

Les superglobales Get, Post et Files

Table des matières

I. Contexte	3
II. Transmettre grâce aux formulaires HTML	3
III. Exercice : Appliquez la notion	5
IV. Méthode GET	7
V. Exercice : Appliquez la notion	9
VI. Les limites de GET	10
VII. Exercice : Appliquez la notion	11
VIII. Méthode POST	13
IX. Exercice : Appliquez la notion	16
X. Envoi de fichiers	19
XI. Exercice : Appliquez la notion	22
XII. Auto-évaluation	24
A. Exercice final	24
B. Exercice : Défi	26
Solutions des exercices	29

I. Contexte

Durée : 1 h

Environnement de travail : Repl.it

Pré-requis : Bases de PHP, HTML

Contexte

Les variables superglobales `GET`, `POST` et `FILES` sont mises à disposition nativement par PHP.

Elles sont particulières du fait de leur syntaxe et de leur utilisation. Celles-ci sont accessibles directement dans le code, et seront utiles dans le contexte de la transmission de données entre les pages d'un site web et ses utilisateurs.

II. Transmettre grâce aux formulaires HTML

Objectifs

- Comprendre comment les formulaires HTML permettent l'interaction entre l'utilisateur d'un site web et le serveur
- Identifier comment sont liées les données d'un formulaire et les données reçues par PHP

Mise en situation

Dans la majorité des cas, ce sont les formulaires HTML qui vont permettre la transmission de données dans un site web.

Nous allons voir comment configurer ce transit d'informations.

Rappel

En HTML, un formulaire est représenté par un ensemble d'éléments, que sont les balises et les attributs HTML.

Un formulaire :

- est déclaré grâce aux balises `<form></form>`
- contient des champs, balises `<input />` ou `<select></select>` par exemple
- dispose d'un bouton permettant de déclencher la soumission du formulaire
- possède une méthode d'envoi, attribut `method=""` qui peut prendre la valeur `get` ou `post`
- est configuré pour une action donnée, attribut `action=""` stipulant le script PHP où seront envoyées et traitées les données

Exemple

```
1 <form method="post" action="script.php">
2   <label for="firstname">Firstname :</label>
3   <input type="text" id="firstname" name="firstname" value="" />
4   <label for="country">Country :</label>
5   <select id="country" name="country">
6     <option value="France">France</option>
7     <option value="Espagne">Espagne</option>
```

```

8     <option value="Italie">Italie</option>
9   </select>
10  <button type="submit">Send</button>
11 </form>

```

Firstname: Country:

L'attribut name

Lorsqu'un champ de formulaire est configuré, cet attribut est important, car c'est sa valeur qui l'identifiera dans le script PHP cible.

C'est la valeur de cet attribut qui définira la clé du tableau, `$_POST` ou `$_GET`.

L'attribut value

La valeur de cet attribut présent dans la définition de chaque champ permettra de définir la valeur associée à la clé `name` dans le tableau `$_POST` ou `$_GET`.

Exemple

Considérons le code ci-dessous comme étant extrait d'un formulaire soumis via la méthode `post` :

```
1 <input type="text" id="address" name="address" value="Rue des formulaires" />
```

Lorsque ce formulaire sera traité par le serveur, nous pourrons alors afficher la valeur transmise sous cette forme :

```
1 echo $_POST['address']; // Affichera "Rue des formulaires";
```

La superglobale `$_POST` contient une valeur "Rue des formulaires" associée à la clé "address".

Transmettre un tableau

Dans certains cas, il est nécessaire de pouvoir transmettre plusieurs données pour un même champ de formulaire. C'est par exemple le cas lorsqu'un *select multiple* ou plusieurs cases à cocher sont présents dans un formulaire.

Afin de pouvoir transmettre l'intégralité des valeurs sélectionnées, il convient de rajouter des crochets `[]` à la fin de la valeur de l'attribut `name`.

Exemple

Prenons pour exemple un formulaire qui permettrait de transmettre à un script une liste de langages.

```

1 <form method="post" action="script.php">
2   <input type="checkbox" name="languages" value="php" />PHP
3   <input type="checkbox" name="languages" value="javascript" />Javascript
4   <input type="checkbox" name="languages" value="perl" />PERL
5   <button type="submit">Send</button>
6 </form>

```

☐ PHP ☐ Javascript ☒ PERL

Dans le cas ci-dessus, l'attribut `name` a été écrit sans crochets.

En procédant ainsi, même si l'utilisateur a coché plusieurs éléments, la valeur de `$_POST['languages']` contiendra uniquement la valeur de la dernière case cochée.

```
1 print_r($_POST['languages']); // Affichera "perl".
```

Le fait de définir correctement l'attribut `name` pour chaque input : `name="languages [] "` permettra cette fois d'obtenir un tableau avec toutes les données.

```
1 print_r($_POST['languages']); // Affichera Array ( [0] => php [1] => javascript [2] => perl ).
```

Syntaxe À retenir

- Les données d'un formulaire HTML sont traitées par le script défini dans l'attribut `action` et sont transmises selon la méthode définie dans l'attribut `method`.
- Qu'importe la méthode, les données seront stockées dans un tableau associatif ayant pour clés les valeurs des différents attributs `name` du formulaire soumis. Les valeurs correspondront quant à elles aux attributs `value`.
- Ces valeurs sont stockées dans les superglobales `$_GET` et `$_POST` selon la méthode choisie pour transmettre les informations.
- Dans certains cas, il est nécessaire d'ajouter `[]` à la fin de l'attribut `name` d'un champ de formulaire, afin de permettre la transmission de données multiples.

Complément

Formulaire HTML¹

III. Exercice : Appliquez la notion

Vous disposez du formulaire HTML suivant :

```
1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8" />
5 </head>
6 <body>
7 <form>
8     <div>
9         <input type="radio" id="Mr" name="civilite" value="Mr">
10        <label for="Mr" class="inline-label">Monsieur</label>
11    </div>
12    <div>
13        <input type="radio" id="Mme" name="civilite" value="Mme">
14        <label for="Mme" class="inline-label">Madame</label>
15    </div>
16
17    <label for="name">Votre nom</label>
18    <input type="text" name="name" id="name" placeholder="Saisissez votre nom">
19
20    <label for="birthDate">Votre date de naissance</label>
21    <input type="date" name="birthDate" id="birthDate" placeholder="Saisissez votre date de
naissance">
22
23    <label for="comment">Commentaire</label>
24    <textarea rows="4" name="comment" id="comment" placeholder="Ajoutez un commentaire
pertinent"></textarea>
25
```

¹ <https://developer.mozilla.org/fr/docs/Web/HTML/Element/Form>

```

26 <label for="email">Votre adresse e-mail</label>
27 <input type="email" name="email" id="email" placeholder="Saisissez votre e-mail">
28
29 <label for="languages">Langages de programmation maîtrisés</label>
30 <select name="languages" id="languages" multiple>
31   <option value="HTML">HTML</option>
32   <option value="JS">JS</option>
33   <option value="PHP">PHP</option>
34   <option value="Python">Python</option>
35   <option value="SQL">SQL</option>
36 </select>
37
38 <label for="tos" class="inline-label">J'accepte les conditions générales
d'utilisation</label>
39 <input type="checkbox" name="tos" id="tos">
40
41 <button type="reset">Réinitialiser les valeurs du formulaire</button>
42 <button type="submit">Soumettre le formulaire</button>
43 </form>
44 </body>
45 <style>
46 /**
47  Attention, ce CSS est là uniquement pour rendre le formulaire "agréable" à la lecture sans
48  que vous n'ayez
à récupérer deux fichiers distincts.
49  Dans un cas d'usage "réel", ces éléments doivent être externalisés
50  */
51  body {
52    font-family: Calibri, serif;
53  }
54
55  form {
56    max-width: 50%;
57  }
58
59  form label {
60    display: block;
61    font-weight: bold;
62    margin-bottom: 10px;
63  }
64
65  label.inline-label {
66    display: inline-block;
67  }
68
69  fieldset {
70    border: 1px solid lightgray;
71    background-color: rgba(225, 233, 255, 0.25);
72  }
73
74  legend {
75    font-style: italic;
76    font-size: 1.1em;
77    padding: 5px;
78  }
79
80  form input, form select, form textarea {
81    display: inline-block;
82    margin-bottom: 10px;

```

```
83     padding: 10px;
84     width: 80%;
85 }
86
87 form input[type="radio"],
88 form input[type="checkbox"],
89 form input[type="submit"] {
90     width: auto;
91 }
92
93 button[type=submit], button[type=reset] {
94     padding: 10px;
95     margin-top: 15px;
96 }
97 </style>
98 </html>
99
```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question

[solution n°1 p.31]

Vous allez devoir permettre sa soumission en définissant ses attributs `method` et `action`, qui ont été oubliés.

