TP2 POO – Création d'un jeu de combat 2/4

TP2 POO - CREATION D'UN JEU DE COMBAT 2/4

DUREE TOTALE DU MODULE : 56H00

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

Travail à faire :

L'objectif de l'ensemble des 4 TP du module est de créer un modèle très simple de jeu de combat. Chaque visiteur pourra créer un nouveau personnage, avec lequel il pourra frapper d'autres personnages. Le personnage frappé se verra infligé un certain nombre de dégâts.

Un personnage est défini selon 2 caractéristiques, son nom et ses dégâts (entre 0 et 100).

Chaque coup porté au personnage lui infligera 5 points de dégâts. Une fois arrivé à 100 points de dégâts, le personnage est mort (supprimé de la BDD).

Notions étudiées durant les 4 TP :

- Les attributs et méthodes ;
- L'instanciation de la classe ;
- L'interconnexion avec la BDD ;
- L'héritage de classes ;

Dans ce second TP, nous focaliserons nos efforts sur la mise en place des interactions avec la BDD.

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

Nous avons créé lors du premier TP notre classe personnage, permettant l'instanciation de nouveaux personnages, et le combat entre nos personnages.

Cependant, les données sont pour le moment stockées uniquement de façon locale au sein du code.

Nous allons donc maintenant améliorer notre code pour gérer la persistance de nos données.

Notre classe Personnage a pour but de **représenter** un personnage présent au sein de notre BDD.

Elle n'a en aucun cas le rôle de les **gérer.**

Pour la gestion, nous allons créer une nouvelle classe, communément appelée **manager.**

Dans notre cas, cette classe nommée : PersonnagesManager.

1) Création de la classe PersonnagesManager

Pour fonctionner, votre manager de personnage a besoin des informations de connexion à votre BDD.
Cette information sera donc un attribut (et le seul) de cette classe.

- Créez la classe PersonnageManager, et déclarez la variable de connexion à la BDD comme attribut privé de cette classe

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

2) Création des accesseurs et du constructeur

Nos managers ayant besoin d'un attribut dbConnection pour fonctionner (créé en étape 2), il est nécessaire de mettre en place un constructeur et un setter pour le définir.

Il n'est pas nécessaire de mettre en place de getter sur cette variable, qui ne sera jamais récupérée.

- Créer un setter permettant d'initialiser la variable de connexion à la BDD (ce setter prendra en paramètre un objet PDO de connexion)
- Créer un constructeur, appelant ce setter pour initialiser la variable db.

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

Point d'étape : Vous devriez à ce point obtenir une classe similaire à la classe définie ci-dessous :

```
class PersonnagesManager

{

  private $_db; // Instance de PDO

  public function __construct($db)

  {

    $this->setDb($db);

  }

  public function setDb(PDO $db)

  {

    $this->_db = $db;

  }

}
```

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

3) Définition des méthodes

Pour pouvoir fonctionner, notre manager a besoin de multiples méthodes permettant la gestion des interactions entre notre classe Personnage et notre BDD.

En plus des fonctionnalités de CRUD, nous allons ajouter 3 méthodes permettant de compter le nombre de personnages, de récupérer une liste des personnages contre lesquels nous sommes susceptibles de nous battre, et de savoir si un personnage existe ou non.

Ces méthodes nous serviront la suite pour rendre notre jeu de combat fonctionnel.

- Ecrire **UNIQUEMENT** la structure des méthodes permettant le CRUD (création, lecture, modification et suppression) d'un Personnage. Le contenu des méthodes se composera uniquement de commentaires pour le moment.
- Ecrire **UNIQUEMENT** la structure des méthodes permettant le compte de personnage, la récupération sous forme de liste, et le test d'existence des personnages. Le contenu sera également du commentaire uniquement pour le moment.

TIPS :

La méthode get du personnage prendra en paramètre soit l'id du personnage, soit son nom.

La méthode existe prendra en paramètre soit l'id du personnage, soit son nom.

La méthode getList prendra en paramètre le nom du personnage actuel, et retournera un tableau d'instances de Personnages

Les méthodes Add, Delete et Update prendront en paramètre un objet Personnage.

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

Point d'étape : Vous devriez à ce point obtenir une classe similaire à la classe définie ci-dessous :

```
class PersonnagesManager
{
  private $_db; // Instance de PDO
  public function __construct($db)
  {
    $this->setDb($db);
  }
  public function add(Personnage $perso)
  {
    // Préparation de la requête d'insertion.
    // Assignation des valeurs pour le nom du personnage.
    // Exécution de la requête.
   }
  public function count()
  {
    // Exécute une requête COUNT() et retourne le nombre de résultats retourné.
  }
  public function delete(Personnage $perso)
  {
    // Exécute une requête de type DELETE.
  }
```

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

```
  public function exists($info)
  {
    // Si le paramètre est un entier, c'est qu'on a fourni un identifiant.
      // On exécute alors une requête COUNT() avec une clause WHERE, et on retourne un bool
ean.
    // Sinon c'est qu'on a passé un nom.
    // Exécution d'une requête COUNT() avec une clause WHERE, et retourne un boolean.
  }
  public function get($info)
  {
    // Si le paramètre est un entier, on veut récupérer le personnage avec son identifiant.
      // Exécute une requête de type SELECT avec une clause WHERE, et retourne un objet Per
sonnage.
    // Sinon, on veut récupérer le personnage avec son nom.
    // Exécute une requête de type SELECT avec une clause WHERE, et retourne un objet Perso
nnage.
  }
  public function getList($nom)
  {
    // Retourne la liste des personnages dont le nom n'est pas $nom.
    // Le résultat sera un tableau d'instances de Personnage.
  }
  public function update(Personnage $perso)
```

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

```
{
  // Prépare une requête de type UPDATE.
  // Assignation des valeurs à la requête.
  // Exécution de la requête.
}
public function setDb(PDO $db)
{
  $this->_db = $db;
}
```

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	

TP2 POO – Création d'un jeu de combat 2/4

4) Ecriture des méthodes

Une fois le contenu des méthodes défini, il ne nous reste « plus » qu'à coder les différentes fonctionnalités.

- Coder la méthode Add()
- Coder la méthode Count()
- Coder la méthode Delete()
- Coder la méthode Exists()
- Coder la méthode Get()
- Coder la méthode GetList()
- Coder la méthode Update()

Auteur(s)	Relu, validé et visé par :		Date de création :	Date dernière MAJ :
Benoît Avenel			11/2022	
			Toute reproduction, représentation, diffusion ou rediffusion, totale ou partielle, de ce document ou de son contenu par quelque procédé que ce soit est interdite sans l'autorisation expresse, écrite et préalable de l'ADRAR.	