1

IOS Development
Interview Questions

What is SwiftUI and how does it differ from UIKit?

SwiftUI is a **declarative** user interface (UI) framework for building apps on Apple platforms, such as iOS, macOS, and watchOS. It uses a modern approach to UI development, allowing developers to create dynamic and interactive user interfaces using simple, easy-to-understand code.

The main difference between SwiftUI and UIKit, Apple's traditional UI framework, is that SwiftUI is designed to be simpler and more intuitive to use, with less boilerplate code and a more natural approach to UI development.

IOS Development

Interview Questions

What are some benefits of using SwiftUI over traditional UIKit?

Some benefits of using SwiftUI over traditional UIKit include

- **Less code**: Declarative syntax for efficient code.
- **Live preview**: Real-time UI preview for faster development.
- **Easy customization**: UI customization is made easy with modifiers.
- **Better performance**: Optimized for efficient resource usage.
- **Cross-platform support**: Works on all Apple platforms.
- **Easier to learn**: Intuitive syntax for easy learning.

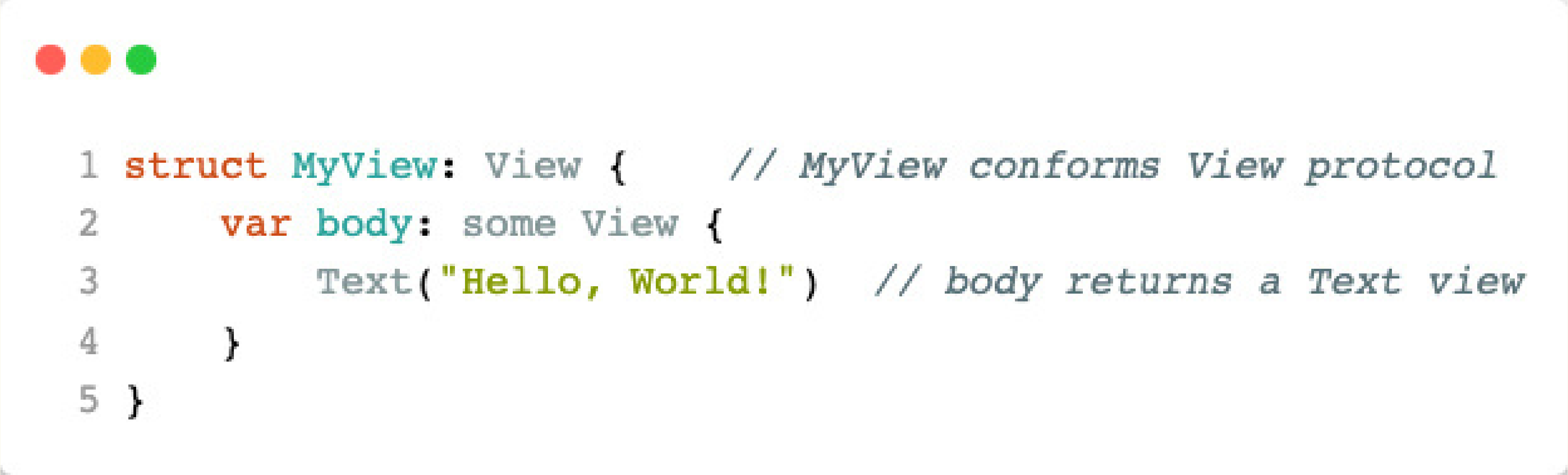
3

IOS Development
Interview Questions

How do you declare a new SwiftUI view?

Declare a new view by creating a new **struct** that conforms to the **View** protocol.