Nous souhaitons que le formulaire soit soumis grâce à la méthode `POST` vers un script intitulé **traitement-formulaire.php**.

Permettez également au champ permettant une sélection multiple d'envoyer l'ensemble des valeurs sélectionnées.

IV. Méthode GET

Objectifs

- Envoyer des données via l'URL
- Manipuler la variable superglobale `$_GET`

Mise en situation

La première méthode que nous allons voir, la plus directe, est `GET`. Il s'agit de la partie émergée de l'iceberg concernant la transmission des données entre les pages d'un site web.

Les données peuvent transiter d'une page à l'autre par l'intermédiaire de l'URL. Avec cette méthode, elles sont inscrites et donc visibles dans l'URL.

Syntaxe Avec des paramètres

Les paramètres seront ajoutés à la fin de l'URL et seront séparés de la partie principale par un "?". Si plusieurs données sont transmises, elles seront séparées entre elles par des "&" : `https://url.php?firstname=john&lastname=doe&age=31`.

1 <https://repl.it/>

- `https://url.php` est l'URL dans son format initial, c'est l'URL du script qui recevra les données.
- `?` indique que des paramètres vont être envoyés.
- `firstname=john` est un premier paramètre ayant pour nom `firstname` et pour valeur `john`.
- `&` indique un autre paramètre.

Ces données vont avoir un but précis, afficher des informations, afficher une partie de la page et en cacher d'autres. Elles sont modifiables dans l'URL, mais le comportement du site pourrait dans ce cas être erroné.

Exemple

L'appel de l'URL `https://developer.mozilla.org/fr/search?q=create+object+url`¹ affiche par exemple le résultat d'une recherche concernant les termes *create object url* dans la documentation MDN.



Nous pouvons aisément modifier la requête pour effectuer une recherche sur les formulaires `https://developer.mozilla.org/fr/search?q=forms`.

Au même titre, nous pourrions également essayer de modifier le nom du paramètre `q` pour voir ce que cela provoque : `https://developer.mozilla.org/fr/search?param=create+object+url`.



Manipulation en PHP

Les données transmises par l'URL seront réceptionnées par le script cible et stockées dans la variable superglobale `$_GET`².

Il s'agit d'un **tableau associatif** dans lequel les clés correspondent aux noms des paramètres.

¹ <https://developer.mozilla.org/fr/search?q=create+object+URL>

² <https://www.php.net/manual/fr/reserved.variables.get.php>

Exemple

```

1 <html>
2   <head>
3     <title>Superglobale $_GET</title>
4   </head>
5   <body>
6     <a href="/script.php?firstname=john&lastname=doe&age=31">Envoi des données à
script.php</a>
7   </body>
8 </html>

```

Le lien ci-dessus permet d'envoyer les données `firstname`, `lastname` et `age` à `script.php`.

Dans ce script, on pourrait ensuite imaginer vouloir afficher le contenu de `$_GET`. Cela se traduirait de cette façon :

```

1 <?php
2 // script.php.
3 print_r($_GET); // Affichera Array ( [firstname] => john [lastname] => doe [age] => 31 )
4 echo $_GET['lastname']; // Affichera 'doe'.
5 ?>

```

Ces données seraient alors utilisables au besoin.

Syntaxe **À retenir**

- Les données sont passées d'une page à une autre via l'URL grâce aux symboles `?` et `&`
- `$_GET`¹ est la superglobale qui stocke les données de l'URL
- L'accès à un paramètre est possible avec la syntaxe : `$_GET['_parametre_']`

V. Exercice : Appliquez la notion

Vous disposez du script suivant, intitulé **methode-get.php** :

```

1 <?php
2
3 if (isset($_GET['nom']) && isset($_GET['age']))
4 {
5     echo sprintf("Bonjour, mon nom est %s et j'ai %s ans. <br/>", $_GET['nom'], $_GET['age']);
6 } else {
7     echo 'Je souhaite afficher mon nom et mon âge. Avez-vous oublié quelque chose ? <br/>';
8 }
9

```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question 1

[solution n°2 p.32]

Exécutez ce script et déterminez l'URL qui permettra d'afficher l'ensemble des valeurs présentes sur cette page.

¹ <https://www.php.net/manual/fr/reserved.variables.get.php>

² <https://repl.it/>

Question 2

[solution n°3 p.32]

Modifiez le script ci-dessus afin qu'un nouveau paramètre permette d'afficher si l'utilisateur est administrateur ou non.

VI. Les limites de GET

Objectif

- Comprendre les dangers de GET

Mise en situation

GET est la méthode la plus simple pour envoyer des données d'une page à l'autre. C'est aussi la plus dangereuse, car ces données sont visibles et facilement modifiables.

Un seul mot d'ordre : ne faites jamais confiance aux données reçues !

Exemple	Modifications des paramètres de l'url
<p>Reprenons l'URL suivante : <code>https://index.php?firstname=john&lastname=doe&age=31</code>.</p> <p>Sur la page index.php, nous afficherons un texte utilisant ces variables :</p> <pre> 1 <?php 2 echo 'Je suis ' . \$_GET['firstname'] . ' ' . \$_GET['lastname'] . ' , ' . \$_GET['age'] . ' ans'; 3 // Affichera 'Je suis john doe, 31 ans'. 4 ?> </pre> <p>Maintenant, si nous modifions ces paramètres :</p> <pre> 1 https://index.php?firstname=31&lastname=john&age=doe 2 3 // Affichera Je suis 31 john, doe ans. </pre> <p>Si nous supprimons l'un des paramètres, ici age :</p> <pre> 1 https://index.php?firstname=john&lastname=doe 2 3 // index.php indiquera l'erreur suivante : PHP Notice: Undefined index: age in /home/runner/formulaires/script.php on line 2 </pre>	

Attention
<p>La modification des paramètres de l'URL peut entraîner des comportements inattendus ou des erreurs au niveau du code PHP.</p>

Limite de taille d'une URL

Il faut également savoir que la taille d'une URL peut être limitée. Les normes n'indiquent pas de taille limite, car celle-ci provient des navigateurs ou serveurs web. Par exemple, Internet Explorer limite la longueur d'une URL à 2048 caractères.

Si le besoin est de transmettre de nombreux paramètres, la méthode GET peut vite s'avérer inappropriée.

L'URL pourrait être tronquée, des données seraient alors manquantes... Ce qui aurait pour effet de déclencher des erreurs dans le code PHP cible, si celui n'effectue pas un minimum de contrôle sur les informations transmises.

Insertion en base de données

Imaginons que les données transmises par l'URL à un script soient destinées à être insérées dans la base de données.

Un utilisateur qui enregistrerait cette URL dans ses favoris ou qui rafraîchirait la page de manière incessante pourrait lancer plusieurs requêtes de création, modification ou suppression, sans vraiment en être conscient.

Pire, cette URL deviendrait un allié de force à toute personne mal intentionnée qui aurait l'idée d'appeler l'URL un très grand nombre de fois.

