TP n°2

Modélisation, programmation et JSON en PHP

Dans la première partie, le but est de reprendre la modélisation du TP précédent et de l'implémenter en PHP. Dans la seconde partie, nous allons exploiter JSON pour stocker les informations dans des fichiers.

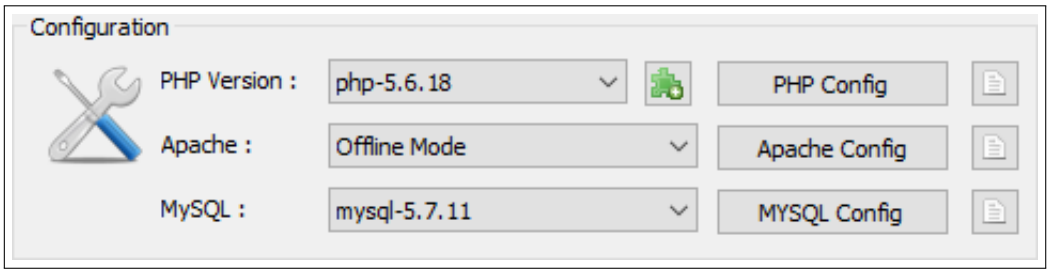
1 Configuration de votre environnement de travail

Pour ce TP, vous aurez besoin d'un serveur Web avec un moteur PHP. Dans les salles machine, il y a deux solutions : démarrer les machines sous *Windows* ou sous *Linux*.

1.1 PHP sous *Windows*

Une solution consiste à utiliser la distribution *uWamp* (qui ne fonctionne que sous *Windows*). Elle comprend notamment un serveur *Apache* et ne nécessite pas d'installation (elle fonctionne depuis une clef USB, par exemple). Téléchargez-la depuis le site officiel au format zip et dézippez l'archive sur votre compte (ou sur une clef). Pour démarrer, exécutez le programme `UwAmp.exe`.

Par défaut, c'est la version PHP 5.6 qui est démarrée, comme illustré sur la capture ci-dessous :



Cliquez sur le bouton à côté de la version pour installer la dernière version possible. Sélectionnez le dépôt *UwAmp PHP dépot*, cochez la version php-7.2.7 et cliquez sur le bouton *Installer*. Une fois cette version installée, cliquez sur le bouton *Fermer*. Maintenant, vous pouvez sélectionner cette version. Pour finir, vérifier que les erreurs sont bien affichées : cliquez sur le bouton *PHP Config*. Dans l'onglet *PHP Setting*, cliquez sur *Display errors*. Vérifiez que la valeur est bien *On*, sinon modifiez-la.

Comme pour une distribution classique, les scripts PHP doivent être placés dans le répertoire `www` de l'application. Une fois le serveur démarré, testez directement son bon fonctionnement depuis l'adresse locale (`http://localhost/`).

1.2 PHP sous *Linux*

Par défaut, PHP est installé sur les comptes. Vous pouvez le vérifier en exécutant la commande `php`. Pour démarrer un serveur Web, placez-vous simplement dans le répertoire de votre choix (celui qui contient vos scripts), puis tapez la commande suivante :

```
php -S localhost:8080
```

Maintenant, vérifiez son bon fonctionnement depuis votre navigateur : `http://localhost:8080/`.

2 Retour sur les voitures

Reprenez l'ensemble des classes que vous avez développées en *Java* et traduisez-les en PHP. Pour le moment, vous pouvez écrire un simple script pour tester vos classes.

3 JSON et PHP

Pour rappel, pour manipuler du JSON en PHP, il n'est pas nécessaire d'installer ou d'utiliser une bibliothèque externe. Les méthodes `json_encode` et `json_decode` sont suffisantes. Cependant, nous aurons besoin d'écrire des méthodes supplémentaires pour personnaliser la sérialisation et la désérialisation pour les objets.

Questions

1. À partir des classes écrites dans l'exercice précédent, créez des objets et affichez le JSON correspondant.
2. Comme vu en cours, il est possible d'ajouter la méthode `jsonSerialize` qui est déclarée dans l'interface `JsonSerializable`. Faites implémenter cette interface par votre classe `Voiture` (ou équivalente) puis ajoutez la méthode `jsonSerialize` et vérifiez que la personnalisation est fonctionnelle.
3. À partir de la chaîne de caractères contenant le JSON de votre objet `Voiture`, effectuez la désérialisation. Qu'obtient-on ? Comment peut-on obtenir un objet de type `Voiture` ?
4. La fonction `json_decode` permet de convertir un document JSON en tableau associatif (ou en objet générique). Créer une méthode `fromJSON` dans la classe `Voiture` qui crée un objet `Voiture` à partir d'un tableau associatif.

4 En route vers le projet n°1

Nous souhaitons pouvoir convertir l'ensemble des objets des classes de l'exercice 2 pour les stocker dans des fichiers et les transférer à une autre application.

Questions

1. Proposez une architecture de fichiers pour stocker vos données.
2. Proposez des structures pour chaque fichier JSON
3. Ajoutez toutes les méthodes nécessaires dans vos classes.