```
import Foundation
import Symbols.NSSymbolEffect
import _Concurrency
import _StringProcessing
import _SwiftConcurrencyShims

/// A symbol effect that applies the
Appear animation to symbol images.
///
/// The Appear animation makes the symbol
visible either as a whole, or
/// one motion group at a time.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct AppearSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting an animation that
    /// appears upwards
    public var up: AppearSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that
    /// appears downwards.
    public var down: AppearSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
group.
```

```
    public var byLayer: 
AppearSymbolEffect { get }

    /// Returns a copy of the effect 
requesting an animation that
    /// applies to all motion groups 
simultaneously.
    public var wholeSymbol: 
AppearSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration: 
SymbolEffectConfiguration { get }

    /// Hashes the essential components 
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform 
to the `Hashable` protocol. The
    /// components used for hashing must 
be the same as the components compared
    /// in your type's `==` operator 
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your 
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the 
`hasher` instance provided,
    ///   or replace it with a different 
instance.
    ///   Doing so may become a compile-
```

```
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
AppearSymbolEffect, b:
AppearSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
```

```
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension AppearSymbolEffect :
TransitionSymbolEffect {
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension AppearSymbolEffect :
IndefiniteSymbolEffect {
}

/// The default symbol effect, resolves
to a particular effect in a
/// context-sensitive manner.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct AutomaticSymbolEffect :
SymbolEffect {

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }
```

```
    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
```

```
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
AutomaticSymbolEffect, b:
AutomaticSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension AutomaticSymbolEffect :
TransitionSymbolEffect {
}
```

```
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension AutomaticSymbolEffect :
ContentTransitionSymbolEffect {
}

/// A symbol effect that applies the
Bounce animation to
/// symbol images.
///
/// The Bounce animation applies a
transitory scaling effect to the symbol.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct BounceSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting an animation that
    /// bounces upwards.
    public var up: BounceSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that
    /// bounces downwards.
    public var down: BounceSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
```

```
group.
    public var byLayer:
BounceSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies to all motion groups
simultaneously.
    public var wholeSymbol:
BounceSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
```

```
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
BounceSymbolEffect, b:
BounceSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
```

```
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension BounceSymbolEffect :
IndefiniteSymbolEffect {
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension BounceSymbolEffect :
DiscreteSymbolEffect {
}

/// A symbol effect that applies the
Breathe animation to
/// symbol images.
///
/// The Breathe animation smoothly scales
a symbol up and down.
@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
public struct BreatheSymbolEffect :
SymbolEffect {
    /// Returns a copy of the effect
requesting an animation that
    /// pulses layers as they breathe.
    public var pulse: BreatheSymbolEffect
{ get }

    /// Returns a copy of the effect
requesting an animation that
    /// makes the symbol breathe with no
additional styling.
    public var plain: BreatheSymbolEffect
{ get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
group.
    public var byLayer:
BreatheSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies to all motion groups
simultaneously.
    public var wholeSymbol:
BreatheSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
```

```
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
BreatheSymbolEffect, b:
BreatheSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension BreatheSymbolEffect :
IndefiniteSymbolEffect {
}

@available(macOS 15.0, iOS 18.0, tvOS
```

```swift
18.0, watchOS 11.0, visionOS 2.0, *)
extension BreatheSymbolEffect :
DiscreteSymbolEffect {
}

/// A symbol effect that animates between
symbols or between different
/// configurations of the same symbol.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public protocol
ContentTransitionSymbolEffect {
}

/// A symbol effect that applies the
Disappear animation to symbol
/// images.
///
/// The Disappear animation makes the
symbol hidden either as a whole,
/// or one motion group at a time.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct DisappearSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting an animation that
    /// disappears upwards.
    public var up: DisappearSymbolEffect
{ get }

    /// Returns a copy of the effect
```

```
requesting an animation that
    /// disappears downwards.
    public var down:
DisappearSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
group.
    public var byLayer:
DisappearSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies to all motion groups
simultaneously.
    public var wholeSymbol:
DisappearSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
```

```
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
DisappearSymbolEffect, b:
DisappearSymbolEffect) -> Bool
```

```
    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension DisappearSymbolEffect :
TransitionSymbolEffect {
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension DisappearSymbolEffect :
IndefiniteSymbolEffect {
}

/// A symbol effect that can perform a
transient animation.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public protocol DiscreteSymbolEffect {
}

/// A symbol effect that continually
affects how a symbol is drawn
/// until it is disabled or removed.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public protocol IndefiniteSymbolEffect {
}

/// A symbol effect that applies the
Pulse animation to
/// symbol images.
