Swift UI 100 days challenge

Week 3 I Day 10 - 15 I Understanding HStack, VStack, Button and making a view with it. Adding that view to the list.

Understanding HStack & VStack

Generally,
1. HStack stands for Horizontal Stack View
2 .VStack stands for Vertical Stack View

HStack and VStack are used to club the views inside the body,
So that we can get a horizontal or vertical aligned view structure

Syntax:

```
HStack {
View1()
View 2()
}
```

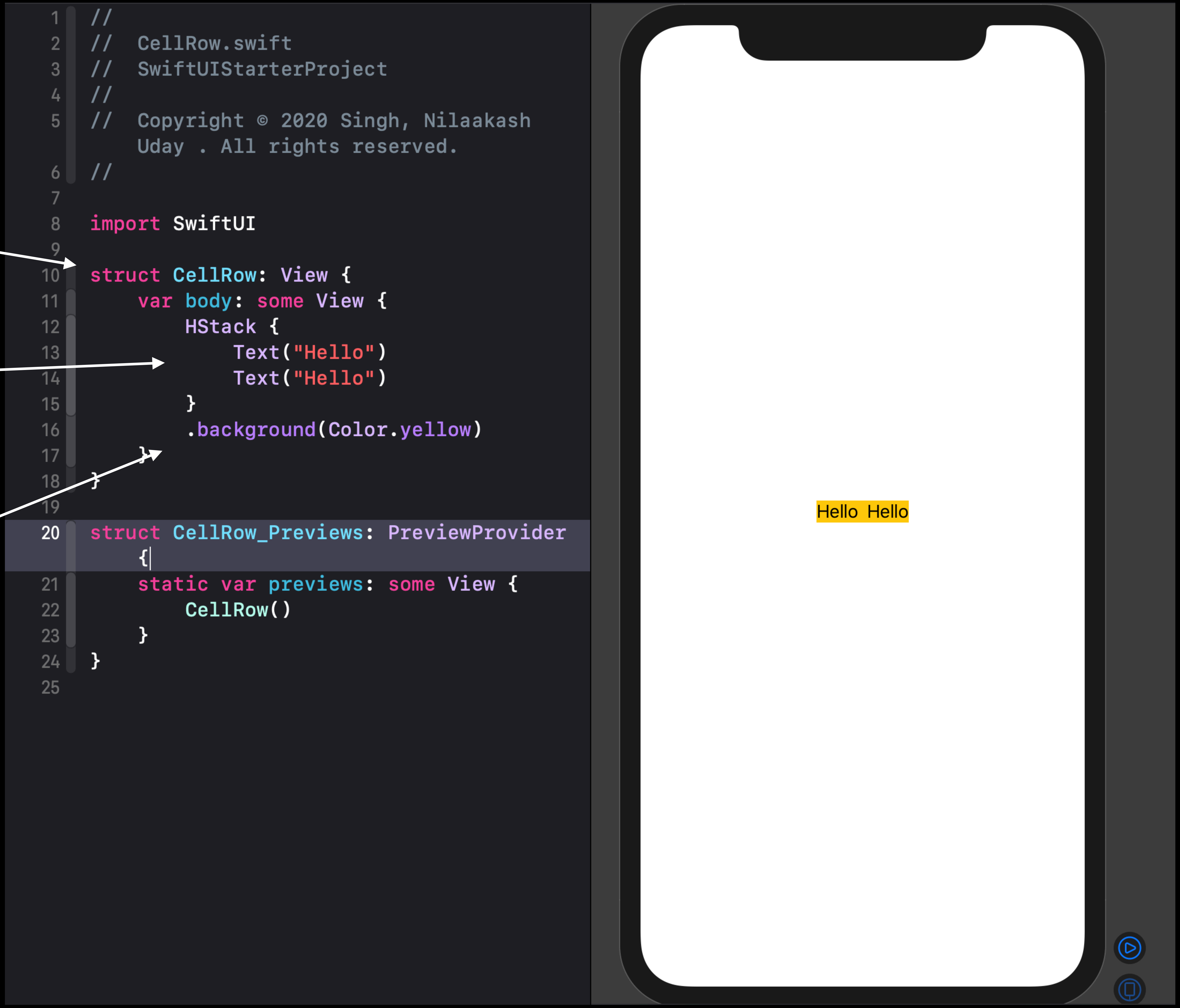
```
VStack {
View1()
View 2()
}
```

Lets get started….

Step 1: Create new SWIFTUI file called “Cell Row”

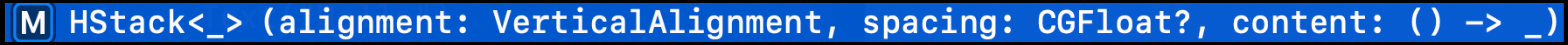
Step 2: Add a HStack with 2 Text in it.

Step 3: Add Background Color to understand the Stack Scope



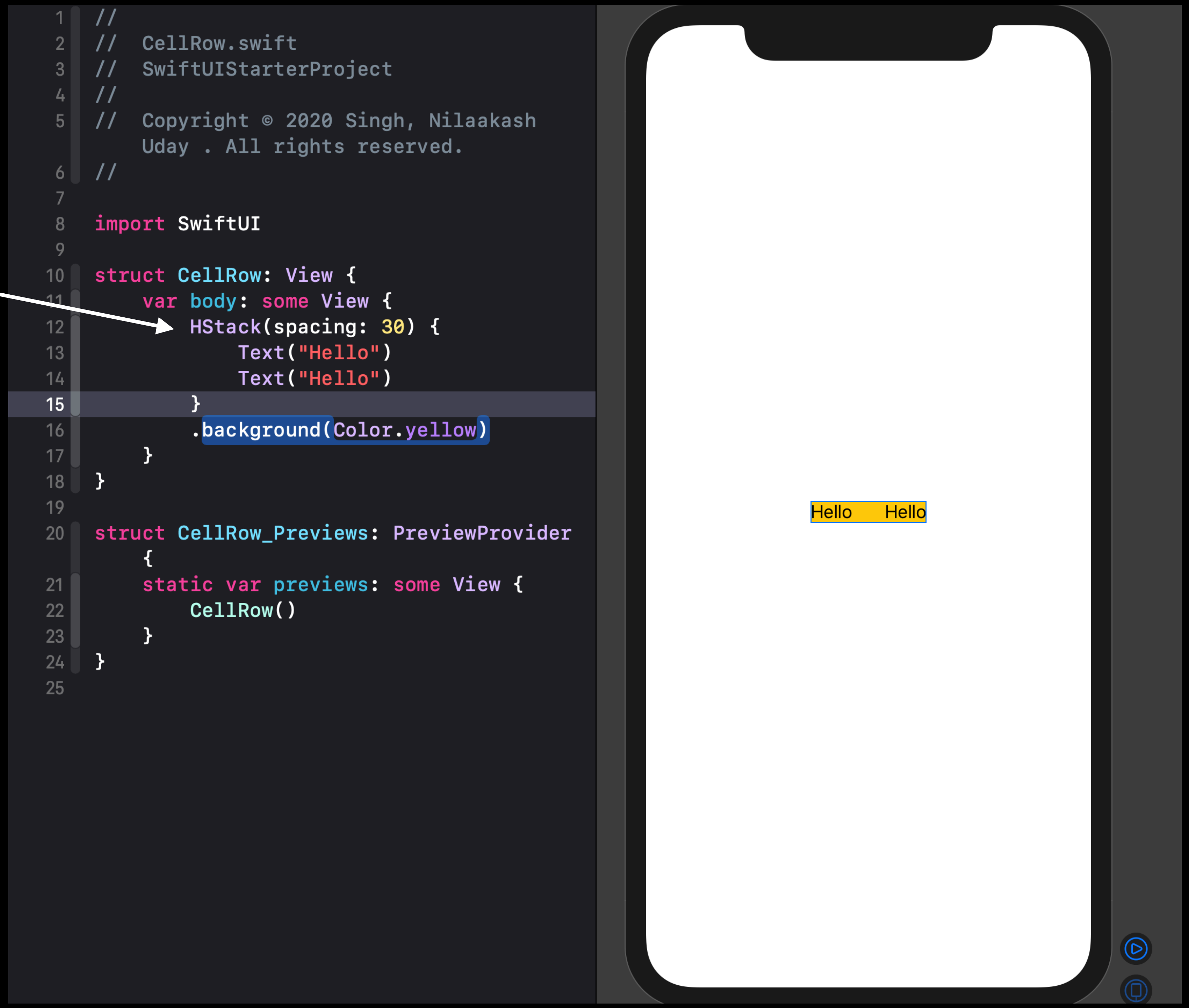
```
//
//  CellRow.swift
//  SwiftUIStarterProject
//
//  Copyright © 2020 Singh, Nilaakash Uday . All rights reserved.
//

import SwiftUI

struct CellRow: View {
    var body: some View {
        HStack {
            Text("Hello")
            Text("Hello")
        }
        .background(Color.yellow)
    }
}

struct CellRow_Previews: PreviewProvider {
    static var previews: some View {
        CellRow()
    }
}
```

