TP 4
Formulaires et pages interactives

Internet et Outils (IO2)

Dans les TPs precedents nous avons introduit PHP et vu des exemples de passage d'informations d'une page a l'autre (GET). Ce TP a pour objet de consolider ces deux aspects, d'une part en organisant mieux le code PHP (fonctions), et d'autre part en progressant dans l'interaction avec le visiteur. Les pages que vous allez creer a cette seance contiendront des *formulaires* HTML qui permettront aux internautes d'envoyer des informations au serveur. Des scripts PHP vous permettront ensuite de traiter ces informations et d'y reagir. Les elements des formulaires pourront être des champs de texte, des cases a cocher, des options a selectionner, des chiers a telecharger...

1 Organisation du PHP

L'idee est d'utiliser au maximum des fonctions ayant un nom explicite. Ainsi pour des fonctions dont les noms seraient `estPersonneConnue(...)` ou `afficheContact(...)` ce qu'on attendra du code suivant sera assez limpide :

```
if (estPersonneConnue("toto"))  {afficheContact("toto");}
```

1.1 De nir ses propres fonctions

Une fonction peut être de nie n'importe ou dans un script, y compris apres avoir ete utilisee. Toutefois, vos fonctions seront regroupees dans la partie haute de votre chier PHP a n de faciliter le travail de comprehension de code. Le mot-cle `return` permet d'arrêter l'execution du code de la fonction et de renvoyer une valeur. Il n'est pas obligatoire.

Voici un exemple de de nition d'une fonction :

```
<?php
function nom_de_la_fonction($x,$y,...){
  ...  // code de la fonction...
  return $var; // renvoie la valeur de $var (facultatif)
}
?>
```

Exercice 1 *Chaîne de caractères comme valeur de retour de la fonction*

1. *Vous vous rappelez certainement que lors du TP précédent certains accès au tableau* GET *produisaient des erreurs car les entrées associées n'étaient pas définies. Écrivez une fonction* PHP *qui prend en argument une chaine de caractères et qui retourne la valeur associée dans* GET *si elle existe.*

Elle retournera la chaine "inconnue" dans le cas contraire. Testez votre fonction.

2. *Écrire une fonction* PHP *qui prend en argument une url, et un tableau, et qui construise une adresse passant en argument* GET *tout le contenu du tableau à l'url donné. Testez votre fonction en prenant l'url d'un TP précédent qui permettait d'afficher le contenu de* GET*.*
3. *Écrire une fonction* PHP *qui produit une chaine de caractères correspondant au code html d'une liste dont les éléments sont les valeurs d'un tableau de nom* `$tab` *passé en argument. Utilisez cette fonction pour produire 2 ou 3 listes à partir de différents tableaux* PHP*.*
4. *Écrire une fonction php qui prend en argument un array d'arrays, et qui produise un tableau html, dont chaque ligne représente l'un des arrays, et dont le contenu est mis en colonnes. (Ecrivez une fonction auxilliaire pour produire une ligne). A la suite du test de la question précédente vous essayerez votre fonction.*

2 Passage de parametres par un formulaire

2.1 Les Formulaires simples (Rappel des TP 2 et 3)

Dans les TP precedents nous avons vu comment faire des formulaires en HTML. Cette semaine nous allons voir comment utiliser les parametres passes par l'utilisateur via ce formulaire. Nous rappelons ici comment faire un formulaire minimal (pour plus de details vous pouvez consulter les TP precedents).

La balise `<form>` permet de delimiter une zone dans laquelle peuvent se trouver plusieurs entrees d'un même formulaire. Les di erents elements d'un formulaire sont ensuite declares a l'aide de l'element `input`. Notez qu'a l'instar de balises comme `br` ou `img`, cet element est \vide". Il dispose en revanche de nombreux attributs qui vont permettre de speci er sa nature et son comportement. L'attribut principal que doit contenir tout `input` est son `type`. La valeur `"text"` permet de de nir une zone dans laquelle le visiteur devra entrer du texte. La valeur `"submit"` permet quant a elle de creer un bouton servant a valider les informations renseignees dans l'ensemble du formulaire.

