

# INTRODUCTION À PHP

# Sommaire

---

- ▶ Introduction
- ▶ Les bases
- ▶ Fichiers et répertoires
- ▶ Traitement des chaînes de caractère
- ▶ Bases de données
- ▶ Gestion de session
- ▶ Mails et Imap

# Historique

---

- ▶ 1994 : Personal Home Page (Scripts Perl)
- ▶ 1995 : moteur de script pour interpréter les formulaires PHP/FI (PHP2)
- ▶ 2001 : PHP4 fondé sur le moteur de script Zend ([www.zend.com](http://www.zend.com))
- ▶ 2004 : PHP5
- ▶ Version 7 : depuis 2015

# Caractéristiques

---

- ▶ Open Source
- ▶ Multi-plateforme
- ▶ Langage interprété coté serveur
- ▶ Langage conçu pour le Web
- ▶ Module (Apache) ou interpréteur CGI (IIS...)

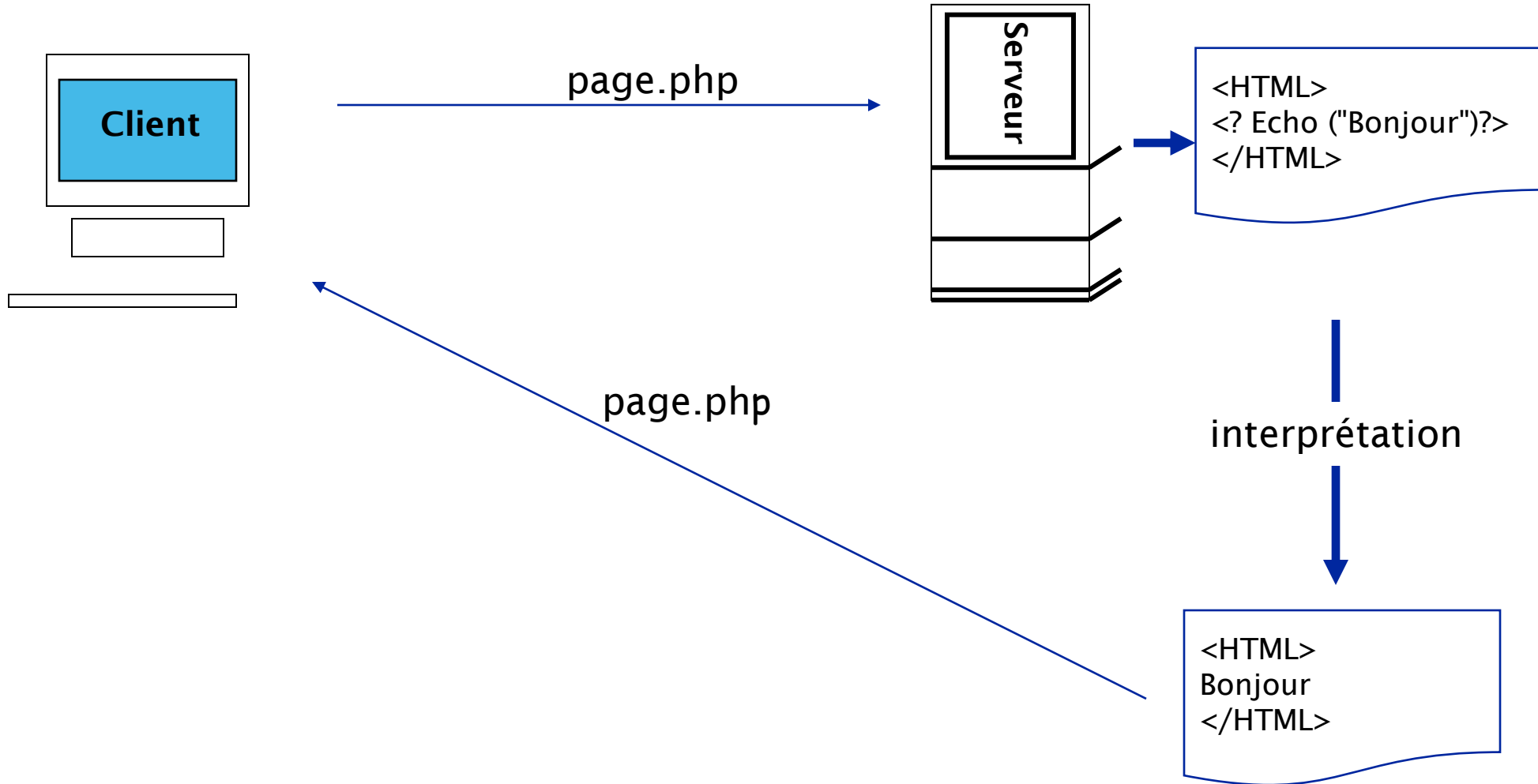
# Modules PHP (API)

---

- ▶ Connexion SGBD : Oracle, MySQL, Informix, Sybase
- ▶ Annuaires LDAP (Light-weight Directory Access Protocol)
- ▶ XML
- ▶ IMAP et SMTP
- ▶ PDF
- ▶ SNMP (gestion réseau)

# Les bases

# Fonctionnement



# Ecrire en PHP

---

```
<?php
```

```
echo ("Bonjour");  
echo ("<H1> Bonjour</H1>");  
echo ("  
<Script Language='javascript'> Alert('Attention')  
</script>")
```

```
$monNom="Renaud";  
echo ("  
<Script Language='javascript'> Alert('Bonjour $monNom')  
</script>")
```

```
?>
```



# Caractère d'échappement

- ▶ le caractère \ permet de déspecifier un caractère spécial en PHP :

- \" affiche le caractère "
- \\ affiche le caractère \
- \\$ affiche le caractère \$
- \' affiche le caractère '
- \n saut à la ligne
- \t tabulation

exemple

```
echo("<A href=\"cv.php?candidat=toto\">")
```

# Opérateurs

## ► Concaténation :

```
$nom = "Dupont";  
$prenom = "Jean";
```

```
echo("Bonjour $prenom $nom")           //interpolation  
echo("Bonjour " . $prenom . " " . $nom) //concaténation
```



**Bonjour Jean Dupont**

```
$sql_query= "SELECT Nom, Prenom ".  
            "FROM Carnet ".  
            "WHERE Codepost=94000 ".  
            "ORDER by Nom";
```

**Auto-concaténation :**

**\$a .= \$i équivalent à \$a = \$a . \$i**

# Opérateurs et fonctions utiles

---

```
$val == 1 ? echo "oui" : echo "non"  
echo ($val == 1 ? "oui" : "non")
```

Renvoie "oui" si l'égalité est vérifiée et "non" sinon

```
gettype($a)  
settype($a, "string")
```

renvoie le type de \$a (string, integer...)  
affecte un type à \$a

```
$toto = (int) $toto  
isset($a) renvoie true
```

transtypage  
si \$a est définie

```
empty($a) renvoie true
```

si \$a n'est pas définie,  
est nulle ou égale à 0

# Tableaux

---

```
$nom[]="Dupond";  
$nom[]="Durand";  
$nom[]="Martin";
```

```
$nom[0]="Dupond";  
$nom[1]="Durand";  
$nom[2]="Martin";
```

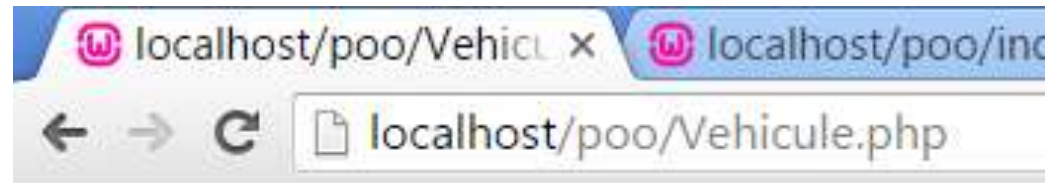
```
$nom=array("Dupond","Durand","Martin");
```

```
echo(count($nom)); affiche 3
```

# Tableaux associatifs et interpolation

```
$tab["nom"]="Dupond";  
$tab["prenom"]="Jean pierre";  
$tab["age"]=34;
```

```
$tab=array("nom"=>"Dupond", "prenom" => "Jean Pierre" ,  
"age" => 34);  
var_dump($tab);
```



```
array (size=3)  
    'nom' => string 'Dupond' (length=6)  
    'prenom' => string 'Jean Pierre' (length=11)  
    'age' => int 34
```

**X** echo "mon nom est \$tab["nom"] "  
**V** echo "mon nom est {\$tab["nom"]} "

# Parcourir un tableau

## ► Parcourir un tableau indexé séquentiellement

```
$nom=array("Dupond","Durand","Martin");  
$nbr_elt=count($nom);
```

```
for ($i=0;$i<$nbr_elt;$i++) {  
    echo ($nom[$i]. " <BR>");  
}
```

## ► Autre méthode (intéressante pour les tableaux associatifs) : boucle foreach

```
$nom=array("Dupond","Durand","Martin");
```

```
foreach($nom as $key => $value){  
    echo("la valeur de l\'indice " . $key . "est " . $value);  
} while (next($nom));
```

# Fonctions pour les tableaux

## ► Vérifier qu'une clé existe

```
$tab=array("nom"=>"Dupond", "prenom" => "Jean Pierre" ,  
"age" => 34);  
if (array_key_exists('nom', $tab))  
{  
    echo 'La clé nom se trouve dans les coordonnées !';  
}
```

## ► Vérifier qu'une valeur existe

```
$nom=array("Dupond","Durand","Martin");  
  
if (in_array('Durand', $nom))  
{  
    echo 'La valeur Durand se trouve dans les noms';  
}
```

# Tableaux superglobaux

---

- ▶ Depuis 4.2.0 (EasyPHP1.7)
- ▶ Une variable **superglobale** est disponible dans tout contexte d'utilisation, que ce soit dans le script, dans une méthode de classe, dans une fonction...
- ▶ Ces variables sont des tableaux associatifs. C'est-à-dire qu'à une clé est associée une valeur pouvant être un nombre, une chaîne de caractères, un tableau... Les clés de ces tableaux sont les noms des variables qu'ils référencent.



