SwiftUI Essentials

Snehal Patil

SwiftUI

SwiftUI is a relatively new framework introduced by **Apple in 2019**. It provides a platform for developers to use a declarative syntax to create user interfaces for **apps running on iOS, iPadOS, macOS, watchOS, and tvOS**.

SwiftUI is a tool that easily interacts with the **Swift framework programming language** to make creating user interfaces for apps simpler and faster.

SwiftUI is ideal for cross-platform compatibility and is renowned for its capacity to design responsive and aesthetically pleasing **user interfaces**. SwiftUI employs a declarative syntax, which allows developers to declare how the UI should look and behave rather than laying down a set of procedural steps, which is one of the main **differences between SwiftUI and UIKit.** As a result, **UI developmen**t is easier and faster, but customization options are more limited.

UIKit

UIKit is the more mature **IOS development framework** having been released in **2007**. Since then it has been used to create some of the **world’s top apps for the IOS platform**.

It offers developers a high level of **customization and flexibility** when they design every aspect of the app and particularly when it comes to the user interface. But that flexibility comes with the price of more complex code that is required to define every aspect of the **app design**.

UIKit is also not **cross-platform like SwiftUI**, it only produces apps for IOS and not the full range of **apple products**.

Migrating from UIKit to SwiftUI

Here's a list to get you started, with UIKit class names followed by SwiftUI names:

- `UITableView`: `List`
- `UICollectionView`: `Grid`, `LazyVGrid` and `LazyHGrid`
- `UIScrollView`: `ScrollView`
- `UILabel`: `Text`
- `UITextField`: `TextField`
- `UITextField` with `isSecureTextEntry` set to true: `SecureField`
- `UITextView`: `TextEditor` (plain strings only)
- `UISwitch`: `Toggle`
- `UISlider`: `Slider`
- `UIButton`: `Button`
- `UINavigationController`: `NavigationStack` or `NavigationSplitView`
- `UIAlertController` with style `.alert`: `.alert()`
- `UIAlertController` with style `.actionSheet`: `.confirmationDialog()`
- `UIStackView` with horizontal axis: `HStack`
- `UIStackView` with vertical axis: `VStack`
- `UIImageView`: `Image`
- `UISegmentedControl`: `Picker`
- `UIStepper`: `Stepper`
- `UIDatePicker`: `DatePicker`
- `UIProgressView`: `ProgressView` with a value
- `UIActivityIndicatorView`: `ProgressView` without a value
- `MKMapView`: `Map`
- `NSAttributedString`: `AttributedString`

Text

```
Text("New York")

   .fontWeight(.bold)

   .font(.system(size: 12, weight: .light, design: .serif))



Text("This sans-serif typeface is the system font for iOS, macOS, and tvOS, and includes a rounded variant. It provides a consistent, legible, and friendly typographic voice.")

           .frame(width: 100)

           .lineLimit(1)

           .lineSpacing(10)
```

Button

```
Button("Press Me") {
    print("Button pressed!")
}
.padding()
.background(Color(red: 0, green: 0, blue: 0.5))
.clipShape(Capsule())
```

Button Style

```
struct BlueButton: ButtonStyle {
    func makeBody(configuration: Configuration) -> some View {
        configuration.label
            .padding()
            .background(Color(red: 0, green: 0, blue: 0.5))
            .foregroundStyle(.white)
            .clipShape(Capsule())
    }
}

struct ContentView: View {
    var body: some View {
        Button("Press Me") {
            print("Button pressed!")
        }
        .buttonStyle(BlueButton())
    }
}
```

Growing Buttons

```
struct GrowingButton: ButtonStyle {
    func makeBody(configuration: Configuration) -> some View {
        configuration.label
            .padding()
            .background(.blue)
            .foregroundStyle(.white)
            .clipShape(Capsule())
            .scaleEffect(configuration.isPressed ? 1.2 : 1)
            .animation(.easeOut(duration: 0.2), value: configuration.isPressed)
    }
}

struct ContentView: View {
    var body: some View {
        Button("Press Me") {
            print("Button pressed!")
        }
        .buttonStyle(GrowingButton())
    }
}
```