///
/// The Pulse animation fades the opacity
of either all layers in
/// the symbol, or of a subset of the
layers in the symbol.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct PulseSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting an animation that only
    /// applies to the layers in each
symbol that have been marked to
    /// always pulse.
    public var byLayer: PulseSymbolEffect
{ get }
```

```
    /// Returns a copy of the effect
requesting an animation where all
    /// layers of the symbol pulse.
    public var wholeSymbol:
PulseSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
```

```
    ///   of this instance.
    public func hash(into hasher: inout 
Hasher)

    /// Returns a Boolean value 
indicating whether two values are equal.
    ///
    /// Equality is the inverse of 
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is 
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to 
compare.
    public static func == (a: 
PulseSymbolEffect, b: PulseSymbolEffect) 
-> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to 
be equal across different executions of
    /// your program. Do not save hash 
values to use during a future execution.
    ///
    /// - Important: `hashValue` is 
deprecated as a `Hashable` requirement. 
To
    ///   conform to `Hashable`, 
implement the `hash(into:)` requirement 
instead.
```

```
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension PulseSymbolEffect :
IndefiniteSymbolEffect {
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension PulseSymbolEffect :
DiscreteSymbolEffect {
}

/// A symbol effect that animates the
replacement of one symbol image
/// with another.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct ReplaceSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting the Down-Up variant of
    /// the Replace animation.
    ///
    /// The initial symbol scales down as
it is removed, and the new
    /// symbol scales up as it is added.
    public var downUp:
```

```
ReplaceSymbolEffect { get }

    /// Returns a copy of the effect
requesting the Up-Up variant of
    /// the Replace animation.
    ///
    /// The initial symbol scales up as
it is removed, and the new
    /// symbol scales up as it is added.
    public var upUp: ReplaceSymbolEffect
{ get }

    /// Returns a copy of the effect
requesting the Off-Up variant of
    /// the Replace animation.
    ///
    /// The initial symbol is removed
with no animation, and the new
    /// symbol scales up as it is added.
    public var offUp: ReplaceSymbolEffect
{ get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
group.
    public var byLayer:
ReplaceSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies to all motion groups
simultaneously.
```

```
    public var wholeSymbol:
ReplaceSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)
```

```
    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
ReplaceSymbolEffect, b:
ReplaceSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
```

```
}

extension ReplaceSymbolEffect {

    /// A symbol effect applies the
MagicReplace animation to
    /// symbol images.
    ///
    /// The MagicReplace effect animates
common elements across
    /// symbol images.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public struct MagicReplace :
SymbolEffect {

        /// The configuration for the
effect.
        public var configuration:
SymbolEffectConfiguration { get }

        /// Hashes the essential
components of this value by feeding them
into the
        /// given hasher.
        ///
        /// Implement this method to
conform to the `Hashable` protocol. The
        /// components used for hashing
must be the same as the components
compared
        /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
```

```
        /// with each of these
components.
        ///
        /// - Important: In your
implementation of `hash(into:)`,
        ///   don't call `finalize()` on
the `hasher` instance provided,
        ///   or replace it with a
different instance.
        ///   Doing so may become a
compile-time error in the future.
        ///
        /// - Parameter hasher: The
hasher to use when combining the
components
        ///   of this instance.
        public func hash(into hasher:
inout Hasher)

        /// Returns a Boolean value
indicating whether two values are equal.
        ///
        /// Equality is the inverse of
inequality. For any values `a` and `b`,
        /// `a == b` implies that `a !=
b` is `false`.
        ///
        /// - Parameters:
        ///   - lhs: A value to compare.
        ///   - rhs: Another value to
compare.
        public static func == (a:
ReplaceSymbolEffect.MagicReplace, b:
```

```
ReplaceSymbolEffect.MagicReplace) -> Bool

        /// The hash value.
        ///
        /// Hash values are not
guaranteed to be equal across different
executions of
        /// your program. Do not save
hash values to use during a future
execution.
        ///
        /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
        ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
        ///   The compiler provides an
implementation for `hashValue` for you.
        public var hashValue: Int { get }
    }

    /// Returns an effect preferring
MagicReplace and a configured
    /// ReplaceEffect if MagicReplace is
not possible.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public func magic(fallback:
ReplaceSymbolEffect) ->
ReplaceSymbolEffect.MagicReplace

    /// Returns an effect requesting the
```

```
Down-Up variant of
    /// the Replace animation.