# Principaux tableaux superglobaux

---

- ▶ `$_GET` : Contient les variables fournies en paramètre au script via la méthode GET du protocole HTTP.
- ▶ `$_POST` : Contient les variables fournies par un formulaire via la méthode POST du protocole HTTP.
- ▶ `$_FILES` : Contient les variables fournies suite à un chargement de fichier par un formulaire via la méthode POST du protocole HTTP.
- ▶ `$_SESSION` : Contient les variables de la session en cours dans le script.

# Récupération des paramètres d'un formulaire

## Fichier test.php

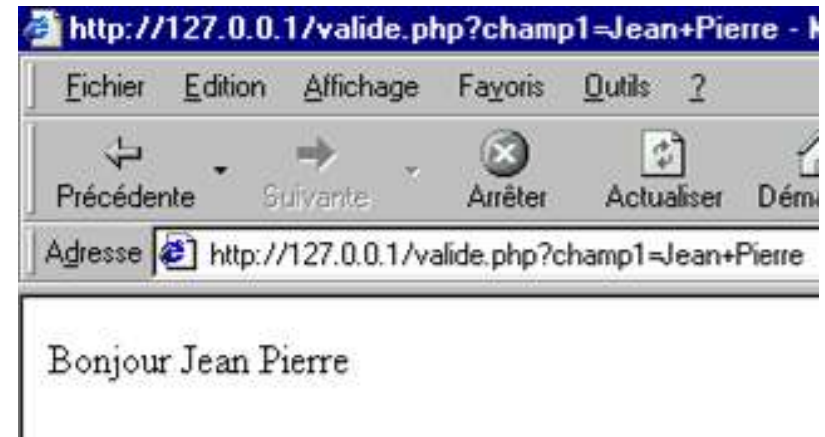
Entrez votre nom :

```
<FORM action="valide.php">
<INPUT type="text" name="champ1">
<INPUT type="submit"
value="Cliquez ici">
</FORM>
```



## Fichier valide.php

```
<html>
<?php
echo ("Bonjour " . $_GET["champ1"] );
?>
</html>
```



# Vérification des checkboxes

Entrez votre nom :

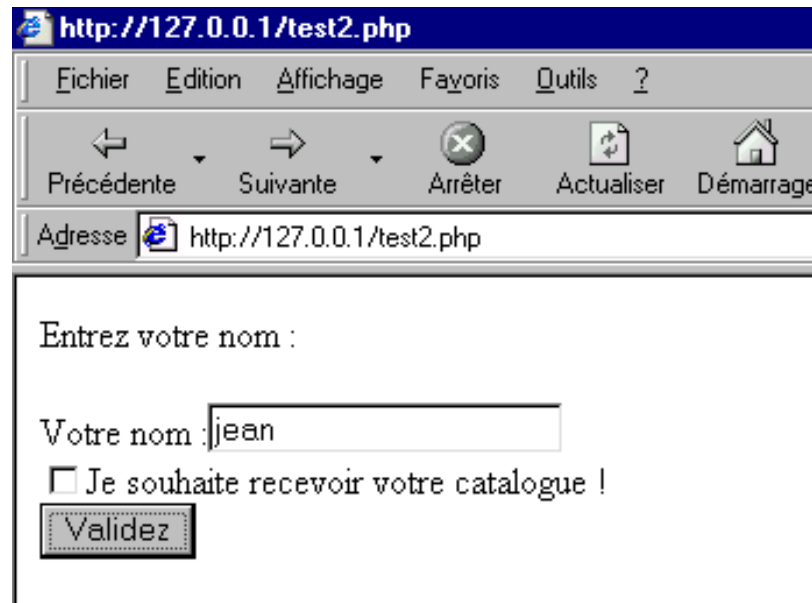
```
<FORM action="valide.php">
```

```
Votre nom :<INPUT type="text" name="champ1"><BR>
```

```
<INPUT type="checkbox" name="pub">Je souhaite recevoir votre catalogue !<BR>
```

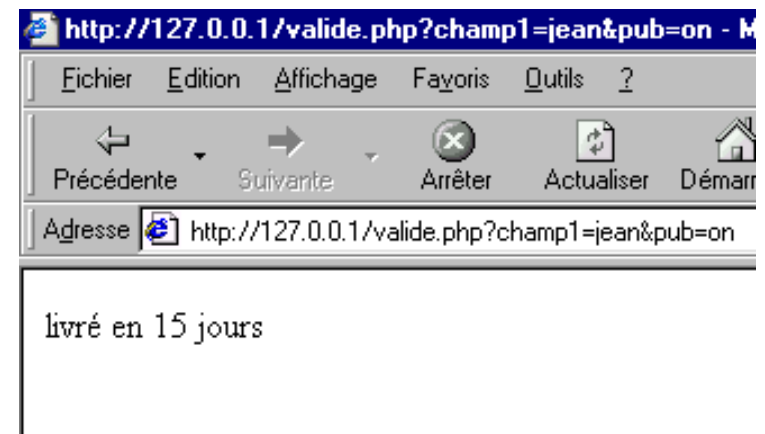
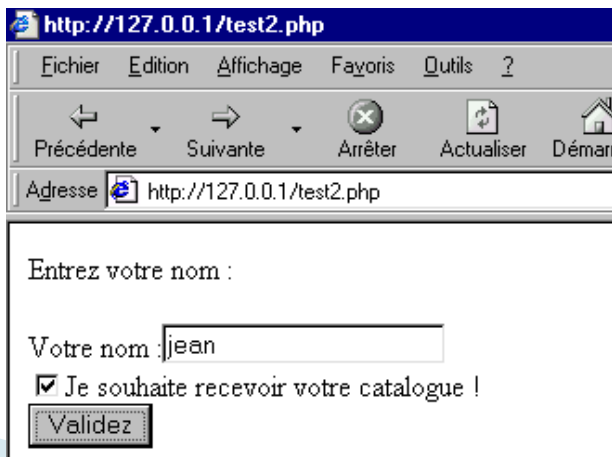
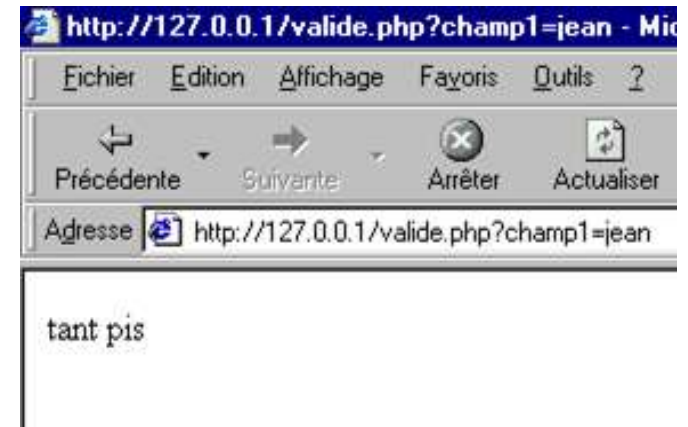
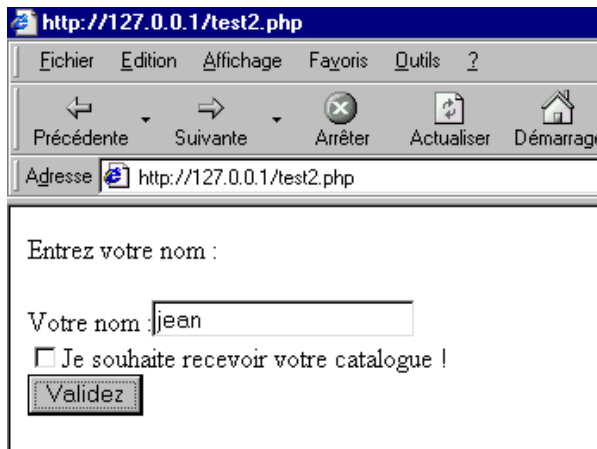
```
<INPUT type="submit" value="Validez">
```

```
</FORM>
```



# Vérification des checkboxes

```
<?php  
echo (isset($_GET["pub"])) ? "livré en 15 jours" : "tant pis");  
?>
```



# Tableaux / formulaires / checkboxes

```
<form action="calcul.php " method=post>
Quels sont vos couleurs préférées?<br>
<input type="checkbox" name="choix[]" value="Red">Red
<input type="checkbox" name="choix[]" value="Blue">Blue
<input type="checkbox" name="choix[]" value="Green">Green
<input type="submit" value="Voir le résultat!"></form>
```

## Code php qui traite le formulaire :

```
<?php
$choix = $_POST['choix'];
echo("Vos couleurs préférées sont : ");
for($i=0;$i<sizeof($choix);$i++)
    { { echo("$choix[$i] - "); }
?>
```

