What's in a Name?

NSPeople

PersonNameComponents

&

PersonNameComponentsFormatter

Components of a Name

```
var grace = PersonNameComponents()
grace.namePrefix = "Mrs."
grace.givenName = "Grace"
grace.middleName = "Murray"
grace.familyName = "Hopper"
```

Styles: abbreviated to long

```
private func log(_ components: PersonNameComponents) {
    print("Components = '\(components)'")
    let defaultName = PersonNameComponentsFormatter.localizedString(from: components,
                                                                    style: .default,
                                                                    options: [])
    print("Default = '\(defaultName)'")
    let longName = PersonNameComponentsFormatter.localizedString(from: components,
                                                                 style: .long,
                                                                 options: [])
    print("Long = '\(longName)'")
    let mediumName = PersonNameComponentsFormatter.localizedString(from: components,
                                                                   style: .medium,
                                                                   options: [])
    print("Medium = '\(mediumName)'")
    let shortName = PersonNameComponentsFormatter.localizedString(from: components,
                                                                  style: .short,
                                                                  options: [])
    print("Short = '\(shortName)'")
    let abbrevName = PersonNameComponentsFormatter.localizedString(from: components,
                                                                   style: .abbreviated,
                                                                   options: [])
    print("Abbreviated = '\(abbrevName)'")
}
```

Output

```
Components =
     'namePrefix: Mrs.
     givenName: Grace
     middleName: Murray
     familyName: Hopper '
Default =    'Grace Hopper'
Long =       'Mrs. Grace Murray Hopper'
Medium =     'Grace Hopper'
Short =      'Grace'
Abbreviated =    'GH'
```

Another Example

```
var bill = PersonNameComponents()
bill.namePrefix = "Mr."
bill.givenName = "William"
bill.middleName = "James \"Frickin\""
bill.familyName = "Murray"
bill.nameSuffix = "Esq."
bill.nickname = "Bill"
```

Another Example

```
var bill = PersonNameComponents()
bill.namePrefix = "Mr."
bill.givenName = "William"
bill.middleName = "James \"Frickin\""
bill.familyName = "Murray"
bill.nameSuffix = "Esq."
bill.nickname = "Bill"

Components =    'namePrefix: Mr. givenName: William
    middleName: James "Frickin"
    familyName: Murray nameSuffix: Esq. nickname: Bill '
Default =   'William Murray'
Long =      'Mr. William James "Frickin" Murray Esq.'
Medium =    'William Murray'
Short =     'Bill'
Abbreviated =   'WM'
```

But what if I have a single, full-name String?

More Murray: String-to-Component

```
let nameFormatter = PersonNameComponentsFormatter()
let william = "Mr. William James (Frickin) 🍸 Murray-Jones de la Rocha Jr. 🕶"
if let nameComps  = nameFormatter.personNameComponents(from: william) {
    log(nameComps)
}
```

More Murray: String-to-Component

```
let nameFormatter = PersonNameComponentsFormatter()
let william = "Mr. William James (Frickin) 🍸 Murray-Jones de la Rocha Jr. 🕶"
if let nameComps  = nameFormatter.personNameComponents(from: william) {
    log(nameComps)
}

Components =    'namePrefix: Mr. givenName: William James
    middleName: Murray-Jones familyName: de la Rocha nameSuffix: Jr. '
Default =   'William James "Frickin" de la Rocha'
Long =      'Mr. William James "Frickin" Murray-Jones de la Rocha Jr.'
Medium =    'William James "Frickin" de la Rocha'
Short =     'William James "Frickin"'
Abbreviated =   'WD'
```

Another Murray: String-to-Component

```
let will = "Murray, Bill"
if let nameComps  = nameFormatter.personNameComponents(from: will) {
    log(nameComps)
}
```

