalendarItem) NSObject (NSObject)
Framework	/System/Library/Frameworks/CalendarStore.framework
Declared in	CalendarStore/CalTask.h
Availability	Available in Mac OS X v10.5 and later.
Companion guide	Calendar Store Programming Guide
Related sample code	CalendarItems Reminders

Overview

A `CalTask` object represents a task added to a calendar in iCal.

You can create a new task using the `task` (page 78) method or get existing tasks using the `tasksWithPredicate:` (page 45) `CalCalendarStore` method. If you create a new task then you need to set the inherited calendar property before saving the task using the `saveTask:error:` (page 44) `CalCalendarStore` method.

The `tasksWithPredicate:` `CalCalendarStore` method takes an `NSPredicate` object as the argument so you can build your own queries. Use the `taskPredicateWithUncompletedTasksDueBefore:calendars:` (page 38) and `taskPredicateWithTasksCompletedSince:calendars:` (page 37) `CalCalendarStore` methods to create `NSPredicate` objects for common queries that you can pass to the `tasksWithPredicate:` method.

Use the properties in this class to set and get information about a task. For example, use the `dueDate` (page 76) property to get the due date of a task and the `isCompleted` (page 77) property to determine if the task is done. The values of the `completedDate` (page 76)and `isCompleted` (page 77) properties are interdependent. Read the property descriptions to learn more.

If you retain task objects, you can observe the `CalTasksChangedNotification` notification to update task objects when they change. Task objects can be added, changed, or deleted locally or externally after you fetch them.

Tasks

Creating and Initializing Tasks

`+ task` (page 78)
Creates and initializes a newly allocated task object.

Getting Properties

`dueDate` (page 76) *property*
The due date and time for this task.

`priority` (page 77) *property*
The priority of this task—an integer ranging from `0` to `9` with `0` representing an undefined priority, `1` the highest priority, and `9` the lowest priority.

`isCompleted` (page 77) *property*
`YES` if this task is completed; otherwise, `NO`.

`completedDate` (page 76) *property*
The task's completed date.

Properties

For more about Objective-C properties, see "Properties" in *The Objective-C Programming Language*.

completedDate

The task's completed date.

```
@property(copy) NSDate *completedDate;
```

Discussion
If you set `completedDate` to `nil`, then `isCompleted` (page 77) is set to `NO`. If you set `completedDate` to a valid date, then `isCompleted` is set to `YES`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalTask.h`

dueDate

The due date and time for this task.

```
@property(copy) NSDate *dueDate;
```

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems
Reminders

Declared In
`CalTask.h`

isCompleted

`YES` if this task is completed; otherwise, `NO`.

```
@property BOOL isCompleted;
```

Discussion
If you set `isCompleted` to `YES`, then `completedDate` (page 76) is set to the current date. If you set `isCompleted` to `NO`, then `completedDate` is set to `nil`.

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalTask.h`

priority

The priority of this task—an integer ranging from `0` to `9` with `0` representing an undefined priority, `1` the highest priority, and `9` the lowest priority.

```
@property CalPriority priority;
```

Discussion
Typically, you use one of the constants described in "`Task Priority Constants`" (page 78) to set this property. However, any integer between `0` and `9` are valid values.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems
Reminders

Declared In
`CalTask.h`

Class Methods

task

Creates and initializes a newly allocated task object.

```
+ (id)task
```

Return Value
A newly created task object.

Availability
Available in Mac OS X v10.5 and later.

Related Sample Code
CalendarItems
Reminders

Declared In
`CalTask.h`

Constants

CalPriority

Type that describes the priority of a task.

```
typedef NSUInteger CalPriority;
```

Availability
Available in Mac OS X v10.5 and later.

Declared In
`CalTask.h`

Task Priority Constants

The priority of a task.

```
enum {
    CalPriorityNone     = 0,
    CalPriorityHigh     = 1,
    CalPriorityMedium   = 5,
    CalPriorityLow      = 9
};
```

Constants

`CalPriorityNone`

The priority is not set for this task.

Available in Mac OS X v10.5 and later.

Declared in `CalTask.h`.

`CalPriorityHigh`

The priority for this task is high.

Available in Mac OS X v10.5 and later.

Declared in `CalTask.h`.

`CalPriorityMedium`

The priority for this task is medium.

Available in Mac OS X v10.5 and later.

Declared in `CalTask.h`.

`CalPriorityLow`

The priority for this task is low.

Available in Mac OS X v10.5 and later.

Declared in `CalTask.h`.

Discussion

These constants can be used to set the `priority` property.

Declared In

`CalTask.h`

Constants

Calendar Store Constants Reference

Framework: CalendarStore/CalendarStore.h

Overview

Constants

Calendar Store Error Domain

The domain for errors created by the Calendar Store framework.

```
extern NSString *const CalCalendarStoreErrorDomain;
```

Constants

`CalCalendarStoreErrorDomain`

The Calendar Store error domain.

Available in Mac OS X v10.5 and later.

Declared in `CalendarStoreErrors.h`.

Declared In

`CalendarStore/CalendarStoreErrors.h`

Calendar Store Errors

The Calendar Store errors that might occur when modifying and saving objects.

```
enum {
    CalCalendarNotEditableError = 1025,
    CalDateInvalidError = 1026,
    CalCalendarNotInRepository = 1027,
    CalCalendarTitleNotUniqueError = 1028
};
```

Constants

`CalCalendarNotEditableError`

Attempted to add events or tasks to a read-only calendar.

Available in Mac OS X v10.5 and later.

Declared in `CalendarStoreErrors.h`.

`CalDateInvalidError`

Attempted to set the start date of an event earlier than its end date.

Available in Mac OS X v10.5 and later.

Declared in `CalendarStoreErrors.h`.

`CalCalendarNotInRepository`

Attempted to set an event or task's calendar object to a nonlocal calendar—a calendar that is not in the user's iCal database.

Available in Mac OS X v10.5 and later.

Declared in `CalendarStoreErrors.h`.

`CalCalendarTitleNotUniqueError`

Attempted to save a calendar object that doesn't have a unique title. All calendar titles must be unique.

Available in Mac OS X v10.5 and later.

Declared in `CalendarStoreErrors.h`.

Declared In

`CalendarStore/CalendarStoreErrors.h`

Document Revision History

This table describes the changes to *Calendar Store Framework Reference*.

Date	Notes
2009-07-28	Added concurrency information.
2007-07-08	New document that describes the Calendar Store framework classes and methods used to access iCal data.

Document Revision History

Index

A

C

D

E

F

H

I

L

M

N

O

P

R

S

T

U

W