Exercice 2 *Un formulaire isolé*
Créez une page `formulaire.html` *contenant un formulaire avec les éléments suivants :*

- *Un champs de texte dans lequel on demande au visiteur son nom.*
- *Un bouton de validation.*

Constatez que, pour l'instant, rien d'utile ne se passe lorsque vous appuyez sur le bouton de validation.

2.2 Des formulaires qui servent a quelque chose

Dans l'exercice precedent les donnees entrees etaient perdues. L'objectif est maintenant de faire traiter ces donnees par un script (par exemple un script PHP). Il faudra deux choses pour cela :

{ Le formulaire doit contenir dans un attribut `action` le nom du script charge de gerer les donnees recueillies.

- Une cle doit être attribue a chaque balise `input` dont vous voudrez recuperer le resultat (grâce a l'attribut `name`).

D'apres tout ce qui precede, un formulaire doit donc avoir la structure :

```
<form action="script_de_traitement.php">
      Login: <input type="text" name="login" />
      <input type="submit" value="Valider" />
</form>
```

Une fois ceci fait, lorsque vous cliquez sur le bouton de validation du formulaire le script est passe avec les couples cle-valeur correspondant a chaque entree du formulaire. Par defaut ce passage de parametres se fait par la methode GET que vous avez vue au TP precedent.

Exercice 3 *Récupération des éléments d'un formulaire, méthode* GET

1. *Modifiez votre page* `formulaire.html` *afin que ses résultats soient traités par un script* `affichage.php`*, et attribuez des clés à toutes les entrées utiles.*
2. *Créez la page* `affichage.php`*, avec un paragraphe affichant le nom entré par le visiteur (par exemple la page pourra afficher "Bonjour Machin" si Machin est le nom fourni par l'utilisateur).*
3. *Pour rendre votre fichier plus lisible, créez un fichier* `bonjour_affichage.php` *qui contient le code qui permet d'afficher votre message de bienvenue personnalisé. Ensuite avec* `include` *appelez ce fichier pour appliquer le message dans la page* `affichage.php`*.*
4. *organisez votre code d'une façon différente, sans* `include` *mais avec une fonction. Avez vous une préférence ?*

Pour des formulaires plus importants, la methode GET a un certain nombre de limites. Le passage de parametres par la barre d'adresse restreint le type des parametres ainsi que leur taille (imaginez une image). Le procede est en outre inadapte a la transmission d'informations sensibles (vous passeriez un mot de passe ?). La seconde methode de passage de parametres, POST, va permettre de mieux traiter ce genre de cas.

- Un formulaire envoie ses donnees par POST si son attribut `method` est de ni a la valeur `"post"` (de même cet attribut peut recevoir la valeur `"get"`). Ainsi `<form>` devient `<form action="script_de_traitement.php" method="post">`.
- Un script accede aux parametres passes par POST par l'intermediaire du tableau associatif `$_POST`, qui fonctionne de même que `$_GET`. Il faut donc comme toujours attribuer une cle aux entrees que vous voulez recuperer !

Exercice 4 *Récupération des éléments d'un formulaire, méthode* POST

1. *Transformez* `formulaire.html` *et* `affichage.php` *pour que la transmission se fasse par la méthode* POST*.*
2. *Observez les différences entre les deux méthodes lors de la validation du formulaire.*

Remarque : Il existe un tableau associatif qui par defaut contient la même chose que l'ensemble des variables `$_GET`, `$_POST` (et `$_COOKIE` que nous n'utilisons pas encore). Il s'agit de `$_REQUEST`.

Exercice 5 *Choisir le style d'affichage*

1. *Créez trois styles d'affichage pour le fichier* `affichage.php` *(vous pouvez par exemple créer trois fichiers CSS différents). Par exemple une avec un fond vert et une police orange (pour les personnes avec un mauvais goût), une avec une police plus grande (pour les malvoyants) et une classique (sans style).*
2. *Ajoutez ensuite un menu déroulant dans* `formulaire.html` *qui permettra à l'utilisateur de choisir avec quel style il veut voir s'afficher les résultats dans* `affichage.php`*. Modifiez aussi* `affichage.php` *pour qu'il respecte l'affichage demandé.*
3. *Ecrivez à présent un fichier* `formulaire.php` *ayant le même comportement que* `formulaire.html`*, mais étant plus général. Dans ce fichier, le menu déroulant sera le produit d'une fonction php, fonction qui travaillera sur un tableau contenant tout les fichiers de style que l'on souhaite.*

3 Un formulaire complet

Les elements des formulaires HTML peuvent être tres varies. Vos ressources preferees vous fourniront toutes les valeurs que peut recevoir l'attribut `type` de la balise `input`. D'autres attributs comme `value` et d'autres balises comme `select`, `textarea` et `label` vous seront egalement utiles.

