



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Informatique Modélisation UML

Objectifs de la séance :

**Création d'un site Web dynamique
PHP**



PHP

- PHP est un **langage de programmation** utilisé pour la construction de **sites Web dynamiques**
 - Pages PHP : pages Web qui contiennent de PHP
 - On va **mélanger le PHP au code HTML / CSS**
 - Le code **PHP** va être **analysé par le serveur**
 - Le **résultat** va être une **nouvelle page Web** mise à jour automatiquement par le code PHP

Le code PHP est à l'intérieur de la balise

<?php ... ?>

ou entouré par la balise

**<script language="php">
... </script>**

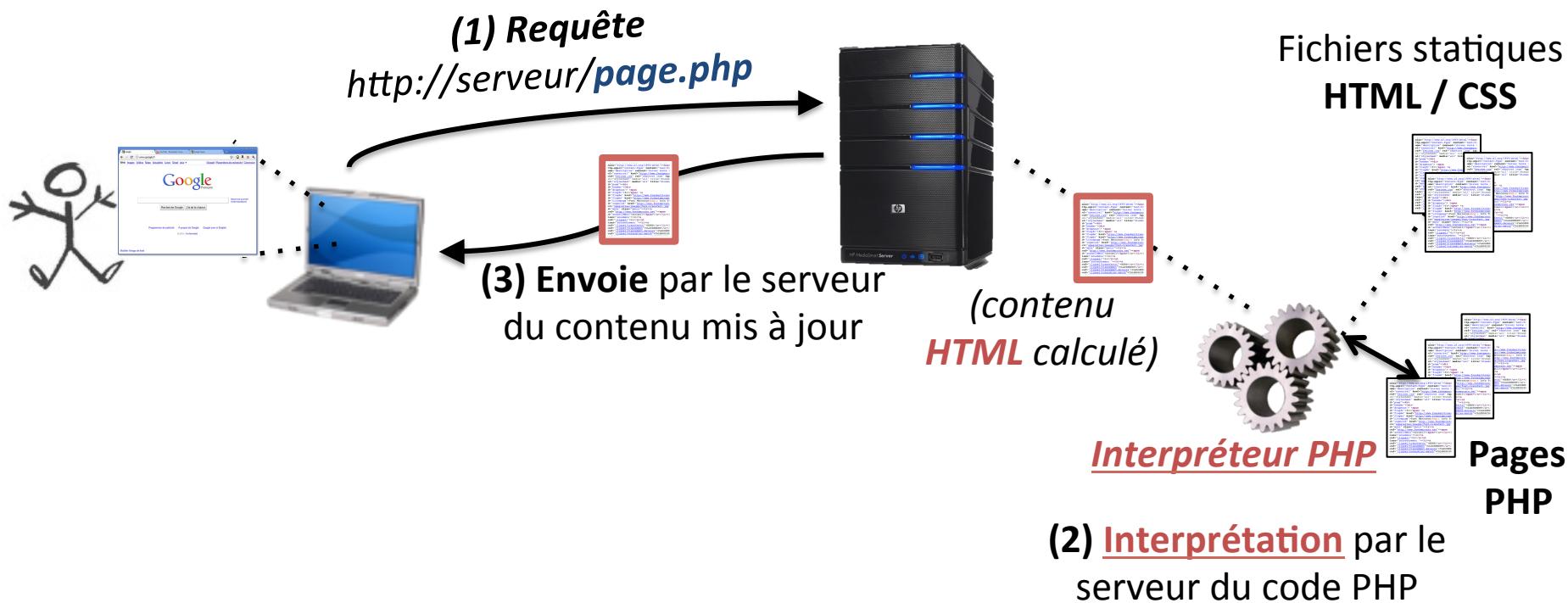
coursPHP-1.php

```
<html> ...
<?php
    date_default_timezone_set("Europe/Paris");
    echo "<p style='font-style: italic;'> Paris, le "
        .date('d / m / Y')."</p>" ;
?
...
</html>
```

PHP

- **Cycle de vie d'une page PHP**

- 1) Le **client** envoie la requête au serveur
- 2) Les pages PHP sont analysées par le serveur, le code PHP est interprété
- 3) Le **contenu** de la page est **mise à jour automatiquement** et envoyé au client



Bienvenue sur le site PHP

Paris, le 28 / 02 / 2012

Il est 19:16:28 .



