

# Intégration Web / PHP

Ryan Cassel  
[cassel@lmsi.fr](mailto:cassel@lmsi.fr)  
 Université Paris XI

## Sommaire

- Conception
- Formulaires
- Gestion de fichiers

PHP 2005 2

## Conception

- Modèles de conception
  - Cascade
  - Spirale

PHP 2005 3

## Conception

- Première étape :
  - Arborescence
    - L'arborescence du site va décrire les différents **liens unissant vos pages**.
    - Ce premier travail se fait généralement à la main, avec un stylo sur papier.
    - Les pages seront nommées ; il ne faudra plus changer ces noms pour ne pas briser de liens.
    - Eviter d'utiliser de mots trop long, ne mettre aucun espaces et accents qui sont mal gérés par les navigateurs.
    - La première page s'appelle toujours "index", ou parfois "main" car ce sont les noms que cherche le navigateur.

PHP 2005 4

## Conception

- Exemple d'arborescence avec un diagramme:

```

graph TD
    index[index.php] --> Lire[LireCommentaire.php]
    index --> Ajouter[AjouterCommentaire.php]
    index --> Admin[Administration.php]
    index --> Contacts[Contacts.php]
    Admin --> Config[Configuration.php]
    Admin --> Gestion[GestionArticle.php]
    Gestion --> AjouterArticle[AjouterArticle.php]
    
```

PHP 2005 5

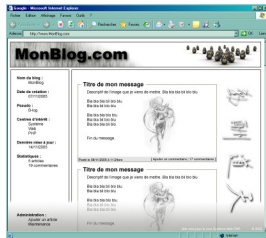
## Conception

- Arborescence visible au niveau des dossiers :
  - ./public\_html/
    - index.php
    - Images
      - Image1.gif
      - Image2.jpeg
    - Administration
      - Administration.php

PHP 2005 6

## Conception

Il faut ensuite dessiner l'aspect que l'on souhaite donner à chacune des pages



PHP

2005

7

## Conception

● Deuxième étape :

- Créer le squelette des pages en HTML
  - Créer l'arborescence
  - Créer les fichiers dans les répertoires
  - Créer les pages en HTML
  - Créer une charte graphique (CSS)

PHP

2005

8

## Conception

● Troisième étape :

- Intégrer les technologies dynamiques (PHP)
  - Lister les fonctionnalités
  - Décrire les données entrantes et les données sortantes de chaque fonctionnalité
  - Procéder par fonctionnalité

PHP

2005

9

## Conception

● Exemple :

- Page de configuration
  - Description :
    - Permet d'entrer les paramètres du propriétaire du blog.
      - Nom, prénom, date de naissance, passions, un commentaire.
  - Cette description sera stockée dans un fichier de configuration : ./administration/config.ini
  - Utilisation d'un seul formulaire pour remplir le fichier de configuration.

PHP

2005

10

## Conception

- L'ergonomie dépend souvent du type de données et de médias à exploiter.
- **Ergonomie et vitesse de chargement** vont bien souvent ensemble.
- Rester plus de 5 secondes devant une page figée est inconfortable.
- Ne faites pas fuir vos visiteurs.

PHP

2005

11

## Conception

● Un site ergonomique est un site dont la **navigation est instinctivement** compréhensible grâce à :

- une unité graphique
- un positionnement judicieux dans la page.
- Il est aussi attirant
  - par son graphisme
  - rapide à charger.

PHP

2005

12

## Temps de chargement

### ● Taille des fichiers

- rapides : < 5Ko
- acceptables : < 15 Ko
- lent : > 15 Ko
- long : > 100 Ko
- vraiment long : > 200 Ko
- trop long : > 500 Ko
- Vraiment bien trop long : > 1 Mo

PHP

2005

13

## Conception

### ● Efficacité

- Utiliser les technologies dynamiques (PHP)
- Code réutilisable
- Penser aux changements de technologie

PHP

2005

14

## Conception

### ● Exemple

- Code réutilisable
  - Entêtes
  - Menus
  - Bas de page
- Changement de technologie
  - Passe d'une gestion de fichier à l'utilisation de base de données

PHP

2005

15

## Conception

- Exemple de fonction :

```
<?php
function entete($titre)
{
    echo "
    <html>
    <head>
    <?xml version='1.0' encoding='utf-8'?>
    <DOCTYPE html PUBLIC "-//W3C/DTD XHTML 1.0 Strict//FR"
    'http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

    <title>$titre</title>
    </head>
    ";
}
?>
```

PHP

2005

16

## Conception

### ● Exercice

- Etape 1 : Faites le diagramme de votre projet.
- Etape 3 : Faites la liste des fonctionnalités.

