
@p1llai

Build Skeleton Loading SwiftUI



@p1llai

1 Shimmer

```
struct Shimmer: ViewModifier {
    @State private var phase: CGFloat = -100

    func body(content: Content) -> some View {
        content
            .overlay(
                LinearGradient(
                    gradient: Gradient(colors: [Color.gray.opacity(0.3),
                                                Color.gray.opacity(0.1), Color.gray.opacity(0.3)]),
                    startPoint: .leading,
                    endPoint: .trailing
                )
                .rotationEffect(.degrees(30))
                .offset(x: phase)
                .mask(content)
            )
            .onAppear {
                withAnimation(.linear(duration: 1).repeatForever(autoreverses: false)) {
                    phase = 200
                }
            }
    }
}

extension View {
    func shimmer() -> some View {
        modifier(Shimmer())
    }
}
```

Add a shimmer effect using a LinearGradient with animation to simulate movement.

@p1llai

2 SkeletonRowView

```
struct SkeletonRow: View {
  var body: some View {
    HStack(spacing: 12) {
      RoundedRectangle(cornerRadius: 8)
        .fill(Color.gray.opacity(0.3))
        .frame(width: 50, height: 50)
        .shimmer()

      VStack(alignment: .leading, spacing: 8) {
        RoundedRectangle(cornerRadius: 4)
          .fill(Color.gray.opacity(0.3))
          .frame(height: 20)
          .shimmer()

        RoundedRectangle(cornerRadius: 4)
          .fill(Color.gray.opacity(0.3))
          .frame(height: 16)
          .frame(maxWidth: 200)
          .shimmer()
      }
    }
    .padding(.vertical, 8)
  }
}
```

Create a view that mimics our actual content layout but with gray rectangles and the shimmer effect applied.

@p1llai

3 Actual Content Row

```
struct ItemRow: View {
    let item: Item

    var body: some View {
        HStack(spacing: 12) {
            Image(systemName: item.iconName)
                .font(.title2)
                .foregroundColor(.blue)
                .frame(width: 50, height: 50)
                .background(Color.blue.opacity(0.1))
                .cornerRadius(8)

            VStack(alignment: .leading, spacing: 4) {
                Text(item.title)
                    .font(.headline)

                Text(item.subtitle)
                    .font(.subheadline)
                    .foregroundColor(.secondary)
            }
        }
        .padding(.vertical, 8)
    }
}
```

Define the view that shows your real data when loaded

@p1llai

4 Toggle Views

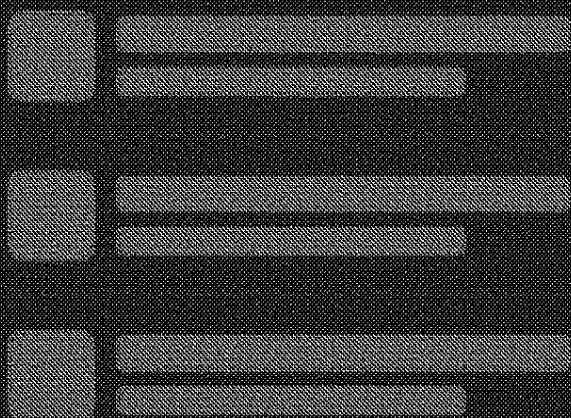
```
struct ParallaxView: View {
    @State private var isLoading = true

    var body: some View {
        NavigationView {
            List {
                // Show 6 placeholders while loading
                if isLoading {
                    ForEach(0..<6) { _ in
                        SkeletonRow()
                    }
                } else {
                    // Show real content after loading
                    ForEach(sampleData) { item in
                        ItemRow(item: item)
                    }
                }
            }
        }
        .navigationTitle("Siddharth Pillai")
        .onAppear {
            // Simulate loading delay
            DispatchQueue.main.asyncAfter(deadline: .now() + 2) {
                withAnimation {
                    isLoading = false
                }
            }
        }
    }
}
```

Build the main view with a `@State` flag to toggle between showing shimmering skeleton rows while loading and real item rows after a simulated delay, all wrapped in a `NavigationView` with a title.

@p1llai

Siddharth Pillai



Siddharth Pillai



SwiftUI Basics

Learn fundamentals



Animations

Make UI lively



Networking

Fetch remote data

@p1llai

Skeleton Loading & Shimmering Effect for your App

And that's it! 🎉 Follow these steps,
and you'll have skeleton loading with
shimmering effect working seamlessly
in your iOS app. Let me know if you
have any questions!

SHARE
WITH FRIENDS

SAVE
FOR LATER

LIKE &
FOLLOW