

# *architecture de l'application*

*Développez Instagrid :  
une application de montage photo*

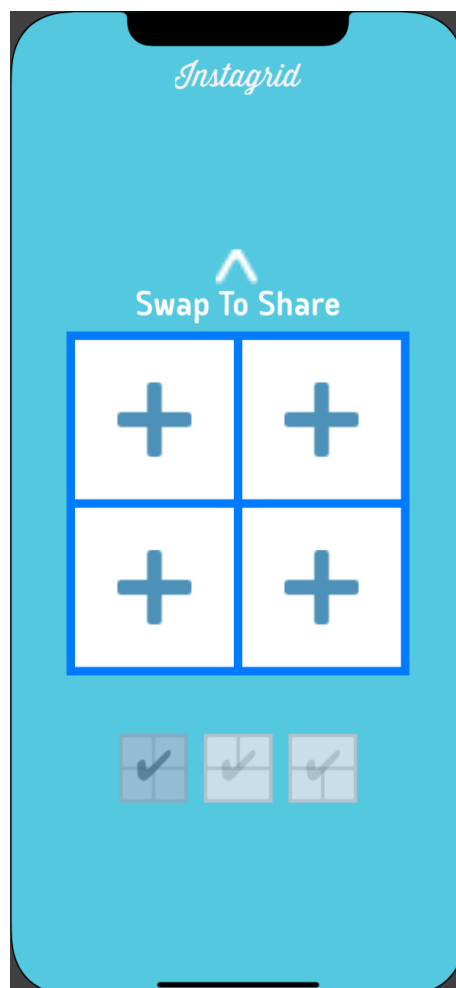
En utilisant Xcode et Swift



# Aperçu

Cette application est le quatrième projet de la parcours “ Développeur d'application - iOS ”, confié par [OPENCLASSROOMS](https://openclassrooms.com).

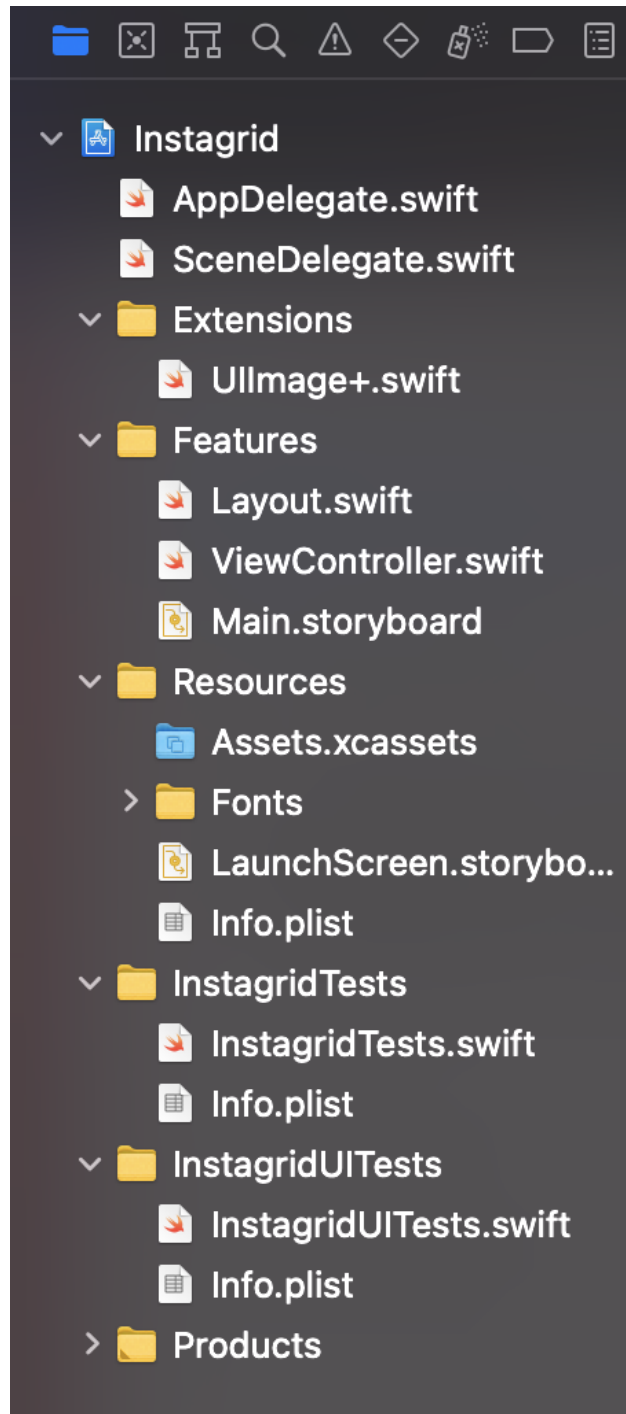
Cette application usé par un utilisateur pour créer un photo composé de trois ou quatre image, à la choix, le choisir a lieu à partir de la galerie ou de la caméra.