Once we defined our view, We can then use it in other views by simply instantiating it like any other Swift struct



```swift
struct MyView: View {     // MyView conforms View protocol
    var body: some View {
        Text("Hello, World!")  // body returns a Text view
    }
}
```

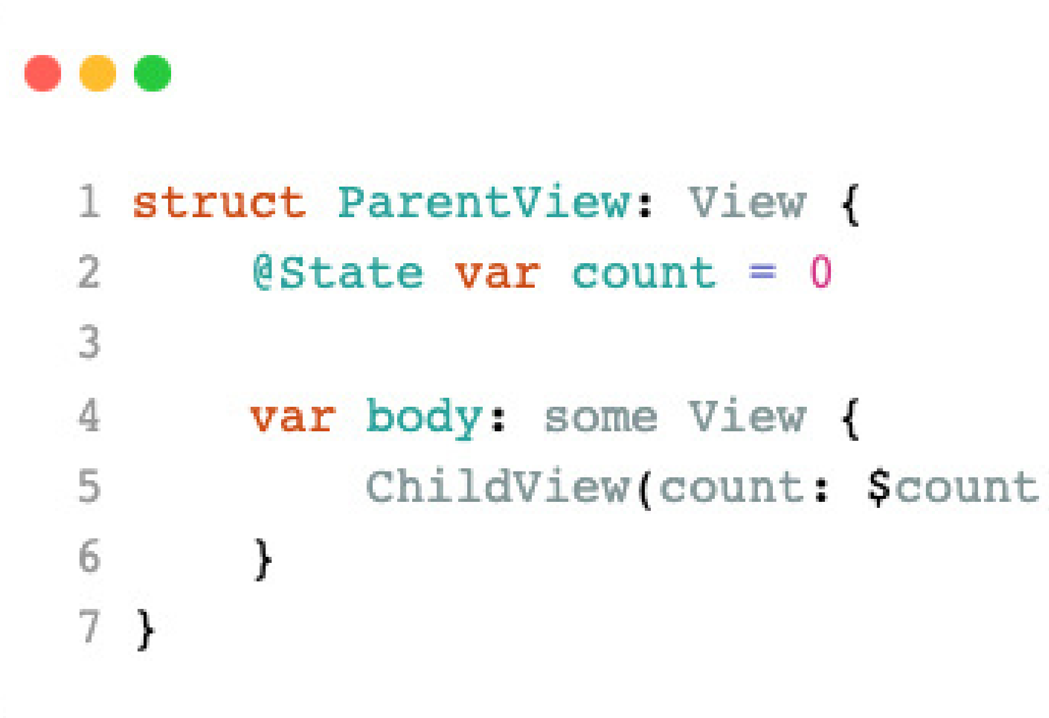
4

IOS Development

Interview Questions

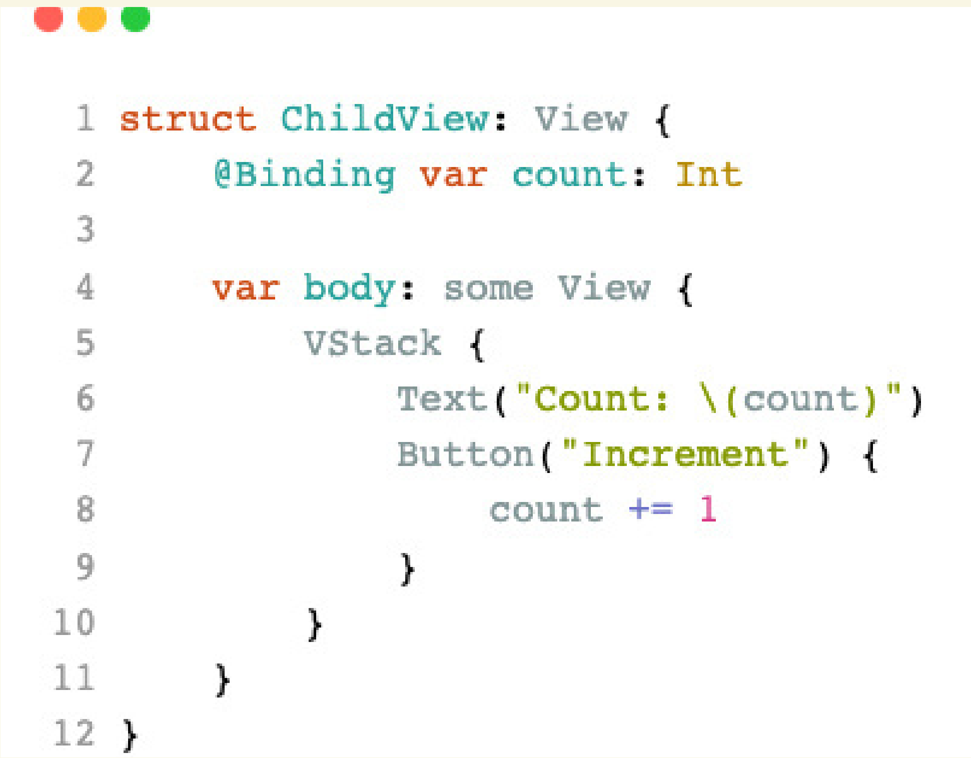
What is the difference between a **@State** and **@Binding** variable in SwiftUI?

