

[SwiftUI](#) / Images

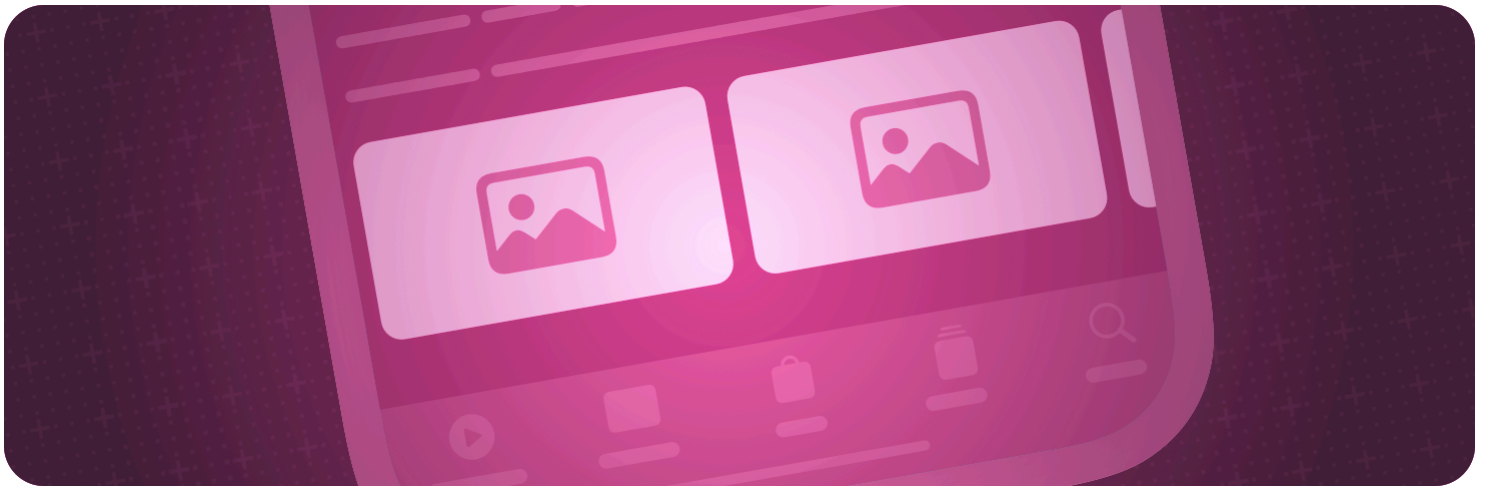
API Collection

Images

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

Overview

Display images, including [SF Symbols](#), images that you store in an asset catalog, and images that you store on disk, using an [Image](#) view.



For images that take time to retrieve — for example, when you load an image from a network endpoint — load the image asynchronously using [AsyncImage](#). You can instruct that view to display a placeholder during the load operation.

For design guidance, see [Images](#) in the Human Interface Guidelines.


Topics

Creating an image

```
struct Image
```

A view that displays an image.

Configuring an image

 Fitting images into available space

Adjust the size and shape of images in your app's user interface by applying view modifiers.

```
func imageScale(Image.Scale) -> some View
```

Scales images within the view according to one of the relative sizes available including small, medium, and large images sizes.

```
var imageScale: Image.Scale
```

The image scale for this environment.

```
enum Scale
```

A scale to apply to vector images relative to text.

```
enum Orientation
```

The orientation of an image.

```
enum ResizingMode
```

The modes that SwiftUI uses to resize an image to fit within its containing view.

Loading images asynchronously

```
struct AsyncImage
```

A view that asynchronously loads and displays an image.

```
enum AsyncImagePhase
```

The current phase of the asynchronous image loading operation.

Setting a symbol variant

```
func symbolVariant(SymbolVariants) -> some View
```

Makes symbols within the view show a particular variant.

```
var symbolVariants: SymbolVariants
```

The symbol variant to use in this environment.

```
struct SymbolVariants
```

A variant of a symbol.

Managing symbol effects

```
func symbolEffect<T>(T, options: SymbolEffectOptions, isActive: Bool) -> some View
```

Returns a new view with a symbol effect added to it.

```
func symbolEffect<T, U>(T, options: SymbolEffectOptions, value: U) -> some View
```

Returns a new view with a symbol effect added to it.

```
func symbolEffectsRemoved(Bool) -> some View
```

Returns a new view with its inherited symbol image effects either removed or left unchanged.

```
struct SymbolEffectTransition
```

Creates a transition that applies the Appear, Disappear, DrawOn or DrawOff symbol animation to symbol images within the inserted or removed view hierarchy.

Setting symbol rendering modes

```
func symbolRenderingMode(SymbolRenderingMode?) -> some View
```

Sets the rendering mode for symbol images within this view.

```
var symbolRenderingMode: SymbolRenderingMode?
```

The current symbol rendering mode, or nil denoting that the mode is picked automatically using the current image and foreground style as parameters.

```
struct SymbolRenderingMode
```

A symbol rendering mode.

```
struct SymbolColorRenderingMode
```

A method of filling a layer in a symbol image.

```
struct SymbolVariableValueMode
```

A method of rendering the variable value of a symbol image.

Rendering images from views

`class ImageRenderer`

An object that creates images from SwiftUI views.

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.
- ☰ Controls and indicators
Display values and get user selections.
- ☰ Menus and commands
Provide space-efficient, context-dependent access to commands and controls.
- ☰ Shapes
Trace and fill built-in and custom shapes with a color, gradient, or other pattern.
- ☰ Drawing and graphics
Enhance your views with graphical effects and customized drawings.