As you see HStack and VStack have few properties, which we can set directly while initialising it



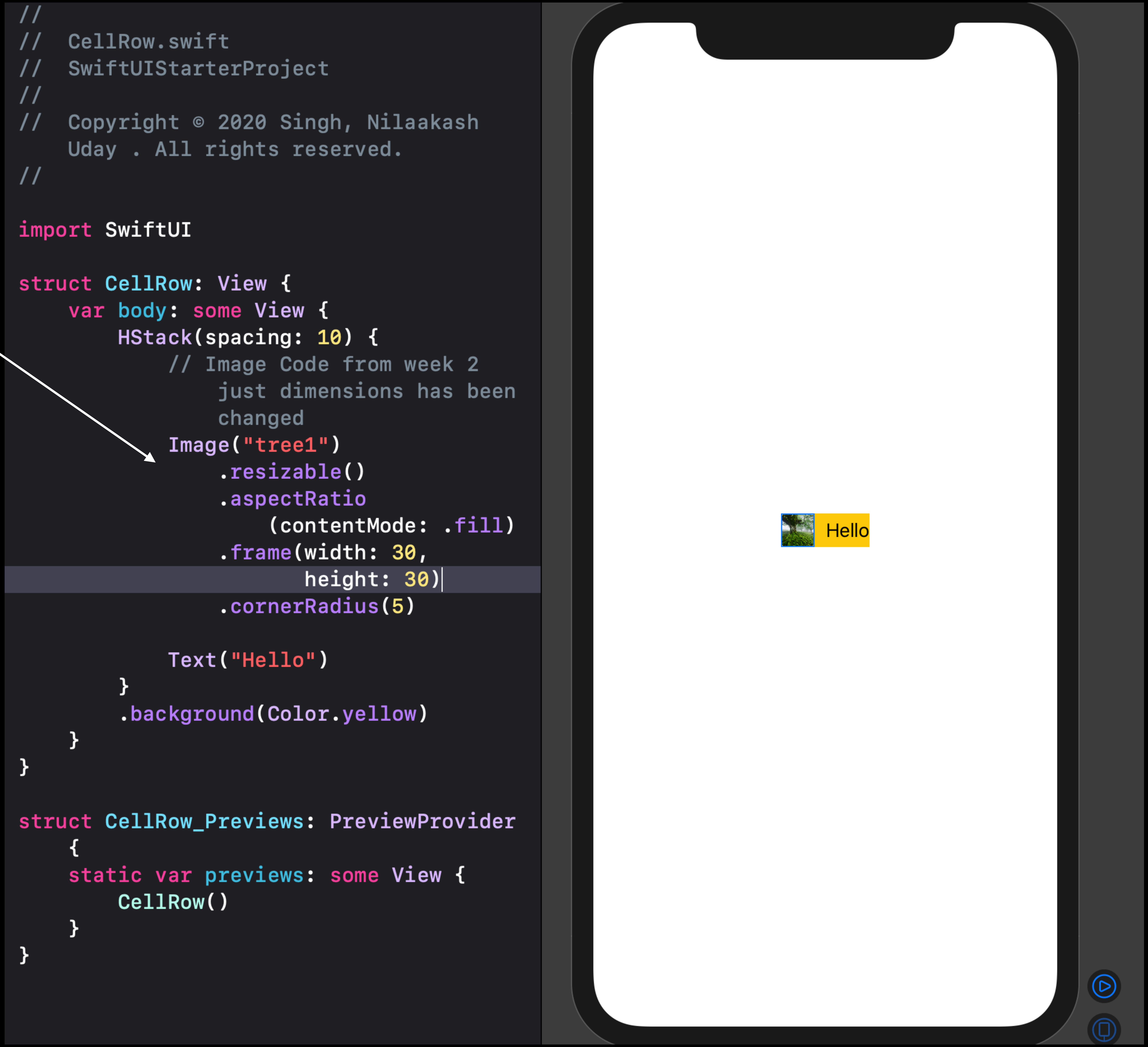
```
M HStack<_> (alignment: VerticalAlignment, spacing: CGFloat?, content: () -> _)
```

Step 4: For now lets add spacing between two Text



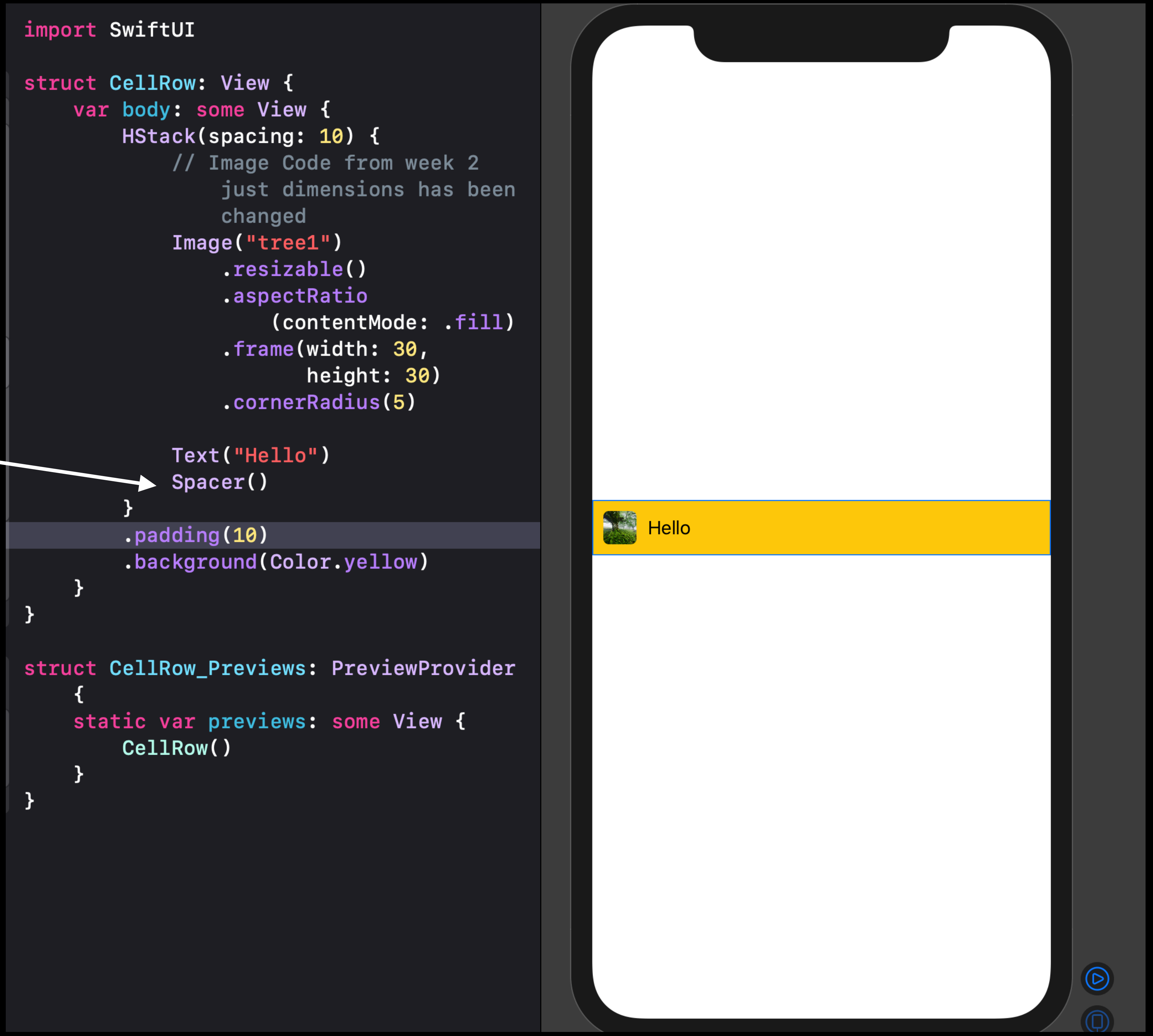
```
//
//  CellRow.swift
//  SwiftUIStarterProject
//
//  Copyright © 2020 Singh, Nilaakash
    Uday . All rights reserved.
//

import SwiftUI

struct CellRow: View {
    var body: some View {
        HStack(spacing: 30) {
            Text("Hello")
            Text("Hello")
        }
        .background(Color.yellow)
    }
}

struct CellRow_Previews: PreviewProvider
    {
    static var previews: some View {
        CellRow()
    }
}
```

