Swift UI

ANIMATIONS

Basics

DANS SWIFTUI, LE TYPE D'ANIMATION LE PLUS SIMPLE EST IMPLICITE : NOUS EXPRIMONS NOTRE POINT DE VUE À L'AVANCE

Basics

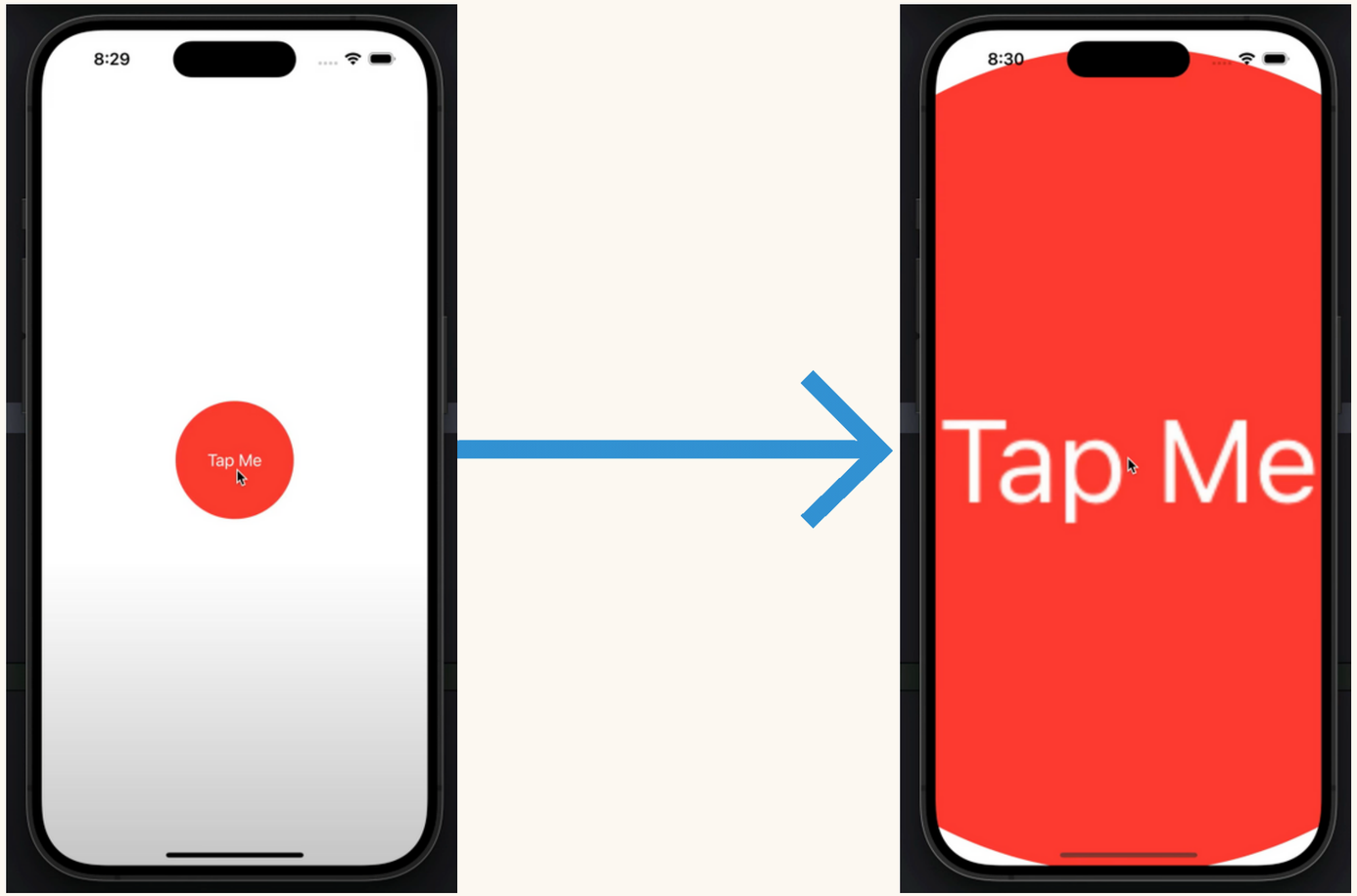
Commençons par ajouter la transformation sur l'objet que nous voulons animer

```swift
//
//  ContentView.swift
//  Animations
//
//  Created by Paul Hudson on 15/10/2023.
//

import SwiftUI

struct ContentView: View {
    @State private var animationAmount = 1.0

    var body: some View {
        Button("Tap Me") {
            animationAmount += 1
        }
        .padding(50)
        .background(.red)
        .foregroundStyle(.white)
        .clipShape(.circle)
        .scaleEffect(animationAmount)
    }
}
```



Ajoutons maintenant l'animation (et du Blur)

```swift
import SwiftUI

struct ContentView: View {
    @State private var animationAmount = 1.0

    var body: some View {
        Button("Tap Me") {
            animationAmount += 1
        }
        .padding(50)
        .background(.red)
        .foregroundStyle(.white)
        .clipShape(.circle)
        .scaleEffect(animationAmount)
        .blur(radius: (animationAmount - 1) * 3)
        .animation(.default, value: animationAmount)
    }
}
```

En ajoutant ceci APRES les différents modifiés

```swift
        .blur(radius: (animationAmount - 1) * 3)
        .animation(.default, value: animationAmount)
```

La place des modifiers est important

Modifiers

NOUS POUVONS CONTRÔLER LE TYPE D'ANIMATION UTILISÉ EN TRANSMETTANT DIFFÉRENTES VALEURS AU MODIFICATEUR.

Modifiers

Par exemple modifions l'apparition de l'objet

```
.animation(
    .easeInOut(duration: 2)
        .delay(1),
    value: animationAmount
)
```

Lorsque nous disons .easeInOut(duration: 2), nous créons en fait une instance d'une structure Animation qui possède son propre ensemble de modificateurs. Nous pouvons donc attacher des modificateurs directement à l'animation pour ajouter un délai comme celui-ci.

```
.animation(
    .easeInOut(duration: 1)
        .repeatCount(3, autoreverses: true),
    value: animationAmount
)
```

Nous pouvons également demander à l'animation de se répéter un certain nombre de fois, et même la faire rebondir en avant et en arrière en définissant l'autoreverse sur true. Cela crée une animation d'une seconde qui rebondira de haut en bas avant d'atteindre sa taille finale :


Explicit
DEMANDER EXPLICITEMENT À SWIFTUI D'ANIMER LES CHANGEMENTS SURVENANT À LA SUITE D'UN CHANGEMENT D'ÉTAT.

Explicit

Par exemple ajoutons une animation de rotation au clic

```swift
import SwiftUI

struct ContentView: View {
    @State private var animationAmount = 0.0

    var body: some View {
        Button("Tap Me") {
            withAnimation {
                animationAmount += 360
            }
        }
        .padding(50)
        .background(.red)
        .foregroundStyle(.white)
        .clipShape(.circle)
        .rotation3DEffect(.degrees(animationAmount))
    }
}

#Preview {
    ContentView()
}
```

Maintenant, cependant, nous expliquons explicitement que nous voulons qu'une animation se produise lorsqu'un changement d'état arbitraire se produit : elle n'est pas attachée à une liaison, ni à une vue, c'est simplement nous qui demandons explicitement qu'une animation particulière se produise parce que d'un changement d'état.