@State variables are used to store and manage simple values within a view while **@Binding** variables are used to pass data between views and to modify values stored in a parent view.



```swift
struct ParentView: View {
    @State var count = 0

    var body: some View {
        ChildView(count: $count
    }
}
```



```swift
struct ChildView: View {
    @Binding var count: Int

    var body: some View {
        VStack {
            Text("Count: \(count)")
            Button("Increment") {
                count += 1
            }
        }
    }
}
```

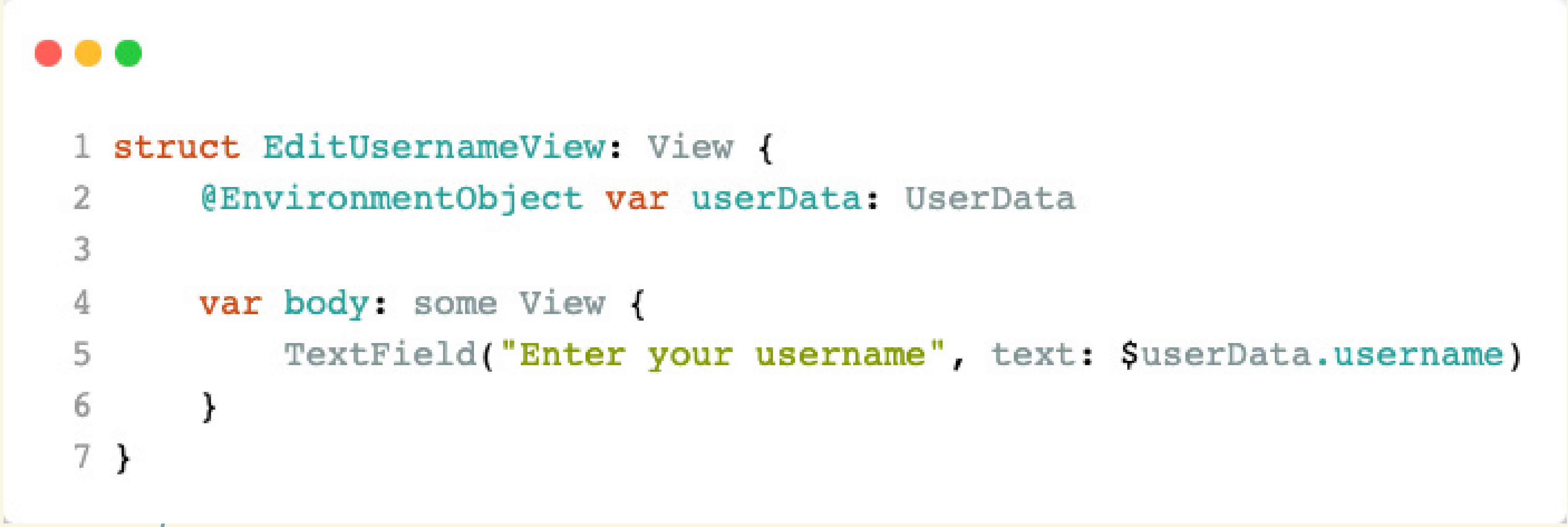
IOS Development

Interview Questions

How do you pass data between views in SwiftUI?

