NSCalendarDate

Inherits From: NSDate

Declared In: foundation/NSDate.h

Class Description

NSCalendarDate is a public subclass of NSDate that creates concrete date objects with time zones and format strings bound to them. These objects are especially suited for representing and manipulating dates according to western calendar systems.

By drawing on the behavior of the NSTimeZone class, objects of NSCalendarDate adjust their visible representations to reflect their associated time zones. Because of this, you can track an NSCalendarDate object across different time zones. You can also present date information from time-zone viewpoints other than the one for the current locale.

Each NSCalendarDate object also has a calendar format string bound to it. This format string contains date-conversion specifiers that are very similar to those used in **strftime()**. By reference to this format string, NSCalendarDate can interpret dates that are represented as strings conforming to the format. Several methods allow you to specify formats other than the one bound to the object, and **setCalendarFormat:** lets you change the default format string for an NSCalendarDate object.

NSCalendarDate provides both class (**calendarDate, date...**) and instance methods for obtaining initialized objects. Some of these methods allow you to initialize dates from strings while others initialize objects from sets of integers corresponding the standard time values (months, hours, seconds, etc.). As always, you are responsible for deallocating any objects obtained through an **init...** method.

To retrieve conventional elements of a date object, use the **...Of...** methods. For example, **dayOfWeek** returns a number that indicates the day of the week (0 is Sunday). The **monthOfYear** method returns a number between 1 and 12 that indicates the month.

NSCalendarDate provides several methods for representing dates as strings. These methods — **description**, **descriptionWithCalendarFormat**:, and **descriptionWithCalendarFormat:timeZone:** — take an implicit or explicit format string.

NSCalendarDate performs date computations based on western calendar systems, primarily the Gregorian. (The algorithms are derived from public domain software

described in “Calendrical Calculations,” a two-part series by Nachum Dershowitz and Edward M Reingold in *Software — Practice and Experience*).

Instance Variables

None declared in this class.

Method Types

Getting an NSCalendarDate object	+ dateWithString:calendarFormat: + dateWithYear:month:day:hour:minute:second: timeZone: – initWithString: – initWithString:calendarFormat: – initWithYear:month:day:hour:minute:second:timeZone:
Retrieving date elements	– dayOfMonth – dayOfWeek – dayOfYear – hourOfDay – minuteOfHour – monthOfYear – secondOfMinute – yearOfCommonEra
Providing adjusted date	– addYear:month:day:hour:minute:second:
Getting string descriptions of dates	– description – descriptionWithCalendarFormat: – descriptionWithCalendarFormat:timeZone:
Getting and setting calendar formats	– calendarFormat – setCalendarFormat:
Getting and setting time zones	– setTimeZone: – timeZoneDetail

Class Methods

dateWithString:calendarFormat:

+ **dateWithString**:(NSString *)*description* **calendarFormat**:(NSString *)*format*

Returns an NSCalendarDate object that is initialized with the date specified in the string *description*. NSCalendarDate interprets this string through *format*, which consists of conversion specifiers similar to those used in **strftime()**. The date conversion specifiers cover a range of date conventions:

%%	a '%' character
%a	abbreviated weekday name
%A	full weekday name
%b	abbreviated month name
%B	full month name
%c	shorthand for %X %x, the locale format for date and time
%d	day of the month as a decimal number (01-31)
%H	hour based on a 24-hour clock as a decimal number (00-23)
%I	hour based on a 12-hour clock as a decimal number (01-12)
%j	day of the year as a decimal number (001-366)
%m	month as a decimal number (01-12)
%M	minute as a decimal number (00-59)
%p	AM/PM designation associated with a 12-hour clock
%S	second as a decimal number (00-61)
%w	weekday as a decimal number (0-6), where Sunday is 0
%x	date using the date representation for the locale
%X	time using the time representation for the locale
%y	year without century (00-99)
%Y	year with century (e.g. 1990)
%Z	time zone name
%z	time zone offset in hours and minutes from GMT (HHMM)

As an example, let's say your company's convention for dates on correspondence takes the form "Friday, 1 July 1994, 11:45 AM" To get an NSCalendarDate object with an temporal value corresponding to this string, you would code something like the following:

```
id today = [NSCalendarDate dateWithString:@"Friday, 1 July 1994,
           11:45 AM" calendarFormat:@"%A, %d %B %Y, %I:%M %p"];
```

Note that this method does not set the default calendar format for the NSCalendar object. To set the default format, use **setCalendarFormat:**.