Slider

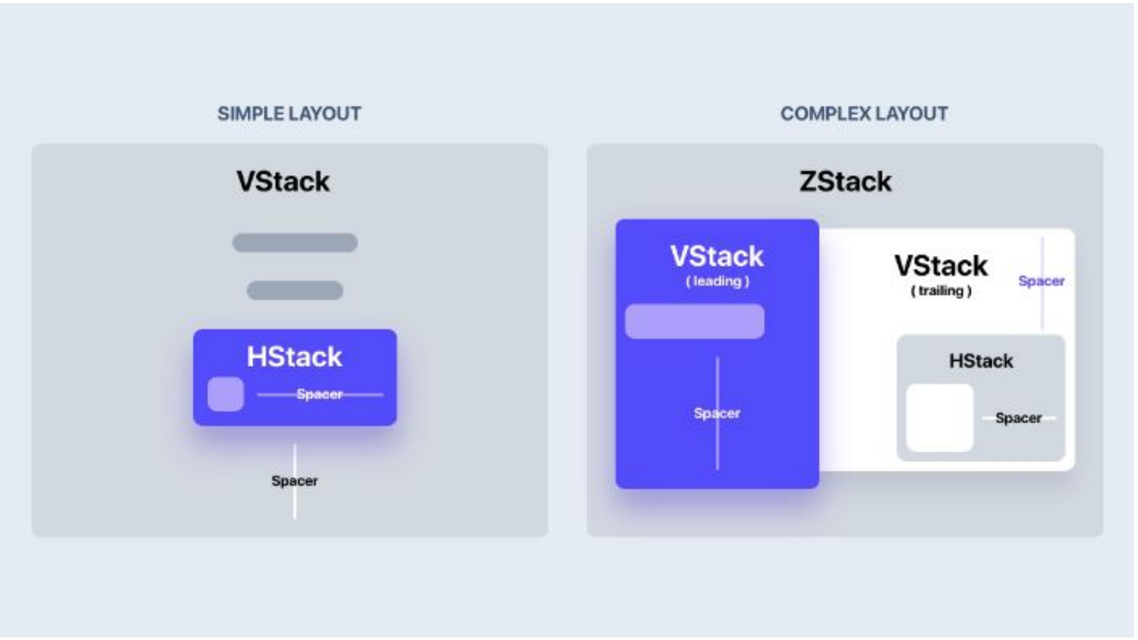
```
VStack {
      Text("Drag the slider to blur me")
          .blur(radius: blurAmount)

      Slider(value: $blurAmount, in: 0...20)
    }
```

Image

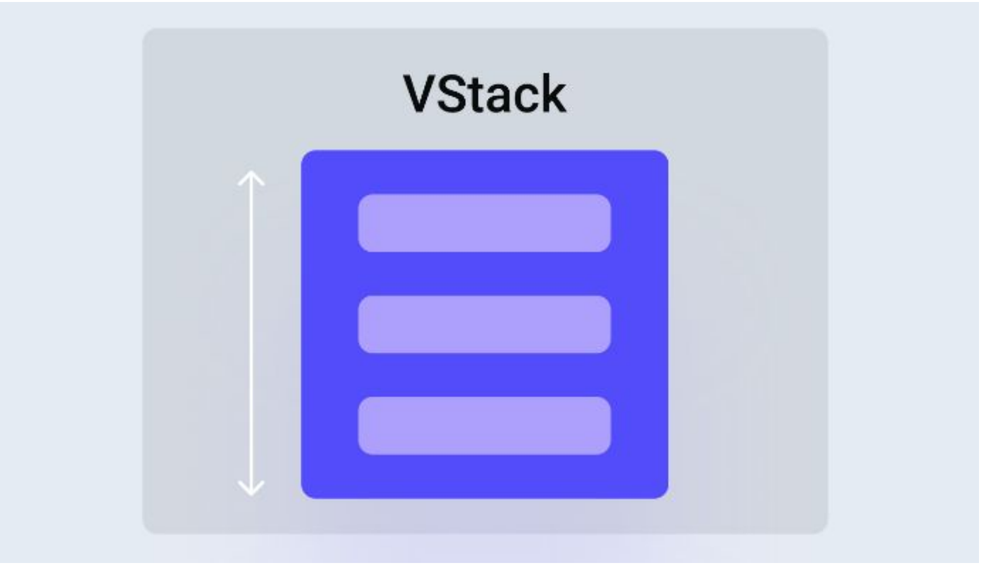
```
Image("Illustration")
    .resizable(resizingMode: .tile)
    .aspectRatio(contentMode: .fit)
```

```
Image("Illustration")
    .resizable()
    .aspectRatio(contentMode: .fit)
    .frame(width: 200, height: 200, alignment: .center)
```