Step5: Lets add image from Week 2 challenge instead of text, We have only changed the dimesions



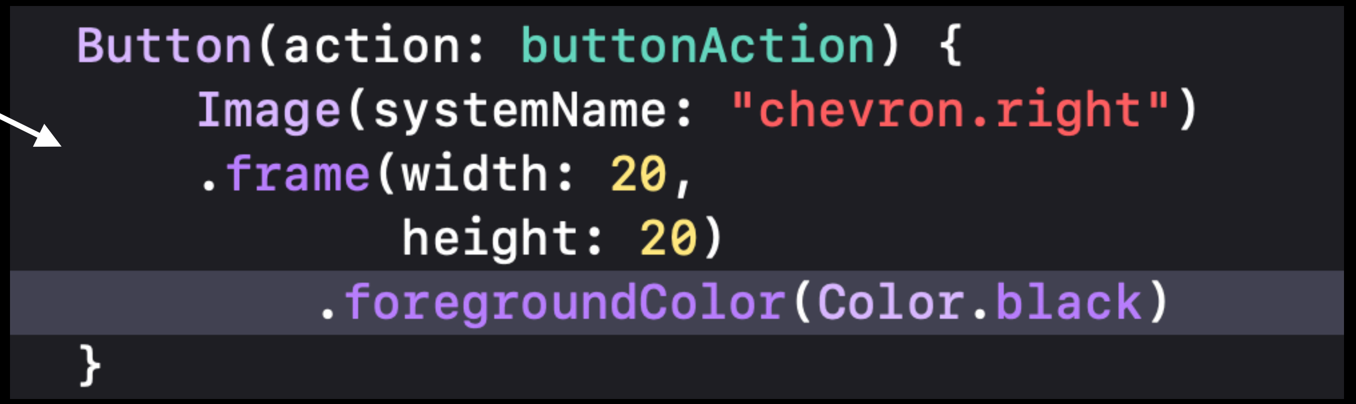
```
//
//  CellRow.swift
//  SwiftUIStarterProject
//
//  Copyright © 2020 Singh, Nilaakash
    Uday . All rights reserved.
//

import SwiftUI

struct CellRow: View {
    var body: some View {
        HStack(spacing: 10) {
            // Image Code from week 2
                just dimensions has been
                changed
            Image("tree1")
                .resizable()
                .aspectRatio
                    (contentMode: .fill)
                .frame(width: 30,
                           height: 30)
                .cornerRadius(5)

            Text("Hello")
        }
        .background(Color.yellow)
    }
}

struct CellRow_Previews: PreviewProvider
    {
    static var previews: some View {
        CellRow()
    }
}
```

Step 6: Now we are adding a Spacer() to take extra space and making our view left aligned

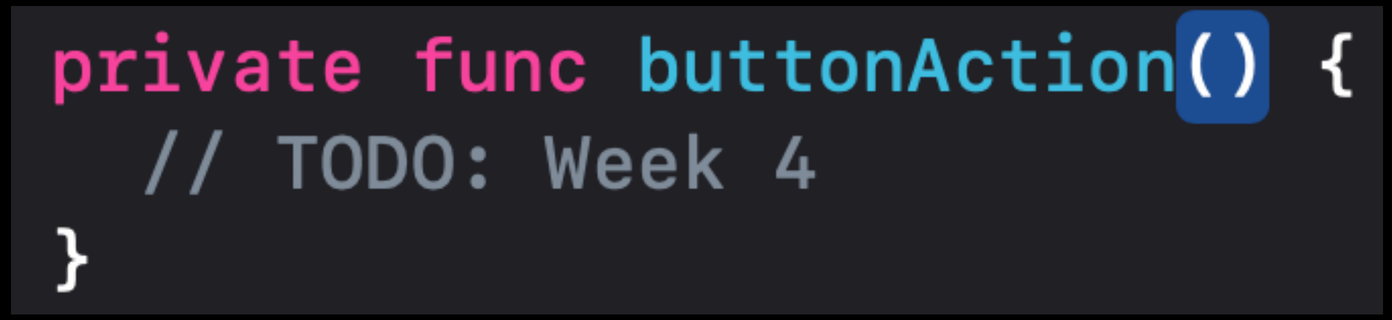


```
import SwiftUI

struct CellRow: View {
    var body: some View {
        HStack(spacing: 10) {
            // Image Code from week 2
               just dimensions has been
               changed
            Image("tree1")
                .resizable()
                .aspectRatio
                    (contentMode: .fill)
                .frame(width: 30,
                       height: 30)
                .cornerRadius(5)

            Text("Hello")
            Spacer()
        }
        .padding(10)
        .background(Color.yellow)
    }
}

struct CellRow_Previews: PreviewProvider
    {
    static var previews: some View {
        CellRow()
    }
}
```

Step 7: Lets add a button after spacer and action for button

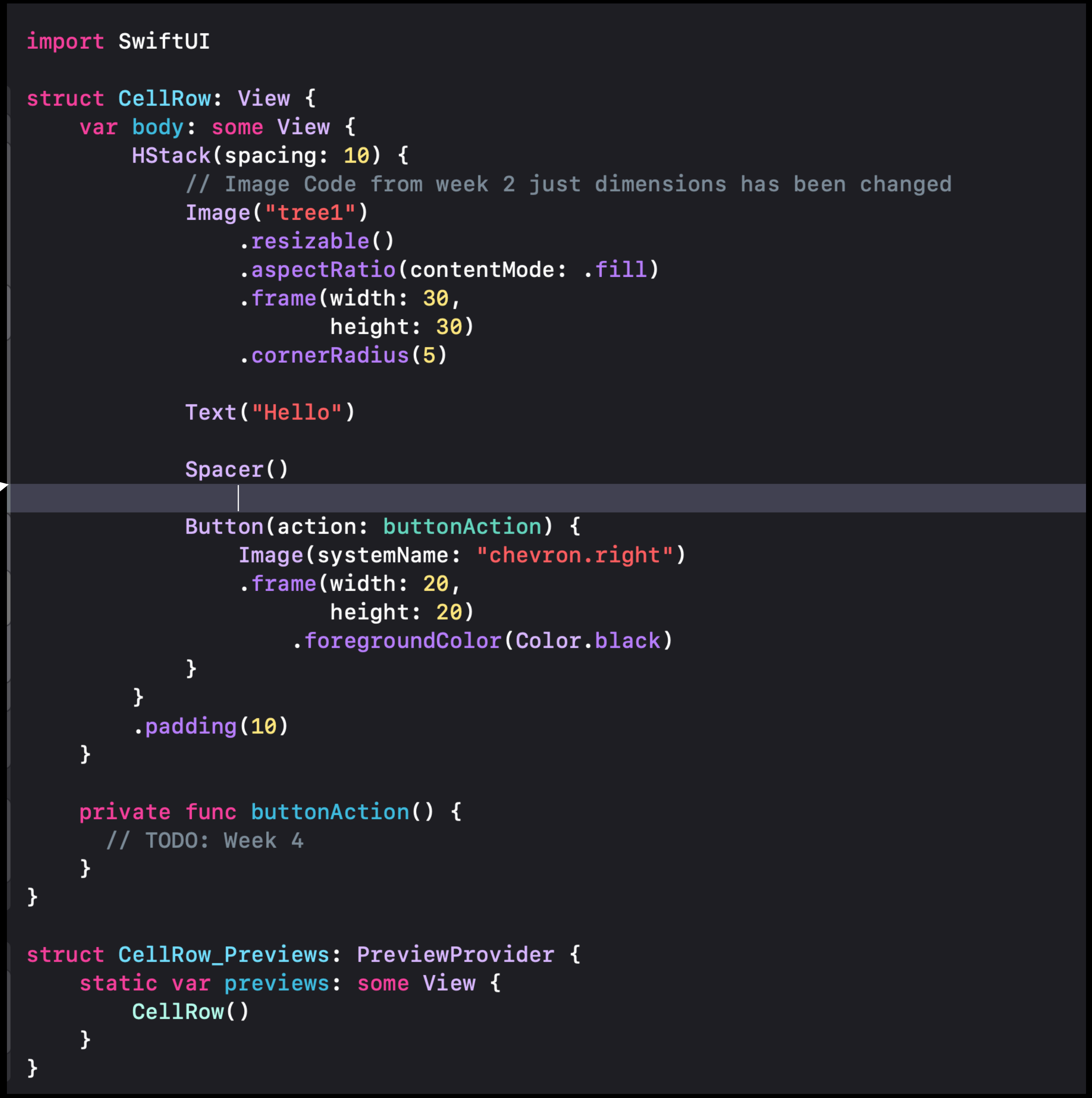


```
Button(action: buttonAction) {
    Image(systemName: "chevron.right")
    .frame(width: 20,
           height: 20)
        .foregroundColor(Color.black)
}
```

