NSTimeZone Class Cluster

Class Cluster Description

From the NSTimeZone class cluster you can obtain immutable objects that represent time zones. NSTimeZone, an abstract class, is the programmatic interface to several private concrete classes. NSTimeZone identifies a time zone by a geopolitical name and groups the related details of common abbreviation and temporal offset from Greenwich Mean Time (GMT). NSTimeZoneDetail, a public subclass of NSTimeZone, specializes the behavior of NSTimeZone by providing the abbreviation, the offset from GMT, and an indication of whether Daylight Savings Time is in effect.

For example, "US/Pacific" identifies the geopolitical time zone for San Francisco and Los Angeles, which falls in the same general longitude as that for the time zone "Canada/Pacific." The US/Pacific time-zone has specific refinements that specify PST (Pacific Standard Time) and PDT (Pacific Daylight Time), which have slightly different offsets from GMT.

You typically associate the objects returned by NSTimeZone (and, by extension, NSTimeZoneDetail) with date objects to affect their behavior. Time-zone objects can be of various types:

- time zones with hour and minute offsets from Greenwich Mean Time (GMT)
- time zones with a single abbreviation and offset
- time zones that vary according to Standard Time and Daylight Savings Time

You refer to the objects you create using NSTimeZone as *time-zone objects*. Because of the nature of class clusters, time-zone objects returned by this class are not instances of that abstract class (or of NSTimeZoneDetail) but of one of their private subclasses. Although a time-zone object's class is private, its interface is public, as declared by the abstract superclass, NSTimeZone. (See "Class Clusters" in the introduction to the Foundation Kit for more information on class clusters and creating subclasses within a cluster.)

You use the class methods **defaultTimeZone**, **localTimeZone**, **timeZoneWithName:**, **timeZoneWithAbbreviation:**, and **timeZoneForSecondsFromGMT:** to get suitable time-zone objects. The instance method **timeZoneDetailForDate:** returns the time-zone detail object for a specific date.

NSTimeZoneDetail does not itself provide any class methods for obtaining time-zone objects. It exists mainly as a refinement of NSTimeZone. It does, however, provide accessor methods for obtaining abbreviations and GMT offsets, and for determining whether Daylight Savings Time is in effect.

NSTimeZone and its subclasses adopt the NSCopying and NSCoding protocol.

▶ NSTimeZone

Inherits From: NSObject

Conforms To: NSCopying
NSCoding
NSObject

Declared In: foundation/NSDate.h

Class Description

NSTimeZone is an abstract class that defines the behavior of time-zone objects. By itself, NSDate represents dates as *universal time*. Universal time treats a date and time value as identical in, for instance, Redwood City and New York City. There is no provision for locale and, consequently, for time zone. Provision for locale is critical for string representations and other expressions of conventional dates and times. NSTimeZone affects the temporal value of date objects so that they reflect locale information related to time zones.

Time-zone objects represent geopolitical regions. Consequently, these objects have names for these regions. Time-zone objects, in their detail form, also represent a temporal offset, either plus or minus, from Greenwich Mean Time (GMT) and an abbreviation.

The system supplies many choices for time zones in the **/etc/zoneinfo** directory. The files in this directory store time-zone information (for the format of these files, see **tzfile (5)**). The ASCII property list **RegionsDictionary** restricts these choices to subsets based on longitude (as used by Preferences). You can access these choices through the **timeZoneArray** class method. Another restriction is the choice of time zone available when there is an ambiguous abbreviation; these choices are available with an invocation of the class method **abbreviationDictionary**. Despite these restrictions, you can obtain an arbitrary, **tzfile**-formatted file through the class method **timeZoneWithName**.

Note: By itself, the NSTimeZone class only *names* a time zone. It does not associate an abbreviation or a temporal offset with a time zone; that is done by NSTimeZoneDetail. An instance of NSTimeZone, however, "knows" about the set of time-zone detail objects related to it.

NSTimeZone provides several class methods to get time-zone objects, with or without detail: **timeZoneWithName:**, **timeZoneWithAbbreviation:**, and **timeZoneForSecondsFromGMT:**. The class also permits you to set the default time zone

for your locale (**setDefaultTimeZone:**) You can access this default time zone at any time, and, with the **localTimeZone** class method, you can also get a relative time-zone object that will decode itself to become the default time zone for any locale in which it finds itself.

Some NSCalendarDate methods return date objects that are automatically bound to time-zone detail objects. These date objects use the functionality of NSTimeZone to adjust dates for the proper locale. Unless you specify otherwise, objects returned from NSCalendarDate are bound to the default time zone for the current locale. A useful instance method is **timeZoneDetailForDate:**, which returns a time-zone detail object associated with a specific date.

Instance Variables

None declared in this class.

Method Types

Getting time zones	+ defaultTimeZone + localTimeZone – timeZoneDetailForDate: + timeZoneForSecondsFromGMT: + timeZoneWithAbbreviation: + timeZoneWithName:
Getting arrays of time zones	+ timeZoneArray – timeZoneDetailArray
Managing time zones	+ setDefaultTimeZone:
Getting time-zone information	+ abbreviationDictionary – timeZoneName

Class Methods

abbreviationDictionary

+ (NSDictionary *)**abbreviationDictionary**

Returns a dictionary object holding the mappings of time-zone region name to time-zone abbreviation. Region names are typically file names in **/etc/zoneinfo**. If you know a region

name for a key, you can obtain a valid abbreviation from the dictionary and use it to obtain a detail time-zone object using **timeZoneWithAbbreviation:**.