# Détails

- Le projet est conformé à la technique MVC.

On peut voir le composants de projet comme suivante:



Les fichiers qui ont été ajoutés sont:

- UIImage+ : dans ce fichier il y a une extension pour ajouter une fonctionnalité à la classe UIImage.  
Cette fonctionnalité est utilisée pour convertir un view en photo utilisable à conserver ou à partager.
  - Layout : dans ce fichier il y a un Enum dedans le layout disponible pour l'image résultante, ( trois ou quatre images).
  - Le dossier Font: dans lequel on a ajouté deux polices.
  - Les images utilisées dans la structure de l'application sont ajoutées dans le dossier (Assets.xcassets).
  - Info.plist: est déjà existant, mais on ajoute des permissions pour permettre à l'utilisateur d' accéder à la galerie ou à la fonctionnalité de caméra, ou pour sauvegarder la photo résultante dans la galerie.
- 
- StoryBoard: les composants ajoutés à la storyboard, sont:
    - 1- UILabel (Instagrid), pour le titre : (*Instagrid* ) en haut de l'application.
    - 2- UIImage (Arrow Up), pour la flèche : ( ^ ).
    - 3- UILabel (Swap To Share), pour le mot : ( swap to share).
    - 4- UIView (ViewMain): c'est le principal view qui va convertir l'image, et qui va bouger vers le haut ou vers la gauche, quand l'utilisateur glisse l'écran.

- UIStackview (Stack ViewFull):

1- UIStackview (Stack ViewUp):

-UIView (View).

-UIImage (Plus): c'est la flèche ( + ).

-UIButton (Image1): c'est la première photo qui va être ajoutée.

-UIView (View2): cette view va être casher, quand l'utilisateur veut une photo en haut.

-UIImage (Plus): c'est la flèche ( + ).

-UIButton (Image2): c'est la deuxième photo qui va être ajoutée.

2- UIStackview (Stack ViewDown):

-UIView(View).

-UIImage (Plus): c'est la flèche ( + ).

-UIButton (Image3): c'est la troisième photo qui va être ajoutée.

-UIView (View4): cette view va être casher, quand l'utilisateur veut une photo en bas.

-UIImage (Plus): c'est la flèche ( + ).

-UIButton (Image4): c'est la quatrième photo qui va être ajoutée.

5- UIStackView (Stack ViewButton):

1-UIButton (Select All): en l'appuyant, on choisit le layout de 4 photos.

2-UIButton (Select Up): en l'appuyant, on choisit le layout de 3 photos, un en haut.

