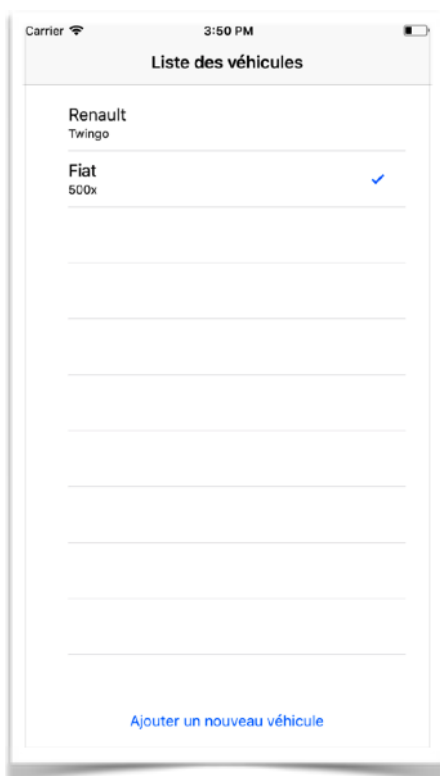


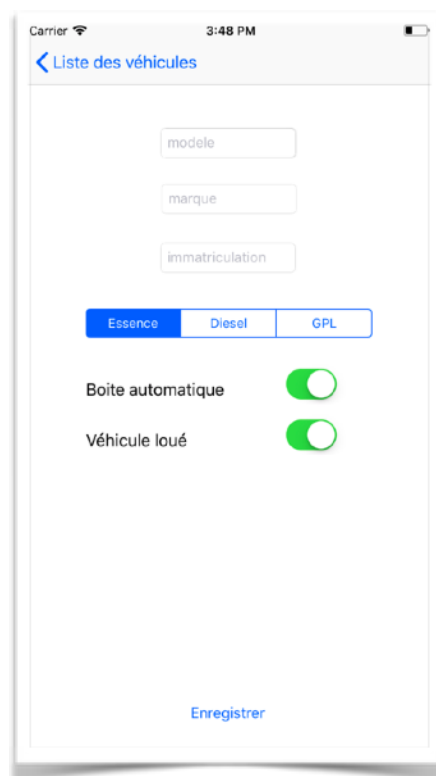
TP4 - Assembler plusieurs écrans

Objectif du TP

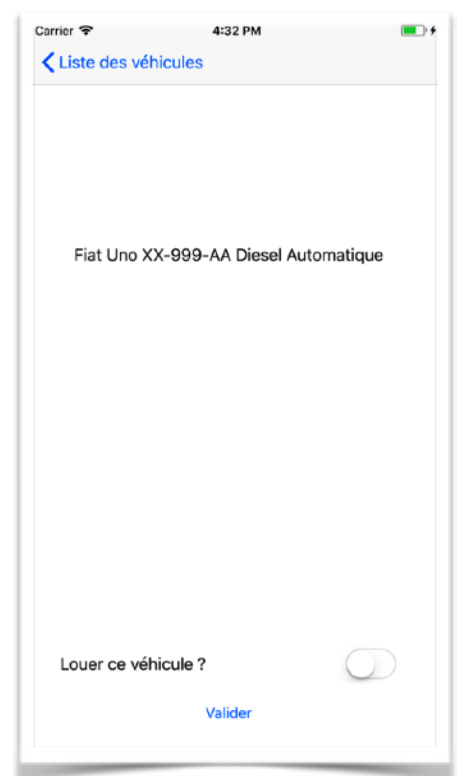
L'objectif du TP consiste manipuler plusieurs écrans dans une application et à transférer des données entre ces différents écrans. L'application que vous développerez au cours de ce TP consiste à compléter le parc automobile d'une agence de location. Le premier écran permet de lister tous les véhicules proposés par l'agence. Le deuxième écran, accessible depuis le bouton « Ajouter un nouveau véhicule », permet de saisir les informations relatives à un nouveau véhicule de la flotte. Finalement le troisième écran, accessible lorsque l'utilisateur sélectionné l'une des lignes de la table, permet de visualiser les informations d'un véhicule et, éventuellement, de le louer.



écran 1



écran 2



écran 3

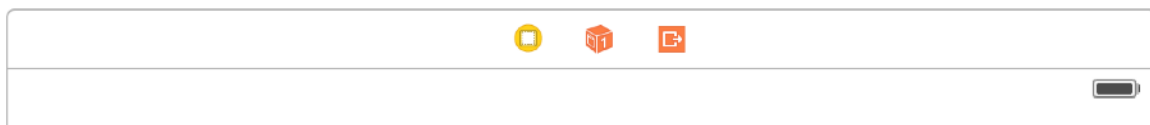
Partie I : Création de l'écran 2

Question 4 : Ajoutez à votre projet le fichier `NouvelleVoitureViewController.swift` qui va contenir le code décrivant le comportement du contrôleur d'une nouvelle vue (vérifier bien que ce nouveau fichier est une sous classes de la classe `UIViewController`).