# Encoder une URL

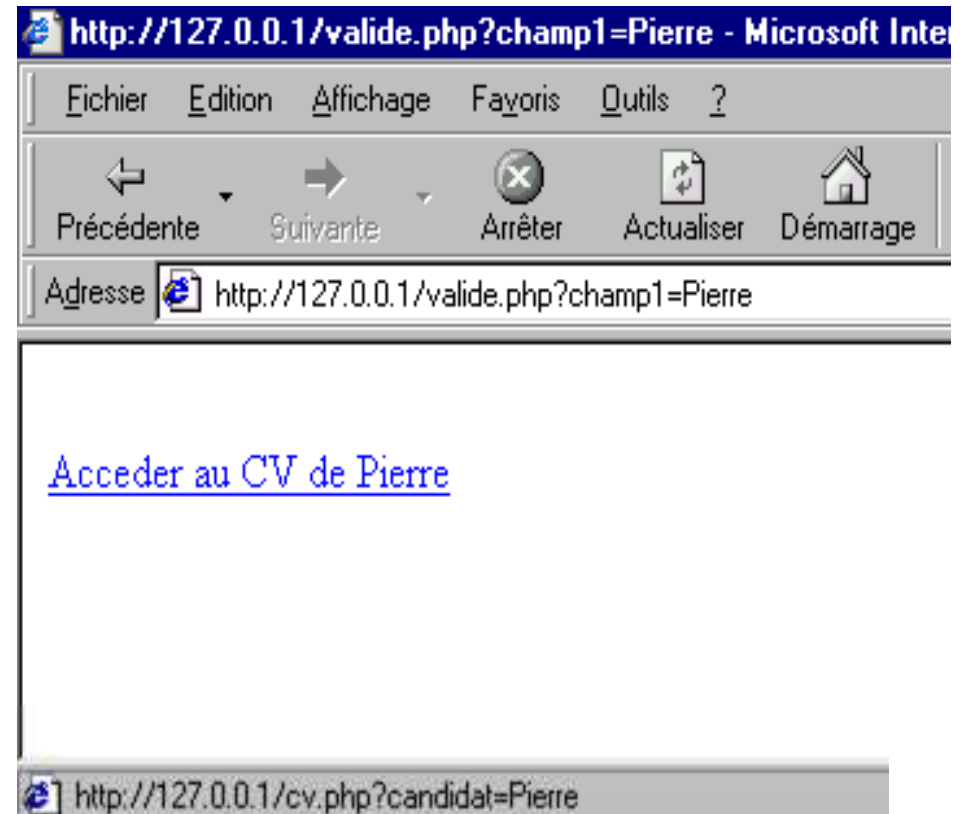
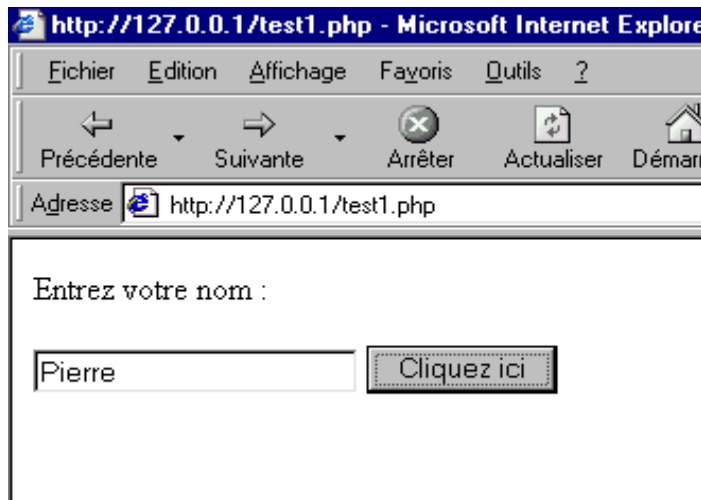
<html>

<?php \$champ1=\$\_GET["champ1"];?>

<A href="cv.php?candidat=<?php echo (\$champ1); ?>">

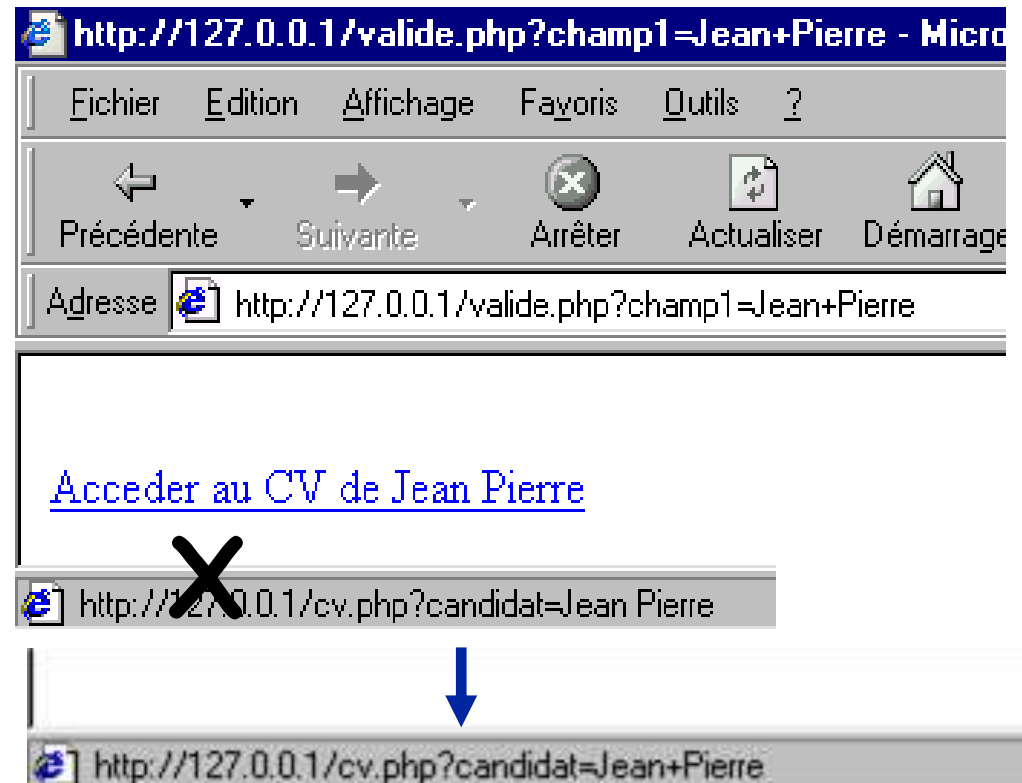
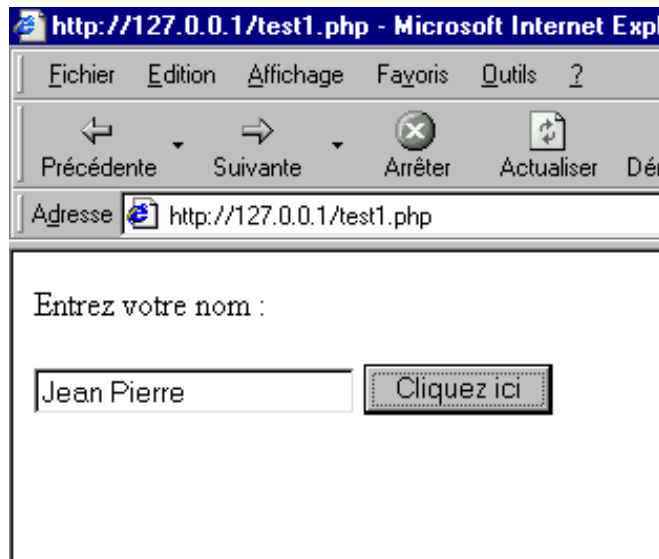
<BR>Acceder au CV de <?php echo (\$champ1); ?> </A>

</html>



# Encoder une URL


```
<html><? $champ1=$_GET["champ1"];?>  
<A href="cv.php?candidat=<?php echo (urlencode($champ1)); ?>">  
<BR>Acceder au CV de <?php echo ($champ1); ?> </A>  
</html>
```



# Interprétation

```
<html>
<A href="cv.php?candidat=<?php echo
(urlencode($champ1)); ?>">
<BR>Acceder au CV de  <?php echo ($champ1); ?> </A>
</html>
```

```
<html>
<?php echo("<A href=\"cv.php?candidat=\" . urlencode($champ1) . "\">") ; ?>
<BR>Acceder au CV de  <?php echo ($champ1); ?> </A>
</html>
```



```
<A href="cv.php?candidat=Jean+Pierre"><BR>Acceder au CV de  Jean Pierre </A>
```



# Incorporer un fichier dans une page PHP

---

- ▶ L'instruction “require” est remplacée par le contenu du fichier lors de l'interprétation
- ▶ L'instruction “include” évalue le contenu du fichier mais celui-ci n'est pas écrit dans le fichier

