1-Créons notre première classe.

Pour le moment on a nos personnages lister d'une même manières dans un tableau associatif qui contient des clés et des valeurs pour chacun d'entre eux.
Comme on peux le voir chacune des lignes est dupliqué dans chacun des tableaux.

On se retrouve bien à avoir une entité propre qui correspond a chacun des perso qui on chacune des valeurs différente mais nous avons bien la même structure dans les tableaux
Ce qui serait interessant serait de modéliser ça d'une manière pertinente grâce aux objets.

Les objets vont nous permettre de définir des structures pour nos différent éléments. Par exemple ici pour tous nos personnages on va définir une structure que l'on va appeler personnage et cette structure en POO s'appelle une Classe.

Une classe va contenir des champs que l'on appelle attribut et va contenir des actions que l'on pouvoir donner a chacun de nos objets.

Par exemple pour afficher un personnage on avait une fonction. Maintenant ce que l'on pouvoir faire grâce à la classe c'est définir que notre objet personnage par exemple P1 est en capable de s'afficher lui même, pareil pour P2 et P3.
On va pouvoir donner des actions a chacun de nos objets et donc pour ça on va pouvoir définir des fonctions dans une Classe.

La programmation Objet est vraiment différente de la programmation procédurale c'est un autre concept qui va permettre d'améliorer la structuration de nos fichiers, la structuration de notre code et qui va permettre une meilleur maintenablité de nos programmes. Donc c'est vraiment une orientation vers laquelle il faut aller le plus possible car c'est aussi la plus utiliser en entreprise au jour d'aujourd'hui.

La Classe est le moule d'un objet. Donc pour chacun de nos personnages on va avoir un objet qui sera créé à partir d'une classe.
Pour définir une classe on va simplement écrire class et on lui donner un nom Personnage.
On lui définit des attributs qui étaient variables de notre personnage.
Pour définir les attributs nous utiliserons ->

2-Fonction de classe.

Nous allons créer une fonction permettant l'affichage de nos personnage.
$this permet de faire référence à l'objet lui même.

```
public function afficherPerso() {

        echo « nom : » .$this->nom . « <br/> »;
        echo « nom : » .$this->age . « <br/> »;
        echo « sexe :  »;
        if($this->sexe){
                echo « homme <br/> »;
        }else {
                echo « femme <br/> »;
        }
        echo « nom : » .$this->force . « <br/> »;
        echo « nom : » .$this->agilité . « <br/> »;
}
$p1= new Personnage();      avant création de la template d'affichage.
$p1-> afficherPerso();
```

Pas obligatoire mais pour afficher l'image a gauche je vais ré-écrire une fonction

```
public function afficherTemplatePerso() {

        echo « <div class=« gauche »> »;
        echo « <img src = « sources/images/« .$this->img. « ‘ alt = ‘« .$this->img »’ /> »;
        echo « </div> »;
        echo « <div class=« gauche »> »;
        $this->afficherPerso();
        echo « </div> »;
        echo « <div class=‘clearB’></div> »;
}

$p1= new Personnage();
$p1-> afficherTemplatePerso()
```

Le Constructeur

On va maintenant améliorer notre code car pour le moment nous avons modifier les informations de notre second joueurs directement dans le main alors qu'il serai plus judicieux de le faire au moment de la création de notre nouveau perso.
Pour pouvoir l'améliorer ce qui serait interessant quand on fait un new Personnage ce serait de pouvoir donner des informations a notre personnage(etc,etc,etc);
On va donc maintenant créer un constructeur de classe. De base si on ne met de pas de constructeur ça appellera le constructeur par défaut mais nous ce que l'on veux c'est pouvoir le redéfinir.

La structuration

Maintenant que nous travaillons avec des classes on va faire en sorte d'avoir un fichier qui va permettre de gérer la classe personnage et un fichier qui va permettre d'afficher le résultat dans un navigateur pour l'utilisateur.
Ça nous permet de segmenter et structurer notre code de la meilleur manière possible pour ensuite être capable de le maintenir simplement et d'utiliser vraiment la notion d'objet.

Les Getter et Setter

le problème des attributs mis en public c'est qu'il sont accessible de n'importe ou et par n'importe qui donc pour éviter ça on va les mettre en private pour qu'ils ne soient accessible de nul part si ce n'est depuis la classe elle même.
Donc maintenant si l'on veux modifier les informations de nos personnages il va falloir créer des fonctions qui peuvent le faire.
Les Getter et les Setter, les getter c'est pour récupérer les informations et les Setter pour modifier champs, on va donc créer un setter et un getter par attribut.
On créer uniquement les getter et setter sur les propriétés auxquelles on veux accéder.
Passer par ses fonctions obligent les développeurs a bien coder.

Les Constantes de Classes

Mettons que l'on donne notre code a un autre dev il ne va pas forcement savoir a quoi correspond notre boolean, ce qui serait bien c'est que l'on puisse mettre HOMME ou FEMME. Donc pour avoir

cette information la on peut le faire grâce aux constantes de classe c'est a dire que l'on va définir une valeur statiques dans notre classe que l'on pourra accéder de partout.
et ensuite on l'appellera dans notre nouveau personnage avec un :: pour faire appel à la constante.
C'est qui va permettre quand on va commencer a coder des pages plus compliqué de mettre des noms sur des choses qui n'en n'ont pas donc on peux faire comme ceci grâce a ces constantes de classes.

Les Attributs et méthodes statiques

Ce sont des valeurs qui sont lié a la classe et non a l'objet ce qui veux dire ces informations seront stocker au niveau de la classe et accessible depuis de la classe.