

# **Interagir avec l'interface**

## **Bouton & Propriété d'état**

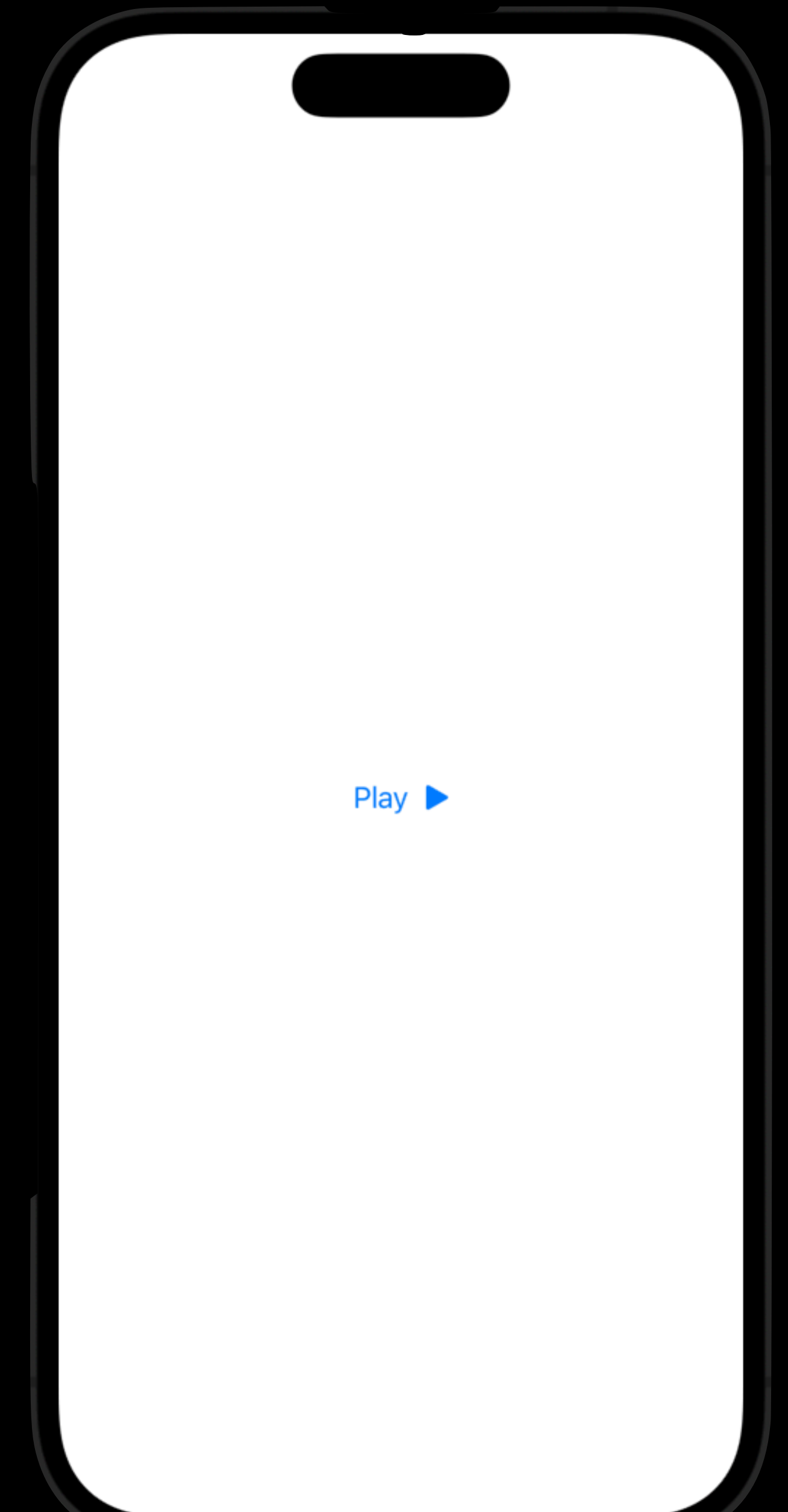
# Bouton

```
Button {  
    print("🔊")  
} label: {  
    HStack{  
        Text("Play")  
        Image(systemName: "play.fill")  
    }  
}
```