In SwiftUI, you can pass data between views using various techniques depending on your specific use case

1. Using **@State** and **@Binding**
2. Using **@EnvironmentObject**
3. Using **@ObservedObject**



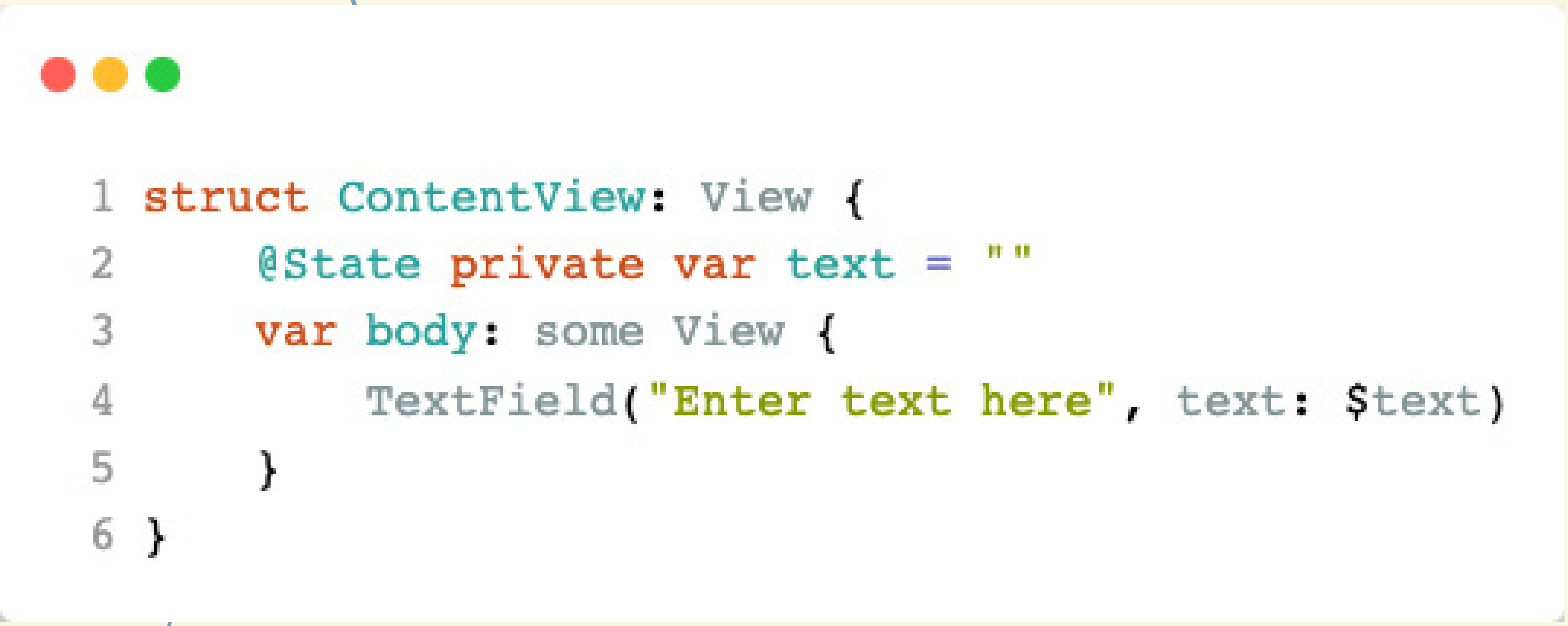
```
struct EditUsernameView: View {
    @EnvironmentObject var userData: UserData

    var body: some View {
        TextField("Enter your username", text: $userData.username)
    }
}
```

IOS Development
Interview Questions

How do you handle user input in SwiftUI?