Attention

Dans la pratique, la manipulation de la base de données avec la méthode `GET` est à éviter. Son utilisation doit se concentrer principalement sur l'affichage avec les contrôles appropriés.

Complément Information textuelle

L'utilisation de la méthode `GET` permet seulement la transmission de données textuelles. Nous ne pourrions, par exemple, pas transmettre de fichiers par l'URL.

Syntaxe À retenir

Il est déconseillé d'utiliser la méthode `GET` lorsque :

- les paramètres trop facilement modifiables,
- des données sont manipulées,
- beaucoup de données doivent être envoyées.

Il n'est pas possible de transmettre des fichiers avec cette méthode.

Son usage doit donc être maîtrisé et essentiellement limité à de l'affichage et de la récupération de données.

VII. Exercice : Appliquez la notion

L'administration de votre site, dont vous trouverez le script ci-dessous, est actuellement accessible à toute personne désireuse d'accéder à la page **administration.php**.

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8"/>
5 </head>
6 <body>
7 <section>
8     <h1>Administration de votre site</h1>
9     <p class="greetings">
10         <?php
11             echo sprintf("Bienvenue sur l'administration de votre site %s", $_GET['name']);
12         ?>
13     </p>
14     <p>
15         Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
16         incididunt ut labore et
17         magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi
18         ut aliquip ex ea

```

```

18         commodo
19         consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum
dolore eu fugiat nulla
20         pariatur
21     </p>
22     <table>
23         <thead>
24             <tr>
25                 <td>Nom</td>
26                 <td>Prénom</td>
27                 <td>Action</td>
28             </tr>
29         </thead>
30         <tbody>
31             <tr>
32                 <td>Lorem</td>
33                 <td>Ipsum</td>
34                 <td><a href="#">Supprimer</a></td>
35             </tr>
36             <tr>
37                 <td>John</td>
38                 <td>Doe</td>
39                 <td><a href="#">Supprimer</a></td>
40             </tr>
41             <tr>
42                 <td>John</td>
43                 <td>Doe</td>
44                 <td><a href="#">Supprimer</a></td>
45             </tr>
46         </tbody>
47     </table>
48 </section>
49
50 </body>
51 <style>
52 /**
53     Attention, ce CSS est là uniquement pour rendre le formulaire "agréable" à la lecture sans
que vous n'ayez
54     à récupérer deux fichiers distincts.
55     Dans un cas d'usage "réel", ces éléments doivent être externalisés
56     */
57     body {
58         font-family: Calibri, serif;
59     }
60
61     h1, h2 {
62         text-align: center;
63     }
64
65     table {
66         border-collapse: collapse;
67         width: 50%;
68         margin: auto;
69     }
70
71     table, th, td {
72         border: 1px solid black;
73     }
74

```

```
75  head {
76      background-color: beige;
77      font-weight: bold;
78      text-align: center;
79  }
80
81  p {
82      width: 50%;
83      margin: auto auto 10px;
84  }
85
86  p.greetings {
87      background-color: ivory;
88      font-weight: bold;
89      font-size: 16px;
90      padding: 5px;
91      border: 1px solid gray;
92  }
93
94 </style>
95 </html>
96
```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question

[solution n°4 p.33]

Vous devez limiter cet affichage aux personnes autorisées uniquement.

Pour cela, modifiez ce script pour n'afficher le contenu de la page qu'aux administrateurs.

- On considérera qu'une personne est administratrice lorsqu'elle accède à la page grâce à l'URL **administration.php?admin=1**
- Si la personne n'est pas autorisée à accéder à la page, affichez un message d'erreur
- Cette page affiche également un message de bienvenue à l'utilisateur en affichant son nom, récupéré grâce à un paramètre `name` dans l'URL
- Faites en sorte de sécuriser l'affichage si cette valeur n'est pas présente

Exemples :

- `http://localhost/administration.php?admin=0` : Vous n'êtes pas autorisé à accéder à la page
- `http://localhost/administration.php?admin=1` : Vous êtes autorisé à accéder à la page, mais vous ne voyez pas le message de bienvenue
- `http://localhost/administration.php?admin=1&name=laurent` : Vous êtes autorisé à accéder à la page et vous voyez le message de bienvenue

Vous remarquerez avec quelle facilité il est possible d'afficher ou de masquer la section d'administration dans cet exemple. C'est pour cela que vous ne **devez pas** vous fier à ce qui vous est renvoyé dans une URL pour des informations sensibles.

VIII. Méthode POST

1 <https://repl.it/>

Objectifs

- Envoyer des données via un formulaire
- Manipuler \$_POST

Mise en situation

La seconde méthode que nous allons voir est `POST`. Nous n'utiliserons plus l'URL, mais des formulaires HTML, grâce auxquels les données transiteront à l'abri des regards indiscrets.

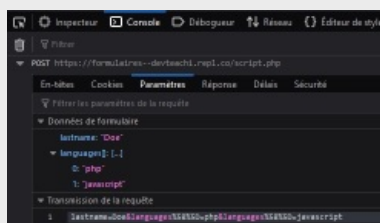
Différence avec GET

Comme indiqué, les données sont transmises de manière masquée pour l'utilisateur. Elles sont en fait accessibles dans les outils **Console** et **Réseaux** du navigateur.

Soit le formulaire :

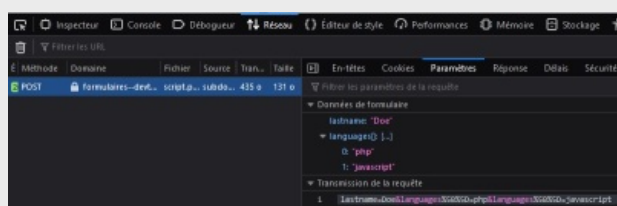
Lastname : ☒ PHP ☒ Javascript ☐ PERL

Exemple Dans la console



Dans l'onglet **Paramètres** de la requête HTTP POST.

Exemple Dans l'onglet réseaux



Dans l'onglet **Paramètres** de la requête HTTP POST.

Taille des données

Avec `POST`, il n'y a pas de limite de taille des données à envoyer. En pratique, les serveurs web peuvent configurer une limite en nombre de champs à poster. Celle-ci est bien entendu modifiable.

Complément

Cette valeur se trouve généralement dans le fichier **php.ini** du serveur.

Type de données

Contrairement à GET, il n'y a pas de limite de type de données à envoyer. Grâce à POST, il sera possible d'envoyer des fichiers.

Manipulation en PHP

Les données transmises par un formulaire seront réceptionnées par le script cible et stockées dans la variable superglobale \$_POST. Comme pour GET, c'est un **tableau associatif** dans lequel les clés sont le nom des paramètres.

Exemple

