Documentation Language: Swift

Foundation / PersonNameComponentsFormatter

Class

PersonNameComponentsFormatter

A formatter that provides localized representations of the components of a person’s name.

iOS 9.0+ | iPadOS 9.0+ | Mac Catalyst 13.1+ | macOS 10.11+ | tvOS 9.0+ | visionOS 1.0+ | watchOS 2.0+

```
class PersonNameComponentsFormatter
```

Overview

Each locale has its own set of rules and conventions for how personal names are structured and represented. These rules vary widely across different locales in a several ways, including the sort and display order of given and family names, the use of salutations and honorifics, and other concerns related to the grammar, spelling, punctuation, and formatting. About the only thing that *is* consistent across all locales is that personal names are significant and meaningful. For this reason, names deserve careful and respectful treatment—perhaps more than any other kind of information your app interacts with.

Formatters can be configured to represent names in a variety of styles, which are described in detail below.

- Default (`PersonNameComponentsFormatter.Style.default`)
- Short (`PersonNameComponentsFormatter.Style.short`)
- Long (`PersonNameComponentsFormatter.Style.long`)
- Abbreviated (`PersonNameComponentsFormatter.Style.abbreviated`)

When determining how to represent a name in a particular style, a formatter takes a number of factors into consideration, in order of priority:

1. **Script derived behaviors** Scripts may specify a strict sort or display order of given and family names, and the availability of styles.
2. **User specified preferences** Users can enable and configure the display of short names, as well as whether or not to display nicknames when available. Users can also override the default sort and display order of given and family names for their current locale.
3. **Locale derived defaults** Locales specify a default sort and display order for given and family names.
4. **Developer specified configuration** The style property value set for the `NSPersonNameComponentsFormatter` object.

When the behavior specified in one factor conflicts with any other factors, the behavior specified by the factor with the most precedence is used. For example, the U.S. English (en–US) locale specifies that names be displayed in “given name followed by the family name” (for example,“John Appleseed”). This behavior would be overridden if the user changed their system preferences to have names displayed as family name followed by given name (for example, “Appleseed, John”), because user-specified preferences take precedence over locale-derived defaults. Furthermore, if the name to be formatted were Japanese (for example, given name: “泰夫”, family name: “木田”), the behavior derived for the name’s script (CJK, for Chinese, Japanese, and Korean languages) would take precedence over any locale-derived defaults or user-specified preferences to have the name displayed as family name followed by given name (for example, “木田 泰夫”).

These considerations extend to the availability of certain formatter styles as well. Because developer-specified configurations have the lowest precedence in determining behavior, the value set for the formatter’s style property can be invalidated if it’s not supported for the locale, user preferences, or script. If the specified style is not available, the next longest valid style is used. For example, a name in Arabic script (for example, “أحمد الراجحي”) does not support the Abbreviated style, so the Short style is used instead. A name that contains more than one script (for example, given name: “John”, family name: “王”) is detected to have “Unknown” script, which has its own set of behaviors and characteristics.

Tip

In Swift, you can use `PersonNameComponents.FormatStyle` rather than `PersonNameComponentsFormatter`. 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.

Styles

`NSPersonNameComponentsFormatter` can be configured to format names in the following styles:

`PersonNameComponentsFormatter.Style.default`
: The minimally necessary features for differentiation in a casual setting. Equivalent to `PersonNameComponentsFormatter.Style.medium`.

`PersonNameComponentsFormatter.Style.short`
: Relies on user preferences and language defaults to display shortened form appropriate for display in space-constrained settings.

`PersonNameComponentsFormatter.Style.long`
: The fully qualified name complete with all known components.

`PersonNameComponentsFormatter.Style.abbreviated`
: The maximally abbreviated form of a name.

	`namePrefix`	**`givenName`**	**`middleName`**	**`familyName`**	**`nameSuffix`**	**`nickname`**
Arabic `(ar-SA)`	د.	أحمد		محمدالمصري		
Chinese `(zh-Hans)`	物理学博士	振宁		杨	先生	
English `(en-US)`	Dr.	Jonathan	Maple	Appleseed	Esq.	Johnny
French `(fr-FR)`	Père	Jean-Philippe		de Zélicourt		JP
German `(de-DE)`	Dr. med.	Max		Mustermann	junior, M.A.	
Hindi `(hi-IN)`	डॉ.	रिय		साहिल		
Japanese		泰夫		木田	先生	

	namePrefix	givenName	middleName	familyName	nameSuffix	nickname
(ja-JP)						
Spanish (es-ES)	Dr.	José Ramiro		Martín González de Rivera	júnior, PhD	Ramiro
Thai (th-TH)	ฯพณฯ	สมชาย	ปีเตอร์	รัตนเรืองรองบวรทิพย์		

Default

The Default, or Medium, style presents names in a way that is suitable for most contexts. It uses the given and family names, as well as a nickname, if provided and enabled by the user in System Preferences.

	Default style
Arabic (ar-SA)	أحمد محمدالمصري
Chinese (zh-Hans)	杨振宁
English (en-US)	Jonathan Appleseed
French (fr-FR)	Jean-Philippe de Zélicourt
German (de-DE)	Max Mustermann
Hindi (hi-IN)	रिय साहिल
Japanese (ja-JP)	木田泰夫
Spanish (es-ES)	José Ramiro Martín González de Rivera
Thai (th-TH)	สมชาย รัตนเรืองรอง บวรทิพย์

Short

The Short style offers an alternative display method for names whose default representation may exceed a certain length constraint. It is only available if the user has enabled “Short Names” in

System Preferences, and only for names with a script that supports Short style. Otherwise, a formatter configured to display with Short style is displayed with Medium style instead.

