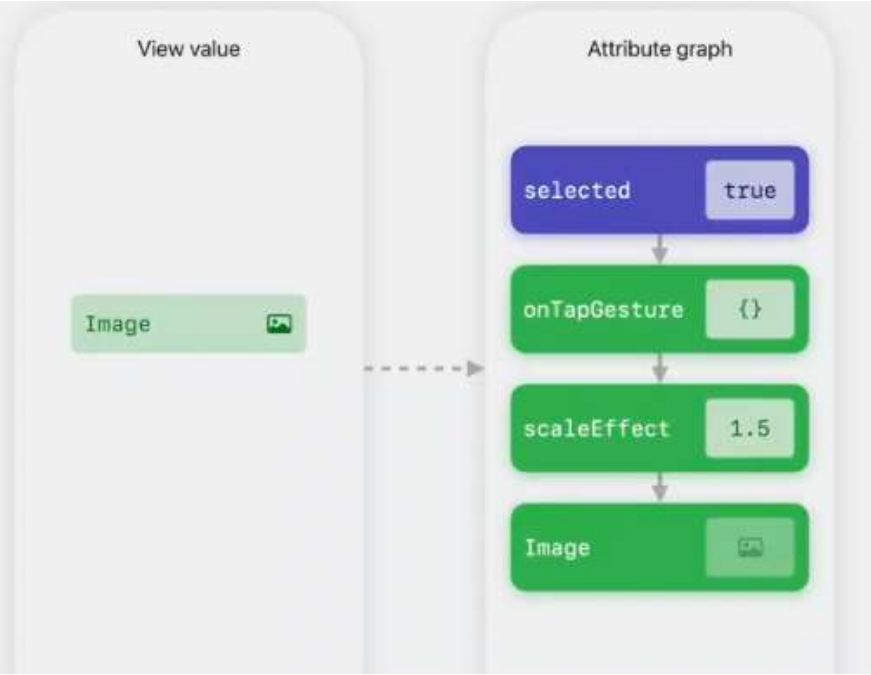
# SwiftUI Animations

How does it works ?

# View Update

- Pour avoir affaire aux animations, il faut savoir que la vue se refresh à chaque update de variable en passant pas les étapes:



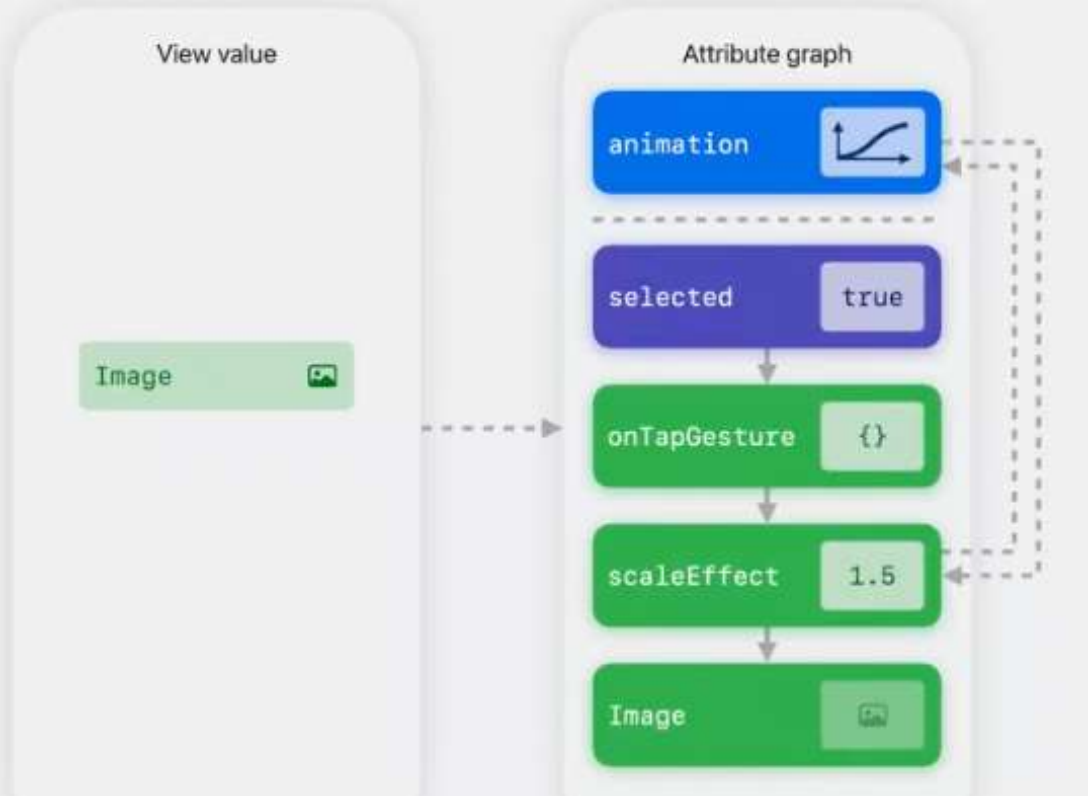

L'update de valeur en bleu
Et les étape de refresh en vert

- OnTapGesture
- ScaleEffect
- Image

C'est dans ces étape de refresh que l'animation va venir s'introduire

# Animation

- Si une animation est spécifié sur le composant, elle va remplacé l'attribut « ScaleEffect » du refresh par elle-même




Scale effect est un attribut de base pour l'animation des composant, l'animation est donc pris en compte dès la composition des composant dans l'UI avec cette attribut. Spécifier une animation signifie donc de remplacer scale effect par l'animation en question

Nous pouvons donc créer des animations, pour cela, il faudra spécifier un animatable et une animation

# Animatable

- Ce sont les données d’une animation
  - La durée
  - La courbe/vecteur que parcours le composant ciblé
- Tout simplement comment le composant doit se comporter quand il est animé

# Animation

- Et donc les animations s’occupe d’animer le composant, c’est lui qui gère frame par frame l’animation en étroite collaboration avec l’animation table, les données permettant de l’éxecuter

# Cas d’utilisations

Le cas des animations

- Nous spécifions l’attribut .scaleEffect différemment de celui de base avec une nouvelle condition: isAnimated, c’est L’animatable
- Enfin, nous spécifions l’animation « .animation » avec « .smooth » et un delay de 4 milliseconds. Puis la condition pour que l’animation remplace le scaleEffect: « isAnimated »