    ///
    /// The initial symbol scales down as
it is removed, and the new
    /// symbol scales up as it is added.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public static var downUp:
ReplaceSymbolEffect { get }

    /// Returns an effect requesting the
Up-Up variant of
    /// the Replace animation.
    ///
    /// The initial symbol scales up as
it is removed, and the new
    /// symbol scales up as it is added.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public static var upUp:
ReplaceSymbolEffect { get }

    /// Returns an effect requesting the
Off-Up variant of
    /// the Replace animation.
    ///
    /// The initial symbol is removed
with no animation, and the new
    /// symbol scales up as it is added.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public static var offUp:
```

```
ReplaceSymbolEffect { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension ReplaceSymbolEffect :
ContentTransitionSymbolEffect {
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension
ReplaceSymbolEffect.MagicReplace :
ContentTransitionSymbolEffect {
}

/// A symbol effect that applies the
Rotate animation to
/// symbol images.
///
/// The Rotate animation rotates parts of
a symbol around a
/// symbol-provided anchor point.
@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
public struct RotateSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting an animation that
    /// rotates clockwise.
    public var clockwise:
RotateSymbolEffect { get }
```

```
    /// Returns a copy of the effect
requesting an animation that
    /// rotates counter-clockwise.
    public var counterClockwise:
RotateSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
group.
    public var byLayer:
RotateSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies to all motion groups
simultaneously.
    public var wholeSymbol:
RotateSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
```

```
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
RotateSymbolEffect, b:
```

```
RotateSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension RotateSymbolEffect :
IndefiniteSymbolEffect {
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension RotateSymbolEffect :
DiscreteSymbolEffect {
}

/// A symbol effect that scales symbol
```

```
images.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct ScaleSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting the scale state be set to the
up position.
    public var up: ScaleSymbolEffect {
get }

    /// Returns a copy of the effect
requesting the scale state be set to the
down position.
    public var down: ScaleSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
group.
    public var byLayer: ScaleSymbolEffect
{ get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies to all motion groups
simultaneously.
    public var wholeSymbol:
ScaleSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
ScaleSymbolEffect, b: ScaleSymbolEffect)
-> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
```

```
extension ScaleSymbolEffect :
IndefiniteSymbolEffect {
}

/// A presentation effect that can be
applied to an SFSymbol.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public protocol SymbolEffect : Hashable,
Sendable {

    /// The configuration for the effect.
    var configuration:
SymbolEffectConfiguration { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
PulseSymbolEffect {

    /// A symbol effect that applies the
Pulse animation to
    /// symbol images.
    ///
    /// The Pulse animation fades the
opacity of either all layers in
    /// the symbol, or of a subset of the
layers in the symbol.
    public static var pulse:
PulseSymbolEffect { get }
}
```

```
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
BounceSymbolEffect {

    /// A symbol effect that applies the
Bounce animation to
    /// symbol images.
    ///
    /// The Bounce animation applies a
transitory scaling effect to the
    /// symbol.
    public static var bounce:
BounceSymbolEffect { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
VariableColorSymbolEffect {

    /// A symbol effect that applies the
Variable Color animation to
    /// symbol images.
    ///
    /// The Variable Color animation
replaces the opacity of variable
    /// layers in the symbol by a
possibly repeating pattern that moves
    /// up and possibly back down the
variable layers. It has no effect
    /// for non-variable color symbol
images.
```

```
    public static var variableColor:
VariableColorSymbolEffect { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
ScaleSymbolEffect {

    /// A symbol effect that scales
symbol images.
    public static var scale:
ScaleSymbolEffect { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
AppearSymbolEffect {

    /// A symbol effect that applies the
Appear animation to symbol
    /// images.
    ///
    /// The Appear animation makes the
symbol visible either as a
    /// whole, or one motion group at a
time.
    public static var appear:
AppearSymbolEffect { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
DisappearSymbolEffect {

    /// A symbol effect that applies the
Disappear animation to symbol
    /// images.