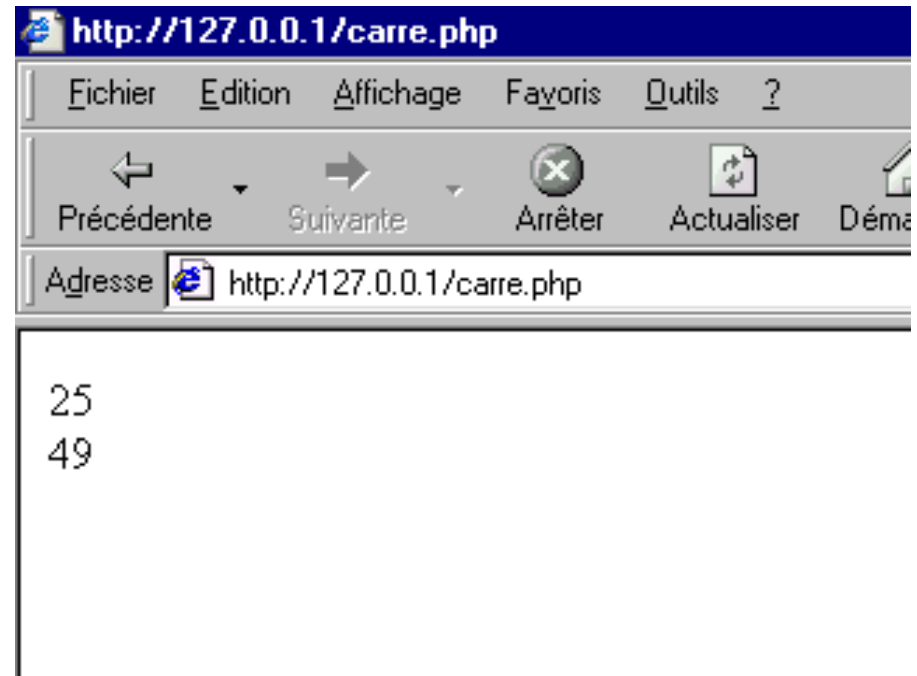
```
<?php  
require ("definitions.php");  
  
include ("toto.php")  
?>
```

# Fonctions

```
<?php
function carre($x) {
return $x*$x;
}

echo (carre(5) . "<BR>");
echo (carre(7));

?>
```



# Regles d'écriture

---

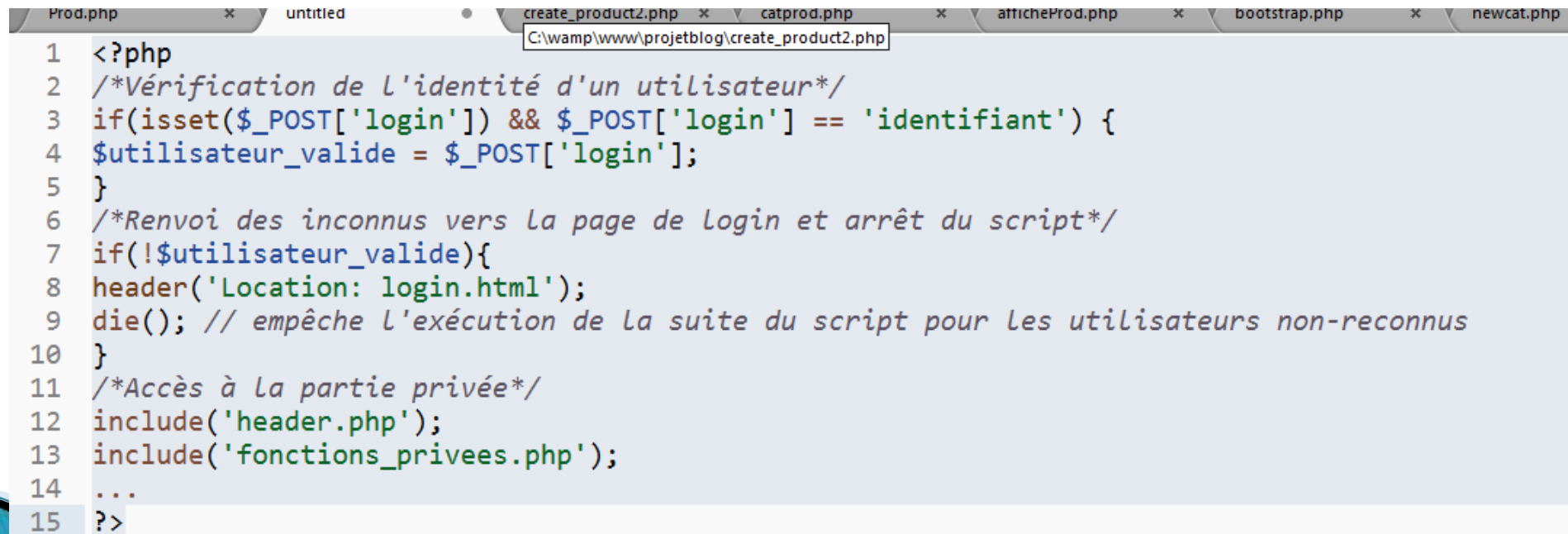
- ▶ Généralement on évite de placer des **echo** dans les fonctions : on **return** le résultat et on l'affiche à l'aide d'un **echo** hors de la fonction

```
function carre($x){  
    return $x*$x;  
}  
echo carre(5)
```

# Exit et Die

- ▶ Les fonctions **exit('message')** et **die('message')** sont identiques. Elles permettent de stopper un script.

```
$result=mysqli_query($connexion,$requete) or die ('erreur requete');  
if( !is_uploaded_file($tmp_file) ) {exit ('erreur upload'); }
```



The screenshot shows a code editor with several tabs open: 'Prod.php', 'untitled', 'create\_product2.php', 'catprod.php', 'afficheProd.php', 'bootstrap.php', and 'newcat.php'. The active tab is 'create\_product2.php', which contains the following PHP code:

```
1 <?php  
2 /*Vérification de l'identité d'un utilisateur*/  
3 if(isset($_POST['login']) && $_POST['login'] == 'identifiant') {  
4 $utilisateur_valide = $_POST['login'];  
5 }  
6 /*Renvoi des inconnus vers la page de login et arrêt du script*/  
7 if(!$utilisateur_valide){  
8 header('Location: login.html');  
9 die(); // empêche l'exécution de la suite du script pour les utilisateurs non-reconnus  
10 }  
11 /*Accès à la partie privée*/  
12 include('header.php');  
13 include('fonctions_privees.php');  
14 ...  
15 ?>
```

# Débogage / affichage d'erreur

---

- ▶ `print_r()` ; affiche des informations à propos d'une variable, de manière à ce qu'elle soit lisible
- ▶ `var_dump()` ; affiche les informations structurées d'une variable, y compris son type et sa valeur.

# Exemple

```
form_cours.php  debug.php x
1 <html>
2 <body>
3
4 <?php
5 $a = array ('a' => 'apple', 'b' => 'banana', 'c' => array ('x', 'y', 'z'));
6 print_r ($a);
7 echo "<br>";
8 var_dump($a);
9 ?>
10 </body>
11 </html>
12
```

Array ( [a] => apple [b] => banana [c] => Array ( [0] => x [1] => y [2] => z ) )

```
array (size=3)
  'a' => string 'apple' (length=5)
  'b' => string 'banana' (length=6)
  'c' =>
    array (size=3)
      0 => string 'x' (length=1)
      1 => string 'y' (length=1)
      2 => string 'z' (length=1)
```

# Auto-redirection et redirection

---

```
<? if (!isset($_POST['process'])) { ?>
```

```
<FORM METHOD="POST" ACTION="<? echo $_SERVER['PHP_SELF'] ?>">
```

```
<input type=test name="process">
```

```
.....
```

```
</FORM>
```

```
<?
```

```
} else {...}
```

```
header('Location:toto.php');
```

```
?>
```

Attention header doit s'employer **AVANT** toute écriture sur la page (en html comme en php)

# Date

---

## La fonction date()

- ▶ Renvoie la date courante du serveur

```
echo date ("d-m-y") ;
```

- ▶ Renvoie la date courante de la machine sous la forme "12-10-06«

```
echo date ("d/m/y", 1172575871) ;
```

- ▶ Renvoie la date correspondante à la 1172575871ème seconde écoulée depuis 1970 soit "27/02/07"



# fichiers et répertoires

# Ouvrir un fichier

---

```
fopen(nom_fichier,mode);
```

modes :	a	ajout. Les données sont écrites à la fin du fichier
	a+	ajout et lecture. idem
	r	lecture seule
	r+	lecture et écriture. Les données sont écrites au début du fichier.
	w	écriture. Les données remplacent le contenu du fichier.
	w+	écriture-lecture. Idem
	b	indique un fichier binaire et non en mode texte

## Exemple :

```
$fic=fopen("toto.txt","w+");  
$image=fopen("voiture.gif","rb");
```

# Manipulation de fichiers

---

- ▶ `fpassthru($fic)` permet d'afficher le contenu du fichier puis le referme
- ▶ `filesize("nom_fichier")` retourne le nombre de caractères du fichier
- ▶ `fread($fic, y)` retourne les y premiers caractères du fichier
- ▶ `fgets ($fic, y)` retourne les y premiers caractères de la ligne courante
- ▶ `fgetss ($fic, y)` retourne les y premiers caractères de la ligne courante sans afficher les balises php ou HTML (celles ci sont néanmoins comptabilisées lors de la lecture)