We can handle user input using various controls such as **buttons**, **text fields**, **sliders**, and **pickers**. Each control has an action or binding property that can be used to respond to user input.



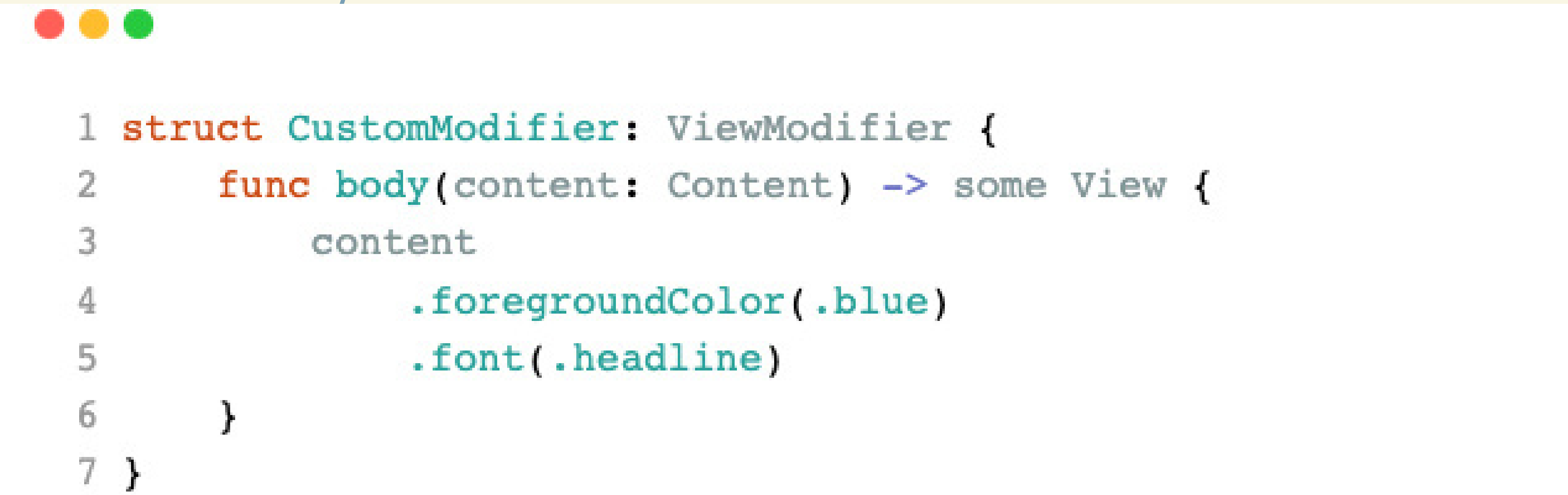
```swift
struct ContentView: View {
    @State private var text = ""
    var body: some View {
        TextField("Enter text here", text: $text)
    }
}
```

IOS Development
Interview Questions

What is the purpose of a ViewModifier in SwiftUI and how do you create one?

ViewModifier is a type that can be used to encapsulate a set of view modifications and apply them to one or more views.

