Architecture et dev web
TP 7 - PHP Objet

I. Comprendre les caractéristiques objets

I.1. Namespace

- Créez un dossier namespace contenant un fichier index.php pour tester votre exercice
- Créez un dossier app dans le dossier namespace contenant une classe Ville. Cette classe possède un attribut « nom », un constructeur et la redéfinition de la méthode « __toString() ». Cette classe se trouve dans le namespace app ;
- Utilisez cette classe dans le fichier index.php

Nous imaginons que vous récupérez un package nommé app2 contenant également la classe Ville. Celle-ci possède un second attribut nommé « population »

- Dupliquez le dossier app en app2 et modifiez la classe
- Créez une deuxième Ville avec cette classe
- Remédiez au conflit

- Dans la classe Ville du dossier app2, ajoutez au _toString la date de recensement en utilisant la classe DateTime. Cette classe appartient au langage php

Remarque :
Dans de nombreux projets, vous retrouverez le namespace « app » et des sous-namespace permettant d'organiser correctement votre code. Il est souvent conseillé d'associer des sous-dossiers à des sous noms de namespace

exemple : namespace app\traits ; avec un dossier traits contenant les différents traits de votre code

I.2. L'héritage

- Créez un dossier « loueur » contenant un fichier index.php et un dossier App. Ce dossier App contient 3 sous-dossiers Entity (qui contiendra les classes), un dossier Traits et un dossier Interfaces. Tous les classes seront placées dans le dossier Entity avec un espace de nom nommé App\Entity. Tous les traits auront un espace de nom App\Traits et ainsi de suite

- Créez La classe Voiture ayant les attributs $couleur, $type(diesel, essence, electrique,hybride), $immatriculation ainsi que son constructeur,les setteur, les guetteur et son __toString qui renvoie le texte « je suis une voiture »

- Créez une classe Zoe qui hérite de la classe Voiture (il s'agit d'une voiture électrique). Elle dispose du constante TYPE_ENERGY= "electrique". Sa méthode __toString() complétera la méthode __toString de sa classe mère en affichant « je suis une voiture Zoe de type électrique immatriculée ...»

- Créez dans l'index.php un objet de la classe Zoe

I.3. classe abstraite

- Modifiez la classe Voiture, pour quelle soit abstraite et qu'elle possède la méthode polluer()
- Modifiez La classe Zoe pour que la méthode polluer() renvoie la chaîne « je roule propre »

I.4. Traits

- Créez le trait Géolocalisation vu en cours
- Créez le trait Rechargeable qui contient un attribut valeurEnergie et une méthode recharger() permettant de d'assigner l'énergie à 100. Il sera également nécessaire d'avoir des getteurs et des setteurs
- Utilisez ces deux traits sur la classe Zoe

I.5. interface

- Créez une interface Vehicule qui possède les méthodes rouler(), naviguer(), voler()
- Implémentez cette interface pour la classe abstraite voiture. Les méthodes renverront soit true soit false.
- Créez une classe nommée parcZoe qui va contenir 3 voitures Zoe dans un attributs mesZoe. Ces voitures seront créées dans le constructeur. Bien sûr nous aurons normalement une méthode pouvant aller chercher les données dans un bd.
- Implémentez cette classe avec countable (vu en cours) pour permettre d'utiliser la fonction count() sur un objet de la classe parcZoe.

I.6. Statique

- Créez un méthode statique nommée pub permettant de renvoyer une phrase de publicité pour inciter l'achat de la Zoe. Soyez imaginatif

I.7. Ajout de classes

Je suis un loueur et je désire acquérir un ou plusieurs jetSki sans précision de marque. La marque sera uniquement précisée dans une variable nom à la création de l'objet

- Créez la ou les classes nécessaires.

II. Exercice

Nous souhaitons gérer certains voyages en informant nos visiteurs de notre site les activités intéressantes (restaurant, parc, monument) ainsi que les animaux(elephant, crocodile) qu'ils vont pouvoir voir lors de leur voyage.

- Concevez les classes intéressantes pour répondre à ce mini-cahier des charges. (classe abstraite et héritage). Toutes ces classes seront dans le namespace App

Tous ces éléments seront géolocalisable sur une carte.

- Réutilisez le Trait Géolocalisable

Certains de ces éléments peuvent se visiter par exemple les parcs et les monuments

- Créez une interface dans un sous dossier « interfaces » nommée Visite qui permettra de déclarer la function infoVisite() ;
- Utilisez cette interface pour les 2 classes et redéfinissez la méthode

Complétez par des initiatives de votre choix