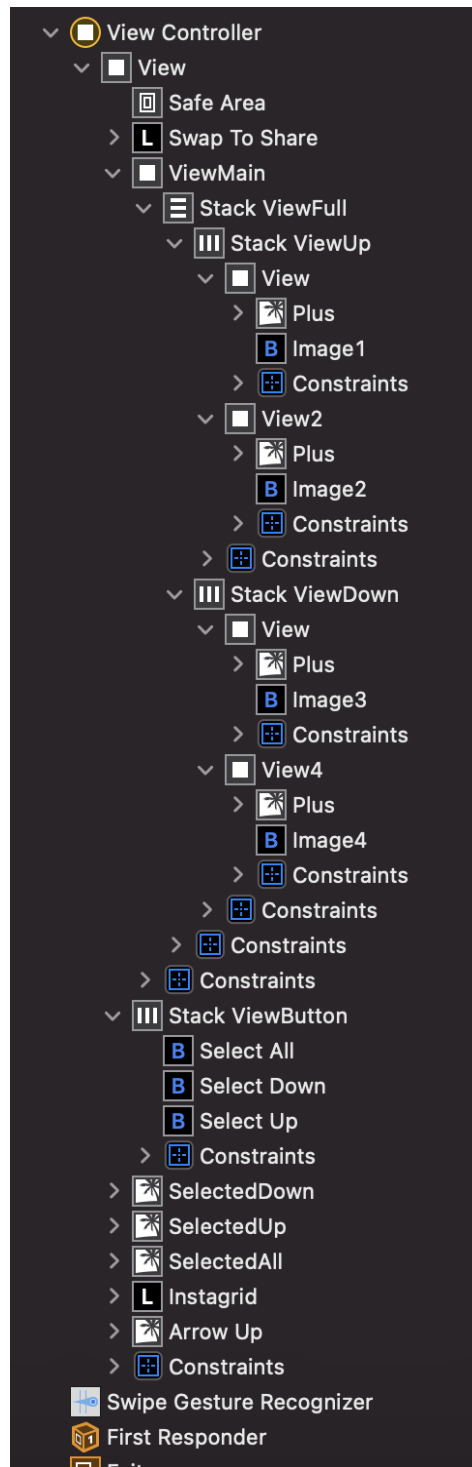
3-UIButton (Select Down): en l'appuyant, on choisit le layout de 3 photos, un en bas.

6- UIImage (SelectedAll): ces sont la flèche ( √ ), une d'elles va être apparue, et les deux vont être casher.

7-UIImage (SelectedUp): ces sont la flèche ( √ ), une d'elles va être apparue, et les deux vont être casher.

8- UIImage (SelectedDown): ces sont la flèche ( ✓ ), une d'elles va être apparue, et les deux vont être casher.

9-Swipe Gesture Recognizer: qui nous permet de glisser l'écran de téléphone.



- Dans le fichier UIImage+: On a ajouté une fonctionnalité à la classe existante UIImage, c'est une fonctionnalité qui permet de convertir un UIView à un UIImage.

```
3 extension UIImage {
4     static func image(with view: UIView) -> UIImage? {
5         UIGraphicsBeginImageContextWithOptions(view.bounds.size, view.isOpaque, 1.0)
6         defer { UIGraphicsEndImageContext() }
7         if let context = UIGraphicsGetCurrentContext() {
8             view.layer.render(in: context)
9             let image = UIGraphicsGetImageFromCurrentImageContext()
10            return image
11        }
12        return nil
13    }
14 }
```

- Layout: dans ce fichier, on trouve Enum, ce sont les choix possibles de l'image résultante.

```
9
10 enum Layout {
11     case fourSquare, rectangleDown, rectangleUp
12 }
13
```

- ViewController: dans ce fichier, on trouve toutes les pièces du logiciel:

#### 1- IBOutlet:

```
12
○ @IBOutlet weak var mainView: UIView!
○ @IBOutlet weak var image1: UIButton!
○ @IBOutlet weak var image2: UIButton!
○ @IBOutlet weak var image3: UIButton!
○ @IBOutlet weak var image4: UIButton!
○ @IBOutlet weak var selectAllImage: UIImageView!
○ @IBOutlet weak var selectUpImage: UIImageView!
○ @IBOutlet weak var selectDownImage: UIImageView!
21
○ @IBOutlet weak var view2: UIView!
○ @IBOutlet weak var view4: UIView!
○ @IBOutlet weak var selectAll: UIButton!
○ @IBOutlet weak var selectDown: UIButton!
○ @IBOutlet weak var selectUp: UIButton!
27
```

Ces les connexions entre les composants et le controller.

