SwiftUI / Menu

Structure

Menu

A control for presenting a menu of actions.

iOS 14.0+ | iPadOS 14.0+ | Mac Catalyst 14.0+ | macOS 11.0+ | tvOS 17.0+ | visionOS 1.0+

```
struct Menu<Label, Content> where Label : View, Content : View
```

Mentioned in

- Building and customizing the menu bar with SwiftUI
- Populating SwiftUI menus with adaptive controls

Overview

The following example presents a menu of three buttons and a submenu, which contains three buttons of its own.

```
Menu("Actions") {
    Button("Duplicate", action: duplicate)
    Button("Rename", action: rename)
    Button("Delete…", action: delete)
    Menu("Copy") {
        Button("Copy", action: copy)
        Button("Copy Formatted", action: copyFormatted)
        Button("Copy Library Path", action: copyPath)
    }
}
```

You can create the menu's title with a LocalizedStringKey, as seen in the previous example, or with a view builder that creates multiple views, such as an image and a text view:

```
Menu {
    Button("Open in Preview", action: openInPreview)
    Button("Save as PDF", action: saveAsPDF)
} label: {
    Label("PDF", systemImage: "doc.fill")
}
```

To support subtitles on menu items, initialize your `Button` with a view builder that creates multiple `Text` views where the first text represents the title and the second text represents the subtitle. The same approach applies to other controls such as `Toggle`:

```
Menu {
    Button(action: openInPreview) {
        Text("Open in Preview")
        Text("View the document in Preview")
    }
    Button(action: saveAsPDF) {
        Text("Save as PDF")
        Text("Export the document as a PDF file")
    }
} label: {
    Label("PDF", systemImage: "doc.fill")
}
```

Note

This behavior does not apply to buttons outside of a menu's content.

Primary action

Menus can be created with a custom primary action. The primary action will be performed when the user taps or clicks on the body of the control, and the menu presentation will happen on a secondary gesture, such as on long press or on click of the menu indicator. The following example creates a menu that adds bookmarks, with advanced options that are presented in a menu.

```
Menu {
    Button(action: addCurrentTabToReadingList) {
            Label("Add to Reading List", systemImage: "eyeglasses")
        }
        Button(action: bookmarkAll) {
            Label("Add Bookmarks for All Tabs", systemImage: "book")
        }
        Button(action: show) {
            Label("Show All Bookmarks", systemImage: "books.vertical")
        }
} label: {
    Label("Add Bookmark", systemImage: "book")
} primaryAction: {
    addBookmark()
}
```

Styling menus

Use the menuStyle(_:) modifier to change the style of all menus in a view. The following example shows how to apply a custom style:

```
Menu("Editing") {
    Button("Set In Point", action: setInPoint)
    Button("Set Out Point", action: setOutPoint)
}
.menuStyle(EditingControlsMenuStyle())
```

Topics

Creating a menu from content

`init(_:content:)`

Creates a menu that generates its label from a localized string key.

`init(content: () -> Content, label: () -> Label)`

Creates a menu with a custom label.

`init(_:image:content:)`

Creates a menu that generates its label from a localized string key and image resource.

`init(_:systemImage:content:)`

Creates a menu that generates its label from a localized string key and system image.

Creating a menu with a primary action

`init(_:content:primaryAction:)`

Creates a menu with a custom primary action that generates its label from a localized string key.

`init(content: () -> Content, label: () -> Label, primaryAction: () -> Void)`

Creates a menu with a custom primary action and custom label.

`init(_:image:content:primaryAction:)`

Creates a menu with a custom primary action that generates its label from a localized string key.

`init(_:systemImage:content:primaryAction:)`

Creates a menu with a custom primary action that generates its label from a localized string key and system image.

Creating a menu from a configuration

`init(MenuStyleConfiguration)`

Creates a menu based on a style configuration.

Relationships

Conforms To

`View`

See Also

Creating a menu

Populating SwiftUI menus with adaptive controls

Improve your app by populating menus with controls and organizing your content intuitively.

```
func menuStyle<S>(S) -> some View
```

Sets the style for menus within this view.