

# Introduction au langage PHP

## Deuxième partie



XAMPP



# Plan

- Les formulaires (Rappel HTML)
- Manipulation des formulaires en PHP
- Les sessions
- Les cookies



# Les formulaires (Rappel HTML)

---

# Pourquoi un formulaire ?

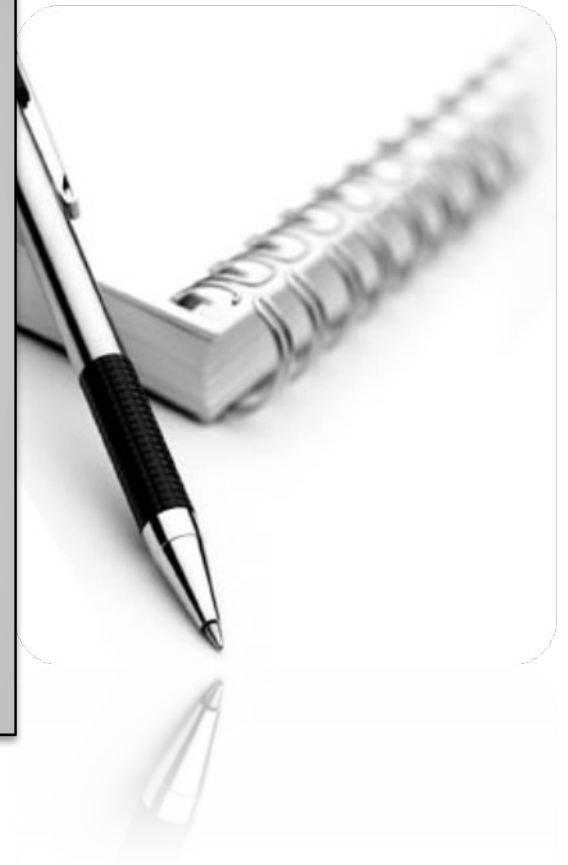
Les formulaires interactifs permettent aux auteurs de pages Web de doter leur page web **d'éléments interactifs** permettant par exemple un dialogue avec les internautes, à la manière des coupons-réponse présents dans certains magazines.

Réf : <http://www.commentcamarche.net/html/htmlform.php3>

# La balise **FORM**

Les formulaires sont délimités par la balise **<FORM> ... </FORM>**.

Cette balise qui permet de regrouper plusieurs éléments de formulaire (boutons, champs de saisie,...) et qui possède les attributs obligatoires **ACTION** et **METHOD**.



# La balise FORM

## 1 – L'attribut : METHOD

**METHOD** indique sous quelle forme seront envoyées les réponses « **POST** » est la valeur qui correspond à un envoi de données stockées dans le corps de la requête, tandis que « **GET** » correspond à un envoi des données codées dans l'URL, et séparées de l'adresse du script par un « ? ».

i.e. : [www.monsite.com/index.php?id=1](http://www.monsite.com/index.php?id=1).

*Pour plus d'information sur la différence entre GET et POST:  
<http://www.cs.tut.fi/~jkorpela/forms/methods.html>*

# La balise FORM

## 2 – L'attribut : ACTION

**ACTION** indique l'adresse d'envoi (script CGI, script PHP ... ou adresse email (mailto:adresse.email@machine))

## 3 – L'attribut : ENCTYPE (optionnel)

**ENCTYPE** spécifie le format des données envoyées, dans le cas où un protocole n'imposerait pas de format précis. Il possède plusieurs valeurs dont :

- "**text/plain**" : Valeur utilisée dans le cas d'un formulaire classique
- "**multipart/form-data**" : Valeur utilisée dans le cas où le formulaire comprendrait un fichier attaché (<input type="file" ...>)

# Les éléments de formulaires

*Les éléments de formulaires sont répartis en 3 classes :*

- **INPUT**  
Champs de saisie de texte et différents types de boutons .
- **SELECT**  
Listes (menus déroulants et ascenseurs) .
- **TEXTAREA**  
Zone de saisie de texte libre.