Play ▶

# Bouton

```
Button {  
    print("🔊") ← Action 🖱️  
} label: {  
    HStack{  
        Text("Play")  
        Image(systemName: "play.fill")  
    }  
}
```



# Bouton

```
Button {  
    print("🔊")  
} label: {  
    HStack{  
        Text("Play")  
        Image(systemName: "play.fill")  
    }  
}
```



Label

Play ▶

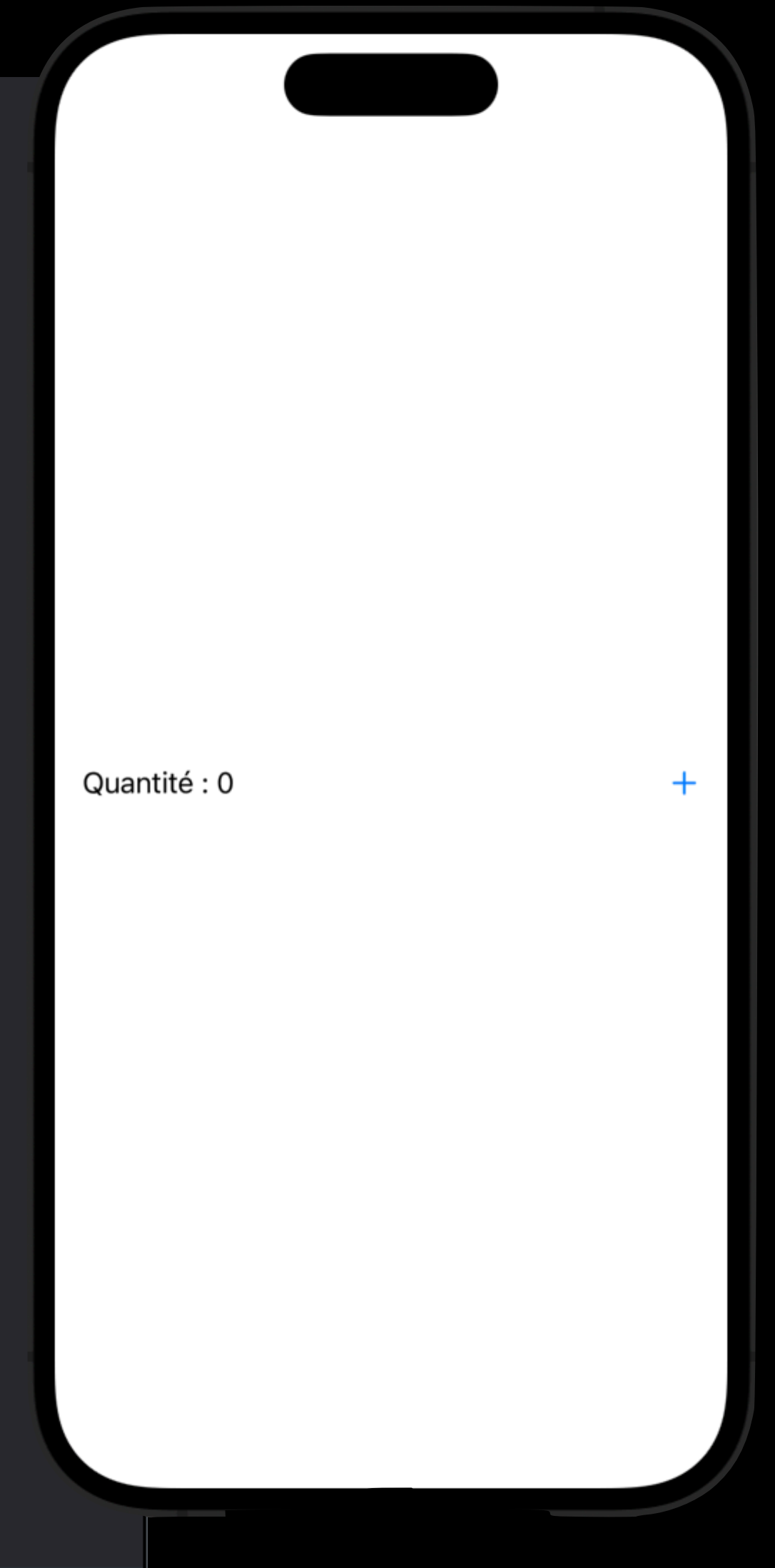
# Propriété d'état

```
struct BasketView: View {  
    var productQuantity = 0  
  
    var body: some View {  
        HStack{  
            Text("Quantité : \("\(productQuantity)")")  
        }  
        .padding(.horizontal)  
    }  
}
```



