POO

En résumé : Ce projet est une introduction aux concepts de la programmation orientée objet (POO)

Introduction:

Vous allez maintenant apprendre à utiliser les principes de la programmation orientée objet avec PHP. Cette méthode de programmation (ou communément appelé “paradigme de programmation”) permet bien des avantages comme l’organisation du code et la productivité !

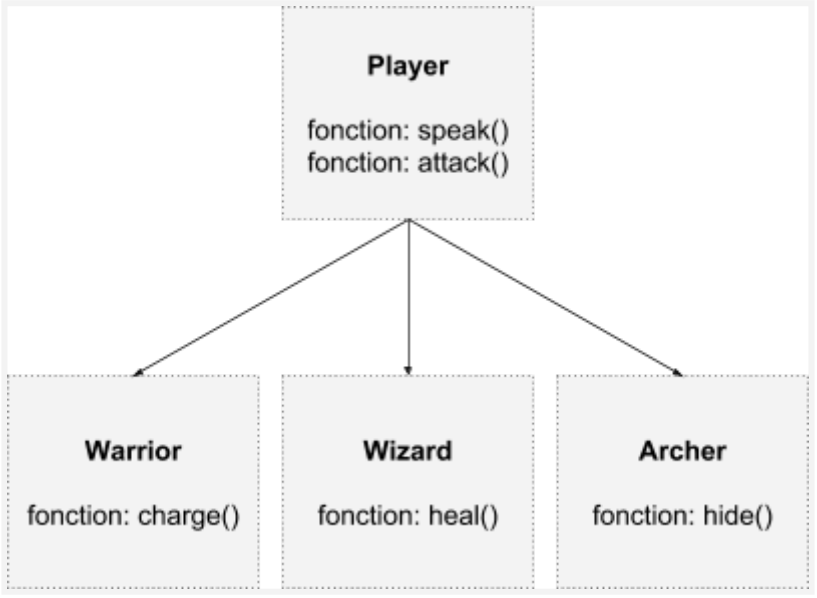
Préambule :

Dire que l'on fait de la programmation orientée objet (POO) revient à dire que l'on utilise une structure particulière dans l'organisation de notre code. Cette structure sera alors "orientée" vers l'utilisation d'objets. Un objet représente un ensemble sémantique, une brique de l'application : un concept, une idée, une entité (voiture, espèce animale, etc.).

La POO permet d'encapsuler un ensemble d'instructions et de le réutiliser sans fin. Il s'agit donc de la notion de réutilisabilité du code qui évite alors de réécrire plusieurs fois la même chose. Mais les bénéfices de la POO ne s'arrête pas là !

L'un des principes fondamentaux de la POO est le principe d'héritage. En effet, un objet peut hérité d'un autre objet.
Pour illustrer cette idée, imaginons le scénario suivant : on souhaite créer un système de jeu de rôle. Dans ce jeu, nous aurons donc des personnages (joueur). Chacun des personnages appartiendra à une spécialité de combattant : guerrière, mage et archère.

Dans ce scénario on aura un objet joueur ("Player") puis un objet pour chacune des spécialités: "Warrior", "Wizard" & "Archer". Voici ce que cela peut donner sous forme de schéma :



On voit alors que chacune des spécialités de personnage hérite de l’objet Player. Donc chacune des spécialités peut parler et attaquer. Puis les objets spécifiques à chaque race permettent d’implémenter la ou les spécificités des races (la guerrière peut charger un ennemie, le mage peut soigner un allié et l’archère peut se cacher).

Un dernier principe essentiel à la POO est la notion de “classe”. Une classe est la définition ou le schéma de notre objet (imaginer cela comme le plan d'architecture d’une maison). C’est donc à partir de la classe que nous serons en capacité de créer des objets (aussi appelés “instances de classe”). Concrètement, il s’agit de coder la classe (le plan) puis d’invoquer la création d’un nouvel objet à partir de cette classe grâce au mot clé “new” :

```
$myPlayer = new Warrior()
```

Dans notre exemple de jeu, nous avons quatres classes : Player, Warrior, Wizard et Archer. Les classes Warrior, Wizard et Archer “héritent” de la classe Player (autrement dit, ils ont accès au code de Player dans leur propre code). Une fois la classe réalisée, on peut créer (“instancier”) autant d’objet que l’on veut !

Objectifs :

- Avec PHP et la POO, réaliser seul un projet de mini-jeu de rôle.
    - Le jeu devra contenir trois spécialités de joueur : la guerrière (“warrior”), le mage (“wizard”) et l’archère (“archer”).
    - Un joueur aura :
        - Des points de vie
        - Des points d’attaque
        - Des points de défense
    - Un joueur pourra
        - Attaquer
        - parler
    - Le guerrier pourra
        - Charger : 50% de chance de faire 2x plus de dégât qu’une attaque normale
    - Le mage pourra
        - Se soigner
    - L’archère pourra
        - Se cacher : 50% de chance d’esquiver la prochaine attaque

Prérequis :

- Installation de PHP
- Avoir un serveur Web

Libertés :

- Tout ce qui n’est pas interdit est autorisé

Ressource :

- POO en PHP avec OpenClassroom : https://openclassrooms.com/courses/programmez-en-oriente-objet-en-php
- POO en PHP avec Grafikart : https://www.grafikart.fr/formations/programmation-objet-php
- Cheatsheet sur la POO avec PHP : https://buzut.fr/poo-en-php/