Question 5 : Ajouter dans le fichier `Main.storyboard` un nouvel objet de type contrôleur de vue en prenant soin de mettre à jour la classe de votre nouveau contrôleur dans l'« Identity Inspector ».

Partie II : Navigation au sein de l'application

Question 6 : Pour relier les deux vues que vous avez créées, vous allez tout d'abord ajouter un contrôleur de navigation à votre première vue (celle correspondant à l'écran d'accueil de l'application). Pour cela, sélectionnez la vue en cliquant en haut de celle-ci dans le storyboard.



Puis dans le menu « Editor », cliquez sur « Embed in » puis sur « Navigation controller ».

Un « Navigation controller » doit apparaître dans votre storyboard et se placer devant votre vue.

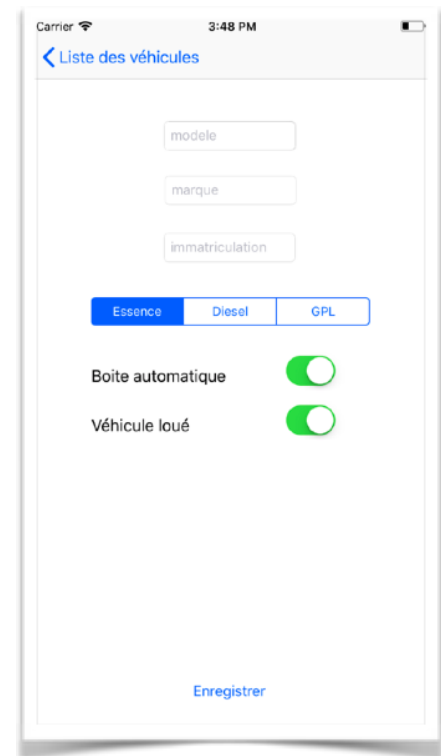
Question 7 : Mettez à jour le titre dans la barre de navigation qui vient d'être créée automatiquement en haut de votre première vue (par exemple : « Liste des véhicules ») en double-cliquant directement sur le titre dans le fichier `Main.Storyboard` ou en le modifiant dans l'onglet « Attributes inspector ».

Question 8 : Créez et identifiez ensuite la transition permettant d'aller de la vue principale de votre application à la vue permettant de saisir une nouvelle voiture lorsque vous cliquez sur le bouton « Ajouter un nouveau véhicule ».

Partie III : Implémentation de la nouvelle vue

Cette nouvelle vue va permettre à l'utilisateur de définir une nouvelle voiture de location en précisant ses différentes caractéristiques. Son interface graphique est composée des éléments suivants :

- ▶ trois objets de type `UITextField`,
- ▶ Un objet de type `UISegmentedControl` pour préciser le type de carburant (« Essence », « Diesel » ou « GPL »)
- ▶ Un objet de type `UISwitch` pour préciser si l'embrayage est une boîte automatique,
- ▶ Un objet de type `UISwitch` pour préciser si la voiture est louée ou non,
- ▶ un bouton « Enregistrer ».



Question 9 : Ajoutez les éléments d'interface graphique et définissez les `IBOutlet`s qui vous semblent nécessaires.

Question 10 : Créez une variable d'instance `voitureCourante` dans votre fichier `NouvelleVoitureViewController.swift`.

Question 11 : Implémentez la méthode `textFieldShouldReturn` du protocole `UITextFieldDelegate` pour que le bouton « Enregistrer » devienne actif uniquement si les trois champs de texte sont remplis (on ne fera pas de vérification du format de l'immatriculation de la voiture).

Partie IV : Passage de données entre les vues

Vous allez maintenant transférer les informations concernant la nouvelle voiture enregistrée, c'est-à-dire la variable d'instance `voitureCourante`, entre les deux vues que vous avez créées et gérer le retour vers la vue initiale lors de l'enregistrement d'une nouvelle voiture.

Ce transfert de données nécessite deux méthodes :

1. la méthode `prepare(segue: UIStoryboardSegue, sender: AnyObject?)` du fichier `NouvelleVoitureViewController.swift` qui sera exécutée lors du passage de la vue permettant d'enregistrer une nouvelle voiture vers la vue initiale (cf. Question 12),
2. la méthode `@IBAction func miseAJourTableView(segue : UIStoryboardSegue)` définie dans le fichier `ViewController.swift` qui sera appelée lors du retour vers la vue initiale après avoir sélectionné le bouton « Enregistrer » dans la seconde vue.

Question 12 : Implémentez la méthode `prepare` dans le fichier `NouvelleVoitureViewController.swift`. Cette méthode doit effectuer les trois instructions suivantes :

- a. tester la valeur de l'identifiant du segue,
- b. récupérer le contrôleur destination du segue dans une variable de type `ViewController`,
- c. mettre à jour la variable d'instance `voitureCourante` puis l'ajouter à la variable d'instance `data` du `viewController`.

Question 13 : Implémentez la méthode `miseAJourTableView` dans le fichier `ViewController.swift`. Cette méthode doit mettre à jour l'objet de type `UITableView` afin que les données du tableau `data` soit de nouveau chargées dans la table.