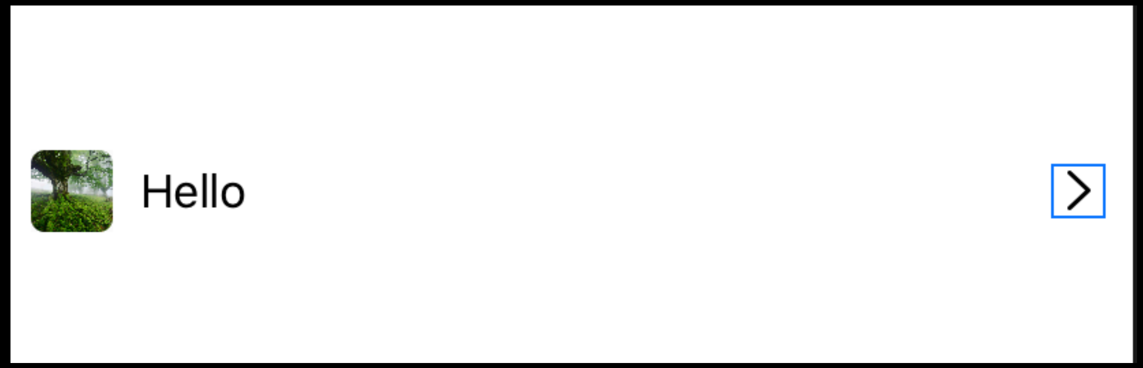


```
private func buttonAction() {
  // TODO: Week 4
}
```

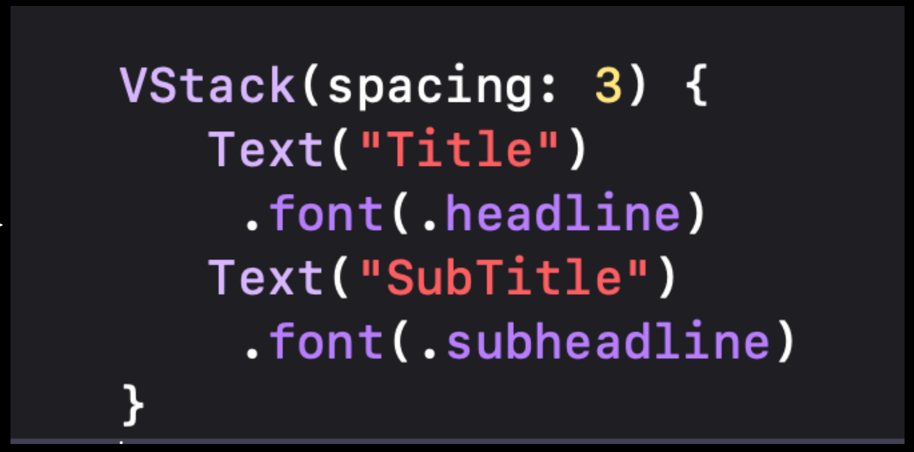
The final code snippet will look like this with the observation below



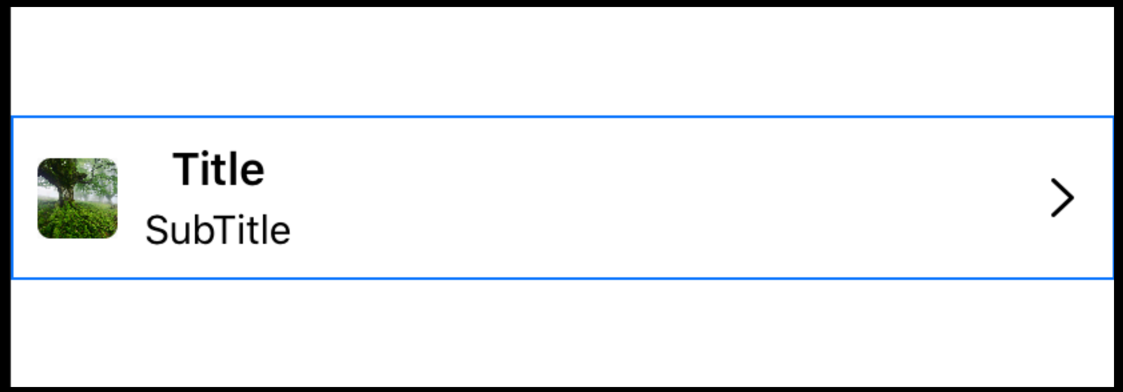
```swift
import SwiftUI

struct CellRow: View {
    var body: some View {
        HStack(spacing: 10) {
            // Image Code from week 2 just dimensions has been changed
            Image("tree1")
                .resizable()
                .aspectRatio(contentMode: .fill)
                .frame(width: 30,
                       height: 30)
                .cornerRadius(5)

            Text("Hello")

            Spacer()

            Button(action: buttonAction) {
                Image(systemName: "chevron.right")
                .frame(width: 20,
                       height: 20)
                    .foregroundColor(Color.black)
            }
        }
        .padding(10)
    }

    private func buttonAction() {
      // TODO: Week 4
    }
}

struct CellRow_Previews: PreviewProvider {
    static var previews: some View {
        CellRow()
    }
}
```



Step 8: Lets add **VStack** and change our Texts to “Title” and “Subtitle”.
We can see that in first observation the texts are centered aligned

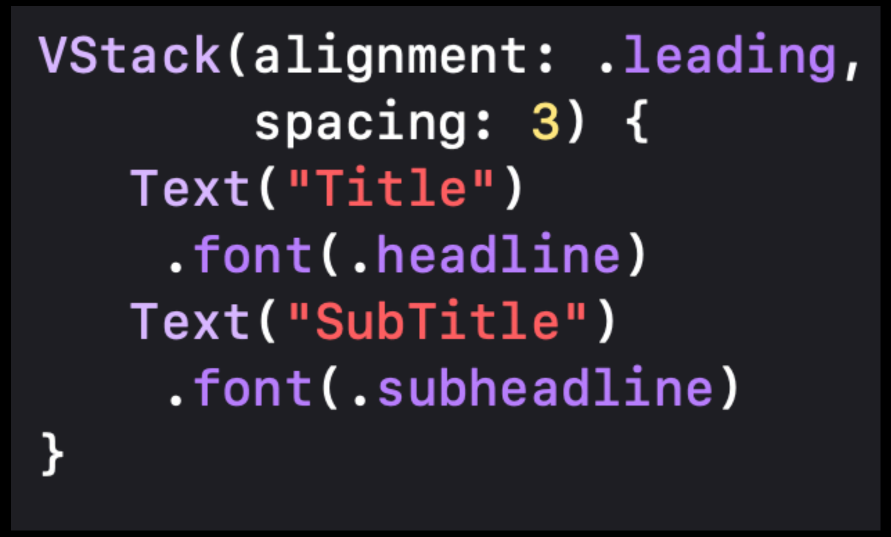


```
VStack(spacing: 3) {
    Text("Title")
     .font(.headline)
    Text("SubTitle")
     .font(.subheadline)
}
```

