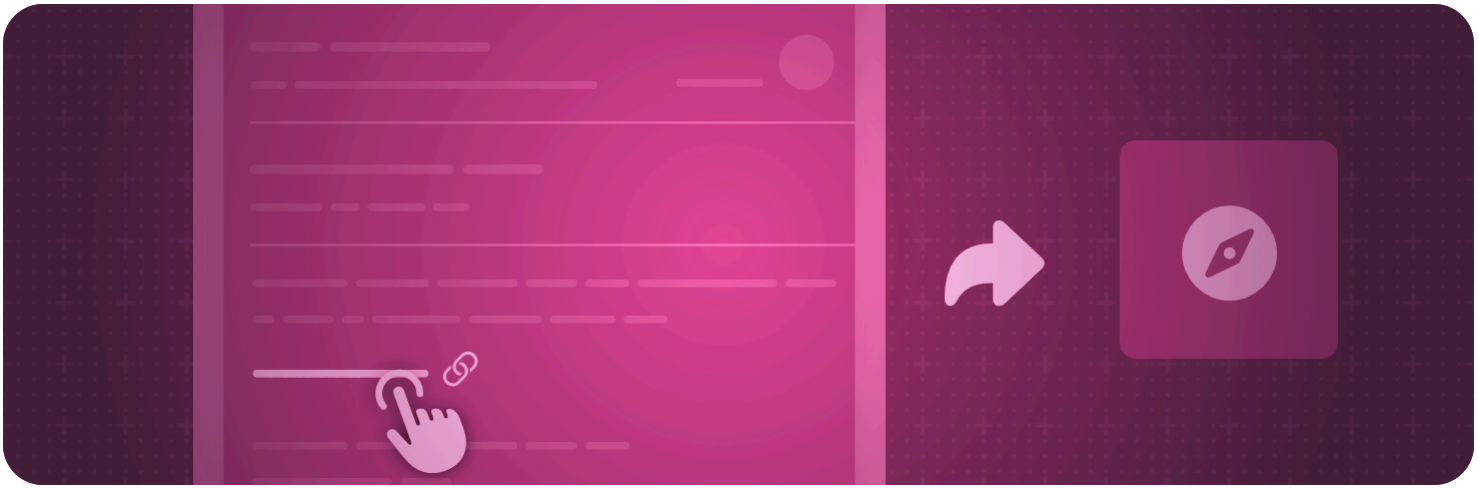
API Collection

System events

React to system events, like opening a URL.

Overview

Specify view and scene modifiers to indicate how your app responds to certain system events. For example, you can use the `onOpenURL(perform:)` view modifier to define an action to take when your app receives a universal link, or use the `backgroundTask(_:action:)` scene modifier to specify an asynchronous task to carry out in response to a background task event, like the completion of a background URL session.



Topics

Sending and receiving user activities

{} Restoring your app's state with SwiftUI

Provide app continuity for users by preserving their current activities.

```
func userActivity<P>(String, element: P?, (P, NSUserActivity) -> ()) -> some View
```

Advertises a user activity type.

```
func userActivity(String, isActive: Bool, (NSUserActivity) -> ()) -> some View
```

Advertises a user activity type.

```
func onContinueUserActivity(String, perform: (NSUserActivity) -> ()) -> some View
```

Registers a handler to invoke in response to a user activity that your app receives.

Sending and receiving URLs

```
var openURL: OpenURLAction
```

An action that opens a URL.

```
struct OpenURLAction
```

An action that opens a URL.

```
func onOpenURL(perform: (URL) -> ()) -> some View
```

Registers a handler to invoke in response to a URL that your app receives.

Handling external events

```
func handlesExternalEvents(matching: Set<String>) -> some Scene
```

Specifies the external events for which SwiftUI opens a new instance of the modified scene.

```
func handlesExternalEvents(preferring: Set<String>, allowing: Set<String>) -> some View
```

Specifies the external events that the view's scene handles if the scene is already open.

Handling background tasks

```
func backgroundTask<D, R>(BackgroundTask<D, R>, action: (D) async -> R) -> some Scene
```

Runs the specified action when the system provides a background task.

```
struct BackgroundTask
```

The kinds of background tasks that your app or extension can handle.

`struct` SnapshotData

The associated data of a snapshot background task.

`struct` SnapshotResponse

Your application's response to a snapshot background task.

Importing and exporting transferable items

`func` importableFromServices<T>(for: T.Type, action: ([T]) -> Bool) -> some View

Enables importing items from services, such as Continuity Camera on macOS.

`func` exportableToServices<T>(@autoclosure () -> [T]) -> some View

Exports items for consumption by shortcuts, quick actions, and services.

`func` exportableToServices<T>(@autoclosure () -> [T], onEdit: ([T]) -> Bool) -> some View

Exports read-write items for consumption by shortcuts, quick actions, and services.

Importing and exporting using item providers

`func` importsItemProviders([UTType], onImport: ([NSItemProvider]) -> Bool) -> some View

Enables importing item providers from services, such as Continuity Camera on macOS.

`func` exportsItemProviders([UTType], onExport: () -> [NSItemProvider]) -> some View

Exports a read-only item provider for consumption by shortcuts, quick actions, and services.

`func` exportsItemProviders([UTType], onExport: () -> [NSItemProvider], onEdit: ([NSItemProvider]) -> Bool) -> some View

Exports a read-write item provider for consumption by shortcuts, quick actions, and services.

See Also

Event handling

Gestures

Define interactions from taps, clicks, and swipes to fine-grained gestures.

- Input events

  Respond to input from a hardware device, like a keyboard or a Touch Bar.

- Clipboard

  Enable people to move or duplicate items by issuing Copy and Paste commands.

- Drag and drop

  Enable people to move or duplicate items by dragging them from one location to another.

- Focus

  Identify and control which visible object responds to user interaction.