# Manipulation de fichiers

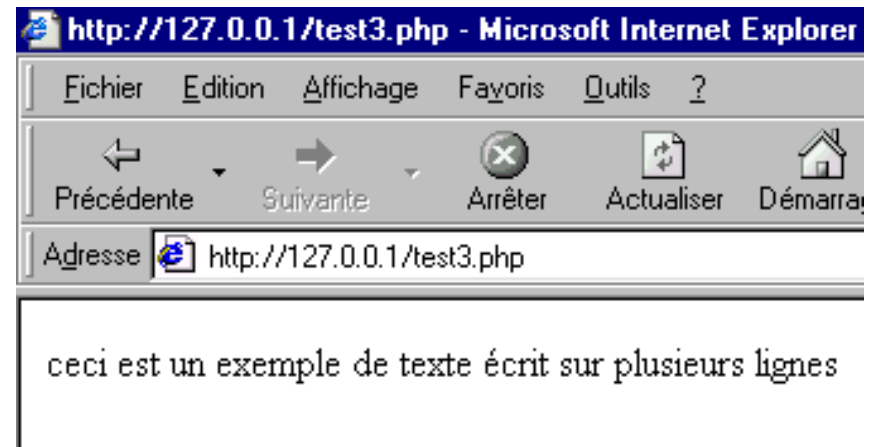
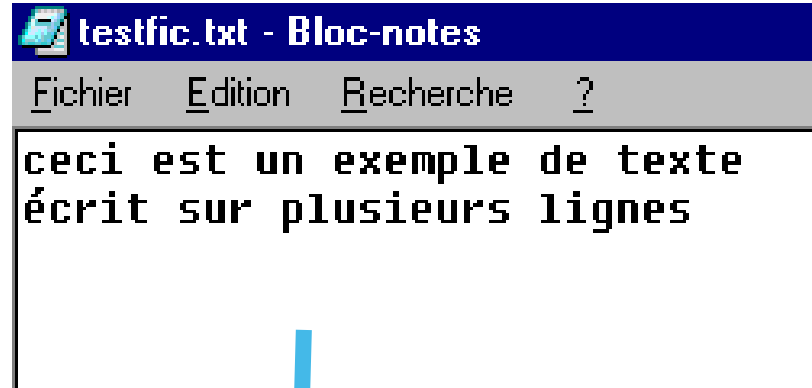
---

- ▶ `fputs($fic, "<BR>bonsoir")` écrit dans le fichier à la position courante
- ▶ `fwrite($fic, "<BR>bonsoir")` écrit dans le fichier à la position
- ▶ `rewind($fic)` place l'indicateur de position en début de fichier
- ▶ `fseek($fic, y)` déplace l'indicateur de position de y caractères
- ▶ `fclose($fic)` referme le fichier

# Exemples

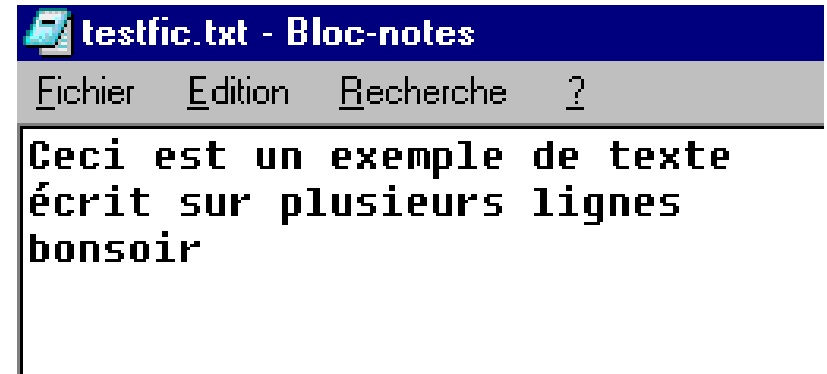
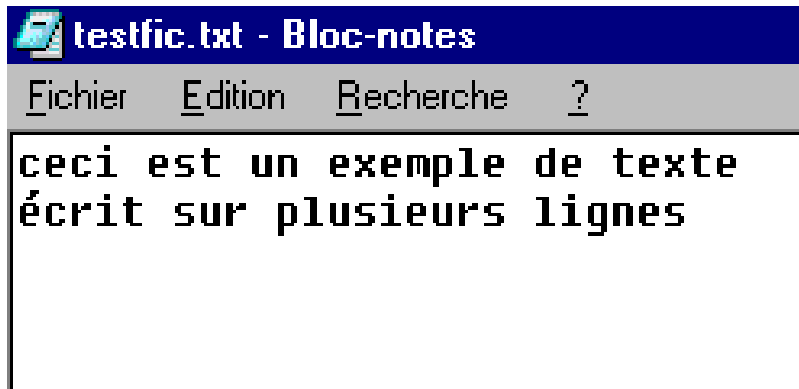
fichier test3.php :

```
<?php
if (!$fic=fopen("testfic.txt","r")){
echo("echec de l'ouverture du fichier");
} else {
fpassthru($fic);
}
?>
```



# Exemples

```
<?php
if (!$fic=fopen("testfic.txt","a")){
echo("echec de l'ouverture du fichier");
} else {
fputs($fic,"\nbonsoir");
fclose($fic);
}
?>
```



# Uploader un fichier

---

## Formulaire :

```
<form method="post" enctype="multipart/form-data"
action="upload.php">

<input type="file" name="fichier" size="30">
<input type="submit" name="upload" value="Uploader">
</form>
```

# Traitement

---

Le fichier uploadé est accessible via `$_FILES` :

- ▶ `$_FILES['fichier']['name']`
  - Contient le nom d'origine du fichier
- ▶ `$_FILES['fichier']['type']`
  - Contient le type MIME du fichier
- ▶ `$_FILES['fichier']['size']`
  - Contient la taille du fichier en octets
- ▶ `$_FILES['fichier']['tmp_name']`
  - Le nom temporaire du fichier qui sera chargé sur la machine serveur (ex : `/home/uploads/FHGJKYUGTDFR`).



## Traitement (2)

---

- ▶ Vérifier que l'opération s'est bien passée en vérifiant la présence du fichier dans le dossier temporaire → pour cela, nous avons à notre disposition la fonction is\_uploaded\_file().
- ▶ Vérifier ce qui nous a été envoyé; Si on s'attend à une image, on vérifie si l'extension est correcte par exemple. (optionnel)
- ▶ Copier le fichier sur notre espace web à l'aide de la fonction move\_uploaded\_file() Copier le fichier sur notre espace web à l'aide de la fonction move\_uploaded\_file(), plus sûre que la fonction copy(), car elle vérifie que le fichier à copier vient bien du dossier temporaire (et donc, provient d'un formulaire d'upload).

## Traitement (3)

```
<?php if( isset($_POST['upload']) ) // si formulaire soumis
{
$content_dir = 'upload/'; // dossier où sera déplacé le fichier
$tmp_file = $_FILES['fichier']['tmp_name'];
if( !is_uploaded_file($tmp_file) )
{
exit("Le fichier est introuvable");
}

// on copie le fichier dans le dossier de destination

$name_file = $_FILES['fichier']['name'];
if( !move_uploaded_file($tmp_file, $content_dir . $name_file) )
{
exit("Impossible de copier le fichier dans $content_dir");
}
echo "Le fichier a bien été uploadé"; } ?>
```

# Fonctions de bases de données

# L'API MySQLi

---

## ► Connexion

```
$connexion=mysqli_connect(hostname,user,passwd)
```

## ► Sélection de la base

```
mysqli_select_db($connexion, "nomBase") ;
```

## ► Requête

```
mysqli_query(connexion,requete)
```

phpMyAdmin 2.2.0rc4 - localhost - Microsoft Internet Explorer

Fichier Edition Affichage Favoris Outils ?

Précédente Suivante Arrêter Actualiser Démarrage Rechercher Favoris Historique Courrier Imprimer Edition Discusi

Adresse <http://localhost/mysql/>

**Accueil**

- ☐ Carnet
- ☒ [adresse](#)
- ☐ mysql
- ☐ test
- ☐ vin

## Base de données Carnet - table adresse

[Afficher](#)

Champ	Type	Attributs	Null	Défaut	Extra	Action
Nom	varchar(30)		Non			<a href="#">Modifier</a> <a href="#">Supprimer</a> <a href="#">Primaire</a> <a href="#">Index</a> <a href="#">Unique</a>
Prenom	varchar(30)		Non			<a href="#">Modifier</a> <a href="#">Supprimer</a> <a href="#">Primaire</a> <a href="#">Index</a> <a href="#">Unique</a>

Index :

Nom de la clé Unique	Champ	Action
PRIMARY	Nom	<a href="#">Supprimer</a>
Nom	Nom	<a href="#">Supprimer</a>

[\[Documentation\]](#)

Espace utilisé :

Type	Espace
Données	60 Octets
Index	3 072 Octets
Total	3 132 Octets

Statistiques :

Information	Valeur
Format	dynamique
Enregistrements	3
Longueur enr. ø	20
Taille enr. ø	1 044 Octets

# Exemple

---

```
<html>
<body>
<?php
$user="php";
$password="php";
$host="localhost";
$nomBase="Carnet";
$nomTable="adresse";

$connexion=mysqli_connect($host,$user,$password);
mysqli_select_db($connexion, $nomBase);
$requete = "SELECT Nom, Prenom ".
           "FROM $nomTable ".
           "ORDER by Nom";
$result=mysqli_query($connexion,$requete);
?>
</body>
</html>
```

# Récupérer les résultats

► `mysqli_num_rows($result)` retourne le nombre de lignes dans l'identificateur de résultat

► `mysqli_fetch_array($result, [typederesultat])` : extrait la ligne sous forme d'un tableau associatif. Cette méthode représente un inconvénient si deux tables portent le même nom de champs lors d'une requête :

