Développement PHP
Partie 7 : Programmation Orientée Objet

DENIS LOEUILLET – IFA - 2017

Partie 7 : Programmation Orientée Objet
Les bases de la POO

1. Introduction à la POO
2. Utiliser la classe
3. L'opérateur de résolution de portée
4. Manipulation de données stockées
5. L'héritage
6. Les méthodes magiques

Partie 7 : Programmation Orientée Objet
Les bases de la POO

La programmation orientée objet, c'est un nouveau moyen d'écrire votre code.

C'est une conception inventée dans les années 1970, qui prend de plus en plus d'importance.

Ce paradigme permet une organisation plus cohérente de vos projets, une maintenance facilitée et une distribution de votre code plus aisée !

Partie 7 : Programmation Orientée Objet
Les bases de la POO

1. Introduction à la POO : qu'est-ce que la POO?

- Le procédural :
  - Actuellement vous écrivez votre code de manière procédurale qui consiste à séparer le traitement des données des données elles-mêmes.
  - Par exemple :
    - ✓ vous avez un système de news sur votre site.
    - ✓ D'un côté, vous avez les données (les news, une liste d'erreurs, une connexion à la BDD, etc.)
    - ✓ et de l'autre côté vous avez une suite d'instructions qui viennent modifier ces données.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

- La programmation orientée objet
  - La POO, c'est tout simplement faire de son site un ensemble d'objets qui interagissent entre eux.
  - En d'autres termes : tout est objet.
- Définition d'un objet
  - Dans la vie de tous les jours on en rencontre partout : un ordinateur, une lampe, une chaise …
  - En programmation, les objets sont sensiblement la même chose.
    - ✓ Exemple du personnage dans un jeu de combat :
      - ❖ Imaginons que nous ayons un objet Personnage dans notre application.
      - ❖ Un personnage a des caractéristiques :
        - Une force
        - une localisation
        - une certaine expérience
        - et enfin des dégâts

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

- Définition d'un objet
  - En programmation, les objets sont sensiblement la même chose.
    - ✓ Exemple du personnage dans un jeu de combat :
      - ❖ Toutes ses caractéristiques correspondent à des valeurs.
      - ❖ les valeurs sont stockées dans des variables.
        - C'est toujours le cas en POO.
        - Ce sont des variables un peu spéciales
      - ❖ Mis à part ces caractéristiques, un personnage a aussi des capacités. Il peut :
        - frapper un autre personnage
        - gagner de l'expérience
        - se déplacer.
      - ❖ Ces capacités correspondent à des fonctions.
      - ❖ Comme pour les variables, ce sont des fonctions un peu spéciales

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

- Définition d'un objet
  - En programmation, les objets sont sensiblement la même chose.
    - ✓ on peut donc avoir des objets dans une application.
    - ✓ d'où sortent-ils ?
    - ✓ Dans la vie réelle, un objet ne sort pas de nulle part.
    - ✓ En effet, chaque objet est défini selon des caractéristiques et un plan bien précis.
    - ✓ En POO, ces informations sont contenues dans ce qu'on appelle des classes.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

