SwiftUI / Drag and drop

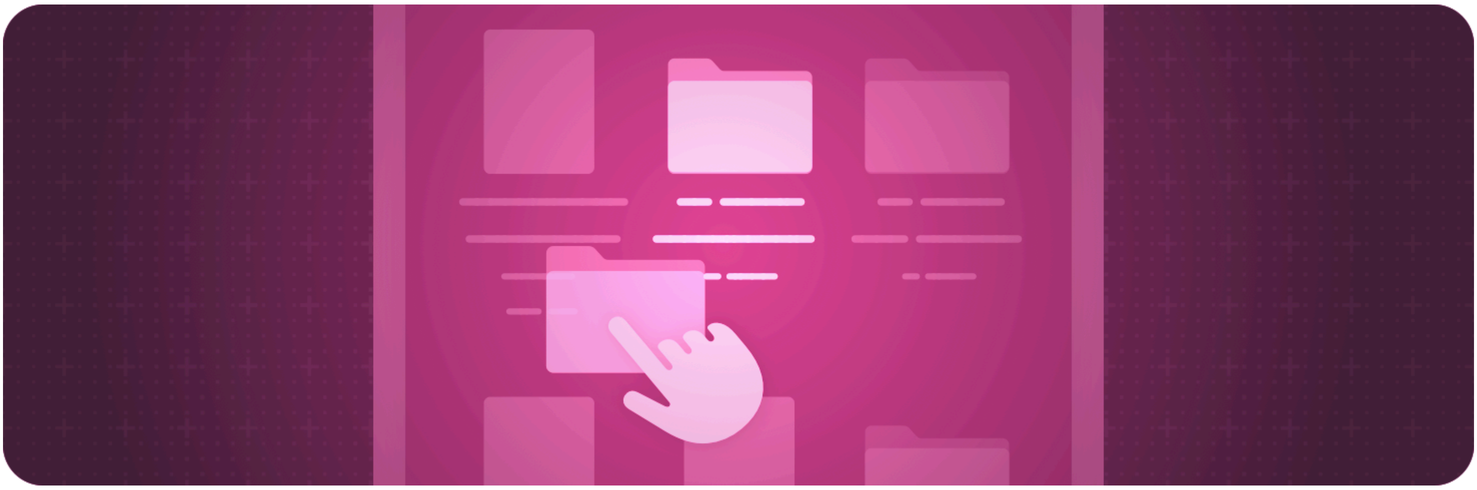
API Collection

Drag and drop

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

Overview

Drag and drop offers people a convenient way to move content from one part of your app to another, or from one app to another, using an intuitive dragging gesture. Support this feature in your app by adding view modifiers to potential source and destination views within your app's interface.



In your modifiers, provide or accept types that conform to the `Transferable` protocol, or that inherit from the `NSItemProvider` class. When possible, prefer using transferable items.

For design guidance, see Drag and drop in the Human Interface Guidelines.

Topics

Essentials

Adopting drag and drop using SwiftUI

Enable drag-and-drop interactions in lists, tables and custom views.

Making a view into a drag source

Adopt draggable API to provide items for drag-and-drop operations.

Configuring drag and drop behavior

`struct DragConfiguration`

The behavior of the drag, proposed by the dragging source.

`struct DropConfiguration`

Describes the behavior of the drop.

Moving items

`struct DragSession`

Describes the ongoing dragging session.

`struct DropSession`

Moving transferable items

`func draggable<T>(@autoclosure () -> T) -> some View`

Activates this view as the source of a drag and drop operation.

`func draggable<V, T>(@autoclosure () -> T, preview: () -> V) -> some View`

Activates this view as the source of a drag and drop operation.

~~func dropDestination<T>(for: T.Type, action: ([T], CGPoint) -> Bool, isTargeted: (Bool) -> Void) -> some View~~

Defines the destination of a drag and drop operation that handles the dropped content with a closure that you specify.

Deprecated

Moving items using item providers

```
func itemProvider(Optional<() -> NSItemProvider?>) -> some View
```

Provides a closure that vends the drag representation to be used for a particular data element.

```
func onDrag<V>(() -> NSItemProvider, preview: () -> V) -> some View
```

Activates this view as the source of a drag and drop operation.

```
func onDrag(() -> NSItemProvider) -> some View
```

Activates this view as the source of a drag and drop operation.

```
func onDrop(of:isTargeted:perform:)
```

Defines the destination of a drag-and-drop operation that handles the dropped content with a closure that you specify.

```
func onDrop(of:delegate:)
```

Defines the destination of a drag and drop operation using behavior controlled by the delegate that you provide.

```
protocol DropDelegate
```

An interface that you implement to interact with a drop operation in a view modified to accept drops.

```
struct DropProposal
```

The behavior of a drop.

```
enum DropOperation
```

Operation types that determine how a drag and drop session resolves when the user drops a drag item.

```
struct DropInfo
```

The current state of a drop.

Describing preview formations

```
struct DragDropPreviewsFormation
```

On macOS, describes the way the dragged previews are visually composed. Both drag sources and drop destination can specify their desired preview formation.

Configuring spring loading

```
func springLoadingBehavior(SpringLoadingBehavior) -> some View
```

Sets the spring loading behavior this view.

```
var springLoadingBehavior: SpringLoadingBehavior
```

The behavior of spring loaded interactions for the views associated with this environment.

```
struct SpringLoadingBehavior
```

The options for controlling the spring loading behavior of views.

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.

Focus

Identify and control which visible object responds to user interaction.

System events

React to system events, like opening a URL.