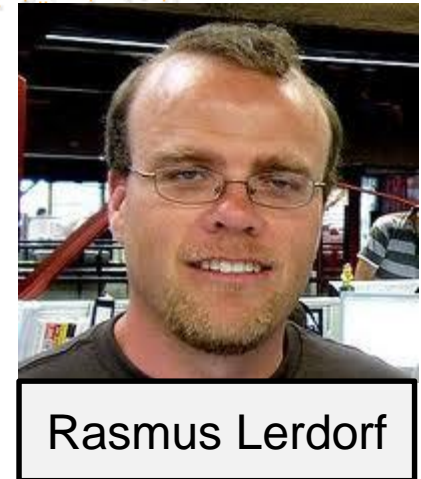


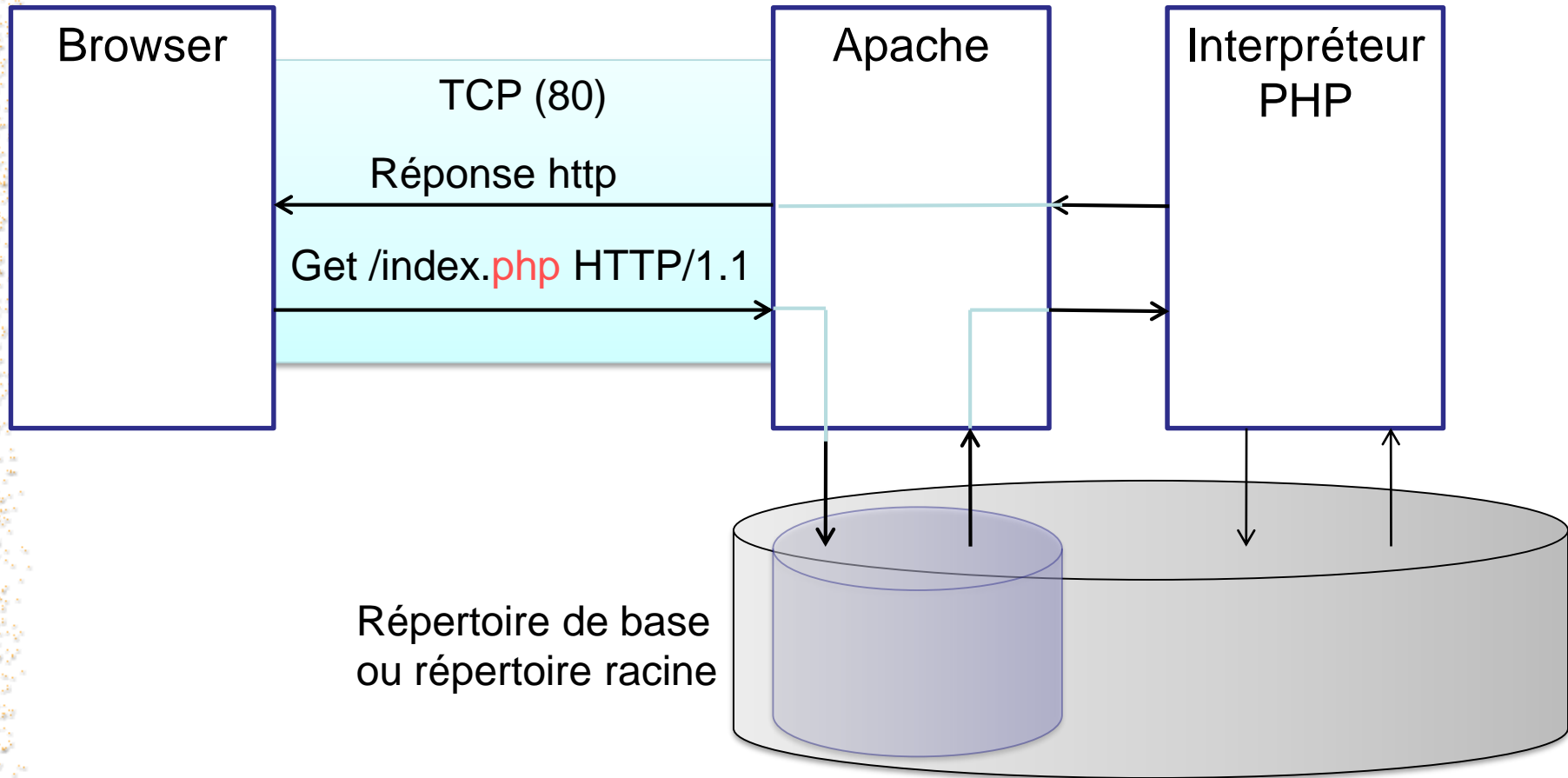
INTERNET → PHP

- Introduction
- Particularités du langage.
- Environnement d'exécution.
- Accès aux données (mysql)
 - phpMyadmin
 - CRUD
- Concept d'application
 - Sessions
 - Cookies



Introduction

- Si l'extension du fichier est « .php » le fichier est d'abord envoyé à l'interpréteur PHP.

