



Conception & Développement Informatique

PHP OBJET - Exercices

PHP, Concept Objet, UML

Mickaël DEVOLDÈRE

MD v1.0.1





17/04/2018







<http://www.arfp.asso.fr>

PHP OBJET - Exercices

PHP, Concept Objet, UML

Technologies et langages	
	HTML
	CSS
	Javascript
	PHP

Légende des icônes	
	Information complémentaire
	Point d'attention particulier
	Intervention du formateur possible
	Lien vers une ressource externe

Sommaire

Préparation du projet pour les exercices.....	3
1 - Calcul d'impôts.....	3
2 – Identification et session	4
3 – La classe MyString	5
4 - Validation de formulaire HTML.....	6
Historique du document	8
Crédits	8

Préparation du projet pour les exercices

Créer un nouveau projet nommé « php_objet » dans C:\wamp64\www.

Le chemin du projet sera donc :

C:\wamp64\www\php_objet\

et sera accessible en mode http via le navigateur à l'adresse :

http://localhost:8080/php_objet/

Chaque exercice de ce document doit être stocké dans un répertoire à l'intérieur du projet.

Par exemple, pour l'exercice « calcul d'impôts », créer le répertoire C:\wamp64\www\php_objet\calcul_impots\

L'exercice sera alors accessible en mode HTTP à l'adresse : http://localhost:8080/php_objet/calcul_impots/



Les noms de répertoire ne doivent pas contenir de caractère accentué, de caractère spécial ou d'espace !

1 - Calcul d'impôts

Dans le projet « php_objet », créer un nouveau répertoire nommé « calcul_impots ». Les fichiers de l'exercice seront à placer dans ce dossier.

Dans un fichier index.php créer un formulaire contenant une zone de saisie pour le nom, le revenu et un bouton OK pour soumettre le formulaire.

Le but de ce formulaire est de permettre le calcul de l'impôt pour une personne. Le taux de l'impôt est de 9% pour des revenus inférieurs à 15 000 euros et de 14 % pour des revenus supérieurs à 15 000.

Pour les revenus supérieurs à 15 000 euros, la part de revenus inférieure à 15 000 euros est taxée à 9% et la part de revenus supérieure à 15 000 euros est taxée à 14%.

Une fois le formulaire envoyé grâce au bouton OK, une phrase s'affiche du style « Mr Devoldère votre impôt est de 600 euros ».

Le programme doit obligatoirement contenir une classe « Impot » dans un fichier « Impot.php » avec un constructeur (__construct), une méthode AfficheImpot() qui retourne une chaîne contenant le message à afficher (« Mr Devoldère votre impôt est de 600 euros » par exemple) et une méthode CalculImpot() qui retourne le montant de l'impôt suivant le revenu.

Les taux de l'impôt (14% et 9%) devront être définis dans des constantes.

La création de l'objet et l'appel des méthodes doivent être réalisés dans un fichier resultatImpot.php qui sera la cible de l'action du formulaire créée dans index.php.

2 – Identification et session

Dans le projet « php_objet », créer un nouveau répertoire nommé « identification ». Les fichiers de l'exercice seront à placer dans ce dossier.

Dans une page index.php créer un formulaire avec une zone de saisie pour le nom d'utilisateur, le mot de passe et un bouton OK. On considère que si l'utilisateur rentre le nom « admin » et le mot de passe « Cdi1234 » il est connecté. Si le mot de passe ou le login est incorrect renvoyer un message d'erreur.

Dès que l'utilisateur est connecté il est renvoyé vers une nouvelle page « info.php » qui contient une liste des dernières actualités. La page « info.php » contient un lien « déconnecter » pour se déconnecter. Un clic sur « déconnecter » renvoie vers le formulaire de connexion (index.php).

Si l'utilisateur n'est pas connecté il n'a pas le droit de voir les actualités de la page « info.php », il doit être renvoyé vers la page de login (index.php).