# Les éléments INPUT

| Type     | Syntaxe   | Exemple   |
|----------|---|---|
| sans     | <code>&lt;input name="ident"&gt;</code>   | <input type="text"/>  |
|          | <code>&lt;input name="ident" value="Par défaut"&gt;</code>  | <input type="text" value="Par défaut"/>   |
| submit   | <code>&lt;input type="submit" value="Envoi"&gt;</code>  | <input type="submit" value="envoi"/>  |
| checkbox | <code>&lt;input type="checkbox" name="pfm" value="linux" checked&gt; Linux&lt;BR&gt;</code><br><code>&lt;input type="checkbox" name="pfm" value="dos"&gt; Dos&lt;BR&gt;</code><br><code>&lt;input type="checkbox" name="pfm" value="win"&gt; Windows</code> | <input checked="" type="checkbox"/> Linux<br><input type="checkbox"/> Dos<br><input type="checkbox"/> Windows |

# Les éléments INPUT


| Type     | Syntaxe  | Exemple  |
|----------|--|--|
| radio    | <pre>&lt;input type="radio" name="media" value="cd" checked&gt; CD-ROM&lt;BR&gt;</pre> <pre>&lt;input type="radio" name="media" value="dk"&gt; Disquette</pre> | <input checked="" type="radio"/> CD-ROM<br><input type="radio"/> Disquette |
| password | <pre>&lt;input type="password" name="pass"&gt;</pre>   | <input type="password"/>   |
| reset    | <pre>&lt;input type="reset" value="Efface"&gt;</pre>   | <input type="reset" value="Efface"/>                                       |
| file     | <pre>&lt;input type="file" name="file" /&gt;</pre>   | <input type="file"/> <input type="button" value="Parcourir..."/>           |

# L'élément SELECT

| Syntaxe  | Exemple   |
|--|---|
| <pre>&lt;select name="menu"&gt; &lt;option&gt; Banane &lt;option&gt; Orange &lt;option &gt; Citron &lt;option selected&gt; Pomme &lt;option&gt; Pêche &lt;option&gt; Poire &lt;/select&gt;</pre> |    |
| <pre>&lt;select name="menu" size=4&gt; ...</pre>   |   |
| <pre>&lt;select name="menu" size=4 multiple&gt; ...</pre>  |  |

# L'élément TEXTAREA

```
<textarea name="comm" rows=10 cols=40>  
Tapez vos commentaires ici  
</textarea>
```



Tapez vos commentaires ici

# Manipulation des formulaires en PHP

The slide features a dark blue header with the title in white. Below the header, there are several horizontal decorative lines: a thick teal line, followed by a thinner teal line, and then two thin light blue lines.

# Manipulation des formulaires en PHP

- Les informations **entrées** dans un formulaire sont **récupérées** sous forme de variables.
- **Le nom de ces variables** dépend de **la méthode d'envoi** du formulaire.

**Exemple :** si la méthode d'envoi est **POST**, il faut mettre comme nom de variable **`$_POST['nom_du_champ']`** (idem pour GET).

# Exemple 1

Deux page :

- `identif.html` : contient le formulaire (HTML).
- `verif.php` : contient le code PHP pour vérifier si login est bien « moi » et le password est « marouan ».

# identif.html

## identification

Login :

Password :

```
<html><body>
<h4>identification</h4>
<form action= "verif.php" method="post">
Login :<input type="text" name= "login" /> <br>
Password :<input type= "password" name= "
password" /><br>
<input type= "reset" value= " clear" />
<input type="submit" /> </form>
</body></html>
```



# process.php

```
<?php
if ($_POST["login"] == "moi" && $_POST["password"] == "marouan")
{
    echo "All rights";
}
else header("location: identif.html"); /* Redirect browser */
?>
```

# Exemple 2

Deux page :

- formulaire.html : contient le formulaire (HTML).
- process.php : contient le code PHP qui va agir sur les données du formulaire.

# formulaire.html

## Tizag Art Supply Order Form

Paint  Quantity:

```
<html><body>
<h4>Tizag Art Supply Order Form</h4>
<form action="process.php" method="post">
<select name="item">
<option>Paint</option>
<option>Brushes</option>
<option>Erasers</option>
</select>
Quantity: <input name="quantity" type="text" />
<input type="submit" /> </form>
</body></html>
```

# process.php

```
<html><body>
<?php
$quantity = $_POST['quantity'];
$item = $_POST['item'];

echo "You ordered ". $quantity . " " . $item . "<br />";
echo "Thank you for ordering from Junior Art Supplies!";

?>
</body></html>
```

```
$quantity = $_POST['quantity'];
$item = $_POST['item'];
```

**Exemple d'exécution :**  
You ordered 6 brushes.



```
extract($_POST);
```

Thank you for ordering from Junior Art Supplies!

# Fonctions utiles

- `isset()` : teste l'existence d'une variable ( savoir si une variable a été définie ou non).
- `is_empty()` or `empty()` : indique qu'un champ d'un formulaire a été rempli ou non.
- `extract()` : permet d'extraire les données d'un formulaire après validation.

# Les sessions

The slide features a dark blue header area. Below the title, there is a decorative element consisting of a solid teal horizontal bar, followed by a white horizontal bar, and then two thin, parallel light blue horizontal lines.