Another Murray: String-to-Component

```
let will = "Murray, Bill"
if let nameComps  = nameFormatter.personNameComponents(from: will) {
    log(nameComps)
}
```

```
Components =     'givenName: Bill familyName: Murray '
Default =        'Bill Murray'
Long =           'Bill Murray'
Medium =         'Bill Murray'
Short =          'Bill'
Abbreviated =    'BM'
```

Junior is now a Doc and more Junior

```
let willy = "Mr. William James \"Frickin\" Murray-Jones de la Rocha MD. III"
if let nameComps  = nameFormatter.personNameComponents(from: willy) {
    log(nameComps)
}
```

```
let willy = "Mr. William James \"Frickin\" Murray-Jones de la Rocha MD. III"
if let nameComps  = nameFormatter.personNameComponents(from: willy) {
    log(nameComps)
}

Components =    'namePrefix: Mr. givenName: MD.
    familyName: William James "Frickin" Murray-Jones de la Rocha nameSuffix: III '
Default =   'MD. William James "Frickin" Murray-Jones de la Rocha'
Long =      'Mr. MD. William James "Frickin" Murray-Jones de la Rocha III'
Medium =    'MD. William James "Frickin" Murray-Jones de la Rocha'
Short =     'MD.'
Abbreviated =   'MW'
```

Mononyms?

```
let stingString = "Sting"
var sting = PersonNameComponents()
sting.givenName = stingString

let formatter = PersonNameComponentsFormatter()
if let nameComps  = formatter.personNameComponents(from: stingString) {
    log(nameComps)
}
```

Mononyms!

```
let stingString = "Sting"
var sting = PersonNameComponents()
sting.givenName = stingString

let formatter = PersonNameComponentsFormatter()
if let nameComps  = formatter.personNameComponents(from: stingString) {
    log(nameComps)
}
```

```
Components =    'givenName: Sting '
Default =       'Sting'
Long =          'Sting'
Medium =        'Sting'
Short =         'Sting'
Abbreviated =   'S'
Components =    'givenName: Sting ':
Default =       'Sting'
Long =          'Sting'
Medium =        'Sting'
Short =         'Sting'
Abbreviated =   'S'
```

Other Formats

Plain & Fancy

```
var dave = PersonNameComponents()
dave.givenName = "Dave"
dave.familyName = "Grohl"

let nameFormatter = PersonNameComponentsFormatter()
print("Plain & unlocalized = '\(nameFormatter.string(from: dave))'")

let attributed = nameFormatter.annotatedString(from: dave)
print("Attributed & annotated = \(attributed)")
```

Fancy & Plain

```
var dave = PersonNameComponents()
dave.givenName = "Dave"
dave.familyName = "Grohl"

let nameFormatter = PersonNameComponentsFormatter()
print("Plain & unlocalized = '\(nameFormatter.string(from: dave))'")

let attributed = nameFormatter.annotatedString(from: dave)
print("Attributed & annotated = \(attributed)")
```

```
Plain & unlocalized = 'Dave Grohl'
Attributed & annotated =
Dave{
    NSPersonNameComponentKey = givenName;
} {
    NSPersonNameComponentKey = delimiter;
}Grohl{
    NSPersonNameComponentKey = familyName;
}
```

What about Non-ASCII names?

CJK Script, Logographic Characters, Phonetics

```
var chiangKaiShek = PersonNameComponents()
chiangKaiShek.familyName = "蔣"
chiangKaiShek.givenName = "介石"
var chiangPhonetic = PersonNameComponents()
chiangPhonetic.familyName = "Chiang"
chiangPhonetic.givenName = "Kai-shek"
chiangKaiShek.phoneticRepresentation = chiangPhonetic
```

Added to log

```
if let phoneticRep = components.phoneticRepresentation {
    print("Phonetic representation = '\(phoneticRep)'")
}
```

