Programmation orientée objet (UAA14)

Exercices : série 8

Objets d'apprentissage :

- ✔ Modéliser une logique de programmation orientée objet.
- ✔ Déclarer une classe.
- ✔ Instancier une classe.
- ✔ Utiliser les méthodes de l'objet instancié.
- ✔ Traduire un algorithme dans un langage de programmation.
- ✔ Commenter des lignes de code.
- ✔ Tester le programme conçu.
- ✔ Caractériser les attributs dans une classe (encapsulation).
- ✔ Caractériser les méthodes dans une classe (encapsulation).
- ✔ Décrire la création d'un constructeur.
- ✔ Extraire d'un cahier des charges les informations nécessaires à la programmation.
- ✔ Programmer en recourant aux instructions et types de données nécessaires au développement d'une application.
- ✔ Corriger un programme défaillant.
- ✔ Développer une classe sur la base d'un cahier des charges en respectant le paradigme de la programmation orientée objet.
- ✔ Programmer en recourant aux classes nécessaires au développement d'une application orientée objet.
- ✔ Améliorer un programme pour répondre à un besoin défini.

Exercice 1

L'objectif de cet exercice est de construire une page HTML contenant un court texte décrivant le fonctionnement d'Ajax. Sur la page, l'utilisateur pourra obtenir la définition plusieurs mots propres à Ajax par un simple clic. En cas de clic, le navigateur demandera la définition correspondante au serveur et l'affichera sur la page actuelle (**sans charger une nouvelle page**).

Étape 1

Crée une page HTML contenant le code suivant :

```
<!doctype html>
<html>
   <head>
      <meta charset="UTF-8" />
      <style>
         mark {
         border-bottom: 1px dashed blue;
         background-color: transparent;
         }
         footer {
         min-height: 20px;
         background-color: beige;
         }
      </style>
   </head>
   <body>
      <h1>Cliquez sur les mots soulignés en pointillés pour obtenir une
         définition.
      </h1>
      <p>Dans une application Web, la méthode <mark>classique</mark> de
         dialogue entre un <mark>navigateur</mark> et un
<mark>serveur</mark> est
         la suivante : lors de chaque manipulation faite par l'utilisateur,
le
         <mark>navigateur</mark> envoie une <mark>requête</mark> contenant
une
         référence à une page Web, puis le <mark>serveur</mark> Web
effectue des
         calculs, et envoie le résultat sous forme d'une page Web à
destination du
         <mark>navigateur</mark>. Celui-ci affichera alors la page qu'il
vient de
         recevoir. Chaque manipulation entraîne la
<mark>transmission</mark> et
         l'affichage d'une nouvelle page. L'utilisateur doit attendre
l'arrivée de
         la réponse pour effectuer d'autres manipulations.
      </p>
      <p>En utilisant <mark>Ajax</mark>, le dialogue entre le
         <mark>navigateur</mark> et le <mark>serveur</mark> se déroule la
plupart
         du temps de la manière suivante : un programme écrit en langage de
         programmation <mark>JavaScript</mark>, incorporé dans une page
web, est
         exécuté par le <mark>navigateur</mark>. Celui-ci envoie en
```

```
arrière-plan
         des <mark>demandes</mark> au serveur Web, puis modifie le contenu
de la
         page actuellement affichée par le <mark>navigateur</mark> Web en
fonction
         du résultat reçu du <mark>serveur</mark>, évitant ainsi la
transmission
         et l'affichage d'une nouvelle page complète.
      </p>
      <footer>
         <p id="pDefinition"></p>
      </footer>
      <script>
         let pDef = document.getElementById("pDefinition");
      </script>
   </body>
</html>
```

Observe attentivement le contenu de la page. Remarque que c'est dans le paragraphe identifié par `#pDefinition` que la définition du mot doit apparaître.

Étape 2

Crée une seconde page (en PHP cette fois ci) nommée `definitions.php`.

Cette page contient uniquement un script PHP définissant un tableau associatif nommé `$defs`. Voici sa définition :

```php
<?php
$defs = [
    "classique" => "habituelle",
    "serveur" => "machine où le site web est hébergé",
    "navigateur" => "programme qui permet de naviguer sur le web",
    "Ajax" => "Asynchroneous Javascript and XML"
];
```

Complète ce script de telle sorte que si on tente d'accéder à `localhost/definitions.php?mot=navigateur` celui-ci affiche (via **`echo`**) la définition correspondant au mot « navigateur ».

Si le mot demandé n'est pas trouvé, le message « Aucune définition trouvée » doit s'afficher.

Si aucun mot n'est donné, le message « Veuillez fournir un mot à définir » doit s'afficher.

Étape 3

Reviens au document HTML. Fais en sorte que lors du survol d'un mot marqué par la souris, celui-ci soit écrit sur un fond gris.

Étape 4

Ajoute maintenant du code JS. Fais en sorte que lors du clic sur un mot, le message « Vous demandez la définition du mot XXX. » s'affiche dans le paragraphe `#pDefinition`.

Étape 5

Il est temps de faire de l'Ajax !

Fais en sorte qu'en cas de clic :

1. le navigateur (la page HTML) envoie une demande de définition au script PHP ;
2. au moment où la réponse est reçue, la définition s'affiche dans le paragraphe `#pDefinition`

> ⓘ Le fichier `definitions.php` doit absolument se trouver sur le serveur (vu qu'il faut un interpréteur PHP pour l'exécuter). Par contre, le fichier HTML ne doit pas obligatoirement se trouver sur le serveur : on pourrait très bien le placer directement sur le disque de dur de l'ordinateur ou encore sur une clé USB !

Étape 6

Jusqu'à présent, les données échangées se limitent uniquement à un texte simple. Mais que faire maintenant si l'on souhaite échanger *plusieurs* informations pour une même définition ?

Au lieu de renvoyer un texte simple, on peut envoyer des informations structurées au format JSON.

Dans un premier temps, modifie la définition du tableau `$defs` pour que chaque mot soit associé à un tableau contenant :

- la clé « def » associée à la définition du mot ;
- la clé « auteur » associée à l'auteur de la définition (au choix) ;
- la clé « source » associée à la source d'où provient la définition (ex. Wikipédia, dictionnaire, etc.).

Étape 7

Modifie maintenant ton script PHP de telle sorte que les trois informations soient transmises à la page HTML lorsque cette dernière exécute une demande de définition. Pour cela, utilise la fonction **`json_encode`**.

Étape 8

Reviens sur ta page HTML et modifie l'affichage proposé lors du clic sur un mot. L'affichage doit devenir :

```
MOTDÉFINI
… définition … (par auteur)
Source : source
```

Références

Les présents exercices ont été élaborés à l'aide des ressources suivantes :

- https://www.pierre-giraud.com/php-mysql-apprendre-coder-cours/introduction-programmation-orientee-objet/
- Cours de « Développement d'applications WEB », Hénallux (2019)