Navigation et cycles de vie des vues

Antoine Moevus

Table des matières

Références

- https://www.kodeco.com/5055396-ios-storyboards-segues-and-more
- https://stackoverflow.com/a/25967267/1076483

Unwind segue

- https://developer.apple.com/documentation/uikit/resource_management/dismissing_a_view_controller_with_an_unwind_segue
- https://www.youtube.com/watch?v=akmPXZ4hDuU

Introduction à la navigation en iOS

L'objectif de cette section est de comprendre le concept de navigation en iOS et comment elle est gérée. Nous aborderons notamment le contrôleur de navigation, la pile de navigation et comment naviguer d'avant en arrière dans cette pile.

Comprendre le contrôleur de navigation

Un contrôleur de navigation est un contrôleur de vue qui gère la navigation entre les différentes vues de votre application. Chaque vue correspond à un contrôleur de vue différent, et le contrôleur de navigation agit comme un gestionnaire de ces contrôleurs de vue, permettant à l'utilisateur de naviguer entre eux.

Il est important de comprendre que dans le modèle MVC (Model-View-Controller) utilisé par iOS, les vues représentent ce que l'utilisateur voit à l'écran, tandis que les contrôleurs de vue gèrent la logique qui fait le lien entre les vues et les données qui doivent y être affichées.

La gestion correcte des contrôleurs de navigation est une partie essentielle de la création d'une application iOS fluide et intuitive.

Voici un résumé des fichiers utilisés pour la navigation:

Tableau 1. Les rôles des fichiers dans la navigation

Concept	Description	Extension de fichier
Vue	Les vues sont les composants de l'interface utilisateur de votre application. Elles représentent tout ce que l'utilisateur voit à l'écran, y compris les boutons, les labels, les images et autres éléments graphiques.	.xib, .storyboard
Contrôleur	Un contrôleur (ou contrôleur de vue) est une partie du modèle MVC qui gère la logique de liaison entre les vues et les données qui doivent y être affichées. Il gère également les interactions de l'utilisateur avec les vues.	.swift

Comprendre les concepts de pile de navigation

Le contrôleur de navigation gère les contrôleurs de vue de votre application en utilisant une structure de données appelée "pile". Cette pile de navigation est simplement une liste ordonnée de contrôleurs de vue.

Le premier contrôleur de vue ajouté à la pile est appelé le contrôleur de vue "racine", et il est généralement le premier écran que voit l'utilisateur. Chaque fois qu'un nouvel écran est présenté, le contrôleur de vue correspondant est "empilé" sur les précédents. Chaque contrôleur de vue sait quel contrôleur de vue l'a présenté, ce qui permet de naviguer en arrière à travers la pile. Nous allons détaillé cela à la prochaine section.

Comment naviguer en avant et en arrière dans la pile de navigation

La navigation en avant dans la pile de navigation est réalisée en présentant un nouveau contrôleur de vue, généralement en réponse à une action de l'utilisateur, comme appuyer sur un bouton.

Pour naviguer en arrière, on "dépile" le contrôleur de vue en haut de la pile, soit en appuyant sur le bouton "Retour" de la barre de navigation, soit programmatiquement en utilisant la méthode `popViewController(animated:)`.

Il est important de noter que lorsque vous dépilez un contrôleur de vue, celui-ci est détruit et supprimé de la mémoire. Par conséquent, toute information qu'il contient sera perdue à moins d'avoir été sauvegardée ailleurs.

Une bonne pratique consiste à maintenir la pile de navigation aussi petite que possible. Cela rend la navigation plus facile pour l'utilisateur et préserve les ressources de l'appareil.

Les contrôleurs de vue et leur cycle de vie

Chaque contrôleur de vue dans une application iOS passe par une série d'états pendant sa durée de vie, depuis son chargement jusqu'à son élimination de la mémoire. Ces étapes constituent le cycle de vie du contrôleur de vue. Comprendre ces étapes est essentiel pour configurer correctement votre contrôleur de vue et gérer les transitions entre les vues.

Les différentes étapes du cycle de vie d'un contrôleur de vue

Le tableau ci-dessous résume les fonctions du controlleur qui permettent de gérer la vue qui lui est assigné.