```
1 <html>
2 <head>
3   <title>Superglobale $_POST</title>
4 </head>
5 <body>
6   <form method="post" action="script.php">
7     <label for="lastname">Lastname :</label>
8     <input type="text" id="lastname" name="lastname" value="" />
9     <input type="checkbox" name="languages[]" value="php" />PHP
10    <input type="checkbox" name="languages[]" value="javascript" />Javascript
11    <input type="checkbox" name="languages[]" value="perl" />PERL
12    <button type="submit">Send</button>
13  </form>
14 </body>
15 </html>
```

Le formulaire ci-dessus permet d'envoyer les données lastname et languages[] à **script.php**. En affichant le contenu de \$_POST, nous obtenons :

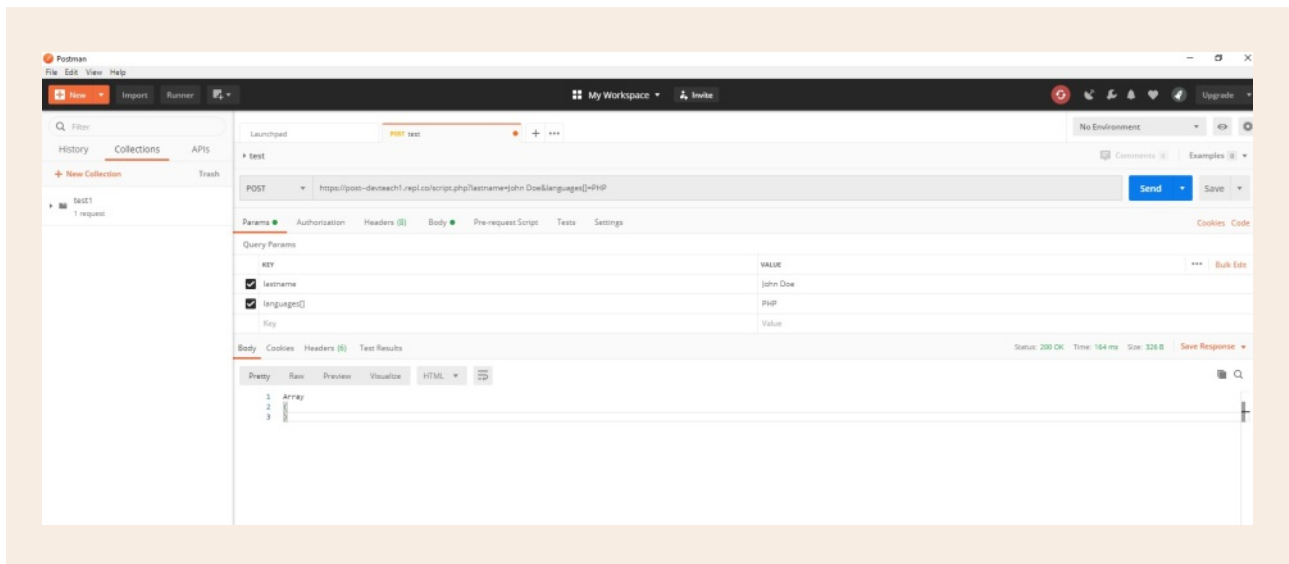
```
1 <?php
2 // script.php.
3 print_r($_POST); // Affichera Array ([lastname] => doe [languages] => Array ([0] => php [1] =>
4 javascript))
5 echo $_POST['lastname']; // Affichera 'doe'.
6 ?>
```

Rappel

Les clés de \$_POST sont les valeurs des attributs name du formulaire HTML.

Complément

Il est difficile de tester une requête post, car les données passées ne sont pas accessibles directement via le navigateur. L'utilisation d'outils comme Postman est conseillée. Il vous permettra de simuler tous types de requêtes.



Syntaxe À retenir

- Nous indiquons au formulaire d'envoyer les données en post grâce à l'attribut `method="post"`.
- La superglobale `$_POST` permet la manipulation de ces données : `$_POST['parametre']`.

Complément

`$_POST`¹

`max_input_vars`²

Postman³

IX. Exercice : Appliquez la notion

Vous disposez du formulaire HTML suivant :

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8" />
5 </head>
6 <body>
7 <form method="post" action="traitement-formulaire.php">
8     <div>
9         <input type="radio" id="Mr" name="civilite" value="Mr">
10        <label for="Mr" class="inline-label">Monsieur</label>
11    </div>
12    <div>
13        <input type="radio" id="Mme" name="civilite" value="Mme">
14        <label for="Mme" class="inline-label">Madame</label>
15    </div>
16
17    <label for="name">Votre nom</label>
18    <input type="text" name="name" id="name" placeholder="Saisissez votre nom">

```

1 <https://www.php.net/manual/fr/reserved.variables.post.php>

2 <https://www.php.net/manual/fr/info.configuration.php#ini.max-input-vars>

3 <https://www.postman.com/>


```

19
20     <label for="birthDate">Votre date de naissance</label>
21     <input type="date" name="birthDate" id="birthDate" placeholder="Saisissez votre date de
naissance">
22
23     <label for="comment">Commentaire</label>
24     <textarea rows="4" name="comment" id="comment" placeholder="Ajoutez un commentaire
pertinent"></textarea>
25
26     <label for="email">Votre adresse e-mail</label>
27     <input type="email" name="email" id="email" placeholder="Saisissez votre e-mail">
28
29     <label for="languages">Langages de programmation maîtrisés</label>
30     <select name="languages[]" id="languages" multiple>
31         <option value="HTML">HTML</option>
32         <option value="JS">JS</option>
33         <option value="PHP">PHP</option>
34         <option value="Python">Python</option>
35         <option value="SQL">SQL</option>
36     </select>
37
38     <label for="tos" class="inline-label">J'accepte les conditions générales
d'utilisation</label>
39     <input type="checkbox" name="tos" id="tos">
40
41     <button type="reset">Réinitialiser les valeurs du formulaire</button>
42     <button type="submit">Soumettre le formulaire</button>
43 </form>
44 </body>
45 <style>
46 /**
47     Attention, ce CSS est là uniquement pour rendre le formulaire "agréable" à la lecture sans
que vous n'ayez
48     à récupérer deux fichiers distincts.
49     Dans un cas d'usage "réel", ces éléments doivent être externalisés
50     */
51     body {
52         font-family: Calibri, serif;
53     }
54
55     form {
56         max-width: 50%;
57     }
58
59     form label {
60         display: block;
61         font-weight: bold;
62         margin-bottom: 10px;
63     }
64
65     label.inline-label {
66         display: inline-block;
67     }
68
69     fieldset {
70         border: 1px solid lightgray;
71         background-color: rgba(225, 233, 255, 0.25);
72     }
73

```

```

74     legend {
75         font-style: italic;
76         font-size: 1.1em;
77         padding: 5px;
78     }
79
80     form input, form select, form textarea {
81         display: inline-block;
82         margin-bottom: 10px;
83         padding: 10px;
84         width: 80%;
85     }
86
87     form input[type="radio"],
88     form input[type="checkbox"],
89     form input[type="submit"] {
90         width: auto;
91     }
92
93     button[type=submit], button[type=reset] {
94         padding: 10px;
95         margin-top: 15px;
96     }
97 </style>
98 </html>
99

```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question

[solution n°5 p.35]

Créez le script PHP permettant le traitement de ce formulaire.

Ce script devra afficher les valeurs contenues dans les champs. Vérifiez que chacune des clés est définie avant de l'afficher.

Vous pouvez également définir une fonction pour éviter la répétition de code.

Voici l'affichage visé :

```

La valeur de la clé civilite est : Mr
La valeur de la clé name est : Justine
La valeur de la clé birthDate est :
La valeur de la clé comment est :
La valeur de la clé email est :
La clé languages n'a pas été définie
La clé tos n'a pas été définie

```

1 <https://repl.it/>

X. Envoi de fichiers

Objectifs

- Envoyer un fichier via un formulaire
- Manipuler un fichier dans un script PHP

Mise en situation

Un formulaire HTML permet d'envoyer tous types de données, y compris des fichiers. Nous l'utilisons de plus en plus dans le cadre de la dématérialisation des données.

Formulaire HTML

Au niveau du formulaire, deux actions sont à réaliser :

- ajouter l'attribut `enctype="multipart/form-data"`
- déclarer un champ `input` de type **file**

Exemple

```
1 <body>
2   <form method="post" enctype="multipart/form-data">
3     <label for="lastname">Lastname :</label>
4     <input type="text" id="lastname" name="lastname" value="" /><br />
5     <label for="cni">Upload ID Card :</label>
6     <input type="file" id="cni" name="cni" />
7   </form>
8 </body>
```

Remarque

L'attribut `enctype` sert à spécifier l'encodage des données. Par défaut, les données sont encodées de manière invisible pour l'utilisateur, mais, dans le cas d'un fichier, ce comportement pourrait corrompre son contenu. La valeur `multipart/form-data` sert ainsi à spécifier que le contenu du formulaire ne devra pas être encodé.