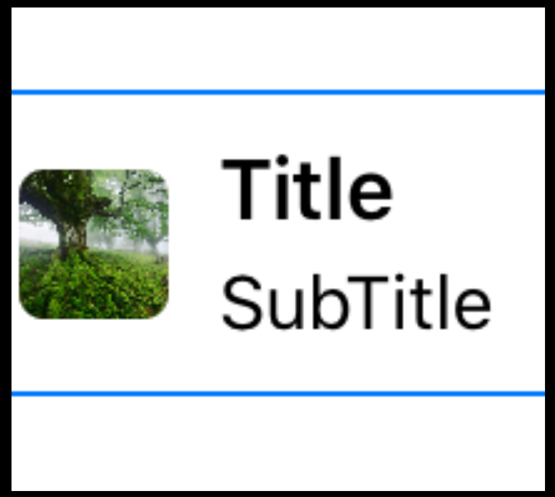


Title
SubTitle

Lets add alignment in VStack

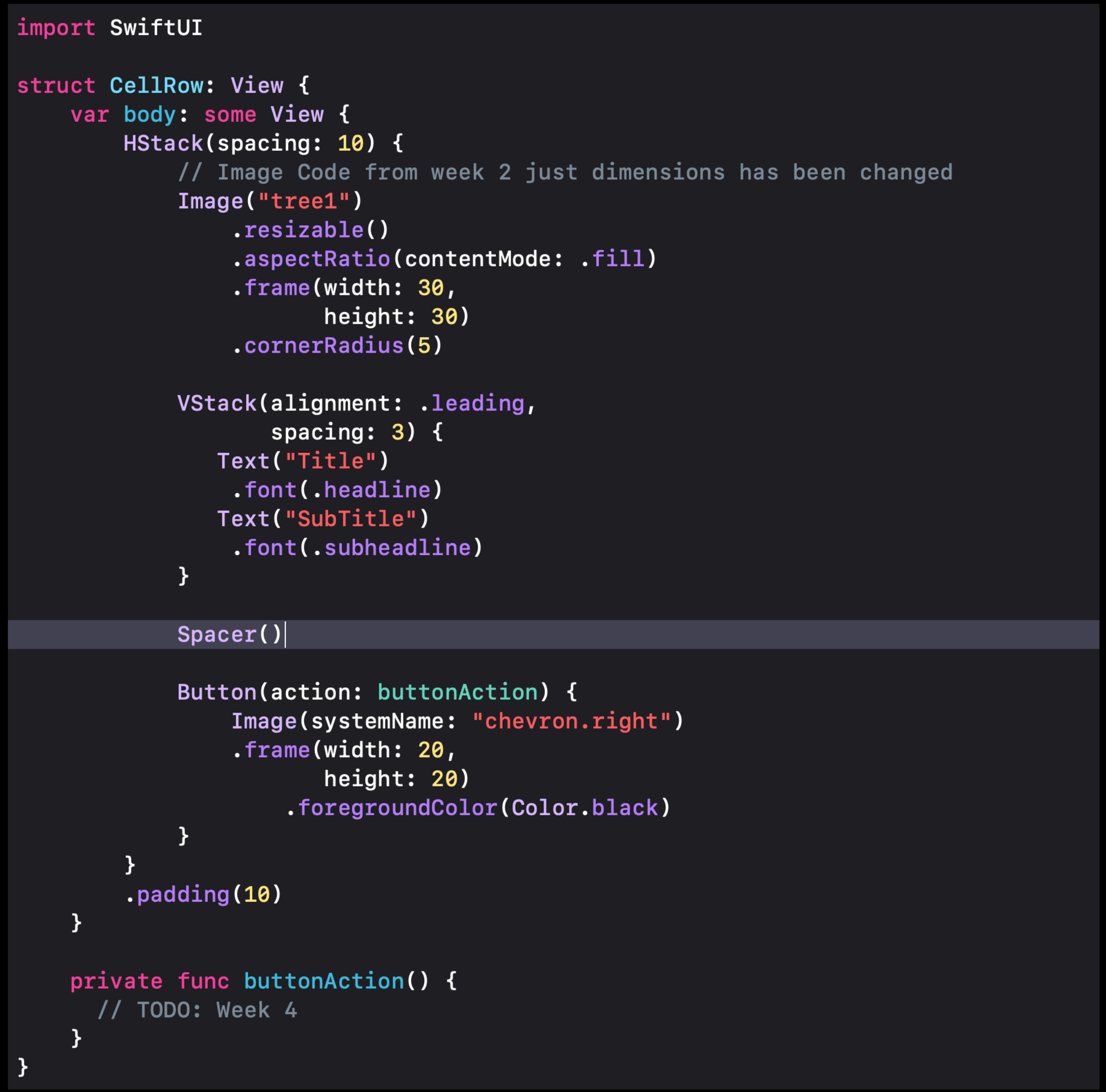


```
VStack(alignment: .leading,
       spacing: 3) {
   Text("Title")
    .font(.headline)
   Text("SubTitle")
    .font(.subheadline)
}
```



Title
SubTitle

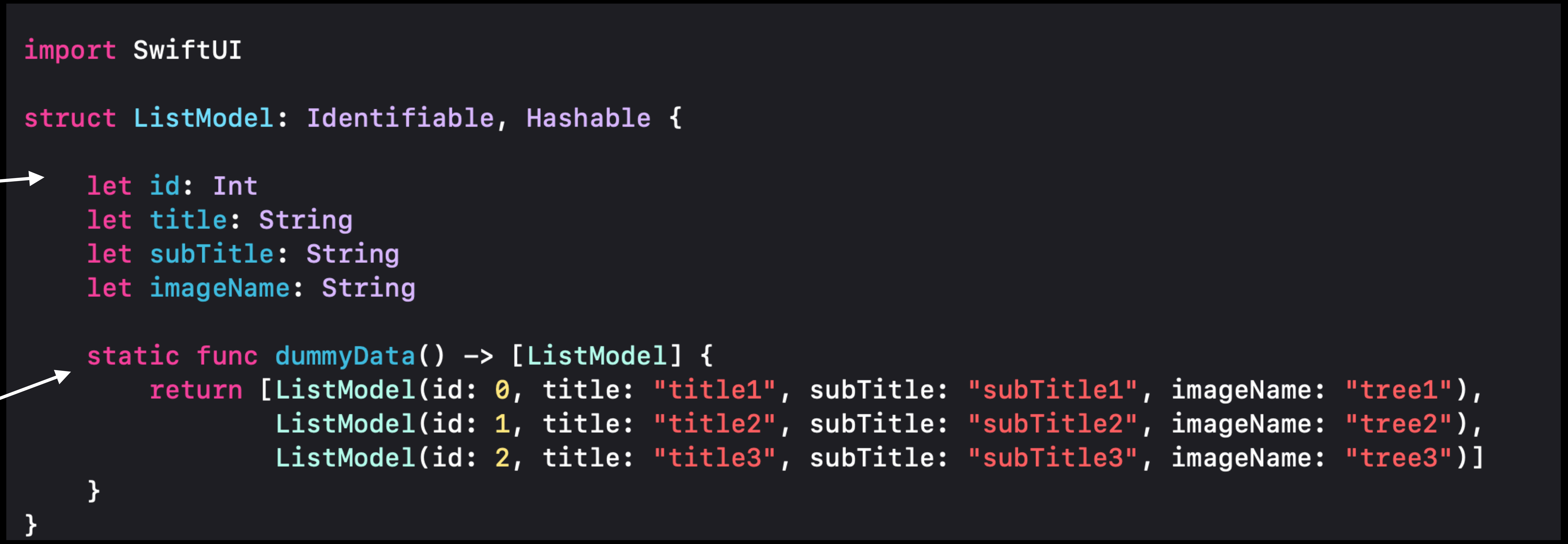
The code snippet will look like this.
Thats it for Cell now lets make a model before going on to list



```
import SwiftUI

struct CellRow: View {
    var body: some View {
        HStack(spacing: 10) {
            // Image Code from week 2 just dimensions has been changed
            Image("tree1")
                .resizable()
                .aspectRatio(contentMode: .fill)
                .frame(width: 30,
                       height: 30)
                .cornerRadius(5)

            VStack(alignment: .leading,
                   spacing: 3) {
               Text("Title")
                .font(.headline)
               Text("SubTitle")
                .font(.subheadline)
            }

            Spacer()

            Button(action: buttonAction) {
                Image(systemName: "chevron.right")
                .frame(width: 20,
                       height: 20)
                    .foregroundColor(Color.black)
            }
        }
        .padding(10)
    }

    private func buttonAction() {
      // TODO: Week 4
    }
}
```

