Structure

Map

A view that displays an embedded map interface.

MapKit | SwiftUI | iOS 14.0+ | iPadOS 14.0+ | Mac Catalyst | macOS 11.0+ | tvOS 14.0+ | visionOS | watchOS 7.0+

```
@MainActor @preconcurrency
struct Map<Content> where Content : View
```

Overview

Use this SwiftUI view to display a Map with markers, annotations, and custom content you provide. You can configure the Map to optionally display the user's location, track a location, and display various controls to allow them to interact with and control the map's display. The following example displays a map of downtown San Francisco that shows different markers, and an annotation with custom view content at specific locations:

```
    struct ContentView: View {
        var body: some View {
            Map {
                Marker("San Francisco City Hall", coordinate: cityHallLocation)
                    .tint(.orange)
                Marker("San Francisco Public Library", coordinate: publicLibraryLoca
                    .tint(.blue)
                Annotation("Diller Civic Center Playground", coordinate: playgroundL
                    ZStack {
                        RoundedRectangle(cornerRadius: 5)
                            .fill(Color.yellow)
                        Text("🛝")
```

```
                            .padding(5)
                    }
                }
            }
            .mapControlVisibility(.hidden)
        }
    }
```

You create markers, annotations, and overlays using `MapContentBuilder` with any of several `MapContent` types including:

- `Annotation`
- `UserAnnotation`
- `Marker`
- `MapCircle`
- `MapPolygon`
- `MapPolyline`

You can also add a variety of controls to allow a person to interact with the map to change the map's scale, display or hide the device's current location, and so on:

- `MapCompass`
- `MapPitchButton`
- `MapPitchSlider`
- `MapScaleView`
- `MapUserLocationButton`
- `MapZoomStepper`

Topics

Creating a map

`init(bounds: MapCameraBounds?, interactionModes: MapInteractionModes, scope: Namespace.ID?)`

Creates a new, empty map with the bounds, interaction modes, and scope you provide.

`init<C>(bounds: MapCameraBounds?, interactionModes: MapInteractionModes, scope: Namespace.ID?, content: () -> C)`

Creates a new map with the bounds, interaction modes, scope, and content you provide.

`init(bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<MapFeature?>, scope: Namespace.ID?)`

Creates a new, empty map with the bounds, interaction modes, a binding to a map feature, and scope you provide.

`init<SelectedValue>(bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<SelectedValue?>, scope: Namespace.ID?)`

Creates a new, empty map with the bounds, interaction modes, the selected map feature, and scope you provide.

`init<C>(bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<MapFeature?>, scope: Namespace.ID?, content: () -> C)`

Creates a new map with the bounds, interaction modes, selected map feature, scope, and map content you provide.

`init<SelectedValue, C>(bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<SelectedValue?>, scope: Namespace.ID?, content: () -> C)`

Creates a new map with the bounds, interaction modes, selected value, scope, and map content you provide.

`init(initialPosition: MapCameraPosition, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, scope: Namespace.ID?)`

Creates a new, empty map with the initial camera position, bounds, interaction modes, and scope you provide.

`init<C>(initialPosition: MapCameraPosition, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, scope: Namespace.ID?, content: () -> C)`

Creates a new map with the initial camera position, bounds, interaction modes, scope, and map content you provide.

`init(initialPosition: MapCameraPosition, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<MapFeature?>, scope: Namespace.ID?)`

Creates a new, empty map with the initial camera position, bounds, interaction modes, selected map feature, and scope you provide.

```
init<C>(initialPosition: MapCameraPosition, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<MapFeature?>, scope: Namespace.ID?, content: () -> C)
```

Creates a new map with the initial camera position, bounds, interaction modes, selected map feature, scope, and content you provide.

```
init<SelectedValue, C>(initialPosition: MapCameraPosition, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<SelectedValue?>, scope: Namespace.ID?, content: () -> C)
```

Creates a new map with the initial camera position, bounds, interaction modes, selected map feature, scope, and content you provide.

```
init(position: Binding<MapCameraPosition>, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, scope: Namespace.ID?)
```

Creates a new, empty map with the initial camera position, bounds, interaction modes, and scope you provide.

```
init<C>(position: Binding<MapCameraPosition>, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, scope: Namespace.ID?, content: () -> C)
```

Creates a new map with the initial camera position, bounds, interaction modes, scope, and content you provide.

```
init(position: Binding<MapCameraPosition>, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<MapFeature?>, scope: Namespace.ID?)
```

Creates a new map with the initial camera position, bounds, interaction modes, scope, and content you provide.

```
init<C>(position: Binding<MapCameraPosition>, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<MapFeature?>, scope: Namespace.ID?, content: () -> C)
```

Creates a new map with the initial camera position, bounds, interaction modes, selected feature, scope, and content you provide.

```
init<SelectedValue, C>(position: Binding<MapCameraPosition>, bounds: MapCameraBounds?, interactionModes: MapInteractionModes, selection: Binding<SelectedValue?>, scope: Namespace.ID?, content: () -> C)
```

Creates a new map with the initial camera position, bounds, interaction modes, selected feature, scope, and content you provide.

`struct` `MapInteractionModes`

Options that indicate the user interactions that the map responds to.

Deprecated

Deprecated Symbols

Map protocols and view modifiers that are no longer supported.

Displaying place information

```
func mapItemDetailSelectionAccessory(MapItemDetailSelectionAccessory
Style?) -> some MapContent
```

Specifies the selection accessory to display for the selected map item content.

Initializers

```
init<SelectedValue, C>(bounds: MapCameraBounds?, interactionModes: Map
InteractionModes, selection: Binding<SelectedValue?>, scope: Namespace.
ID?, content: () -> C)
```

```
init<SelectedValue, C>(initialPosition: MapCameraPosition, bounds: Map
CameraBounds?, interactionModes: MapInteractionModes, selection:
Binding<SelectedValue?>, scope: Namespace.ID?, content: () -> C)
```

```
init<SelectedValue, C>(position: Binding<MapCameraPosition>, bounds:
MapCameraBounds?, interactionModes: MapInteractionModes, selection:
Binding<SelectedValue?>, scope: Namespace.ID?, content: () -> C)
```

Relationships

Conforms To

`Sendable`, `SendableMetatype`, `View`

See Also

Essentials

`struct` **MapStyle**

A style that you can apply to a map.