Progression PHP objet

**Objectif : Savoir développer des programmes en PHP orienté objet.**

Nous allons reprendre l’application “boutique” que nous avons élaborée lors des étapes précédentes.

**Etape 1 – utilisation des classes**

*Modalité*

* Travail individuel.

*Objectif (compétences)*

* Définir des classes
* Instancier des objets
* Manipuler les collections d’objets

*Résultat attendu*

* Production individuelle : afficher un catalogue d’articles en utilisant des objets

*Consignes*

* Créer une classe Article ayant comme attributs :
* Nom
* description
* Prix
* Image
* Poids
* Stock
* Disponible (O/N)
* ...

On créera une fonction **displayArticle()** qui admet en paramètre un objet Article, et qui permettra l’affichage de l’article en Html.

* Créer la classe Catalogue, contenant une tableau d’objets Article. Le constructeur de cette classe permettra le remplissage du catalogue à partir de la base de données.

On affichera le catalogue en utilisant une fonction **displayCat()** qui admet en paramètre un objet Catalogue, et qui permettra l’affichage du catalogue en Html..

* Modifier ensuite le programme **catalogue.php** qui affiche une liste d’articles en utilisant cette classe Catalogue.
* Nous allons créer de la même manière une classe Client et une classe ListeClients, ainsi qu’un programme **clients.php** qui affichera la liste des clients.

**Ressources**

* *Doc* [*https://openclassrooms.com/courses/programmez-en-oriente-objet-en-php*](https://openclassrooms.com/courses/programmez-en-oriente-objet-en-php) *(partie 1, chapitre 1,2,3,4,et 6)*
* [*https://www.learn-php.org/en/Objects*](https://www.learn-php.org/en/Objects)
* [*http://php.net/manual/en/language.oop5.basic.php*](http://php.net/manual/en/language.oop5.basic.php)

*Production attendues*

* Code commenté du programme **catalogue.php** et des classes **Article** et **Catalogue**
* Code commenté du programme **clients.php** et des classes **Client** et **listeClients**
* Glossaire des différents concepts objets et syntaxes utilisées : **glossaire PHP**

**Etape 2 – Mise en oeuvre de l’héritage**

*Modalité*

* Travail individuel.

*Objectif (compétences)*

* Mettre en oeuvre l’héritage objet

*Résultat attendu*

* Programme qui affiche différents types d’articles, en utilisant l’héritage objet.

*Consignes*

Certains articles possèdent des attributs complémentaires par rapport aux articles standards (par exemple les chaussures ont une pointure, les vêtements ont une taille, etc...).

* Créer les classes Chaussure et Vêtement, qui héritent de la classe Article. Définissez leurs attributs spécifiques et leur constructeur spécifique.
* Modifier la base de données en rajoutant les deux tables **Chaussures** et **Vetements**. La table Chaussures aura comme attribut un id (auto incrémenté), un idArticle qui est une clé étrangère pointant vers l’article (permettant de récupérer toutes les informations de l’article chaussure), et un attribut pointure.

De même, la table Vêtements aura comme attribut un id (auto incrémenté), un idArticle qui est une clé étrangère pointant vers l’article , et un attribut taille.

Remplir ces tables avec quelques données de test.

* Modifier **catalogue.php**  en incluant dans la liste d’articles des articles de base, des chaussures, et des vêtements.

*Ressources*

* *Héritage:* *tuto learn-php ( objets ) :* [*https://www.learn-php.org/fr/Objets*](https://www.learn-php.org/fr/Objets)
* *Tutos openclassroom* [*https://openclassrooms.com/fr/courses/1665806-programmez-en-oriente-objet-en-php/1666684-lheritage*](https://openclassrooms.com/fr/courses/1665806-programmez-en-oriente-objet-en-php/1666684-lheritage)
* *Vidéo Grafikart : La POO en PHP (8/31) : L'héritage*

[*https://www.youtube.com/watch?v=ccuoGgWz5Do*](https://www.youtube.com/watch?v=ccuoGgWz5Do)

*Production attendues*

* Code commenté des différents programmes.
* Glossaire des différentes fonctions et syntaxes utilisées.