Fonction du cycle de vie	Description
`viewDidLoad()`	Cette méthode est appelée après que le contrôleur de vue a chargé sa vue dans la mémoire. Habituellement, vous utilisez cette méthode pour initialiser votre contrôleur de vue, par exemple pour configurer les vues et les données source.
`viewWillAppear()`	Cette méthode est appelée juste avant que la vue du contrôleur de vue ne soit ajoutée à la hiérarchie des vues de l'application. Vous pouvez effectuer des mises à jour supplémentaires de l'interface utilisateur dans cette méthode.
`viewDidAppear()`	Cette méthode est appelée après que la vue du contrôleur de vue a été ajoutée à la hiérarchie des vues de l'application. Vous pouvez utiliser cette méthode pour démarrer des animations ou charger des données qui doivent être affichées à l'utilisateur.
`viewWillDisappear()`	Cette méthode est appelée lorsque la vue du contrôleur de vue est sur le point d'être retirée de la hiérarchie des vues de l'application.
`viewDidDisappear()`	Cette méthode est appelée lorsque la vue du contrôleur de vue vient d'être retirée de la hiérarchie des vues de l'application.

Comment le cycle de vie affecte la configuration d'un contrôleur de vue

Le cycle de vie d'un contrôleur de vue affecte sa configuration en déterminant quand et où certaines tâches devraient être réalisées. Par exemple, vous ne devriez pas essayer de mettre à jour l'interface utilisateur dans la méthode `viewDidLoad()`, car la vue du contrôleur de vue pourrait ne pas être entièrement configurée à ce moment-là.

Au contraire, vous devriez effectuer la majorité de la configuration de l'interface utilisateur dans `viewWillAppear()` ou `viewDidAppear()`. De même, si vous avez besoin de sauvegarder des données ou de réaliser des tâches de nettoyage avant que le contrôleur de vue ne disparaisse, vous devriez le

faire dans `viewWillDisappear()`.

Navigation controller et `segue`

Navigation Controller

Le `Navigation Controller` est un composant clé dans la gestion de la navigation entre les vues dans une application iOS. Il agit comme un gestionnaire de pile pour les différentes vues (ou écrans) de l'application. Chaque fois qu'une nouvelle vue est poussée sur la pile, elle devient la vue active et est affichée à l'utilisateur.

Quand vous ajoutez un `Navigation Controller` à votre storyboard, il vient avec une vue par défaut appelée `Root View Controller`. C'est le point de départ de la navigation dans votre application.

On trouve le `Navigation Controller` dans la librairie d'objets (le même endroit que les `label` et `button`).

Transition entre deux contrôleurs : les segues

Une `segue` (prononcé "seg-way") est un lien ou une transition entre deux contrôleurs de vue dans votre storyboard. Vous pouvez l'utiliser pour définir comment vous voulez passer d'une vue à une autre.

Pour créer une segue entre deux contrôleurs de vue dans le storyboard, Ctrl-glissez de l'objet de départ (qui peut être un contrôleur de vue, un bouton, une cellule de tableau, etc.) vers le contrôleur de vue de destination.

Il existe différent types de segues dans UIKit :

Type de Segue	**Description**	**Exemple**
Show	Pousse le contrôleur de vue de destination sur la pile de navigation, glissant de droite à gauche, fournissant un bouton de retour pour revenir - s'il n'est pas intégré dans un contrôleur de navigation, il sera présenté de manière modale	Navigation dans les paramètres, par exemple en appuyant sur Général > À propos
Show Detail	Pour utilisation dans un contrôleur de vue partagée, remplace le contrôleur de vue secondaire lorsqu'il est dans une interface à plusieurs colonnes, ou s'il est réduit à une seule colonne, il poussera dans le contrôleur de navigation	Dans Messages, en appuyant sur une conversation, on affiche les détails de la conversation - remplaçant le contrôleur de vue sur la droite lorsqu'il est dans une disposition à deux colonnes, ou poussant la conversation lorsqu'il est dans une disposition à une seule colonne

Type de Segue	Description	Exemple
Present Modally	Présente un contrôleur de vue au-dessus du contrôleur de vue actuel de différentes manières, comme défini par le style de présentation et de transition modale - le plus souvent utilisé pour présenter un contrôleur de vue dans une feuille qui anime du bas vers le haut	Sélection de Face ID & Passcode dans les paramètres
Popover Presentation	Lorsqu'exécuté sur iPad, la destination apparaît dans une fenêtre contextuelle, et taper n'importe où à l'extérieur la fermera - les fenêtres contextuelles sont également prises en charge sur iPhone, mais par défaut il présentera le contrôleur de vue de manière modale	En appuyant sur le bouton + dans le Calendrier
Custom	Vous pouvez implémenter votre propre segue personnalisée et contrôler son comportement	
Embed	Vous pouvez intégrer un contrôleur de vue dans un autre contrôleur de vue, comme les contrôleurs de navigation, de barre d'onglets, et de vue partagée ainsi que les conteneurs personnalisés	
Unwind	Vous pouvez utiliser une segue de déroulement pour revenir à un contrôleur de vue précédent, même s'il y a de nombreux écrans poussés/présentés par dessus, tous seront supprimés	