Exercice 6 *Des entrées de tous types*
Complétez votre page `formulaire.php` *pour en faire un formulaire de renseignement complet. Il devra demander les informations suivantes :*

- *Nom (champ texte, nom du champ* `nom`*),*
- *Prénom (nom du champ* `prenom`*),*
- *Date de naissance (champ date, nom du champ* `date`*)*
- *Adresse électronique (nom du champ* `mail`*),*
- *Mot de passe (champ texte masqué, nom du champ* `pass`*),*
- *Confirmez le mot de passe (champ texte masqué, nom du champ* `pass2`*),*
- *Outil préféré :* HTML, PHP, *ou* CSS *(liste déroulante, nom du champ* `outil`*),*
- *Vous aimez ce cours ? oui ou non (choix exclusifs, valeur par défaut oui, nom du champ* `cours`*),*
- *Qu'aimeriez-vous faire ensuite ? un forum, un site de partage d'images, un réseau social (options non exclusives à cocher, noms des champs* `projet1`*,* `projet2` *et* `projet3`*, une case de votre choix cochée par défaut),*
- *Commentaires (zone de texte active de 17 colonnes et 10 lignes, nom du champ* `comment`*),*
- *Un bouton de réinitialisation avec l'étiquette "RAZ" (un bouton de type* `reset`*),*
-

2. *Modifiez* `affichage.php` *pour donner un compte-rendu des principales informations (non sensibles) renseignées.*

Exercice 8 *Question subsidiaire, revenez-y à la fin*
Donnez un nom unique à tous les champs `projet*` *et observez ce qui se passe quand plusieurs options sont cochées.*
Maintenant, donnez un nom unique à tous les champs `projet*`*, mais en vous assurant de pouvoir ensuite récupérer les valeurs de toutes les cases cochées. Modifiez le script d'affichage en conséquence. Vous allez sans doute devoir utiliser des tableaux et des boucles* `foreach` *(pensez par exemple à remplacer les noms* `projet*` *par* `projet[]` *et observez ce qui se passe).*

4 Traitement des donnees du formulaire

Exercice 9 *Retour à* PHP
Modifiez le fichier `affichage.php` *afin qu'il ne se contente plus de retranscrire brutalement tous les paramètres qui lui sont passés. On veut en plus vérifier les choses suivantes :*

- *les champs essentiels ont bien été remplis,*
- *les deux occurrences du mot de passe sont identiques,*

Ces tests se feront par l'intermédiaire de fonctions php, une pour chaque test que l'on souhaite effectuer. Leur type retour sera booléen. Que faut-il leur passer en paramètre ? Faites en sorte qu'en cas de problème, la page `affichage.php` *affiche un avertissement et un lien de retour au formulaire. Au contraire, remerciez le visiteur si tout va bien.*

Exercice 10 *Résultats du bac*

1. *Créez une page* `bac.html` *avec un formulaire qui demande les notes au bac des matières suivantes : Français, Maths, Histoire-géo, EPS et Anglais. Cette page doit aussi contenir un bouton qui appelle le fichier* `bac.php` *de la question suivante (cf. figure 1).*
2. *Le fichier* `bac.php` *doit afficher les notes précédentes en deux listes séparées (une pour les notes supérieures ou égales à 10, une pour les notes strictement inférieures).*
   Vous pourrez créer une liste d'association ayant les cinq matières pour clés et les notes associées pour valeurs. Vous pourrez parcourir cette liste d'association avec la syntaxe `foreach($tableau as $cle => $valeur)` *vue au TP précédent. Vous écrirez cette portion de code dans une fonction* `affiche_notes($notes)`
3. *En réalité les seuils de séparation des notes peuvent varier selon ce qu'on souhaite regarder. En effet dans certains cas, une note entre 7 et 10 pourrait donner lieu à un ratrapage, une note inférieure à 7 serait éliminatoire, et une note inférieure à 2 pourrait donner lieu à un entretien. Modifiez votre fonction d'affichage pour mettre en valeur ces différents seuils. Pour cela vous pourrez écrire une fonction* `filtre_valeurs` *qui prendrait en argument un tableau de valeurs* `$v`*, une valeur de seuil* `$lim`*, et un sens* `$sens` *de filtre. Il retournerait alors un tableau composé des valeurs de* `$v`*, supérieures au seuil* `$lim` *si* `$sens` *est positif. (Inférieures à* `$lim` *si* `$sens` *est négatif).*