PHP

2005

17

## Formulaire

### ● Création en HTML

- Balise <form> et </form>
- Balise d'entrée : <input>
- Deux méthodes de passage des paramètres :
  - GET
  - POST

Remarque : en XHTML la syntaxe change légèrement.

PHP

2005

18

## Formulaires

- Ouverture des balises :

```
<form method='POST' action='nom_du_script.php' name='nom_facultatif'>  
ou  
<form method='GET' action='nom_du_script.php' name='nom_facultatif'>
```

- Le champ action est égal à l'adresse du script PHP vers lequel envoyer les données
- Mettre ensuite le code des champs objet.

- Fermeture des balises :

```
</form>
```

PHP

2005

19

## Formulaires

- Champ de texte :

```
<input type='text' name='nom' size='20' value='valeur initiale'>
```

valeur initiale

PHP

2005

20

## Formulaires

- Champ de mot de passe :

```
<input type='password' name='psswd' size='20' >
```

\*\*\*\*\*

PHP

2005

21

## Formulaires

- Zone de texte

```
<textarea rows='3' name='nom' cols='16'>Valeur initiale</textarea>
```

valeur initiale

PHP

2005

22

## Formulaires

- Case à cocher

```
<input type='checkbox' name='nom' value='ON' checked>
```

PHP

2005

23

## Formulaires

- Menu déroulant

```
<select size='1' name='nom'>  
<option value='valeur1'>valeur1</option>  
<option value='valeur2'>valeur2</option>  
<option value='valeur3'>valeur3</option>  
</select>
```

valeur1 ▼  
valeur1  
valeur2  
valeur3

PHP

2005

24

## Formulaires

### Bouton radio

```
<input type='radio' value='val1' checked name='nom_groupe'> a
<input type='radio' value='val2' name='nom_groupe'> b
<input type='radio' value='val3' name='nom_groupe'> c
```

- a
- b
- c

PHP

2005

25

## Formulaire

### Boutons

```
<input type='submit' value='Envoyer' name='nom'>
<input type='reset' value='Annuler' name='nom'>
```

Envoyer

Annuler

PHP

2005

26

## Formulaire

### Les champs cachés

```
<INPUT TYPE="hidden" NAME="variable" VALUE="cachée">
```

- Permet de transmettre des données non visibles sur la page web.
- Attention, ces variables sont visibles dans le code source de la page HTML générée.

PHP

2005

27

## Formulaire

### Création en HTML

```
<form method="post" action="<?=$PHP_SELF ?>">
```

```
Nom:<input type="text" name="nom" size="30"><br>
```

```
Prénom:<input type="text" name="prenom" value="valeur par défaut"><br>
```

```
email:<input type="text" name="email" size="30"><br>
```

```
<input type="submit" name="envoyer" value="OK">
```

```
<input type="reset" name="reset" value="Effacer">
```

```
</form>
```

PHP

2005

28

## Formulaire

Nom:

Prénom:

email:

PHP

2005

29

## Formulaire

### Interaction avec PHP

```
<?PHP
$nom = $_POST['nom'];
$prenom = $_POST['prenom'];
$email = $_POST['email'];

if($_POST['envoyer'] == "OK")
{
    echo "Un formulaire a été rempli et envoyé <br>";
    echo "Nom : $nom<br>";
    echo "Prénom : $prenom<br>";
    echo "email : $email<br>";
}
else
    echo "formulaire vide";
?>
```

PHP

2005

30

## Formulaire

Un formulaire a été rempli et envoyé :

Nom : Fonfec  
Prénom : Sophie  
email : sophie.fonfec@c.naze.com

Nom:   
Prénom:   
email:

PHP

2005

31

## Formulaire

### ● Différence entre GET et POST

#### ○ GET

- Va mettre les données dans l'URL

<http://monsite.com?variable1=valeur1&var2=val2>

#### ○ POST

- Cache les données et ne sont pas directement visibles. Elles sont envoyées dans le corps de la requête HTTP.

