Animations SwiftUI

Fonctionnement

- Une animation entraine une recomposition de la vue, elle peut donc se recomposer plus de 60 fois par secondes.
- Il y a des éléments animables
- Dès lors que l’UI détecte un changement au niveau d’une variable d’animation, la vue se recompose pour l’effectuer.

Exemple d’utilisation sur un scale simple

- Pour un scale simple par exemple, il suffit de déclarer une variable booléenne isAnimated, puis on déclare cette variable en temps que variable d’animation grace a l’extention .animation(animation value:isAnimated). Dès lors que isAnimated change, alors la vue se recomposera pour effectuer l’animation.
- Text(« mon texte »)
- .scaleEffect(isAnimated ? 1.5:0)
- .animation(.boucy.speed(0.8), value:isAnimated)

Activation de l’animation

- Il faut en suite pouvoir faire changer la valeur de isAnimated au moment ou l’on veut effectuer l’animation. On peut utiliser plusieurs solutions ex:
- .onAppear{
  - isAnimated = true
- }
- Ou
- .task{ isAnimated = true
- } « ce code vient a la suite de .animtation() »

C’est tout !

- On peut ensuite anier tout ce que l’on souhaite assez facilement a l’aide d’une seule variable, si on souhaite ajouter des delais, ceux-ci sont disponibles au niveau du chois de l’animation ex:
- .boucy.delay(0.2) va attendre 0.2 sec avant de lancer l’animation
- Ou on peut effectuer un try await Task.sleep(nanoseconds: valeur) dans le corps du .task{}

Il y a déjà un bon nombre d’aniumations disponibles :

.spring .easeIntOut etc …