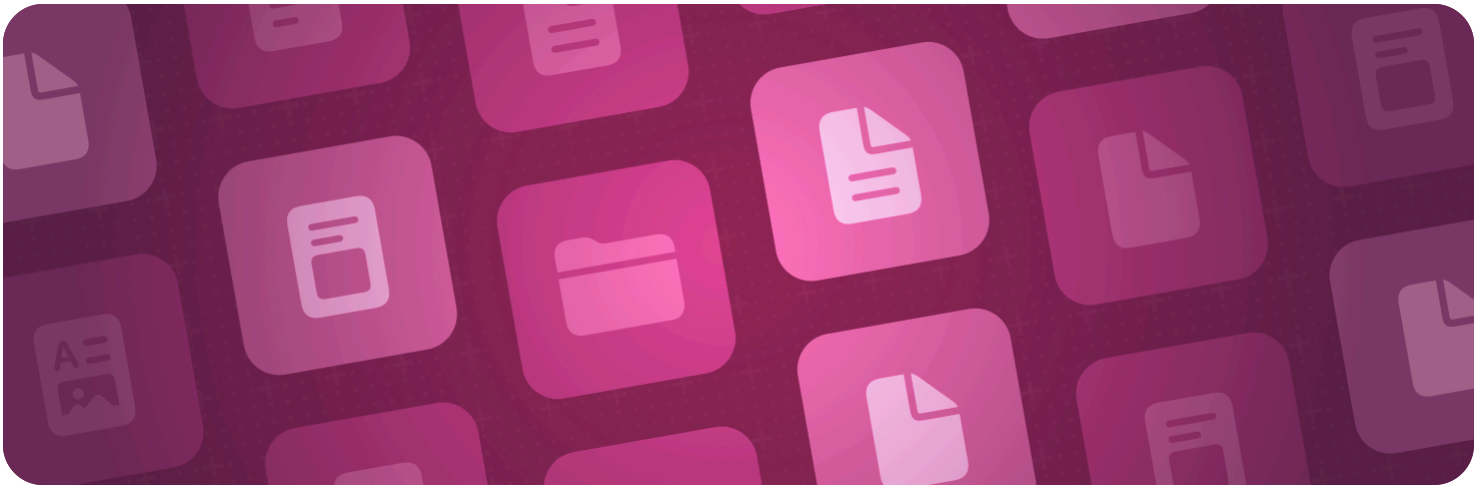
API Collection

Documents

Enable people to open and manage documents.

Overview

Create a user interface for opening and editing documents using the `DocumentGroup` scene type.



You initialize the scene with a model that describes the organization of the document's data, and a view hierarchy that SwiftUI uses to display the document's contents to the user. You can use either a value type model, which you typically store as a structure, that conforms to the `FileDocument` protocol, or a reference type model you store in a class instance that conforms to the `ReferenceFileDocument` protocol. You can also use SwiftData-backed documents using an initializer like `init(editing:contentType:editor:prepareDocument:)`.

SwiftUI supports standard behaviors that users expect from a document-based app, appropriate for each platform, like multiwindow support, open and save panels, drag and drop, and so on. For related design guidance, see Patterns in the Human Interface Guidelines.

Topics

Creating a document

{} Building a document-based app with SwiftUI

Create, save, and open documents in a multiplatform app.

{} Building a document-based app using SwiftData

Code along with the WWDC presenter to transform an app with SwiftData.

`struct DocumentGroup`

A scene that enables support for opening, creating, and saving documents.

Storing document data in a structure instance

`protocol FileDocument`

A type that you use to serialize documents to and from file.

`struct FileDocumentConfiguration`

The properties of an open file document.

Storing document data in a class instance

`protocol ReferenceFileDocument`

A type that you use to serialize reference type documents to and from file.

`struct ReferenceFileDocumentConfiguration`

The properties of an open reference file document.

`var undoManager: UndoManager?`

The undo manager used to register a view's undo operations.

Accessing document configuration

`var documentConfiguration: DocumentConfiguration?`

The configuration of a document in a `DocumentGroup`.

`struct DocumentConfiguration`

Reading and writing documents

`struct` `FileDocumentReadConfiguration`

The configuration for reading file contents.

`struct` `FileDocumentWriteConfiguration`

The configuration for serializing file contents.

Opening a document programmatically

`var` `newDocument: NewDocumentAction`

An action in the environment that presents a new document.

`struct` `NewDocumentAction`

An action that presents a new document.

`var` `openDocument: OpenDocumentAction`

An action in the environment that presents an existing document.

`struct` `OpenDocumentAction`

An action that presents an existing document.

Configuring the document launch experience

`struct` `DocumentGroupLaunchScene`

A launch scene for document-based applications.

`struct` `DocumentLaunchView`

A view to present when launching document-related user experience.

`struct` `DocumentLaunchGeometryProxy`

A proxy for access to the frame of the scene and its title view.

`struct` `DefaultDocumentGroupLaunchActions`

The default actions for the document group launch scene and the document launch view.

`struct` `NewDocumentButton`

A button that creates and opens new documents.

`protocol` `DocumentBaseBox`

A Box that allows setting its Document base not requiring the caller to know the exact types of the box and its base.

Renaming a document

`struct RenameButton`

A button that triggers a standard rename action.

`func renameAction(_:)`

Sets a closure to run for the rename action.

`var rename: RenameAction?`

An action that activates the standard rename interaction.

`struct RenameAction`

An action that activates a standard rename interaction.

See Also

App structure

App organization

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

Scenes

Declare the user interface groupings that make up the parts 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.

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.