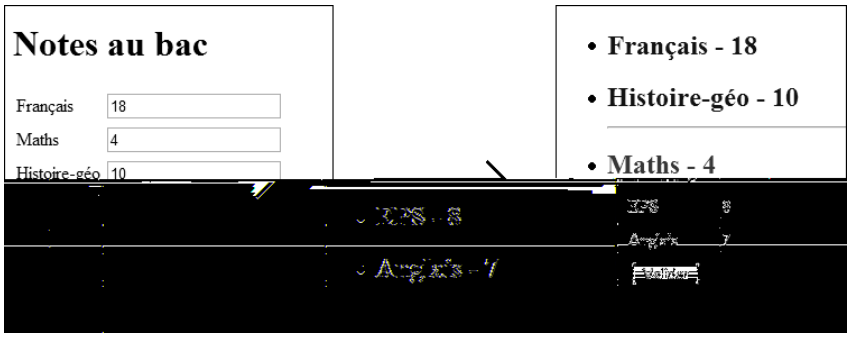

Figure 1 { Un exemple de formulaire rempli et du resultat correspondant

4. *Améliorez votre script, en écrivant une fonction dans* `bac.php` *qui se charge d'afficher un message d'erreur et un bouton ramenant à la page* `bac.html` *dans le cas où l'une des notes rentrées par l'utilisateur n'est pas un nombre (fonction* `is_numeric`*) compris entre 0 et 20.*
5. *Pour finir, écrivez des fonctions* `moyenne($notes)`, `mention($notes)` *et* `affiche_resultat($notes)` *pour que le script affiche la moyenne générale (on suppose qu'il n'y a pas de coefficients) et le résultat (recalé, rattrapage, reçu avec mention passable, assez bien, bien ou très bien).*

Exercice 11 *Référendum*
Pour cet exercice, vous aurez besoin de boucles `for`*. Elles fonctionnent en* PHP *de la même façon qu'en* Java*, avec la syntaxe* `for ( ... ; ... ; ...) { ... }`*. Par exemple le programme*

```
<?php
  for ($i=1; $i<=6; $i++) {
    echo $i;
  }
?>
```

va afficher `123456`*.*

1. *Créez un fichier* `referendum.php` *qui affiche la liste des électeurs avec en face de chacun un choix Oui/Non (sous forme de bouton radio). Le nombre d'électeurs affiché dépendra d'un paramètre* `nbe` *passé en argument dans l'URL (méthode* `GET`*). (Un exemple est donné sur la figure 2.)*
2. *Pour ne pas avoir à cocher de nombreux boutons lors de tests avec de grandes valeurs pour* `nbe`*, on donne une valeur par défaut aux boutons radios (attribut* `checked`*) dépendant de la parité du numéro du candidat ("Non" pour les candidats de numéro impair et "Oui" pour les candidats de numéro pair).*
3. *Faites en sorte que, lorsqu'on appuie sur le bouton "Afficher le résultat", le résultat du référendum soit affiché en pourcentage.*
   On prendra soin de choisir des noms adéquats pour les attributs `name` *des boutons radios de manière à rendre cette tâche facilement automatisable (par exemple "radio-1", "radio-2", etc.). Il suffira alors de parcourir le tableau* `POST` *pour toutes ces valeurs jusqu'à "radio-nbe" et de comptabiliser les votes.*

Figure 2 { Un referendum avec 5 electeurs

4. *(Question bonus à faire seulement si vous avez tout terminé). Ajoutez une représentation graphique du résultat sous forme de deux barres dont la taille est proportionnelle aux scores pondérés du "Oui" et du "Non".*