A closure-based view in Swift is a view that is created using a closure. Closures are self-contained blocks of code that can be passed around and used in your code.

To create a closure-based view, you simply pass a closure to the `View` protocol. The closure will be executed when the view is rendered.

For example, the following code creates a closure-based view that displays a text view:

```Swift
let textView: some View = {
  TextView(text: "Hello, world!")
}
```

This closure-based view will display a text view with the text "Hello, world!".

Closure-based views can also be used to create more complex views, such as buttons, images, and navigation bars. For example, the following code creates a closure-based view that displays a button:

```Swift
let button: some View = {
  Button(action: {
    // Code to execute when the button is tapped
  }, label: {
    Text("Tap Me")
  })
}
```

This closure-based view will display a button with the text "Tap Me". When the button is tapped, the code inside the `action` closure will be executed.

Closure-based views are a powerful way to create custom views in SwiftUI. They allow you to create views that are dynamic and interactive.

Here are some additional tips for using closure-based views:

- You can use closure-based views to create views that are conditional, meaning that they are only displayed if certain conditions are met.
- You can use closure-based views to create views that are nested within other views.

- You can use closure-based views to create views that are passed as arguments to other functions.

I hope this helps!