Stacks and Spacer

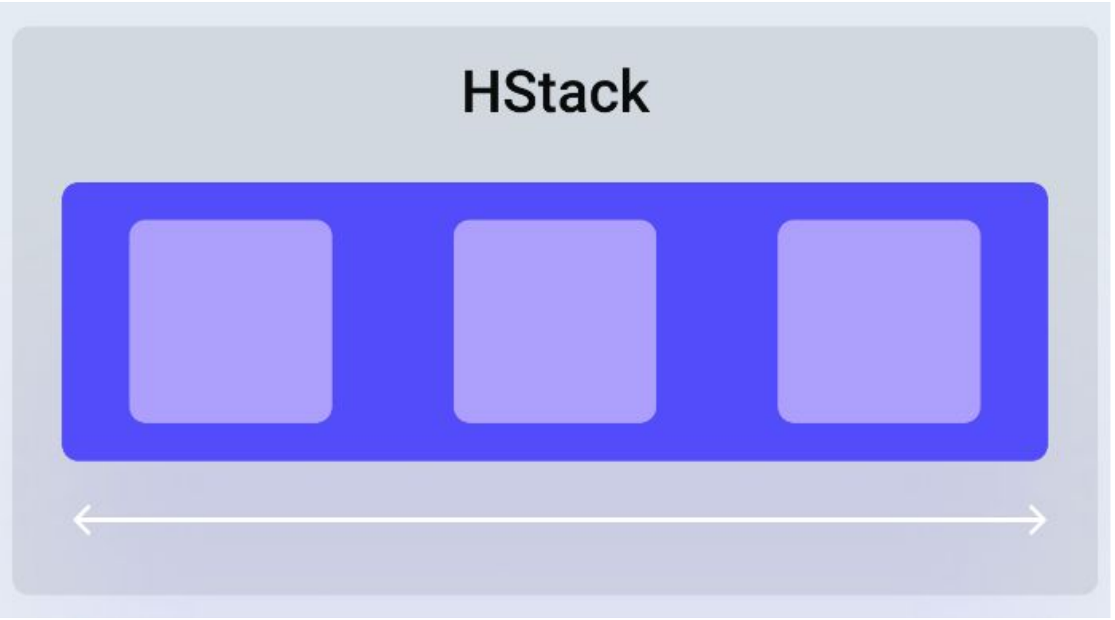


VStack



```
VStack(alignment: .leading, spacing: 16) {
        Text("Hello, world!")
                .font(.title)
        Spacer()
        Text("Second line")
  }
```

