Compte rendu TP Programmation mobile Apple

Romain BIZOT
ESIREM 4A ILC

Le projet est de développer une application de gestion de tâche “Todo list”. Le développement de cette application s’est fait en respectant le modèle MVC.
Tout d’abord, j’ai créé 2 classes afin de pouvoir manipuler les données de cette application : La première classe est **MyData**, composée d’une description, d’un nom de tâche, d’un état de réalisation (type : Boolean, si elle est réalisé, donc à l'état “true”, une coche apparaît “✓”, sinon, une croix apparaît) et d’une date (type : Date), le tout stocké dans un fichier appelé **Data**. La deuxième classe est **MyCategory**, composée d’un état nous informant si la description est ouverte ou non (type : Boolean), d’un nom de catégorie et d’un tableau de tâche (type : MyData).
L’application est composée de 4 vues. 1 vue principale, une vue pour la création de nouvelle tâche, une vue pour la création de nouvelle catégorie et une dernière vu afin d’obtenir plus de précision sur une tâche :

- **La vue principale** est composée de catégories elles-même composée de tâches. On y retrouve également 3 boutons : 1 bouton pour trier les tâches dans un ordre croissant des dates de réalisation. 1 autre bouton pour accéder à la vue nous permettant de créer des nouvelles tâches et un dernier bouton nous permettant d’accéder à la vue où nous pourrons créer des nouvelles catégories. Cette vue possède également un tableView composé de catégorie. Lors d’un clic sur celle-ci, la liste des tâches associées à cette catégorie apparaît en dessous. Un swip vers la gauche permet de supprimer une tâche ou une catégorie. Si une catégorie est effacée, toutes les tâches à l’intérieur de celle-ci le sont aussi. Vous ne pouvez pas ajouter une tâche sans spécifier une catégorie, pour faire, il faut ouvrir une catégorie(en cliquant dessus) puis appuyer sur le bouton permettant d’ajouter une tâche. Ainsi la tâche s’ajoute automatiquement dans cette catégorie. Une searchBar est également disponible. (fichier : **ViewController**)
- **La vue concernant la création de nouvelle tâche** est composée de deux textfield : un pour le nom de la tâche et un pour la description de celle-ci. On retrouve également un datePicker afin d’associer une date à une tâche. Vous pourrez cliquer sur la croix en haut afin de quitter cette vue. Vous pourrez également sauvegarder la création de la tâche en appuyant sur le bouton save (fichier : **EditViewController**)
- **La vue concernant la création d’une nouvelle catégorie** est uniquement composée d’un textfield pour le nom de celle-ci. Vous pourrez cliquer sur la croix en haut afin de quitter cette vue. Vous pourrez également sauvegarder la création de la catégorie en appuyant sur le bouton save (fichier : **CategorieViewController)**

- **La dernière vue** est celle où nous retrouvons les **détails concernant une tâche**. Elle est composée de 3 labels affichant : le nom de la tâche, la description et la date de réalisation de celle-ci. On retrouve également un bouton afin de pouvoir changer l'état de la tâche : passé de "à faire" à "fait" et inversement (fichier : **DetailViewController**).

Concernant les dates, un format a été donné : "dd/MM/yyyy", ainsi avec la fonction "formatter", il est possible de transformer une chaîne de caractère string en une date de type Date.
La fonction **numberOfSection()** retourne le nombre d'éléments qu'il y a dans le tableau myCategory : initialement, il y en a 2.
La fonction **tableView(didSelectRowAt)** a pour rôle de changer l'état du boolean "opened", lors d'un clic sur une catégorie, présent dans la classe MyCategory.
La fonction **tableView(numberOfRowsInSection)** retourne 1 si on a pas cliqué sur une catégorie, en revanche, si on clique sur une catégorie, on entre dans la condition du if et on affiche le nombre de tâche présent dans cette catégorie.
La fonction **tableView(cellForRowAt)** comprend 2 cellules prototypes : "tachecell" pour les tâches et "categoryCell" pour les catégories. De base on retourne les cellules concernant les catégories, sinon, on retourne les cellules qui concernent les tâches avec leur nom, leur date et leur état.
La fonction **tableView(editingStyle**) permet de supprimer une tâche ou une catégorie en swippant vers la gauche. Une fois la suppression effectuée, on appelle la fonction reloadData() afin d'afficher de mettre à jour la vue.
Dans la fonction **viewDidLoad()**, on remplit le tableau dateTache() avec les dates des tâches. On remplit ensuite le tableau myData avec les tableaux : descTache (description de la tâche), nomTache (nom de la tâche), etatTache (état de la tâche) et dateTache (date de la tâche). On va ensuite remplir le tableau myCategory avec opened (catégorie ouverte ou non), titleCategory (nom de la catégorie) ainsi que des taches (éléments du tableau myData).
La fonction **triDateTache()** est la fonction associée au bouton nous permettant de trier les tâches par ordre croissant de date dans la vue principale. Dans cette fonction on va parcourir toutes les catégories puis on va répéter l'action autant de fois qu'il y a de tache dans celle-ci. Si une tâche est plus ancienne qu'une autre, alors on la stock dans une variable tampon et on fait passer l'autre tâche avant celle-ci et ainsi de suite. Il s'agit d'un tri à bulle.

Passons maintenant aux fonctions nous permettant de communiquer avec les autres vues :
La fonction **prepare()** va nous permettre d'envoyer des données à la vue DetailViewController : celle-ci correspond à la vue détaillant les tâches. Pour choisir quelles informations sont envoyées à cette vue, on récupère l'indice de la section ainsi que celui de la tâche. Ensuite dans le fichier DetailViewController, on crée une tâche de type myData afin d'afficher dans les labels les informations souhaitées (nom, description, etat, date).

Pour communiquer avec la vue qui permet de créer une nouvelle tâche, deux fonctions ont été créées, une fonction **close** et une fonction **save**. Les deux ont été associées à un bouton présent dans la vue de création de tâche. La fonction **close** correspond au bouton en forme de croix, elle permet naturellement de fermer la page. La fonction **save** permet de sauvegarder une nouvelle tâche en l'ajoutant à la catégorie ouverte au moment de la création de la tâche. Les deux conditions pour la créer sont : mettre un titre et une description (au minimum) et ouvrir une catégorie avant de créer la tâche elle doit être associée à une catégorie.
Pour communiquer avec la vue de la création de nouvelle catégorie, deux fonctions ont été créées en suivant le même raisonnement que pour les fonctions de création de nouvelles tâches. La fonction **closeCategoriePage()** permet de fermer la page et est associée à la croix dans la vue de création de catégorie. La fonction **saveCategoriePage()** permet de sauvegarder une nouvelle catégorie et est associée au bouton save.
Pour finir, une searchBar a été créée, cependant, nous avons vu ensemble qu'une erreur apparaît lorsqu'on place une searchBar dans le storyBoard (UISearchDisplayController is unavailable when deploying to iOS 13.0 or later).
Nous avons donc trouvé une solution afin de faire apparaître une searchbar via des lignes de code dans la fonction viewDidLoad().
Tout se passe désormais dans la fonction **updateSearchResultat()**.
Lorsqu'on écrit un mot dans la searchBar, la vue principale est mise à jour afin d'afficher toutes les tâches possédant ce mot dans leur titre. L'erreur arrive maintenant, lorsqu'on efface notre recherche dans la searchbar, uniquement la tâche qui a été trouvée lors de la recherche précédente existe encore, toutes les autres ont été supprimées.