

Miguel Ruiz

iOS Developer in Cornershop



@migueloruiz



@hola_miguelo

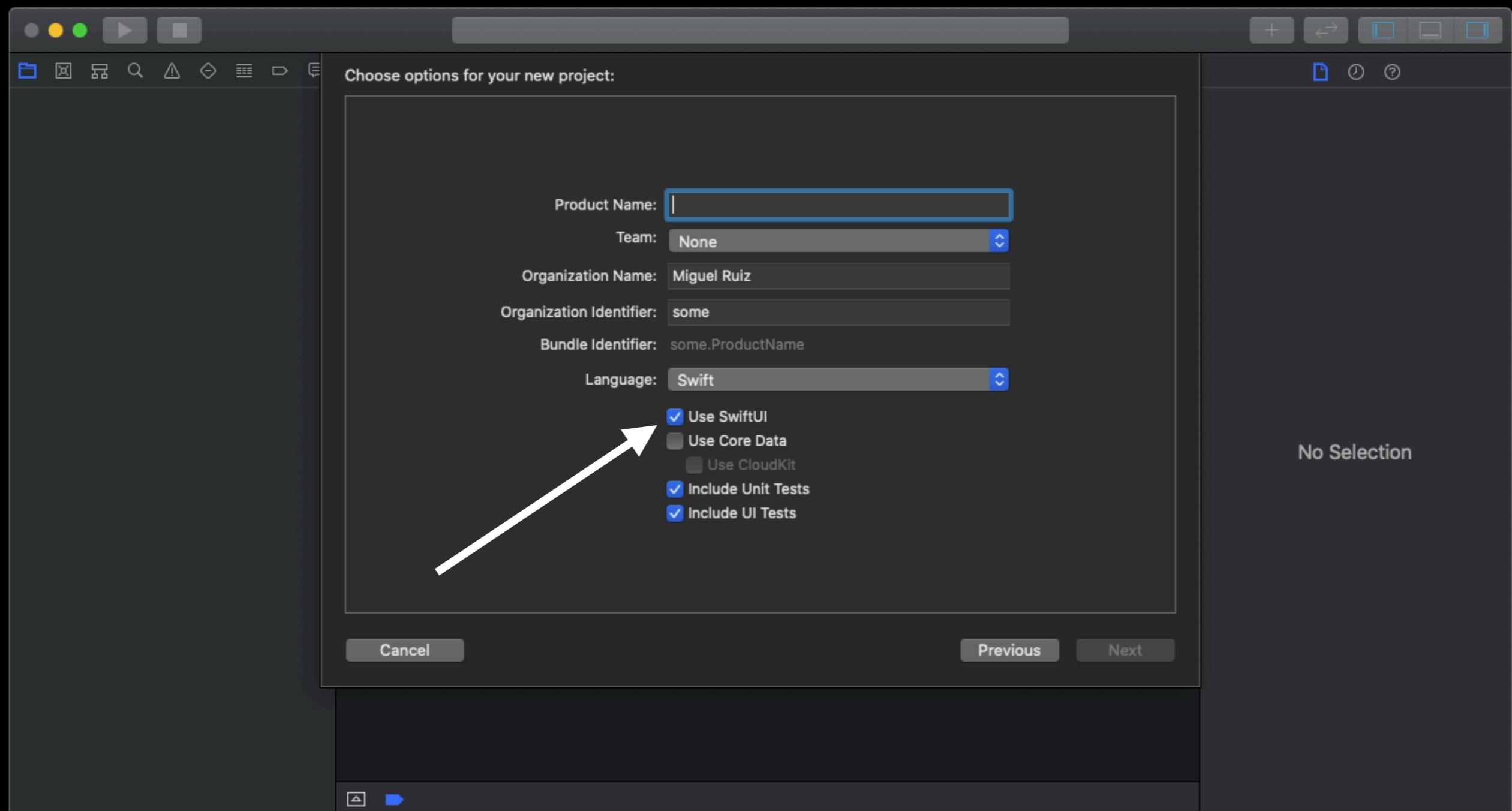
Explorando SwiftUI

Disclaimer

- Xcode 11 Beta 5
- Mac OS Catalina Beta 10.15



Activar SwiftUI



Cards | Preview Cards: Succeeded | Today at 23:25

```
1 import SwiftUI
2
3 struct CardView: View {
4     var body: some View {
5         Text("Hello World!")
6     }
7 }
8
9
10 #if DEBUG
11 struct CardView_Previews : PreviewProvider {
12     static var previews: some View {
13         CardView()
14     }
15 }
16 #endif
17
```



The image shows a screenshot of a macOS application window titled "Cards | Preview Cards: Succeeded | Today at 23:25". The window contains a code editor on the left and a preview area on the right. The code editor displays Swift code for a SwiftUI view named "CardView" and its preview provider "CardView_Previews". The preview area shows an iPhone Xr simulator with a white screen displaying the text "Hello World!". Below the simulator are two circular icons: one with a play button and another with a camera. The bottom right corner of the preview area has a "Preview" label. The window has standard OS X-style controls at the top and bottom.

OpaqueTypes

```
3
4 struct CardView: View {
5     var body: some View {
6         Text("Hello World!")
7     }
8 }
9
```

View

```
9338 /// A piece of user interface.  
9339 ///  
9340 /// You create custom views by declaring types that conform to the `View`  
9341 /// protocol. Implement the required `body` property to provide the content  
9342 /// and behavior for your custom view.  
9343 @available(iOS 13.0, OSX 10.15, tvOS 13.0, watchOS 6.0, *)  
9344 public protocol View {  
9345  
9346     /// The type of view representing the body of this view.  
9347     ///  
9348     /// When you create a custom view, Swift infers this type from your  
9349     /// implementation of the required `body` property.  
9350     associatedtype Body : View  
9351  
9352     /// Declares the content and behavior of this view.  
9353     var body: Self.Body { get }  
9354 }
```

Cards | Preview Cards: Succeeded | Today at 23:25

```
1 import SwiftUI
2
3 struct CardView: View {
4     var body: some View {
5         Text("Hello World!")
6     }
7 }
8
9
10 #if DEBUG
11 struct CardView_Previews : PreviewProvider {
12     static var previews: some View {
13         CardView()
14     }
15 }
16 #endif
17
```



The image shows a screenshot of a macOS application window titled "Cards | Preview Cards: Succeeded | Today at 23:25". The window contains a code editor on the left and a preview area on the right. The code editor displays Swift code for a SwiftUI view named "CardView" and its preview provider "CardView_Previews". The preview area shows an iPhone Xr simulator with a white screen displaying the text "Hello World!". Below the simulator are two circular icons: one with a play button and another with a camera. The bottom right corner of the preview area has a "Preview" label. The window has standard OS X-style controls at the top and bottom.

Cards | Preview Cards: Failed | Today at 23:02

! 1

Cards > iPhone XR

No Selection

Failed to build the scheme "Cards" ⓘ Try Again Diagnostics

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7     }
8 }
9
10
11 #if DEBUG
12 struct CardIDLabel_Previews: PreviewProvider {
13     static var previews: some View {
14         CardIDLabel() ⓘ Missing argument for parameter 'value' in call
15     }
16 }
17 #endif
18
```

iPhone XR

Failed to build the scheme "Cards"

Try Again Diagnostics

Preview

50%

CardIDLabel.swift

Hello World!

The screenshot shows the Xcode interface with a dark theme. In the top bar, it says 'Cards | Preview Cards: Failed | Today at 23:02'. A red exclamation mark icon with the number '1' indicates an error. Below the bar, the file path 'Cards > iPhone XR' is shown, along with 'No Selection'. A message 'Failed to build the scheme "Cards"' with a help icon is displayed. There are 'Try Again' and 'Diagnostics' buttons. The main area shows a Swift code editor with syntax highlighting and a red callout for a warning. To the right is a preview window for an iPhone XR showing the text 'Hello World!'. Below the preview are icons for play and stop, and the word 'Preview'. At the bottom, there's a zoom control set to 50%.

Cards | Preview Cards: Succeeded | Today at 23:03

```
1 |
2 import SwiftUI
3
4 struct CardIDLabel: View {
5     let value: String
6     var body: some View {
7         Text(self.value)
8     }
9 }
10
11 #if DEBUG
12 struct CardIDLabel_Previews: PreviewProvider {
13     static var previews: some View {
14         CardIDLabel(value: "34567453346")
15     }
16 }
17 #endif
18
```

34567453346

Preview

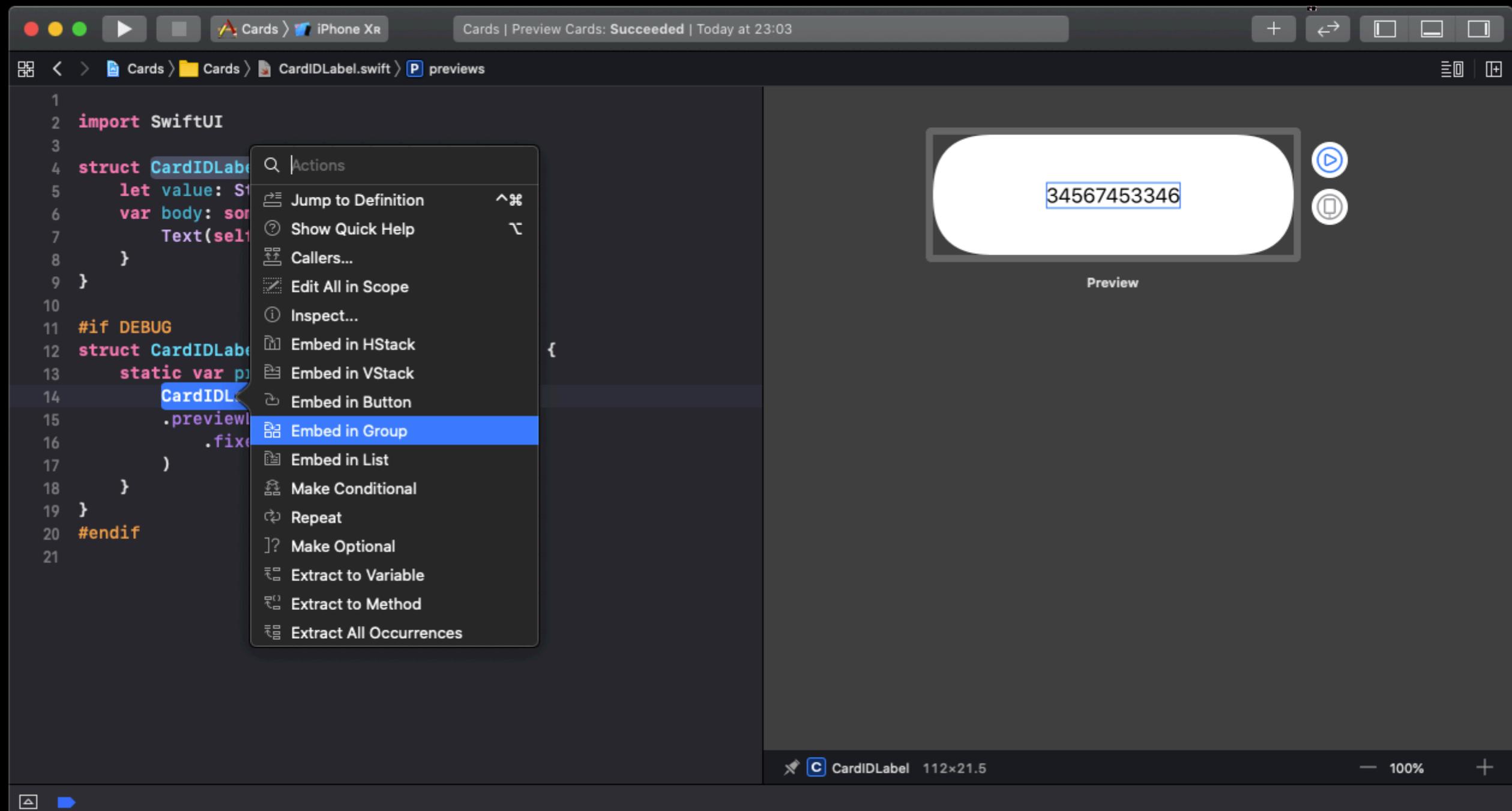
— 50% +

Previews

Cards | Preview Cards: Succeeded | Today at 23:03

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7     }
8 }
9
10 #if DEBUG
11 struct CardIDLabel_Previews: PreviewProvider {
12     static var previews: some View {
13         CardIDLabel(value: "34567453346")
14         .previewLayout(
15             .fixed(width: 300, height: 100)
16         )
17     }
18 }
19 }
20 #endif
21
```

The image shows a screenshot of the Xcode IDE. The top bar displays the title 'Cards | Preview Cards: Succeeded | Today at 23:03'. The main area contains the Swift code for 'CardIDLabel.swift'. On the right side, there is a preview window showing a rounded rectangular view with the text '34567453346'. Below the preview window, the word 'Preview' is visible. To the right of the preview window, there are two circular icons: one with a play symbol and another with a device symbol. The bottom right corner of the Xcode window shows a zoom level of '100%'.



⌘ + Click

Cards | Preview Cards: Succeeded | Today at 23:03

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7     }
8 }
9
10 #if DEBUG
11 struct CardIDLabel_Previews: PreviewProvider {
12     static var previews: some View {
13         Group {
14             CardIDLabel(value: "34567453346")
15                 .environment(\.colorScheme, .light)
16
17             CardIDLabel(value: "34567453346")
18                 .environment(\.colorScheme, .dark)
19         }
20         .previewLayout(
21             .fixed(width: 300, height: 100)
22         )
23     }
24 }
25 #endif
26
```

Preview

Preview

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:03 + ↗ □ □ □

Cards > Cards > CardIDLabel.swift > No Selection

```
1 |
2 import SwiftUI
3
4 struct CardIDLabel: View {
5     let value: String
6     var body: some View {
7         Text(self.value)
8             .foregroundColor(.secondary)
9     }
10 }
11
12 #if DEBUG
13 struct CardIDLabel_Previews: PreviewProvider {
14     static var previews: some View {
15         Group {
16             CardIDLabel(value: "34567453346")
17                 .environment(\.colorScheme, .light)
18
19             CardIDLabel(value: "34567453346")
20                 .environment(\.colorScheme, .dark)
21         }
22         .previewLayout(
23             .fixed(width: 300, height: 100)
24         )
25     }
26 }
27 #endif
28
```

34567453346

Preview

34567453346

Preview

— 100% +

Cards | Preview Cards: Succeeded | Today at 23:03

```
1 |
2 import SwiftUI
3
4 struct CardIDLabel: View {
5     let value: String
6     var body: some View {
7         Text(self.value)
8             .foregroundColor(.secondary)
9             .fontWeight(.bold)
10    }
11 }
12
13 #if DEBUG
14 struct CardIDLabel_Previews: PreviewProvider {
15     static var previews: some View {
16         Group {
17             CardIDLabel(value: "34567453346")
18                 .environment(\.colorScheme, .light)
19
20             CardIDLabel(value: "34567453346")
21                 .environment(\.colorScheme, .dark)
22         }
23         .previewLayout(
24             .fixed(width: 300, height: 100)
25         )
26     }
27 }
28 #endif
29
```

Preview

Preview

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:03 + ← □ □

Cards > Cards > CardIDLabel.swift > No Selection

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7             .foregroundColor(.secondary)
8             .fontWeight(.bold)
9             .background(Color.gray)
10    }
11 }
12
13
14 #if DEBUG
15 struct CardIDLabel_Previews: PreviewProvider {
16     static var previews: some View {
17         Group {
18             CardIDLabel(value: "34567453346")
19                 .environment(\.colorScheme, .light)
20
21             CardIDLabel(value: "34567453346")
22                 .environment(\.colorScheme, .dark)
23         }
24         .previewLayout(
25             .fixed(width: 300, height: 100)
26         )
27     }
28 }
29 #endif
30
```

34567453346

Preview

34567453346

Preview

— 100% +

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:03 + ↗ □ □

Cards > Cards > CardIDLabel.swift > body

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7             .foregroundColor(.secondary)
8             .fontWeight(.bold)
9             .background(Color.gray)
10            .padding(10)
11    }
12 }
13
14 #if DEBUG
15 struct CardIDLabel_Previews: PreviewProvider {
16     static var previews: some View {
17         Group {
18             CardIDLabel(value: "34567453346")
19                 .environment(\.colorScheme, .light)
20
21             CardIDLabel(value: "34567453346")
22                 .environment(\.colorScheme, .dark)
23         }
24         .previewLayout(
25             .fixed(width: 300, height: 100)
26         )
27     }
28 }
29
30 #endif
```

Preview

Preview

Text 137.5x41.5 — 100% +

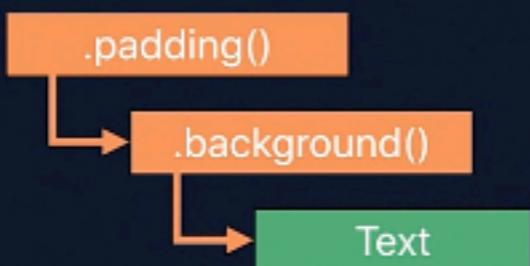
Modifiers

```
func foregroundColor(_ color: Color?) -> some View
```

```
func background<S>(_ content: StaticMember<S>) -> some View where S : Shape  
Style
```