Fondamental

L'envoi d'un fichier via un formulaire implique obligatoirement l'utilisation de la méthode `POST`.

Utilisation de \$_FILES

La superglobale \$_FILES permet de manipuler les informations sur le fichier dans le script PHP qui le recevra.

Il s'agit d'un tableau associatif qui contient :

- **name** : le nom du fichier, valeur de l'attribut HTML `name`
- **type** : le type du fichier (png, jpeg, txt)
- **tmp_name** : le nom et le répertoire temporaire du fichier
- **error** : indique si une erreur est survenue lors de l'envoi
- **size** : la taille du fichier en octets

Syntaxe

\$_FILES est un tableau qui contient un autre tableau.

L'accès à un fichier s'écrit `$_FILES['monFichier']`, l'accès aux informations du fichier s'écrit : `$_FILES['monFichier']['attribut']`.

Attention

PHP limite par défaut la taille des fichiers qu'il peut recevoir via la méthode POST (généralement 8 Mo). Cette valeur est modifiable dans la configuration du serveur.

Exemple

Dans notre cas, si l'on soumet le formulaire évoqué précédemment, on obtiendrait l'affichage suivant :

```
1 <?php
2 print_r($_FILES);
3
4 //Affichera : Array ( [cni] => Array ( [name] => file.jpg [type] => image/jpeg [tmp_name] =>
5 /tmp/phpzCmHEp [error] => 0 [size] => 8973 ) )
6 ?>
```

Mécanisme d'envoi de fichiers

Une fois validé, le formulaire envoie les données vers la page spécifiée.

À ce moment précis, les fichiers sont stockés dans un répertoire temporaire. Cela permet de laisser la liberté au développeur de contrôler le fichier avant son enregistrement définitif sur le serveur.

Il est nécessaire de vérifier :

- si le fichier a bien été envoyé, grâce à la fonction `isset()` de PHP,
- s'il n'y a pas eu d'erreurs pendant le téléchargement,
- l'extension du fichier,
- la taille du fichier.

Fondamental

Les fichiers portant l'extension **.php** doivent être interdits pour éviter l'exécution de scripts extérieurs.

Exemple

Voici un exemple de vérifications que vous pourriez apporter :

```

1 <?php
2 $extensions = array('jpg', 'png', 'gif');
3
4 if (isset($_FILES['cni']) && !$_FILES['cni']['error']) {
5     $fileInfo = pathinfo($_FILES['cni']['name']);
6
7     if ($_FILES['cni']['size'] <= 2000000) {
8         if (in_array($fileInfo['extension'], $extensions)) {
9             // Scripts à exécuter quand les contrôles sont bons.
10        } else {
11            echo 'Ce type de fichier est interdit';
12        }
13    } else {
14        echo 'Le fichier dépasse la taille autorisée';
15    }
16 } else {
17     echo 'Une erreur est survenue lors de l\'envoi du fichier';
18 }
19 ?>

```

- Si le formulaire est validé sans le fichier ou s'il y a eu une erreur lors de l'envoi, le script affichera : *Une erreur est survenue lors de l'envoi du fichier.*
- Si la taille dépasse 2 Mo, dans notre cas, le script affichera : *Le fichier dépasse la taille autorisée.*
- Si l'extension du fichier ne fait pas partie de notre liste, le script affichera : *Ce type de fichier est interdit.*

La dernière étape est l'enregistrement du fichier dans un dossier du serveur.

Pour cela, il est possible d'utiliser la fonction `move_uploaded_file1()` de PHP. Elle accepte deux paramètres :

- le nom et le répertoire temporaire : `$_FILES['monFichier']['tmp_name']`
- le nom et le répertoire définitif : `$_FILES['monFichier']['name']`

Exemple

```

1 move_uploaded_file($_FILES['cni']['tmp_name'], 'cni/'.$_FILES['cni']['name']);
2 echo 'Le fichier a été envoyé sur le serveur';

```

Quand le fichier aura passé tous les contrôles, il sera enregistré dans le répertoire **cni/** et le script affichera : *Le fichier a été envoyé sur le serveur.*

Attention

Si un fichier ayant le même nom est téléchargé, celui déjà présent sur le serveur sera écrasé. Il convient de renommer les fichiers de manière unique pour éviter ce genre de désagrément.

¹ <https://www.php.net/manual/fr/function.move-uploaded-file.php>

Syntaxe À retenir

Pour envoyer un fichier via un formulaire HTML il faut :

- utiliser la méthode POST,
- ajouter l'attribut `enctype` avec la valeur `multipart/form-data`,
- utiliser un `input` de type `file`.

Pour manipuler le fichier côté PHP, il faut :

- utiliser la superglobale `$_FILES`,
- contrôler le fichier (présence, erreur, taille et extension),
- enregistrer définitivement le fichier.

Complément

`$_FILES`¹

`isset()`²

`pathinfo()`³

`move_uploaded_file()`⁴

XI. Exercice : Appliquez la notion

Vous disposez du formulaire HTML suivant, ainsi que de son script de traitement :

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8" />
5 </head>
6 <body>
7 <form method="post" action="traitement-formulaire.php">
8     <div>
9         <input type="radio" id="Mr" name="civilite" value="Mr">
10        <label for="Mr" class="inline-label">Monsieur</label>
11    </div>
12    <div>
13        <input type="radio" id="Mme" name="civilite" value="Mme">
14        <label for="Mme" class="inline-label">Madame</label>
15    </div>
16
17    <label for="name">Votre nom</label>
18    <input type="text" name="name" id="name" placeholder="Saisissez votre nom">
19
20    <label for="tos" class="inline-label">J'accepte les conditions générales
d'utilisation</label>
21    <input type="checkbox" name="tos" id="tos">
22
23    <button type="reset">Réinitialiser les valeurs du formulaire</button>
24    <button type="submit">Soumettre le formulaire</button>

```

1 <https://www.php.net/manual/fr/reserved.variables.files.php>

2 <https://www.php.net/manual/fr/function.isset.php>

3 <https://www.php.net/manual/fr/function.pathinfo.php>

4 <https://www.php.net/manual/fr/function.move-uploaded-file.php>

```
25 </form>
26 </body>
27 <style>
28 /**
29     Attention, ce CSS est là uniquement pour rendre le formulaire "agréable" à la lecture sans
    que vous n'ayez
    à récupérer deux fichiers distincts.
30     Dans un cas d'usage "réel", ces éléments doivent être externalisés
31     */
32
33     body {
34         font-family: Calibri, serif;
35     }
36
37     form {
38         max-width: 50%;
39     }
40
41     form label {
42         display: block;
43         font-weight: bold;
44         margin-bottom: 10px;
45     }
46
47     label.inline-label {
48         display: inline-block;
49     }
50
51     fieldset {
52         border: 1px solid lightgray;
53         background-color: rgba(225, 233, 255, 0.25);
54     }
55
56     legend {
57         font-style: italic;
58         font-size: 1.1em;
59         padding: 5px;
60     }
61
62     form input, form select, form textarea {
63         display: inline-block;
64         margin-bottom: 10px;
65         padding: 10px;
66         width: 80%;
67     }
68
69     form input[type="radio"],
70     form input[type="checkbox"],
71     form input[type="submit"] {
72         width: auto;
73     }
74
75     button[type=submit], button[type=reset] {
76         padding: 10px;
77         margin-top: 15px;
78     }
79 </style>
80 </html>
```