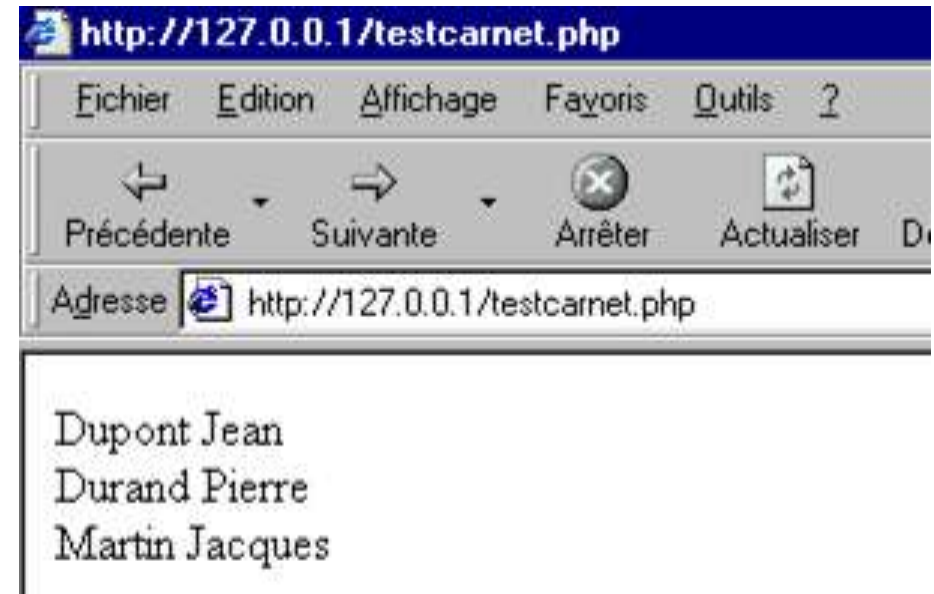
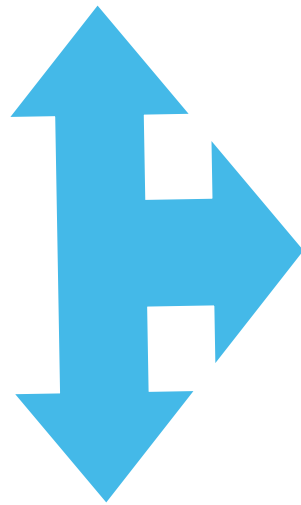
```
SELECT table1.nom, table2.nom FROM table1,table2...
```

► `mysqli_fetch_row($result)` : extrait la ligne sous forme d'un tableau énuméré

► `mysqli_fetch_assoc(...)`

# Exemple

```
$result=mysqli_query($connexion,$requete);  
while($row=mysqli_fetch_row($result)){  
echo($row[0] ." ". $row[1] . "<BR>");  
}
```



```
$result=mysqli_query($connexion,$requete);  
while($row=mysqli_fetch_array($result)){  
echo($row["Nom"] ." ". $row["Prenom"] . "<BR>");  
}
```



# Autres fonctions utiles

---

- ▶ `mysqli_free_result($result)` libère la mémoire associée à l'identificateur de résultat
- ▶ `mysqli_list_tables(nom,$connexion)` retourne la liste des tables correspondant à une base
- ▶ `mysqli_affected_rows($connexion)` retourne le nombre de lignes affectées par la dernière requête

# Remplir un formulaire à partir d'une table

Nom de la table:  Ajouter  colonne(s)

Structure

Nom	Type	Taille/Valeurs*	Défaut	Interclassement	Attributs	Null	Index	A_I	Commentaires
<input type="text" value="id_cat"/>	<input type="text" value="INT"/>	<input type="text"/>	<input type="text" value="Aucune"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="PRIMARY"/>	<input checked="" type="checkbox"/>	<input type="text"/>
<input type="text" value="nom_cat"/>	<input type="text" value="VARCHAR"/>	<input type="text" value="50"/>	<input type="text" value="Aucune"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>	<input type="text"/>







Nom de la table:  Ajouter  colonne(s)





Structure

Nom	Type	Taille/Valeurs*	Défaut	Interclassement	Attributs	Null	Index	A_I	Commentaires
<input type="text" value="id_prod"/>	<input type="text" value="INT"/>	<input type="text"/>	<input type="text" value="Aucune"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="PRIMARY"/>	<input checked="" type="checkbox"/>	<input type="text"/>
<input type="text" value="nom_prod"/>	<input type="text" value="VARCHAR"/>	<input type="text" value="50"/>	<input type="text" value="Aucune"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>	<input type="text"/>
<input type="text" value="description"/>	<input type="text" value="VARCHAR"/>	<input type="text" value="255"/>	<input type="text" value="Aucune"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>	<input type="text"/>
<input type="text" value="prix"/>	<input type="text" value="INT"/>	<input type="text"/>	<input type="text" value="Aucune"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>	<input type="text"/>
<input type="text" value="ext_cat"/>	<input type="text" value="INT"/>	<input type="text"/>	<input type="text" value="Aucune"/>	<input type="text"/>	<input type="text"/>	<input type="checkbox"/>	<input type="text" value="---"/>	<input type="checkbox"/>	<input type="text"/>













# Remplir un formulaire à partir d'une table





+ Options

				id_cat	nom_cat	
<input type="checkbox"/>		Modifier		Copier		Effacer
				1	Produits laitiers	
<input type="checkbox"/>		Modifier		Copier		Effacer
				2	Biscuits	

 ☐ Tout cocher    Pour la sélection :  Modifier  Effacer  Exporter

+ Options

				id_prod	nom_prod	description	prix	ext_cat
<input type="checkbox"/>		Modifier		Copier		Effacer		
				1	Lait	Pack de 6 bouteilles	8	1
<input type="checkbox"/>		Modifier		Copier		Effacer		
				2	Beurre	plaquette de 250g de beurre	2	1
<input type="checkbox"/>		Modifier		Copier		Effacer		
				3	Granola	Paquet de 18 biscuits	4	2
<input type="checkbox"/>		Modifier		Copier		Effacer		
				4	Crème fraîche	pot de 30 cl	3	1

 ☐ Tout cocher    Pour la sélection :  Modifier  Effacer  Exporter

# Le code

```
1 <html>
2 <body>
3 <?php
4 $connexion=mysqli_connect("localhost","root","");
5 mysqli_select_db($connexion,"ecommerce") or die ("erreur de connexion");
6
7 $requete = "SELECT * from categorie";
8 $result=mysqli_query($connexion,$requete) or die ("erreur de query");
9 ?>
10
11 <form action=<?php echo $_SERVER["PHP_SELF"] ?>>
12     <select name="categorie">
13     <?php
14     while($row=mysqli_fetch_array($result)){
15         echo("<option value=\"".$row["id_cat"]."\">". $row["nom_cat"]."</option>");
16     }
17     ?>
18 </select>
19 </body>
20 </html>
21
```

# Le résultat

---



# PDO : PHP Data Object

# Quelques rappels : Objets et Classes

- ▶ Une **Classe** est un « moule » permettant la création d'**objets**. La classe est un *prototype* qui définit un type d'objet.
- ▶ Les **objets** sont appelés des **instances** de la classe
- ▶ Les objets possèdent des **propriétés** (des variables décrivant les caractéristiques de l'objet) et des **méthodes** permettant de manipuler les propriétés
- ▶ Notation
  - `$objet->methode()`
  - `$objet->ppte`
- ▶ Instancier une classe : `$objet=new classe()`

# L'API PDO

---

- ▶ PDO signifie **P**hp **D**ata **O**bject.

Il s'agit une couche d'abstraction des fonctions d'accès aux bases de données.

- ▶ Les fonctions d'accès sont universelles et indépendantes de la base

```
$connexion = new PDO('mysql:host=localhost;dbname=test', $user, $passwd);
```



# PDO et PDOStatement

- ▶ `query()` : méthode de PDO qui retourne un *jeu de résultats* sous la forme d'un objet *PDOStatement*,
- ▶ `exec()` : méthode PDO qui retourne uniquement le nombre de lignes affectées
- ▶ `execute()` : méthode de PDOStatement permet d'exécuter des requêtes préparées.
- ▶ On utilise `query()` pour des requêtes de sélection (SELECT) et `exec()` pour des requêtes d'insertion (INSERT), de modification (UPDATE) ou de suppression (DELETE).

```
$query = 'SELECT * FROM client WHERE id_client =1;';  
$result = $pdo->query($query)->fetch();
```

```
$query = 'DELETE FROM client WHERE id_client=1;';  
$rowCount = $pdo->exec($query);
```

# Requêtes préparées

---

Requêtes préparées =

- ▶ rigueur de programmation
- ▶ optimisation du temps d'exécution requis pour les requêtes exécutées plus d'une fois
- ▶ plus grande sécurité au niveau des requêtes en prévenant les injections
- ▶ `prepare($query)` est une méthode PDO qui retourne un `PDOStatement`

# Exemple de requête préparée

```
$query = 'SELECT * FROM client' . ' WHERE cat=?' . ' AND ville=?';

$prep = $pdo->prepare($query); // Associer des valeurs aux placeholders
$prep->bindValue(1, 7, PDO::PARAM_INT);
$prep->bindValue(2, 'paris', PDO::PARAM_STR);

// Compiler et exécuter la requête
$prep->execute();
```

Les Place holders peuvent également être nommés :

```
$query = 'SELECT * FROM client' . ' WHERE cat=:cat' . ' AND
ville=:ville';

$prep = $pdo->prepare($query);
$prep->bindValue(:cat, 7, PDO::PARAM_INT);
$prep->bindValue(:ville, 'paris', PDO::PARAM_STR);
```