PHP

2005

32

## PHP

### ● Manipulation de fichier

- Ouverture de fichier
- Affichage de fichiers
- Lecture de fichiers
- Ecriture de fichiers
- Vérifier l'existence d'un fichier
- Copier coller des fichiers
- Création de répertoire

PHP

2005

33

## PHP

### ● Ouverture de fichier

#### ● fonction fopen() :

`fopen(string nom_du_fichier, string mode);`

#### ● nom\_du\_fichier :

- nom du fichier à ouvrir

#### ● mode :

- mode d'ouverture (lecture, ajout, écriture...) :

#### ● Valeur Opérations permises :

- a Ouverture du fichier pour : écrire et créer le fichier. L'écriture commence à la fin du fichier
- a+ Mêmes fonctions que ci-dessus sauf que la lecture est permise.
- r Ouverture d'un fichier en lecture seule.
- r+ Mêmes fonctions que ci-dessus sauf qu'il est possible d'écrire dans le fichier. L'écriture commence au début du fichier.
- w Ouverture du fichier en écriture seulement. Création du fichier si celui-ci n'existe pas sauf que les données contenues précédemment sont effacées.
- w+ Mêmes fonctions sauf qu'il est possible de lire dans le fichier.

PHP

2005

34

## PHP

### ● Ouverture de fichier

```
< ?php
if($ouverture = @fopen("fichier.txt", "r"))
{
    echo "L'ouverture du fichier est possible car la
    fonction fopen retourne TRUE";
}
else
{
    echo "Ouverture du fichier impossible car fopen
    retourne FALSE";
}
?>
```

PHP

2005

35

## PHP

### ● Affichage de fichiers

- Pour afficher tout le contenu d'un fichier dans le navigateur, on utilise la fonction `fpassthru()` :

`fpassthru(string pointeur);`

- Exemple d'envoi d'un fichier `essai.txt` au navigateur :

```
< ?php
$fichier = fopen("essai.txt", "r");
fpassthru($fichier);
?>
```

- Remarque : la fonction `readfile()` est similaire et utilise successivement un `fopen` et un `fpassthru`.

PHP

2005

36

● Lecture de fichiers

- Pour n'afficher qu'une partie d'un fichier, il existe plusieurs fonctions différentes :

- fgetc()
- fgets()
- fgets()
- fread()
- file()

● Lecture de fichiers

- fgetc()
  - extrait le premier caractère du fichier :

```
fgetc(string pointeur);
```

● Exemple :

```
< ?php
    $fichier = fopen("essai.txt","r");
    $premier = fgetc($fichier);
    echo "Premier Caractère : " . $premier;
    fclose($fichier);

?>
```

● Lecture de fichiers

- fgets()
  - extrait une chaîne d'une certaine longueur.
  - extrait la chaîne de la longueur précise définie en argument, plus un caractère
  - la fonction s'arrête aux sauts de lignes :

```
fgets(string pointeur, string longueur);
```

● Exemple :

```
< ?php
    $fichier = fopen("essai.txt","r");
    $premier = fgets($fichier, 10);
    echo "Dix Premier Caractères : " . $premier;
    fclose($fichier);

?>
```

● Lecture de fichiers

- fgets()
  - La fonction fgets() a quasiment la même utilité que la précédente sauf qu'elle n'extrait ni les balises HTML, ni les balises PHP :

```
fgets(string pointeur, string longueur);
```

● Exemple :

Pour un fichier contenant :

```
'< b>Bonjour !< /b>< br>Test de la fonction fgets'
```

la fonction ne renverra que les caractères, la mise en forme HTML sera ignorée :

```
'Bonjour !Test de la fonction fgets'
```

● Lecture de fichiers

- fread()
  - Lit une chaîne de caractère dans un fichier ouvert, jusqu'à la longueur indiquée en argument :

```
fread(string pointeur, string longueur);
```

● Exemple :

```
< ?php
    $fichier = fopen("essai.txt","r");
    $premier = fread($fichier, 10);
    echo "Dix Premiers Caractères : " . $premier;
    fclose($fichier);

?>
```

● Lecture de fichiers

- file()
  - Met le contenu entier d'un fichier ouvert dans un tableau:

```
file(string fichier);
```

● Exemple :

```
< ?php
    $premier = file("essai.txt");
    echo "Première Ligne du fichier : " . $premier[0];

?>
```

● **Ecriture de fichiers**

- Pour écrire dans un fichier, on peut utiliser au choix la fonction `fwrite()` ou `fputs()` :

```
fwrite(string pointeur, string chaîne, (string longueur));
fputs(string pointeur, string chaîne, (string longueur));
```

- Le paramètre longueur permet de limiter le nombre de caractères qui pourra être écrit dans le fichier. Arrivé a cette longueur, la fonction s'arrête.