# Les sessions

- *Le support des sessions en PHP est un moyen de **préserver** des données, **relatives** au visiteur, **entre plusieurs accès**.*
- *Elles permettent de stocker des types de **données simples** (texte, nombres, ...) mais pas de ressources comme des images ou bdd.*

# Pourquoi utiliser les sessions ?

- Pour conserver de page en page les valeurs de certaines variables.
- Pour pister le parcours du visiteur.
- Pour effectuer des statistiques fines en termes de visiteurs réels et pas en hits (nombre d'appel d'un fichier).



# Démarrer une session

## *Syntaxe*

```
<?php session_start(); ?>
```

Ce code permet de **démarrer une session**. Si un fichier existe sur le serveur pour cette session, **les variables de sessions seront récupérées**, si ce n'est pas le cas, **un nouveau fichier sera créé**.

La session doit être déclarée dans le code **tout en haut** de votre page car le cache du navigateur doit être vide pour démarrer une session. Donc **aucun** code HTML avant le démarrage d'une session !!

# Créer une variable de session

Nous créons ici une variable de session nommée variable qui vaut **\$valeur**.

## *Syntaxe*

```
<?php $_SESSION['variable'] = $valeur ; ?>
```

Les variables de sessions sont accessibles, une fois que la sessions est démarrée, via un tableau super global : **\$\_SESSION**

# Utiliser la valeur d'une variable de session

## *Exemple*

```
<?php if(isset($_SESSION['ensi']))  
    echo 'La variable "ensi" existe et vaut: ' . $_SESSION['ensi'];  
?>
```

**isset()** permet de savoir si **la variable de session** « ensi » existe ou non.

# Supprimer une variable de session

## *Syntaxe*

```
<?php unset($_SESSION['variable']);  
echo 'La variable de session "variable" est maintenant détruite';  
?>
```

**unset()** permet de supprimer une variable de session.

# Détruire toutes les variables de session

*Syntaxe*

```
<?php session_unset(); ?>
```

# Détruire une session

*Syntaxe*

```
<?php session_destroy(); ?>
```

# Les cookies

The slide features a dark blue header. Below the title, there is a decorative element consisting of a solid teal bar, followed by a light blue bar, and then two thin white lines.

# Qu'est-ce qu'un cookie ?

Un **cookie** est un fichier que le serveur envoie sur la machine de l'utilisateur. Il est souvent utilisé pour reconnaître les utilisateurs.

Pour explorer le répertoire cookies sous votre machine Windows ; faites **Démarrer/Exécuter** et taper **Cookies**.

# Démarrer une session

## *Syntaxe*

```
<?php setcookie(name, value, expire, path, domain); ?>
```

La fonction **setcookie()** doit être placé avant tout code HTML, car le cache du navigateur doit être vide pour que cette fonction marche convenablement.



# Exemple 1

L'exemple ci-dessous génère un cookie appelé "**nom\_cookie**", avec pour contenu "Léon" et pendant une période de **10 heures**.

```
<?php
// génère le cookie
$contentu = 'léon'; // le contenu de votre cookie
setcookie("nom_cookie", $contentu, time()+36000);
?>
<html>
<body>
<p>
Un cookie a été généré sur cette page.
</p>
</body>
</html>
```

# Comment retrouver la valeur d'un cookie

L'exemple ci-dessous test l'existence d'un cookie appelé "**nom\_cookie**" et affiche si le cookie existe ou n'existe pas.

```
<html>
<body>
<?php
// test l'existence d'un cookie appelé "nom_cookie"
if (isset($_COOKIE["nom_cookie"]))
echo 'Le cookie existe ' . $_COOKIE["nom_cookie"] .
'!<br />';
else
echo 'Le cookie n\'existe pas <br />';
?>
</body>
```

# Afficher tous les cookies

*Syntaxe*

```
<?php  
print_r($_COOKIE);  
?>
```

## Remarque Session Vs Cookie

*La différence entre les sessions et les cookies est que les cookies sont stockés sur le poste du visiteur tandis que **les sessions sont dans des fichiers présents sur le serveur.***

# Références

- <http://www.snv.jussieu.fr/archambault/cours/html/textes/forms.html>
- <http://www.php.net/manual/fr/ref.session.php>
- <http://www.phpsources.org/tutoriel-cookies.htm>
- <http://www.phpsources.org/tutoriel-sessions.htm>
- <http://cyberzoide.developpez.com/php4/faqsession/>
- <http://www.allhtml.com/articles/detail/370>





# Questions ?