See also: – **initWithString:calendarFormat:, setCalendarFormat:**

dateWithYear:month:day:hour:minute:second:timeZone:

+ **dateWithYear:**(int)*year*
month:(unsigned int)*month*
day:(unsigned int)*day*
hour:(unsigned int)*hour*
minute:(unsigned int)*minute*
second:(unsigned int)*second*
timeZone*:*(id <NSTimeZone>)*timeZone*

Returns an NSCalendarDate object that is initialized with the year, month, day, hour and second offsets, as well as the time-zone object that are specified as arguments. The offsets can be positive (future) or negative (past), except for years, which cannot be negative. The *year* value must include the century (for example, 1995 instead of 95). The other values are the standard ones: 1 through 12 for months, 1 through 31 for days, 0 through 23 for hours and 0 through 59 for both minutes and seconds.

The method verifies the time zone supplied as an argument and can substitute an alternative time zone. If the method does supply a new time zone, it applies the difference in offsets-from-GMT values between the substituted and the original time zone to the date object being created.

The following code fragment shows a date object created with a date on the fourth of July, 7 PM, Eastern Standard Time (**timeZoneForKey:** gets a time zone from a local collection of time zones, based on a key).

```
id fireWorks = [NSCalendarDate dateWithYear:1994 month:7 day:4
              hour:19 minute:0 second:0
              timeZone:[NSTimeZone timeZoneForKey:@"EST"]];
```

See also: – **initWithYear:month:day:hour:minute:second:timeZone:**

Instance Methods

addYear:month:day:hour:minute:second:

– (NSCalendarDate*)**addYear:**(int)*year*
month:(int)*month*
day:(int)*day*
hour:(int)*hour*
minute:(int)*minute*
second:(int)*second*

Returns an NSCalendarDate object that is updated with the year, month, day, hour and second offsets specified as arguments. The offsets can be positive (future) or negative (past). This method extracts and modifies the unit components rather than computing the vector as seconds. This approach preserves "clock time" across changes in Daylight Savings Time zones and leap years. (For example, adding one month to a date object with a time of 12 noon correctly maintains time at 12 noon.)

The following code fragment shows a date object created with a date a week later than an existing date object.

```
NSCalendarDate *now = [NSDate date];
NSCalendarDate *nextWeek = [now addYear:0 month:0 day:7 hour:0
                            minute:0 second:0];
```

calendarFormat

– (NSString *)**calendarFormat**

Returns the calendar format for the NSCalendar object. If you do not specify with **setCalendarFormat:** a format to be used as the default when invoking the **description** method, NSCalendarDate substitutes its own default: an international format of "%Y-%m-%d %H:%M:%S %z" (for example, 1994-01-14 16:45:12 +0900). See the description of the class method **dateWithString:calendarFormat:** for a discussion of date conversion specifiers.

dayOfMonth

– (int)**dayOfMonth**

Returns a number that indicates the day of the month (1 through 31) of the receiving object.

See also: – **dayOfYear**; – **dayOfWeek**; – **hourOfDay**; – **hourOfDay**; – **minuteOfHour**; – **monthOfYear**; – **secondOfMinute**; – **yearOfCommonEra**

dayOfWeek

– (int)**dayOfWeek**

Returns a number that indicates the day of the week (0 through 6) of the receiving object; 0 indicates Sunday.

See also: – **dayOfMonth**; – **dayOfYear**; – **hourOfDay**;; – **minuteOfHour**; – **monthOfYear**; – **secondOfMinute**; – **yearOfCommonEra**

dayOfYear

– (int)**dayOfYear**

Returns a number that indicates the day of the year (1 through 366) of the receiving object.