    ///
    /// The Disappear animation makes the
symbol hidden either as a
    /// whole, or one motion group at a
time.
    public static var disappear:
DisappearSymbolEffect { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
ReplaceSymbolEffect {

    /// A symbol effect that animates the
replacement of one symbol
    /// image with another.
    public static var replace:
ReplaceSymbolEffect { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension SymbolEffect where Self ==
AutomaticSymbolEffect {
    /// The default symbol effect,
resolves to a particular effect in a
    /// context-sensitive manner.
    public static var automatic:
AutomaticSymbolEffect { get }
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension SymbolEffect where Self ==
WiggleSymbolEffect {

    /// A symbol effect that applies the
Wiggle animation to
    /// symbol images.
    ///
    /// The Wiggle animation applies a
transitory translation or rotation
    /// effect to the symbol.
    public static var wiggle:
WiggleSymbolEffect { get }
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension SymbolEffect where Self ==
RotateSymbolEffect {

    /// A symbol effect that applies the
Rotate animation to
    /// symbol images.
    ///
    /// The Rotate animation rotates
```

```
parts of a symbol around a
    /// symbol-provided anchor point.
    public static var rotate:
RotateSymbolEffect { get }
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension SymbolEffect where Self ==
BreatheSymbolEffect {

    /// A symbol effect that applies the
Breathe animation to
    /// symbol images.
    ///
    /// The Breathe animation smoothly
scales a symbol up and down.
    public static var breathe:
BreatheSymbolEffect { get }
}

/// A configuration for an SFSymbol
effect.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct SymbolEffectConfiguration :
Hashable, Sendable {

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
```

```
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
```

```
compare.
    public static func == (a:
SymbolEffectConfiguration, b:
SymbolEffectConfiguration) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

/// Options configuring how symbol
effects apply to symbol views.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct SymbolEffectOptions :
Hashable, Sendable {

    /// The default set of symbol effect
options.
    public static var `default`:
```

```
SymbolEffectOptions { get }

    /// Creates a set of symbol effect
options with a preferred speed
    /// multiplier.
    ///
    /// - Parameter speed: The preferred
speed multiplier to play the effect with.
    /// The default multiplier is `1.0`.
Very large or small values may
    /// be clamped.
    ///
    /// - Returns: A new set of symbol
effect options with the preferred speed
    /// multiplier.
    public static func speed(_ speed:
Double) -> SymbolEffectOptions

    /// Sets the preferred speed
multiplier of a set of symbol effect
    /// options.
    ///
    /// - Parameter speed: The preferred
speed multiplier to play the effect with.
    /// The default multiplier is `1.0`.
Very large or small values may
    /// be clamped.
    ///
    /// - Returns: A new set of symbol
effect options with the preferred speed
    /// multiplier.
    public func speed(_ speed: Double) ->
SymbolEffectOptions
```

```
    /// Creates a set of symbol effect
options with a preferred repeat
    /// count.
    ///
    /// - Parameter count: The preferred
number of times to play the
    ///   effect, or nil to request the
default number of repeats. Very
    ///   large or small values may be
clamped.
    ///
    /// - Returns: A new set of symbol
effect options with the preferred
    /// repeat count.
    @available(macOS, introduced: 14.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    @available(iOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    @available(tvOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    @available(watchOS, introduced: 10.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    @available(visionOS, introduced: 1.0,
deprecated: 100000.0, renamed:
```

```
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    public static func `repeat`(_ count:
Int?) -> SymbolEffectOptions

    /// Sets the preferred number of
times to play the effect.
    ///
    /// - Parameter count: The preferred
number of times to play the
    ///   effect, or nil to request the
default number of repeats. Very
    ///   large or small values may be
clamped.
    ///
    /// - Returns: A new set of symbol
effect options with the preferred
    /// repeat count.
    @available(macOS, introduced: 14.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    @available(iOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    @available(tvOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    @available(watchOS, introduced: 10.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
```

```
dic(_:delay:)")
    @available(visionOS, introduced: 1.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic(_:delay:)")
    public func `repeat`(_ count: Int?)
-> SymbolEffectOptions

    /// A set of symbol effect options
that prefers to repeat indefinitely.
    @available(macOS, introduced: 14.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(iOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(tvOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(watchOS, introduced: 10.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(visionOS, introduced: 1.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    public static var repeating:
SymbolEffectOptions { get }
```

```
    /// Returns a copy of the options
that prefers to repeat indefinitely.