```

1 <?php
2
3 function printPostFormValue($key)
4 {
5     if (!isset($_POST[$key])) {
6         echo sprintf("La clé %s n'a pas été définie <br/>", $key);
7         return;
8     }
9
10    echo sprintf("La valeur de la clé %s est : " . PHP_EOL, $key);
11    print_r($_POST[$key]);
12    echo '<br/>';
13 }
14
15 printPostFormValue('civilite');
16 printPostFormValue('name');
17 printPostFormValue('tos');
18

```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question 1

[solution n°6 p.35]

Modifiez le formulaire afin d'y ajouter deux champs de type fichier. L'un permettra d'uploader une photo, l'autre une pièce d'identité.

N'oubliez pas de modifier les attributs du formulaire afin de permettre l'envoi de fichiers lors de sa soumission.

Question 2

[solution n°7 p.37]

Modifiez le script de traitement du formulaire. Ajoutez-y une fonction qui permettra de traiter les fichiers reçus.

Cette fonction devra :

- Vérifier que le fichier donné en paramètre existe
- Que sa taille fait moins de 2 Mo
- Que son extension fait partie des extensions suivantes : jpg, png, pdf, gif
- Vérifier que le répertoire correspondant à la clé du fichier existe, et le créer au besoin grâce aux fonctions `is_dir` et `mkdir`
- Déplacer le fichier dans ce répertoire

XII. Auto-évaluation

A. Exercice final

Exercice 1

[solution n°8 p.38]

Exercice

1 <https://repl.it/>

Dans un formulaire HTML, quel attribut d'un champ permet d'identifier la valeur saisie lors du traitement par le serveur ?

- ☐ id
- ☐ class
- ☐ name

Exercice

Quel attribut d'un formulaire convient-il de renseigner pour indiquer quel script traitera les données d'un formulaire ?

Exercice

Lorsque l'on souhaite faire transiter des données de façon sécurisée pour effectuer un traitement, la méthode à privilégier est...

- ☐ GET
- ☐ POST

Exercice

Dans une requête GET, quel caractère permet de séparer les différents attributs lorsque plusieurs valeurs sont présentes ?

- ☐ ?
- ☐ &
- ☐ +

Exercice

Si l'on souhaite accéder à la valeur du champ de formulaire de type texte dont le nom est "firstName", soumis via la méthode POST, quelle syntaxe doit-on utiliser ?

- ☐ \$_FORM['firstName']
- ☐ \$POST['firstName']
- ☐ \$_POST['firstName']
- ☐ \$_GET['firstName']

Exercice

Quelle méthode est-il nécessaire d'utiliser lorsqu'un formulaire contient un élément de type fichier ?

- ☐ POST
- ☐ GET

Exercice

Quel attribut de la balise `form` convient-il d'ajouter lorsqu'un formulaire contient un élément de type fichier ?

- ☐ action
- ☐ method
- ☐ enctype

Exercice

Si l'on souhaite accéder à la valeur du champ de formulaire de type fichier dont le nom est "picture", soumis via la méthode POST, quelle syntaxe doit-on utiliser ?

- ☐ `$_FORM['picture']`
- ☐ `$POST['picture']`
- ☐ `$_FILES['picture']`
- ☐ `$_POST['picture']`
- ☐ `$_GET['picture']`

Exercice

Lorsqu'un fichier est réceptionné par le serveur, plusieurs informations sont récupérées automatiquement. Lesquelles ?

- ☐ Le nom du fichier
- ☐ Sa taille
- ☐ Sa date de création
- ☐ Le nom de son auteur
- ☐ Son type

Exercice

Afin de permettre l'envoi de toutes les valeurs d'un champ à choix multiples, la syntaxe à adopter est...

- ☐ `id="monChamp[]"`
- ☐ `id="monChamp()"`
- ☐ `name="monChamp()"`
- ☐ `name="monChamp[]"`

B. Exercice : Défi

Dans le cadre de sa refonte, un site Internet souhaite mettre à la disposition de ses utilisateurs une page de contact.

Vous disposez du formulaire HTML suivant, ainsi que de deux fonctions permettant de lire et écrire dans un fichier intitulé **contact.txt** :

```
1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4   <meta charset="utf-8" />
5 </head>
6 <body>
7 <h1>Formulaire de contact</h1>
```

```

8 <form method="post" action="contact.php" enctype="multipart/form-data">
9   <label for="name">Votre nom</label>
10  <input type="text" name="name" id="name" placeholder="Saisissez votre nom">
11
12  <label for="email">Votre adresse e-mail</label>
13  <input type="email" name="email" id="email" placeholder="Saisissez votre e-mail">
14
15  <label for="message">Votre message</label>
16  <textarea rows="4" name="message" id="message"></textarea>
17
18  <label for="tos" class="inline-label">J'accepte les conditions générales
d'utilisation</label>
19  <input type="checkbox" name="tos" id="tos">
20
21  <button type="submit">Soumettre le formulaire</button>
22 </form>
23 </body>
24 <style>
25   body {
26     font-family: Calibri, serif;
27   }
28
29   form {
30     max-width: 50%;
31   }
32
33   form label {
34     display: block;
35     font-weight: bold;
36     margin-bottom: 10px;
37   }
38
39   label.inline-label {
40     display: inline-block;
41   }
42
43   fieldset {
44     border: 1px solid lightgray;
45     background-color: rgba(225, 233, 255, 0.25);
46   }
47
48   legend {
49     font-style: italic;
50     font-size: 1.1em;
51     padding: 5px;
52   }
53
54   form input, form select, form textarea {
55     display: inline-block;
56     margin-bottom: 10px;
57     padding: 10px;
58     width: 80%;
59   }
60
61   form input[type="radio"],
62   form input[type="checkbox"],
63   form input[type="submit"] {
64     width: auto;

```

```

65     }
66
67     button[type=submit], button[type=reset] {
68         padding: 10px;
69         margin-top: 15px;
70     }
71 </style>
72 </html>
73

```

```

1 <?php
2
3 function readContactFile()
4 {
5     $file = 'contact.txt';
6
7     if (!is_file($file)) {
8         return null;
9     }
10    return file_get_contents($file);
11 }
12
13 function writeInContactFile($message)
14 {
15     $file = 'contact.txt';
16
17     if (!is_file($file)) {
18         $current = '';
19     } else {
20         $current = file_get_contents($file);
21     }
22
23     $current .= $message;
24     $current .= PHP_EOL;
25
26     file_put_contents($file, $current);
27 }

```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question

[solution n°9 p.40]

- Vous allez devoir créer un script permettant de traiter le contenu de ce fichier lorsque le formulaire de contact est soumis.
- Le script permettra de construire un message au format HTML à partir des données du formulaire, et celles-ci seront enregistrées au moyen de la fonction `writeInContactFile`.
- Une fois le formulaire soumis, l'utilisateur pourra également visualiser le message qu'il a envoyé.
- Lorsqu'un utilisateur accédera à la page **contact.php** directement, s'il dispose d'un accès administrateur, il pourra visualiser l'ensemble des messages de contact postés.
- Dans le but de simplifier la démarche, on considérera qu'un utilisateur est administrateur s'il accède à la page **contact.php** avec un paramètre `admin=1`.

1 <https://repl.it/>

- S'il n'est pas administrateur, un message lui indiquera qu'il ne peut visualiser les messages préalablement postés.

Solutions des exercices

p.7 Solution n°1

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8" />
5 </head>
6 <body>
7 <form method="post" action="traitement-formulaire.php">
8     <div>
9         <input type="radio" id="Mr" name="civilite" value="Mr">
10        <label for="Mr" class="inline-label">Monsieur</label>
11    </div>
12    <div>
13        <input type="radio" id="Mme" name="civilite" value="Mme">
14        <label for="Mme" class="inline-label">Madame</label>
15    </div>
16
17    <label for="name">Votre nom</label>
18    <input type="text" name="name" id="name" placeholder="Saisissez votre nom">
19
20    <label for="birthDate">Votre date de naissance</label>
21    <input type="date" name="birthDate" id="birthDate" placeholder="Saisissez votre date de
naissance">
22
23    <label for="comment">Commentaire</label>
24    <textarea rows="4" name="comment" id="comment" placeholder="Ajoutez un commentaire
pertinent"></textarea>
25
26    <label for="email">Votre adresse e-mail</label>
27    <input type="email" name="email" id="email" placeholder="Saisissez votre e-mail">
28
29    <label for="languages">Langages de programmation maîtrisés</label>
30    <select name="languages[]" id="languages" multiple>
31        <option value="HTML">HTML</option>
32        <option value="JS">JS</option>
33        <option value="PHP">PHP</option>
34        <option value="Python">Python</option>
35        <option value="SQL">SQL</option>
36    </select>
37
38    <label for="tos" class="inline-label">J'accepte les conditions générales
d'utilisation</label>
39    <input type="checkbox" name="tos" id="tos">
40
41    <button type="reset">Réinitialiser les valeurs du formulaire</button>
42    <button type="submit">Soumettre le formulaire</button>
43 </form>
44 </body>
45 <style>
46 /**
47     Attention, ce CSS est là uniquement pour rendre le formulaire "agréable" à la lecture
sans que vous n'ayez
48     à récupérer deux fichiers distincts.
49     Dans un cas d'usage "réel", ces éléments doivent être externalisés
50     */
51     body {
52         font-family: Calibri, serif;