HStack

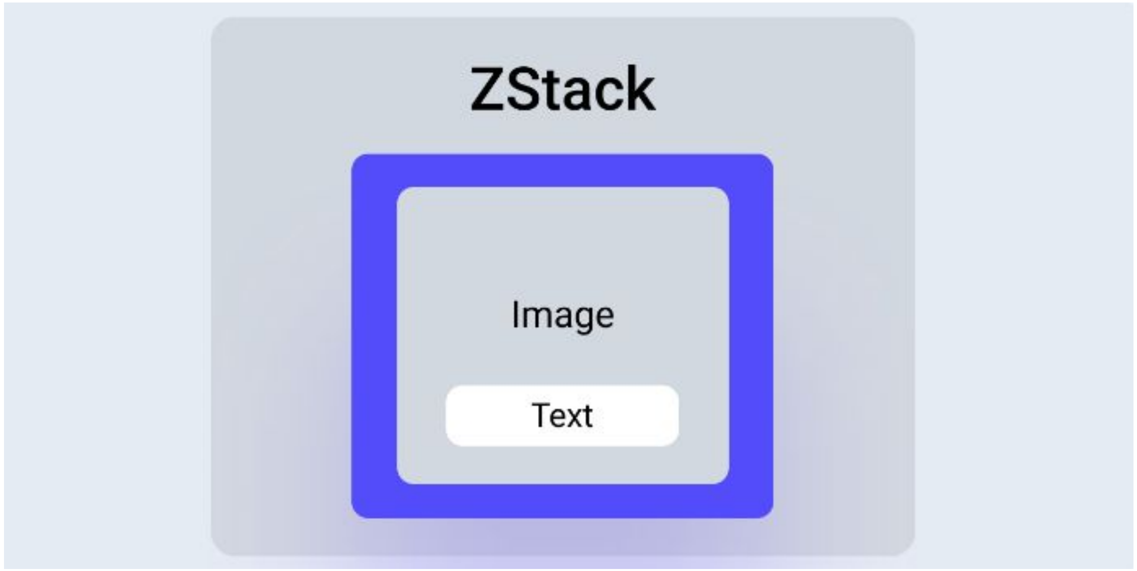


```
HStack(alignment: .bottom,
spacing: 16) {

        Text("Hello, world!")

                .font(.title)

        Spacer()

        Text("Second line")

}
```

Spacer

```
HStack(alignment: .bottom, spacing: 16) {
        Text("Hello, world!")
                .font(.title)
        Spacer()
        Text("Second line")
}
.padding()
.frame(width: 320)
```

ZStack



```
ZStack(alignment: .topLeading) {
    Rectangle()
        .foregroundColor(.blue)
    Text("Hello, world!")
        .font(.title)
    Spacer()
    Text("Second line")
}
.padding()
.frame(width: 320)
```

ZStack - Overlapping Content

```
ZStack {
    Image("cat")
    Text("My Fav Pet")
        .font(.largeTitle)
        .background(.black)
        .foregroundStyle(.white)
}
```

Shapes and Strokes

```
Circle()
        .stroke(Color.black, lineWidth: 2)
        .frame(width: 44, height: 44)
Ellipse()
        .stroke(Color.black, lineWidth: 2)
        .frame(width: 44, height: 88)
Rectangle()
   .foregroundColor(.blue)
     .ignoresSafeArea()
```

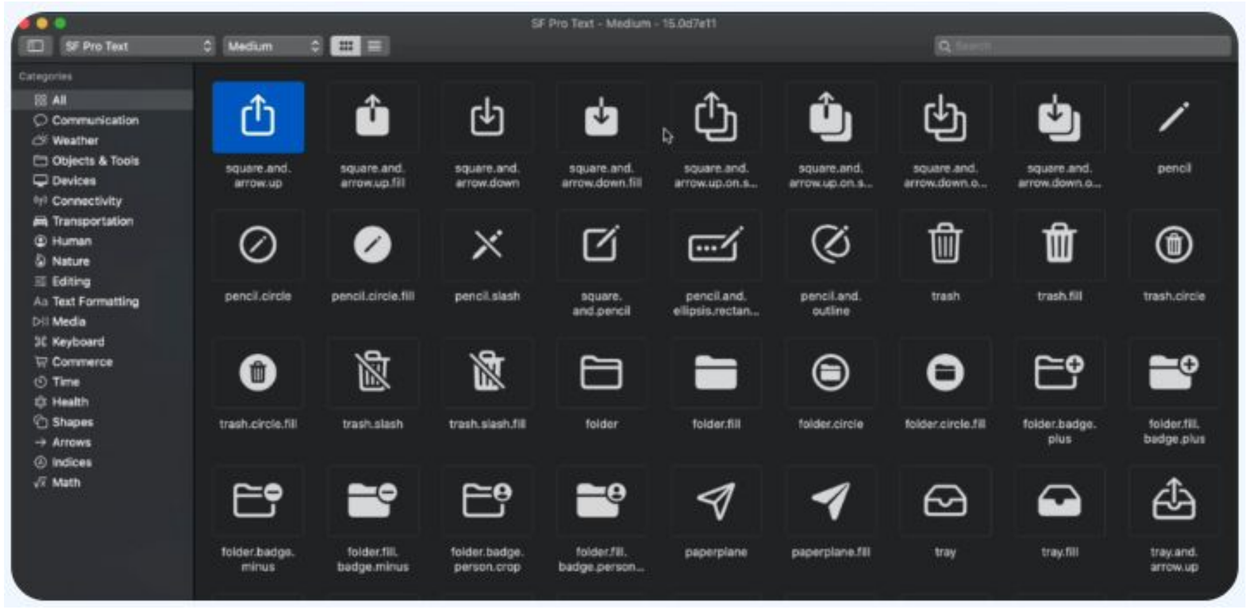
Continued

```
RoundedRectangle(cornerRadius: 30, style: .continuous)
        .fill(Color.green)
        .frame(height: 44)
        .overlay(Text("Sign up").bold())
Capsule()
        .fill(Color.green)
        .frame(height: 44)
        .overlay(Text("Sign up").bold())
```