See also: – **dayOfMonth**; – **dayOfWeek**; – **hourOfDay**; – **minuteOfHour**; – **monthOfYear**; – **secondOfMinute**; – **yearOfCommonEra**

description

– (NSString*)**description**

Returns the date as a string, represented according to the default format string for the object. The system default for the calendar format string is "%Y-%m-%d %H:%M:%S %z" (for example, 1994-01-14 16:45:12 +0900). You can override this and set your own default format string by invoking the **setCalendarFormat:** method. See the description of the class method **dateWithString:calendarFormat:** for a listing of conversion specifiers for dates.

See also: – **calendarFormat**, – **descriptionWithCalendarFormat:**, – **descriptionWithCalendarFormat:timeZone:**

descriptionWithCalendarFormat:

– (NSString *)**descriptionWithCalendarFormat:**(NSString *)*format***;**

Returns a string representation of the NSCalendarDate object that is formatted as specified by the conversion specifiers in the format string *format*. The default time zone for the locale is assumed. The conversion specifiers cover a range of date conventions. See the description of the class method **dateWithString:calendarFormat:** for a listing of these specifiers.

This example gets the current date, formats it in the form "Tues 3/1/94 3:30 PM" and displays the date string in a text field:

```
NSCalendarDate *now = [NSDate date];
NSString *datestr =
```

```
            [now descriptionWithCalendarFormat:@"%a %m/%d/%y %I:%M %p"];
[dateField setStringValue:[datestr cString]];
```

See also: – **calendarFormat**, – **description**,
– **descriptionWithCalendarFormat:timeZone:**

descriptionWithCalendarFormat:timeZone

– (NSString *)**descriptionWithCalendarFormat:**(NSString *)*format*
timeZone:(NSTimezone *)*timeZone*

Returns a string representation of the NSCalendarDate object that is formatted as specified by the conversion specifiers in the format string *format*. Specify the time zone for the date in *timeZone* (this should be different from the default time zone). The conversion specifiers cover a range of date conventions. See the description of the class method **dateWithString:calendarFormat:** for a listing of these specifiers.

See also: **– calendarFormat**, – **description, – descriptionWithCalendarFormat:**, –s**etCalendarFormat:**

hourOfDay

– (int)**hourOfDay**

Returns the hour value (0 through 23) of the receiver. On “spring back” days a value of 1 is returned for two consecutive hours, but with a different time zone.

See also: – **dayOfMonth**; – **dayOfWeek**; – **dayOfYear**; – **minuteOfHour**;
– **monthOfYear**; – **secondOfMinute**; – **yearOfCommonEra**

initWithString:

– **initWithString:**(NSString *)*description*

Returns an NSCalendarDate object initialized with the date specified as a string in *description*. This string must conform the international format for date representation YYYY-MM-DD HH:MM:SS +HHMM, where +HHMM is an hour and minute offset from Greenwich Mean Time. An example of such a representation is “1995-05-11 13:27:23 +0700”.

initWithString:calendarFormat:

– **initWithString:**(NSString *)*description* **calendarFormat:**(NSString *)*format*

Returns an NSCalendarDate object initialized with the date specified as a string in *description* and interpreted by the format string *format*. The format string consists of conversion specifiers for dates that are similar to those used in **strftime()**. See the description of the class method **dateWithString:calendarFormat:** for a listing of these specifiers.

For an example, let's assume you want to initialize an NSCalendarDate object with a string obtained from a text field. This date string takes the form "03.21.94 22:00 PST":

```
NSCalendarDate *newDate = [[NSCalendarDate alloc]
              initWithString:[NSString stringWithCString:
              [dateField stringValue]]
              calendarFormat:@"%m.%d.%y %H:%M %Z"]
```

Note that this method does not set the default calendar format for the NSCalendar object. To set the default format, use **setCalendarFormat:**.