<http://serveur/courPHP-1.php>

PHP

...

```
<h1> Bienvenue sur le site PHP </h1>
<p style='font-style: italic;'> Paris, le
28 / 02 / 2012</p>
<p> Il est 18:45:12 .</p>
```

...

Contenu **HTML/CSS**
calculé



Interpréteur PHP

Page **PHP originelle**
(PHP & HTML / CSS)

...

```
<h1> Bienvenue sur le site PHP </h1>
<?php
date_default_timezone_set("Europe/Paris");
echo "<p style='font-style: italic;'> Paris, le "
.date('d / m / Y')."</p>" ;
?>
<script language="php">
echo "<p> Il est ".date("H:i:s")." .</p>" ;
</script>
```

...

PHP

- Exemple code PHP

<!DOCTYPE html>

<?php

Portion de code PHP
<?php ?>

date_default_timezone_set("Europe/Paris") ;

\$today = date("d-m-Y H:i:s") ;

\$variable = "PHP5" ;

?>

<html> <head> ... </head>

<body>

<h1> Exemple PHP </h1>

<p>Contenu statique : ça ne change pas </p>

<?php

/* ce contenu va être interprété par le serveur */

echo "<p>Contenu en \$variable </p>" ;

echo "<p>Aujourd'hui c'est le \$today </p>" ;

?>

Portion de code PHP

</body> </html>

coursPHP-2.php

Toute instruction PHP se termine par un « ; »

La séquence /* ... */ délimite un **commentaire**, visible pour l'auteur, invisible pour le client

L'instruction « echo » permet d'écrire dans le document final

- Code une fois interprété par le serveur...

```
<!DOCTYPE html>
<html>
<head> ... </head>
<body>
<h1> Exemple PHP </h1>
<p>Contenu statique : ça ne change pas </p>
<p>Contenu en PHP5 </p> <p> Aujourd'hui
c'est le 28-02-2012 19:38:54 </p>
</body>
</html>
```

Résultat des
instructions « echo »





- **La notion de variable**

- Une variable est **un conteneur de valeur**
- On peut lui affecter une valeur, qu'on va utiliser plus tard

\$variable = "PHP5" ;

Le « \$ » indique une variable

Le nom de variable commence toujours par une **lettre** ou un « _ », **sans espace**

Le « = » est une **affectation**
On attribut une valeur à la variable

echo "... **\$variable** ..." ;

On récupère la valeur gardée dans la variable par son nom

PHP

- Exemple :

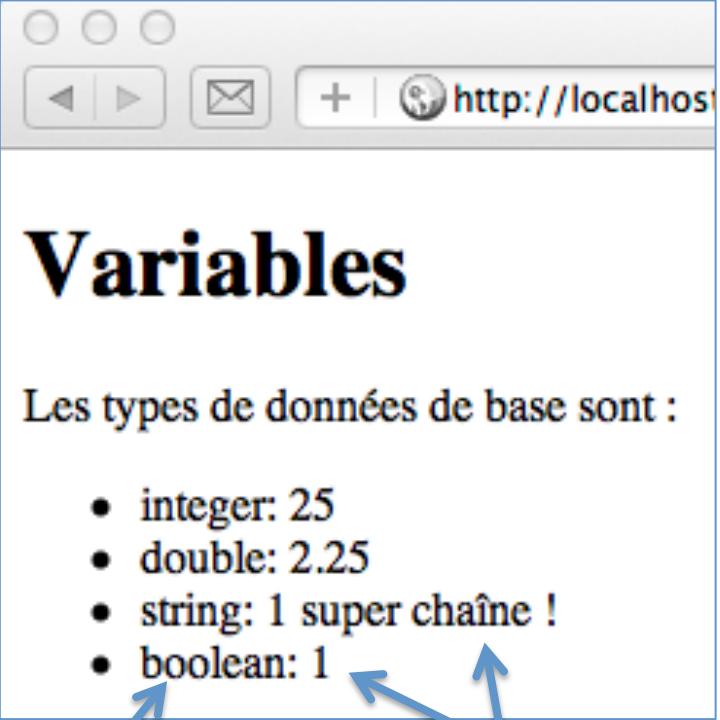
```
<?php  
$entier = 25;  
$decimal = 2.25;  
$chaine = "1 super chaîne !";  
$boolean = true;
```