If a user has enabled the use of short names, the user can choose from one of four variations:

- Given Name - Family Initial
- Family Name - Given Initial
- Given Name Only
- Family Name Only

Short style is not available for names in CJK script and is restricted to Given Name Only or Family Name Only for names in Arabic or Devanagari script. If the specified Short style is unavailable, the Medium style is used instead.

	Given Name - Family Initial	**Family Name - Given Initial**	**Given Name Only**	**Family Name Only**
Arabic (ar-SA)	*N/A*	*N/A*	أحمد	محمدالمصري
Chinese (zh-Hans)	*N/A*	*N/A*	*N/A*	*N/A*
English (en-US)	Jonathan A	J Appleseed	Jonathan	Appleseed
French (fr-FR)	Jean-Philippe d	J de Zélicourt	Jean-Philippe	de Zélicourt
German (de-DE)	Max M	M Mustermann	Max	Mustermann
Hindi (hi-IN)	*N/A*	*N/A*	रिय	साहिल
Japanese (ja-JP)	*N/A*	*N/A*	*N/A*	*N/A*
Spanish (es-ES)	José Ramiro M	J Martín González de Rivera	José Ramiro	Martín González de Rivera
Thai (th-TH)	สมชาย ร	ส รัตนเรืองรองบวรทิพย์	สมชาย	รัตนเรืองรองบวรทิพย์

Important

`NSPersonNameComponentsFormatter` does not currently account for prepositional particles. Representations using the Short style that specify a family name initial naively use the first letter unit of the particle as the initial.

Long

The Long style provides the most explicit representation of names. It uses all available name components, with the exception of nickname.

	Long style
Arabic (ar-SA)	د. أحمد محمدالمصري
Chinese (zh-Hans)	物理学博士杨振宁先生
English (en-US)	Dr. Jonathan Maple Appleseed Esq.
French (fr-FR)	Père Jean-Philippe de Zélicourt
German (de-DE)	Dr. med. Max Mustermann junior, M.A.
Hindi (hi-IN)	डॉ. रिय साहिल
Japanese (ja-JP)	木田泰夫先生
Spanish (es-ES)	Dr. José Ramiro Martín González de Rivera júnior, PhD
Thai (th-TH)	ฯพณฯ สมชาย ปีเตอร์ รัตนเรืองรอง บวรทิพย์

Abbreviated

The Abbreviated style offers the most compact representation of names, similar to a monogram.

Abbreviated style is supported for names in several scripts, with the following general characteristics:

- For names in Cyrillic, Greek, or Latin script, the first characters of `givenName`, `middleName`, and `familyName` may be used.
- For names in Chinese or Japanese script, `familyName` may be used. If `familyName` is too long, or if the family name is `nil`, the Short or Medium style may be used instead.

- For names in Korean script, `givenName` may be used. If `givenName` is too long, the first character of `givenName` may be used. If `givenName` is `nil`, the `familyName` may be used instead.
- For names in Bengali, Devanagari, Gujarati, Gurmukhi, Kannada, Malayalam, Oriya, Sinhala, Tamil, Telugu, Tibetan, or Thai script, the first character of `givenName` may be used. If `givenName` is `nil`, the first character of `familyName` may be used instead.
- For names that contain more than one script, the abbreviated style may use the `familyName`, `givenName`, or the first characters of `givenName` and/or `familyName`.

If the Abbreviated style is unavailable, the Short style is used instead—unless that too is unsupported, in which case the Medium style is used instead.

	Abbreviated style
Arabic (ar-SA)	*N/A*
Chinese (zh-Hans)	杨
English (en-US)	JMA
French (fr-FR)	Jd
German (de-DE)	MM
Hindi (hi-IN)	मि
Japanese (ja-JP)	木田
Spanish (es-ES)	JM
Thai (th-TH)	ส

Important

`NSPersonNameComponentsFormatter` doesn’t currently account for prepositional particles or compound names. Representations using the Abbreviated style uses the first letter unit of each name component, regardless.

Topics

Configuring Formatter Behavior

```
var style: PersonNameComponentsFormatter.Style
```

The formatting style of the receiver.

```
var isPhonetic: Bool
```

A Boolean value that specifies whether the receiver should use only the phonetic representations of name components.

Converting Between Person Name Components and Strings

```
class func localizedString(from: PersonNameComponents, style: PersonNameComponentsFormatter.Style, options: PersonNameComponentsFormatter.Options) -> String
```

Returns a string formatted for a given `NSPersonNameComponents` object using the provided style and options.

```
func string(from: PersonNameComponents) -> String
```

Returns a string formatted for a given `NSPersonNameComponents` object.

```
func annotatedString(from: PersonNameComponents) -> NSAttributedString
```

Returns an attributed string formatted for a given `NSPersonNameComponents` object, with attribute annotations for each component.

```
func personNameComponents(from: String) -> PersonNameComponents?
```

Returns a person name components object from a given string.

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

Returns by reference a person name components object after creating it from a given string.

Constants

```
enum Style
```

```
struct Options
```

Attributed String Key

This constant is used as a key for person name component attributes in attributed strings returned by the `annotatedString(from:)` method

☰ Attributed String Components

These constants are used to identify individual components of attributed strings returned by the `annotatedString(from:)` method.

☰ Component Delimiter

This constant defines the delimiter used to separate name components.

Instance Properties

`var locale: Locale!`

Relationships

Inherits From

`Formatter`

Conforms To

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

See Also

Names

`struct PersonNameComponents`

The separate parts of a person’s name, allowing locale-aware formatting.