See also: + **timeZoneArray**; + **timeZoneDetailArray**

defaultTimeZone

+ (NSTimeDetailZone *)**defaultTimeZone**

Returns the default time zone set for the current locale.

See also: + **localTimeZone**; + **setDefaultTimeZone:**

localTimeZone

+ (NSTimeZone *)**localTimeZone**

Returns the default time zone in the current locale. This adjustment happens during decoding, during which the object replaces itself the current default time zone. This behavior is particularly useful for NSCalendarDate objects that are archived or sent as Distributed Objects and interpreted in different locales.

See also: + **defaultTimeZone**; + **setDefaultTimeZone:**

setDefaultTimeZone:

+ (void)**setDefaultTimeZone***:*(NSTimeZone *)*aTimeZone*

Sets the time zone appropriate for the current locale. There can be only one default time zone, so by setting a new default time zone, you lose the previous one. The system automatically sets the default time zone to be the one selected in Preferences.

See also: + **defaultTimeZone**; + **localTimeZone**

timeZoneArray

+ (NSArray *)**timeZoneArray**

Returns an array of string-object arrays showing all current region names for each time zone, grouping them as subarrays per latitudinal region. These arrays are a direct reflection of what you see in the “time” view of the Preferences application.

See also: + **abbreviationDictionary**, – **timeZoneDetailArray**

timeZoneForSecondsFromGMT:

+ (NSTimeZone *)**timeZoneForSecondsFromGMT**:(int)*seconds*

Returns the time-zone object with *seconds* offset from Greenwich Mean Time. If there is no object matching the offset, this method creates and returns a new time-zone object bearing the value *seconds* as a name.

See also: + **timeZoneWithAbbreviation:**; + **timeZoneWithName:**

timeZoneWithAbbreviation:

+ (NSTimeZoneDetail *)**timeZoneWithName**:(NSString *)*abbreviation*

Returns the time-zone object identified by the abbreviated time zone, *abbreviation*. If there is no match on *abbreviation*, this method returns **nil**.

See also: + **timeZoneForSecondsFromGMT:**; + **timeZoneWithName:**;

timeZoneWithName:

+ (NSTimeZone *)**timeZoneWithName**:(NSString *)*aTimeZoneName*

Returns the time-zone object identified by the name *aTimeZoneName*. It searches both the regions dictionary and **/etc/zoneinfo** for matching names. If there is no match on the name, this method returns **nil**.

See also: + **timeZoneForSecondsFromGMT:**; + **timeZoneWithAbbreviation:**

Instance Methods

timeZoneDetailArray

– (NSArray *)**timeZoneDetailArray**

Returns an array object that contains an array of NSTimeZoneDetail objects associated with the current time-zone object. For example, the “US/Pacific” time zone supplies an array of three specializations identified by the abbreviations PDT, PST, and PWT and specifying the ranges of time associated with Pacific Daylight Time, Pacific Standard Time, and Pacific War Time (circa 1942)

See also: + **timeZoneArray**

timeZoneDetailForDate:

– (NSTimeZoneDetail *)**timeZoneForDate:**(NSDate *)*date*

Returns the detail time-zone object (with its abbreviation and GMT offset) that is associated with a *date* object.

See also: – **timeZoneName**

timeZoneName

– (NSString *)**timeZoneName**

Returns the geopolitical name that identifies the time zone.

See also: – **timeZoneDetailForDate**

▶ NSTimeZoneDetail

Inherits From: NSTimeZone : NSObject

Conforms To: NSCopying
NSCoding
NSObject

Declared In: foundation/NSDate.h

Class Description

NSTimeZone is an abstract class that refines the behavior provided by NSTimeZone. NSTimeZone identifies a geopolitical area with a name (such as US/Pacific). NSTimeZoneDetail augments this region name with more specific information: an abbreviation, an offset (in seconds) from Greenwich Mean Time (GMT), and an indication of whether Daylight Savings Time is in effect. The specificity afforded through NSTimeZoneDetail helps to resolve conflicts between abbreviations and offsets that can arise within regions.

Even though it is an abstract subclass of NSTimeZone, NSTimeZoneDetail does *not* have "factory" class methods that create and return time-zone objects. See the specification of NSTimeZone for methods that do.

However, NSTimeZoneDetail does have accessor methods that allow you to get the abbreviation and temporal offset of a time-zone object, as well as to determine whether Daylight Savings Time is in effect.

Instance Variables

None declared in this class.

Method Types

Getting time-zone information
- isDaylightSavingsTimeZone
- timeZoneAbbreviation
- timeZoneSecondsFromGMT

Instance Methods

isDaylightSavingsTimeZone

– (BOOL)**isDaylightSavingsTimeZone**

Returns YES if the time-zone detail object is used in the representation of dates during Daylight Savings Time.

timeZoneAbbreviation

– (NSString *)**timeZoneAbbreviation**

Returns the abbreviation for the time-zone object, such as EDT (Eastern Daylight Time).

timeZoneSecondsFromGMT

– (int)**timeZoneSecondsFromGMT**

Returns the difference in seconds between the receiving time-zone object and Greenwich Mean Time (GMT). The offset can be a positive or negative value.