SwiftUI / Scenes

API Collection

Scenes

Declare the user interface groupings that make up the parts of your app.

Overview

A scene represents a part of your app's user interface that has a life cycle that the system manages. An `App` instance presents the scenes it contains, while each `Scene` acts as the root element of a `View` hierarchy.

The system presents scenes in different ways depending on the type of scene, the platform, and the context. A scene might fill the entire display, part of the display, a window, a tab in a window, or something else. In some cases, your app might also be able to display more than one instance of the scene at a time, like when a user simultaneously opens multiple windows based on a single `WindowGroup` declaration in your app. For more information about the primary built-in scene types, see Windows and Documents.

You configure scenes using modifiers, similar to how you configure views. For example, you can adjust the appearance of the window that contains a scene — if the scene happens to appear in a window — using the `windowStyle(_:)` modifier. Similarly, you can add menu commands that become available when the scene is in the foreground on certain platforms using the `commands(content:)` modifier.

Topics

Creating scenes

`protocol Scene`

A part of an app's user interface with a life cycle managed by the system.

`struct SceneBuilder`

A result builder for composing a collection of scenes into a single composite scene.

Monitoring scene life cycle

`var scenePhase: ScenePhase`

The current phase of the scene.

`enum ScenePhase`

An indication of a scene's operational state.

Managing a settings window

`struct Settings`

A scene that presents an interface for viewing and modifying an app's settings.

`struct SettingsLink`

A view that opens the Settings scene defined by an app.

`struct OpenSettingsAction`

An action that presents the settings scene for an app.

`var openSettings: OpenSettingsAction`

A Settings presentation action stored in a view's environment.

Building a menu bar

Building and customizing the menu bar with SwiftUI

Provide a seamless, cross-platform user experience by building a native menu bar for iPadOS and macOS.

Creating a menu bar extra

`struct MenuBarExtra`

A scene that renders itself as a persistent control in the system menu bar.

`func menuBarExtraStyle<S>(S) -> some Scene`

Sets the style for menu bar extra created by this scene.

`protocol MenuBarExtraStyle`

A specification for the appearance and behavior of a menu bar extra scene.

Creating watch notifications

`struct WKNotificationScene`

A scene which appears in response to receiving the specified category of remote or local notifications.

See Also

App structure

- App organization

  Define the entry point and top-level structure of your app.

- Windows

  Display user interface content in a window or a collection of windows.

- Immersive spaces

  Display unbounded content in a person’s surroundings.

- Documents

  Enable people to open and manage documents.

- Navigation

  Enable people to move between different parts of your app’s view hierarchy within a scene.

- Modal presentations

  Present content in a separate view that offers focused interaction.

- Toolbars

  Provide immediate access to frequently used commands and controls.

- Search

  Enable people to search for text or other content within your app.

- App extensions

  Extend your app’s basic functionality to other parts of the system, like by adding a Widget.