W1 – Piscine PHP

Jour 05

Responsables Pédagogiques

pedagowac@epitech.eu

Bienvenue dans cette cinquième journée de piscine Web@cadémie ! ☺

Au cours de cette journée nous allons aborder les variables superglobales.

Sommaire

Exercice 01 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_01.php

Restrictions : Aucune

Créez une fonction nommée « declare_globals » qui créé les variables globales suivantes :

- a = “hello”
- b = “world”
- c = “le”
- d = “monde”
- e = “n’est”
- f = “que”
- g = “PHP”

Prototype : void declare_globals(void);

Exercice 02 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_02.php

Restrictions : Aucune

Créez une fonction « my_aff_global » qui affiche toutes les variables globales seulement si elles ne sont pas de type array. Le format sera : ”[nom] => [valeur]\n”.

Prototype : void my_aff_global(void);

Exercice 03 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_03.php

Restrictions : Aucune

Créez une fonction qui met dans un cookie la clé et la valeur données en paramètre. La valeur devra être modifiée pour rajouter “swag” à la fin.

Prototype : void my_add_to_cookie(string $key, string $value);

Exemple :

my_add_to_cookie(“pseudo”, “Max_”);

// Crée un cookie à la clef “pseudo” et a pour contenu “Max_swag”.

Exercice 04 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_04.php

Restrictions : Aucune

Écrire une fonction qui affiche le contenu du cookie possédant la clef définie par le premier paramètre, suivi d'un retour à la ligne. Si le cookie n'existe pas, rien n'est affiché.

Prototype : void my_print_cookie(string $key);

Exercice 05 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_05.php

Restrictions : Aucune

Créez une fonction qui met dans une session la clé et la valeur données en paramètre. La valeur devra être modifiée pour rajouter “swag” à la fin.

Prototype : void my_add_to_session(string $key, string $value);

Exemple :

my_add_to_session(“pseudo”, “Max_”);

// Crée une variable de session à la clef “pseudo” ayant pour contenu “Max_swag”

Exercice 06 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_06.php

Restrictions : Aucune

Écrire une fonction qui affiche le contenu de la session possédant la clef définie par le premier paramètre, suivi d'un retour à la ligne. Si la session n'existe pas, rien n'est affiché.

Prototype : void my_print_session(string $key);

Exercice 07 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_07.php

Restrictions : Aucune

Créez une fonction qui vide la session, la détruit et la réinitialise.

Prototype : void my_reset_session(void);

Exercice 08 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_08.php

Restrictions : Aucune

Créez une fonction qui retourne la valeur de l'élément de POST possédant la clé "helix" s'il existe ou NULL en cas d'erreur.

Prototype : mixed helix_post_finder(void)**;**

Exercice 09 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_09.php

Restrictions : Aucune

Créez une fonction qui renvoie la distance de levenshtein entre l'élément “string_one” et “string_two” de POST. En cas d’erreur la fonction retourne NULL.

Prototype : mixed post_levenshtein_score(void);

Exemple :

Si on a dans POST :
{
“string_one” => “Sophie”,
“string_two” => “Julien”
}

post_levenshtein_score() renvoie : 5

Exercice 10 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_10.php

Restrictions : Aucune

Créez une fonction qui affiche un certain nombre d'éléments du GET et renvoie 0. Ce nombre est lui-même passé en paramètre à GET avec la clef “nbr”. En cas d’erreur, retourne NULL.

Prototype : mixed my_get_weird_info(void);

Exemple :

Si mon GET contient :

$_GET[‘nbr’] = 5
$_GET[‘var1’] = “premier test”
$_GET[‘var2’] = “deuxième test”
$_GET[‘var3’] = “dernier test”
$_GET[‘directrice’] = “Sophie”
$_GET[‘responsable’] = “Julien”
$_GET[‘pangolin1’] = “Samy”
$_GET[‘pangolin2’] = “Henri”

my_get_weird_info() affiche :

$_GET[‘nbr’] = 5
$_GET[‘var1’] = “premier test”
$_GET[‘var2’] = “deuxième test”
$_GET[‘var3’] = “dernier test”
$_GET[‘directrice’] = “Sophie”

Exercice 11 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_11.php

Restrictions : Aucune

Écrire une fonction qui affiche toutes les clés et valeurs du tableau passé en paramètre sous la forme « clé : valeur ». Un retour à la ligne sera effectué après chaque affichage.

Prototype : void print_array_with_key(array $my_array);

Exercice 12 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_12.php

Restrictions : Aucune

Créez une fonction utilisant la superglobale SERVER pour déterminer avec précision le temps qu'a pris votre script php à s'exécuter. Elle retourne ce temps en secondes avec une précision de 5 chiffres après la virgule.

Prototype : int get_execution_time(void);

Exemple :

Si le script a mis 0.0021234342s à s'exécuter, la fonction retourne :
0.00212

Si le script a mis 0.0013382s à s'exécuter, la fonction retourne :
0.00134

Si le script a mis 0.00091923s à s'exécuter, la fonction retourne :
0.00092

Exercice 13 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_13.php

Restrictions : Aucune

Créez une fonction "my_sort_files" qui prend en paramètre une référence d'un tableau de chaines de caractères, puis qui trie les valeurs par ordre alphabétique décroissant. Elle doit ensuite afficher chaque valeur du tableau suivi d'un retour à la ligne.