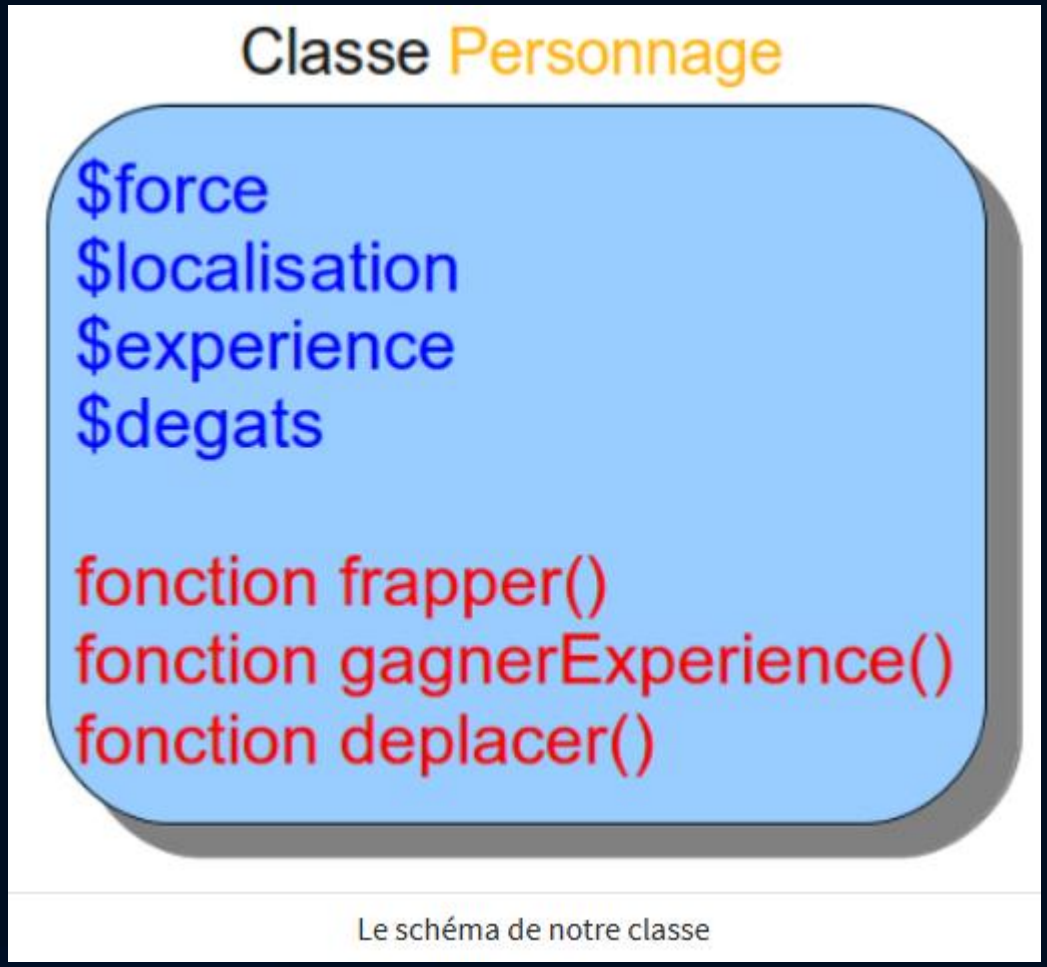
- Définition d'une classe
  - Les classes contiennent la définition des objets que l'on va créer par la suite.
  - Prenons l'exemple le plus simple du monde :
    - ✓ les gâteaux et leur moule.
      - ❖ Le moule est unique. Il peut produire une quantité infinie de gâteaux.
      - ❖ Dans ce cas-là, les gâteaux sont les objets et le moule est la classe :
        - o le moule va définir la forme du gâteau.
        - o La classe contient donc le plan de fabrication d'un objet et on peut s'en servir autant qu'on veut afin d'obtenir une infinité d'objets.
  - Une classe est une entité regroupant des variables et des fonctions.
  - Chacune de ces fonctions aura accès aux variables de cette entité.
  - Dans le cas du personnage, nous aurons une fonction frapper () :
    - ✓ Cette fonction devra simplement modifier la variable $degats du personnage en fonction de la variable $force.
  - Une classe est donc un regroupement logique de variables et fonctions que tout objet issu de cette classe possédera.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

- Définition d'une instance
  - Une instance, c'est tout simplement le résultat d'une instanciation.
  - Une instanciation, c'est le fait d'instancier une classe.
  - Instancier une classe, c'est se servir d'une classe afin qu'elle nous crée un objet.
  - En gros, une instance est un objet.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

- Exemple : création d'une classe
  - Nous allons créer une classe Personnage (sous forme de schéma).
  - Celle-ci doit contenir la liste des variables et des fonctions que l'on a citées plus haut :
    - ✓ c'est la base de tout objet Personnage.
    - ✓ Chaque instance de cette classe possédera ainsi toutes ces variables et fonctions.
  - Voici donc cette fameuse classe à la figure suivante.
    - ✓ Vous voyez donc les variables et fonctions stockées dans la classe Personnage.
    - ✓ en réalité, on ne les appelle pas comme ça, il s'agit :
    - ✓ d'attributs (ou propriétés) : Un attribut désigne une variable
    - ✓ et de méthodes : une méthode désigne une fonction.
    - ✓ Ainsi, tout objet Personnage aura ces attributs et méthodes. On pourra modifier ces attributs et invoquer ces méthodes sur notre objet afin de modifier ses caractéristiques ou son comportement.