# Propriété d'état

```
struct BasketView: View {  
    var productQuantity = 0  
  
    var body: some View {  
        HStack{  
            Text("Quantité : \(productQuantity)")  
            Spacer()  
            Button {  
  
            } label: {  
                Image(systemName: "plus")  
            }  
        }.padding(.horizontal)  
    }  
}
```

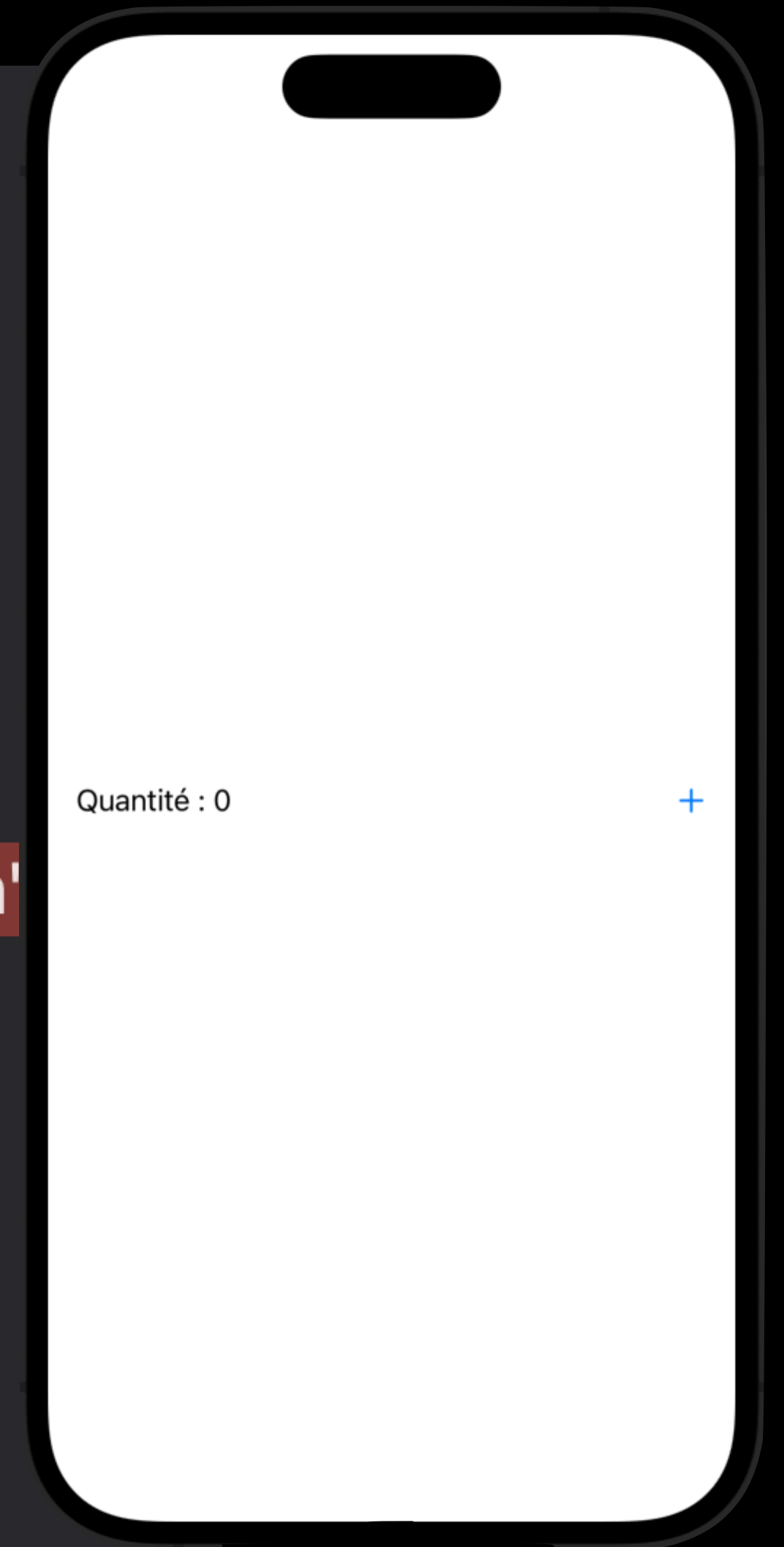


# Propriété d'état

```
struct BasketView: View {  
    var productQuantity = 0  
  
    var body: some View {  
        HStack{  
            Text("Quantité : \(productQuantity)")  
            Spacer()  
            Button {  
                productQuantity += 1  
            } label: {  
                Image(systemName: "plus")  
            }  
        }.padding(.horizontal)  
    }  
}
```

