Protocol

ViewModifier

A modifier that you apply to a view or another view modifier, producing a different version of the original value.

iOS 13.0+ | iPadOS 13.0+ | Mac Catalyst 13.0+ | macOS 10.15+ | tvOS 13.0+ | visionOS 1.0+ | watchOS 6.0+

```
@MainActor @preconcurrency
protocol ViewModifier
```

Mentioned in

Reducing view modifier maintenance

Overview

Adopt the `ViewModifier` protocol when you want to create a reusable modifier that you can apply to any view. The example below combines several modifiers to create a new modifier that you can use to create blue caption text surrounded by a rounded rectangle:

```
struct BorderedCaption: ViewModifier {
    func body(content: Content) -> some View {
        content
            .font(.caption2)
            .padding(10)
            .overlay(
                RoundedRectangle(cornerRadius: 15)
                    .stroke(lineWidth: 1)
            )
            .foregroundColor(Color.blue)
```

```
    }
}
```

You can apply `modifier(_:)` directly to a view, but a more common and idiomatic approach uses `modifier(_:)` to define an extension to `View` itself that incorporates the view modifier:

```
extension View {
    func borderedCaption() -> some View {
        modifier(BorderedCaption())
    }
}
```

You can then apply the bordered caption to any view, similar to this:

```
Image(systemName: "bus")
    .resizable()
    .frame(width:50, height:50)
Text("Downtown Bus")
    .borderedCaption()
```

A type conforming to this protocol inherits `@preconcurrency @MainActor` isolation from the protocol if the conformance is included in the type's base declaration:

```
struct MyCustomType: Transition {
    // `@preconcurrency @MainActor` isolation by default
}
```

Isolation to the main actor is the default, but it's not required. Declare the conformance in an extension to opt out of main actor isolation:

```
extension MyCustomType: Transition {
    // `nonisolated` by default
}
```

Topics

Creating a view modifier

`func body(content: Self.Content) -> Self.Body`

Gets the current body of the caller.

Required Default implementation provided.

`associatedtype Body : View`

The type of view representing the body.

Required

`typealias Content`

The content view type passed to `body()`.

Adding animations to a view

`func animation(Animation?) -> some ViewModifier`

Returns a new version of the modifier that will apply `animation` to all animatable values within the modifier.

`func concat<T>(T) -> ModifiedContent<Self, T>`

Returns a new modifier that is the result of concatenating `self` with `modifier`.

Handling view taps and gestures

`func transaction((inout Transaction) -> Void) -> some ViewModifier`

Returns a new version of the modifier that will apply the transaction mutation function `transform` to all transactions within the modifier.

Relationships

Inherited By

`AnimatableModifier`, `EnvironmentalModifier`, `GeometryEffect`

Conforming Types

`AccessibilityAttachmentModifier`
`EmptyModifier`
`LayoutRotationUnaryLayout`
`ManipulableModifier`
`ManipulableResponderModifier`
`ManipulableTransformBindingModifier`
`ManipulationGeometryModifier`
`ManipulationGestureModifier`
`ManipulationUsingGestureStateModifier`
`ModifiedContent`

Conforms when `Content` conforms to `ViewModifier` and `Modifier` conforms to `ViewModifier`.

See Also

Modifying a view

Configuring views

Adjust the characteristics of a view by applying view modifiers.

Reducing view modifier maintenance

Bundle view modifiers that you regularly reuse into a custom view modifier.

`func modifier<T>(T) -> ModifiedContent<Self, T>`

Applies a modifier to a view and returns a new view.

`struct` EmptyModifier

An empty, or identity, modifier, used during development to switch modifiers at compile time.

`struct` ModifiedContent

A value with a modifier applied to it.

`protocol` EnvironmentalModifier

A modifier that must resolve to a concrete modifier in an environment before use.

`struct` ManipulableModifier

`struct` ManipulableResponderModifier

`struct` ManipulableTransformBindingModifier

`struct` ManipulationGeometryModifier

`struct` ManipulationGestureModifier

`struct` ManipulationUsingGestureStateModifier

`enum` Manipulable

A namespace for various manipulable related types.