# Exemple mysql / PDO

```
$connexion=mysqli_connect($host,$user,$passwd);
mysqli_select_db('client',$connexion);
$requete = "SELECT Nom, Prenom FROM $nomTable";
$result=mysqli_query($requete,$connexion);
while($row=mysqli_fetch_row($result)){
echo($row[0] ." ". $row[1] . "<BR>");
}
```

```
$pdo = new PDO('mysql:host=localhost;dbname=client',$user,$passwd);
$requete = "SELECT Nom, Prenom FROM $nomTable ORDER by Nom";
$prep = $pdo->prepare($requete);
$result = $prep->execute();
while($row = $result->fetch(PDO::FETCH_NUM)){
echo($row[0] ." ". $row[1] . "<BR>" );
}
```

Autres Fetch modes :

PDO::FETCH\_ASSOC

PDO::FETCH\_BOTH

# Gestion de session

# Définition

---

- ▶ A chaque connexion correspond une session utilisateur. Gérer la session permet de :
  - conserver la valeur des variables qui peuvent être utiles pendant la session (applications de panier électronique, accès à certaines parties du site en fonction de droits utilisateur, gestion de compte client...)
  - Cette solution est la plus sûre et la plus légère
- ▶ Alternatives :
  - transmission des paramètres d'url en url
  - cookies

# Fonctionnement

---

- ▶ Lors de l'ouverture d'une session, un **id de session** est attribué au client
- ▶ Les variables de session sont enregistrées sur le serveur dans le répertoire `session.save_path`
- ▶ Les paramètres généraux des sessions se trouvent dans **php.ini**

# Configuration de php

http://localhost/index.php?to=phpinfo

PCRE Library Version

3.4 22-Aug-2000

## session

Session Support

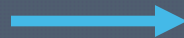
enabled

Directive	Local Value	Master Value
session.auto_start	Off	Off
session.cache_expire	180	180
session.cache_limiter	nocache	nocache
session.cookie_domain	no value	no value
session.cookie_lifetime	0	0
session.cookie_path	/	/
session.cookie_secure	Off	Off
session.entropy_file	no value	no value
session.entropy_length	0	0
session.gc_maxlifetime	1440	1440
session.gc_probability	1	1
session.name	PHPSESSID	PHPSESSID
session.referer_check	no value	no value
session.save_handler	files	files
session.save_path	C:\Program Files\EasyPHP\tmp\	C:\Program Files\EasyPHP\tmp\
session.serialize_handler	php	php
session.use_cookies	On	On

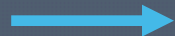
Nb de minutes  
avant expiration



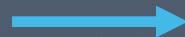
Nb de secondes  
avant suppression  
du fichier de  
session



Nom de l'id de  
session  
(remplacer)



Utilisation de  
cookies par la  
session (passer à  
Off)





# Fonctions de session

---

## ► Démarrage d'une session

```
session_start()
```

## ► Gestion des variables dans les paramètres de session

```
$_SESSION["mavARIABLE"]=$toto;  
$_SESSION["mavARIABLE"]=$_GET["toto"];  
unset($_SESSION["mavARIABLE"]);
```

## ► Fin de session

```
session_destroy()
```

# Propagation de l'id de session

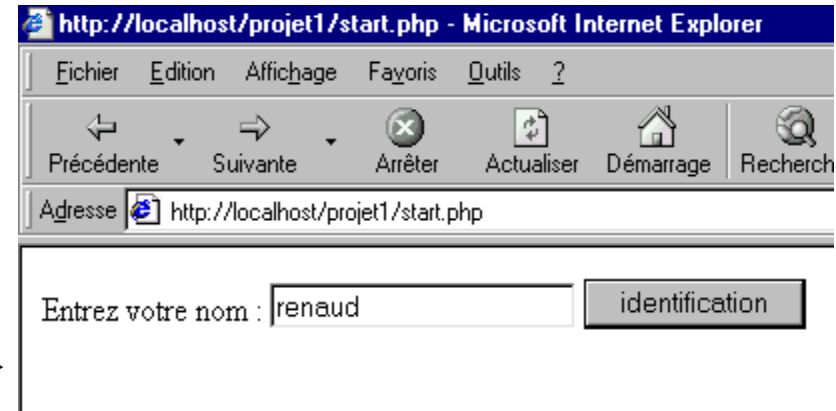
---

- ▶ La clé de session est stockée dans la variable globale `$PHPSESSID`
- ▶ Lorsque l'on initialise une session on utilise `session_start()`; qui renvoie une clé de session. Dans les autres pages du site on utilise `session_start()` pour rappeler la session correspondant au bon utilisateur.
- ▶ Ainsi, toutes les variables "enregistrées" sont accessibles durant la session.

# Exemple

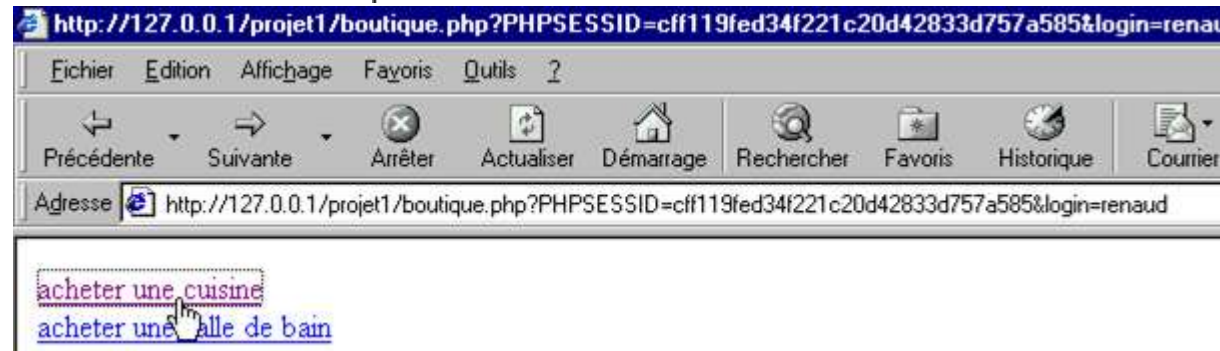
1

```
<?php
session_start();
?>
<form action="boutique.php">
Entrez votre nom :
<input type="text" name="login">
<input type="submit" value="identification">
</form>
```



2

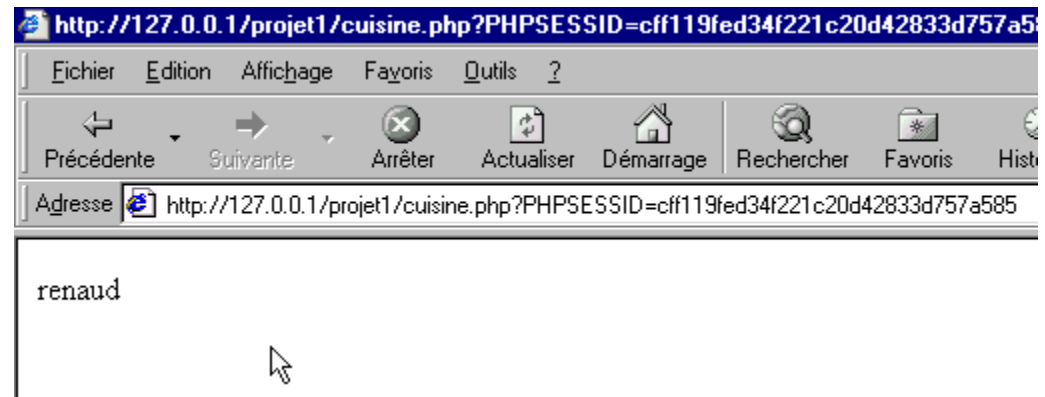
```
<?php
session_start();
$_SESSION["login"]=$_GET["login"];
?>
<a href="cuisine.php">acheter une cuisine</a> <BR>
<a href="sdb.php">acheter une salle de bain</a> <BR>
```