● **Ecriture de fichiers**

- Exemple :

```
< ?php
$fichier = fopen("essai.txt","w");
if(fwrite($fichier, "TEXTE A ECRIRE"))
{
    echo "OK !";
}
else
    echo "Erreur";
fclose($fichier);
?>
```

- La fonction `fwrite()` retourne TRUE si l'écriture se passe normalement, sinon elle retourne FALSE.
- Attention : le fichier doit bien être ouvert en mode écriture : w, a ou r+.

● **Vérifier l'existence d'un fichier**

- Pour vérifier si un fichier existe, on utilise la fonction `file_exists()`:

```
file_exists(string fichier);
```

- L'argument fichier est le chemin permettant d'y accéder. (Ex : `inclu/file.txt`)

- Exemple :

```
< ?php
if(file_exists("essai.txt"))
{
    echo "Fichier existant ";
}
else
    echo "Introuvable !";
?>
```

● **Copier coller des fichiers**

- Pour copier, supprimer ou encore renommer des fichiers, PHP vous propose ces fonctions :

- Copier → `copy()`
- Renommer → `rename()`
- Supprimer → `unlink()`

● **Copier coller des fichiers**

- Pour copier un fichier, il faut utiliser la fonction `copy()` :

```
copy(string fichier_depart, string fichier_destination);
```

- La fonction retourne TRUE si tout se passe correctement et sinon FALSE.

- Exemple :

```
< ?php
if(@copy("essai.txt","complet/copy_fichier.txt"))
{
    echo "Le fichier essai.txt a été copié dans le répertoire complet/copy_fichier.txt.";
}
else
    echo "Erreur";
?>
```

● **Copier coller des fichiers**

- Pour renommer un fichier, il faut utiliser la fonction `rename()` :

```
rename(string nom_depart, string nom_nouveau);
```

- Exemple :

```
< ?php
if(@rename("essai.txt","nouveau_fichier.txt"))
{
    echo "Le fichier essai.txt a été renommé en nouveau_fichier.txt.";
}
else
    echo "Erreur";
?>
```

## PHP

### • Copier coller des fichiers

- Pour supprimer un fichier, c'est la fonction `unlink()` que l'on utilise:

```
unlink(string fichier);
```

L'argument fichier correspond au chemin complet du fichier.

- Exemple :

```
<?php
if(@unlink("essai.txt"))
{
    echo "Le fichier essai.txt a été supprimé.";
}
else
    echo "Erreur";
?>
```

PHP

2005

49

## PHP

### • Création de répertoire

- bool `mkdir` ( string `pathname` , int `mode` ) `mkdir` tente de créer un dossier dans le chemin `pathname`.

- Notez que vous aurez à préciser le mode en base octale, ce qui signifie que vous aurez probablement un 0 comme premier chiffre.

- Le mode par défaut est le mode 0777, ce qui correspond au maximum de droits possible.

- Exemple avec `mkdir`

```
<?php
mkdir ("/chemin/de/mon/dossier", 0700);
?>
```

- Cette fonction retourne TRUE en cas de succès, FALSE en cas d'échec.

PHP

2005

50

## PHP

### • Soumettre des fichiers (upload)

- Pouvoir soumettre de images
- Toujours limiter la taille des fichiers soumis
- Soumission à l'aide de formulaire et de PHP

PHP

2005

51

## PHP

### • Soumettre des fichiers

- Formulaire

```
<FORM method="POST" ENCTYPE="multipart/form-data">
<INPUT type=hidden name=MAX_FILE_SIZE VALUE=2048>
<INPUT type=file name="nom_du_fichier">
<INPUT type=submit value="Envoyer">
</FORM>
```

PHP

2005

52

## PHP

### • Soumettre des fichiers

- Le fichier, ainsi que les informations le concernant, sont accessibles via la variable superglobale `$_FILES[]`.