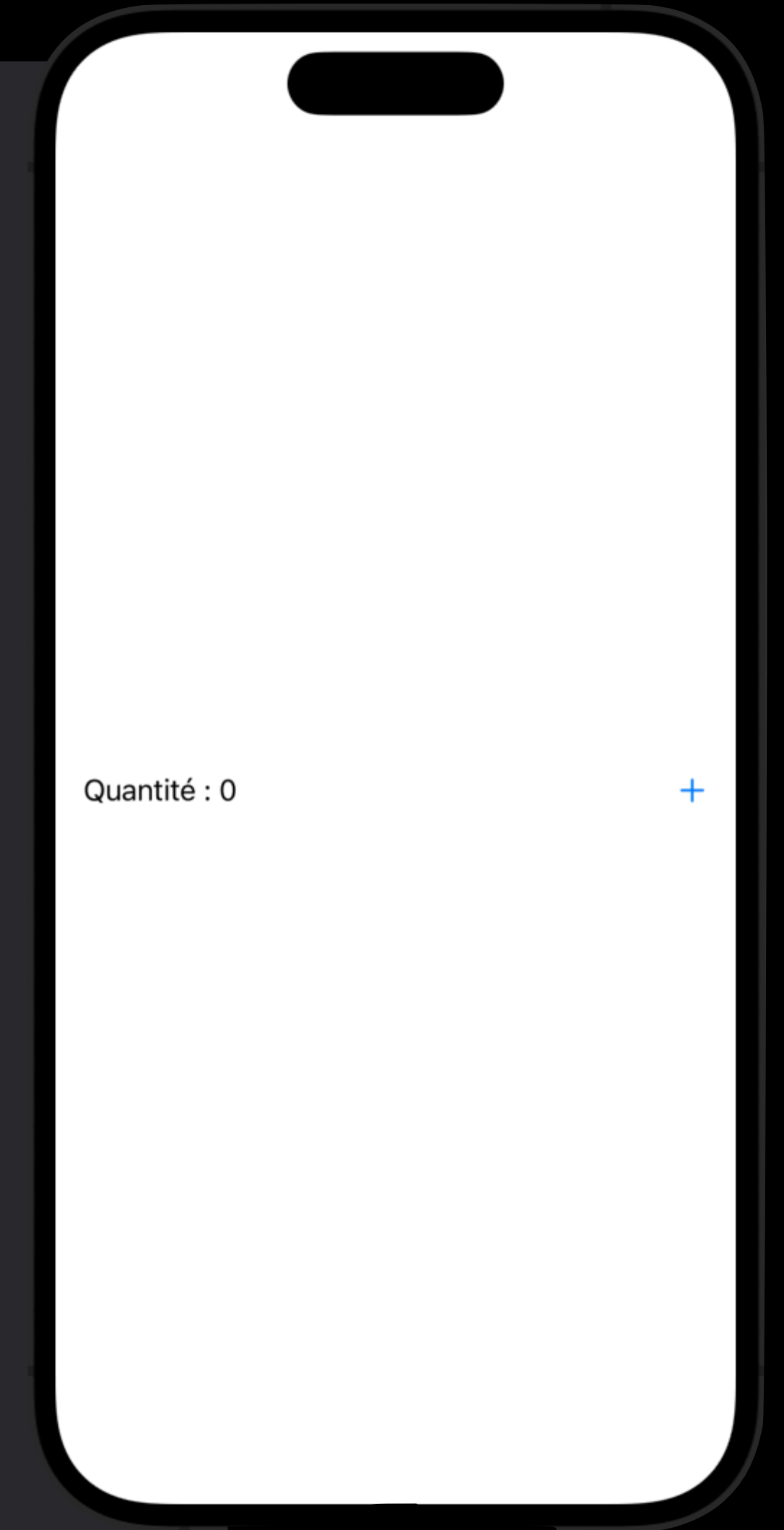


Left side of mutating operator isn't



# Propriété d'état

```
struct BasketView: View {
    @State var productQuantity = 0
    var body: some View {
        HStack{
            Text("Quantité : \(productQuantity)")
            Spacer()
            Button {
                productQuantity += 1
            } label: {
                Image(systemName: "plus")
            }
        }
        .padding(.horizontal)
    }
}
```