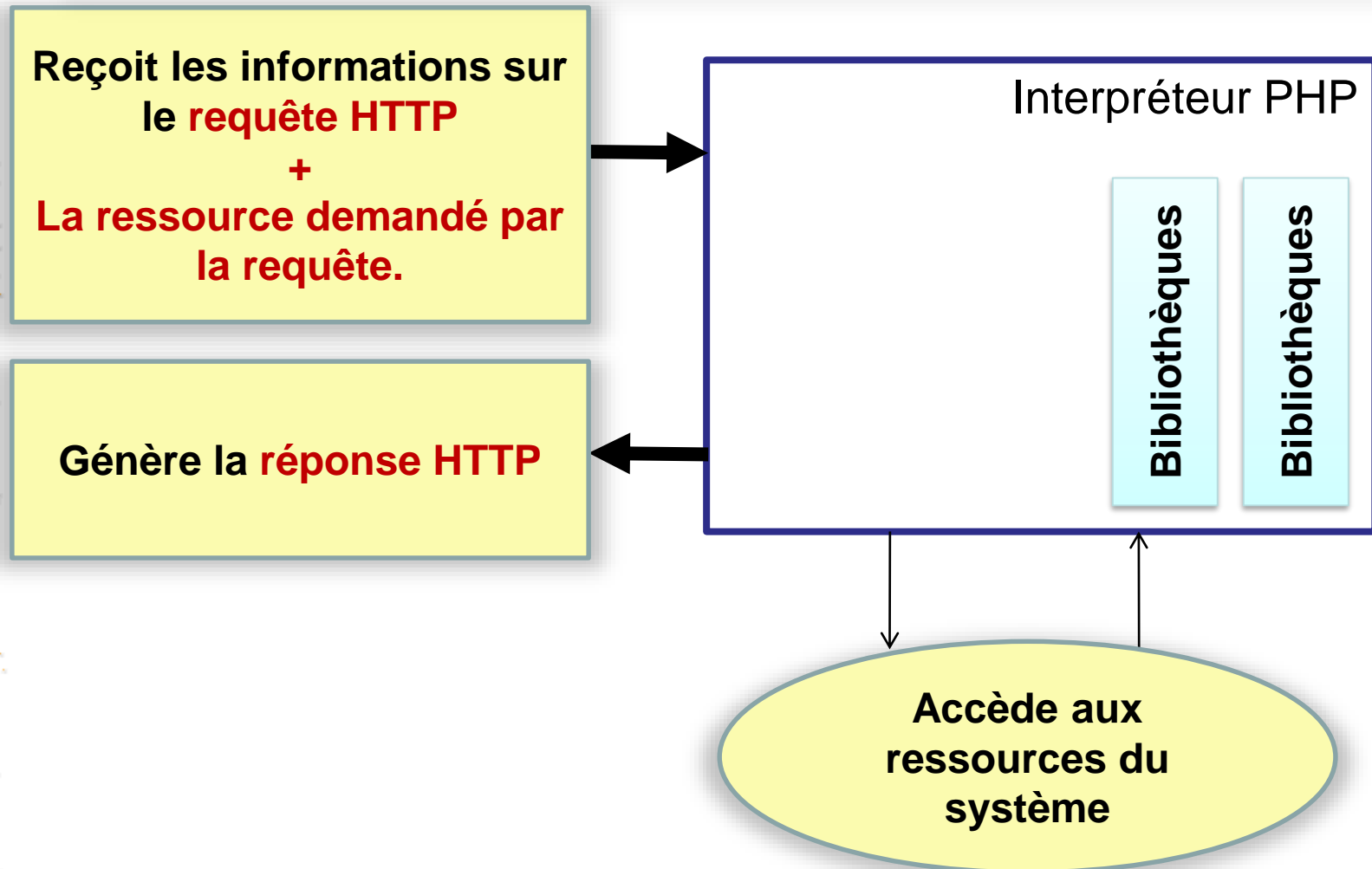


Introduction

- Quand l'interpréteur reçoit une page contenant des scripts PHP. Celui-ci sera interprété et le résultat sera retourné au serveur web.
- Les scripts PHP sont entourés par les balises `<?php` et `?>`
- Il est possible d'utiliser `<? Et ?>` mais c'est déconseillé.