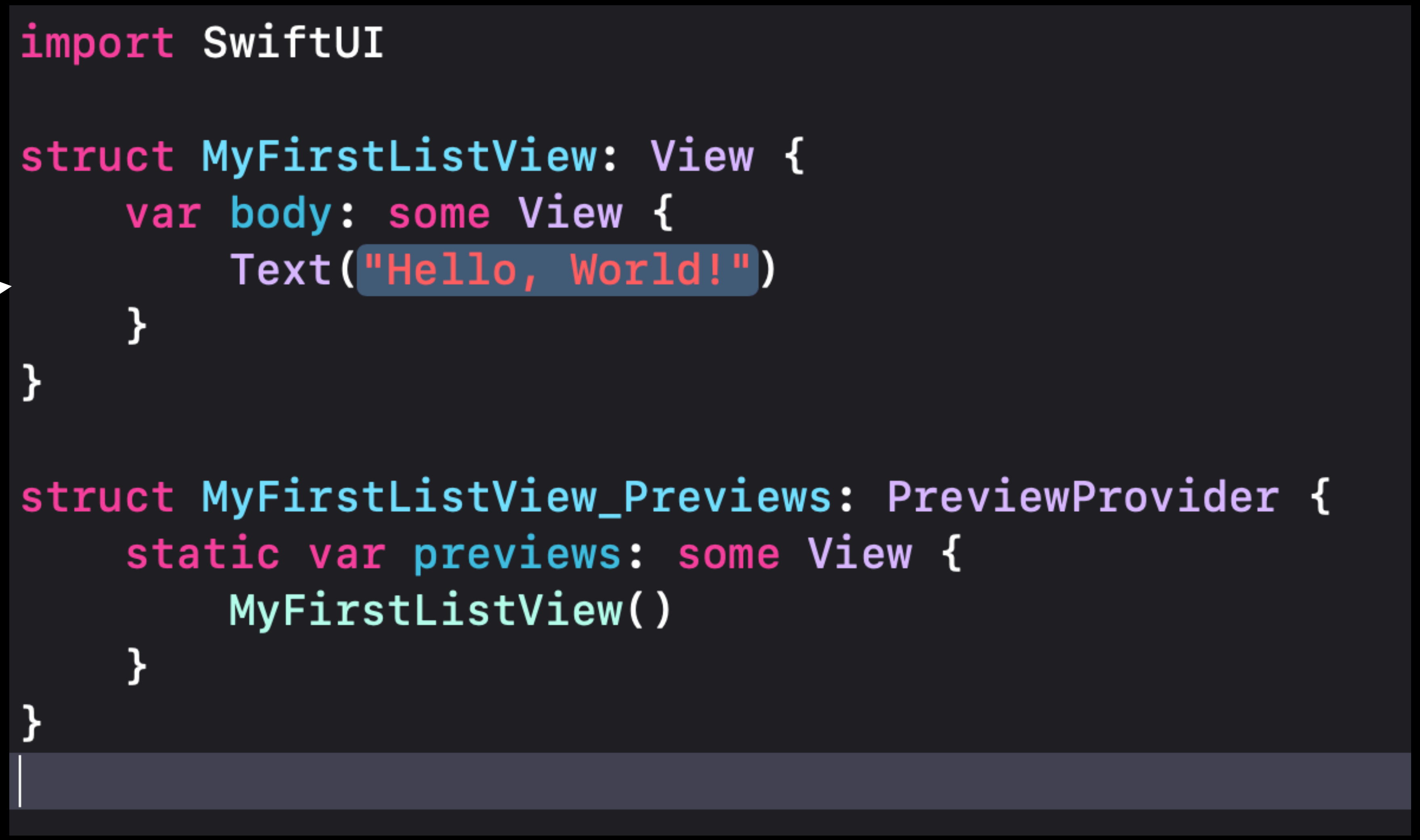
Step 9: This is the snippet for model with SwiftUI way, Generally the model should be hashable and Identifiable
For more info please feel free to research

The method below is just kind a dummy method for data



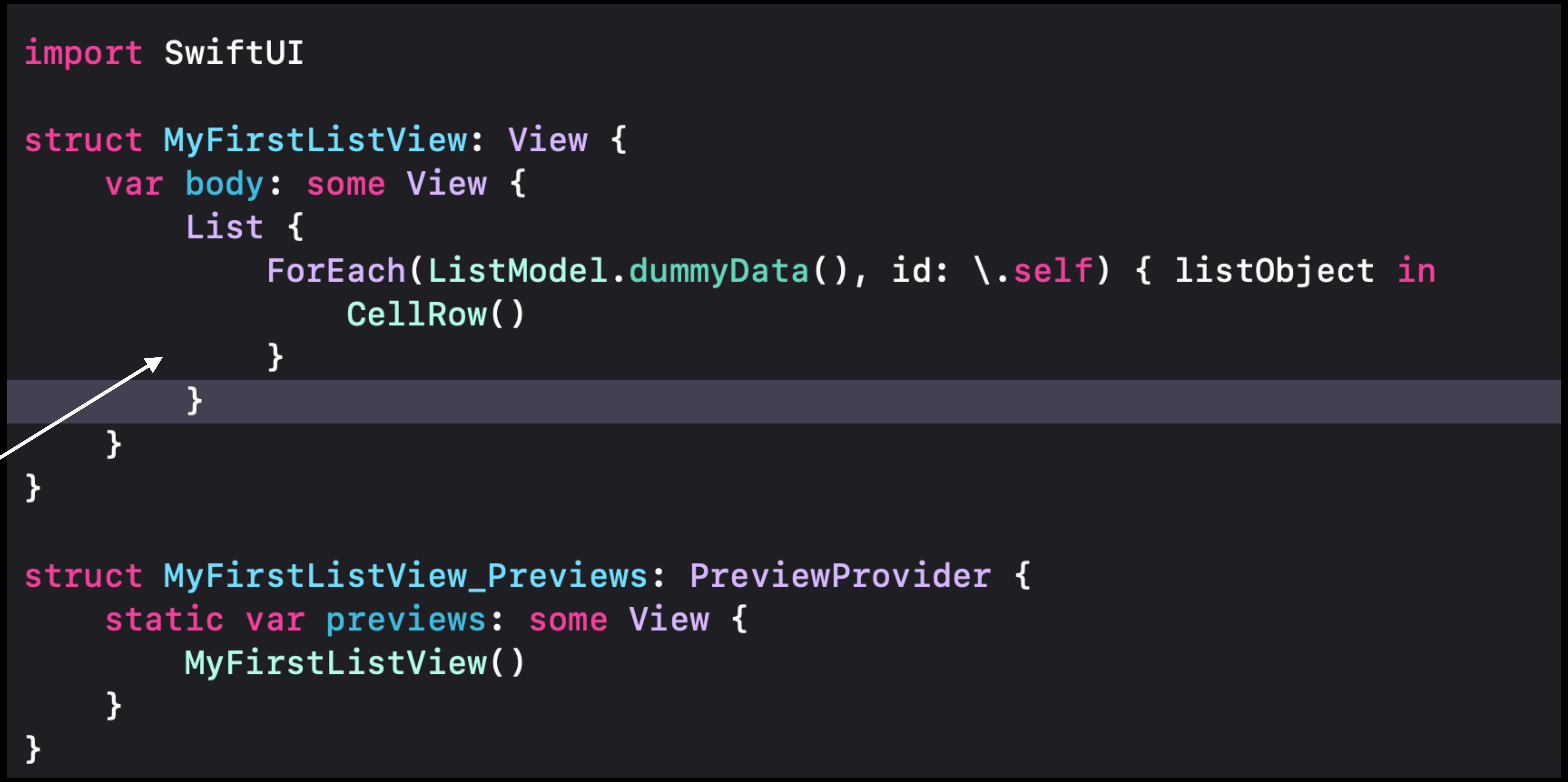
```swift
import SwiftUI

struct ListModel: Identifiable, Hashable {

    let id: Int
    let title: String
    let subTitle: String
    let imageName: String

    static func dummyData() -> [ListModel] {
        return [ListModel(id: 0, title: "title1", subTitle: "subTitle1", imageName: "tree1"),
                ListModel(id: 1, title: "title2", subTitle: "subTitle2", imageName: "tree2"),
                ListModel(id: 2, title: "title3", subTitle: "subTitle3", imageName: "tree3")]
    }
}
```

Step 10: Now that we have made model, lets make a new file called “MyFirstListView”