The purpose of a ViewModifier is to promote code reuse and improve readability by encapsulating common view styling patterns.



```
struct CustomModifier: ViewModifier {
    func body(content: Content) -> some View {
        content
            .foregroundColor(.blue)
            .font(.headline)
    }
}
```

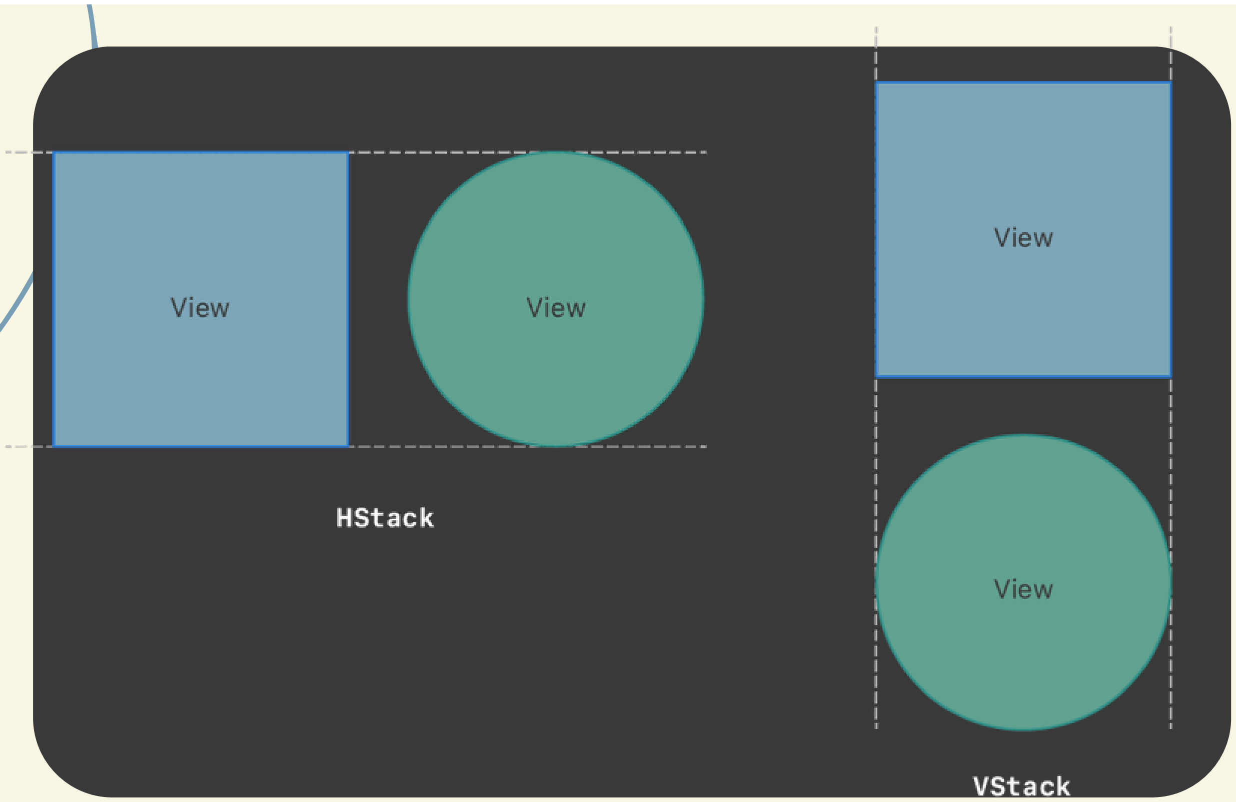
8

IOS Development

Interview Questions

What is the difference between a **VStack** and an **HStack** in SwiftUI?

The main difference between VStack and HStack is the direction in which they arrange their child views. VStack arranges its child views **vertically**, from top to bottom, while HStack arranges its child views **horizontally**, from left to right.



