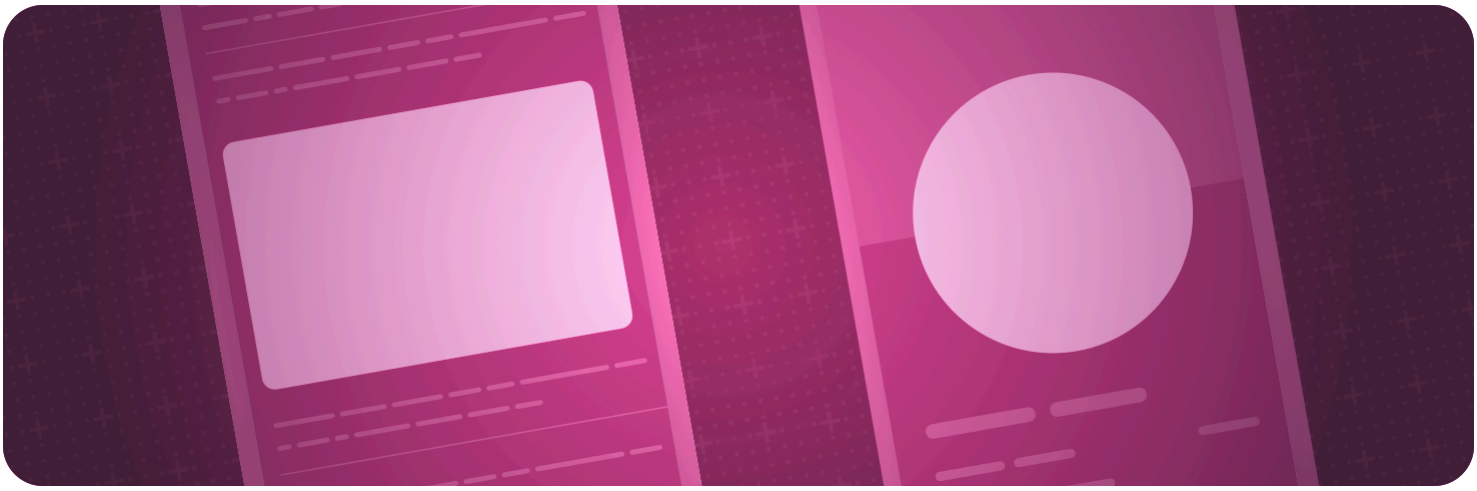
API Collection

Shapes

Trace and fill built-in and custom shapes with a color, gradient, or other pattern.

Overview

Draw shapes like circles and rectangles, as well as custom paths that define shapes of your own design. Apply styles that include environment-aware colors, rich gradients, and material effects to the foreground, background, and outline of your shapes.



If you need the efficiency or flexibility of immediate mode drawing — for example, to create particle effects — use a `Canvas` view instead.

Topics

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.

`struct` `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.

Creating circular shapes

`struct` `Circle`

A circle centered on the frame of the view containing it.

`struct` `Ellipse`

An ellipse aligned inside the frame of the view containing it.

`struct` `Capsule`

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

Drawing custom shapes

`struct` **Path**

The outline of a 2D shape.

Defining shape behavior

`protocol` **ShapeView**

A view that provides a shape that you can use for drawing operations.

`protocol` **Shape**

A 2D shape that you can use when drawing a view.

`struct` **AnyShape**

A type-erased shape value.

`enum` **ShapeRole**

Ways of styling a shape.

`struct` **StrokeStyle**

The characteristics of a stroke that traces a path.

`struct` **StrokeShapeView**

A shape provider that strokes its shape.

`struct` **StrokeBorderShapeView**

A shape provider that strokes the border of its shape.

`struct` **FillStyle**

A style for rasterizing vector shapes.

`struct` **FillShapeView**

A shape provider that fills its shape.

Transforming a shape

`struct` **ScaledShape**

A shape with a scale transform applied to it.

`struct` **RotatedShape**

A shape with a rotation transform applied to it.

`struct OffsetShape`

A shape with a translation offset transform applied to it.

`struct TransformedShape`

A shape with an affine transform applied to it.

Setting a container shape

`func containerShape(_:)`

Sets the container shape to use for any container relative shape or concentric rectangle within this view.

`protocol InsettableShape`

A shape type that is able to inset itself to produce another shape.

`struct ContainerRelativeShape`

A shape that is replaced by an inset version of the current container shape. If no container shape was defined, is replaced by a rectangle.

See Also

Views

- View fundamentals

  Define the visual elements of your app using a hierarchy of views.

- View configuration

  Adjust the characteristics of views in a hierarchy.

- View styles

  Apply built-in and custom appearances and behaviors to different types of views.

- Animations

  Create smooth visual updates in response to state changes.

- Text input and output

  Display formatted text and get text input from the user.

- Images

  Add images and symbols to your app's user interface.

- Controls and indicators

  Display values and get user selections.

- Menus and commands

  Provide space-efficient, context-dependent access to commands and controls.

- Drawing and graphics

  Enhance your views with graphical effects and customized drawings.