```
func padding(_ insets: EdgeInsets) -> some View
```

```
Text("🥑🍞")  
.background(Color.green, cornerRadius: 12)  
.padding(.all)
```



```
Text("🥑🍞")  
.padding(.all)  
.background(Color.green, cornerRadius: 12)
```



Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:03 + ↗ □ □

Cards > Cards > CardIDLabel.swift > body

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7             .foregroundColor(.secondary)
8             .fontWeight(.bold)
9             .background(Color.gray)
10            .padding(10)
11    }
12 }
13
14 #if DEBUG
15 struct CardIDLabel_Previews: PreviewProvider {
16     static var previews: some View {
17         Group {
18             CardIDLabel(value: "34567453346")
19                 .environment(\.colorScheme, .light)
20
21             CardIDLabel(value: "34567453346")
22                 .environment(\.colorScheme, .dark)
23         }
24         .previewLayout(
25             .fixed(width: 300, height: 100)
26         )
27     }
28 }
29
30 #endif
```

Preview

Preview

Text 137.5x41.5 — 100% +

Cards | Preview Cards: Succeeded | Today at 23:03

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7             .foregroundColor(.secondary)
8             .fontWeight(.bold)
9             .padding(10)
10            .background(Color.gray)
11    }
12 }
13
14 #if DEBUG
15 struct CardIDLabel_Previews: PreviewProvider {
16     static var previews: some View {
17         Group {
18             CardIDLabel(value: "34567453346")
19                 .environment(\.colorScheme, .light)
20
21             CardIDLabel(value: "34567453346")
22                 .environment(\.colorScheme, .dark)
23         }
24         .previewLayout(
25             .fixed(width: 300, height: 100)
26         )
27     }
28 }
29
30 #endif
```

Preview

Preview

Text 137.5x41.5 100% +

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:03 + ↗ □ □

Cards > Cards > CardIDLabel.swift > No Selection

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7             .foregroundColor(.secondary)
8             .fontWeight(.bold)
9             .padding(10)
10            .background(Color.gray)
11            .cornerRadius(20)
12    }
13 }
14
15 #if DEBUG
16 struct CardIDLabel_Previews: PreviewProvider {
17     static var previews: some View {
18         Group {
19             CardIDLabel(value: "34567453346")
20                 .environment(\.colorScheme, .light)
21
22             CardIDLabel(value: "34567453346")
23                 .environment(\.colorScheme, .dark)
24         }
25         .previewLayout(
26             .fixed(width: 300, height: 100)
27         )
28     }
29 }
30
31 #endif
```

Preview

Preview

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:03 + ↗ □ □

Cards > Cards > CardIDLabel.swift > No Selection

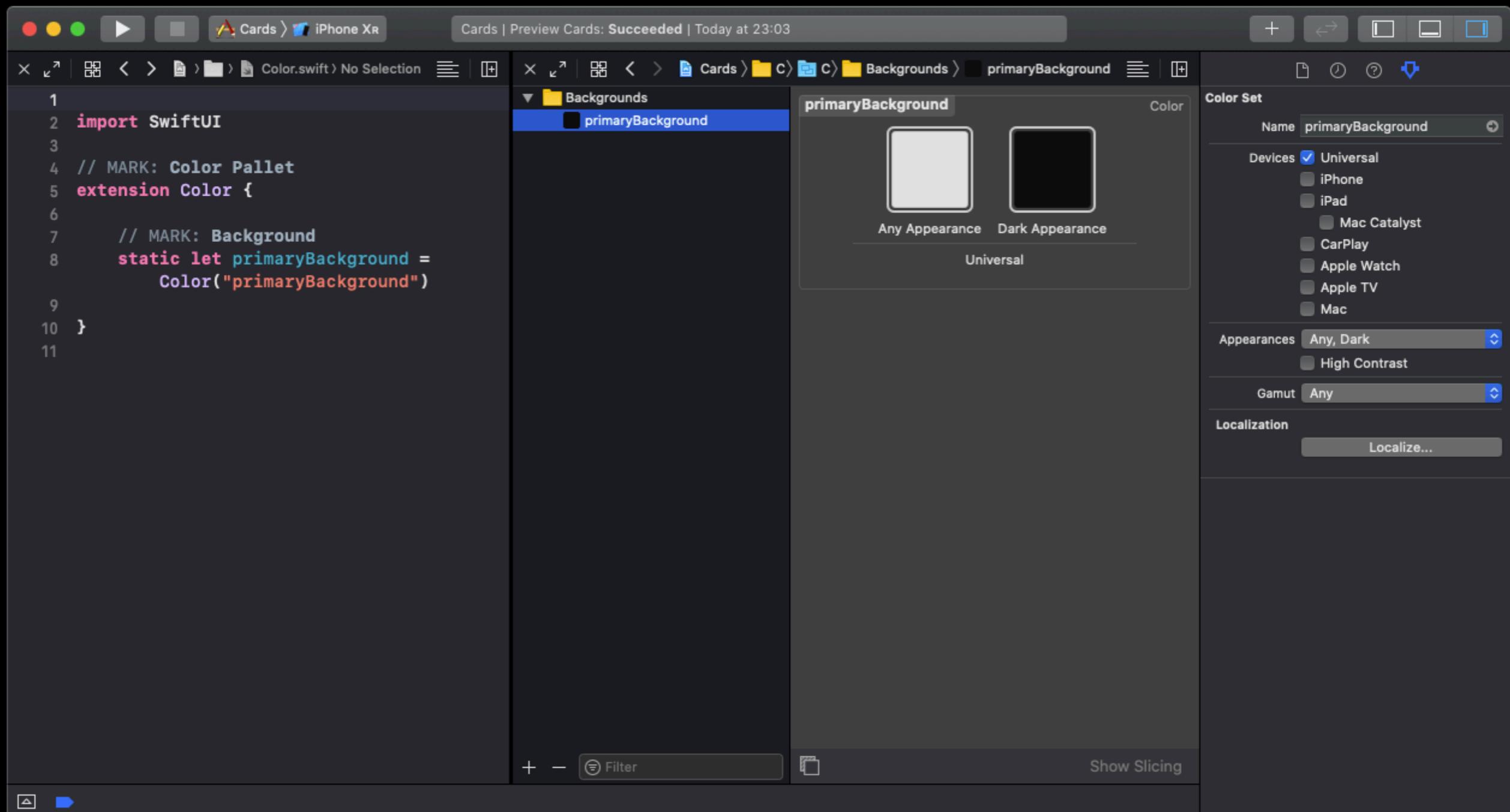
```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7             .foregroundColor(.secondary)
8             .fontWeight(.bold)
9             .frame(minWidth: 0, maxWidth: .infinity)
10            .padding(10)
11            .background(Color.gray)
12            .cornerRadius(20)
13    }
14 }
15
16
17 #if DEBUG
18 struct CardIDLabel_Previews: PreviewProvider {
19     static var previews: some View {
20         Group {
21             CardIDLabel(value: "34567453346")
22                 .environment(\.colorScheme, .light)
23
24             CardIDLabel(value: "34567453346")
25                 .environment(\.colorScheme, .dark)
26         }
27         .previewLayout(
28             .fixed(width: 300, height: 100)
29         )
30     }
31 }
```

Preview

Preview

— 100% +

Assets colors for dark mode



Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:19 + ↻ □ □

Cards > Cards > CardIDLabel.swift > body

```
Automatic preview updating paused ⓘ Resume
```

```
1 import SwiftUI
2
3 struct CardIDLabel: View {
4     let value: String
5     var body: some View {
6         Text(self.value)
7             .foregroundColor(.secondary)
8             .fontWeight(.bold)
9             .frame(minWidth: 0, maxWidth: .infinity)
10            .padding(10)
11            .background(Color.primaryBackground)
12            .cornerRadius(20)
13    }
14 }
15
16
17 #if DEBUG
18 struct CardIDLabel_Previews: PreviewProvider {
19     static var previews: some View {
20         Group {
21             CardIDLabel(value: "34567453346")
22                 .environment(\.colorScheme, .light)
23
24             CardIDLabel(value: "34567453346")
25                 .environment(\.colorScheme, .dark)
26         }
27         .previewLayout(
28             .fixed(width: 300, height: 100)
29         )
30     }
31 }
```

34567453346

Preview

34567453346

Preview

— 100% +

Cards | Preview Cards: Succeeded | Today at 23:25

```
1 import SwiftUI
2
3 struct CardView: View {
4     var body: some View {
5         Text("Hello World!")
6     }
7 }
8
9
10 #if DEBUG
11 struct CardView_Previews : PreviewProvider {
12     static var previews: some View {
13         CardView()
14     }
15 }
16 #endif
17
```



The image shows a screenshot of a macOS application window titled "Cards | Preview Cards: Succeeded | Today at 23:25". The window contains a code editor on the left and a preview area on the right. The code editor displays Swift code for a SwiftUI view named "CardView" and its preview provider "CardView_Previews". The preview area shows an iPhone Xr simulator with a white screen displaying the text "Hello World!". Below the simulator are two circular icons: a play button and a refresh/circular arrow. The bottom right corner of the preview area has a "Preview" label. The window has standard OS X window controls (red, yellow, green buttons) and a toolbar with various icons.

Cards | Preview Cards: Succeeded | Today at 23:25

```
1 |
2 import SwiftUI
3
4 struct CardView: View {
5     var body: some View {
6         Text("Hello World!")
7     }
8 }
9
10 #if DEBUG
11 struct CardView_Previews : PreviewProvider {
12     static var previews: some View {
13         Group {
14             CardView()
15         }
16         .previewLayout(.fixed(width: 400, height: 400))
17     }
18 }
19 #endif
20
```

Hello World!

Preview

Play

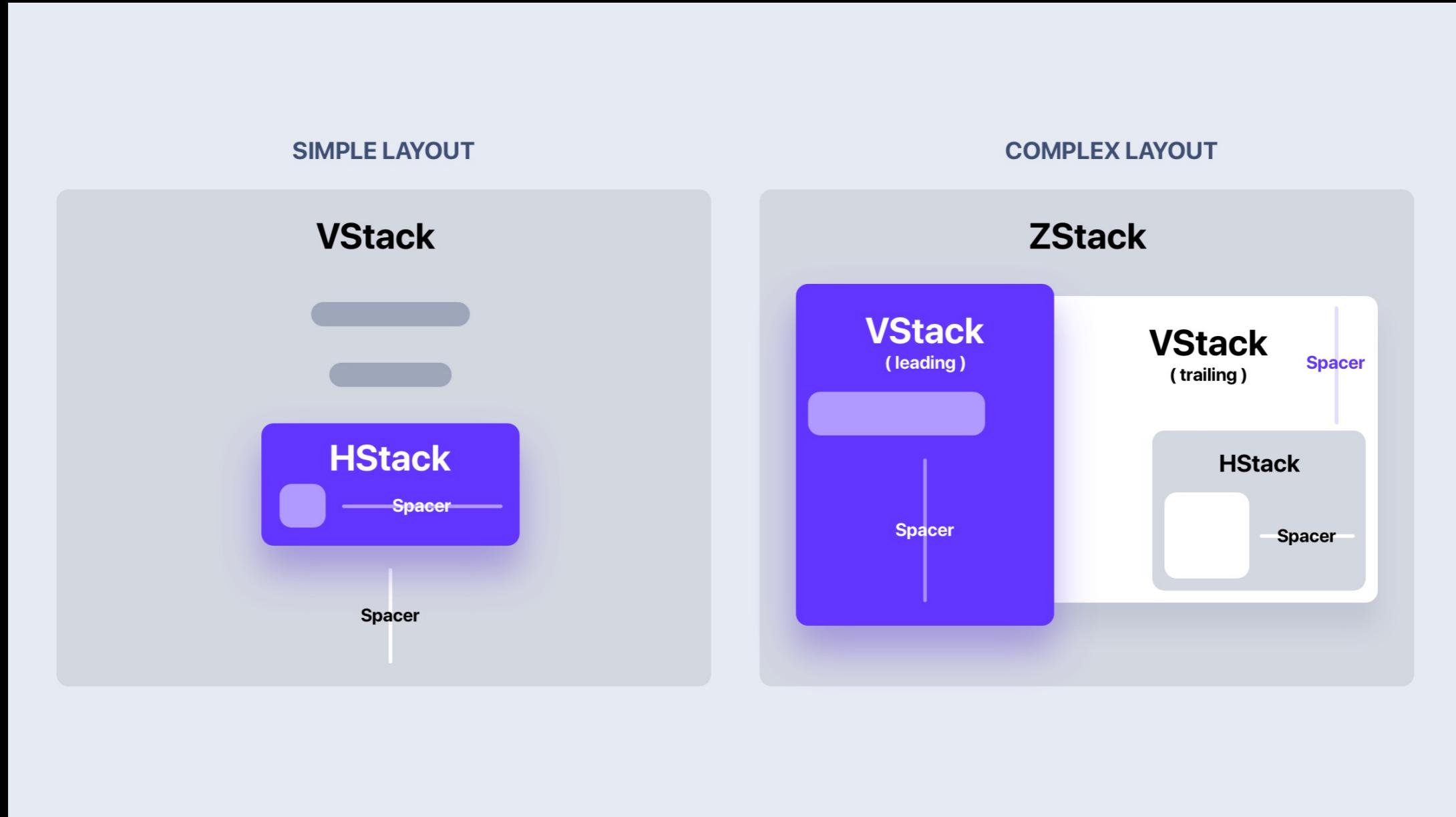
iPhone

Cards | Preview Cards: Succeeded | Today at 23:27

```
1 |
2 import SwiftUI
3
4 struct CardView: View {
5
6     var body: some View {
7         Image("bip_logo")
8     }
9 }
10
11 #if DEBUG
12 struct CardView_Previews : PreviewProvider {
13     static var previews: some View {
14         Group {
15             CardView()
16         }
17         .previewLayout(.fixed(width: 400, height: 400))
18     }
19 }
20 #endif
21
```

The screenshot shows the Xcode interface with the preview window open. The preview window displays a white rounded rectangle containing the text "bip!" in a large, bold, yellow font. Below the preview window, there are two small circular icons: one with a play button symbol and another with a phone symbol. At the bottom right of the preview window, the word "Preview" is visible. The overall interface is dark-themed.

Stacks and Spacer



*Por defecto todos los views intentaran ocupar el menor espacio

Cards | Preview Cards: Succeeded | Today at 23:27

```
1 |
2 import SwiftUI
3
4 struct CardView: View {
5
6     var body: some View {
7         HStack(alignment: .top) {
8             Image("bip_logo")
9             Spacer()
10        }
11    }
12 }
13
14 #if DEBUG
15 struct CardView_Previews : PreviewProvider {
16     static var previews: some View {
17         Group {
18             CardView()
19         }
20         .previewLayout(.fixed(width: 400, height: 400))
21     }
22 }
23 #endif
24
```

The screenshot shows the Xcode Preview interface. On the left, the code for a SwiftUI view named `CardView` is displayed. The code defines a simple card layout with an image and a spacer. A preview of the card is shown on the right, featuring a white background with the word "bip!" in a large, bold, yellow font. Below the preview, there are two circular icons: one with a play button and another with a device. At the bottom of the preview window, the word "Preview" is visible. The overall interface has a dark mode aesthetic.

Cards | Preview Cards: Succeeded | Today at 23:27

```
1 import SwiftUI
2
3 struct CardView: View {
4     var body: some View {
5         VStack {
6             HStack(alignment: .top) {
7                 Image("bip_logo")
8                 Spacer()
9             }
10            CardIDLabel(value: "3453245345")
11        }
12    }
13
14 #if DEBUG
15 struct CardView_Previews : PreviewProvider {
16     static var previews: some View {
17         Group {
18             CardView()
19         }
20         .previewLayout(.fixed(width: 400, height: 400))
21     }
22 }
23
24 #endif
25
26
27
28
```

bip!

3453245345

Preview

Play

iPhone

Cards | Preview Cards: Succeeded | Today at 23:30

```
1 |
2 import SwiftUI
3
4 struct CardView: View {
5
6     var body: some View {
7         VStack {
8             HStack(alignment: .top) {
9                 Image("bip_logo")
10                Spacer()
11            }
12
13            Spacer()
14
15            CardIDLabel(value: "3453245345")
16        }
17    }
18 }
19
20 #if DEBUG
21 struct CardView_Previews : PreviewProvider {
22     static var previews: some View {
23         Group {
24             CardView()
25         }
26         .previewLayout(.fixed(width: 400, height: 400))
27     }
28 }
29 #endif
30
```

The image shows a SwiftUI preview interface. On the left is the code editor with the CardView.swift file open. On the right is the preview window titled 'Preview'. The preview displays a white rounded rectangle containing a yellow 'bip!' logo at the top, followed by a large white space (Spacer), and a grey button-like element at the bottom with the text '3453245345'.

