```
import Foundation
import
NaturalLanguage.NLContextualEmbedding
import NaturalLanguage.NLEmbedding
import NaturalLanguage.NLGazetteer
import NaturalLanguage.NLLanguage
import
NaturalLanguage.NLLanguageRecognizer
import NaturalLanguage.NLModel
import NaturalLanguage.NLScript
import NaturalLanguage.NLTagScheme
import NaturalLanguage.NLTagger
import NaturalLanguage.NLTokenizer
import _Concurrency
import _StringProcessing
import _SwiftConcurrencyShims

@available(macOS 10.14, iOS 12.0, watchOS
5.0, tvOS 12.0, *)
extension NLTokenizer {

    @nonobjc public func tokenRange(at
index: String.Index) ->
Range<String.Index>

    @available(macOS 11.0, iOS 14.0,
watchOS 7.0, tvOS 14.0, *)
    @nonobjc public func tokenRange(for
range: Range<String.Index>) ->
Range<String.Index>

    @nonobjc public func
enumerateTokens(in range:
```

```
Range<String.Index>, using block:
(Range<String.Index>,
NLTokenizer.Attributes) -> Bool)

    @nonobjc public func tokens(for
range: Range<String.Index>) ->
[Range<String.Index>]
}

@available(macOS 14.0, iOS 17.0, watchOS
10.0, tvOS 17.0, *)
extension NLContextualEmbeddingResult {

    @nonobjc public func tokenVector(at
index: String.Index) -> ([Double],
Range<String.Index>)?

    @nonobjc public func
enumerateTokenVectors(in range:
Range<String.Index>, using block:
([Double], Range<String.Index>) -> Bool)
}

@available(macOS 10.14, iOS 12.0, watchOS
5.0, tvOS 12.0, *)
extension NLTagger {

    @nonobjc public func tokenRange(at
index: String.Index, unit: NLTokenUnit)
-> Range<String.Index>

    @available(macOS 11.0, iOS 14.0,
watchOS 7.0, tvOS 14.0, *)
```

```
    @nonobjc public func tokenRange(for
range: Range<String.Index>, unit:
NLTokenUnit) -> Range<String.Index>

    @nonobjc public func tag(at index:
String.Index, unit: NLTokenUnit, scheme:
NLTagScheme) -> (NLTag?,
Range<String.Index>)

    @available(macOS 11.0, iOS 14.0,
watchOS 7.0, tvOS 14.0, *)
    @nonobjc public func tagHypotheses(at
index: String.Index, unit: NLTokenUnit,
scheme: NLTagScheme, maximumCount: Int)
-> ([String : Double],
Range<String.Index>)

    @nonobjc public func enumerateTags(in
range: Range<String.Index>, unit:
NLTokenUnit, scheme: NLTagScheme,
options: NLTagger.Options = [], using
block: (NLTag?, Range<String.Index>) ->
Bool)

    @nonobjc public func tags(in range:
Range<String.Index>, unit: NLTokenUnit,
scheme: NLTagScheme, options:
NLTagger.Options = []) -> [(NLTag?,
Range<String.Index>)]

    @nonobjc public func setLanguage(_
language: NLLanguage, range:
Range<String.Index>)
```

```
    @nonobjc public func setOrthography(_
orthography: NSOrthography, range:
Range<String.Index>)
}

@available(macOS 11.0, iOS 14.0, watchOS
7.0, tvOS 14.0, *)
extension NLModel {

    @nonobjc public func
predictedLabelHypotheses(for string:
String, maximumCount maxCount: Int) ->
[String : Double]

    @nonobjc public func
predictedLabelHypotheses(forTokens
tokens: [String], maximumCount maxCount:
Int) -> [[String : Double]]
}

@available(macOS 10.15, iOS 13.0, watchOS
6.0, tvOS 13.0, *)
extension NLEmbedding {

    @nonobjc public func distance(between
firstString: String, and secondString:
String, distanceType: NLDistanceType
= .cosine) -> NLDistance

    @nonobjc public func
enumerateNeighbors(for string: String,
maximumCount maxCount: Int, distanceType:
```

```
NLDistanceType = .cosine, using block:
(String, NLDistance) -> Bool)

    @nonobjc public func neighbors(for
string: String, maximumCount maxCount:
Int, distanceType: NLDistanceType
= .cosine) -> [(String, NLDistance)]

    @nonobjc public func
enumerateNeighbors(for vector: [Double],
maximumCount maxCount: Int, distanceType:
NLDistanceType = .cosine, using block:
(String, NLDistance) -> Bool)

    @nonobjc public func neighbors(for
vector: [Double], maximumCount maxCount:
Int, distanceType: NLDistanceType
= .cosine) -> [(String, NLDistance)]

    @nonobjc public func vector(for
string: String) -> [Double]?

    @nonobjc public class func write(_
dictionary: [String : [Double]],
language: NLLanguage?, revision: Int, to
url: URL) throws
}

@available(macOS 10.14, iOS 12.0, watchOS
5.0, tvOS 12.0, *)
extension NLLanguageRecognizer {

    @nonobjc public var languageHints:
```

```
[NLLanguage : Double]

    @nonobjc public func
languageHypotheses(withMaximum
maxHypotheses: Int) -> [NLLanguage :
Double]
}
```