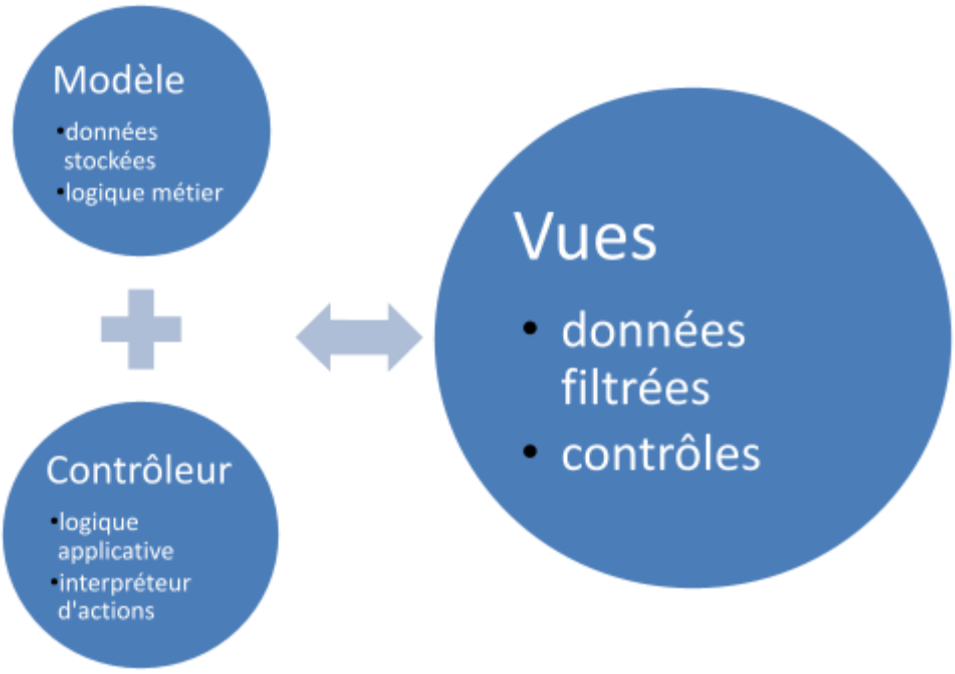
Approches MVC en web

L'approche MVC consiste à séparer le code en trois parties : modèle, vue et contrôleur.



Le modèle contient les données en mémoire vive et implémente aussi la logique métier. Ceci comprend toute règle de validation ou d'intégrité des données, mais aussi les opérations de transformation qui sont inhérentes à ces types de données. Dans le cours je les classe dans **modele**.

Les vues impliquent toutes les interfaces utilisateurs qui permettent à un humain de visualiser les données et aussi d'interagir avec l'application. La vue permet à l'humain de voir les données mais aussi à l'application de voir les actions de l'humain. C'est pourquoi les vues impliquent des liens de navigation et aussi des formulaires de dialogue et d'entrée de données. Pour un même modèle de données, il peut exister plusieurs vues dans une même application ou encore plusieurs applications de vues sur ces données. Dans le cours je les classe dans **public**.

Le contrôleur, c'est la logique applicative – à distinguer de la logique métier – qui permet à l'utilisateur d'agir. Grâce au contrôleur, on interprète et exécute les demandes des usagers (membres ou administrateurs) lorsqu'il s'agit d'écrire, modifier, effacer des données, créer un compte, annuler une commande, etc. Dans le cours, je les classe dans **action**.

DAO

Les DAO sont une interface à la base de données permettant de gérer d'une manière applicative les accès à celle-ci. C'est une classe qui implémente les actions SQL sur une base de données tout en présentant à l'application des fonctions qui s'utilisent avec des objets du modèle.

L'avantage des DAO est la concentration du code BDD et aussi l'isolation de cette couche pour les évolutions possibles (changement de SGBD ou encore migration vers un CLOUD avec un service de données). Dans le cours on les classe dans **accesseur**.

Ils remplacent l'utilisation de procédures stockées qui est une approche où la logique métier se répartit entre la base de données et l'application en permettant de concentrer cette logique seulement dans l'application.

Les implémentations

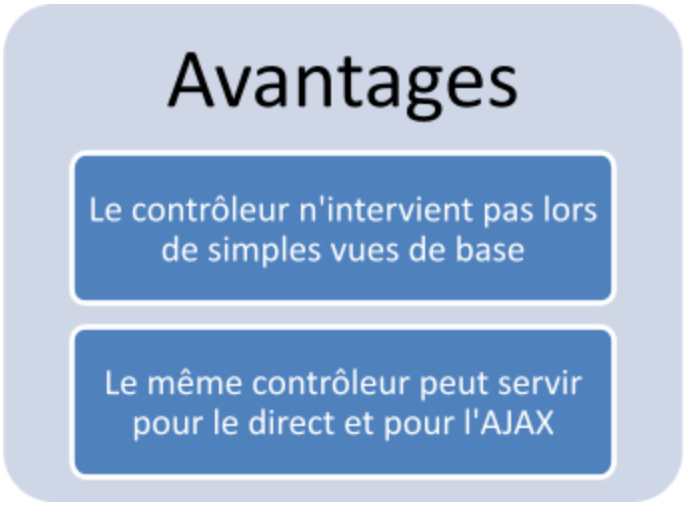
Dans le domaine du web en PHP, on reconnait facilement ces éléments dans nos sites web :