3 – La classe MyString

Si vous avez déjà codé en Javascript vous aurez certainement vu l'objet « string » et les méthodes qui permettent de le manipuler (comme: length, charAt(), indexOf(), substring()...)

L'objectif de cet exercice est de créer une classe PHP nommée « MyString » qui contient les méthodes du même nom que celles de Javascript et qui font les mêmes tâches. Autrement dit, à l'aide de la classe que vous aurez créé, quelqu'un qui s'y connaît en Javascript peut bien traiter les chaînes de caractères en PHP même s'il n'a pas de connaissances poussées en ce langage.

Pour récapituler voilà ce que font les méthodes que l'on va créer :

Nom de l'attribut/méthode	Utilité
length	Cet attribut retourne le nombre de caractères contenus dans la chaîne.
charAt()	La méthode charAt(x) (avec A majuscule) permet de retourner le caractère qui se trouve à la position x passé en paramètre. Le paramètre est un entier qui commence de 0. La valeur 0 indexe le premier caractère de la chaîne.
indexOf()	La méthode indexOf(car) (avec O en majuscule) permet de retourner la position du caractère car passé en paramètre. Si le caractère existe dans la chaîne, alors sa position (comprise entre 0 et la longueur de la chaîne - 1) est retournée, sinon (le caractère ne figure pas dans la chaîne) alors la valeur -1 est retournée. La méthode indexOf() peut accueillir un deuxième paramètre qui est un entier qui indique à partir de quel position de la chaîne on commencera la recherche du caractère passé en premier paramètre.
substring()	La méthode substring(début,fin) permet d'extraire une partie de la chaîne de caractères commençant de la position début et finissant à la position fin-1.
split()	La méthode split(occurrence) permet de retourner un tableau à partir des fractions de la chaîne de caractères obtenues en divisant celle-ci au niveau de l'occurrence.
toLowerCase()	La méthode toLowerCase() permet de retourner la chaîne de caractère entièrement en minuscules.
toUpperCase()	La méthode toUpperCase() permet de retourner la chaîne de caractère entièrement en majuscules.

Créez la classe MyString et testez son fonctionnement dans un script PHP.



L'attribut \$length doit être déclaré « protected ».

Pour y accéder de l'extérieur de la classe, nous utiliserons la méthode magique __get()

<http://php.net/manual/fr/language.oop5.magic.php>

4 - Validation de formulaire HTML

L'objectif de cet exercice est de créer et implémenter une classe qui permettra de valider un formulaire HTML.

Créer un fichier « FormValidation.php » qui contiendra la classe FormValidation.

Le constructeur de la classe FormValidation accepte un paramètre de type array qui contiendra la liste des noms des champs à valider.



Le nom d'un champ de formulaire HTML correspond à la valeur de l'attribut « name ».

Exemple :

<input type='text' id='firstname' name='prenom'> : Le nom du champ est « prenom ».

Le tableau respectera cette forme :

```
Array
(
    'nom_du_champ' => 'type_de_données', // pour valider un certain type de données
    'un_autre_champ' => null, // pour accepter tout type de données
)
```

La classe FormValidation contient au moins 2 méthodes publiques en plus du constructeur :

- isFilled() vérifie que les champs obligatoires du formulaire sont bien renseignés
- isValid() vérifie que les champs respectent le format attendu (string, entier...)

Exemple d'utilisation de la classe FormValidation pour valider un formulaire contenant les champs « nom » (texte) et « revenus » (nombre entier):

```
<?php

$champs_requis = array
(
    'nom' => 'string',
    'revenus' => 'int',
)

$form = new FormValidation($champs_requis);

if($form->isFilled() && $form->isValid())
{
    // Le formulaire est validé
}
else
{
    // Le formulaire n'est pas validé
}
```


--- Fin du document ---

Historique du document

Auteur	Date	Observations
Mickaël DEVOLDÈRE	17/04/2018	Création du document
Mickaël DEVOLDÈRE	24/04/2018	Correction Taux exercice 1

Crédits

<http://php.net>