    @available(macOS, introduced: 14.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(iOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(tvOS, introduced: 17.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(watchOS, introduced: 10.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    @available(visionOS, introduced: 1.0,
deprecated: 100000.0, renamed:
"SymbolEffectOptions.RepeatBehavior.perio
dic")
    public var repeating:
SymbolEffectOptions { get }

    /// A set of symbol effect options
that prefers not to repeat.
    public static var nonRepeating:
SymbolEffectOptions { get }

    /// Returns a copy of the options
that prefers not to repeat.
    public var nonRepeating:
```

```
SymbolEffectOptions { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
```

```
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
SymbolEffectOptions, b:
SymbolEffectOptions) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

extension SymbolEffectOptions {

    /// The behavior to use when
```

```
requesting any repetition on a
`SymbolEffect`.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public struct RepeatBehavior {

        /// A repeat behavior that
prefers to repeat indefinitely using
periodic animations.
        /// Periodic animations play the
effect at regular intervals starting and
stopping each time.
        ///
        /// - Returns: A new behavior
that prefers to repeat indefinitely using
periodic animation.
        public static var periodic:
SymbolEffectOptions.RepeatBehavior {
get }

        /// A repeat behavior with a
preferred play count and delay using
periodic animations.
        /// Periodic animations play the
effect at regular intervals starting and
stopping each time.
        ///
        /// - Parameter count: The
preferred number of times to play the
        ///   effect, or nil to request
it play indefinitely. Very
        ///   large or small values may
be clamped.
```

```
        ///
        /// - Parameter delay: The
preferred delay between repetitions,
        ///   in seconds, or nil to
request the default delay.
        ///
        /// - Returns: A new behavior
with the preferred
        /// play count and delay using
periodic animations.
        public static func periodic(_
count: Int? = nil, delay: Double? = nil)
-> SymbolEffectOptions.RepeatBehavior

        /// A repeat behavior that
prefers to repeat indefinitely, using
continuous animations if available.
        /// Continuous animations have an
intro, a body that runs as long as the
effect is enabled, and an outro.
        /// If available these animations
provide a smoother animation when an
effect repeats indefinitely.
        ///
        /// - Returns: A new behavior
that prefers to repeat indefinitely with
continuous animations.
        public static var continuous:
SymbolEffectOptions.RepeatBehavior {
get }
    }

    /// Creates a set of symbol effect
```

```
options that repeats with a preferred
behavior.
    ///
    /// - Parameter behavior: The
preferred behavior when the effect is
repeated.
    ///
    /// - Returns: A new set of symbol
effect options with the preferred
    /// repeat behavior.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public func `repeat`(_ behavior:
SymbolEffectOptions.RepeatBehavior) ->
SymbolEffectOptions

    /// Sets the preferred repeat
behavior.
    ///
    /// - Parameter behavior: The
preferred behavior when the effect is
repeated.
    ///
    /// - Returns: A new set of symbol
effect options with the preferred
    /// repeat behavior.
    @available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
    public static func `repeat`(_
behavior:
SymbolEffectOptions.RepeatBehavior) ->
SymbolEffectOptions
}
```

```
/// A symbol effect that animates a
symbol in or out.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public protocol TransitionSymbolEffect {
}

/// A symbol effect that applies the
Variable Color
/// animation to symbol images.
///
/// The Variable Color animation replaces
the opacity of variable
/// layers in the symbol by a possibly
repeating pattern that moves
/// up and possibly back down the
variable layers. It has no effect
/// for non-variable color symbol images.
@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
public struct VariableColorSymbolEffect :
SymbolEffect {

    /// Returns a copy of the effect
requesting that the Variable Color
    /// animation play in reverse each
time it repeats.
    public var reversing:
VariableColorSymbolEffect { get }

    /// Returns a copy of the effect
requesting that the Variable Color
```

```
    /// animation not play in reverse
each time it repeats.
    public var nonReversing:
VariableColorSymbolEffect { get }

    /// Returns a copy of the effect
requesting that the Variable Color
    /// animation applies its Cumulative
variant, where each sucessive
    /// variable layer is enabled and
stays enabled until the end of
    /// the animation cycle. This cancels
the `iterative` variant.
    public var cumulative:
VariableColorSymbolEffect { get }

    /// Returns a copy of the effect
requesting that the Variable Color
    /// animation applies its Iterative
variant, where each sucessive
    /// variable layer is active for a
short period of time and then
    /// inactive until the animation
cycle repeats. This cancels the
    /// `cumulative` variant.
    public var iterative:
VariableColorSymbolEffect { get }

    /// Returns a copy of the effect
requesting that the Variable
    /// Color animation hide inactive
layers completely, rather than
    /// drawing with reduced (but non-
zero) opacity.
