

[UIKit](#) / Accessibility for UIKit

API Collection

Accessibility for UIKit

Make your UIKit apps accessible to everyone who uses iOS and tvOS.

Overview

Making your app accessible means making it usable by everyone. By designing your app with accessibility in mind, you make it possible for everyone to enjoy your app. For more information, see [Accessibility](#).

UIKit controls and views come with built-in accessibility, providing an accessible user experience by default. Typically, you don't need to do extra work to enable the standard accessibility features.

In some cases, you might want to modify the default values to better represent your app, to provide additional context, or to modify the user's flow through the app. UIKit makes these customizations straightforward, involving a few lines of code or Interface Builder adjustments as you define your user interface. For more information about customizing accessibility for UIKit elements, see [UIAccessibility](#).

If your app contains custom user interface elements that don't inherit from [UIView](#) or one of the other UIKit classes with built-in accessibility, make those elements accessible by subclassing [UIAccessibilityElement](#).

If you build your app with SwiftUI, see [Accessibility modifiers](#).

Topics

Essentials

☰ [UIAccessibility](#)

A set of methods that provides accessibility information about views and controls in an app's user interface.

`UIAccessibilityContainer`

Provide a set of methods that view subclasses use to make subcomponents accessible as separate elements.

Supporting VoiceOver in your app

Add VoiceOver support to make your iOS app more accessible to users who are blind or have low vision.

Behaviors

`UIAccessibilityFocus`

An informal protocol that provides a way to determine whether an assistive app, such as VoiceOver, has focus on an accessible element.

`protocol UIAccessibilityIdentification`

Methods that associate a unique identifier with elements in your user interface.

`protocol UIAccessibilityReadingContent`

Methods to implement for an object that represents content that users read, such as a book or an article.

`protocol UIAccessibilityContentSizeCategoryImageAdjusting`

Methods to determine when to adjust images for different content size categories.

`struct UIAccessibilityTextualContext`

Constants that describe a named context that helps identify and classify the type of text inside an element.

Guided Access

```
static func configureForGuidedAccess(features: UIGuidedAccessAccessibilityFeature, enabled: Bool, completionHandler: (Bool, (any Error)?) -> Void)
```

Enables or disables the specified accessibility features while using Guided Access.

`struct UIGuidedAccessAccessibilityFeature`

Constants that describe accessibility features for Guided Access.

enum Code

Error codes for Guided Access.

Actions



UIAccessibilityAction

A set of methods that accessibility elements can use to support specific actions.

class UIAccessibilityCustomAction

A custom action to perform on an accessible object.

typealias Handler

A closure type that defines a handler to perform for an action.



Delivering an exceptional accessibility experience

Make improvements to your app's interaction model to support assistive technologies such as VoiceOver.

Elements

class UIAccessibilityElement

An element that should be accessible to users with disabilities, but that isn't accessible by default.

protocol UIScrollViewAccessibilityDelegate

A set of methods you can implement to provide accessibility information for a scroll view.

protocol UIPickerViewAccessibilityDelegate

A set of methods you can implement to provide accessibility information for individual components of a picker view.

Containers

protocol UIAccessibilityContainerDataTable

Methods that convey information about the contents of a table.

protocol UIAccessibilityContainerDataTableCell

Methods that provide the location of a cell in a table.

enum UIAccessibilityContainerType

Constants that indicate the type of content in a data-based container.

Navigation

`class UIAccessibilityCustomRotor`

A context-sensitive function that helps VoiceOver users find the next instance of a related element.

`class UIAccessibilityCustomRotorItemResult`

A target element that a custom rotor references.

`class UIAccessibilityCustomRotorSearchPredicate`

The search parameters that help determine the next matching custom rotor item result.

Drag and drop support

`class UIAccessibilityLocationDescriptor`

An accessibility descriptor for a specific geometric point of interest within a view, for use by assistive apps.

Notifications

⋮ Notification names

The names of notifications that the accessibility system generates.

⋮ Notification dictionary keys

Handle notifications with keys in the user info dictionary.

`static func post(notification: UIAccessibility.Notification, argument: Any?)`

Posts a notification to assistive apps.

Conversions

`static func convertToScreenCoordinates(CGRect, in: UIView) -> CGRect`

Converts the specified rectangle from view coordinates to screen coordinates.