NOTE	Les segues obsolètes sont essentiellement les équivalents non adaptatifs de ceux décrits ci-dessus. Ces types de segues ont été dépréciés dans iOS 8 : Push, Modal, Popover, Replace.

Pour plus d'informations, vous pouvez consulter la documentation "Using Segues" qui explique également les types de segues et comment les utiliser dans un Storyboard. Consultez également la Session 216 "Building Adaptive Apps with UIKit" de la WWDC 2014. Ils ont parlé de comment vous pouvez construire des applications adaptatives en utilisant ces nouvelles Segues Adaptatives, et ils ont construit un projet de démonstration qui utilise ces segues.

Exemple récapitulatif

Mise en place du projet

https://vimeo.com/851435132/027e1a810d

Ajout d'une page avec un Segue

https://vimeo.com/851439356/b6e95fb90e

Modification de la barre de navigation

https://vimeo.com/851441749/dada13f367?share=copy

Ajouter un controlleur et analyse des cycles de vie

https://vimeo.com/851447540/ab431e8bce

Utilisation de Cocoa Touch Class

`Cocoa Touch Class`` est un moyen rapide de créer une nouvelle classe qui hérite d'une certaine classe de base Cocoa Touch, alors qu'un fichier Swift (`.swift`) vous donne un fichier vierge dans lequel vous pouvez écrire n'importe quel code Swift.

Lorsque vous créez une nouvelle Cocoa Touch Class dans Xcode, vous créez en réalité une nouvelle classe Swift (ou Objective-C, selon les préférences du projet) qui hérite d'une classe de la bibliothèque de frameworks Cocoa Touch d'Apple. Cocoa Touch est l'ensemble de frameworks UI pour le développement d'applications iOS, donc lorsque vous créez une nouvelle Cocoa Touch Class, vous avez généralement l'intention de personnaliser une certaine partie de l'interface utilisateur de votre application. Par exemple, vous pouvez créer une Cocoa Touch Class pour définir un contrôleur de vue personnalisé, une cellule de tableau personnalisée, etc.

Cocoa Touch est le framework de plus haut niveau pour le développement d'applications iOS. Il encapsule les frameworks UIKit, Foundation et Core Graphics, ainsi que d'autres bibliothèques essentielles pour le développement d'applications iOS. Il comprend des fonctionnalités liées à l'interface utilisateur multi-touch, à la prise en charge des événements d'entrée, à la gestion des vues, aux contrôleurs de vues, au multitâche, aux notifications push, etc. De plus, il fournit des abstractions pour les interactions de bas niveau avec le système, telles que la gestion des fichiers et des threads, l'accès au réseau et les fonctionnalités du système d'exploitation.

UIKit, quant à lui, est un sous-ensemble de Cocoa Touch et est spécifiquement conçu pour aider les développeurs à construire et à gérer l'interface utilisateur d'une application iOS. Il comprend un grand nombre de classes pour gérer les animations, les vues et les contrôleurs de vues, les événements tactiles, la gestion du clavier, la navigation et le contrôle, ainsi que l'affichage de texte, d'images et de PDF.

En résumé, Cocoa Touch est le framework global pour le développement d'applications iOS, et UIKit

est un composant de celui-ci, centré sur la construction et la gestion des interfaces utilisateur. À partir de iOS 13, Apple a introduit un nouveau framework pour construire des interfaces utilisateur, SwiftUI, qui est destiné à remplacer UIKit à l'avenir, mais UIKit reste très largement utilisé et est toujours activement maintenu par Apple.