Cards | Preview Cards: Succeeded | Today at 23:30

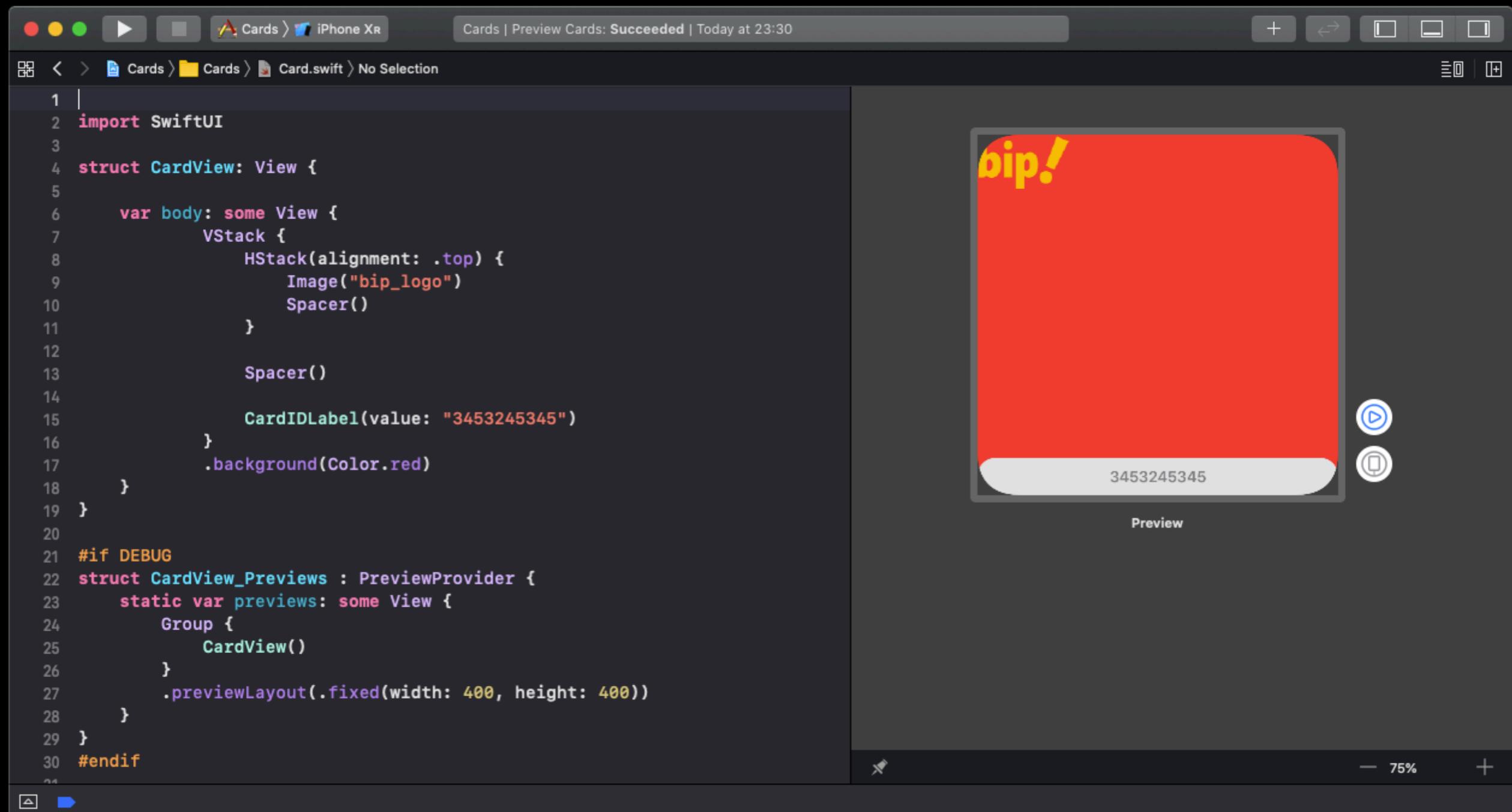
```
1 |
2 import SwiftUI
3
4 struct CardView: View {
5
6     var body: some View {
7         VStack {
8             HStack(alignment: .top) {
9                 Image("bip_logo")
10                Spacer()
11            }
12
13            Spacer()
14
15            CardIDLabel(value: "3453245345")
16        }
17        .background(Color.red)
18    }
19 }
20
21 #if DEBUG
22 struct CardView_Previews : PreviewProvider {
23     static var previews: some View {
24         Group {
25             CardView()
26         }
27         .previewLayout(.fixed(width: 400, height: 400))
28     }
29 }
30 #endif
```

bip!

3453245345

Preview

— 75% +



Cards | Preview Cards: Succeeded | Today at 23:30

```
1 |
2 import SwiftUI
3
4 struct CardView: View {
5
6     var body: some View {
7         VStack {
8             HStack(alignment: .top) {
9                 Image("bip_logo")
10                Spacer()
11            }
12
13            Spacer()
14
15            CardIDLabel(value: "3453245345")
16        }
17        .padding(10)
18        .background(Color.red)
19    }
20 }
21
22 #if DEBUG
23 struct CardView_Previews : PreviewProvider {
24     static var previews: some View {
25         Group {
26             CardView()
27         }
28         .previewLayout(.fixed(width: 400, height: 400))
29     }
30 }
31 #endif
```

bip!

3453245345

Preview

— 75% +



Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:33 + ↗

Cards > Cards > Card.swift > No Selection

Automatic preview updating paused ⓘ Resume

```
1 import SwiftUI
2
3 struct CardView: View {
4
5     private struct Constants {
6         static let cardAspectRatio: CGFloat = 1.58
7     }
8
9
10    var body: some View {
11        VStack {
12            HStack(alignment: .top) {
13                Image("bip_logo")
14                Spacer()
15            }
16
17            Spacer()
18
19            CardIDLabel(value: "3453245345")
20        }
21        .padding(10)
22        .background(Color.red)
23        .aspectRatio(CGSize(width: Constants.cardAspectRatio, height:
24            1), contentMode: .fit)
25    }
26
27 #if DEBUG
28 struct CardView_Previews : PreviewProvider {
29     static var previews: some View {
30         Group {
```

bip!

3453245345

Preview

— 75% +

Cards | Preview Cards: Succeeded | Today at 23:35

```
1
2 import SwiftUI
3
4 struct CardView: View {
5
6     private struct Constants {
7         static let cardAspectRatio: CGFloat = 1.58
8         static let cornerRadius: CGFloat = 8
9     }
10
11    var body: some View {
12        VStack {
13            HStack(alignment: .top) {
14                Image("bip_logo")
15                Spacer()
16            }
17
18            Spacer()
19
20            CardIDLabel(value: "3453245345")
21        }
22        .padding(10)
23        .background(Color.red)
24        .aspectRatio(CGSize(width: Constants.cardAspectRatio, height:
25            1), contentMode: .fit)
26        .cornerRadius(Constants.cornerRadius)
27    }
28
29 #if DEBUG
30 struct CardView_Previews: PreviewProvider {
31     static var previews: some View {
32         CardView()
33     }
34 }
```

bip!

3453245345

Preview

75%

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:35 + ↗ □ □ □

Cards > Cards > Card.swift > No Selection

```
1 import SwiftUI
2
3 struct CardView: View {
4
5     private struct Constants {
6         static let cardAspectRatio: CGFloat = 1.58
7         static let cornerRadius: CGFloat = 8
8     }
9
10    var body: some View {
11        VStack {
12            HStack(alignment: .top) {
13                Image("bip_logo")
14                Spacer()
15            }
16
17            Spacer()
18
19            CardIDLabel(value: "3453245345")
20        }
21        .padding(10)
22        .background(
23            Rectangle()
24            .fill(
25                LinearGradient(
26                    gradient: Gradient(colors: [Color.red, Color.blue]),
27                    startPoint: .top,
28                    endPoint: .bottom
29                )
30            )
31        )
32        .cornerRadius(Constants.cornerRadius)
33    }
34    .aspectRatio(CGSize(width: Constants.cardAspectRatio, height: 1), contentMode: .fit)
35    .cornerRadius(Constants.cornerRadius)
36}
37
38 #if DEBUG
39 struct CardView_Previews : PreviewProvider {
40     static var previews: some View {
41         CardView()
42     }
43 }
```

bip!

3453245345

Preview

75%



Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 23:35 + ↗ □ □ □

Cards > Cards > Card.swift > previews

```
22     .padding(10)
23     .background(
24         Rectangle()
25             .fill(
26                 LinearGradient(
27                     gradient: Gradient(colors: [Color.red, Color.blue]),
28                     startPoint: .top,
29                     endPoint: .bottom
30                 )
31             )
32             .cornerRadius(Constants.cornerRadius)
33     )
34     .aspectRatio(CGSize(width: Constants.cardAspectRatio, height: 1), contentMode: .fit)
35     .cornerRadius(Constants.cornerRadius)
36 }
37 }
38
39 #if DEBUG
40 struct CardView_Previews : PreviewProvider {
41     static var previews: some View {
42         Group {
43             CardView()
44                 .environment(\.colorScheme, .light)
45
46             CardView()
47                 .environment(\.colorScheme, .dark)
48         }
49         .padding(20)
50         .previewLayout(.fixed(width: 400, height: 400))
51     }
52 }
53 #endif
54
```

bip!

3453245345

Preview

bip!

3453245345

Preview

CardView 360x228 — 50% +

CardsStack

Cards | Preview Cards: Succeeded | Today at 23:43

```
1 import SwiftUI
2
3 struct CardsStackView: View {
4     var body: some View {
5         VStack {
6             CardView()
7             CardView()
8             CardView()
9             CardView()
10            CardView()
11        }
12    }
13 }
14
15 #if DEBUG
16 struct CardsStackView_Previews: PreviewProvider {
17     static var previews: some View {
18         CardsStackView()
19     }
20 }
21
22 #endif
```

The image shows a screenshot of a Mac OS X desktop with a SwiftUI code editor window open. The code editor displays a Swift file named 'CardsStackView.swift' containing a struct definition for 'CardsStackView' that uses a VStack to stack multiple 'CardView' instances. A preview panel on the right shows an iPhone Xr displaying five identical cards, each with the 'bip!' logo and the phone number '3453245345'. Below the preview are two circular icons: a play button and a stop button. The bottom of the window has a toolbar with a close button, a 50% zoom button, and a plus sign button.

Cards | Preview Cards: Succeeded | Today at 23:43

```
1 import SwiftUI
2
3 struct CardsStackView: View {
4     var body: some View {
5         ZStack {
6             CardView()
7             CardView()
8             CardView()
9             CardView()
10            CardView()
11        }
12    }
13 }
14
15
16 #if DEBUG
17 struct CardsStackView_Previews: PreviewProvider {
18     static var previews: some View {
19         CardsStackView()
20     }
21 }
22 #endif
23
```

The screenshot shows a SwiftUI preview of an iPhone Xr displaying a stack of five cards. The top card has a red gradient background and the word "bip!" in white. Below it is a blue card with the number "3453245345". The preview interface includes a play button, a stop button, and a zoom slider set to 50%.

Cards | Preview Cards: Succeeded | Today at 23:45

```
1 |
2 import SwiftUI
3
4 struct CardsStackView: View {
5     var body: some View {
6         ZStack {
7             CardView()
8             CardView()
9             CardView()
10            CardView()
11            CardView()
12        }
13    }
14 }
15
16 #if DEBUG
17 struct CardsStackView_Previews : PreviewProvider {
18     static var previews: some View {
19         Group {
20             CardsStackView()
21         }
22         .padding(20)
23         .previewLayout(.fixed(width: 400, height: 400))
24     }
25 }
26 #endif
27
28
```

bip!

3453245345

Preview

Play

iPhone

Cards | Preview Cards: Succeeded | Today at 23:45

```
1 |
2 import SwiftUI
3
4 struct CardsStackView: View {
5     var body: some View {
6         ZStack {
7             CardView()
8                 .opacity(0.5)
9             CardView()
10                .opacity(0.5)
11             CardView()
12                .opacity(0.5)
13             CardView()
14                 .opacity(0.5)
15         }
16     }
17 }
18
19 #if DEBUG
20 struct CardsStackView_Previews : PreviewProvider {
21     static var previews: some View {
22         Group {
23             CardsStackView()
24         }
25         .padding(20)
26         .previewLayout(.fixed(width: 400, height: 400))
27     }
28 }
29 #endif
30
31
```

Preview

75%

Cards | Preview Cards: Succeeded | Today at 23:45

```
1 |
2 import SwiftUI
3
4 struct CardsStackView: View {
5     var body: some View {
6         ZStack {
7             CardView()
8                 .scaleEffect(0.7)
9                 .opacity(0.5)
10            CardView()
11                .scaleEffect(0.8)
12                .opacity(0.5)
13            CardView()
14                .scaleEffect(0.9)
15                .opacity(0.5)
16            CardView()
17                .scaleEffect(1)
18                .opacity(0.5)
19        }
20    }
21 }
22
23 #if DEBUG
24 struct CardsStackView_Previews : PreviewProvider {
25     static var previews: some View {
26         Group {
27             CardsStackView()
28         }
29         .padding(20)
30         .previewLayout(.fixed(width: 400, height: 400))
31     }
32 }
```

Preview

The image shows a Xcode preview of a SwiftUI view named 'CardsStackView'. The view consists of a stack of four cards. Each card is a 'CardView' with specific opacity and scale effects applied. The cards are labeled 'bip!' and feature a blue gradient bar at the bottom with the number '3453245345'. The preview interface includes a play button and a device icon.

Cards | Preview Cards: Succeeded | Yesterday at 23:45

```
1 import SwiftUI
2
3 struct CardsStackView: View {
4     var body: some View {
5         ZStack {
6             CardView()
7                 .offset(y: 60)
8                 .scaleEffect(0.7)
9                 .opacity(0.5)
10            CardView()
11                .offset(y: 40)
12                .scaleEffect(0.8)
13                .opacity(0.5)
14            CardView()
15                .offset(y: 20)
16                .scaleEffect(0.9)
17                .opacity(0.5)
18            CardView()
19                .offset(y: 0)
20                .scaleEffect(1)
21                .opacity(0.5)
22        }
23    }
24 }
25
26 #if DEBUG
27 struct CardsStackView_Previews : PreviewProvider {
28     static var previews: some View {
29         Group {
30             CardsStackView()
```

Preview



— 75% +

Cards | Preview Cards: Succeeded | Today at 0:03

```
1 |
2 import SwiftUI
3
4 struct CardsStackView: View {
5     var body: some View {
6         ZStack {
7             CardView()
8                 .offset(y: 60)
9                 .scaleEffect(0.7)
10            CardView()
11                .offset(y: 40)
12                .scaleEffect(0.8)
13            CardView()
14                .offset(y: 20)
15                .scaleEffect(0.9)
16            CardView()
17                .offset(y: 0)
18                .scaleEffect(1)
19        }
20    }
21 }
22
23 #if DEBUG
24 struct CardsStackView_Previews : PreviewProvider {
25     static var previews: some View {
26         Group {
27             CardsStackView()
28         }
29         .padding(20)
30         .previewLayout(.fixed(width: 400, height: 400))
31     }
32 }
```

Preview

— 100% +

Identifiable

```
struct Card: Identifiable {
    let id: String
    let balance: String
    let vendor: CardVendor
}

enum CardVendor: CaseIterable {
    case bip
    case valpo

    var description: String { ... }
    var image: String { ... }
    var colorTop: Color { ... }
    var colorBottom: Color { ... }
}
```

```
40  #if DEBUG
41  extension Card {
42      static let previewContent: [Card] = [
43          Card(
44              id: UUID().uuidString,
45              balance: "1000",
46              vendor: .bip
47          ),
48          Card(
49              id: UUID().uuidString,
50              balance: "1500",
51              vendor: .valpo
52          ),
53          Card(
54              id: UUID().uuidString,
55              balance: "2000",
56              vendor: .bip
57          ),
58          Card(
59              id: UUID().uuidString,
60              balance: "500",
61              vendor: .valpo
62      )
63  ]
64 }
65 #endif
```

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 21:36 + ↗ □ □ □

Cards > Cards > Card.swift > body

```
40
41 #if DEBUG
42 struct CardView_Previews : PreviewProvider {
43     static var previews: some View {
44         Group {
45             CardView(card: Card.previewContent[1])
46                 .scaledToFit()
47                 .environment(\.colorScheme, .light)
48
49             CardView(card: Card.previewContent[0])
50                 .scaledToFit()
51                 .environment(\.colorScheme, .dark)
52         }
53         .padding(20)
54         .previewLayout(.fixed(width: 400, height: 400))
55     }
56 }
57 #endif
58 }
```