Phonetics Mo'netics

```
var chiangKaiShek = PersonNameComponents()
chiangKaiShek.familyName = "蔣"
chiangKaiShek.givenName = "介石"
var chiangPhonetic = PersonNameComponents()
chiangPhonetic.familyName = "Chiang"
chiangPhonetic.givenName = "Kai-shek"
chiangKaiShek.phoneticRepresentation = chiangPhonetic

Components =    'givenName: 介石 familyName: 蔣
    phoneticRepresentation: givenName: Kai-shek familyName: Chiang  '
Default =   '蔣介石'
Long =      '蔣介石'
Medium =    '蔣介石'
Short =     '蔣介石'
Abbreviated =   '蔣'
Phonetic representation = 'givenName: Kai-shek familyName: Chiang '
```

Another Log Method

```
private func logPhonetics(_ components: PersonNameComponents) {
    let nameFormatter = PersonNameComponentsFormatter()
    print("Before: formatter is set to be phonetic? \(nameFormatter.isPhonetic)")
    print("Before = '\(nameFormatter.string(from: components))'")
    nameFormatter.isPhonetic = true
    print("After: formatter is set to be phonetic? \(nameFormatter.isPhonetic)")
    print("After = '\(nameFormatter.string(from: components))'")
}
```

Pho-netics Fo-real

```
logPhonetics(chiangKaiShek)
logPhonetics(chiangPhonetic)
chiangPhonetic.phoneticRepresentation = chiangKaiShek
logPhonetics(chiangPhonetic)
```

```
Formatting components with phonetics:
Before: formatter is set to be phonetic? false
Before = '蒋介石'
After: formatter is set to be phonetic? true
After = 'Chiang Kai-shek'
Formatting the phonetic component version directly:
Before: formatter is set to be phonetic? false
Before = 'Kai-shek Chiang'
After: formatter is set to be phonetic? true
After = ''
Formatting the phonetic component version after making the relationship bidirectional:
Before: formatter is set to be phonetic? false
Before = 'Kai-shek Chiang'
After: formatter is set to be phonetic? true
After = '介石 蒋'
```

UTF FTW

```
var chiangKaiShek😎 = PersonNameComponents()
chiangKaiShek😎.familyName = "蔣"
chiangKaiShek😎.givenName = "介(你好)石😅"
chiangKaiShek.phoneticRepresentation = chiangPhonetic
```

You Tee Eph

```
var chiangKaiShek😎 = PersonNameComponents()
chiangKaiShek😎.familyName = "蔣"
chiangKaiShek😎.givenName = "介(你好)石😅"
chiangKaiShek.phoneticRepresentation = chiangPhonetic

Components =      'givenName: 介(你好)石😅 familyName: 蔣 '
Default =         '蔣介(你好)石😅'
Long =            '蔣介(你好)石😅'
Medium =          '蔣介(你好)石😅'
Short =           '蔣介(你好)石😅'
Abbreviated =     '蔣'
```

Some Things to Take NSNote Of

PersonNameComponents vs NSPersonNameComponents

```
let ada = NSPersonNameComponents()
ada.givenName = "Augusta"
ada.middleName = "Ada"
ada.familyName = "Byron King"
ada.nameSuffix = "Countess of Lovelace"
ada.nickname = "Ada"
```

let comps = nsComponents as PersonNameComponents

```
Components =    '<NSPersonNameComponents: 0x6000008f8100> {
    givenName = Augusta, familyName = Byron King, middleName = Ada, namePrefix = (null),
    nameSuffix = Countess of Lovelace, nickname = Ada
    phoneticRepresentation = (null) }'
Default =   'Augusta Byron King'
Long =      'Augusta Ada Byron King Countess of Lovelace'
Medium =    'Augusta Byron King'
Short =     'Ada'
Abbreviated =   'AB'
```

From the Docs

Clauses, Limitations, Caveats, Cautions, and Provisos

From the Docs

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.

From the Docs

`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.

From the Docs

`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.

Questions? Answers?

Kevin Munc

@muncman