Bilan Itération 1

Classe créée:

Modèle :

Tâche:

Une tâche pouvant contenir plusieurs tâches enfants, nous avons décidé de faire un patron composite. Pour l'instant seul les tâches mère peuvent avoir des enfants. Nous avons donc la classe abstraite tâche qui possède un nom et une description ainsi qu'une méthode affichage. Une tâche mère possède en plus une liste de sous-tâches.

Conteneur:

Un conteneur contient plusieurs tâches et pour l'instant ne peut être affiché qu'en colonne. Le conteneur possède aussi un titre. On peut ajouter une tâche au conteneur ou supprimer une tâche.

Tableau:

Un tableau contient plusieurs conteneurs. Pour l'instant lors de l'itération 1, le tableau contient deux conteneurs de base et l'utilisateur ne peut pas en rajouter ni en supprimer

MVC:

Modèle:

Notre modèle possède une liste d'observateurs, un tableau, une entier représentant la colonne sélectionnée par l'utilisateur, il possède aussi un boolean formulaire utilisé pour savoir s' il faut montrer le formulaire ou non. Le Modèle possède une méthode pour changer la colonne sélectionnée par l'utilisateur, une méthode pour créer une tâche et une méthode pour mettre le boolean formulaire à true si il est false et vice versa.

Controller:

ControllerAfficherFormulaire: Controller utilisé pour afficher le formulaire quand le bouton ajouter un tâche est cliqué

ControllerCréerTache: controller utilisé pour Créer la tache et retirer le formulaire de l'affichage quand le bouton créer tache ou annuler est cliqué. On peut noter qu'il possède deux attributs en plus du modèle qui sont deux textField pour que le controller puisse récupérer le nom et la description de la tâche donnée par l'utilisateur.

Vue:

VueContainers : vue qui gère l'affichage des tache dans les containers

VueFormulaire : vue qui gère l'affichage du formulaire

Fonctionnalité:

Base Graphique:

Notre application affiche une page principale avec deux colonnes et des tache dedans, dans chaque colonne, on trouve un bouton ajouter tache sur lequel l'utilisateur peut cliquer. Le nom de l'application est aussi afficher

Créer Tâche (Jawad, Hugo):

Quand l'utilisateur clique sur le bouton ajouter tâche le bouton appelle le controller ControllerAfficherFormulaire celui ci appelle la fonction du modèle switchFormulaire() et le modèle va notifier ses vues, la vueFormulaire va donc s'afficher. L'utilisateur va ensuite pouvoir renseigner dans le formulaire le nom de la tâche et l'éventuelle description, que l'utilisateur appuie sur le bouton annuler ou créer tache, le controller ControllerCreerTache si l'utilisateur a appuyer sur le bouton créer tâche, le controller va appeler la méthode créerTache du modèle qui va ajouter la tâche à son tableau et va notifier ses observateur, enfin que l'utilisateur ait appuyé sur le bouton annuler ou créer tache, le controller appelle la méthode switch formulaire et le formulaire va donc se fermer.

Diagramme De Séquence

Afficher Tâche (Timothée, Maxime) :

L'affichage des tâches est gérer par la vue conteneurs, la vue va construire le tableau à de manière itérative à partir de la classe tableau, puis des conteneurs de celui-ci, et enfin des tâches de chaque conteneur.
Toutes les hboxs des taches seront donc créées et stockée dans la hbox de leurs colonne respectives, puis chaque colonne sera stockée dans les enfants de la vueConteneur qui est une Vbox et qui représente un tableau