Shapes

```
ZStack {
        Rectangle()
            .fill(Color.blue).ignoresSafeArea()
        VStack {
            Text("MyBook").bold()
            Capsule()
                .foregroundColor(Color.green)
                .frame(height: 44)
                .overlay(Text("Sign up"))
            Capsule()
                .foregroundColor(Color.green)
                .frame(height: 44)
                .overlay(Text("Sign In"))
            Capsule()
                .foregroundColor(Color.green)
                .frame(height: 44)
                .overlay(Text("Forgot Username/Password"))
        }
        .padding()
        .background(Color.white)
        .clipShape(RoundedRectangle(cornerRadius: 25.0, style: .continuous))
        .padding()
    }
```

SF Symbols



```
Image(systemName: "gear")
        .font(.system(size: 20,
weight: .light))

Image(systemName: "gear")
        .imageScale(.large)

Image(systemName:
"paperplane.circle.fill")
        .renderingMode(.original)
```

Toolbar Continued

```
ToolbarItemGroup(placement: .bottomBar) {
        Image(systemName: "person")
        Spacer()
        Image(systemName: "ellipsis")
        Spacer()
        Image(systemName: "trash")
}
```

Toolbar Continued

```
NavigationView {
    Text("My app")
    .toolbar {
        ToolbarItemGroup(placement: .bottomBar) {
            Image(systemName: "person")
            HStack {
                Image(systemName: "ellipsis")
                Divider()
                Image(systemName: "trash")
                    .frame(width: 32, height: 32)
                    .background(Color.blue)
                    .mask(Circle())
            }
        }
    }
}
```

SideBar (Navigation Bar)

```
NavigationView {
    List {
        Label("Courses", systemImage: "book")
        Label("Tutorials", systemImage: "square")
    }
    .navigationTitle("Learn")
}
```

NavigationView

```
struct ContentView: View {
    var body: some View {
        NavigationView {
            List {
                NavigationLink(destination: CoursesHomeView()) {
                    Label("Courses", systemImage: "book")
                }
                NavigationLink(destination: TutorialsHomeView()) {
                    Label("Tutorials", systemImage: "square")
                }
            }
            .navigationTitle("Learn")
        }
    }
}

struct CoursesHomeView: View {
    var body: some View {
        Text("Courses")
            .navigationTitle("Courses")
    }
}

struct TutorialsHomeView: View {
    var body: some View {
        Text("Tutorials")
            .navigationTitle("Tutorials")
    }
}
```

Navigation View Toolbar

```
struct ContentView: View {
    var body: some View {
        NavigationView {
            List {
                NavigationLink(destination: CoursesHomeView()) {
                    Label("Courses", systemImage: "book")
                }
                NavigationLink(destination: TutorialsHomeView()) {
                    Label("Tutorials", systemImage: "square")
                }
            }
            .navigationTitle("Learn")
            .toolbar {
                ToolbarItemGroup(placement: .confirmationAction) {
                    Image(systemName: "person")
                    Image(systemName: "ellipsis")
                }
            }
        }
    }
}
```