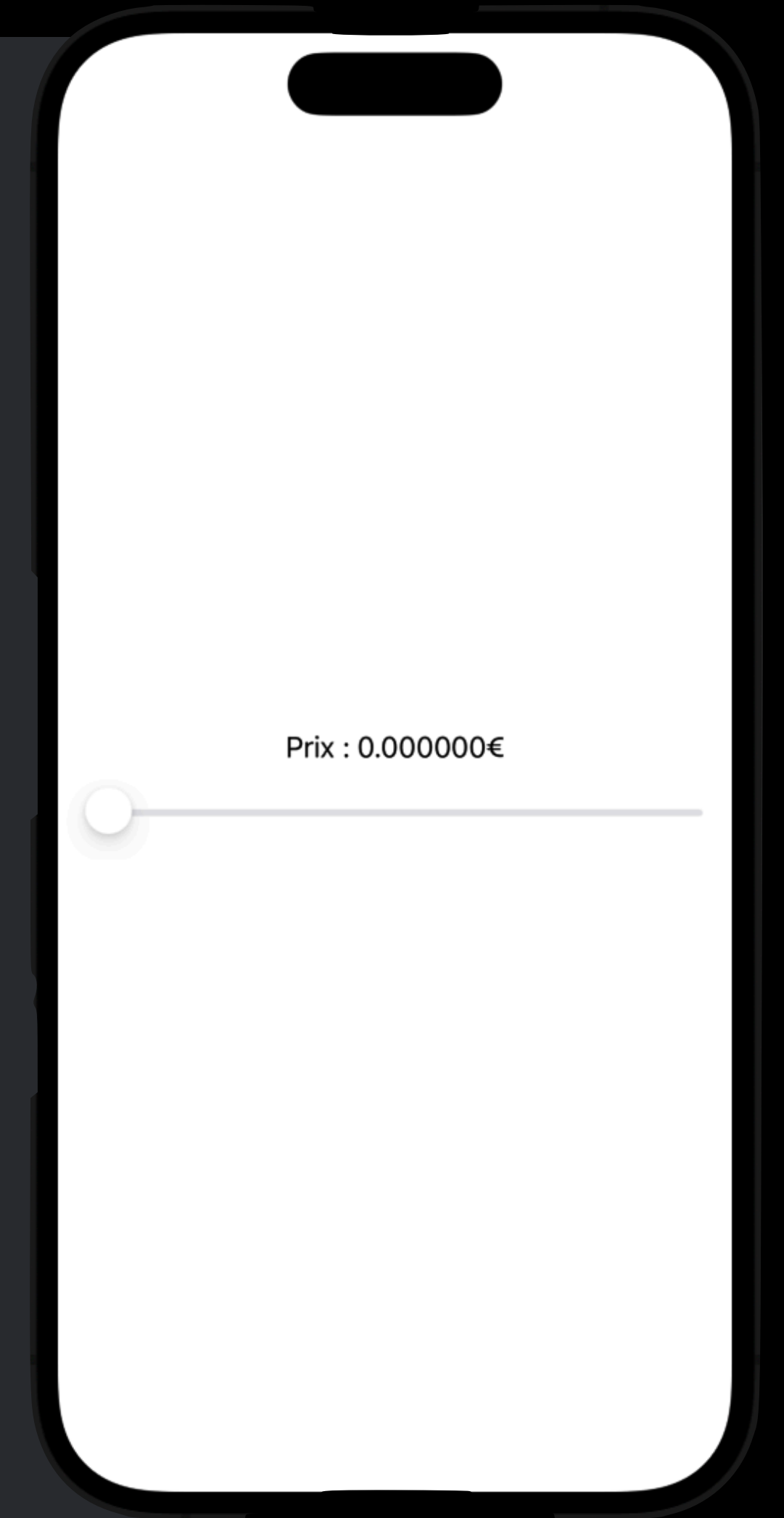
# Propriété d'état

```
struct BasketView: View {
    @State var productQuantity = 0

    var body: some View {
        HStack{
            Text("Quantité : \(productQuantity)")
            Spacer()
            Button {
                productQuantity += 1
            } label: {
                Image(systemName: "plus")
            }
        }
        .padding(.horizontal)
    }
}
```