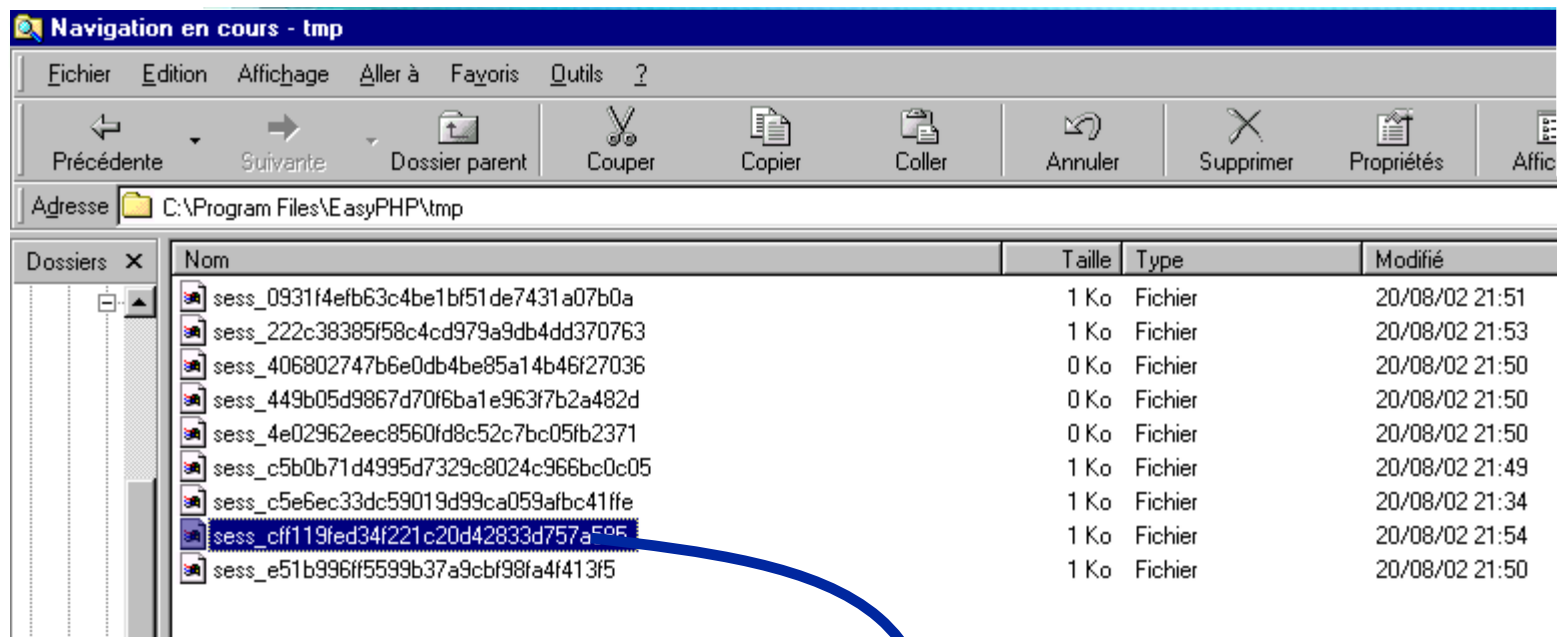


## Exemple (suite)

3

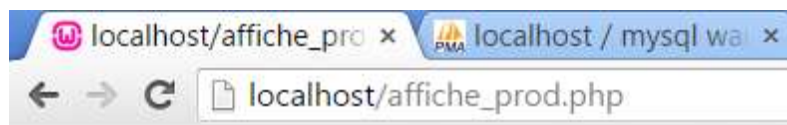
```
<?php  
session_start();  
echo $_SESSION["login"];  
?>
```





login|s:6:"renaud";

# Suite de l'exemple ...

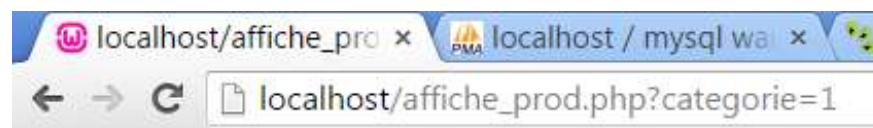


**Sélectionnez votre catégorie de produ**

Produits laitiers ▼ OK

Produits laitiers

Biscuits



**Sélectionnez votre catégorie de produits**

Produits laitiers ▼ OK

Lait	<a href="#">addToCard</a>
Beurre	<a href="#">addToCard</a>
Crème fraîche	<a href="#">addToCard</a>

# Le code

```
1
2 <?php
3 session_start();
4 include ("Connect.php");
5 ?>
6
7
8 <!doctype html>
9 <html lang="fr">
10 <head>
11     <meta charset="utf-8">
12 </head>
13
14 <?php
15 $requete = "SELECT * from categorie";
16 $result=mysqli_query($connexion,$requete) or die ("erreur de query");
17 ?>
18 <H2>Sélectionnez votre catégorie de produits</H2>
19 <form action=<?php echo $_SERVER["PHP_SELF"] ?>>
20     <select name="categorie">
21     <?php
22     while($row=mysqli_fetch_array($result)){
23         echo("<option value=\"". $row["id_cat"] . "\">". $row["nom_cat"] . "</option>");
24     }
25     ?>
26 </select>
27 <input type="submit" value="OK">
28 </form>
29 <BR>
30 |
31 <table BORDER=1>
32 <?php
33 if (isset($_GET["categorie"])){
34
35     $req_affiche="SELECT * from categorie, produit WHERE produit.ext_cat=categorie.id_cat AND categorie.id_cat=".$_GET["categorie"];
36     $result=mysqli_query($connexion,$req_affiche) or die ("erreur de query");
37     while($row=mysqli_fetch_array($result)){
38         echo("<tr><td>". $row["nom_prod"] . "</td><td>". "<a href=\"add_basket.php?produit=\". $row["id_prod"] . "\">addToCard</a></td></tr>");
39     }
40 }
```

# Traitement des chaînes de caractères



# Fonctions utiles

---

- ▶ Comparer deux chaînes de caractères

```
int strcmp($chaine1,$chaine2)
```

Renvoie 0 si les deux chaînes sont égales, <0 si chaîne1 est plus petit (ordre lexicographique) et >0 si \$chaîne1 est plus grand.

- ▶ Découper une chaîne de caractères

```
array explode (char,$chaine)
```

Découpe une chaîne en fonction d'un caractère char et place chaque partie découpée dans les cases d'un tableau

**Ex:** \$tab=explode('/', \$url)

# Expressions régulières (1)

- ▶ **"^debut"**: chaîne qui commence par *"debut"*  
**"fin\$"**: chaîne qui se termine par *"fin"*  
**"^chaîne\$"**: chaîne qui commence et se termine par *"chaîne"*  
**"abc"**: chaîne contenant la chaîne *"abc"*
- ▶ **\*** : *"zero ou plusieurs"*,  
**+** : *"un ou plusieurs"*,  
**?** : *"un ou aucun"*
- ▶ **"abc+"**: chaîne qui contient *"ab"* suivie de un ou plusieurs *"c"*  
(*"abc"*, *"abcc"* etc..)  
**"abc\*"**: chaîne qui contient *"ab"* suivie de zero ou plusieurs *"c"*  
(*"ab"*, *"abc"* etc..)  
**"abc?"**: chaîne qui contient *"ab"* suivie de zero ou un *"c"* (*"ab"* ou *"abc"*)  
**"^abc+"**: chaîne qui commence par *"ab"* suivie de un ou plusieurs *"c"*  
(*"abc"*, *"abcc"* etc..)

## Expressions régulières (2)

- ▶ Les parenthèses ( ) permettent de représenter une séquence de caractères :

"**a(bc)\***": chaîne qui contient "*a*" suivie de zero "*bc*" ou plus

- ▶ Le point . indique n'importe quel caractère (une fois) :

"**^. {3}\$**": chaîne qui contient 3 caractères

# Expressions régulières (3)

► `int ereg ( $pattern, $string [, $regs])`

Recherche dans la chaîne *string* les séquences de caractères qui correspondent au masque *pattern*.

Si au moins une séquence est trouvée, et que la fonction est appelée avec un troisième argument *regs*, les résultats seront enregistrés dans *regs*.

`$regs[1]` contiendra la première occurrence, `$regs[2]` contiendra la deuxième occurrence, et ainsi de suite. `$regs[0]` contient une copie de la chaîne.

► `ereg()` retourne **TRUE** si une occurrence a été trouvée dans la chaîne et **FALSE** dans le cas contraire, ou si une erreur est survenue.

```
$fp = fopen("http://www.toto.fr", "r");  
while (!feof($fp)) {  
    $page .= fgets($fp, 4096);  
    $titre = eregi("<title>(.*?)</title>", $page, $regs);
```

# Expressions régulières (4)

- ▶ `eregi()` : Recherche par expression régulière insensible à la casse.
- ▶ `ereg_replace($expr1, $expr2, $string)` : substitution d'une expression `expr1` par une autre `expr2` dans une chaîne de caractères

```
<?
$string = "One      Two      Three    Four";
$var = eregi_replace(" +", " ", $string);
?>
```

mail et imap

# mail

---

- ▶ la fonction mail permet d'envoyer un mail en accédant à la messagerie locale

*mail (destinataire,objet,message) ;*

Exemple :

```
<?php  
mail ("toto@univ-mlv.fr,tutu@univ-mlv.fr","bonjour",  
"bonjour \r ceci est un message");  
?>
```

# Message contenu dans un fichier

---

```
<?php
if (!$fic=fopen("testfic.txt","r")){
echo("echec de l'ouverture du fichier");
} else {
$taille=filesize("testfic.txt");
$message=fread($fic,$taille);
fclose($fic);
mail ("toto@univ-mlv.fr","bonjour",$message);

}
?>
```



# Fichiers attachés et classe MIME

---

- ▶ MIME = Multipurpose Internet Mail Extensions
- ▶ PHP gère les types de fichiers à l'aide de la classe `mime_mail`

```
$mail = new mime_mail;
```

- ▶ Les principales propriétés sont :

```
$mail->de, $mail->à, $mail->objet, $mail->body
```

- ▶ Les principales méthodes sont :

```
$mail->envoyer()
```

```
$mail->ajoute_piece($donnees, $nomfichier, $type)
```

# Exemple de mail avec pièce jointe

```
<?php
include "mime_mail.inc";

$nouveau_mail=new mime_mail;
$mail->de = "eppstein@univ-mlv.fr";
$mail->à = "toto.univ-mlv.fr";
$mail->objet = "image de voiture";
$mail->body = "Ci-joint une image de voiture";

$nom_fichier="voiture.jpg";
$fic=fopen($nom_fichier,"r");
$fic_type="image/jpeg";
$message=fread($fic,filesize($nom_fichier));
fclose($fic);

$mail->ajoute_piece($message,$nom_fichier,$fic_type);

$mail->envoyer();

?>
```

# Serveur de mail distant

---

## ► Envoyer un mail :

php utilise la classe smtp\_mail

```
include "smtp_mail.inc";
```

```
$smtp->envoyer($serveur_smtp, $de, $a, $message);
```

## ► Lire un mail :

php dispose de fonctions IMAP...