See also: setCalendarFormat:

initWithYear:month:day:hour:minute:second:timeZone:

– **initWithYear:**(int)*year*
month:(unsigned int)*month*
day:(unsigned int)*day*
hour:(unsigned int)*hour*
minute:(unsigned int)*minute*
second:(unsigned int)*second*
timeZone:(id <NSTimeZone>)*timeZone*

Returns an NSCalendarDate object that is initialized with the year, month, day, hour and second offsets, as well as the time-zone object that are specified as arguments. The offsets can be positive (future) or negative (past), except for years, which cannot be negative. The *year* value must include the century (for example, 1995 instead of 95). The other values are the standard ones: 1 through 12 for months, 1 through 31 for days, 0 through 23 for hours and 0 through 59 for both minutes and seconds.

The method verifies the time zone supplied as an argument and can substitute an alternative time zone. If the method does supply a new time zone, it applies the difference in offsets-from-GMT values between the substituted and the original time zone to the date object being created.

The following code fragment shows a NSDate object created with a date on the fourth of July, 7 PM, Eastern Standard Time.

```
NSCalendarDate *fireWorks = [[NSCalendarDate alloc] initWithYear:1994
                        month:7 day:4 hour:19 minute:0 second:0
                        timeZone:[NSTimeZone timeZoneForKey:@"EST"]];
```

The **timeZoneForKey:** method (from NSTimeZone) gets an object based on a key value, in this case the standard abbreviation for a time zone.

minuteOfHour

– (int)**minuteOfHour**

Returns the minutes value (0 through 59) of the receiver.

See also: – **dayOfMonth**; – **dayOfWeek**; – **dayOfYear**; – **hourOfDay**; – **monthOfYear**; – **secondOfMinute**; – **yearOfCommonEra**

monthOfYear

– (int)**monthOfYear**

Returns a number that indicates the month of the year (1 through 12) of the receiver.

See also: – **dayOfMonth**; – **dayOfWeek**; – **dayOfYear**; – **hourOfDay**; – **minuteOfHour**; – **secondOfMinute**; – **yearOfCommonEra**

secondOfMinute

– (int)**secondOfMinute**

Returns the seconds value (0 through 59) of the receiver.

See also: – **dayOfMonth**; – **dayOfWeek**; – **dayOfYear**; – **hourOfDay**; – **minuteOfHour**; – **monthOfYear**; – **yearOfCommonEra**

setCalendarFormat:

– (void)**setCalendarFormat**:(NSString *)*format*

Sets the default calendar format for the NSCalendar object. A calendar format string is a string formatted with date-conversion specifiers. If you do not specify a calendar format for an object, NSCalendarDate substitutes its own default for methods such as **description** and

initWithString:. This is the international format of "%Y-%m-%d %H:%M:%S %z" (for example, 1994-01-14 16:45:12 +0900). See the description of the class method **dateWithString:calendarFormat:** for a listing of date-conversion specifiers.

See also: – **calendarFormat**

setTimeZone:

– (void)**setTimeZone:**(NSTimeZone *)*timeZone*

Sets the time zone that is associated with the NSCalendarDate object. When you create an NSCalendarDate object with the methods **dateWithString:calendarFormat**:, **initWithString:**, or **initWithString:calendarFormat:**, NSCalendarDate sets the associated time zone to be the default time zone for the locale. With this method you can set it to another time zone.

Note: The methods **dateWithYear:month:day:hour:minute:second:timeZone** and **initWithYear:month:day:hour:minute:second:timeZone** return NSCalendarDate objects initialized to a known time zone.

See also: – **timeZoneDetail**

timeZoneDetail

– (NSTimeZoneDetail *)**timeZoneDetail**

Returns the time-zone detail object that is associated with the NSCalendarDate object.

See also: – **setTimeZone:**

yearOfCommonEra

– (int)**yearOfCommonEra**

Returns a number that indicates the year, including the century, of the receiver (for example, 1995). The base year of the Common Era is 1 A.C.E. (which is the same as 1 A.D).

See also: – **dayOfMonth**; – **dayOfWeek**; – **dayOfYear**; – **hourOfDay**; – **minuteOfHour**; – **monthOfYear**; – **secondOfMinute**