Prototype : void my_sort_files(array &$tab);

Exercice 14 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_14.php

Restrictions : Aucune

Créer une fonction "display_names" qui retourne un tableau contenant :

- le nom du script courant à l'index « 0 »
- le nom du répertoire courant à l'index « 4 »

Il n'est pas attendu des chemins, mais bien seulement des noms.

Prototype : array display_names(void);

Exercice 15 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_15.php

Restrictions : Aucune

Créer une fonction "convert_number" qui affiche la valeur ASCII du nombre donné en paramètre. Si le nombre donné en paramètre est supérieur à 100, la fonction affichera la chaine de caractères : "Vive les pangolins". Tout affichage se termine par un retour à la ligne.

Prototype : void convert_number(int $nbr);

Exercice 16 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_16.php

Restrictions : Aucune

Écrire une fonction "check_types" qui prend un tableau à deux dimensions en paramètre et qui vérifie la correspondance entre les clés et les tableaux de valeurs associés. La fonction retourne TRUE si toutes les valeurs rencontrées correspondent aux bons types, sinon FALSE.

Prototype : void check_types(array $types);

Exemple :

```
$types = ["integer"=> [5121, 454, -5464],
          "double" => [22.0, 262.65, -54.54],
          "NULL"   => [NULL, NULL, NULL],
          "string" => ["a", "a", "ab"],
          "object" => [new stdClass()]];
check_types($types); // return true

$types = ["array"    => [[]]];
check_types($types); // return true

$types = ["float"    => [2.0]];
check_types($types); // return false
```

Exercice 17 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_17.php

Restrictions : Aucune

Créez une fonction "check_url" qui devra vérifier que les paramètres envoyés au script via la méthode HTTP « GET » contiennent une clé "token".

La valeur associée à la clé "token" doit être une chaine dont chaque caractère doit appartenir à la "whiteList" passée en paramètre de la fonction (sous forme de tableau).

Si la valeur du token" est valide, la fonction retourne 1, sinon elle devra retourner 0.

Attention à bien vérifier l'existence de la clé "token" parmi les données GET, si elle n'existe pas la fonction doit retourner FALSE.

Prototype : mixed check_url(array $whiteList);

Exemple :

```
// avec comme valeur de token : acbeacddf
$whiteList = array("a", "b", "c", "d", "e", "f");
check_url($whiteList); // return 1

// avec comme valeur de token : 4ea3fcdf1
$whiteList = array("a", "b", "c", "d", "e", "f");
check_url($whiteList); // return 0
```

Exercice 18 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_18.php

Restrictions : Aucune

Créez une fonction "check_acl" qui vérifie les permissions d'un utilisateur donné.

Votre fonction devra prendre 2 paramètres :

- un tableau contenant les informations d'un utilisateur

- la permission dont il faut vérifier la présence

La fonction doit retourner TRUE si l'utilisateur possède la permission, sinon FALSE.

Prototype : bool check_acl(array $user, string $permission);

Exemple :

```
$user = ["login"       => "deslog_m",
         "permissions" => ["launch_pangolinette", "troll_student"]];


check_acl($user, "launch_pangolinette"); // retourne TRUE
check_acl($user, "kill_kittens"); // retourne FALSE
```

Exercice 19 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_19.php

Restrictions : Aucune

Créez une fonction "get_authorized_users" qui affiche le ou les logins suivis chacun d'un saut de ligne dont les utilisateurs ont les permissions demandées.

La fonction accepte deux paramètres :

- un tableau d'utilisateurs

- la permission dont il faut vérifier l'existence

Prototype : void get_authorized_users (array $users, string $permission);

Exemple :

```
$users = [["login"       => "deslog_m",
           "permissions" => ["launch_pangolinette", "troll_student"]],
          ["login"       => "pellet_f",
           "permissions" => ["launch_pangolinette", "troll_student"]],
          ["login"       => "student_x",
           "permissions" => ["offer_sweets_to_pangolin"]]];

get_authorized_users($users, "launch_pangolinette"); // deslog_m\npellet_f\n
get_authorized_users($users, "offer_sweets_to_pangolin"); // student_x\n
```

Exercice 20 (1 pts)

Nom du rendu : Piscine_PHP_Jour_05/ex_20.php

Restrictions : Aucune

Créez un script qui affiche "email valide\n" si les données envoyées au script contiennent un email correctement formaté, sinon la fonction affichera "email invalide\n".

Les données sont envoyées au script par un formulaire via la méthode POST, la valeur à vérifier se trouvera dans le champ "email".

L'email sera considéré valide (correctement formaté) s'il répond à ses règles :

- ne comporte ni espace, ni tabulation

- comporte exactement un caractère "@"

- comporte au moins un caractère "." rencontré après le caractère "@"

Prototype : void get_authorized_users (array $users, string $permission);

Exemple :

```
// avec comme valeur d'email "pedago-samsung@epitech.eu"
user@exam> php ex_20.php
email valide
user@exam>

// avec comme valeur d'email "je troll@les_pangolins"
user@exam> php ex_20.php
email invalide
user@exam>
```