Le schéma de notre classe

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

- Le principe d'encapsulation
  - L'un des gros avantages de la POO est que l'on peut masquer le code à l'utilisateur (l'utilisateur est ici celui qui se servira de la classe, pas celui qui chargera la page depuis son navigateur).
  - Le concepteur de la classe a englobé dans celle-ci un code qui peut être assez complexe :
    - ✓ il est donc inutile voire dangereux de laisser l'utilisateur manipuler ces objets sans aucune restriction.
    - ✓ Ainsi, il est important d'interdire à l'utilisateur de modifier directement les attributs d'un objet.
  - Prenons l'exemple d'un avion où sont disponibles des centaines de boutons.
    - ✓ Chacun de ces boutons constituent des actions que l'on peut effectuer sur l'avion.
    - ✓ C'est l'interface de l'avion.
    - ✓ Le pilote se moque de quoi est composé l'avion : son rôle est de le piloter.
    - ✓ Pour cela, il va se servir des boutons afin de manipuler les composants de l'avion.
    - ✓ Le pilote ne doit pas se charger de modifier manuellement ces composants :
      - ❖ il pourrait faire de grosses bêtises.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Qu'est-ce que la POO ?

- Le principe d'encapsulation
  - Le principe est exactement le même pour la POO :
    - ✓ l'utilisateur de la classe doit se contenter d'invoquer les méthodes en ignorant les attributs.
    - ✓ Comme le pilote de l'avion, il n'a pas à les trifouiller.
    - ✓ Pour instaurer une telle contrainte, on dit que les attributs sont privés.

Partie 7 : Programmation Orientée Objet
Les bases de la POO

1. Introduction à la POO : créer une classe

- Syntaxe de base
  - Le but de cette section va être de traduire la figure précédente en code PHP.
  - Avant cela, je vais vous donner la syntaxe de base de toute classe en PHP :

```php
<?php
class Personnage // Présence du mot-clé class suivi du nom de la classe.
{
  // Déclaration des attributs et méthodes ici.
}
```

  - Nous venons de créer le moule, le plan qui définira nos objets
  - La déclaration d'attributs dans une classe se fait en écrivant le nom de l'attribut à créer, précédé de sa **visibilité**.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Visibilité d'un attribut ou d'une méthode
  http://php.net/manual/fr/language.oop5.visibility.php
  - La visibilité d'un attribut ou d'une méthode indique à partir d'où on peut y avoir accès.
  - Nous allons voir ici trois types de visibilité :
    - ✓ public
    - ✓ private
    - ✓ protected
  - public, est le plus simple.
    - ✓ Si un attribut ou une méthode est public, alors on pourra y avoir accès :
      - depuis n'importe où :
        - depuis l'intérieur de l'objet (dans les méthodes qu'on a créées),
        - comme depuis l'extérieur

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Visibilité d'un attribut ou d'une méthode
  - public, est le plus simple.
    - ✓ Quand on crée un objet, c'est principalement pour pouvoir exploiter ses attributs et méthodes.
    - ✓ L'extérieur de l'objet, c'est tout le code qui n'est pas dans votre classe.
    - ✓ En effet, quand vous créerez un objet, cet objet sera représenté par une variable, et c'est à partir d'elle qu'on pourra :
      - ❖ modifier l'objet,
      - ❖ appeler des méthodes,
      - ❖ etc.
    - ✓ Vous allez donc dire à PHP :
      - ❖ « dans cet objet, donne-moi cet attribut »
      - ❖ ou « dans cet objet, appelle cette méthode » :
        - ○ c'est ça, appeler des attributs ou méthodes depuis l'extérieur de l'objet.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Visibilité d'un attribut ou d'une méthode
  - ➢ private, impose quelques restrictions.
    - ✓ On n'aura accès aux attributs et méthodes seulement depuis l'intérieur de la classe :
      - ❖ c'est-à-dire que seul le code voulant accéder à un attribut privé ou une méthode privée écrit(e) à l'intérieur de la classe fonctionnera.
      - ❖ Sinon, une jolie erreur fatale s'affichera disant que vous ne pouvez pas accéder à telle méthode ou tel attribut parce qu'il ou elle est privé(e).
    - ➢ le principe d'encapsulation ! C'est de cette manière qu'on peut interdire l'accès à nos attributs.