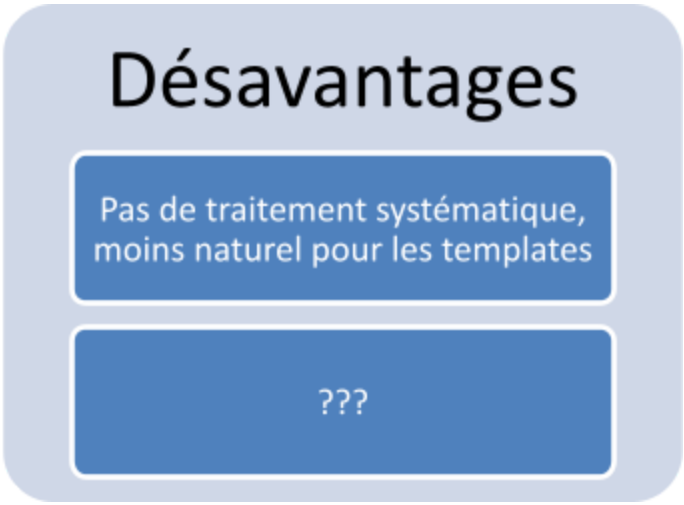
- Les vues sont les **pages web** visibles combinant HTML, CSS et données.
- Les modèles sont des **classes** PHP conçues pour contenir un enregistrement, un objet, agrégat ou non
- Les contrôleurs sont les **scripts** PHP qui gèrent les entrées de l'utilisateur et ses commandes, souvent des formulaires

Organisation Vue en Avant

Dans cette organisation MVC web, la vue est directement accessible par son URL. Par conséquent, la requête GET correspondant à 'lire' une page sans paramètre n'est pas traitée par les contrôleurs et constitue en fait, une exception.

Dans cette approche, seules les actions plus 'actives' de l'utilisateur nécessitent l'intervention du contrôleur qui ne retourne que sa portion de traitement (données modifiées, message d'erreur, etc.). C'est la vue qui appelle le contrôleur lorsque nécessaire : soit en AJAX ou soit par un include qui entre en jeu lorsque l'action est renseignée.





Organisation Contrôleur en Avant

Dans cette organisation MVC web, le contrôleur est ce qu'on accède par son lien et ce sont les paramètres de celui-ci qui déterminent si on est en action de lecture ou d'écriture. Cette approche systématise toutes les requêtes de l'utilisateur en traitant le GET comme les autres ce qui plait à une approche applicative et qui se prête au jeu de la machine d'état. C'est l'approche du J2EE avec JSP et Spring MVC, approche imitée par CakePHP, Symfony, Laravel. Cette approche a été introduite dans le web par des développeurs provenant du monde applicatif, très expérimentés en objet mais peu expérimentés en web.

Avantages

- Traitement systématique des demandes usager
- Utilisé par certains framework connus

Désavantages

- Le contrôleur traite des données en trop (il outrepasse sa responsabilité
- Ne marche pas avec AJAX, on doit réimplémenter des contrôleurs séparés
- Décourage l'adressabilité des URL

Discussion

L'approche Contrôleur devant est la plus utilisée mais elle est une aberration dans le monde du web.

En fait, elle constitue une dérogation au réel principe du MVC dans lequel on souhaite que la vue soit le point d'entrée et la seule interface entre le client et l'application. C'est une dérogation car dans un découpage où l'on a pris la peine de séparer la vue des actions et des modèles afin notamment de permettre à l'usager d'interagir avec la vue, on se demande bien pourquoi on permettrait à l'usager de parler directement au contrôleur ? Une partie de la réponse nous est fournie par le monde applicatif et ses limitations.

Dans le monde applicatif, on se retrouve dans une situation où la VUE doit être instanciée par le programme, et les programmeurs se disent : pourquoi ne pas confier cette responsabilité au contrôleur qui décide pour tous les autres événements ce qui se passe. Par conséquent, de faire instancier (ou inclure) les vues par le contrôleur constitue une solution à une limitation de l'infrastructure logicielle de type DESKTOP ou non-web.

Cette limitation du monde applicatif (limitation qui est que la vue a besoin d'être créée) n'existe pas dans le monde web non-CGI où le document est servi d'abord par Apache et que les parties dynamiques constituent des exceptions à la page web standard. Si cette limitation n'existe pas, alors il serait absurde de copier la solution à ce problème inexistant. Cette approche s'est popularisée par la copie aveugle de la solution MVC telle qu'implémentée d'abord dans les milieux du monde Java. Pourtant, le web nous offre l'opportunité d'avoir un PUR MVC alors ne manquons pas cette occasion. Évitons de reculer à l'époque des CGI.