Modèle de fonctionnement

Chaque requête obéit à ce modèle .



Particularités de PHP - Les variables

- Les variables en PHP commencent par \$

```
$a = 25; // entier  
$b = "azerty"; // chaîne de caractère  
$c = true; // booleen  
$d = array("abc", 24) // tableau  
$e = new MaClasse() // Instanciation
```

- Les variables ne sont pas typées.
- La valeur définit le type de la variable.
- Il n'y a pas de mot clé pour déclarer une variable.

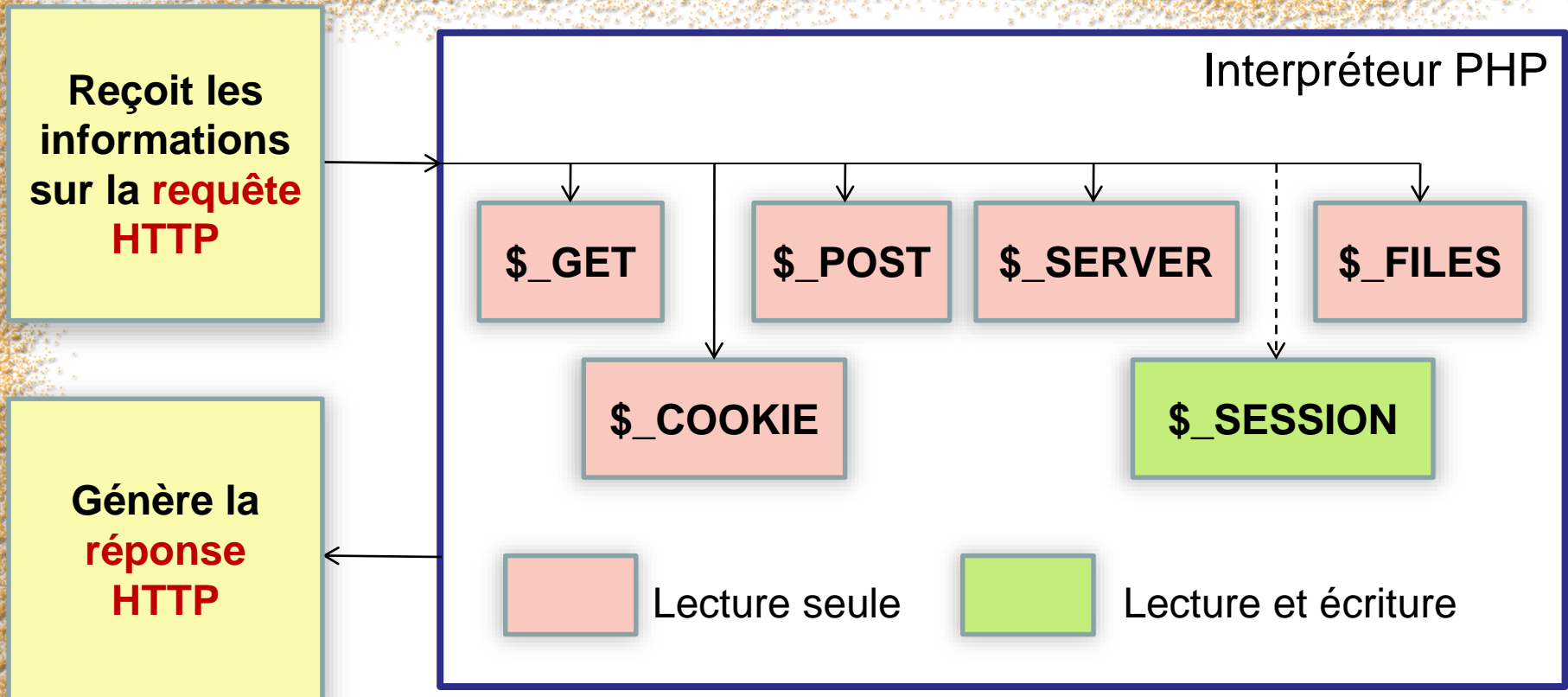
Particularités de PHP - Les types

- Boolean. `true`, `false`
- Entier. `0`, `-10`, `200`
- Nombre à virgule flottante. `1.234`, `0.23`
- Chaînes de caractères `"ab"`, `'abc'`
- Tableaux `$a[1]`, `$b[3][0]`, `$c['key']`
- Objets `$a->attrib`; `$a->method()`
- Ressources
- Null. signifie aucune valeur !