Étapes pour ajouter un controlleur

Voici les étapes pour lier un controlleur à une vue:

1. Dans Xcode, allez dans `File > New > File…` ou utilisez le raccourci `Command+N`.
2. Sélectionnez `Cocoa Touch Class`, qui se trouve sous `iOS > Source`, puis cliquez sur `Next`.
3. Nommez votre classe (par exemple, `MyViewController`), assurez-vous de sélectionner `UIViewController` dans le menu déroulant "Subclass of" et assurez-vous que `Swift` est sélectionné comme langage. Cliquez sur `Next`.
4. Choisissez l'emplacement où vous voulez sauvegarder votre fichier et cliquez sur `Create`. Vous devriez maintenant avoir un nouveau fichier `MyViewController.swift` dans votre navigateur de projet.
5. Ouvrez maintenant votre fichier Storyboard (généralement `Main.storyboard`).
6. Dans le volet de gauche, vous verrez une liste de scènes dans votre storyboard. Si vous avez ajouté une nouvelle vue, vous devriez voir quelque chose qui ressemble à "View Controller Scene". Cliquez sur le petit triangle à côté pour dérouler cette scène.
7. Cliquez sur l'icône qui ressemble à un petit cube (c'est l'icône du contrôleur de vue). Il se trouve généralement en haut de la hiérarchie des éléments de la scène, sous la barre de navigation si elle est présente.
8. Une fois que le contrôleur de vue est sélectionné, dirigez-vous vers le panneau d'inspecteur de l'identité sur la droite. C'est le troisième bouton à partir de la droite.
9. Dans l'inspecteur d'identité, sous la section "Custom Class", il y a un champ appelé "Class". Ici, vous pouvez saisir le nom du contrôleur de vue que vous avez créé précédemment (dans cet exemple, `MyViewController`). Assurez-vous de bien orthographier le nom, car il est sensible à la casse.
10. Appuyez sur la touche Enter et Xcode devrait automatiquement compléter le reste du nom pour vous s'il reconnaît la classe. Si ce n'est pas le cas, vérifiez l'orthographe du nom de votre classe.
11. Maintenant, votre vue est liée à son contrôleur. Vous pouvez commencer à ajouter des éléments d'interface utilisateur à votre vue et à les relier à votre classe de contrôleur de vue à l'aide de sorties (IBOutlet) et d'actions (IBAction).

NOTE	N'oubliez pas que le contrôleur de vue que vous attribuez à la vue doit hériter de `UIViewController` ou d'une de ses sous-classes.

Ajouter un `unwind segue`

Démonstration: lien

Qu'est-ce qu'un `unwind segue` ?

Un `unwind segue` est utilisé pour revenir en arrière dans votre séquence de contrôleurs de vue. Cela permet à l'utilisateur de retourner à une vue précédente, tout en fournissant une manière structurée de passer des données entre les contrôleurs.

Création d'une méthode `unwind`

Dans le contrôleur de vue de **destination** (l'endroit où vous voulez revenir), définissez une fonction `unwind action` avec une signature spécifique :

```
@IBAction func votreUnwindAction(unwindSegue: UIStoryboardSegue) {
    // Code à exécuter après le retour à cette vue
}
```

Notez l'annotation `@IBAction` et que la méthode prend un argument de type `UIStoryboardSegue`.

Configuration du `unwind segue` dans Interface Builder

1. Ouvrez votre Storyboard dans Interface Builder.
2. À partir du contrôleur de vue (ou d'un élément de ce contrôleur) d'où vous souhaitez déclencher le `unwind`, Ctrl + glisser vers l'icône "Exit" (sortie) en haut du contrôleur de vue dans le Storyboard.
3. Une liste des actions unwind disponibles s'affichera. Sélectionnez l'action unwind que vous avez définie précédemment pour créer le `unwind segue`.

Autres

1. **Passer des données lors de l'utilisation d'un `unwind segue`**

Si vous souhaitez passer des données entre le contrôleur de vue source et le contrôleur de vue de destination, surchargez la méthode `prepare(for:sender:)` :

```
override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
    if segue.identifier == "YourUnwindSegueIdentifier" {
        if let destinationVC = segue.destination as? YourDestinationViewControllerType
{
            // Configurez destinationVC comme nécessaire
        }
    }
}
```

Veillez à définir un identifiant pour votre `unwind segue` dans Interface Builder afin de pouvoir le vérifier dans la méthode `prepare(for:sender:)`.

1. **Déclencher le `unwind segue` programmation**

Bien que les `unwind segues` soient souvent déclenchés via des actions d'interface utilisateur dans Interface Builder, vous pouvez également les déclencher en code en utilisant `performSegue(withIdentifier:sender:)`.

```
self.performSegue(withIdentifier: "YourUnwindSegueIdentifier", sender: self)
```