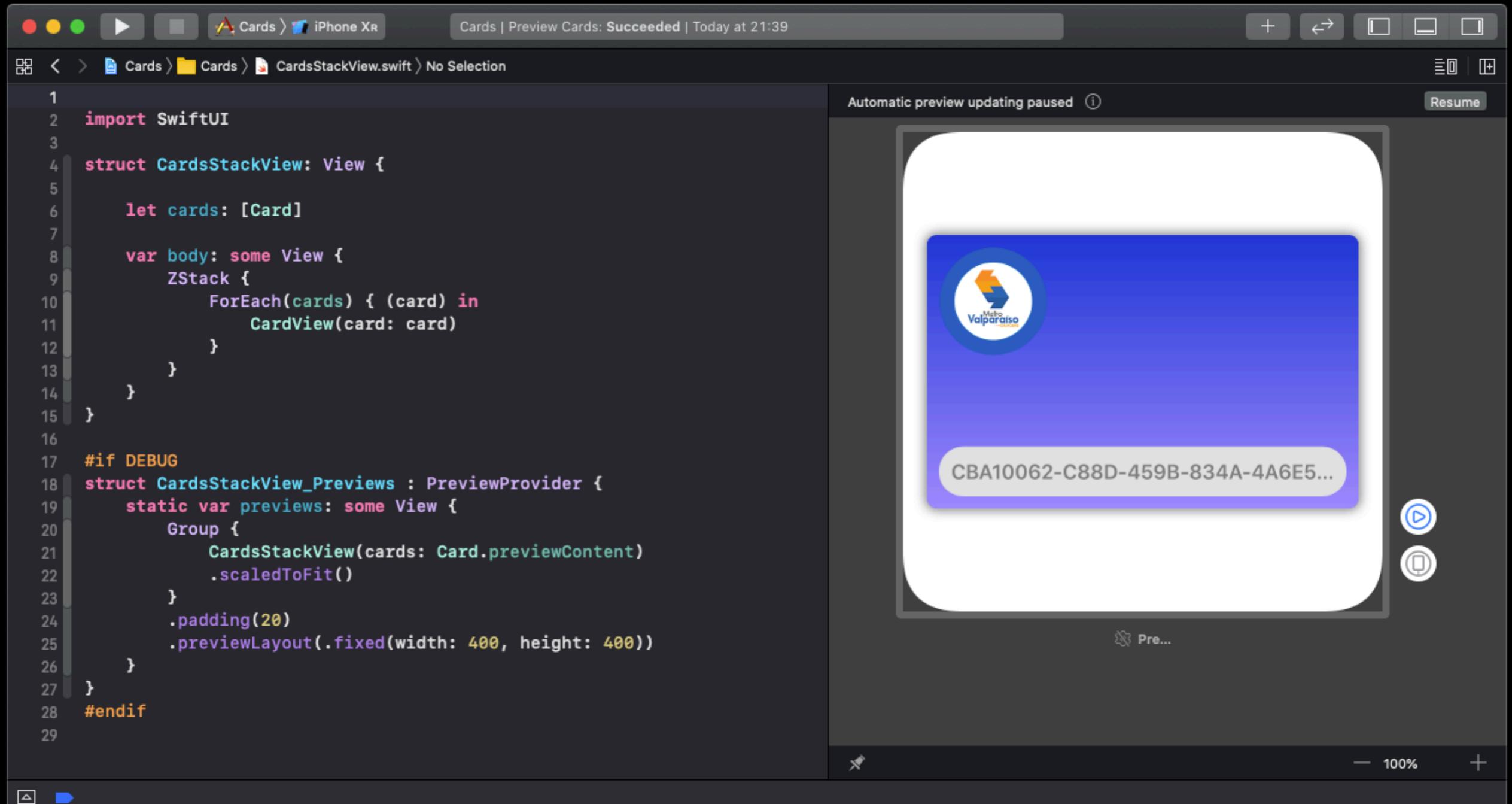
Preview

Preview

CardIDLabel 340x41.5 50% +

The screenshot shows the Xcode interface with a preview of two cards. The top card is light blue with a circular logo containing a stylized orange and yellow shape, and a placeholder ID '0840D203-2F4F-4804-8967-640AC...'. The bottom card is dark blue with the word 'bip!' in white, and a placeholder ID '2026C41F-81AC-4102-9CD0-91CB1C...'. Both cards have a white footer bar with a placeholder ID.

ForEach



A screenshot of the Xcode IDE showing a SwiftUI preview of a stack view. The code in the editor is for a `CardsStackView` struct, which contains a `ForEach` loop to render multiple `CardView`s. The preview shows a blue card with a circular logo for "Metro Valparaíso" and a purple card below it with a long ID: "CBA10062-C88D-459B-834A-4A6E5...". The Xcode interface includes a toolbar at the top, a navigation bar, and various controls for the preview.

```
1 import SwiftUI
2
3 struct CardsStackView: View {
4
5     let cards: [Card]
6
7     var body: some View {
8         ZStack {
9             ForEach(cards) { (card) in
10                 CardView(card: card)
11             }
12         }
13     }
14 }
15
16 #if DEBUG
17 struct CardsStackView_Previews : PreviewProvider {
18     static var previews: some View {
19         Group {
20             CardsStackView(cards: Card.previewContent)
21                 .scaledToFit()
22         }
23         .padding(20)
24         .previewLayout(.fixed(width: 400, height: 400))
25     }
26 }
27 }
28 #endif
29
```

```
private func getIndex(for card: Card) -> Int {
    return cards.firstIndex(where: { $0.id == card.id })!
}

private func cardScale(forIndex index: Int, totalCards: Int) -> CGFloat {
    guard totalCards > 1 else { return 1 }

    // Line Ecuation: x = (-y + a) / a
    let scaleFactor: Float = Float(-index + totalCards - 1) / Float(totalCards - 1)
    let scaleOffset = Constants.cardsScaleOffset * scaleFactor

    return CGFloat(1 - scaleOffset)
}

private func yPosition(forIndex index: Int, totalCards: Int) -> CGFloat {
    guard totalCards > 1 else { return 0 }

    // Line Ecuation: y = (x - a) * b
    let yPos: Float = Float(index - (totalCards - 1)) * -Constants.cardsVerticalOffset
    return CGFloat(yPos)
}
```

Cards > iPhone XR Cards | Preview Cards: Succeeded | Today at 22:08

Cards > Cards > CardsStackView.swift > CardsStackView

```
5
6     private struct Constants {
7         static let cardsVerticalOffset: Float = 15
8         static let cardsScaleOffset: Float = 0.2
9     }
10
11    let cards: [Card]
12
13    var body: some View {
14        ZStack {
15            ForEach(cards) { (card) in
16                CardView(card: card)
17                    .scaleEffect(self.cardScale(
18                        forIndex: self.getIndex(for: card),
19                        totalCards: self.cards.count)
20                    )
21                    .offset(y: self.yPosition(
22                        forIndex: self.getIndex(for: card),
23                        totalCards: self.cards.count)
24                    )
25            }
26        }
27    }
28
29    private func getIndex(for card: Card) -> Int {
30        return cards.firstIndex(where: { $0.id == card.id })!
31    }
32
33    private func cardScale(forIndex index: Int, totalCards: Int) -> CGFloat {
34        guard totalCards > 1 else { return 1 }
35
36        // Line Ecuacion: x = (-y + a) / a
37        let scaleFactor: Float = Float(-index + totalCards - 1) / Float(totalCards - 1)
38        let scaleOffset = Constants.cardsScaleOffset * scaleFactor

```

A screenshot of the Xcode interface showing a code editor and a preview window. The code editor on the left contains Swift code for a `CardsStackView` that displays a stack of cards. The preview window on the right shows an iPhone Xr screen displaying three cards. The top card features a circular logo with orange and blue geometric shapes and the text "Metro Valparaíso". The middle card has a purple gradient background and contains the placeholder text "AF7A4ABD-60DC-44D1-BDF0-705C3...". The bottom card is partially visible with a cyan gradient. The preview window includes standard Xcode controls for orientation and zoom.

Custom Modifiers

```
1 |
2 import SwiftUI
3
4 struct CardStyle {
5     let scale: CGFloat
6     let yPos: CGFloat
7 }
8
9 struct CardStyleModifier: ViewModifier {
10    let style: CardStyle
11
12    func body(content: Content) -> some View {
13        content
14            .scaleEffect(style.scale)
15            .offset(y: style.yPos)
16    }
17 }
18
```

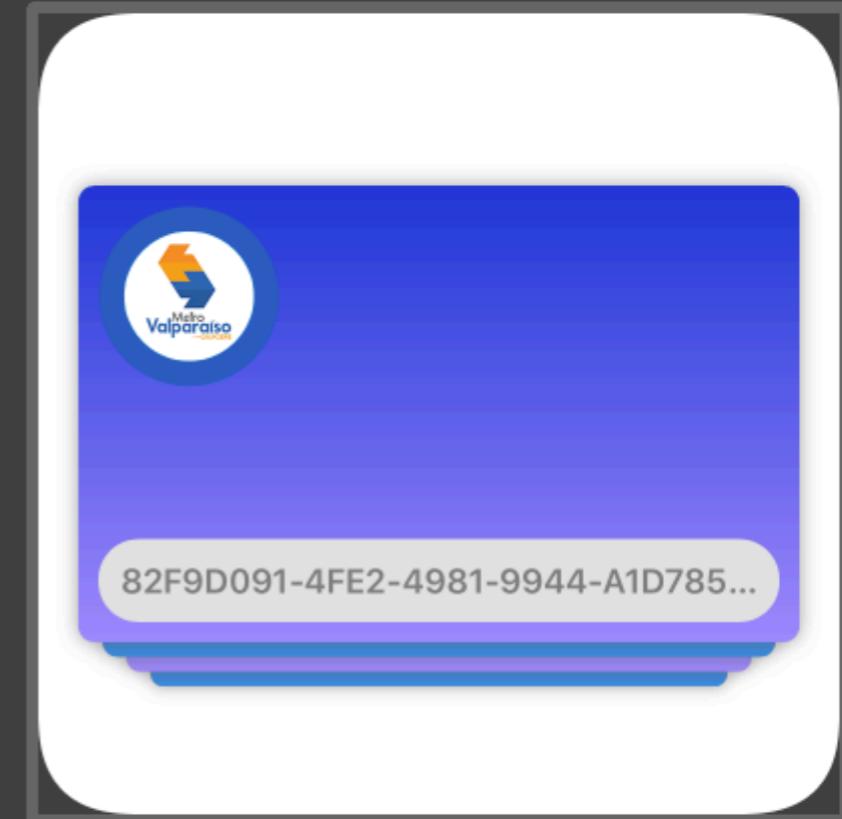
```
private func style(for card: Card) -> CardStyle {
    let totalCards = cards.count
    let index = getIndex(for: card)

    let scale = cardScale(
        forIndex: index,
        totalCards: totalCards
    )

    let yPos = yPosition(
        forIndex: index,
        totalCards: totalCards
    )

    return CardStyle(scale: scale,
                      yPos: yPos)
}
```

```
1 import SwiftUI
2
3 struct CardsStackView: View {
4
5     private struct Constants {
6         static let cardsVerticalOffset: Float = 15
7         static let cardsScaleOffset: Float = 0.2
8     }
9
10    let cards: [Card]
11
12    var body: some View {
13        ZStack {
14            ForEach(cards) { (card) in
15                CardView(card: card)
16                    .modifier(
17                        CardStyleModifier(
18                            style: self.style(for: card)
19                                )
20                        )
21                    )
22                }
23            }
24        }
25
26        private func getIndex(for card: Card) -> Int {
27            return cards.firstIndex(where: { $0.id == card.id })!
28        }
29
30        private func cardScale(forIndex index: Int, totalCards: Int) -> CGFloat {
31            guard totalCards > 1 else { return 1 }
32
33            // Line Equation: x = (-y + a) / a
34            let scaleFactor: Float = Float(-index + totalCards - 1) / Float(totalCards)
```



```
func getCardView(card: Card) -> ModifiedContent<CardView, CardStyleModifier> {
    CardView(card: card)
        .modifier(
            CardStyleModifier(
                style: self.style(
                    for: card,
                    frontCardDragOffset: self.frontCardDragOffset
                )
            )
        )
}
```

Cards | Preview Cards: Succeeded | Today at 22:15

```
1 import SwiftUI
2
3 struct CardsStackView: View {
4
5     private struct Constants {
6         static let cardsVerticalOffset: Float = 15
7         static let cardsScaleOffset: Float = 0.2
8     }
9
10    let cards: [Card]
11
12    var body: some View {
13        ZStack {
14            ForEach(cards) { (card) in
15                self.getCardView(card: card)
16            }
17        }
18    }
19
20    func getCardView(card: Card) -> ModifiedContent<CardView, CardStyleModifier> {
21        CardView(card: card)
22            .modifier(
23                CardStyleModifier(
24                    style: self.style(for: card)
25                )
26            )
27    }
28
29    private func getIndex(for card: Card) -> Int {
30        return cards.firstIndex(where: { $0.id == card.id })!
31    }
32
33    private func cardScale(forIndex index: Int, totalCards: Int) -> CGFloat {
34
```



Preview

Multiple 360x228 100%

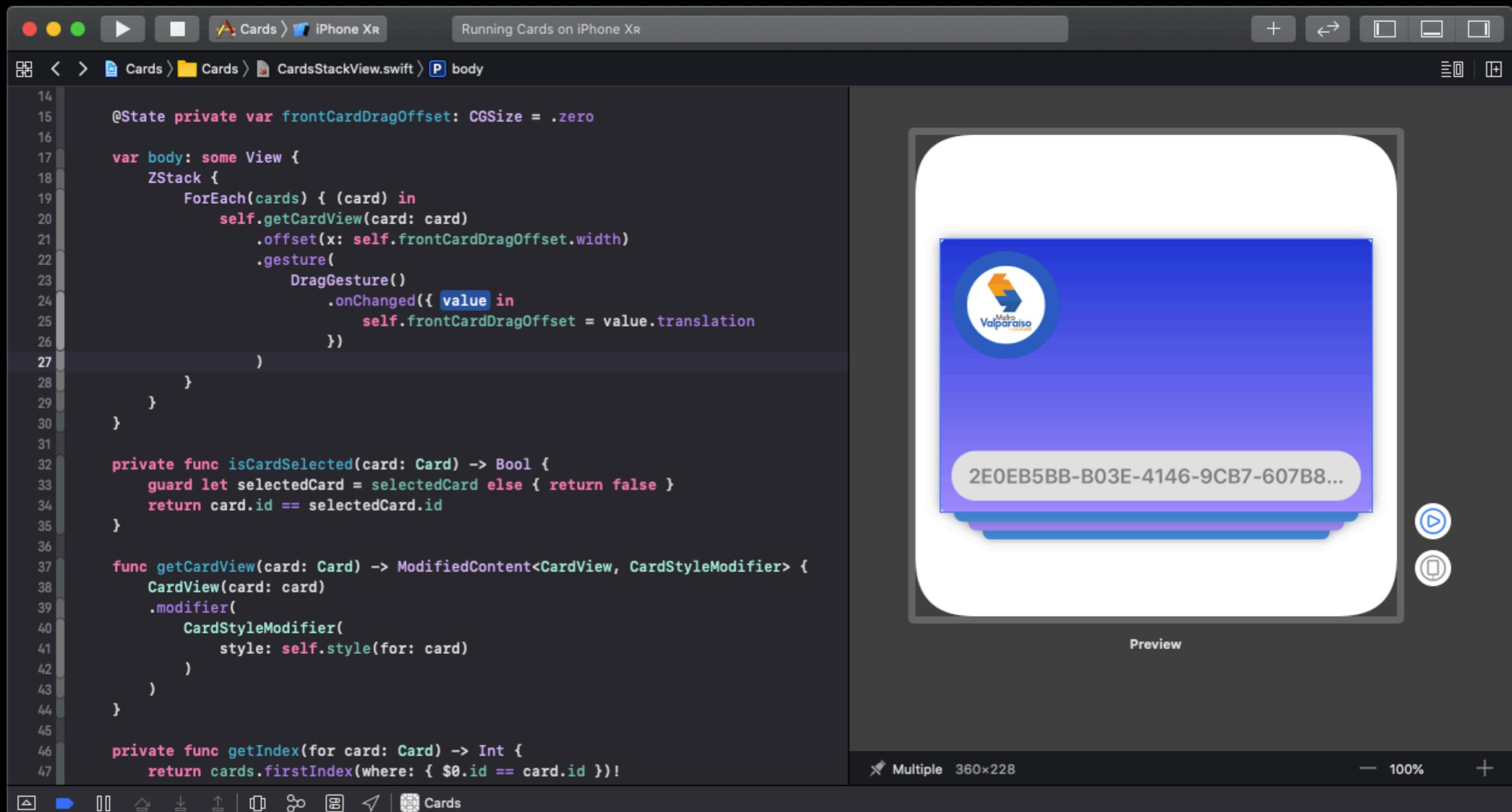
Gestures

The screenshot shows the Xcode interface with a project named "Cards" running on an iPhone XR simulator. The code editor displays `CardsStackView.swift` with the following content:

```
14 @State private var frontCardDragOffset: CGSize = .zero
15
16 var body: some View {
17     ZStack {
18         ForEach(cards) { (card) in
19             self.getCardView(card: card)
20                 .offset(x: self.frontCardDragOffset.width)
21                 .gesture(
22                     DragGesture()
23                         .onChanged({ value in
24                             self.frontCardDragOffset = value.translation
25                         })
26                 )
27         }
28     }
29 }
30
31 private func isCardSelected(card: Card) -> Bool {
32     guard let selectedCard = selectedCard else { return false }
33     return card.id == selectedCard.id
34 }
35
36 func getCardView(card: Card) -> ModifiedContent<CardView, CardStyleModifier> {
37     CardView(card: card)
38         .modifier(
39             CardStyleModifier(
40                 style: self.style(for: card)
41             )
42         )
43 }
44
45 private func getIndex(for card: Card) -> Int {
46     return cards.firstIndex(where: { $0.id == card.id })!
47 }
```

The preview window shows a stack of cards. The top card is blue and features the logo of "Metro Valparaíso" with the text "2E0EB5BB-B03E-4146-9CB7-607B8...". The bottom card is white. The Xcode toolbar at the bottom includes icons for file, project, search, and navigation.

