Le model MVC :

Le MVC est un modèle de structuration de code ([Design Pattern](https://www.youtube.com/watch?v=tDxnyop48mY&list=PLsyeobzWxl7r2ZX1fl-7CKnayxHJA_1ol&index=1)), principalement utilisé dans le cadre du développement d’application plutôt complexe qui nécessite une multitude d’interface, affichage multiple et possédant de nombreuses classes. Le MVC se construit sous la forme d’un système de communication entre trois entités, le Controller, la Vue et le Modèle, chacun communiquant entre eux à l’aide d’évènements (« voir [la programmation événementielle](https://www.youtube.com/watch?v=NtYE9naR0ys) », Programmation Objet, [Modèle Observer](https://youtu.be/98DiwRp-KZk), et Singleton).

Suivre le MVC permet d’obtenir une application fiable, sans bug et performante. Il permet l’ajout de nouvelles fonctionnalités rapidement et avec un risque minimisé de bugs.

Il est composé de trois entités :

* La Vue : C’est une interface machine (IHM). Cet ensemble est composé de toutes les interfaces graphiques du programme. Cela peut être une page web, interface de jeu ou d’application.
* Le Modèle : Il contient toutes les données du programme c’est à dire l’ensemble des algorithmes, la configuration de l’application, la logique, …c’est le cœur de l’application.
* Le Contrôleur : il contrôle le flux de données. Il permet de transmettre les informations entre la vue et le Modèle. Ainsi il permet par exemple de notifier d’un chargement terminé que l’utilisateur effectue une action.

Implémentation du MVC dans le cadre de notre projet :

* Il peut avoir plusieurs modèles, vues et contrôleurs.
* On aura une interface (IHM) permettant de déplacer le Pac-Man, d’afficher le jeu, le menu, …ce sera la Vue. Elle correspond aux différentes interfaces et aux différents affichages de notre programme.
* Le Modèle enregistre les différentes modifications des positions du Pac-Man, déplace les fantômes, etc., …et fait tourner le jeu selon une logique imposée.
* Le contrôleur fera office de passerelles entre les vues et les modèles, il interprètera les actions de l’utilisateur pour en informer le Modèle.

Différents modèles de conception MVC:

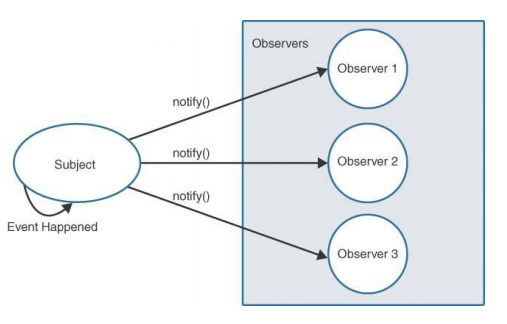
* [Observer](https://youtu.be/98DiwRp-KZk): Basé sur la programmation évènementielle pour communiquer sous formes évènements entre la Vue, le Modèle et le Contrôleur.
* Command : Utilisé par le contrôleur pour lancer plusieurs événements pour les vues concernées.
* Stratégie : Le Contrôleur se charge des entrées utilisateurs, de les transmettre au Modèle et la Vue effectue seulement l’affichage.
* Composite : Un pattern organisé par plusieurs Vues que l’on superpose.

Observer design pattern :

Le modèle Observer a été créé pour résoudre le problème du partage d'informations entre un objet et de nombreux autres objets. Ce type de modèle décrit un comportement très utile pour les systèmes distribués qui doivent partager des informations de configuration ou des détails sur les modifications qu'elles se produisent. Le modèle Observer est vraiment simple et se compose de seulement deux éléments logiques :

- Sujet : Le sujet représente l'état de l'objet observé - en d'autres termes, les données qui doivent être synchronisées. Le sujet fournira un moyen de permettre à d'autres composants de l'application, même à des systèmes distants, de s'enregistrer pour détecter une modification de l'objet suivi. Après l'enregistrement, l'abonné recevra une notification de mise à jour chaque fois qu'il y a un changement dans les données du sujet, à partir de laquelle les parties se synchronisent.

- Observateur : est un objet qui s'enregistre auprès du sujet, ce qui permet au sujet d'identifier quels observateurs suivent l'objet et de trouver un moyen de communiquer avec lui. La seule fonction de l'observateur est de synchroniser ses données avec le sujet lorsqu'il est appelé. La chose importante à comprendre à l'Observer est qu'il n'obtient aucune donnée directement, il attend juste que le sujet envoie les données. Ce processus peut être très inefficace avec un plus grand nombre d'observateurs inscrivant le même sujet.



Le modèle d'observation est souvent utilisé pour gérer la communication entre le Modèle et la Vue dans le modèle MVC. Par exemple, supposons que vous ayez deux Vues différentes pour l'utilisateur final. Une Vue fournit le graphique en barres et une autre Vue fournit le graphique de dispersion. Les deux utilisent la même source de données d'un modèle. Lors de la modification des données ou de la mise à jour, deux vues doivent être mis à jour. Il s'agit d'un travail parfait pour le modèle d'observation.

Exemple : Une chaine YouTube sort une nouvelle vidéo et veut le notifier à tous ses abonnées , on PUSH AND REQUEST….