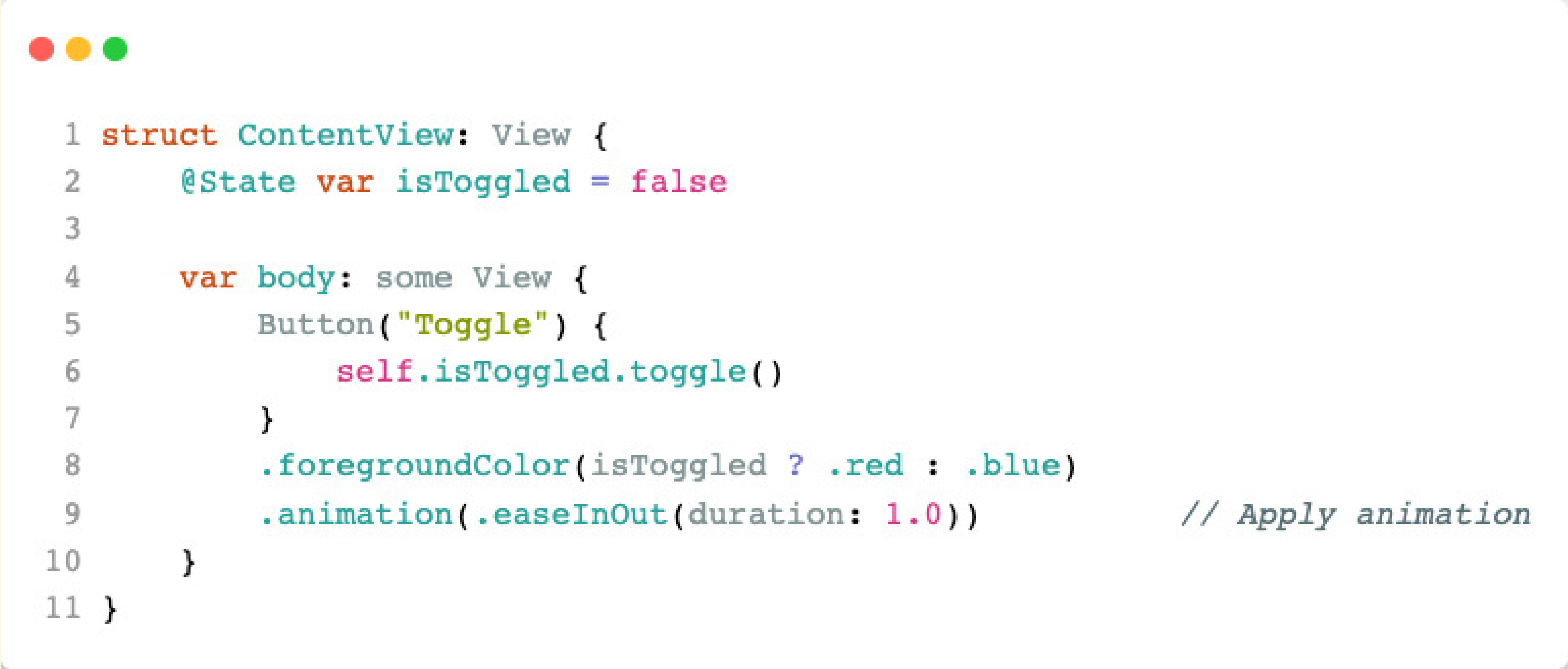
www.ferdous.tech

NEXT ➡

IOS Development
Interview Questions

How do you apply animations to views in SwiftUI?

Animations can be applied to views in SwiftUI using various techniques, including the **withAnimation** function and the **animation** modifier.