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Visibilité d'un attribut ou d'une méthode
  - protected :
    - ✓ L'accès aux éléments protégés (protected) est limité à la classe elle-même, ainsi qu'aux classes qui en héritent

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Création d'attributs
  - Pour déclarer des attributs, on va donc les écrire :
    - ✓ entre les accolades,
    - ✓ les uns à la suite des autres,
    - ✓ en faisant précéder leurs noms du mot-clé private
      - ❖ chaque attribut est précédé d'un underscore (« _ »).
      - ❖ Ceci est une notation qu'il est préférable de respecter (il s'agit de la notation PEAR) qui dit que chaque nom d'élément privé (attributs ou méthodes) doit être précédé d'un underscore.

```php
<?php
class Personnage
{
  private $_force;        // La force du personnage
  private $_localisation; // Sa localisation
  private $_experience;   // Son expérience
  private $_degats;       // Ses dégâts
}
```

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Création d'attributs
  - Vous pouvez initialiser les attributs lorsque vous les déclarez (par exemple, leur mettre une valeur de o ou autre). Exemple :

```php
<?php
class Personnage
{
  private $_force = 50;            // La force du personnage, par défaut à 50.
  private $_localisation = 'Lyon'; // Sa localisation, par défaut à Lyon.
  private $_experience = 1;        // Son expérience, par défaut à 1.
  private $_degats = 0;            // Ses dégâts, par défaut à 0.
}
```

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Création des méthodes
  - Il suffit de faire précéder le mot-clé « function » à la visibilité de la méthode.
  - Les types de visibilité des méthodes sont les mêmes que les attributs.
  - Les méthodes n'ont en général pas besoin d'être masquées à l'utilisateur :
    - ✓ vous les mettrez souvent en public : à moins que vous teniez absolument à ce que l'utilisateur ne puisse pas appeler cette méthode
    - ✓ par exemple s'il s'agit d'une fonction qui simplifie certaines tâches sur l'objet mais qui ne doit pas être appelée n'importe comment).

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Création des méthodes
  - De même que l'underscore précédant les noms d'éléments privés
    - ✓ vous pouvez remarquer que le nom des classes commence par une majuscule. Il s'agit aussi du respect de la notation PEAR.

```php
<?php
class Personnage
{
  private $_force;        // La force du personnage
  private $_localisation; // Sa localisation
  private $_experience;   // Son expérience
  private $_degats;       // Ses dégâts

  public function deplacer() // Une méthode qui déplacera le personnage (modifiera sa localisation).
  {

  }

  public function frapper() // Une méthode qui frappera un personnage (suivant la force qu'il a).
  {

  }

  public function gagnerExperience() // Une méthode augmentant l'attribut $experience du personnage.
  {

  }
}
```

Partie 7 : Programmation Orientée Objet
Les bases de la POO : introduction à la POO – Créer une classe

- Résumé
  - Une classe, c'est un ensemble de variables et de fonctions (attributs et méthodes).
  - Un objet, c'est une instance de la classe pour pouvoir l'utiliser.
  - Tous vos attributs doivent être privés. Pour les méthodes, peu importe leur visibilité. C'est ce qu'on appelle le principe d'encapsulation.
  - On déclare une classe :
    - ✓ avec le mot-clé « class » suivi du nom de la classe,
    - ✓ et enfin deux accolades ouvrantes et fermantes qui encercleront la liste des attributs et méthodes.