

[SwiftUI](#) / Accessible controls

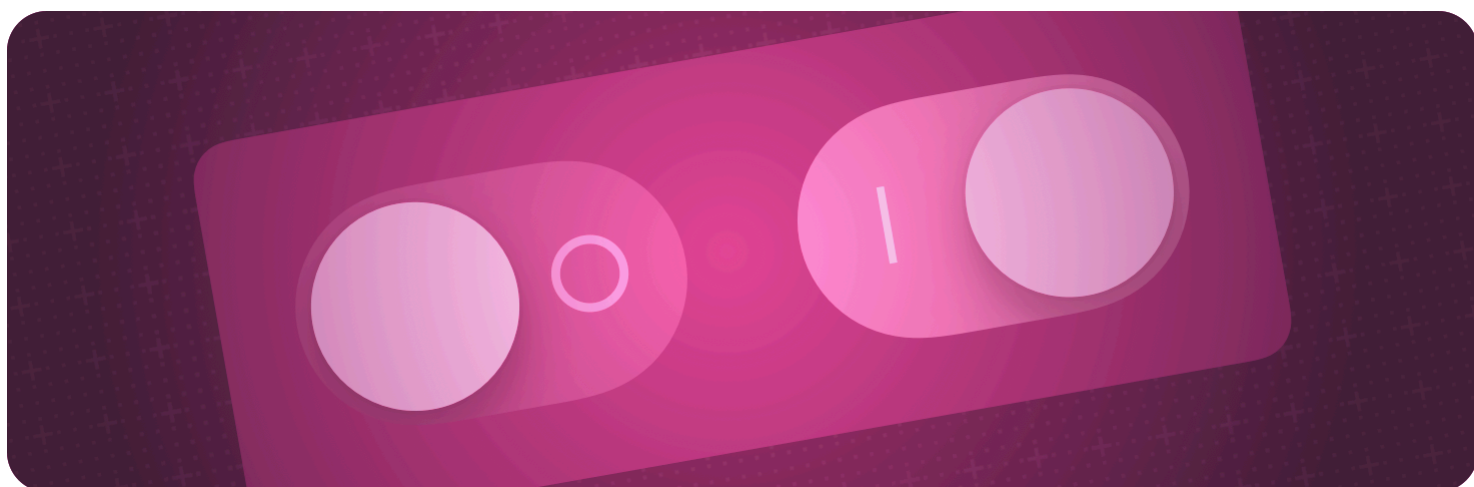
API Collection

# Accessible controls

Improve access to actions that your app can undertake.

## Overview

Help people using assistive technologies to gain access to controls in your app.



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

## Topics

### Adding actions to views

```
func accessibilityAction(AccessibilityActionKind, () -> Void) ->  
    ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Adds an accessibility action to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action.

```
func accessibilityActions<Content>(() -> Content) -> some View
```

Adds multiple accessibility actions to the view.

```
func accessibilityAction(named:_:)
```

Adds an accessibility action to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action.

```
func accessibilityAction<Label>(action: () -> Void, label: () -> Label) -> some View
```

Adds an accessibility action to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action.

```
func accessibilityAction<I, Label>(intent: I, label: () -> Label) -> some View
```

Adds an accessibility action labeled by the contents of `label` to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action. When the action is performed, the `intent` will be invoked.

```
func accessibilityAction<I>(AccessibilityActionKind, intent: I) -> ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Adds an accessibility action representing `actionKind` to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action. When the action is performed, the `intent` will be invoked.

```
func accessibilityAction(named:intent:)
```

Adds an accessibility action labeled name to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action. When the action is performed, the `intent` will be invoked.

```
func accessibilityAdjustableAction((AccessibilityAdjustmentDirection) -> Void) -> ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Adds an accessibility adjustable action to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action.