@State



The screenshot shows the Xcode interface with a preview of a SwiftUI application running on an iPhone XR. The code in the editor is for a file named `CardsStackView.swift`, specifically the `body` part of a `View`. The code uses `@State` to manage the drag offset of the front card. It iterates over a list of cards, applying a drag gesture to each one. A preview window shows three cards. The top card has a circular logo for 'Metro Valparaiso' and a purple gradient background. The middle card is blue and contains the text '2E0EB5BB-B03E-4146-9CB7-607B8...'. The bottom card is white. There are also icons for play and device preview.

```
14 @State private var frontCardDragOffset: CGSize = .zero
15
16 var body: some View {
17     ZStack {
18         ForEach(cards) { (card) in
19             self.getCardView(card: card)
20                 .offset(x: self.frontCardDragOffset.width)
21                 .gesture(
22                     DragGesture()
23                         .onChanged({ value in
24                             self.frontCardDragOffset = value.translation
25                         })
26                 )
27         }
28     }
29 }
30
31
32 private func isCardSelected(card: Card) -> Bool {
33     guard let selectedCard = selectedCard else { return false }
34     return card.id == selectedCard.id
35 }
36
37 func getCardView(card: Card) -> ModifiedContent<CardView, CardStyleModifier> {
38     CardView(card: card)
39         .modifier(
40             CardStyleModifier(
41                 style: self.style(for: card)
42             )
43         )
44 }
45
46 private func getIndex(for card: Card) -> Int {
47     return cards.firstIndex(where: { $0.id == card.id })!
```

Running Cards on iPhone XR

```
14 @State private var frontCardDragOffset: CGSize = .zero
15
16 var body: some View {
17     ZStack {
18         ForEach(cards) { (card) in
19             self.getCardView(card: card)
20                 .offset(x: self.frontCardDragOffset.width)
21                 .gesture(
22                     DragGesture()
23                         .onChanged({ value in
24                             self.frontCardDragOffset = value.translation
25                         })
26                 )
27         }
28     }
29 }
30
31 private func isCardSelected(card: Card) -> Bool {
32     guard let selectedCard = selectedCard else { return false }
33     return card.id == selectedCard.id
34 }
35
36
37 func getCardView(card: Card) -> ModifiedContent<CardView, CardStyleModifier> {
38     CardView(card: card)
39         .modifier(
40             CardStyleModifier(
41                 style: self.style(for: card)
42             )
43         )
44 }
45
46 private func getIndex(for card: Card) -> Int {
47     return cards.firstIndex(where: { $0.id == card.id })!
```



Preview

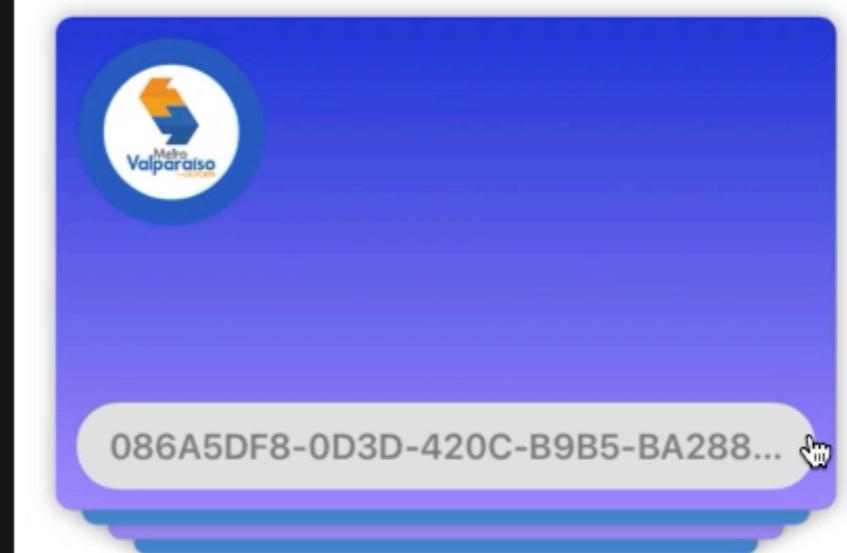
Running Cards on iPhone XR

```
14 @State private var frontCardDragOffset: CGSize = .zero
15
16 var body: some View {
17     ZStack {
18         ForEach(cards) { (card) in
19             self.getCardView(card: card)
20                 .offset(x: self.frontCardDragOffset.width)
21                 .gesture(
22                     DragGesture()
23                         .onChanged({ value in
24                             self.frontCardDragOffset = value.translation
25                         })
26                         .onEnded({ value in
27                             self.frontCardDragOffset = .zero
28                         })
29                     )
30                 )
31             }
32         }
33     }
34
35     private func isCardSelected(card: Card) -> Bool {
36         guard let selectedCard = selectedCard else { return false }
37         return card.id == selectedCard.id
38     }
39
40     func getCardView(card: Card) -> ModifiedContent<CardView, CardStyleModifier> {
41         CardView(card: card)
42             .modifier(
43                 CardStyleModifier(
44                     style: self.style(for: card)
45                 )
46             )
47     }
}
```

The screenshot shows an iPhone Xr running a SwiftUI application. The app displays a stack of cards. The top card is blue and features a circular logo for 'Metro Valparaíso' with a stylized orange and blue design. Below the logo, the card ID '8CBD404A-3BA1-4904-8F62-07345B...' is partially visible, blurred for privacy. The background of the card stack is a light purple color. On the left side of the screen, there is a code editor window titled 'CardsStackView.swift' showing the Swift code for creating the card stack view. The code uses a ZStack and a ForEach loop to render multiple cards, each with a gesture recognizer for dragging. The bottom of the screen shows the iOS navigation bar with icons for back, forward, search, and other controls.

Running Cards on iPhone XR

```
14 @State private var frontCardDragOffset: CGSize = .zero
15
16 var body: some View {
17     ZStack {
18         ForEach(cards) { (card) in
19             self.getCardView(card: card)
20                 .offset(x: self.frontCardDragOffset.width)
21                 .gesture(
22                     DragGesture()
23                         .onChanged({ value in
24                             self.frontCardDragOffset = value.translation
25                         })
26                         .onEnded({ value in
27                             self.frontCardDragOffset = .zero
28                         })
29                 )
30             }
31         }
32     }
33
34     private func isCardSelected(card: Card) -> Bool {
35         guard let selectedCard = selectedCard else { return false }
36         return card.id == selectedCard.id
37     }
38
39
40     func getCardView(card: Card) -> ModifiedContent<CardView, CardStyleModifier> {
41         CardView(card: card)
42             .modifier(
43                 CardStyleModifier(
44                     style: self.style(for: card)
45                 )
46             )
47     }
}
```



086A5DF8-0D3D-420C-B9B5-BA288...

```
private func isCardSelected(card: Card) -> Bool {  
    guard let selectedCard = selectedCard else { return false }  
    return card.id == selectedCard.id  
}
```

```
func configureTopCard(card: ModifiedContent<CardView, CardStyleModifier>) -> some View {  
    return card  
        .gesture(  
            DragGesture()  
                .onChanged({ value in  
                    self.frontCardDragOffset = value.translation  
                })  
                .onEnded({ value in  
                    self.frontCardDragOffset = .zero  
                })  
        )  
}
```

Cards > iPhone XR Running Cards on iPhone XR 4

Cards > Cards > CardsStackView.swift > CardsStackView

STRUCT CARDSSTACKVIEW: VIEW {

private struct Constants {
 static let cardsVerticalOffset: Float = 15
 static let cardsScaleOffset: Float = 0.2
}

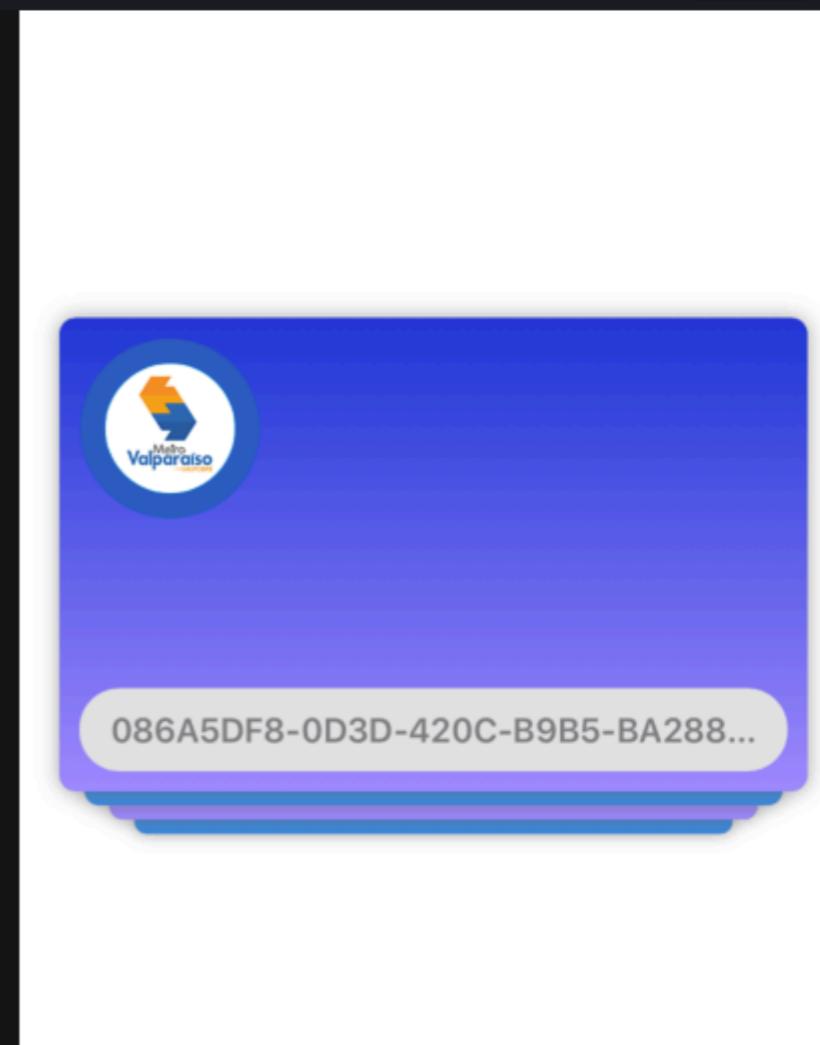
let cards: [Card]
let selectedCard: Card?
let didCardMove: () -> Void

@State private var frontCardDragOffset: CGSize = .zero

var body: some View {
 ZStack {
 ForEach(cards) { (card) in
 let cardView = self.getCardView(card: card)
 if self.isCardSelected(card: card) {
 self.configureTopCard(card: cardView)
 } else {
 cardView
 }
 }
 }
}

func configureTopCard(card: ModifiedContent<CardView, CardStyleModifier>) -> some View {
 return card
 .gesture(
 DragGesture()
 .onChanged({ value in
 self.frontCardDragOffset = value.translation
 })
 .onEnded({ value in
 self.didCardMove()
 })
)
}

! Failed to build the scheme "Cards" ⓘ Try Again Diagnostics



086A5DF8-0D3D-420C-B9B5-BA288...