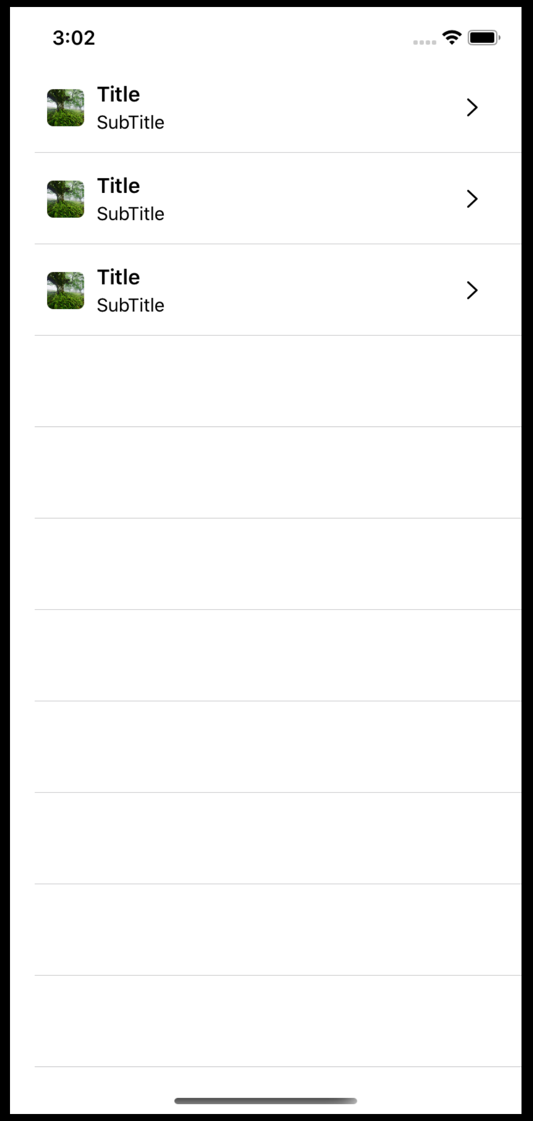


```swift
import SwiftUI

struct MyFirstListView: View {
    var body: some View {
        Text("Hello, World!")
    }
}

struct MyFirstListView_Previews: PreviewProvider {
    static var previews: some View {
        MyFirstListView()
    }
}
```

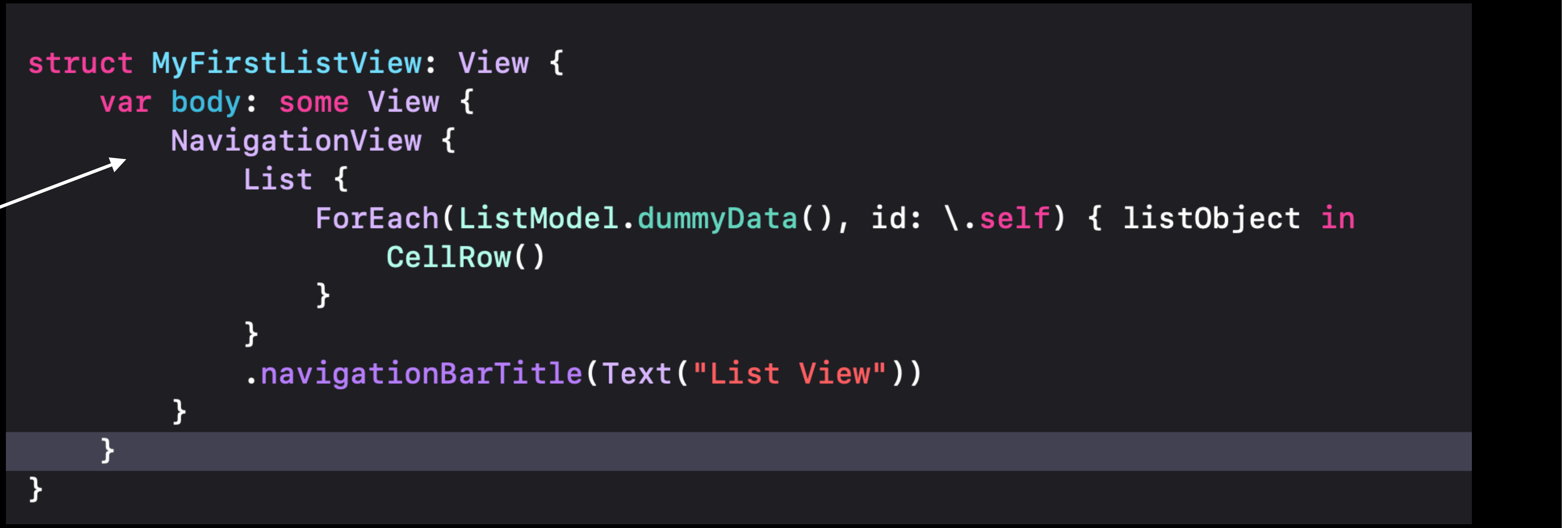
Step 11: Now lets add List to our new file, We have used ForEach Technique to populate list,
You guys can use any technique



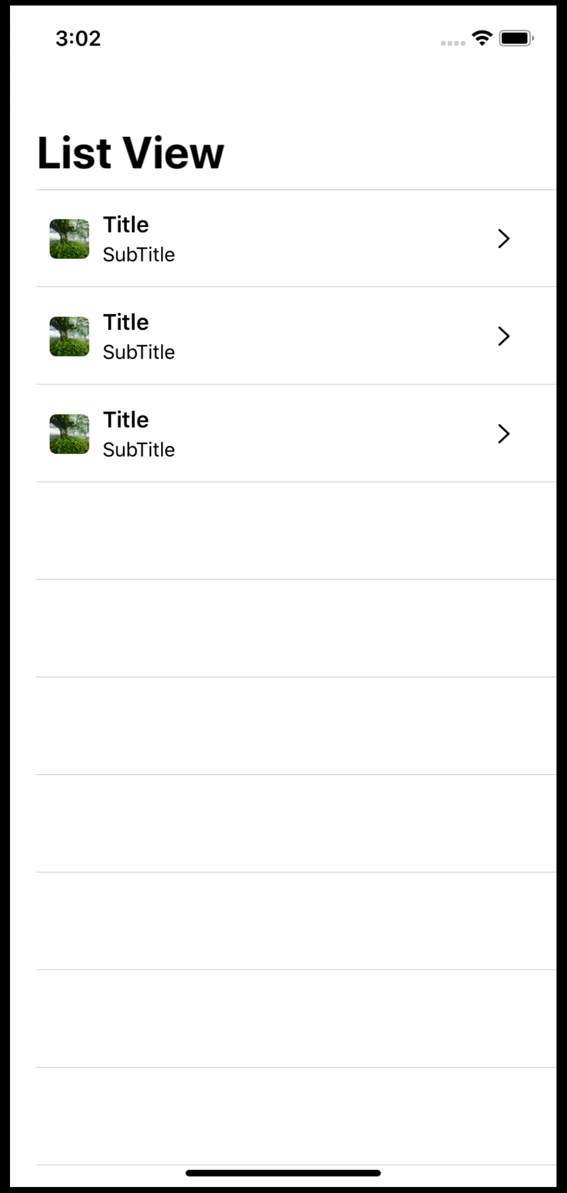
```
import SwiftUI

struct MyFirstListView: View {
    var body: some View {
        List {
            ForEach(ListModel.dummyData(), id: \.self) { listObject in
                CellRow()
            }
        }
    }
}

struct MyFirstListView_Previews: PreviewProvider {
    static var previews: some View {
        MyFirstListView()
    }
}
```



Also since the list was aligned to much on top, We have added “NavigationView”