**Etape 3 – Gestion du panier**

*Modalités*

* Travail individuel, puis travail de groupe.

*Objectif (compétences)*

* Définir des classes
* Instancier des objets

*Résultat attendu*

* Programme qui affiche le panier en utilisant un objet **Panier**

*Consignes*

* Créer un objet **Panier**, qui a comme attribut un tableau associatif avec comme clé l’id de l’article et comme valeur une quantité, ainsi que les méthodes suivante :
* *add* : qui admet en paramètre un id article, et qui crée la ligne de Panier avec une quantité à 1 si l’article n’est pas déjà présent dans le Panier, ou qui rajoute 1 à la quantité correspondante si l’article est déjà présent dans le Panier.
* *update* : qui admet en paramètre un id article et une quantité, et qui rajoute cette quantité à la ligne de panier correspondant à l’article.
* *delete* : qui admet en paramètre un id article, et qui supprime la ligne de Panier correspondante
* Afficher le panier à l’aide d’une fonction **displayPanier()**, qui aura en paramètre l’objet Panier.
* Reprendre les programmes **catalogue.php** et **panier.php** pour créer, remplir et afficher maintenant l’objet Panier.

*Ressources*

* *Doc* [*https://openclassrooms.com/courses/programmez-en-oriente-objet-en-php*](https://openclassrooms.com/courses/programmez-en-oriente-objet-en-php) *(partie 1, chapitre 1,2,3,4,et 6)*
* [*https://www.learn-php.org/en/Objects*](https://www.learn-php.org/en/Objects)
* [*http://php.net/manual/en/language.oop5.basic.php*](http://php.net/manual/en/language.oop5.basic.php)

*Production attendues*

* Code commenté des différents programmes.
* Glossaire des différentes fonctions et syntaxes utilisées.

**Etape 4 – Structuration de code (facultatif)**

*Généralités*

On va essayer dans ce qui suit, de structurer et d’organiser notre code en respectant un certain nombre de bonnes pratiques. Nous avons déjà (normalement !) factorisé notre code en utilisant au maximum des fonctions.

L’étape suivante va être de découper et de séparer les traitements qui concernent l’affichage (ce que l’on appelle aussi la **vue**), et ce qui concerne tous les accès aux données (ce que l’on appelle souvent le **modèle**). Nous reviendrons ultérieurement (et longuement !) sur ces concepts et sur le modèle **MVC** (modèle, vue, contrôleur), qui est un modèle indépendant des langages et outils de développement qu’il fait partie des bonnes pratiques du développeur.

Pour l’instant, au delà de tout aspect théorique, nous allons simplement essayer de structurer notre code de manière très concrète.

*Modalité*

* Travail individuel, puis travail de groupe.

*Objectif*

* Structuration de code

*Résultat attendu*

* Code structuré et découpage net entre affichage et accès aux données

*Consignes*

Pour séparer l’accès aux données de notre affichage, il convient d’enlever de nos pages PHP (qui servent à afficher les informations), toute référence aux bases de données, et notamment tout ce qui sert à la connection à la base, ainsi qu’à l’exécution des requêtes SQL et au traitement des jeux de résultats. Pour cela, une bonne pratique consiste à regrouper toutes les requêtes d’accès à une table données (table Articles par exemple) au sein d’une même classe (nommée ArticleManager, ou ArticleDao, ou autres…). Chaque requête sur la table Articles sera une méthode de cette classe ArticleManager.

* Modifier les programmes précédents en créant et utilisant la classe ArticleManager.

*Ressources*

* [*https://openclassrooms.com/fr/courses/1665806-programmez-en-oriente-objet-en-php/1666289-manipulation-de-donnees-stockees*](https://openclassrooms.com/fr/courses/1665806-programmez-en-oriente-objet-en-php/1666289-manipulation-de-donnees-stockees)

*Production attendues*

* Code commenté des différents programmes.
* Glossaire des différentes fonctions et syntaxes utilisées.