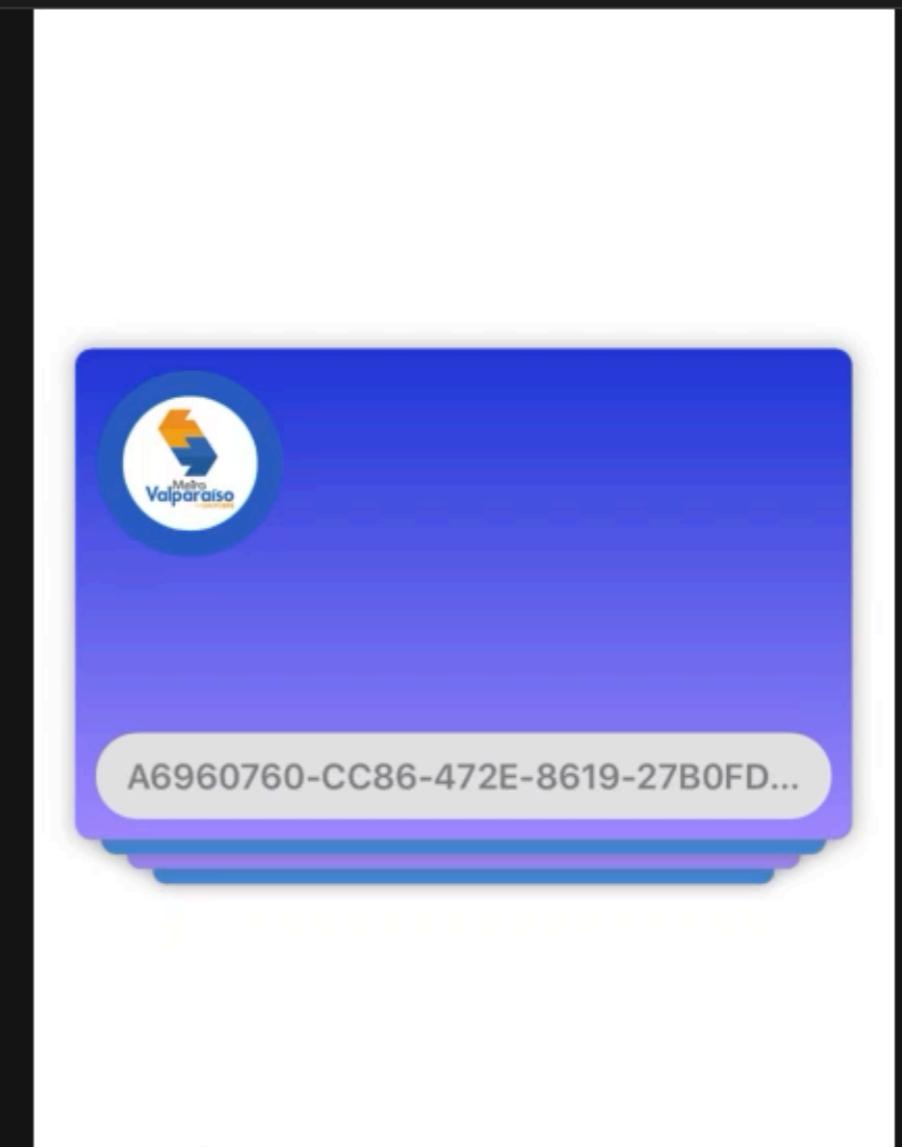
Running Cards on iPhone XR

```
10
11    let cards: [Card]
12    let selectedCard: Card?
13    let didCardMove: () -> Void
14
15    @State private var frontCardDragOffset: CGSize = .zero
16
17    var body: some View {
18        ZStack {
19            ForEach(cards) { (card) -> AnyView in
20                let cardView = self.getCardView(card: card)
21                if self.isCardSelected(card: card) {
22                    return AnyView(
23                        self.configureTopCard(card: cardView)
24                            .offset(x: self.frontCardDragOffset.width)
25                    )
26                } else {
27                    return AnyView(cardView)
28                }
29            }
30        }
31    }
32
33    func configureTopCard(card: ModifiedContent<CardView, CardStyleModifier>) -> some
34        View {
35        return card
36        .gesture(
37            DragGesture()
38                .onChanged({ value in
39                    self.frontCardDragOffset = value.translation
40                })
41                .onEnded({ value in
42                    self.frontCardDragOffset = .zero
43                })
44        )
45    }
46
47    func getCardView(card: Card) -> CardView {
48        return CardView(card: card)
49    }
50
51    func isCardSelected(card: Card) -> Bool {
52        return selectedCard == card
53    }
54
55    func selectCard(card: Card) {
56        selectedCard = card
57        didCardMove?()
58    }
59
60    func deselectCard() {
61        selectedCard = nil
62    }
63
64    func didCardMove() {
65        didCardMove?()
66    }
67
68    func getCardView(card: Card) -> CardView {
69        return CardView(card: card)
70    }
71
72    func isCardSelected(card: Card) -> Bool {
73        return selectedCard == card
74    }
75
76    func selectCard(card: Card) {
77        selectedCard = card
78        didCardMove?()
79    }
80
81    func deselectCard() {
82        selectedCard = nil
83    }
84
85    func didCardMove() {
86        didCardMove?()
87    }
88
89    func getCardView(card: Card) -> CardView {
90        return CardView(card: card)
91    }
92
93    func isCardSelected(card: Card) -> Bool {
94        return selectedCard == card
95    }
96
97    func selectCard(card: Card) {
98        selectedCard = card
99        didCardMove?()
100   }
101
102   func deselectCard() {
103       selectedCard = nil
104   }
105
106   func didCardMove() {
107       didCardMove?()
108   }
109
110   func getCardView(card: Card) -> CardView {
111       return CardView(card: card)
112   }
113
114   func isCardSelected(card: Card) -> Bool {
115       return selectedCard == card
116   }
117
118   func selectCard(card: Card) {
119       selectedCard = card
120       didCardMove?()
121   }
122
123   func deselectCard() {
124       selectedCard = nil
125   }
126
127   func didCardMove() {
128       didCardMove?()
129   }
130
131   func getCardView(card: Card) -> CardView {
132       return CardView(card: card)
133   }
134
135   func isCardSelected(card: Card) -> Bool {
136       return selectedCard == card
137   }
138
139   func selectCard(card: Card) {
140       selectedCard = card
141       didCardMove?()
142   }
143
144   func deselectCard() {
145       selectedCard = nil
146   }
147
148   func didCardMove() {
149       didCardMove?()
150   }
151
152   func getCardView(card: Card) -> CardView {
153       return CardView(card: card)
154   }
155
156   func isCardSelected(card: Card) -> Bool {
157       return selectedCard == card
158   }
159
160   func selectCard(card: Card) {
161       selectedCard = card
162       didCardMove?()
163   }
164
165   func deselectCard() {
166       selectedCard = nil
167   }
168
169   func didCardMove() {
170       didCardMove?()
171   }
172
173   func getCardView(card: Card) -> CardView {
174       return CardView(card: card)
175   }
176
177   func isCardSelected(card: Card) -> Bool {
178       return selectedCard == card
179   }
180
181   func selectCard(card: Card) {
182       selectedCard = card
183       didCardMove?()
184   }
185
186   func deselectCard() {
187       selectedCard = nil
188   }
189
190   func didCardMove() {
191       didCardMove?()
192   }
193
194   func getCardView(card: Card) -> CardView {
195       return CardView(card: card)
196   }
197
198   func isCardSelected(card: Card) -> Bool {
199       return selectedCard == card
200   }
201
202   func selectCard(card: Card) {
203       selectedCard = card
204       didCardMove?()
205   }
206
207   func deselectCard() {
208       selectedCard = nil
209   }
210
211   func didCardMove() {
212       didCardMove?()
213   }
214
215   func getCardView(card: Card) -> CardView {
216       return CardView(card: card)
217   }
218
219   func isCardSelected(card: Card) -> Bool {
220       return selectedCard == card
221   }
222
223   func selectCard(card: Card) {
224       selectedCard = card
225       didCardMove?()
226   }
227
228   func deselectCard() {
229       selectedCard = nil
230   }
231
232   func didCardMove() {
233       didCardMove?()
234   }
235
236   func getCardView(card: Card) -> CardView {
237       return CardView(card: card)
238   }
239
240   func isCardSelected(card: Card) -> Bool {
241       return selectedCard == card
242   }
243
244   func selectCard(card: Card) {
245       selectedCard = card
246       didCardMove?()
247   }
248
249   func deselectCard() {
250       selectedCard = nil
251   }
252
253   func didCardMove() {
254       didCardMove?()
255   }
256
257   func getCardView(card: Card) -> CardView {
258       return CardView(card: card)
259   }
260
261   func isCardSelected(card: Card) -> Bool {
262       return selectedCard == card
263   }
264
265   func selectCard(card: Card) {
266       selectedCard = card
267       didCardMove?()
268   }
269
270   func deselectCard() {
271       selectedCard = nil
272   }
273
274   func didCardMove() {
275       didCardMove?()
276   }
277
278   func getCardView(card: Card) -> CardView {
279       return CardView(card: card)
280   }
281
282   func isCardSelected(card: Card) -> Bool {
283       return selectedCard == card
284   }
285
286   func selectCard(card: Card) {
287       selectedCard = card
288       didCardMove?()
289   }
290
291   func deselectCard() {
292       selectedCard = nil
293   }
294
295   func didCardMove() {
296       didCardMove?()
297   }
298
299   func getCardView(card: Card) -> CardView {
300       return CardView(card: card)
301   }
302
303   func isCardSelected(card: Card) -> Bool {
304       return selectedCard == card
305   }
306
307   func selectCard(card: Card) {
308       selectedCard = card
309       didCardMove?()
310   }
311
312   func deselectCard() {
313       selectedCard = nil
314   }
315
316   func didCardMove() {
317       didCardMove?()
318   }
319
320   func getCardView(card: Card) -> CardView {
321       return CardView(card: card)
322   }
323
324   func isCardSelected(card: Card) -> Bool {
325       return selectedCard == card
326   }
327
328   func selectCard(card: Card) {
329       selectedCard = card
330       didCardMove?()
331   }
332
333   func deselectCard() {
334       selectedCard = nil
335   }
336
337   func didCardMove() {
338       didCardMove?()
339   }
340
341   func getCardView(card: Card) -> CardView {
342       return CardView(card: card)
343   }
344
345   func isCardSelected(card: Card) -> Bool {
346       return selectedCard == card
347   }
348
349   func selectCard(card: Card) {
350       selectedCard = card
351       didCardMove?()
352   }
353
354   func deselectCard() {
355       selectedCard = nil
356   }
357
358   func didCardMove() {
359       didCardMove?()
360   }
361
362   func getCardView(card: Card) -> CardView {
363       return CardView(card: card)
364   }
365
366   func isCardSelected(card: Card) -> Bool {
367       return selectedCard == card
368   }
369
370   func selectCard(card: Card) {
371       selectedCard = card
372       didCardMove?()
373   }
374
375   func deselectCard() {
376       selectedCard = nil
377   }
378
379   func didCardMove() {
380       didCardMove?()
381   }
382
383   func getCardView(card: Card) -> CardView {
384       return CardView(card: card)
385   }
386
387   func isCardSelected(card: Card) -> Bool {
388       return selectedCard == card
389   }
390
391   func selectCard(card: Card) {
392       selectedCard = card
393       didCardMove?()
394   }
395
396   func deselectCard() {
397       selectedCard = nil
398   }
399
400   func didCardMove() {
401       didCardMove?()
402   }
403
404   func getCardView(card: Card) -> CardView {
405       return CardView(card: card)
406   }
407
408   func isCardSelected(card: Card) -> Bool {
409       return selectedCard == card
410   }
411
412   func selectCard(card: Card) {
413       selectedCard = card
414       didCardMove?()
415   }
416
417   func deselectCard() {
418       selectedCard = nil
419   }
420
421   func didCardMove() {
422       didCardMove?()
423   }
424
425   func getCardView(card: Card) -> CardView {
426       return CardView(card: card)
427   }
428
429   func isCardSelected(card: Card) -> Bool {
430       return selectedCard == card
431   }
432
433   func selectCard(card: Card) {
434       selectedCard = card
435       didCardMove?()
436   }
437
438   func deselectCard() {
439       selectedCard = nil
440   }
441
442   func didCardMove() {
443       didCardMove?()
444   }
445
446   func getCardView(card: Card) -> CardView {
447       return CardView(card: card)
448   }
449
450   func isCardSelected(card: Card) -> Bool {
451       return selectedCard == card
452   }
453
454   func selectCard(card: Card) {
455       selectedCard = card
456       didCardMove?()
457   }
458
459   func deselectCard() {
460       selectedCard = nil
461   }
462
463   func didCardMove() {
464       didCardMove?()
465   }
466
467   func getCardView(card: Card) -> CardView {
468       return CardView(card: card)
469   }
470
471   func isCardSelected(card: Card) -> Bool {
472       return selectedCard == card
473   }
474
475   func selectCard(card: Card) {
476       selectedCard = card
477       didCardMove?()
478   }
479
480   func deselectCard() {
481       selectedCard = nil
482   }
483
484   func didCardMove() {
485       didCardMove?()
486   }
487
488   func getCardView(card: Card) -> CardView {
489       return CardView(card: card)
490   }
491
492   func isCardSelected(card: Card) -> Bool {
493       return selectedCard == card
494   }
495
496   func selectCard(card: Card) {
497       selectedCard = card
498       didCardMove?()
499   }
500
501   func deselectCard() {
502       selectedCard = nil
503   }
504
505   func didCardMove() {
506       didCardMove?()
507   }
508
509   func getCardView(card: Card) -> CardView {
510       return CardView(card: card)
511   }
512
513   func isCardSelected(card: Card) -> Bool {
514       return selectedCard == card
515   }
516
517   func selectCard(card: Card) {
518       selectedCard = card
519       didCardMove?()
520   }
521
522   func deselectCard() {
523       selectedCard = nil
524   }
525
526   func didCardMove() {
527       didCardMove?()
528   }
529
530   func getCardView(card: Card) -> CardView {
531       return CardView(card: card)
532   }
533
534   func isCardSelected(card: Card) -> Bool {
535       return selectedCard == card
536   }
537
538   func selectCard(card: Card) {
539       selectedCard = card
540       didCardMove?()
541   }
542
543   func deselectCard() {
544       selectedCard = nil
545   }
546
547   func didCardMove() {
548       didCardMove?()
549   }
550
551   func getCardView(card: Card) -> CardView {
552       return CardView(card: card)
553   }
554
555   func isCardSelected(card: Card) -> Bool {
556       return selectedCard == card
557   }
558
559   func selectCard(card: Card) {
560       selectedCard = card
561       didCardMove?()
562   }
563
564   func deselectCard() {
565       selectedCard = nil
566   }
567
568   func didCardMove() {
569       didCardMove?()
570   }
571
572   func getCardView(card: Card) -> CardView {
573       return CardView(card: card)
574   }
575
576   func isCardSelected(card: Card) -> Bool {
577       return selectedCard == card
578   }
579
580   func selectCard(card: Card) {
581       selectedCard = card
582       didCardMove?()
583   }
584
585   func deselectCard() {
586       selectedCard = nil
587   }
588
589   func didCardMove() {
590       didCardMove?()
591   }
592
593   func getCardView(card: Card) -> CardView {
594       return CardView(card: card)
595   }
596
597   func isCardSelected(card: Card) -> Bool {
598       return selectedCard == card
599   }
600
601   func selectCard(card: Card) {
602       selectedCard = card
603       didCardMove?()
604   }
605
606   func deselectCard() {
607       selectedCard = nil
608   }
609
610   func didCardMove() {
611       didCardMove?()
612   }
613
614   func getCardView(card: Card) -> CardView {
615       return CardView(card: card)
616   }
617
618   func isCardSelected(card: Card) -> Bool {
619       return selectedCard == card
620   }
621
622   func selectCard(card: Card) {
623       selectedCard = card
624       didCardMove?()
625   }
626
627   func deselectCard() {
628       selectedCard = nil
629   }
630
631   func didCardMove() {
632       didCardMove?()
633   }
634
635   func getCardView(card: Card) -> CardView {
636       return CardView(card: card)
637   }
638
639   func isCardSelected(card: Card) -> Bool {
640       return selectedCard == card
641   }
642
643   func selectCard(card: Card) {
644       selectedCard = card
645       didCardMove?()
646   }
647
648   func deselectCard() {
649       selectedCard = nil
650   }
651
652   func didCardMove() {
653       didCardMove?()
654   }
655
656   func getCardView(card: Card) -> CardView {
657       return CardView(card: card)
658   }
659
660   func isCardSelected(card: Card) -> Bool {
661       return selectedCard == card
662   }
663
664   func selectCard(card: Card) {
665       selectedCard = card
666       didCardMove?()
667   }
668
669   func deselectCard() {
670       selectedCard = nil
671   }
672
673   func didCardMove() {
674       didCardMove?()
675   }
676
677   func getCardView(card: Card) -> CardView {
678       return CardView(card: card)
679   }
680
681   func isCardSelected(card: Card) -> Bool {
682       return selectedCard == card
683   }
684
685   func selectCard(card: Card) {
686       selectedCard = card
687       didCardMove?()
688   }
689
690   func deselectCard() {
691       selectedCard = nil
692   }
693
694   func didCardMove() {
695       didCardMove?()
696   }
697
698   func getCardView(card: Card) -> CardView {
699       return CardView(card: card)
700   }
701
702   func isCardSelected(card: Card) -> Bool {
703       return selectedCard == card
704   }
705
706   func selectCard(card: Card) {
707       selectedCard = card
708       didCardMove?()
709   }
710
711   func deselectCard() {
712       selectedCard = nil
713   }
714
715   func didCardMove() {
716       didCardMove?()
717   }
718
719   func getCardView(card: Card) -> CardView {
720       return CardView(card: card)
721   }
722
723   func isCardSelected(card: Card) -> Bool {
724       return selectedCard == card
725   }
726
727   func selectCard(card: Card) {
728       selectedCard = card
729       didCardMove?()
730   }
731
732   func deselectCard() {
733       selectedCard = nil
734   }
735
736   func didCardMove() {
737       didCardMove?()
738   }
739
740   func getCardView(card: Card) -> CardView {
741       return CardView(card: card)
742   }
743
744   func isCardSelected(card: Card) -> Bool {
745       return selectedCard == card
746   }
747
748   func selectCard(card: Card) {
749       selectedCard = card
750       didCardMove?()
751   }
752
753   func deselectCard() {
754       selectedCard = nil
755   }
756
757   func didCardMove() {
758       didCardMove?()
759   }
760
761   func getCardView(card: Card) -> CardView {
762       return CardView(card: card)
763   }
764
765   func isCardSelected(card: Card) -> Bool {
766       return selectedCard == card
767   }
768
769   func selectCard(card: Card) {
770       selectedCard = card
771       didCardMove?()
772   }
773
774   func deselectCard() {
775       selectedCard = nil
776   }
777
778   func didCardMove() {
779       didCardMove?()
780   }
781
782   func getCardView(card: Card) -> CardView {
783       return CardView(card: card)
784   }
785
786   func isCardSelected(card: Card) -> Bool {
787       return selectedCard == card
788   }
789
790   func selectCard(card: Card) {
791       selectedCard = card
792       didCardMove?()
793   }
794
795   func deselectCard() {
796       selectedCard = nil
797   }
798
799   func didCardMove() {
800       didCardMove?()
801   }
802
803   func getCardView(card: Card) -> CardView {
804       return CardView(card: card)
805   }
806
807   func isCardSelected(card: Card) -> Bool {
808       return selectedCard == card
809   }
810
811   func selectCard(card: Card) {
812       selectedCard = card
813       didCardMove?()
814   }
815
816   func deselectCard() {
817       selectedCard = nil
818   }
819
820   func didCardMove() {
821       didCardMove?()
822   }
823
824   func getCardView(card: Card) -> CardView {
825       return CardView(card: card)
826   }
827
828   func isCardSelected(card: Card) -> Bool {
829       return selectedCard == card
830   }
831
832   func selectCard(card: Card) {
833       selectedCard = card
834       didCardMove?()
835   }
836
837   func deselectCard() {
838       selectedCard = nil
839   }
840
841   func didCardMove() {
842       didCardMove?()
843   }
844
845   func getCardView(card: Card) -> CardView {
846       return CardView(card: card)
847   }
848
849   func isCardSelected(card: Card) -> Bool {
850       return selectedCard == card
851   }
852
853   func selectCard(card: Card) {
854       selectedCard = card
855       didCardMove?()
856   }
857
858   func deselectCard() {
859       selectedCard = nil
860   }
861
862   func didCardMove() {
863       didCardMove?()
864   }
865
866   func getCardView(card: Card) -> CardView {
867       return CardView(card: card)
868   }
869
870   func isCardSelected(card: Card) -> Bool {
871       return selectedCard == card
872   }
873
874   func selectCard(card: Card) {
875       selectedCard = card
876       didCardMove?()
877   }
878
879   func deselectCard() {
880       selectedCard = nil
881   }
882
883   func didCardMove() {
884       didCardMove?()
885   }
886
887   func getCardView(card: Card) -> CardView {
888       return CardView(card: card)
889   }
890
891   func isCardSelected(card: Card) -> Bool {
892       return selectedCard == card
893   }
894
895   func selectCard(card: Card) {
896       selectedCard = card
897       didCardMove?()
898   }
899
900   func deselectCard() {
901       selectedCard = nil
902   }
903
904   func didCardMove() {
905       didCardMove?()
906   }
907
908   func getCardView(card: Card) -> CardView {
909       return CardView(card: card)
910   }
911
912   func isCardSelected(card: Card) -> Bool {
913       return selectedCard == card
914   }
915
916   func selectCard(card: Card) {
917       selectedCard = card
918       didCardMove?()
919   }
920
921   func deselectCard() {
922       selectedCard = nil
923   }
924
925   func didCardMove() {
926       didCardMove?()
927   }
928
929   func getCardView(card: Card) -> CardView {
930       return CardView(card: card)
931   }
932
933   func isCardSelected(card: Card) -> Bool {
934       return selectedCard == card
935   }
936
937   func selectCard(card: Card) {
938       selectedCard = card
939       didCardMove?()
940   }
941
942   func deselectCard() {
943       selectedCard = nil
944   }
945
946   func didCardMove() {
947       didCardMove?()
948   }
949
950   func getCardView(card: Card) -> CardView {
951       return CardView(card: card)
952   }
953
954   func isCardSelected(card: Card) -> Bool {
955       return selectedCard == card
956   }
957
958   func selectCard(card: Card) {
959       selectedCard = card
960       didCardMove?()
961   }
962
963   func deselectCard() {
964       selectedCard = nil
965   }
966
967   func didCardMove() {
968       didCardMove?()
969   }
970
971   func getCardView(card: Card) -> CardView {
972       return CardView(card: card)
973   }
974
975   func isCardSelected(card: Card) -> Bool {
976       return selectedCard == card
977   }
978
979   func selectCard(card: Card) {
980       selectedCard = card
981       didCardMove?()
982   }
983
984   func deselectCard() {
985       selectedCard = nil
986   }
987
988   func didCardMove() {
989       didCardMove?()
990   }
991
992   func getCardView(card: Card) -> CardView {
993       return CardView(card: card)
994   }
995
996   func isCardSelected(card: Card) -> Bool {
997       return selectedCard == card
998   }
999
1000  func selectCard(card: Card) {
1001      selectedCard = card
1002      didCardMove?()
1003  }
1004
1005  func deselectCard() {
1006      selectedCard = nil
1007  }
1008
1009  func didCardMove() {
1010      didCardMove?()
1011  }
```

Cards > iPhone XR Running Cards on iPhone XR

Cards > Cards > CardsStackView.swift > body

```
10
11 let cards: [Card]
12 let selectedCard: Card?
13 let didCardMove: () -> Void
14
15 @State private var frontCardDragOffset: CGSize = .zero
16
17 var body: some View {
18     ZStack {
19         ForEach(cards) { (card) -> AnyView in
20             let cardView = self.getCardView(card: card)
21             if self.isCardSelected(card: card) {
22                 return AnyView(
23                     self.configureTopCard(card: cardView)
24                     .offset(x: self.frontCardDragOffset.width)
25                 )
26             } else {
27                 return AnyView(cardView)
28             }
29         }
30     }
31 }
32
33 func configureTopCard(card: ModifiedContent<CardView, CardStyleModifier>) -> some
34     View {
35     return card
36     .gesture(
37         DragGesture()
38             .onChanged({ value in
39                 self.frontCardDragOffset = value.translation
40             })
41             .onEnded({ value in
42                 self.frontCardDragOffset = .zero
43             })
44     )
45 }
```



A6960760-CC86-472E-8619-27B0FD...

