Structure

TermOfAddress

The type for representing grammatical gender in localized text.

iOS 17.0+ | iPadOS 17.0+ | Mac Catalyst 17.0+ | macOS 14.0+ | tvOS 17.0+ | visionOS 1.0+ | watchOS 10.0+

```
struct TermOfAddress
```

Overview

Many languages rely on gender for their grammar. Without knowing the subject's gender or pronoun preferences, some localized strings may have grammatical errors, resulting in a poor user experience.

`TermOfAddress` is a type that enables the system to make pronoun substitutions in localized text based on gender. You don't create instances of this type directly. Instead, use the predefined types to specify the gender to use when referring to people in translated text. Or define your own pronoun terms for a specific language when the predefined types are insufficient.

> **Note**
>
> Not all terms of address exist in all languages.

For example, to substitute the masculine pronoun *He*, for the neutral pronoun *They*, do the following:

```
// Define the resource you want to apply grammatical agreement to.
let resource = LocalizedStringResource("^[They](referentConcept: 1) liked your post.

// Set the inflection concept to use a term of address.
```

```
var options = AttributedString.LocalizationOptions()
options.concepts = [.termsOfAddress([.masculine])]

// Make a new string imposing grammatical agreement on the resource from the term of
let result = AttributedString(localized: resource, options: options)
// result == "He liked your post."
```

If the `masculine`, `feminine`, and `neutral` terms of address are insufficient, create your own term of address specifying the pronouns and language.

```
// Define the various morphologies.
var nominativeMorphology = Morphology()
var accusativeMorphology = Morphology()
var genitiveMorphology = Morphology()
var genetiveIndependent = Morphology()
var reflexive = Morphology()

nominativeMorphology.grammaticalCase = .nominative
accusativeMorphology.grammaticalCase = .accusative
genitiveMorphology.grammaticalCase = .genitive
genitiveMorphology.determination = .dependent
genetiveIndependent.grammaticalCase = .genitive
genetiveIndependent.determination = .independent
reflexive.pronounType = .reflexive

// Define the pronouns.
let xey  = Morphology.Pronoun(pronoun:"xey", morphology: nominativeMorphology)
let xem  = Morphology.Pronoun(pronoun:"xem", morphology: accusativeMorphology)
let xeir = Morphology.Pronoun(pronoun:"xeir", morphology: genitiveMorphology)
let xeirs = Morphology.Pronoun(pronoun: "xeirs", morphology: genetiveIndependent)
let xemself = Morphology.Pronoun(pronoun: "xemself", morphology: reflexive)

// Create the localized term of address.
let xemTermOfAddress = TermOfAddress.localized(language: .init(identifier: "en"), pr

// Define the resources you want to apply gender agreement to.
let resources = [
    LocalizedStringResource("^[They](referentConcept: 1) liked your post."),
    LocalizedStringResource("Anne read the post to ^[them](referentConcept: 1)."),
    LocalizedStringResource("You liked ^[their](referentConcept: 1) post."),
    LocalizedStringResource("The post was ^[theirs](referentConcept: 1)."),
    LocalizedStringResource("^[They](referentConcept: 1) posted it ^[themselves](ref
]
```

```
var options = AttributedString.LocalizationOptions()

// Set the inflection concept to use the new term of address.
options.concepts = [.termsOfAddress([xemTermOfAddress])]

let results = [
    AttributedString(localized: resources[0], options: options), // "Xey liked your
    AttributedString(localized: resources[1], options: options), // "Anne read the p
    AttributedString(localized: resources[2], options: options), // "You liked xeir
    AttributedString(localized: resources[3], options: options), // "The post was xe
    AttributedString(localized: resources[4], options: options), // "Xey posted it x
]
```

For examples of how to use terms of address, see:

- AttributeScopes.FoundationAttributes.ReferentConceptAttribute
- AttributeScopes.FoundationAttributes.AgreementConceptAttribute

Topics

Using predefined terms of address

`static let feminine: TermOfAddress`

The term of address for representing feminine pronouns and grammatical gender.

`static let masculine: TermOfAddress`

The term of address for representing masculine pronouns and grammatical gender.

`static let neutral: TermOfAddress`

The term of address for representing gender-neutral pronouns and grammatical gender.

Defining your own terms of address

`static func localized(language: Locale.Language, pronouns: [Morphology.Pronoun]) -> TermOfAddress`

Returns a term of address restricted to a specific language for a group of pronouns.

`var language: Locale.Language?`

The specific language associated with a term of address.

`var pronouns: [Morphology.Pronoun]`

The pronouns associated with a term of address.

Type Properties

`static let currentUser: TermOfAddress`

Relationships

Conforms To

`Copyable`
`Decodable`
`Encodable`
`Equatable`
`Hashable`
`Sendable`
`SendableMetatype`

See Also

Automatic grammar agreement

`enum InflectionRule`

A rule that affects how an attributed string performs automatic grammatical agreement.

`struct Morphology`

A description of the grammatical properties of a string.

`enum InflectionConcept`

An inflection method to use when localizing text.

`struct Pronoun`

A custom pronoun for referring to a third person.