```

```

53     }
54
55     form {
56         max-width: 50%;
57     }
58
59     form label {
60         display: block;
61         font-weight: bold;
62         margin-bottom: 10px;
63     }
64
65     label.inline-label {
66         display: inline-block;
67     }
68
69     fieldset {
70         border: 1px solid lightgray;
71         background-color: rgba(225, 233, 255, 0.25);
72     }
73
74     legend {
75         font-style: italic;
76         font-size: 1.1em;
77         padding: 5px;
78     }
79
80     form input, form select, form textarea {
81         display: inline-block;
82         margin-bottom: 10px;
83         padding: 10px;
84         width: 80%;
85     }
86
87     form input[type="radio"],
88     form input[type="checkbox"],
89     form input[type="submit"] {
90         width: auto;
91     }
92
93     button[type=submit], button[type=reset] {
94         padding: 10px;
95         margin-top: 15px;
96     }
97 </style>
98 </html>
99

```

p. 9 Solution n°2

<http://localhost/methode-get.php?nom=laurent&age=27>

p. 10 Solution n°3

Voici le code que vous avez dû produire, vous pouvez le tester de la façon suivante :

`http://localhost/methode-get.php?nom=laurent&age=27`

`http://localhost/methode-get.php?nom=laurent&age=27&admin=0`

`http://localhost/methode-get.php?nom=laurent&age=27&admin=1`

```

1 <?php
2
3 if (isset($_GET['nom']) && isset($_GET['age']))
4 {
5     echo sprintf("Bonjour, mon nom est %s et j'ai %s ans. <br/>", $_GET['nom'],
6     $_GET['age']);
7 } else {
8     echo 'Je souhaite afficher mon nom et mon âge. Avez-vous oublié quelque chose ? <br/>';
9 }
10
11 if (isset($_GET['admin']) && 1 == (int) $_GET['admin']) {
12     echo 'Je suis administrateur <br/>';
13 } else {
14     echo 'Je ne suis pas administrateur <br/>';
15 }

```

p. 13 Solution n°4

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8"/>
5 </head>
6 <body>
7 <section>
8     <h1>Administration de votre site</h1>
9     <?php
10     if (isset($_GET['admin']) && 1 == (int) $_GET['admin']) {
11         ?>
12         <p class="greetings">
13             <?php
14             if (isset($_GET['name'])) {
15                 echo sprintf("Bienvenue sur l'administration de votre site %s",
16                 $_GET['name']);
17             }
18             ?>
19         </p>
20         <p>
21             Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor
22             incididunt ut labore et
23             dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris
24             nisi ut aliquip ex ea
25             commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum
26             dolore eu fugiat nulla
27             pariatur
28         </p>
29         <table>
30             <thead>
31             <tr>
32                 <td>Nom</td>

```

```

31         <td>Prénom</td>
32         <td>Action</td>
33     </tr>
34 </thead>
35 <tbody>
36 <tr>
37     <td>Lorem</td>
38     <td>Ipsum</td>
39     <td><a href="#">Supprimer</a></td>
40 </tr>
41 <tr>
42     <td>John</td>
43     <td>Doe</td>
44     <td><a href="#">Supprimer</a></td>
45 </tr>
46 <tr>
47     <td>John</td>
48     <td>Doe</td>
49     <td><a href="#">Supprimer</a></td>
50 </tr>
51 </tbody>
52 </table>
53 <?php
54 } else {
55     echo "<p>Vous n'êtes pas autorisés à accéder à cette partie du site.</p>";
56 }
57 ?>
58 </section>
59
60 </body>
61 <style>
62 /**
63     Attention, ce CSS est là uniquement pour rendre le formulaire "agréable" à la lecture
64     sans que vous n'ayez
65     à récupérer deux fichiers distincts.
66     Dans un cas d'usage "réel", ces éléments doivent être externalisés
67     */
68     body {
69         font-family: Calibri, serif;
70     }
71
72     h1, h2 {
73         text-align: center;
74     }
75
76     table {
77         border-collapse: collapse;
78         width: 50%;
79         margin: auto;
80     }
81
82     table, th, td {
83         border: 1px solid black;
84     }
85
86     thead {
87         background-color: beige;
88         font-weight: bold;

```

```

88     text-align: center;
89 }
90
91 p {
92     width: 50%;
93     margin: auto auto 10px;
94 }
95
96 p.greetings {
97     background-color: ivory;
98     font-weight: bold;
99     font-size: 16px;
100    padding: 5px;
101    border: 1px solid gray;
102 }
103
104 </style>
105 </html>
106

```

p. 18 Solution n°5

```

1 <?php
2
3 function printPostFormValue($key)
4 {
5     if (!isset($_POST[$key])) {
6         echo sprintf("La clé %s n'a pas été définie <br/>", $key);
7         return;
8     }
9
10    echo sprintf("La valeur de la clé %s est : " . PHP_EOL, $key);
11    print_r($_POST[$key]);
12    echo '<br/>';
13 }
14
15 printPostFormValue('civilite');
16 printPostFormValue('name');
17 printPostFormValue('birthDate');
18 printPostFormValue('comment');
19 printPostFormValue('email');
20 printPostFormValue('languages');
21 printPostFormValue('tos');
22

```

p. 24 Solution n°6

Voici le formulaire que vous avez dû obtenir :

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8" />
5 </head>
6 <body>
7 <form method="post" action="traitement-formulaire.php" enctype="multipart/form-data">

```

```

8     <div>
9         <input type="radio" id="Mr" name="civilite" value="Mr">
10        <label for="Mr" class="inline-label">Monsieur</label>
11    </div>
12    <div>
13        <input type="radio" id="Mme" name="civilite" value="Mme">
14        <label for="Mme" class="inline-label">Madame</label>
15    </div>
16
17    <label for="name">Votre nom</label>
18    <input type="text" name="name" id="name" placeholder="Saisissez votre nom">
19
20    <label for="picture">Votre photo</label>
21    <input type="file" name="picture" id="picture">
22
23    <label for="cardId">Votre justificatif d'identité</label>
24    <input type="file" name="cardId" id="cardId">
25
26    <label for="tos" class="inline-label">J'accepte les conditions générales
d'utilisation</label>
27    <input type="checkbox" name="tos" id="tos">
28
29    <button type="reset">Réinitialiser les valeurs du formulaire</button>
30    <button type="submit">Soumettre le formulaire</button>
31 </form>
32 </body>
33 <style>
34 /**
35     Attention, ce CSS est là uniquement pour rendre le formulaire "agréable" à la lecture
sans que vous n'ayez
36     à récupérer deux fichiers distincts.
37     Dans un cas d'usage "réel", ces éléments doivent être externalisés
38     */
39     body {
40         font-family: Calibri, serif;
41     }
42
43     form {
44         max-width: 50%;
45     }
46
47     form label {
48         display: block;
49         font-weight: bold;
50         margin-bottom: 10px;
51     }
52
53     label.inline-label {
54         display: inline-block;
55     }
56
57     fieldset {
58         border: 1px solid lightgray;
59         background-color: rgba(225, 233, 255, 0.25);
60     }
61
62     legend {
63         font-style: italic;
64         font-size: 1.1em;