```
func accessibilityScrollAction((Edge) -> Void) -> ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Adds an accessibility scroll action to the view. Actions allow assistive technologies, such as the VoiceOver, to interact with the view by invoking the action.

```
func accessibilityActions<Content>(category: AccessibilityAction
Category, () -> Content) -> some View
```

Adds multiple accessibility actions to the view with a specific category. Actions allow assistive technologies, such as VoiceOver, to interact with the view by invoking the action and are grouped by their category. When multiple action modifiers with an equal category are applied to the view, the actions are combined together.

```
struct AccessibilityActionKind
```

The structure that defines the kinds of available accessibility actions.

```
enum AccessibilityAdjustmentDirection
```

A directional indicator you use when making an accessibility adjustment.

```
struct AccessibilityActionCategory
```

Designates an accessibility action category that is provided and named by the system.

## Offering Quick Actions to people

```
func accessibilityQuickAction<Style, Content>(style: Style, content: ()
-> Content) -> some View
```

Adds a quick action to be shown by the system when active.

```
func accessibilityQuickAction<Style, Content>(style: Style, isActive:
Binding<Bool>, content: () -> Content) -> some View
```

Adds a quick action to be shown by the system when active.

```
protocol AccessibilityQuickActionStyle
```

A type that describes the presentation style of an accessibility quick action.

## Making gestures accessible

```
func accessibilityActivationPoint(_:)
```

The activation point for an element is the location assistive technologies use to initiate gestures.

```
func accessibilityActivationPoint(_:isEnabled:)
```

The activation point for an element is the location assistive technologies use to initiate gestures.

```
func accessibilityDragPoint(_:description:)
```

The point an assistive technology should use to begin a drag interaction.

```
func accessibilityDragPoint(_:description:isEnabled:)
```

The point an assistive technology should use to begin a drag interaction.

```
func accessibilityDropPoint(_:description:)
```

The point an assistive technology should use to end a drag interaction.

```
func accessibilityDropPoint(_:description:isEnabled:)
```

The point an assistive technology should use to end a drag interaction.

```
func accessibilityDirectTouch(Bool, options: AccessibilityDirectTouchOptions) -> ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Explicitly set whether this accessibility element is a direct touch area. Direct touch areas passthrough touch events to the app rather than being handled through an assistive technology, such as VoiceOver. The modifier accepts an optional `AccessibilityDirectTouchOptions` option set to customize the functionality of the direct touch area.

```
func accessibilityZoomAction((AccessibilityZoomGestureAction) -> Void) -> ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Adds an accessibility zoom action to the view. Actions allow assistive technologies, such as VoiceOver, to interact with the view by invoking the action.

```
struct AccessibilityDirectTouchOptions
```

An option set that defines the functionality of a view's direct touch area.

```
struct AccessibilityZoomGestureAction
```

Position and direction information of a zoom gesture that someone performs with an assistive technology like VoiceOver.

## Controlling focus

```
func accessibilityFocused(AccessibilityFocusState<Bool>.Binding) -> some View
```

Modifies this view by binding its accessibility element's focus state to the given boolean state value.

```
func accessibilityFocused<Value>(AccessibilityFocusState<Value>.Binding, equals: Value) -> some View
```

Modifies this view by binding its accessibility element's focus state to the given state value.

```
struct AccessibilityFocusState
```

A property wrapper type that can read and write a value that SwiftUI updates as the focus of any active accessibility technology, such as VoiceOver, changes.

## Managing interactivity

```
func accessibilityRespondsToUserInteraction(Bool) -> ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Explicitly set whether this Accessibility element responds to user interaction and would thus be interacted with by technologies such as Switch Control, Voice Control or Full Keyboard Access.

```
func accessibilityRespondsToUserInteraction(Bool, isEnabled: Bool) -> ModifiedContent<Self, AccessibilityAttachmentModifier>
```

Explicitly set whether this Accessibility element responds to user interaction and would thus be interacted with by technologies such as Switch Control, Voice Control or Full Keyboard Access.

---

## See Also

### Accessibility

- ☰ Accessibility fundamentals  
Make your SwiftUI apps accessible to everyone, including people with disabilities.
- ☰ Accessible appearance  
Enhance the legibility of content in your app's interface.
- ☰ Accessible descriptions  
Describe interface elements to help people understand what they represent.
- ☰ Accessible navigation  
Enable users to navigate to specific user interface elements using rotors.