2- Propriétés: ces Propriétés vont nous aider à compléter les missions.

3- Les fonctions et les méthodes qui font les missions demandées.

- viewDidLoad (): est la méthode qui est appelée une fois que le MainView d'un ViewController a été chargé.
- viewWillTransition(): cette méthode va être appelée quand le téléphone change ses orientations.  
En notre application, on utilise cette méthode pour changer le paramètre qui bouge le principe view ( en haut ou à gauche).
- selectAllRect(): avec ce fonction, on choisi le layout qui consiste de 4 images.




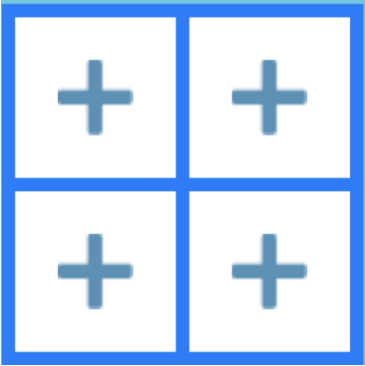

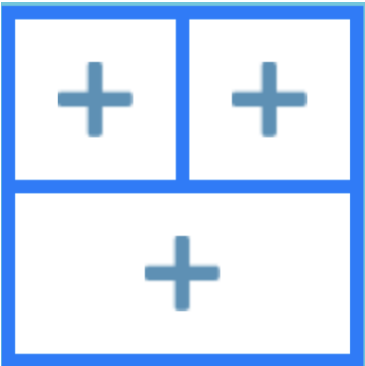

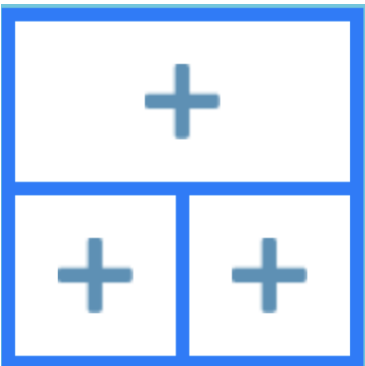

- selectUpRect(): avec cette fonction, on choisit le layout qui consiste de 3 images, une image en haut.
- selectDownRect(): avec cette fonction, on choisit le layout qui consiste de 3 images, une image en bas.
- setUpmainView(): cette fonction appelée à partir de les fonctions précédentes, cette fonction applique le layout.
- selectImage(), avec cette méthode on sait quelle bouton est appuyé, puis on appelle la méthode showAlert() qui ajoute l'image à le bouton.

Dès qu' on glisse l'écran, l'utile UISwipeGestureRecognizer, va appeler la méthode shareImage().

