Structure

ListFormatStyle

A type that formats lists of items with a separator and conjunction appropriate for a given locale.

iOS 15.0+ | iPadOS 15.0+ | Mac Catalyst 15.0+ | macOS 12.0+ | tvOS 15.0+ | visionOS 1.0+ | watchOS 8.0+

```
struct ListFormatStyle<Style, Base> where Style : Format
Style, Base : Sequence, Style.FormatInput == Base.Element, Style.Format
Output == String
```

Overview

A list format style creates human readable text from a `Sequence` of values. Customize the formatting behavior of the list using the `width`, `listType`, and `locale` properties. The system automatically caches unique configurations of `ListFormatStyle` to enhance performance.

Use either `formatted()` or `formatted( :)`, both instance methods of `Sequence`, to create a string representation of the items.

The `formatted()` method applies the default list format style to a sequence of strings. For example:

```
["Kristin", "Paul", "Ana", "Bill"].formatted()
// Kristin, Paul, Ana, and Bill
```

You can customize a list's `type` and `width` properties.

- The `listType` property specifies the semantics of the list.
- The `width` property determines the size of the returned string.

The `formatted( :)` method to applies a custom list format style. You can use the static factory method `list(type:width:)` to create a custom list format style as a parameter to the method.

This example formats a sequence with a `ListFormatStyle.ListType.and` list type and `ListFormatStyle.Width.short` width:

```
["Kristin", "Paul", "Ana", "Bill"].formatted(.list(type: .and, width: .short))
// Kristin, Paul, Ana, & Bill
```

You can provide a member format style to transform each list element to a string in applications where the elements aren't already strings. For example, the following code sample uses an `IntegerFormatStyle` to convert a range of integer values into a list:

```
(5201719 ... 5201722).formatted(.list(memberStyle: IntegerFormatStyle(), type: .or,
// For locale: en_US: 5,201,719, 5,201,720, 5,201,721, or 5,201,722
// For locale: fr_CA: 5 201 719, 5 201 720, 5 201 721, ou 5 201 722
```

> **Note**
>
> The generated string is locale-dependent and incorporates linguistic and cultural conventions of the user.

You can create and reuse a list format style instance to format similar sequences. For example:

```
let percentStyle = ListFormatStyle<FloatingPointFormatStyle.Percent, StrideThrough<D
stride(from: 7.5, through: 9.0, by: 0.5).formatted(percentStyle)
// 7.5%, 8%, 8.5%, and 9%
stride(from: 89.0, through: 95.0, by: 2.0).formatted(percentStyle)
// 89%, 91%, 93%, and 95%
```

Topics

Creating a list format style

`init(memberStyle: Style)`

Creates an instance using the provided format style.

Modifying a list format style

`var width: ListFormatStyle<Style, Base>.Width`

The size of the list.

`enum Width`

The type representing the width of a list.

`var listType: ListFormatStyle<Style, Base>.ListType`

The type of the list.

`enum ListType`

A type that describes whether the returned list contains cumulative or alternative elements.

`var locale: Locale`

The locale to use when formatting items in the list.

Applying currency styles

`struct Currency`

A format style that converts between integer currency values and their textual representations.

Applying measurement styles

`struct FormatStyle`

A type that provides localized representations of measurements.

Relationships

Conforms To

`Decodable`
`Encodable`
`Equatable`
`FormatStyle`
`Hashable`
`Sendable`

`SendableMetatype`

See Also

Data formatting in Swift

{} Language Introspector

Converts data into human-readable text using formatters and locales.

`protocol FormatStyle`

A type that converts a given data type into a representation in another type, such as a string.

`struct IntegerFormatStyle`

A structure that converts between integer values and their textual representations.

`struct FloatingPointFormatStyle`

A structure that converts between floating-point values and their textual representations.

`struct FormatStyle`

A structure that converts between decimal values and their textual representations.

`struct StringStyle`

`struct FormatStyle`

A structure that converts between URL instances and their textual representations.

`struct FormatStyleCapitalizationContext`

The capitalization formatting context used when formatting dates and times.

Format Style Configurations

Behaviors for traits like numeric precision, rounding, and scale, used for formatting and parsing numeric values.