Conseil : Utilisez la méthode `reloadData` de la classe `UITableView`.

<https://developer.apple.com/documentation/uikit/uitableview>

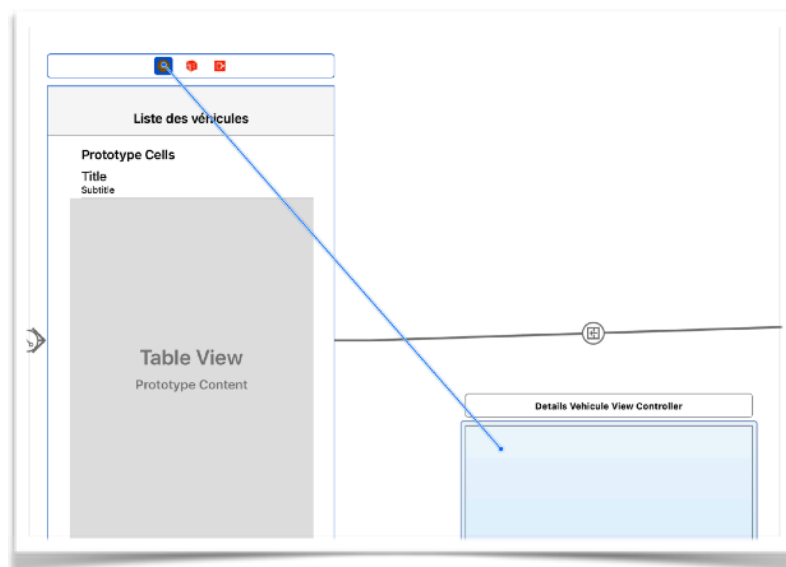
Question 14 : Dans le storyboard, créez le mécanisme de « unwind segue » à partir du bouton « Enregistrer » de la vue de définition d'une nouvelle voiture.

Partie V : Création de l'écran 3

Vous allez maintenant créer le dernier écran de votre application et transférer les informations concernant la voiture dont vous souhaitez visualiser les détails vers votre nouvelle vue. Vous devez également permettre la modification de l'attribut `enLocation` de ce véhicule et transférer cette modification vers la vue initiale.

Question 14 : Créez un nouveau contrôleur de vue et un nouveau fichier, appelé `DetailsVehiculeViewController.swift`, héritant de la classe `UIViewController` dans votre projet (cf. Questions 4 et 5).

Question 15 : Créez une nouvelle transition (segue) entre votre contrôleur de vue initial `viewController` et votre nouveau contrôleur de vue `DetailsVehiculeViewController`. Pour cela, vous n'allez pas sélectionner un élément de votre interface graphique mais directement l'icône de votre contrôleur de vue. Identifiez votre transition.



En effet nous souhaitons que la transition soit créée lorsque l'utilisateur sélectionne une des lignes de l'objet de type `UITableView`, c'est à dire lorsque la méthode `tableView(_: tableView: UITableView, didSelectRowAt indexPath: IndexPath)` du protocole `UITableViewDelegate` est appelée. Nous allons donc créer la transition directement dans le code.

Question 15 : Ajoutez les éléments d'interface graphique et les IBOutlet qui vous semblent nécessaires à l'interface graphique de votre nouvelle vue.

Question 16 : Ajoutez une variable d'instance `selectedVehicule` à votre fichier `viewController.swift` et une variable d'instance `detailedVehicule` à votre fichier `DetailsVehiculeViewController.swift`.

Question 17 : Modifier la méthode `tableView(_ tableView: UITableView, didSelectRowAt indexPath: IndexPath)` du fichier `viewController.swift` afin qu'elle :

- ◆ mette à jour la variable d'instance `selectedVehicule`
- ◆ appelle la méthode `performSegue(withIdentifier: "idSegue", sender: self)`. Cette méthode va permettre d'appeler la méthode `prepare` et d'effectuer la transition identifier par l'identifiant « `idSegue` ».

Question 18 : Écrivez la méthode `prepare` du fichier `viewController.swift` afin qu'elle mette à jour la variable d'instance `detailedVehicule` de votre fichier `DetailsVehiculeViewController.swift` à partir de la variable d'instance `selectedVehicule` de votre fichier `viewController.swift`.

Question 19 : Modifiez le code du fichier afin que la description du véhicule sélectionné s'affiche dans le label et que l'objet de type `UISwitch` soit initialisé correctement (en position ON si le véhicule est loué et en position OFF sinon).

Question 20 : Écrivez la méthode `prepare` du fichier `DetailsVehiculeViewController.swift` afin qu'elle mette à jour la variable d'instance `data` du fichier `ViewController.swift`. Cette fonction doit parcourir le tableau `data` afin de récupérer l'indice de la voiture présentée dans cette vue, supprimer la voiture du tableau `data`, modifier la voiture si nécessaire (mis à jour de l'attribut `enLocation`) et l'ajouter à sa position initiale dans le tableau `data`.

La comparaison de deux objets en Swift se fait avec l'opérateur `===`.