Fonctions clés

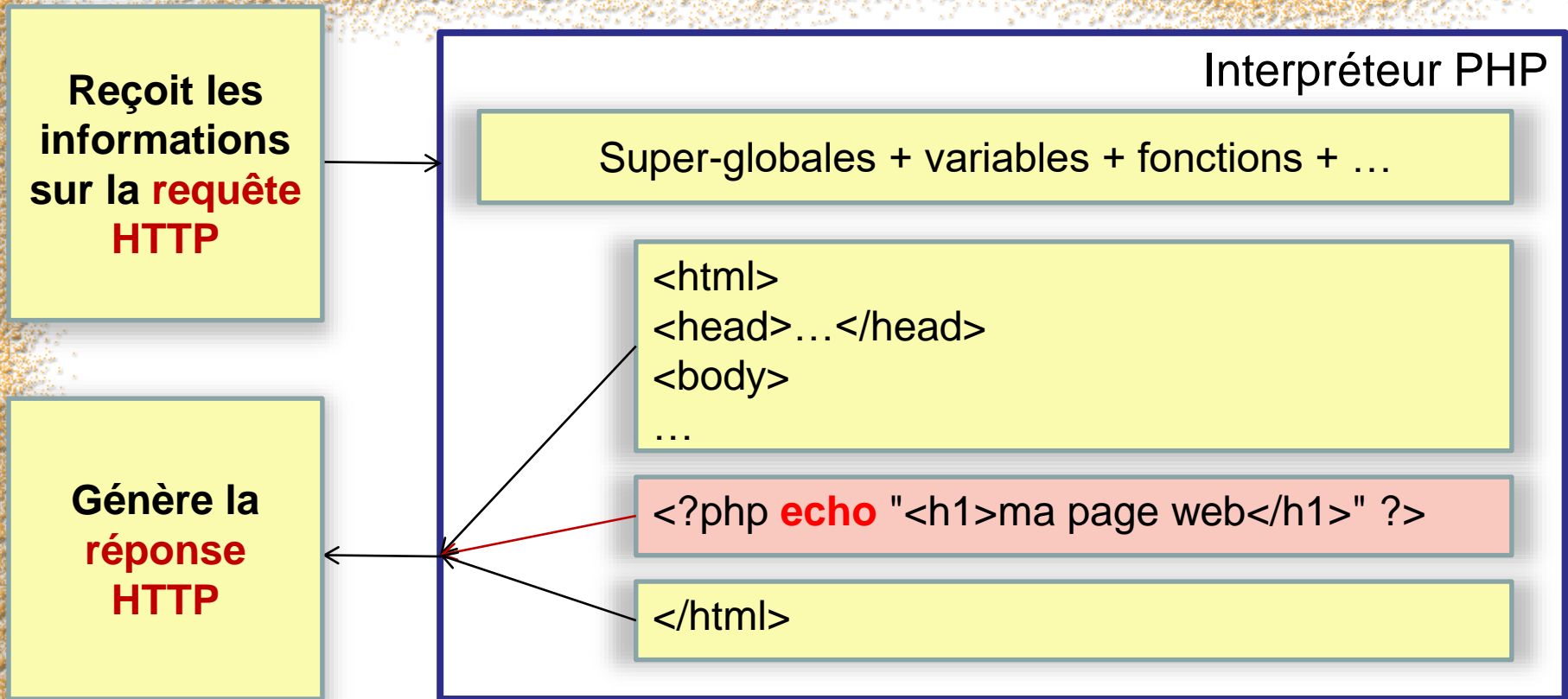
```
isset($a); // ret true si $a est définie
unset($a); // détruit la variable $a
unset($b[5]) // détruit un élément du tableau
empty($c); // ret true si $c vide ou nulle
count($d); // ret le nbre d'élm. du tableau
strlen("abc") // retourne 3
/* tester les types */
is_int($var); is_float($var); is_string($var);
is_array($var); is_object($var); ...
/* logging valeur et types de variables
var_dump($var) // type et contenu
print_r($var) // affiche le tableau $var
```

Environnement d'exécution



PHP utilise les informations de la requête HTTP pour générer les variables super-globales. Le contenu de ces variables est accessible dans toute la page. Ces variables sont des tableaux associatifs.

Environnement d'exécution



Le code HTML et le code généré par les fonctions de sortie sont dirigés vers la sortie standard (Corps de la réponse HTTP).

