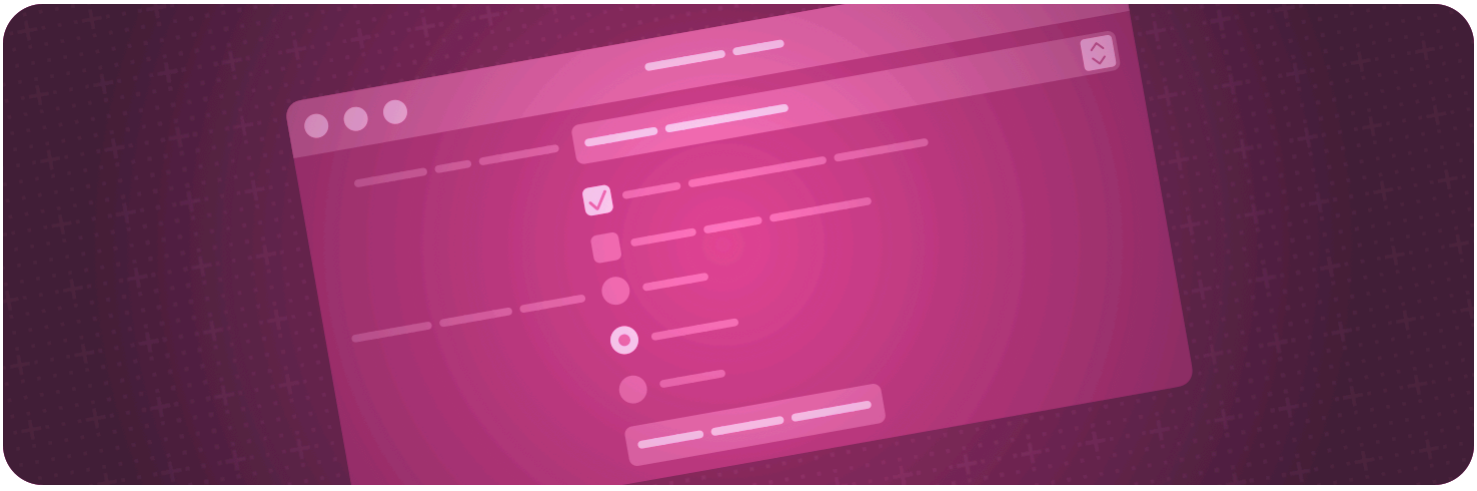
API Collection

View groupings

Present views in different kinds of purpose-driven containers, like forms or control groups.

Overview

You can create groups of views that serve different purposes.



For example, a `Group` construct treats the specified views as a unit without imposing any additional layout or appearance characteristics. A `Form` presents a group of elements with a platform-specific appearance that's suitable for gathering input from people.

For design guidance, see Layout in the Human Interface Guidelines.

Topics

Grouping views into a container

{} Creating custom container views

Access individual subviews to compose flexible container views.

`struct` `Group`

A type that collects multiple instances of a content type — like views, scenes, or commands — into a single unit.

`struct` `GroupElementsOfContent`

Transforms the subviews of a given view into a resulting content view.

`struct` `GroupSectionsOfContent`

Transforms the sections of a given view into a resulting content view.

Organizing views into sections

`struct` `Section`

A container view that you can use to add hierarchy within certain views.

`struct` `SectionCollection`

An opaque collection representing the sections of view.

`struct` `SectionConfiguration`

Specifies the contents of a section.

Iterating over dynamic data

`struct` `ForEach`

A structure that computes views on demand from an underlying collection of identified data.

`struct` `ForEachSectionCollection`

A collection which allows a view to be treated as a collection of its sections in a for each loop.

`struct` `ForEachSubviewCollection`

A collection which allows a view to be treated as a collection of its subviews in a for each loop.

`protocol` `DynamicViewContent`

A type of view that generates views from an underlying collection of data.

Accessing a container's subviews

`struct Subview`

An opaque value representing a subview of another view.

`struct SubviewsCollection`

An opaque collection representing the subviews of view.

`struct SubviewsCollectionSlice`

A slice of a SubviewsCollection.

`func containerValue<V>(WritableKeyPath<ContainerValues, V>, V) -> some View`

Sets a particular container value of a view.

`struct ContainerValues`

A collection of container values associated with a given view.

`protocol ContainerValueKey`

A key for accessing container values.

Grouping views into a box

`struct GroupBox`

A stylized view, with an optional label, that visually collects a logical grouping of content.

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

Sets the style for group boxes within this view.

Grouping inputs

`struct Form`

A container for grouping controls used for data entry, such as in settings or inspectors.

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

Sets the style for forms in a view hierarchy.

`struct LabeledContent`

A container for attaching a label to a value-bearing view.

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

Sets a style for labeled content.

Presenting a group of controls

`struct ControlGroup`

A container view that displays semantically-related controls in a visually-appropriate manner for the context

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

Sets the style for control groups within this view.

See Also

View layout

- Layout fundamentals

  Arrange views inside built-in layout containers like stacks and grids.

- Layout adjustments

  Make fine adjustments to alignment, spacing, padding, and other layout parameters.

- Custom layout

  Place views in custom arrangements and create animated transitions between layout types.

- Lists

  Display a structured, scrollable column of information.

- Tables

  Display selectable, sortable data arranged in rows and columns.

- Scroll views

  Enable people to scroll to content that doesn’t fit in the current display.