- Pour afficher son contenu :

```
<? print_r($_FILES); ?>
```

- La sortie de ce code sera de la forme suivante :

```
Array ( [nom_du_fichier] => Array (
    [name] => MaBelleImage.jpg
    [type] => image/jpeg
    [tmp_name] => chemin_complet_du_fichier_uploadé
    [error] => 0 [size] => 1000 ) )
```

Dans le cas ci-dessus il s'agit d'une image JPEG pesant 1 Mo.

PHP

2005

53

## PHP

### • Soumettre des fichiers

- Les erreurs peuvent être traitées de la façon suivante

```
<?
if ($_FILES["nom_du_fichier"]["error"])
{
    switch ($_FILES["nom_du_fichier"]["error"])
    {
        case 1: // UPLOAD_ERR_INI_SIZE
            echo "Le fichier dépasse la limite autorisée par le serveur (fichier php.ini)";
            break;
        case 2: // UPLOAD_ERR_FORM_SIZE
            echo "Le fichier dépasse la limite autorisée dans le formulaire HTML.";
            break;
        case 3: // UPLOAD_ERR_PARTIAL
            echo "L'envoi du fichier a été interrompu pendant le transfert.";
            break;
        case 4: // UPLOAD_ERR_NO_FILE
            echo "Le fichier que vous avez envoyé a une taille nulle.";
            break;
    }
}
else
{
    // $_FILES["nom_du_fichier"]["error"] vaut 0 soit UPLOAD_ERR_OK
    // ce qui signifie qu'il n'y a eu aucune erreur
}
?>
```

PHP

2005

54

● Soumettre un fichier

- Grâce à la fonction `move_uploaded_files()` il est possible de transférer l'image du répertoire temporaire vers un répertoire de destination :

```
<?
if ((isset($_FILES['nom_du_fichier']) && ($_FILES['nom_du_fichier']['error'] == UPLOAD_ERR_OK))
{
    $chemin_destination = '/home/etudiants/chezmoi/';
    move_uploaded_file(
        $_FILES['nom_du_fichier']['tmp_name'],
        $chemin_destination.$_FILES['nom_du_fichier']['name']
    );
}
?>
```

● Sessions

- Une session est un fichier conservé sur le serveur et accessible par les scripts en fonction d'un identifiant généré à la création.
- Chaque fois qu'un visiteur génère une session, un identifiant lui est attribué.
- Tout ce qui est dans cette session est accessible à tous les scripts.

● Sessions

- Alternative aux cookies
- Permet de garder des informations propres à un visiteur

● Exemple :

- Session administrateur après une connexion avec mot de passe.

● Sessions

- **session\_start();**
  - Démarre une session OU appelle la session existante.
  - Elle doit donc être présente sur toutes les pages pour garder la session 'en vie'.
  - Prendre l'habitude de la placer au début du fichier, avant quoi que ce soit d'autre et tout se passera bien.
- **session\_destroy();**
  - Cette fonction détruit la session en cours.

● Sessions

- **session\_register(nom1, nom2, ...)**
- Cette fonction permet de stocker une variable dans une session.
- Son emploi demeure simple, il suffit de passer le nom de la variable à stocker en tant qu'argument.

```
<?php
session_start();
$login = 'maverick';
$password = '1234';
session_register('login', 'password');
?>
```

- Cet exemple stockera les variables `$login` et `$password` dans la session, et par cette méthode, elles seront accessibles pour les autres pages affichées durant cette session.

● Sessions

● **session\_unset()**

Cette fonction, comme dit plus haut, détruit les variables apportées par la session. Pas d'argument, juste un simple appel.

# PHP

- Sessions

- Au moment de l'identification :

```
<?php
session_start();
if ( $is_identified )
{
    $member = 1;
    session_register('member');
}
else
{
    /* Affichage du formulaire
    d'identification */
}
?>
```

- Et pour l'accès aux fonctions de l'espace membre :

```
<?php
session_start();
if ($member == 1)
{
    /* Accès aux fonctions de
    l'espace membre */
}
else
{
    echo 'Il faut être identifié pour
    accéder à cette section';
}
?>
```

PHP

2005

61

# Projet

- Idée pour la gestion de messages

- Un répertoire par article

- Nom :
  - timestamp
- Contenu :
  - les messages lié à cet article

- Un fichier par commentaire

- Nom :
  - timestamp
- Contenu :
  - la première ligne, l'auteur
  - ensuite le commentaire

PHP

2005

62