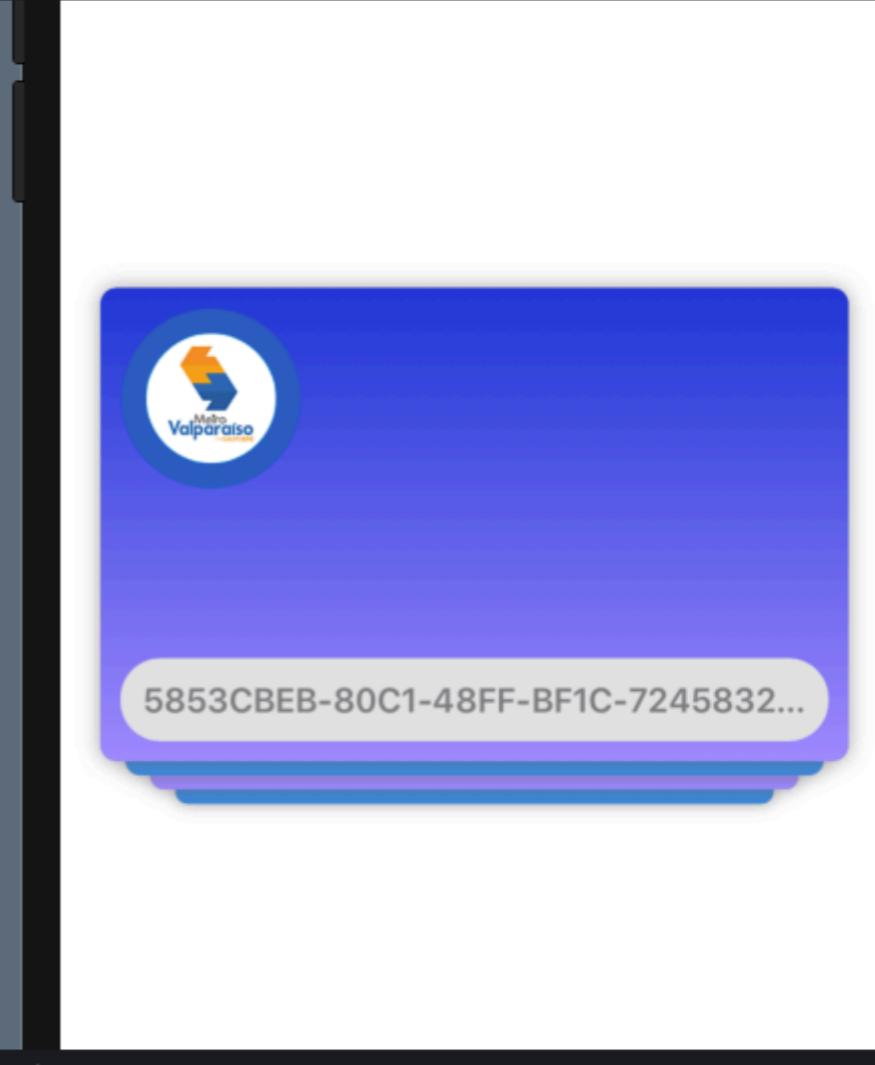
```
func configureTopCard(card: ModifiedContent<CardView, CardStyleModifier>) -> some View {
    return card
        .gesture(
            DragGesture()
                .onChanged({ value in
                    self.frontCardDragOffset = value.translation
                })
                .onEnded({ value in
                    let finalXTranslation = value.translation.width
                    let movesAtEnd = abs(finalXTranslation) >
                        abs(self.frontCardDragOffset.width)

                    self.frontCardDragOffset = .zero
                    if movesAtEnd {
                        self.didCardMove()
                    }
                })
        )
}
```

Cards | Preview Cards: Succeeded | Today at 22:40

Automatic preview updating paused ⓘ Resume

```
120
121
122
123
124
125
126     let alpha = card.isPinned ? 1 : 1 - animationPercentage
127
128     return CardStyle(scale: scale,
129                        yPos: yPos,
130                        dragOffset: dragOffset,
131                        alpha: alpha)
132
133 }
134
135 #if DEBUG
136 struct CardsStackView_Previews : PreviewProvider {
137     static var previews: some View {
138         Group {
139             CardsStackView(cards: Card.previewContent,
140                            selectedCard: Card.previewContent.last,
141                            didCardMove: {})
142                 .scaledToFit()
143
144             .padding(20)
145             .previewLayout(.fixed(width: 400, height: 400))
146         }
147     }
148 #endif
149 }
```



The screenshot shows an iPhone Xr simulator displaying a UI component. A blue rectangular card is visible, featuring a circular logo with orange and blue geometric shapes and the text "Metro Valparaíso". Below the card is a purple rounded rectangle containing the text "5853CBEB-80C1-48FF-BF1C-7245832...". The top status bar indicates "Cards | Preview Cards: Succeeded | Today at 22:40". The Xcode interface includes a navigation bar with "Cards > iPhone Xr", a code editor with Swift code, and a preview pane showing the rendered UI.

```
private func cardDragOffset(isFrontCard: Bool, frontCardDragOffset: CGSize) ->
    CGFloat {
    guard isFrontCard else { return .zero }
    return frontCardDragOffset.width
}
```

```
1 import SwiftUI
2
3
4 struct CardStyle {
5     let scale: CGFloat
6     let yPos: CGFloat
7     let dragOffset: CGFloat
8     let alpha: Double
9 }
10
11 struct CardStyleModifier: ViewModifier {
12
13     private struct Constants {
14         static let animation: Animation = .linear(duration: 0.2)
15     }
16
17     let style: CardStyle
18
19     func body(content: Content) -> some View {
20         content
21             .scaleEffect(style.scale)
22             .offset(y: style.yPos)
23             .offset(x: style.dragOffset)
24             .opacity(style.alpha)
25             .animation(Constants.animation)
26     }
27 }
28
```

One
Eternity
Later

Cards | Preview Cards: Succeeded | Today at 22:40

Automatic preview updating paused ⓘ

Resume

```
totalCards: totalCards,
isFrontCard: isFrontCard,
animationPercentage: animationPercentage
)

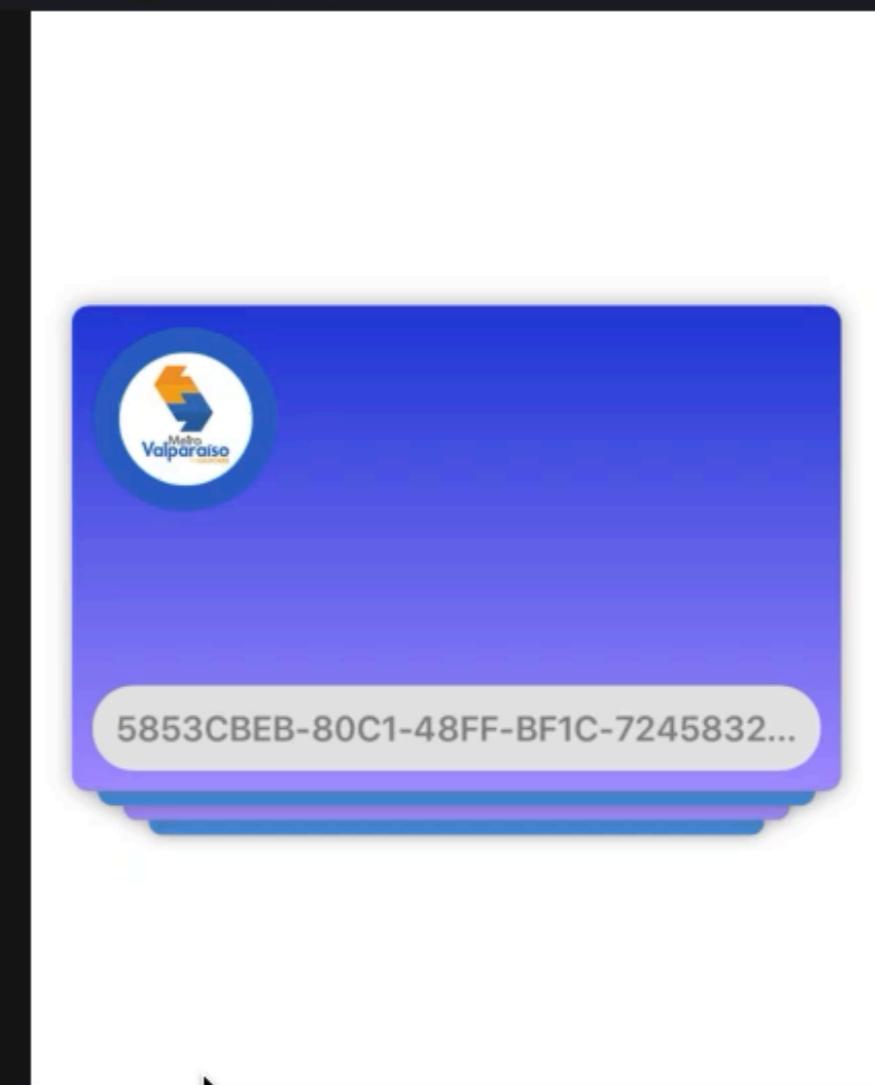
let yPos = yPosition(
    forIndex: index,
    totalCards: totalCards,
    isFrontCard: isFrontCard,
    animationPercentage: animationPercentage
)

let dragOffset = cardDragOffset(isFrontCard: isFrontCard,
                                frontCardDragOffset: frontCardDragOffset)

let alpha = cardAlpha(isFrontCard: isFrontCard,
                      animationPercentage: animationPercentage)

return CardStyle(scale: scale,
                 yPos: yPos,
                 dragOffset: dragOffset,
                 alpha: alpha)
}

#if DEBUG
struct CardsStackView_Previews : PreviewProvider {
    static var previews: some View {
        Group {
            CardsStackView(cards: Card.previewContent,
                           selectedCard: Card.previewContent.last,
                           didCardMove: {})
                .scaledToFit()
        }
    }
}
```

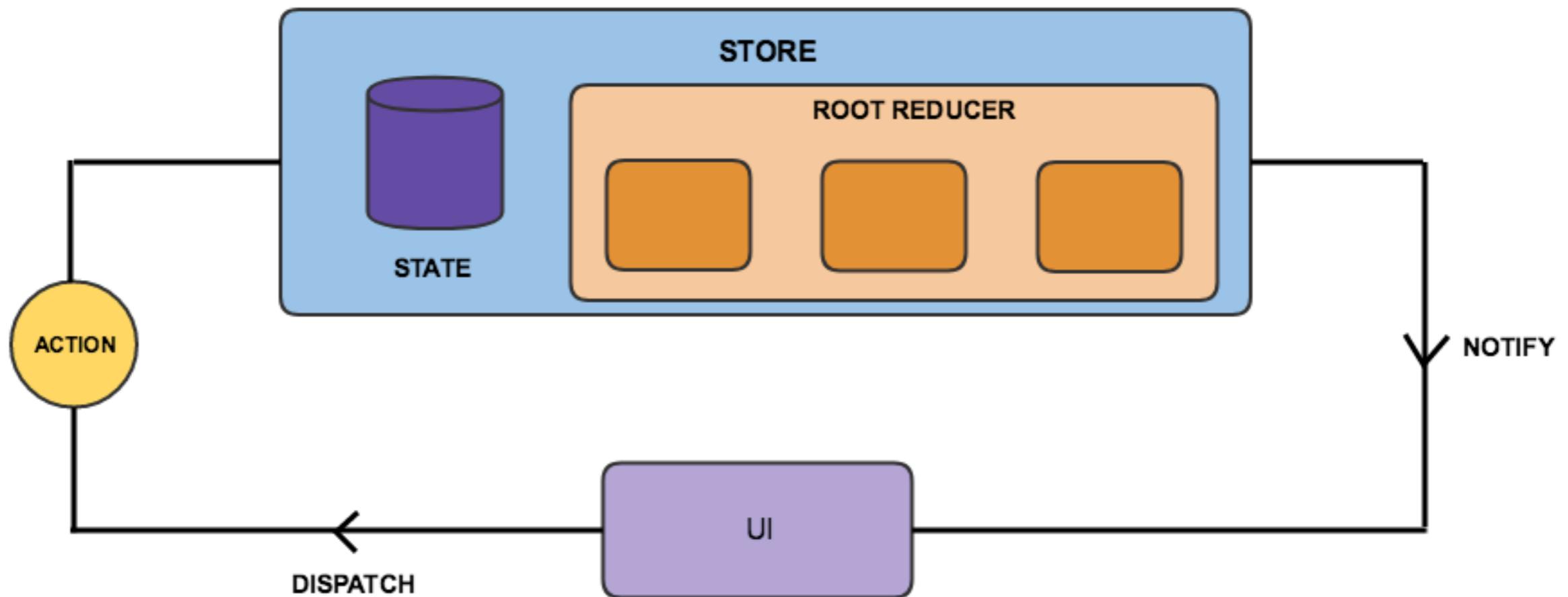


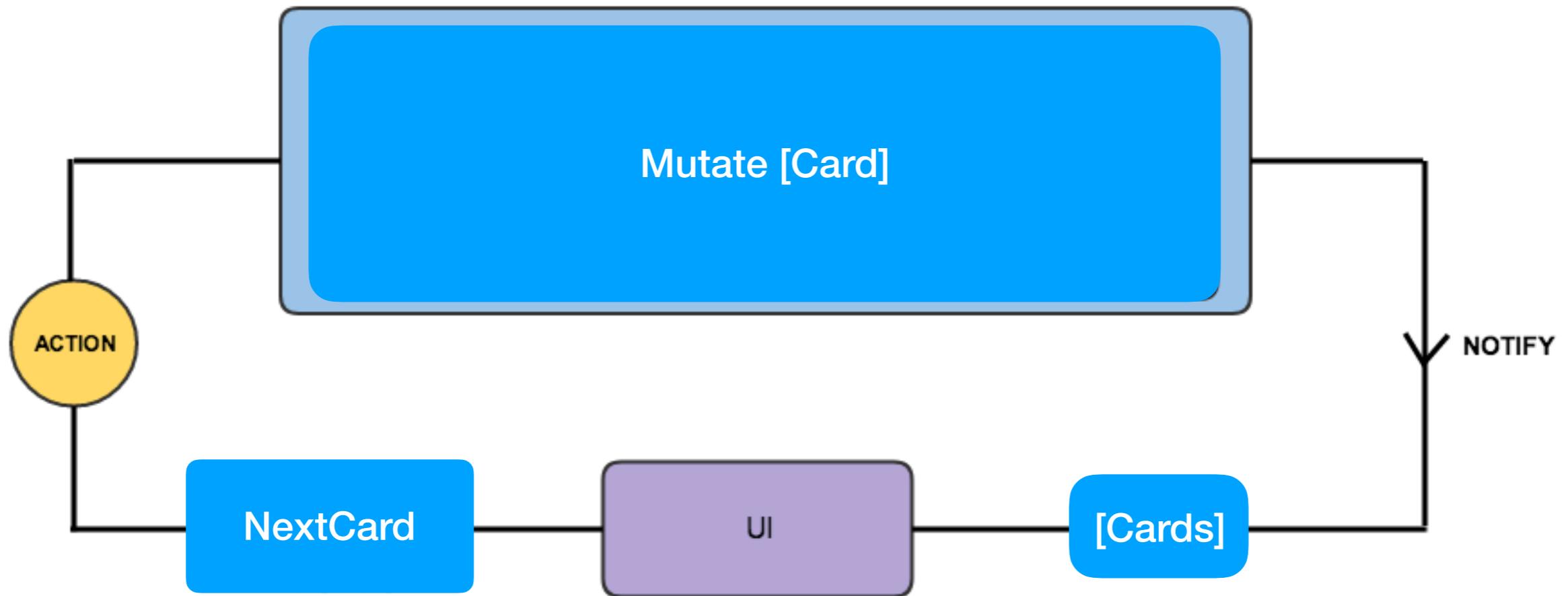
5853CBEB-80C1-48FF-BF1C-7245832...