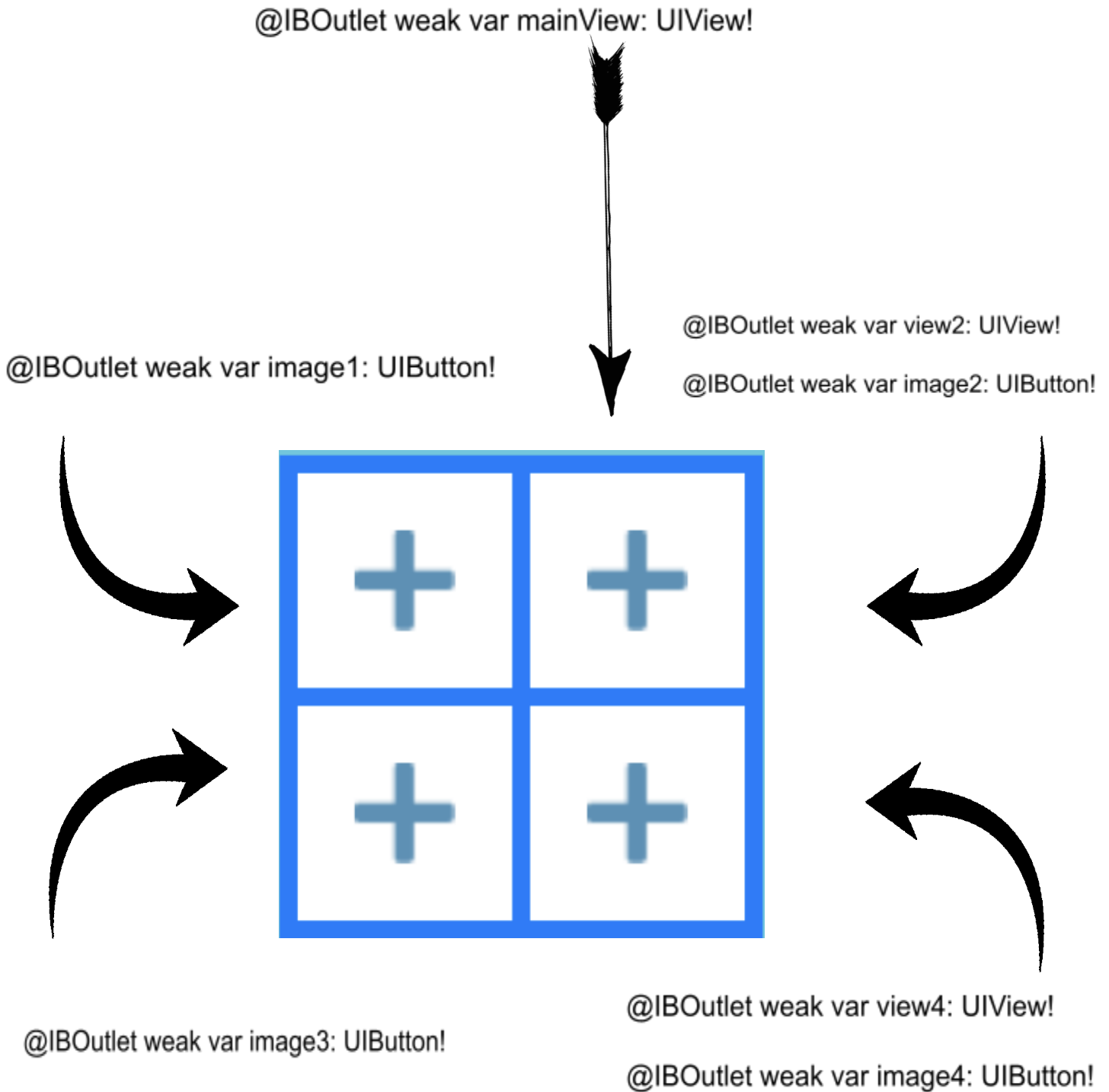
- shareImage(): cette méthode appelle la méthode moveForward(), puis la methode image(), puis la méthode moveBack().
- moveForward(): pour bouger la principe view à la direction convient à l'orientation de téléphone.
- moveBack(): pour retourner la principe view à sa propre place
- l'outil UISwipeGestureRecognizer celui-ci qui rattrape le glisser de l'écran.