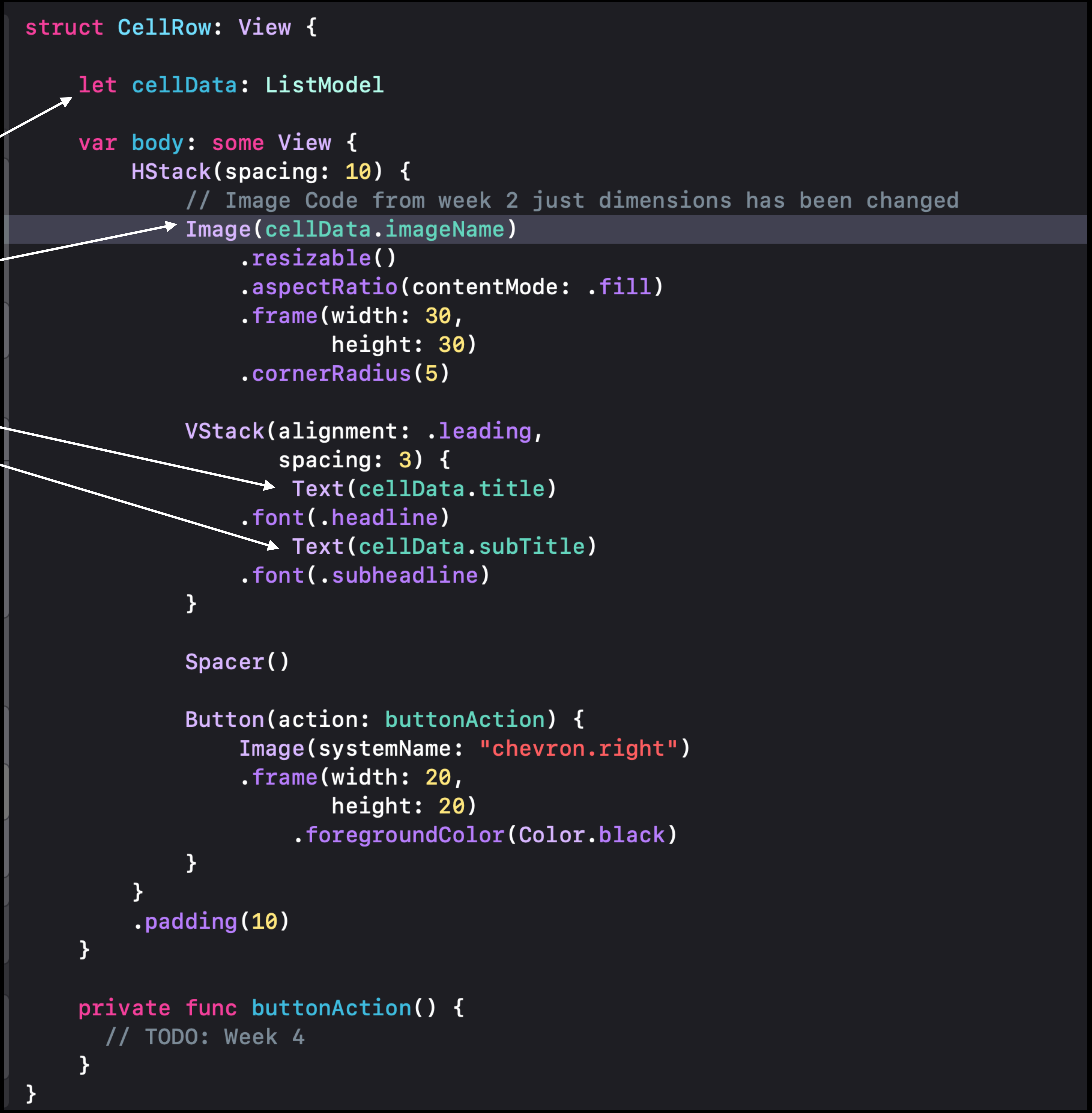


```
struct MyFirstListView: View {
    var body: some View {
        NavigationView {
            List {
                ForEach(ListModel.dummyData(), id: \.self) { listObject in
                    CellRow()
                }
            }
            .navigationBarTitle(Text("List View"))
        }
    }
}
```



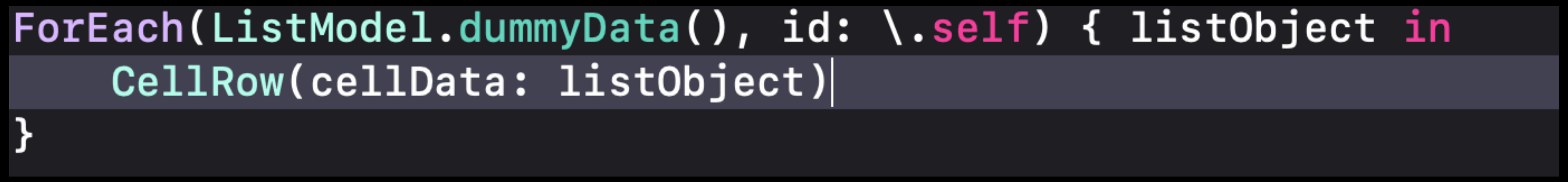
Observations

Step 12: Now lets update our CellRow File, So that we can populate dynamic list

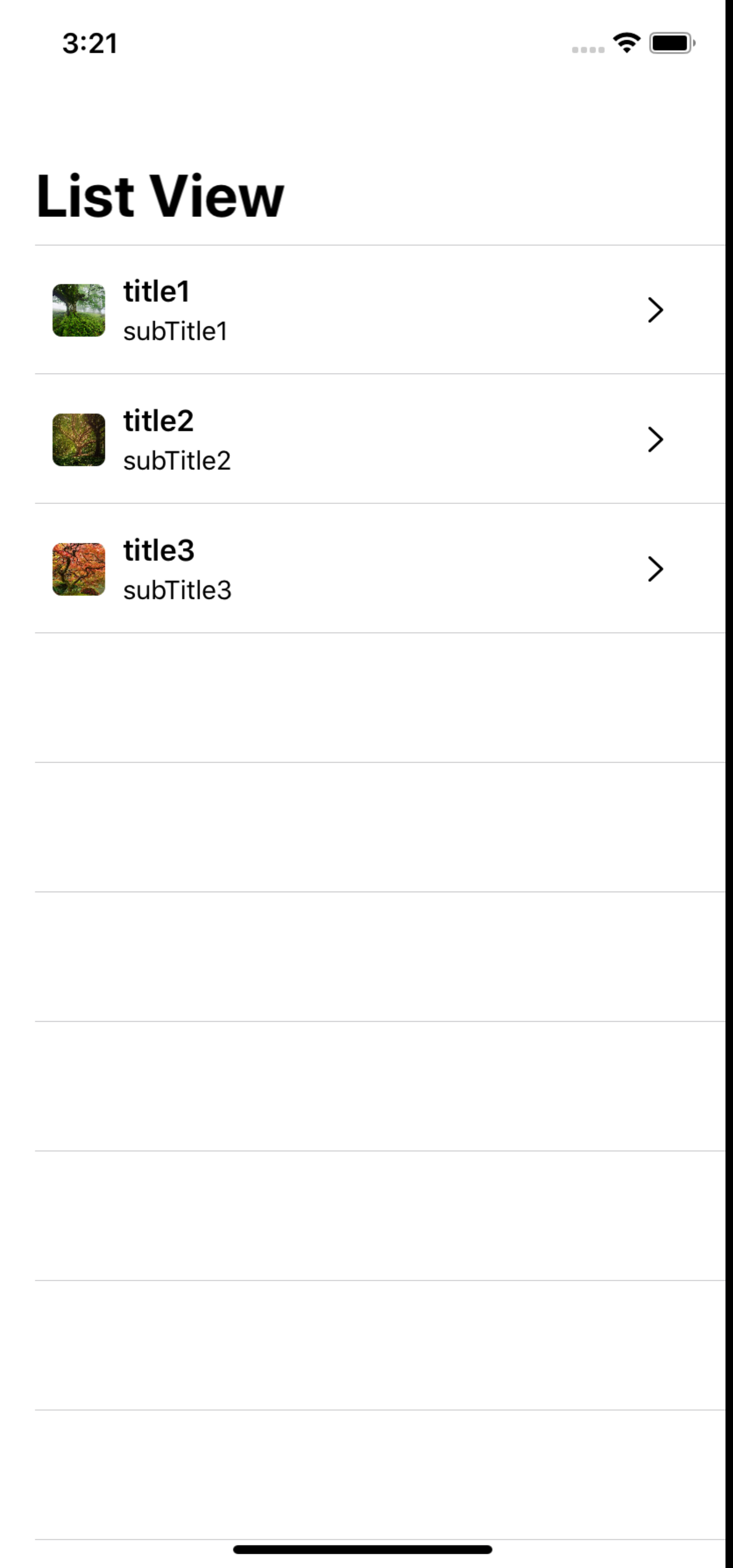


```
struct CellRow: View {

    let cellData: ListModel

    var body: some View {
        HStack(spacing: 10) {
            // Image Code from week 2 just dimensions has been changed
            Image(cellData.imageName)
                .resizable()
                .aspectRatio(contentMode: .fill)
                .frame(width: 30,
                       height: 30)
                .cornerRadius(5)

            VStack(alignment: .leading,
                   spacing: 3) {
                    Text(cellData.title)
                .font(.headline)
                    Text(cellData.subTitle)
                .font(.subheadline)
            }

            Spacer()

            Button(action: buttonAction) {
                Image(systemName: "chevron.right")
                .frame(width: 20,
                       height: 20)
                    .foregroundColor(Color.black)
            }
        }
        .padding(10)
    }

    private func buttonAction() {
      // TODO: Week 4
    }
}
```

Final step: Let’s update the List file CellRow Line to get the following observation



```
ForEach(ListModel.dummyData(), id: \.self) { listObject in
    CellRow(cellData: listObject)
}
```



3:21

List View

title1
subTitle1

title2
subTitle2

title3
subTitle3

Thats all folks for Week 3

Do same exercise with scroll view for extra 5 points.

See you in week 4 ……