```

```

65     padding: 5px;
66 }
67
68 form input, form select, form textarea {
69     display: inline-block;
70     margin-bottom: 10px;
71     padding: 10px;
72     width: 80%;
73 }
74
75 form input[type="radio"],
76 form input[type="checkbox"],
77 form input[type="submit"] {
78     width: auto;
79 }
80
81 button[type=submit], button[type=reset] {
82     padding: 10px;
83     margin-top: 15px;
84 }
85 </style>
86 </html>
87

```

p. 24 Solution n°7

Voici le script que vous avez dû obtenir :

```

1 <?php
2
3 function printPostFormValue($key)
4 {
5     if (!isset($_POST[$key])) {
6         echo sprintf("La clé %s n'a pas été définie <br/>", $key);
7         return;
8     }
9
10    echo sprintf("La valeur de la clé %s est : " . PHP_EOL, $key);
11    print_r($_POST[$key]);
12    echo '<br/>';
13 }
14
15
16 function manageFile($key)
17 {
18     $extensions = array('jpg', 'png', 'gif', 'pdf');
19
20     if (isset($_FILES[$key]) && !$_FILES[$key]['error']) {
21         $fileInfo = pathinfo($_FILES[$key]['name']);
22         if ($_FILES[$key]['size'] <= 2000000) {
23             if (in_array($fileInfo['extension'], $extensions)) {
24                 if (!is_dir($key)) {
25                     try {
26                         mkdir($key);
27                         echo 'Le répertoire a été créé ';
28                     } catch (Exception $e) {
29                         echo 'Une erreur est survenue : ' . $e->getMessage();

```

```

30         }
31     }
32
33     try {
34         move_uploaded_file($_FILES[$key]['tmp_name'], $key . '/' . $_FILES[$key]
35 ['name']);
36         echo 'Le fichier a été envoyé sur le serveur';
37     } catch (Exception $e) {
38         echo 'Une erreur est survenue : ' . $e->getMessage();
39     }
40 } else {
41     echo 'Ce type de fichier est interdit';
42 }
43 } else {
44     echo 'Le fichier dépasse la taille autorisée';
45 }
46 } else {
47     echo 'Une erreur est survenue lors de l\'envoi du fichier';
48 }
49
50 printPostFormValue('civilite');
51 printPostFormValue('name');
52 printPostFormValue('tos');
53 manageFile('picture');
54 manageFile('cardId');
55

```

Exercice p. 24 Solution n°8

Exercice

Dans un formulaire HTML, quel attribut d'un champ permet d'identifier la valeur saisie lors du traitement par le serveur ?

- ☐ id
- ☐ class
- ☒ name

Exercice


Quel attribut d'un formulaire convient-il de renseigner pour indiquer quel script traitera les données d'un formulaire ?

action

Exercice

Lorsque l'on souhaite faire transiter des données de façon sécurisée pour effectuer un traitement, la méthode à privilégier est...


- ☐ GET
- ☒ POST

 Il est préférable d'utiliser la méthode POST lorsque l'on souhaite faire transiter des données de façon sécurisée. La méthode GET fait transiter ces informations de façon visible et modifiable par l'utilisateur.

Exercice

Dans une requête `GET`, quel caractère permet de séparer les différents attributs lorsque plusieurs valeurs sont présentes ?

- ☐ ?
- ☒ &
- ☐ +

 Le caractère & sert à délimiter les différents paramètres, tandis que ? permet d'indiquer que l'URL contient au moins un paramètre.

Exercice


Si l'on souhaite accéder à la valeur du champ de formulaire de type texte dont le nom est `"firstName"`, soumis via la méthode `POST`, quelle syntaxe doit-on utiliser ?

- ☐ `$_FORM['firstName']`
- ☐ `$POST['firstName']`
- ☒ `$_POST['firstName']`
- ☐ `$_GET['firstName']`

Exercice

Quelle méthode est-il nécessaire d'utiliser lorsqu'un formulaire contient un élément de type fichier ?

- ☒ `POST`
- ☐ `GET`

 Seule la méthode `POST` permet de faire transiter des fichiers via une URL.

Exercice

Quel attribut de la balise `form` convient-il d'ajouter lorsqu'un formulaire contient un élément de type fichier ?

- ☐ `action`
- ☐ `method`
- ☒ `enctype`

Exercice

Si l'on souhaite accéder à la valeur du champ de formulaire de type fichier dont le nom est `"picture"`, soumis via la méthode `POST`, quelle syntaxe doit-on utiliser ?

- ☐ `$_FORM['picture']`
- ☐ `$POST['picture']`
- ☒ `$_FILES['picture']`
- ☐ `$_POST['picture']`
- ☐ `$_GET['picture']`

Exercice

Lorsqu'un fichier est réceptionné par le serveur, plusieurs informations sont récupérées automatiquement. Lesquelles ?

- ☒ Le nom du fichier
- ☒ Sa taille
- ☐ Sa date de création
- ☐ Le nom de son auteur
- ☒ Son type

Exercice

Afin de permettre l'envoi de toutes les valeurs d'un champ à choix multiples, la syntaxe à adopter est...

- ☐ id="monChamp[]"
- ☐ id="monChamp()"
- ☐ name="monChamp()"
- ☒ name="monChamp[]"

p. 28 Solution n°9

```

1 <!DOCTYPE html>
2 <html lang="fr">
3 <head>
4     <meta charset="utf-8"/>
5 </head>
6 <body>
7 <section>
8
9 </section>
10 <?php
11 if (!empty($_POST)) {
12     $message = buildContactMessage();
13     writeInContactFile($message);
14     ?>
15     <h1>Merci d'avoir pris contact</h1>
16
17     <p><?php echo $message; ?></p>
18     <?php
19 } else {
20     if (isset($_GET['admin']) && (int)$_GET['admin'] === 1) {
21     ?>
22     <h1>Voici la liste des messages reçus</h1>
23
24     <p><?php echo readContactFile(); ?></p>
25 <?php
26 }
27     ?>
28     <h1>Erreur</h1>
29
30     <p>Vous n'avez pas accès à cette page</p>
31     <?php

```



```

32     }
33 ?>
34 </body>
35 <style>
36     body {
37         font-family: Calibri, serif;
38     }
39 </style>
40 </html>
41
42 <?php
43 function writeInContactFile($message)
44 {
45     $file = 'contact.txt';
46
47     if (!is_file($file)) {
48         $current = '';
49     } else {
50         $current = file_get_contents($file);
51     }
52
53     $current .= $message;
54     $current .= PHP_EOL;
55
56     file_put_contents($file, $current);
57 }
58
59 function readContactFile()
60 {
61     $file = 'contact.txt';
62
63     if (!is_file($file)) {
64         return null;
65     }
66     return file_get_contents($file);
67 }
68
69 function buildContactMessage()
70 {
71     $message = 'Date de création : ' . date('d/m/y H:i:s') . '<br/>' . PHP_EOL;
72
73     if (isset($_POST['name'])) {
74         $message .= 'Nom : ' . $_POST['name'] . '<br/>' . PHP_EOL;
75     }
76
77     if (isset($_POST['email'])) {
78         $message .= 'Email : ' . $_POST['email'] . '<br/>' . PHP_EOL;
79     }
80
81     if (isset($_POST['message'])) {
82         $message .= 'Message : ' . $_POST['message'] . '<br/>' . PHP_EOL;
83     }
84
85     return $message;
86 }
87
88 ?>

```