# Propriété d'état



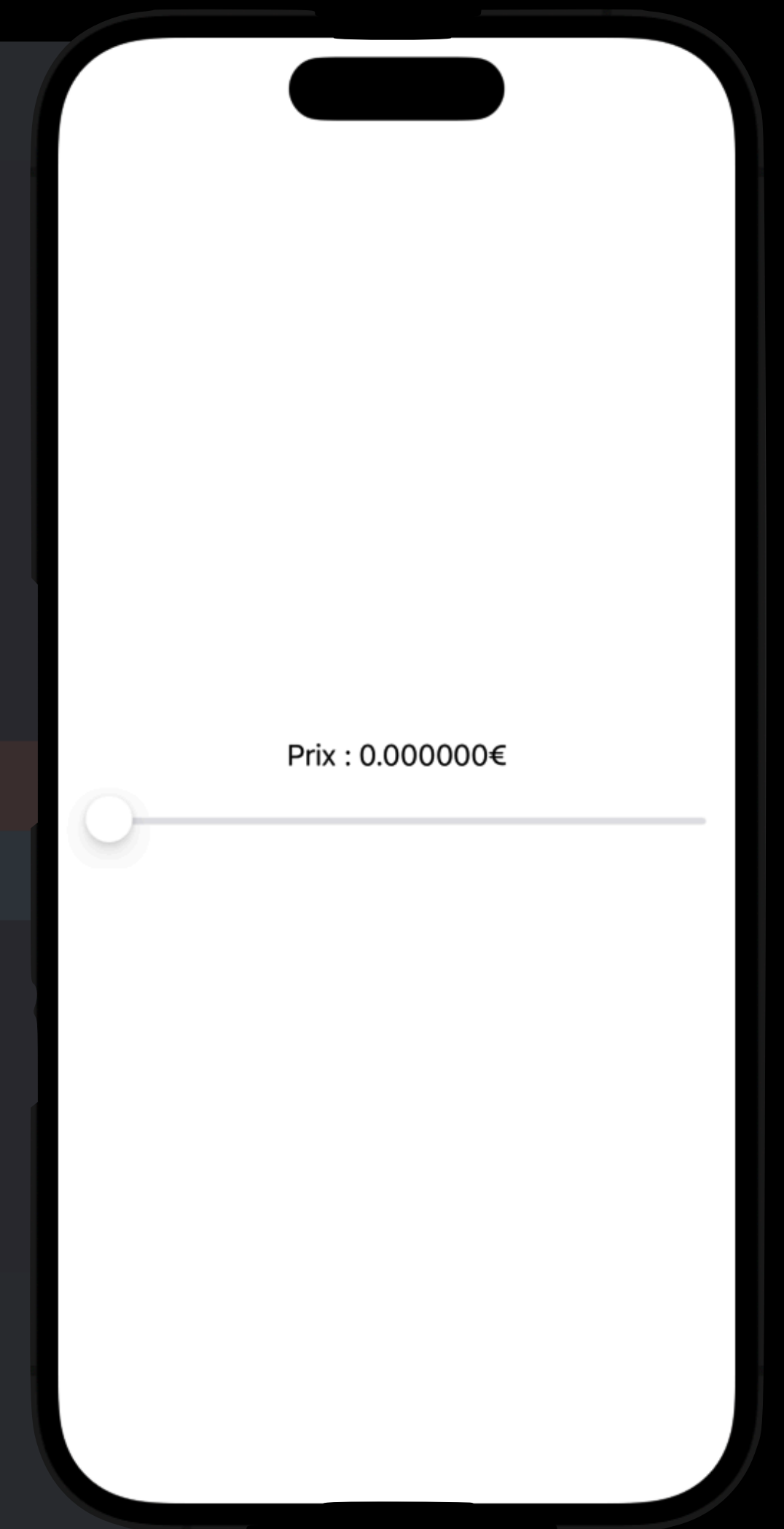
# Propriété d'état

```
struct ProductView: View {  
    @State var productPrice = 0.0  
  
    var body: some View {  
        VStack{  
            Text("Prix : \ (productPrice)€")  
            Slider(value ← productPrice, in: 0...100, step: 1)  
        }  
        .padding(.horizontal, 10)  
    }  
}
```

Cannot convert value 'productPrice' of type 'Double' to expected type 'Binding<Double>', use wrapper instead