# Le Layout correspondante pour chaque interaction d'utilisateur

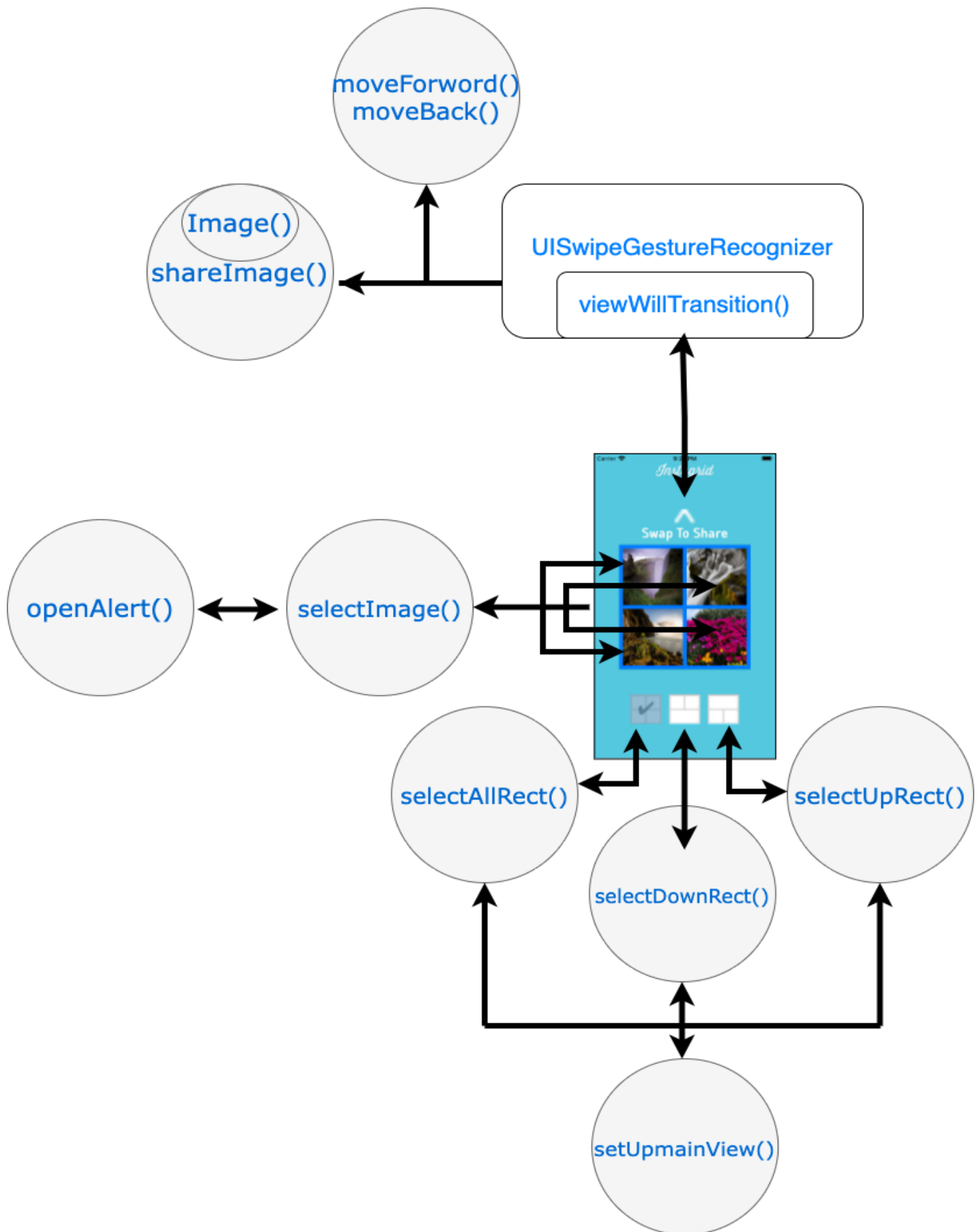
## Et l'IBOutlet correspondante.

 <p>@IBOutlet weak var selectAll: UIButton!</p>	
 <p>@IBOutlet weak var selectDown: UIButton!</p>	
 <p>@IBOutlet weak var selectUp: UIButton!</p>	
	<p>@IBOutlet weak var selectAllImage: UIImageView!</p> <p>@IBOutlet weak var selectUpImage: UIImageView!</p> <p>@IBOutlet weak var selectDownImage: UIImageView!</p>

# IBOutlet correspondant pour chaque éléments



# Le schéma général de l'architecture des fonctions et méthodes d'application



M E C  
R I