Le Modèle Vue Contrôleur

Le MVC est un motif de conception (*design pattern*) qui propose une solution générale au problème de la structuration d'une application. Le MVC définit des règles qui déterminent dans quelle **couche** de l'architecture, et dans quelle **classe** (orientée-objet) de cette couche, doit être intégrée une fonctionnalité spécifique.

Les applications peuvent être complexes, ainsi que l'interface correspondante. Le principe est donc de séparer les données (le modèle), l'interface homme-machine (la vue) et la logique de contrôle (le contrôleur).

La Vue

Elle représente l'interface utilisateur (l'interface graphique avec ses différents composants), et ce avec quoi il interagit. Elle n'effectue aucun traitement, elle se contente d'afficher les données que lui fournit le modèle. Il peut tout à fait y avoir plusieurs vues qui présentent les données d'un même modèle. La vue est également responsable de la mise en forme des données (pour formater une date par exemple) et doit d'ailleurs se limiter à cette tâche.

Le Modèle

Le modèle implante les fonctionnalités de l'application, indépendamment des aspects interactifs. Il représente les données de l'application qui contient tous les attributs et méthodes qui doivent participer à la tâche principale à accomplir. Il définit aussi l'interaction avec la base de données et le traitement de ces données.

Quand le modèle est modifié, il faut en général actualiser l'interface graphique ; le modèle n'a pas besoin pour cela de connaître l'interface graphique. On remarque que le modèle ne connaît ni la vue, ni le contrôleur, que la vue connaît le modèle, que le contrôleur connaît la vue et le modèle.

Contrôleurs et actions

Le rôle des contrôleurs est de récupérer les données utilisateurs, de les filtrer et de les contrôler, de déclencher le traitement approprié (via le modèle), et finalement de déléguer la production du document de sortie à la vue.

Il gère l'interface entre le modèle et le client. Il va interpréter la requête de ce dernier pour lui envoyer la vue correspondante. Il effectue la synchronisation entre le modèle et les vues. Il contient l'ensemble des listeners et qui, lorsque des événements parviennent via la vue, prévient le modèle en conséquence.

L'utilisation de contrôleurs a également pour effet de donner une structure hiérarchique à l'application, ce qui facilite la compréhension du code et l'accès rapide aux parties à modifier. Les contrôleurs, comme leur nom l'indique, ont pour rôle essentiel de coordonner les séquences d'actions/réactions d'une application.

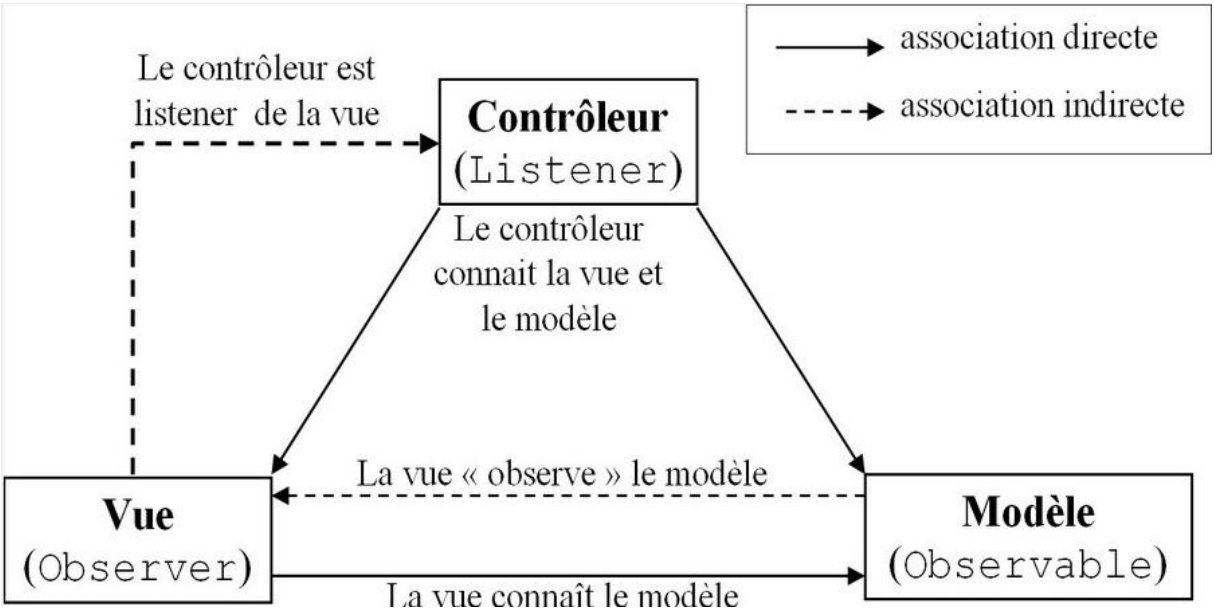
Dans notre cas, pour la partie jeu :

Le modèle est équivalent à la classe **Stage.java**

La vue est équivalent à la classe **Enchainement.java**

Le contrôleur est équivalent à la classe **Deplacement.java**

Voici un schéma des interactions entre les différentes couches :



La synchronisation entre la vue et le modèle se passe avec le pattern Observer. Il permet de générer des événements lors d'une modification du modèle et d'indiquer à la vue qu'il faut se mettre à jour.

Ce modèle de conception permet principalement deux choses :

- le changement d'une couche sans altérer les autres. C'est-à-dire que comme toutes les couches sont clairement séparées, on doit pouvoir en changer une sans porter atteinte aux autres couches. On pourrait aussi donc changer le modèle sans toucher à la vue et au contrôleur. Cela rend les modifications plus simples ;
- la synchronisation des vues. Avec ce design pattern, toutes les vues qui montrent la même chose sont synchronisées.

Il faut tout de même garder en mémoire, que la mise en œuvre de MVC dans une application n'est pas des plus simples. En effet, ce modèle de conception introduit tout de même un niveau de complexité assez élevé. De plus, implémenter MVC dans votre application nécessite une bonne conception dès le départ. Ce qui peut prendre du temps. Ce pattern n'est donc à conseiller que pour les moyennes et grandes applications.

Sources :

https://perso.telecom-paristech.fr/hudry/coursJava/interSwing/boutons5.html

http://orm.bdpedia.fr/mvc.html

https://baptiste-wicht.developpez.com/tutoriels/conception/mvc/