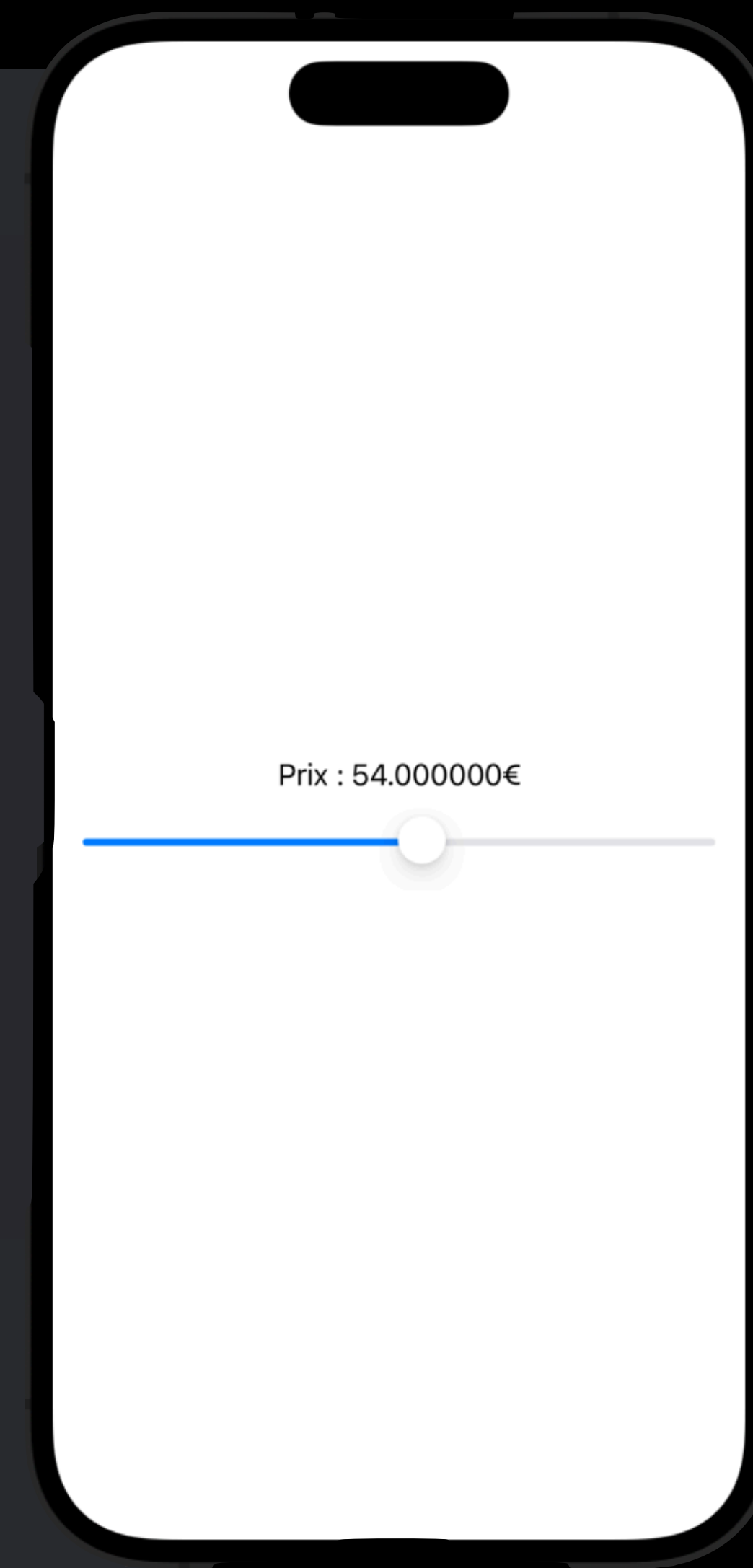
Insert '\$'

Fix



# Propriété d'état

```
struct ProductView: View {  
    @State var productPrice = 0.0  
  
    var body: some View {  
        VStack{  
            Text("Prix : \$(productPrice)€")  
            Slider(value: $productPrice, in: 0...100, step: 1)  
        }  
        .padding(.horizontal)  
    }  
}
```



**@State, @Binding ou pas**

Dans une Struct View, je défini une propriété

Dont je veux modifier la valeur dans le body

Dont je ne vais pas modifier la valeur dans le body

Je modifie la valeur d'une propriété qui est lié à une autre View (à la source)

J'utilise un propriété d'état « bindé »

```
@Binding var name: String
```

Je modifie la valeur d'une propriété qui est stocké dans la View elle même

J'utilise une propriété d'état @State

```
@State var name = ""
```

J'utilise une propriété stocké simple

```
var name = ""
```