Exemple - 1

```
GET /image.php HTTP/1.1
```

```
<?php
// écriture dans l'entête de la réponse http
header("content-type:image/jpeg"); ?>

// écriture dans le corps de la réponse http
readfile("images/monimage.jpg");


?>
```

Il est important de noter que les données sont envoyées au fur et à mesure qu'elles sont générées. Il n'est pas donc sensé d'écrire dans l'entête après avoir commencé à écrire dans le corps. C'est une erreur commune à éviter. (Voir aussi les cookies)

Exemple - 2

```
GET /index.php?var=value HTTP/1.1
```

```
<html>
  <head>...</head>
  <body>
    <h3>Le contenu de la requête est :
    <b> <?php echo $_GET["var"] ?> </b>
    </h3>
  </body>
</html>
```



`$_GET` permet d'accéder aux paramètres d'URL. Ceux-ci peuvent être générés par un formulaire utilisant la méthode « **get** » ou directement définis dans un lien hypertexte.

Exemple - 3

```
POST /index.php HTTP/1.1
```

```
<html>
  <head>...</head>
  <body>
    <h3>Le contenu de la requête est :
    <b> <?php echo $_POST["var"] ?> </b>
    </h3>
  </body>
</html>
```

`$_POST` permet d'accéder aux paramètres envoyés par un formulaire utilisant la méthode « **post** ». Dans le cas de la méthode « **post** », il est possible d'envoyer aussi des fichiers.

Exemple - 4

```
POST /server.php HTTP/1.1
```

```
<html>
  <head>...</head>
  <body>
    <h3>Le contenu de $_SERVER est :</h3>
    <pre> <?php print_r($_SERVER) ?> </pre>
  </body>
</html>
```

\$_SERVER contient des informations sur le client, le protocole de communication et le serveur. **Elle présente un contexte particulier.** La balise **<pre>** permet de préserver le format de sortie de la fonction « **print_r()** » ;
Ici, on imprime le contenu de la variable **\$_SERVER**.

Exemple - 5

```
POST /cookie.php HTTP/1.1
```

```
<html>
  <head>...</head>
  <body>
    <h3>Le contenu de $_COOKIE est </h3>
    <pre> <?php print_r($_COOKIE) ?> </pre>
  </body>
</html>
```

A chaque requête, le navigateur envoie tous les **cookies** associés au domaine du site web. **\$_COOKIE** contient tous ses **cookies**.

Ici, on imprime le contenu de la variable **\$_COOKIE**. Les cookies sont enregistrés dans le cache du navigateur. L'utilisateur pourra y accéder et les supprimer.

Ici le domaine est « localhost » et la variable **\$_COOKIE** est vide !

Exemple – 5 (suite)

```
POST /cookie.php HTTP/1.1
```

```
<?php setcookie("monCookie", "maValeur") ?>
<html>
  <head>...</head>
  <body>
    <h3>Le contenu de $_COOKIE est </h3>
    <pre> <?php print_r($_COOKIE) ?> </pre>
  </body>
</html>
```

La première fois qu'on accède à cette page, le serveur va envoyer dans l'entête de sa réponse **une entête http « set-cookie »** qui inscrira le cookie dans le cache du navigateur. Cette fois-ci \$_COOKIE est toujours vide. Toutes les requêtes suivantes vers le site enverront **l'entête http « cookie »** qui continent le nom et la valeur du cookie.

Exemple – 6

```
GET /photo.php HTTP/1.1
```

```
<html>
  <head>...</head>
  <body>
    <form method="post" action="upload.php"
          enctype="multipart/form-data" >
      <input type="file" name="photo" /><br />
      Légende : <input name="legende" /><br />
      <input type="submit" value="Envoyer" />
    </form>
  </body>
</html>
```

Le formulaire permet d'envoyer à l'aide de la méthode « post » des paramètres et des fichiers. L'attribut « **enctype="multipart/form-data"** » est obligatoire, il permet au serveur de savoir que des fichiers sont aussi envoyés dans la requête. Les données sont traitées par « **upload.php** »

Exemple – 6

```
GET /upload.php HTTP/1.1
```

```
<html>
  <head>...</head>
  <body>
    <h3>Le contenu de $_FILES est : </h3>
    <pre> <?php print_r($_FILES) ?> </pre>
    <h3>Le contenu de $_POST est : </h3>
    <pre> <?php print_r($_POST) ?> </pre>
  </body>
</html>
```

Ici la variable **\$_POST** permet d'accéder aux paramètres de la requête.
\$_FILES permet d'accéder aux fichiers téléchargés.