Foundation / DateComponentsFormatter

Class

DateComponentsFormatter

A formatter that creates string representations of quantities of time.

iOS 8.0+ | iPadOS 8.0+ | Mac Catalyst 13.1+ | macOS 10.10+ | tvOS 9.0+ | visionOS 1.0+ | watchOS 2.0+

```
class DateComponentsFormatter
```

Overview

An `DateComponentsFormatter` object takes quantities of time and formats them as a user-readable string. Use a date components formatter to create strings for your app's interface. The formatter object has many options for creating both abbreviated and expanded strings. The formatter takes the current user's locale and language into account when generating strings.

To use this class, create an instance, configure its properties, and call one of its methods to generate an appropriate string. The properties of this class let you configure the calendar and specify the date and time units you want displayed in the resulting string. The listing below shows how to configure a formatter to create the string "About 5 minutes remaining".

Swift Objective-C

```
let formatter = DateComponentsFormatter()
formatter.unitsStyle = .full
formatter.includesApproximationPhrase = true
formatter.includesTimeRemainingPhrase = true
formatter.allowedUnits = [.minute]

// Use the configured formatter to generate the string.
let outputString = formatter.string(from: 300.0)
```

The methods of this class may be called safely from any thread of your app. It is also safe to share a single instance of this class from multiple threads, with the caveat that you should not change the configuration of the object while another thread is using it to generate a string.

Tip

In Swift, you can use `Date.RelativeFormatStyle` rather than `DateComponentsFormatter`. The `FormatStyle` API offers a declarative idiom for customizing the formatting of various types. Also, Foundation caches identical `FormatStyle` instances, so you don't need to pass them around your app, or risk wasting memory with duplicate formatters.

Topics

Formatting Values

`func string(from: DateComponents) -> String?`

Returns a formatted string based on the specified date component information.

`func string(for: Any?) -> String?`

Returns a formatted string based on the date information in the specified object.

`func string(from: Date, to: Date) -> String?`

Returns a formatted string based on the time difference between two dates.

`func string(from: TimeInterval) -> String?`

Returns a formatted string based on the specified number of seconds.

`class func localizedString(from: DateComponents, unitsStyle: DateComponentsFormatter.UnitsStyle) -> String?`

Returns a localized string based on the specified date components and style option.

Configuring the Formatter Options

`var allowedUnits: NSCalendar.Unit`

The bitmask of calendrical units such as day and month to include in the output string.

`var allowsFractionalUnits: Bool`

A Boolean indicating whether non-integer units may be used for values.

`var calendar: Calendar?`

The default calendar to use when formatting date components.

`var collapsesLargestUnit: Bool`

A Boolean value indicating whether to collapse the largest unit into smaller units when a certain threshold is met.

`var includesApproximationPhrase: Bool`

A Boolean value indicating whether the resulting phrase reflects an inexact time value.

`var includesTimeRemainingPhrase: Bool`

A Boolean value indicating whether output strings reflect the amount of time remaining.

`var maximumUnitCount: Int`

The maximum number of time units to include in the output string.

`var unitsStyle: DateComponentsFormatter.UnitsStyle`

The formatting style for unit names.

`var zeroFormattingBehavior: DateComponentsFormatter.ZeroFormattingBehavior`

The formatting style for units whose value is 0.

Constants

`enum UnitsStyle`

Constants for specifying how to represent quantities of time.

`struct ZeroFormattingBehavior`

Formatting constants for when values contain zeroes.

Instance Properties

`var formattingContext: Formatter.Context`

`var referenceDate: Date?`

Instance Methods

```
func getObjectValue(AutoreleasingUnsafeMutablePointer<AnyObject?>?, for
: String, errorDescription: AutoreleasingUnsafeMutablePointer<NSString?
>?) -> Bool
```

Relationships

Inherits From

`Formatter`

Conforms To

`CVarArg`
`CustomDebugStringConvertible`
`CustomStringConvertible`
`Equatable`
`Hashable`
`NSCoding`
`NSCopying`
`NSObjectProtocol`
`Sendable`
`SendableMetatype`

See Also

Dates and times

`class DateFormatter`

A formatter that converts between dates and their textual representations.

`class RelativeDateTimeFormatter`

A formatter that creates locale-aware string representations of a relative date or time.

`class DateIntervalFormatter`

A formatter that creates string representations of time intervals.

`class ISO8601DateFormatter`

A formatter that converts between dates and their ISO 8601 string representations.