    public var hideInactiveLayers:
VariableColorSymbolEffect { get }

    /// Returns a copy of the effect
requesting that the Variable Color
    /// animation draw inactive layers
with reduced (but non-zero)
    /// opacity.
    public var dimInactiveLayers:
VariableColorSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
```

```
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
VariableColorSymbolEffect, b:
VariableColorSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension VariableColorSymbolEffect :
IndefiniteSymbolEffect {
}

@available(macOS 14.0, iOS 17.0, tvOS
17.0, watchOS 10.0, visionOS 1.0, *)
extension VariableColorSymbolEffect :
DiscreteSymbolEffect {
}

/// A symbol effect that applies the
Wiggle animation to symbol images.
///
/// The Wiggle animation applies a
transitory translation or rotation effect
/// to the symbol.
@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
public struct WiggleSymbolEffect :
SymbolEffect {
    /// Returns a copy of the effect
requesting an animation that
    /// rotates back and forth, starting
by rotating clockwise.
    public var clockwise:
WiggleSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// rotates back and forth, starting
by rotating counter-clockwise.
    public var counterClockwise:
WiggleSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// moves back and forth
horizontally, starting by moving left.
    public var left: WiggleSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that
    /// moves back and forth
horizontally, starting by moving right.
    public var right: WiggleSymbolEffect
{ get }

    /// Returns a copy of the effect
requesting an animation that
    /// moves back and forth vertically,
starting by moving up.
```

```
    public var up: WiggleSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that
    /// moves back and forth vertically,
starting by moving down.
    public var down: WiggleSymbolEffect {
get }

    /// Returns a copy of the effect
requesting an animation that moves back
and forth
    /// horizontally based on the current
locale, starting by moving forward.
    public var forward:
WiggleSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that moves back
and forth
    /// horizontally based on the current
locale, starting by moving backward.
    public var backward:
WiggleSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that moves back
and forth
    /// along an axis with the passed in
angle.
    ///
    /// The angle is in degrees moving
```

```
clockwise from the positive x-axis.
    public func custom(angle: Double) ->
WiggleSymbolEffect

    /// Returns a copy of the effect
requesting an animation that
    /// applies separately to each motion
group.
    public var byLayer:
WiggleSymbolEffect { get }

    /// Returns a copy of the effect
requesting an animation that
    /// applies to all motion groups
simultaneously.
    public var wholeSymbol:
WiggleSymbolEffect { get }

    /// The configuration for the effect.
    public var configuration:
SymbolEffectConfiguration { get }

    /// Hashes the essential components
of this value by feeding them into the
    /// given hasher.
    ///
    /// Implement this method to conform
to the `Hashable` protocol. The
    /// components used for hashing must
be the same as the components compared
    /// in your type's `==` operator
implementation. Call `hasher.combine(_:)`
    /// with each of these components.
    ///
    /// - Important: In your
implementation of `hash(into:)`,
    ///   don't call `finalize()` on the
`hasher` instance provided,
    ///   or replace it with a different
instance.
    ///   Doing so may become a compile-
time error in the future.
    ///
    /// - Parameter hasher: The hasher to
use when combining the components
    ///   of this instance.
    public func hash(into hasher: inout
Hasher)

    /// Returns a Boolean value
indicating whether two values are equal.
    ///
    /// Equality is the inverse of
inequality. For any values `a` and `b`,
    /// `a == b` implies that `a != b` is
`false`.
    ///
    /// - Parameters:
    ///   - lhs: A value to compare.
    ///   - rhs: Another value to
compare.
    public static func == (a:
WiggleSymbolEffect, b:
WiggleSymbolEffect) -> Bool

    /// The hash value.
    ///
    /// Hash values are not guaranteed to
be equal across different executions of
    /// your program. Do not save hash
values to use during a future execution.
    ///
    /// - Important: `hashValue` is
deprecated as a `Hashable` requirement.
To
    ///   conform to `Hashable`,
implement the `hash(into:)` requirement
instead.
    ///   The compiler provides an
implementation for `hashValue` for you.
    public var hashValue: Int { get }
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension WiggleSymbolEffect :
IndefiniteSymbolEffect {
}

@available(macOS 15.0, iOS 18.0, tvOS
18.0, watchOS 11.0, visionOS 2.0, *)
extension WiggleSymbolEffect :
DiscreteSymbolEffect {
}
```