Structure

ConcentricRectangle

A shape that is replaced by a concentric version of the current container shape. If the container shape is a rectangle derived shape with four corners, this shape could choose to respect corners individually.

iOS 26.0+ | iPadOS 26.0+ | Mac Catalyst 26.0+ | macOS 26.0+ | tvOS 26.0+ | visionOS 26.0+ | watchOS 26.0+

```
struct ConcentricRectangle
```

Overview

To use `ConcentricRectangle`, first make sure the ancestor view hierarchy provides the container shape. Some container shapes are provided by platform kits, but you could provide your own by writing the `containerShape(_:)` modifier with a shape. Then, either call one of the initializers of `ConcentricRectangle` or use the static `Shape` convenience to create the concentric shape.

```
ZStack {
    Color.cyan
        .fill(.rect(corners: .concentric(minimum: 12), isUniform: false))
        .padding(.all, padding)
}
.containerShape(.rect(cornerRadius: 24))
```

The above example will generate a shape that is inset by `padding` on all sides. It will stay concentric to its container shape which has a 24 pts corner radius, so the shape will have a smaller radius with more padding. However, since here we provided a minimum value, the radius will be at least 12 pts, ensuring the shape to never have a square corner.

ConcentricRectangle also provides per-corner control with the initializers that take different styles for each corner.

```
ConcentricRectangle(
    topLeadingCorner: 0,
    topTrailingCorner: 28,
    bottomLeadingCorner: .concentric,
    bottomTrailingCorner: .concentric(minimum: 12)
)
```

The above example will have the top leading corner as square, the top trailing corner as constant 28 pts radius, the bottom leading as concentric to container shape, and bottom trailing as concentric with a minimum radius.

If you wish ConcentricRectangle to be symmetric on two corners, create the shape with initializers that takes an uniform corner style for any two corners on the rectangle. ConcentricRectangle will still resolves corner radius individually, but use the maximum value out of the corners as the final, shared corner radius of the uniform corners.

When the container shape provided is not a RoundedRectangularShape compatible shape, ConcentricRectangle will not be able to resolve corner radius based on concentricity. In those cases, it will fallback to be a ContainerRelativeShape, which is an inset version of the container shape.

See Also

ContainerRelativeShape.

See Also

Shape/rect(corners:isUniform:)), rect(uniformTopCorners:uniformBottomCorners:), rect(uniformLeadingCorners:uniformTrailingCorners:), rect(uniformTopCorners:bottomLeadingCorner:bottomTrailingCorner:), rect(uniformBottomCorners:topLeadingCorner:topTrailingCorner:), rect(uniformLeadingCorners:topTrailingCorner:bottomTrailingCorner:), rect(uniformTrailingCorners:topLeadingCorner:bottomLeadingCorner:).

Topics

Initializers

`init()`

Create a concentric rectangle with each corner being concentric individually to the container shape.

`init(corners: Edge.Corner.Style, isUniform: Bool)`

Create a rectangle with the same corner style set on four corners.

`init(topLeadingCorner: Edge.Corner.Style, topTrailingCorner: Edge.Corner.Style, bottomLeadingCorner: Edge.Corner.Style, bottomTrailingCorner: Edge.Corner.Style)`

Create a rectangle with individual styles of four corners.

`init(uniformBottomCorners: Edge.Corner.Style, topLeadingCorner: Edge.Corner.Style, topTrailingCorner: Edge.Corner.Style)`

Create a rectangle with a corner style set on the top two corners uniformly, and two other styles for the bottom two corners respectively.

`init(uniformLeadingCorners: Edge.Corner.Style, topTrailingCorner: Edge.Corner.Style, bottomTrailingCorner: Edge.Corner.Style)`

Create a rectangle with a corner style set on the leading two corners uniformly, and two other styles for the trailing two corners respectively.

`init(uniformLeadingCorners: Edge.Corner.Style, uniformTrailingCorners: Edge.Corner.Style)`

Create a rectangle with a corner style set on the leading two corners uniformly, and another style set on the trailing two corners uniformly.

`init(uniformTopCorners: Edge.Corner.Style, bottomLeadingCorner: Edge.Corner.Style, bottomTrailingCorner: Edge.Corner.Style)`

Create a rectangle with a corner style set on the top two corners uniformly, and two other styles for the bottom two corners respectively.

`init(uniformTopCorners: Edge.Corner.Style, uniformBottomCorners: Edge.Corner.Style)`

Create a rectangle with a corner style set on the top two corners uniformly, and another style set on the bottom two corners uniformly.

`init(uniformTrailingCorners: Edge.Corner.Style, topLeadingCorner: Edge.Corner.Style, bottomLeadingCorner: Edge.Corner.Style)`

Create a rectangle with a corner style set on the trailing two corners uniformly, and two other styles for the leading two corners respectively.

Relationships

Conforms To

`Animatable`
`Sendable`
`SendableMetatype`
`Shape`
`View`

See Also

Creating rectangular shapes

`struct Rectangle`

A rectangular shape aligned inside the frame of the view containing it.

`struct RoundedRectangle`

A rectangular shape with rounded corners, aligned inside the frame of the view containing it.

`enum RoundedCornerStyle`

Defines the shape of a rounded rectangle's corners.

`protocol RoundedRectangularShape`

A protocol of `InsettableShape` that describes a rounded rectangular shape.

`struct RoundedRectangularShapeCorners`

A type describing the corner styles of a `RoundedRectangularShape`.

`struct UnevenRoundedRectangle`

A rectangular shape with rounded corners with different values, aligned inside the frame of the view containing it.

`struct RectangleCornerRadii`

Describes the corner radius values of a rounded rectangle with uneven corners.

`struct RectangleCornerInsets`

The inset sizes for the corners of a rectangle.