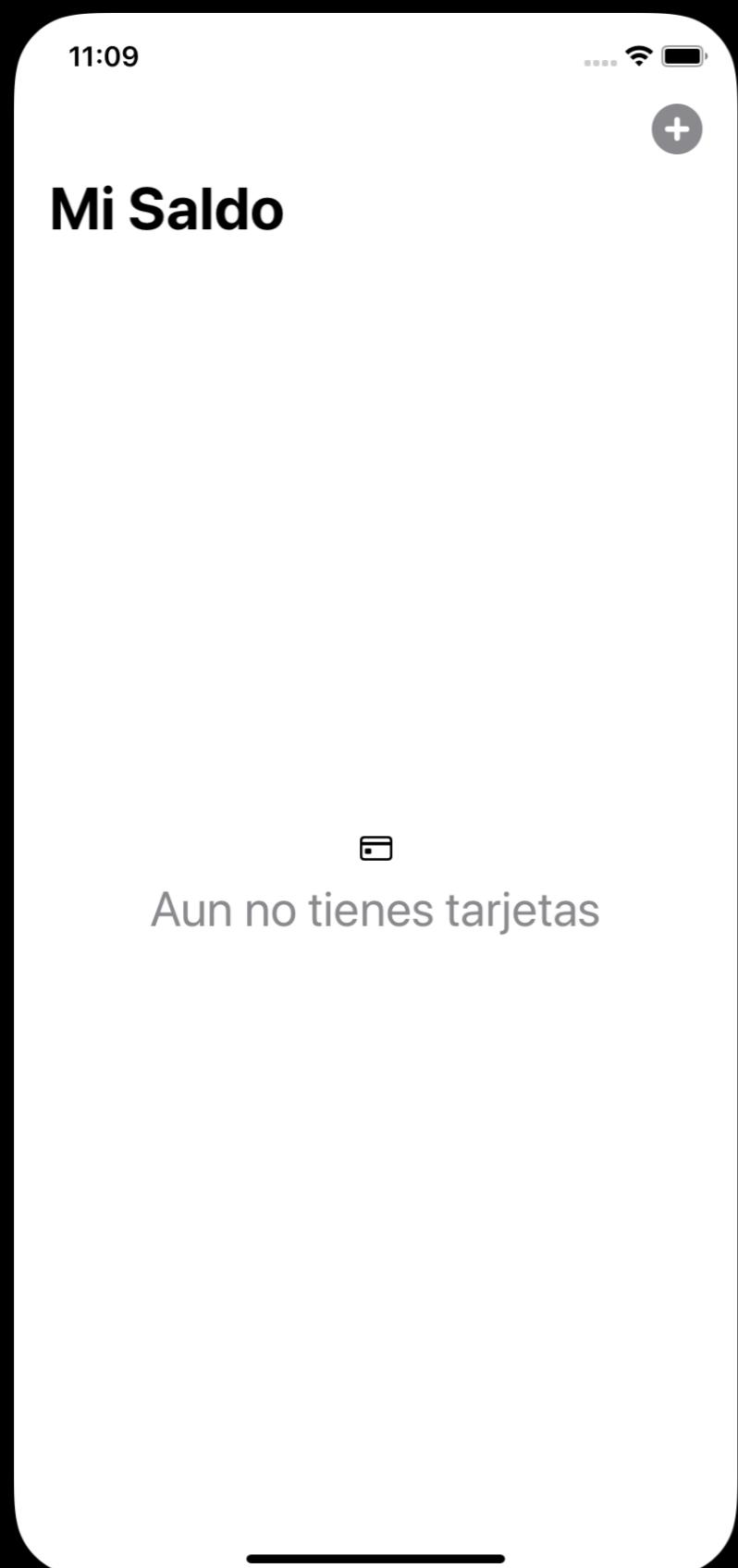


Redux

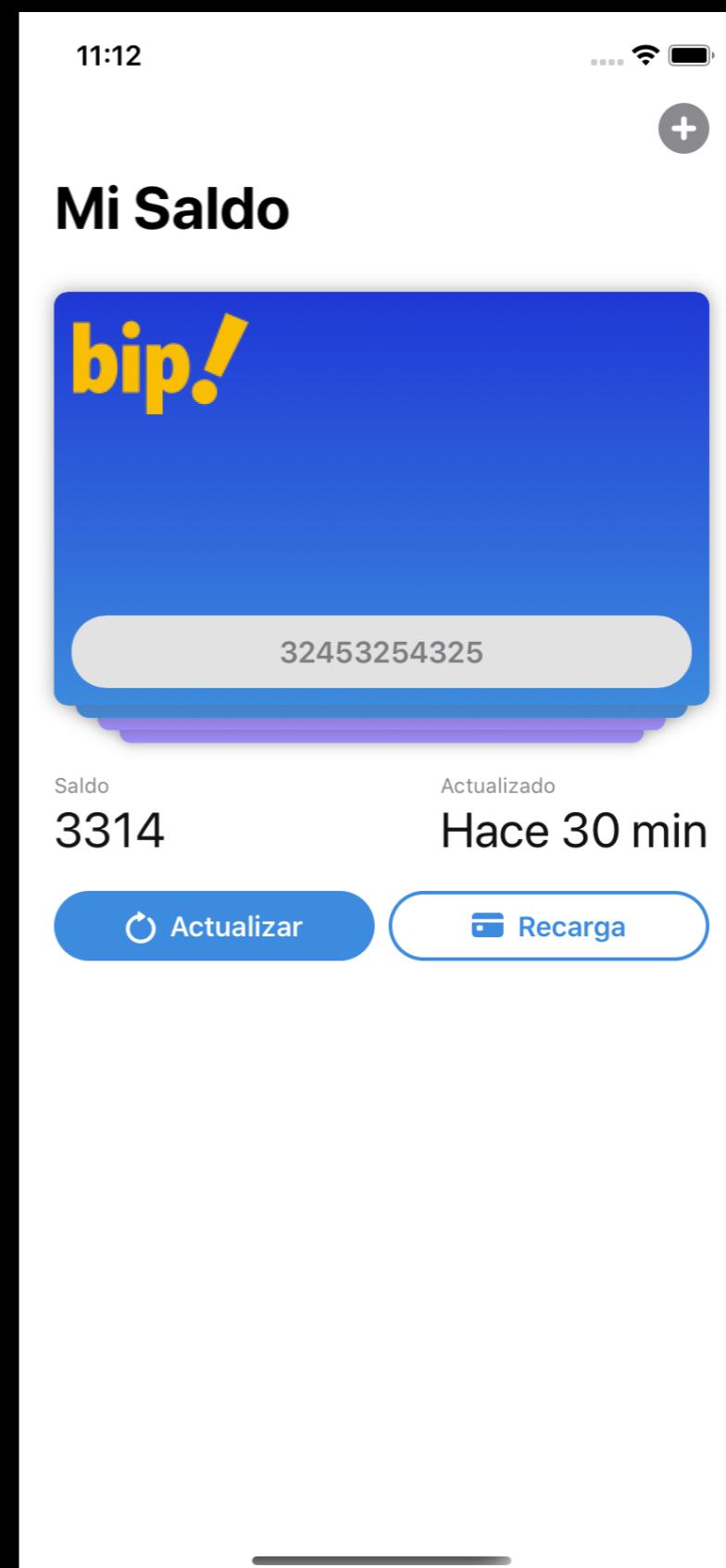




Empty



Display



```
import SwiftUI
import Combine

class HomeStore {
    var state: HomeState
    lazy var actions: HomeActions = HomeActions(delegate: self)

    fileprivate var cards: [Card] = []
    fileprivate var selectedCard: Card?

    init() {
        self.state = HomeState(initialState: .empty)
    }
}
```

ObservableObject

```
import Combine

class HomeState: ObservableObject {
    @Published var viewState: ViewStateCase

    enum ViewStateCase {
        case displaying(cards: [Card], selectedCard: Card?)
        case empty
    }

    init(initialState: HomeState.ViewStateCase) {
        self.viewState = initialState
    }
}
```

@Published

@ObservedObject

```
struct HomeView: View {

    let actions: HomeActions
    @State private var isAddCardPresented = false
    @ObservedObject var state: HomeState

    func currentView(for state: HomeState.ViewStateCase) -> AnyView {
        switch state {
        case .displaying(let cards, let selectedCard):
            return AnyView(
                CardsStackView(
                    cards: cards,
                    selectedCard: selectedCard,
                    didCardMove: {
                        self.actions.moveNextCardToFront()
                    }
                )
            )
        case .empty:
            return AnyView(
                CardsEmptyView()
            )
        }
    }

    var body: some View {
        currentView(for: self.$state.viewState.value)
        .onAppear {
            self.actions.loadCards()
        }
    }
}
```

```
import Foundation

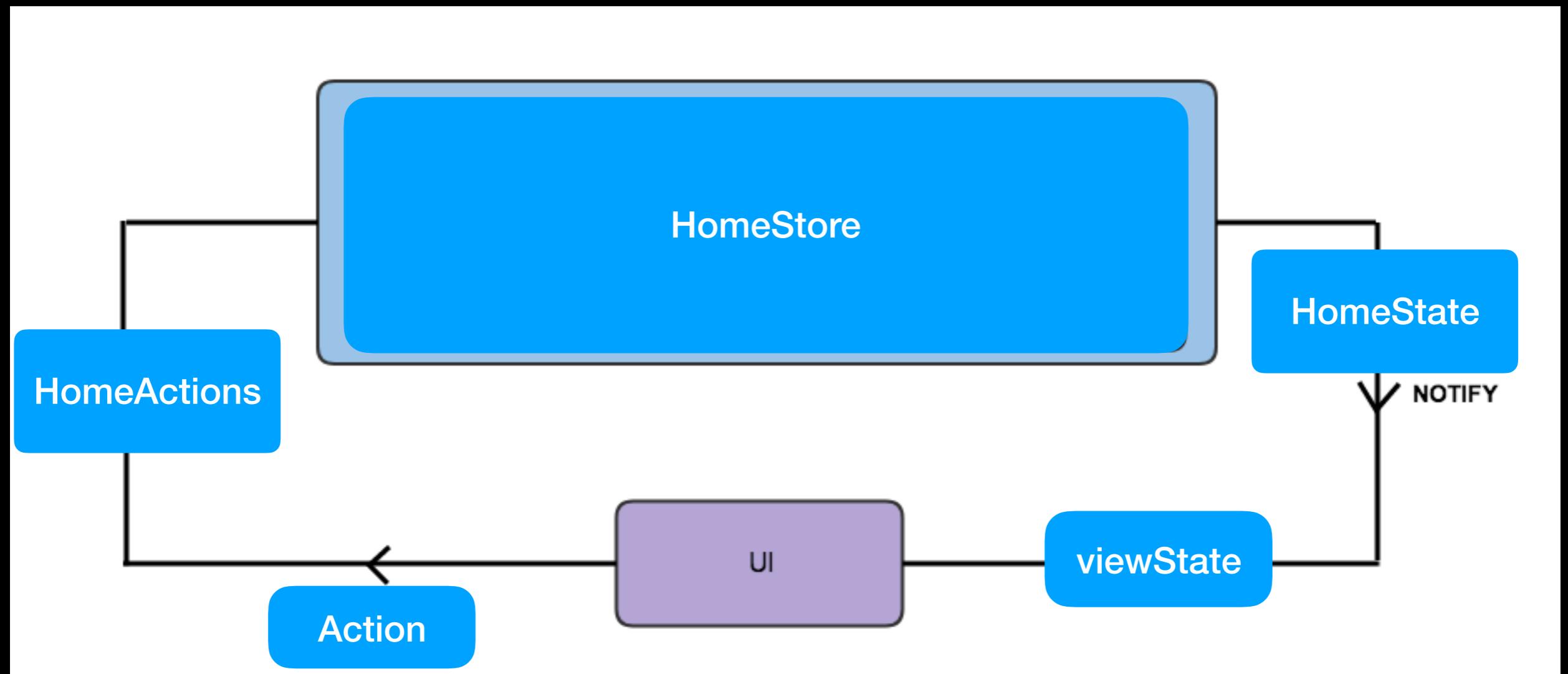
protocol HomeActionsDelegate: class {
    func loadCards()
    func moveNextCardToFront()
    func addCard(withId id: String, vendor: CardVendor)
}

class HomeActions: HomeActionsDelegate {
    weak var delegate: HomeActionsDelegate?

    init(delegate: HomeActionsDelegate? = nil) {
        self.delegate = delegate
    }

    func loadCards() {
        delegate?.loadCards()
    }
    func moveNextCardToFront() {
        delegate?.moveNextCardToFront()
    }

    func addCard(withId id: String, vendor: CardVendor) {
        delegate?.addCard(withId: id, vendor: vendor)
    }
}
```



```
import UIKit
import SwiftUI

class SceneDelegate: UIResponder, UIWindowSceneDelegate {

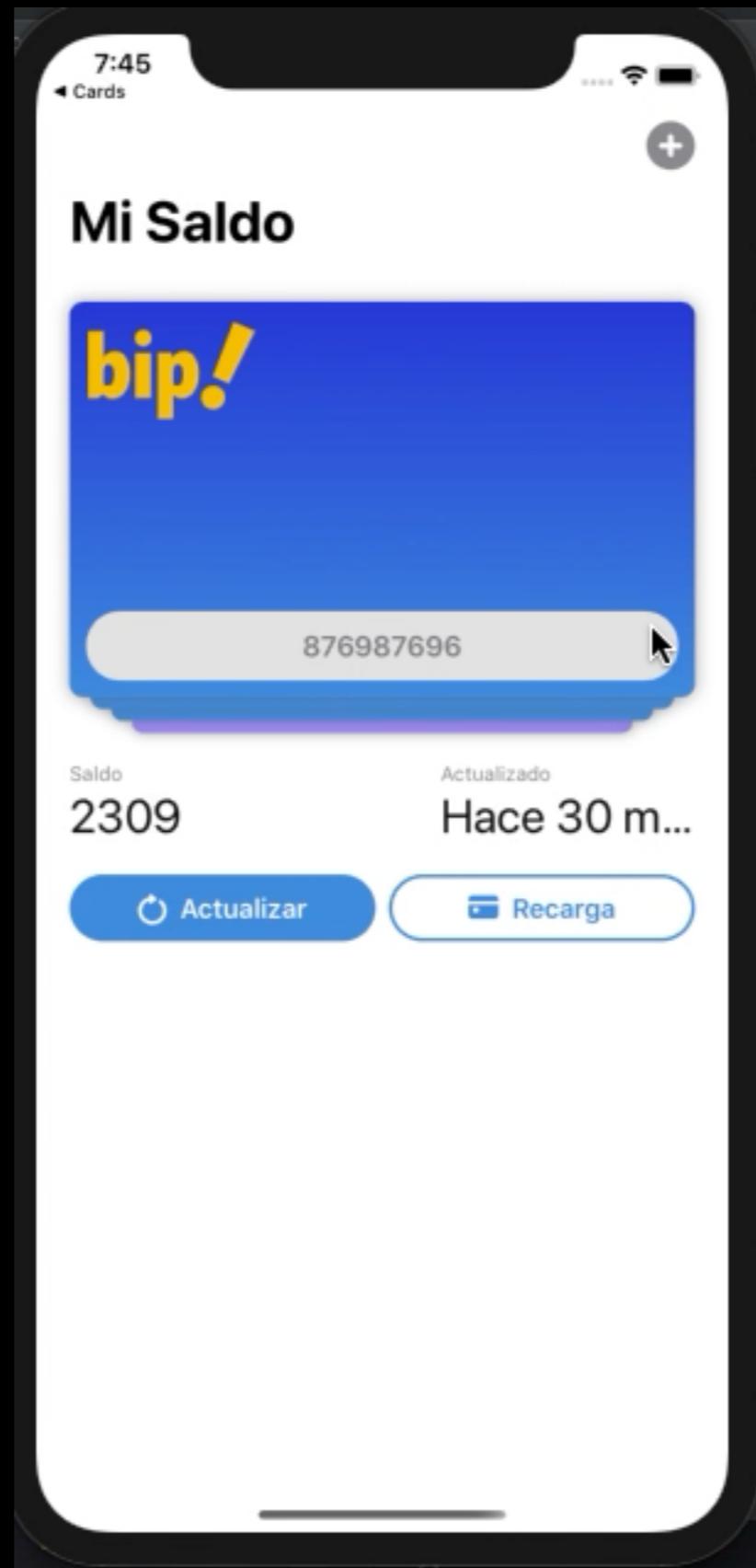
    var window: UIWindow?
    lazy var store = HomeStore()

    func scene(_ scene: UIScene, willConnectTo session: UISceneSession, options connectionOptions: UIScene.ConnectionOptions) {
        guard let windowScene = scene as? UIWindowScene else { return }

        let window = UIWindow(windowScene: windowScene)
        window.rootViewController = UIHostingController(
            rootView: HomeView(actions: store.actions, state: store.state)
        )
        self.window = window
        window.makeKeyAndVisible()
    }
}
```

```
#if DEBUG
struct HomeView_Previews : PreviewProvider {
    static var previews: some View {
        Group {
            HomeView(
                actions: HomeActions(),
                state: HomeState(initialState: .empty)
            )

            HomeView(
                actions: HomeActions(),
                state: HomeState(initialState: .displaying(
                    cards: Card.previewContent,
                    selectedCard: Card.previewContent[0]
                ))
            )
        }
    }
}
#endif
```



Donde continuar

- Proyecto: <https://github.com/migueloruiz/MiSaldo-App>
- Twitter: Majid Jabrayilov @mecid
- Twitter: Thomas Ricouard @Dimillian
- Twitter: Sarun W. @sarunw
- MovieSwiftUI App: <https://github.com/Dimillian/MovieSwiftUI>
- SwiftUI By Examples: <https://www.hackingwithswift.com/quick-start/swiftui>
- About SwiftUI: <https://github.com/Juanpe/About-SwiftUI>