ToolBarItemGroup

```
NavigationView {
          List {
             NavigationLink(destination: CoursesHomeView()) {
                Label("Courses", systemImage: "book")
             }
             NavigationLink(destination: TutorialsHomeView()) {
                Label("Tutorials", systemImage: "square")
             }
          }
          .navigationTitle("Learn")
          .toolbar {
             ToolbarItemGroup(placement: .confirmationAction) {
                Button(action: {print("option A")}, label: {Label("My option A", systemImage: "folder.badge.plus")})
                Button(action: {print("option B")}, label: {Label("My option B", systemImage: "doc.badge.plus")})
             }
          }
       }
```

Toolbar and Menu

```
struct ContentView: View {
    var body: some View {
        NavigationView {
                List {
                    NavigationLink(destination: CoursesHomeView()) {
                        Label("Courses", systemImage: "book")
                    }
                    NavigationLink(destination: TutorialsHomeView()) {
                        Label("Tutorials", systemImage: "square")
                    }
                }
                .navigationTitle("Learn")
                .toolbar {
                        ToolbarItem(placement: .navigationBarTrailing) {
                            Menu(content: {
                                Label("My option A", systemImage: "folder.badge.plus")
                                    .onTapGesture {
                                        print("My option A")
                                    }
                                Label("My option B", systemImage: "doc.badge.plus")
                                    .onTapGesture {
                                        print("My option B")
                                    }
                            }, label: {
                                Image(systemName: "plus")
                                    .imageScale(.large)
                                    .background(Color.red)
                            })
                        }
                    }
            }
    }
}
```

Grid

```
Grid {
    GridRow {
        Text("Hello")
        Image(systemName: "globe")
    }
    GridRow {
        Image(systemName: "hand.wave")
        Text("World")
    }
}
```

Gestures

```
struct ContentView: View {
    var body: some View {
        Image(systemName: "star.circle.fill")
                .font(.system(size: 200))
                .foregroundColor(.green)
                .gesture(
                    TapGesture()
                        .onEnded({
                            print("Tapped!")
                        })
                )
    }
}
```

Creating and Combining Views - Landmarks project

```
import SwiftUI
struct ContentView: View {
    var body: some View {
        VStack {
            MapView()
                .ignoresSafeArea(edges: .top)
                .frame(height: 300)
            CircleImage()
                .offset(y: -130)
                .padding(.bottom, -130)
            VStack(alignment: .leading) {
                Text("Turtle Rock")
                    .font(.title)
                HStack {
                    Text("Joshua Tree National Park")
                    Spacer()
                    Text("California")
                }
                .font(.subheadline)
                .foregroundColor(.secondary)
                Divider()
                Text("About Turtle Rock")
                    .font(.title2)
                Text("Descriptive text goes here.")
            }
            .padding()
            Spacer()
        }
    }
}
struct ContentView_Previews: PreviewProvider {
    static var previews: some View {
        ContentView()
    }
}
```

CircleImage

```
import SwiftUI
struct CircleImage: View {
    var body: some View {
        Image("turtlerock")
            .clipShape(/*@START_MENU_TOKEN@*/Circle()/*@END_MENU_TOKEN@*/)
            .overlay {
                Circle().stroke(.white, lineWidth: 4)
            }
            .shadow(radius: 7)
    }
}
struct CircleImage_Previews: PreviewProvider {
    static var previews: some View {
        CircleImage()
    }
}
```

MapView

```
import SwiftUI
import MapKit
struct MapView: View {
    @State private var region = MKCoordinateRegion(
        center: CLLocationCoordinate2D(latitude: 34.011_286, longitude: -116.166_868),
        span: MKCoordinateSpan(latitudeDelta: 0.2, longitudeDelta: 0.2)
    )
    var body: some View {
        Map(coordinateRegion: $region)
    }
}
struct MapView_Previews: PreviewProvider {
    static var previews: some View {
        MapView()
    }
}
```

References

https://developer.apple.com/tutorials/swiftui/

https://bluewhaleapps.com/blog/uikit-vs-swiftui

https://designcode.io/swiftui-handbook

https://www.hackingwithswift.com/