`static func convertToScreenCoordinates(UIBezierPath, in: UIView) -> UIBezierPath`

Converts the specified path object to screen coordinates and returns a new path object with the results.

Convenience functions

```
static func focusedElement(using: UIAccessibility.AssistiveTechnologyIdentifier?) -> Any?
```

Returns the accessibility element that's currently in focus by the specified assistive app.

```
static var hearingDevicePairedEar: UIAccessibility.HearingDeviceEar
```

The current pairing status of Made for iPhone hearing devices.

```
struct HearingDeviceEar
```

Constants that specify how a person is using a hearing device.

```
static func registerGestureConflictWithZoom()
```

Warns users that app-specific gestures conflict with the system-defined Zoom accessibility gestures.

```
static func requestGuidedAccessSession(enabled: Bool, completionHandler : (Bool) -> Void)
```

Transitions the app to or from Single App mode asynchronously.

```
static func zoomFocusChanged(zoomType: UIAccessibility.ZoomType, to Frame: CGRect, in: UIView)
```

Notifies the system when the app's focus changes to a new location.

Capabilities

```
static var isAssistiveTouchRunning: Bool
```

A Boolean value that indicates whether AssistiveTouch is in an enabled state.

```
static var isVoiceOverRunning: Bool
```

A Boolean value that indicates whether VoiceOver is in an enabled state.

```
static var isSwitchControlRunning: Bool
```

A Boolean value that indicates whether the Switch Control setting is in an enabled state.

```
static var isShakeToUndoEnabled: Bool
```

A Boolean value that indicates whether the Shake to Undo setting is in an enabled state.

`static var isClosedCaptioningEnabled: Bool`

A Boolean value that indicates whether the Closed Captions + SDH setting is in an enabled state.

`static var isBoldTextEnabled: Bool`

A Boolean value that indicates whether the Bold Text setting is in an enabled state.

`static var isDarkerSystemColorsEnabled: Bool`

A Boolean value that indicates whether the Increase Contrast setting is in an enabled state.

`static var isGrayscaleEnabled: Bool`

A Boolean value that indicates whether the Color Filters and the Grayscale settings are in an enabled state.

`static var isGuidedAccessEnabled: Bool`

A Boolean value that indicates whether the Guided Access setting is in an enabled state.

`static var isInvertColorsEnabled: Bool`

A Boolean value that indicates whether the Classic Invert setting is in an enabled state.

`static var isMonoAudioEnabled: Bool`

A Boolean value that indicates whether the Mono Audio setting is in an enabled state.

`static var isReduceMotionEnabled: Bool`

A Boolean value that indicates whether the Reduce Motion setting is in an enabled state.

`static var isReduceTransparencyEnabled: Bool`

A Boolean value that indicates whether the Reduce Transparency setting is in an enabled state.

`static var isSpeakScreenEnabled: Bool`

A Boolean value that indicates whether the Speak Screen setting is in an enabled state.

`static var isSpeakSelectionEnabled: Bool`

A Boolean value that indicates whether the Speak Selection setting is in an enabled state.

`static var isOnOffSwitchLabelsEnabled: Bool`

A Boolean value that indicates whether the On/Off Labels setting is in an enabled state.

`static var isVideoAutoplayEnabled: Bool`

A Boolean value that indicates whether the Auto-Play Video Previews setting is in an enabled state.

~~static var buttonShapesEnabled: Bool~~

A Boolean value that indicates whether the Button Shapes setting is in an enabled state.

Deprecated

static var prefersCrossFadeTransitions: Bool

A Boolean value that indicates whether the Reduce Motion and the Prefer Cross-Fade Transitions settings are in an enabled state.

static var shouldDifferentiateWithoutColor: Bool

A Boolean value that indicates whether the Differentiate Without Color setting is in an enabled state.

See Also

User interactions

☰ Touches, presses, and gestures

Encapsulate your app's event-handling logic in gesture recognizers so that you can reuse that code throughout your app.

☰ Menus and shortcuts

Simplify interactions with your app using menu systems, contextual menus, Home Screen quick actions, and keyboard shortcuts.

☰ Drag and drop

Bring drag and drop to your app by using interaction APIs with your views.

☰ Pointer interactions

Support pointer interactions in your custom controls and views.

☰ Apple Pencil interactions

Handle user interactions like double tap and squeeze on Apple Pencil.

☰ Focus-based navigation

Navigate the interface of your UIKit app using a remote, game controller, or keyboard.