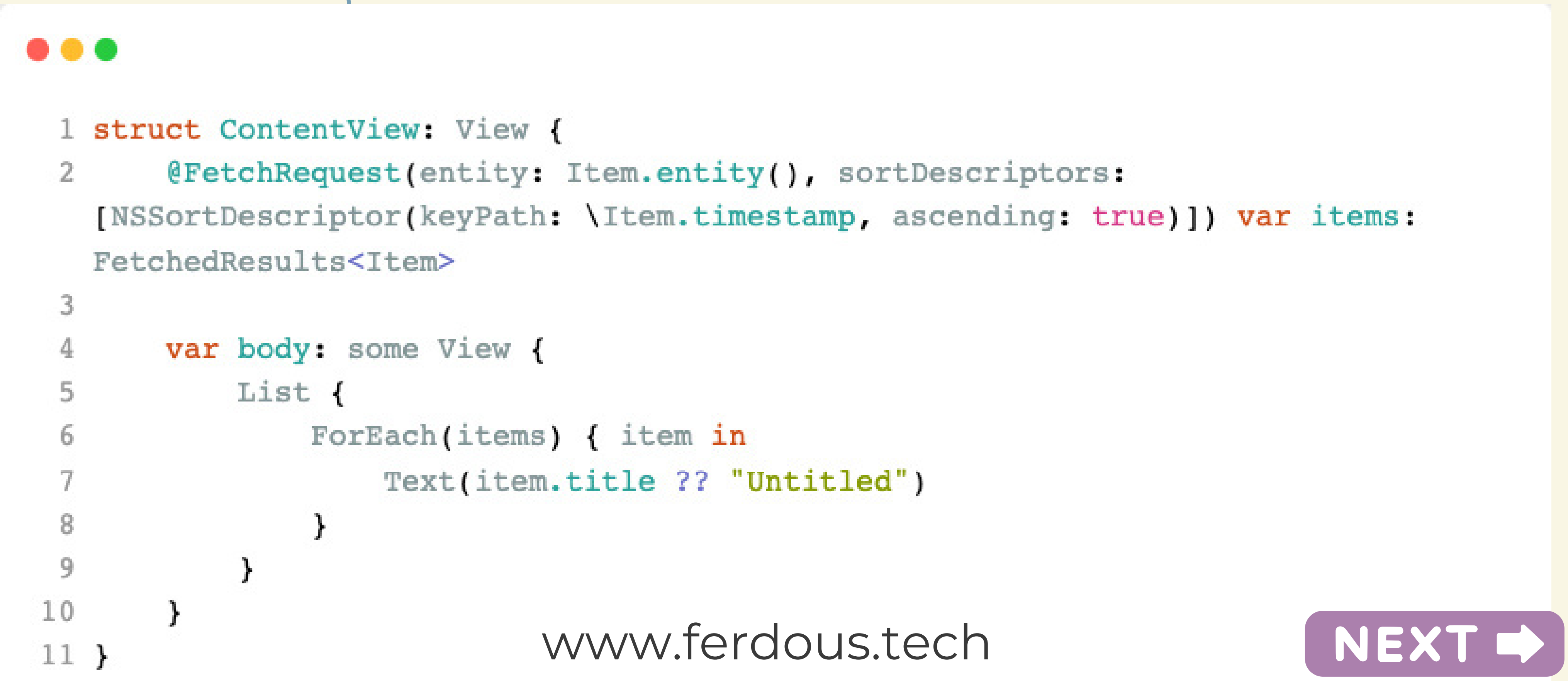


```swift
struct ContentView: View {
    @State var isToggled = false

    var body: some View {
        Button("Toggle") {
            self.isToggled.toggle()
        }
        .foregroundColor(isToggled ? .red : .blue)
        .animation(.easeInOut(duration: 1.0))          // Apply animation
    }
}
```

What is the **@FetchRequest** property wrapper in SwiftUI and how is it used?

The **@FetchRequest** property wrapper is used to fetch data from **Core Data** in SwiftUI. It is a convenient way to create a fetch request and automatically update the view when the underlying data changes.



```
struct ContentView: View {
    @FetchRequest(entity: Item.entity(), sortDescriptors: [NSSortDescriptor(keyPath: \Item.timestamp, ascending: true)]) var items: FetchedResults<Item>

    var body: some View {
        List {
            ForEach(items) { item in
                Text(item.title ?? "Untitled")
            }
        }
    }
}
```

@ferdous19

Ferdous Mahmud Akash

WAS THIS POST HELPFUL?

like comment share save

Share the information with your friends if it was useful. Every like or comment helps promote the post. **Thank you!**

www.ferdous.tech