Design Patern

Patron de conception

vingt-trois décrit dans un livre« patrons Gang of Four »

Un patron de conception est la meilleure solution connue à un problème de conception récurrent.

En **informatique**, et plus particulièrement en **développement logiciel**, un **patron de conception** (souvent appelé *design pattern*) est un arrangement caractéristique de modules, reconnu comme bonne pratique en réponse à un problème de conception d'un logiciel. Il décrit une solution standard, utilisable dans la conception de différents logiciels.

En clair, les patrons de conception doivent être perçus comme un outil de conception et de structuration formelle des applications informatiques. Ce sont des solutions éprouvées et figurant parmi les bonnes pratiques de programmation à adopter.

singleton

le **singleton** est un **patron de conception** *(design pattern)* qui vise à assurer qu'il n'y a toujours qu'une seule instance d'une classe en fournissant une interface pour la manipuler C'est un des patrons les plus simples.

L'objet qui ne doit exister qu'en une seule instance comporte une méthode pour obtenir cette unique instance et un mécanisme pour empêcher la création d'autres instances.

Le Singleton, en programmation orientée objet, répond à la problématique de n'avoir qu'une seule et unique instance d'une même classe dans un programme. Par exemple, dans le cadre d'une application web dynamique, la connexion au serveur de bases de données est unique. Afin de préserver cette unicité, il est judicieux d'avoir recours à un objet qui adopte la forme d'un singleton. Il suffit donc par exemple de créer dans un variable globale du programme afin que l'on puisse y accéder de n'importe ou dans le script.

Modèle-vue-contrôleur

Combinaison des patrons *observateur*, *stratégie* et *composite*, ce qui forme ainsi un patron d'architecture.

Modèle-vue-contrôleur ou **MVC** est un motif d'**architecture logicielle** destiné aux **interfaces graphiques** lancé en 1978 et très populaire pour les **applications web**. Le motif est composé de trois types de modules ayant trois responsabilités différentes : les modèles, les vues et les contrôleurs.

- Un modèle (Model) contient les données à afficher.
- Une vue (View) contient la présentation de l'interface graphique.
- Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

Le pattern MVC permet de bien organiser son code source. Il va vous aider à savoir quels fichiers créer, mais surtout à définir leur rôle. Le but de MVC est justement de séparer la logique du code en trois parties que l'on retrouve dans des fichiers distincts.

- **Modèle** : cette partie gère les *données* de votre site. Son rôle est d'aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le contrôleur. On y trouve donc entre autres les requêtes SQL.
- **Vue** : cette partie se concentre sur l'*affichage*. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, pour afficher par exemple une liste de messages.
- **Contrôleur** : cette partie gère la logique du code qui prend des *décisions*. C'est en quelque sorte l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du PHP. C'est notamment lui qui détermine si le visiteur a le droit de voir la page ou non (gestion des droits d'accès).