Définition d'une variable

```
echo "<li>" . gettype($entier) . ": $entier </li>";  
echo "<li>" . gettype($decimal) . ": $decimal </li>";  
echo "<li>" . gettype($chaine) . ": $chaine </li>";  
echo "<li>" . gettype($boolean) . ": $boolean </li>";
```

```
?>
```

On récupère la valeur de la variable \$boolean



gettype()
informe le type
de la variable

Valeur de chaque variable

- ## Opérateurs

- Différents opérateurs permettent de manipuler des valeurs, qu'ils soient dans les variables ou pas

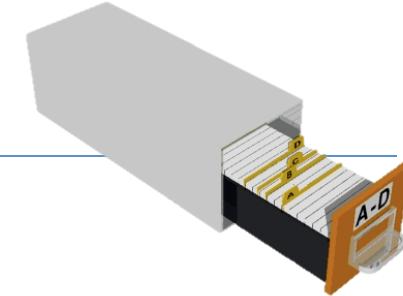
Opérateurs mathématiques	Opérateurs String	Opérateurs de comparaison	Opérateurs logiques
+ - * / %	. <i>(concaténation)</i>	== != <= < >= >	(OR) && (AND) ! (not)

```
<?php
    $a = 2 + 3 ;
    $b = 4 - $a ;
    $nom = "Toto";
    echo "Salut " . $nom;
    echo "<p> 4 - $a vaut $b </p>";
?>
```

Exemple avec les opérateurs :

Salut Toto

4 - 5 vaut -1



- **Tableaux**
 - Variables capables d'enregistrer plusieurs valeurs d'un type
- **Tableaux à indice :**
 - Chaque position est identifiée par un numéro (commençant par **0**)
 - `$tableau [0] = "A";`
 - `$tableau [1] = "B";`
 - `$tableau [3] = "Fin";`
 - `$tableau [] = "Suite";`
- **Tableaux associatifs :**
 - Chaque position reçoit un identifiant (un label)
 - `$tableauAssoc ["Prenom"] = "Jean";`
 - `$tableauAssoc ["Nom"] = "Dupont" ;`

Attention a définir toutes les positions avant de les utiliser ou il y aura une **message d'erreur**.



Jean	Dupont
Prenom	Nom

```

<head> ...
    <style>... </style>
</head>
<body> ...
<h2>Tableaux à indice </h2>
<table>
<?php
    $tableau [0] = "A";
    $tableau [1] = "B";
    $tableau [3] = "Fin";
    $tableau [] = "Suite";

    echo "<tr> <td>". $tableau[0] . "</td> <td>". $tableau[1]
        . "</td> <td>". $tableau[2] . "</td><td>". $tableau[3]
        . " </td><td>". $tableau[4] . "</td></tr>";

?
</table>
...

```

Tableaux en PHP

localhost:8888/exemplesPHP/coursPHP-6.php Lecteur >

Tableaux

Tableaux à indice

Notice: Undefined offset: 2 in /Users/kirsch/Documents/Sites/exemplesPHP/coursPHP-6.php on line 29

Alice	B		Fin	Suite
-------	---	--	-----	-------

Message d'erreur car le contenu de la position 2 (\$tableau[2]) n'a pas été défini auparavant.

Contenu de la position 4 (\$tableau[4])

67

...

```
<h2>Tableau associatif </h2>
<table>
  <tr> <th> Nom </th> <th> Prénom </th> </tr>

<?php
  $tableauAssoc ["Prénom"] = "Jean";
  $tableauAssoc ["Nom"] = "Dupont" ;

  echo "<tr> <td>" . $tableauAssoc ["Nom"] . "</td>" ;
  echo "<td>" . $tableauAssoc ["Prénom"] . " </td></tr>" ;
?

</table>
</body>
```

Tableau associatif

Nom	Prénom
Dupont	Jean



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

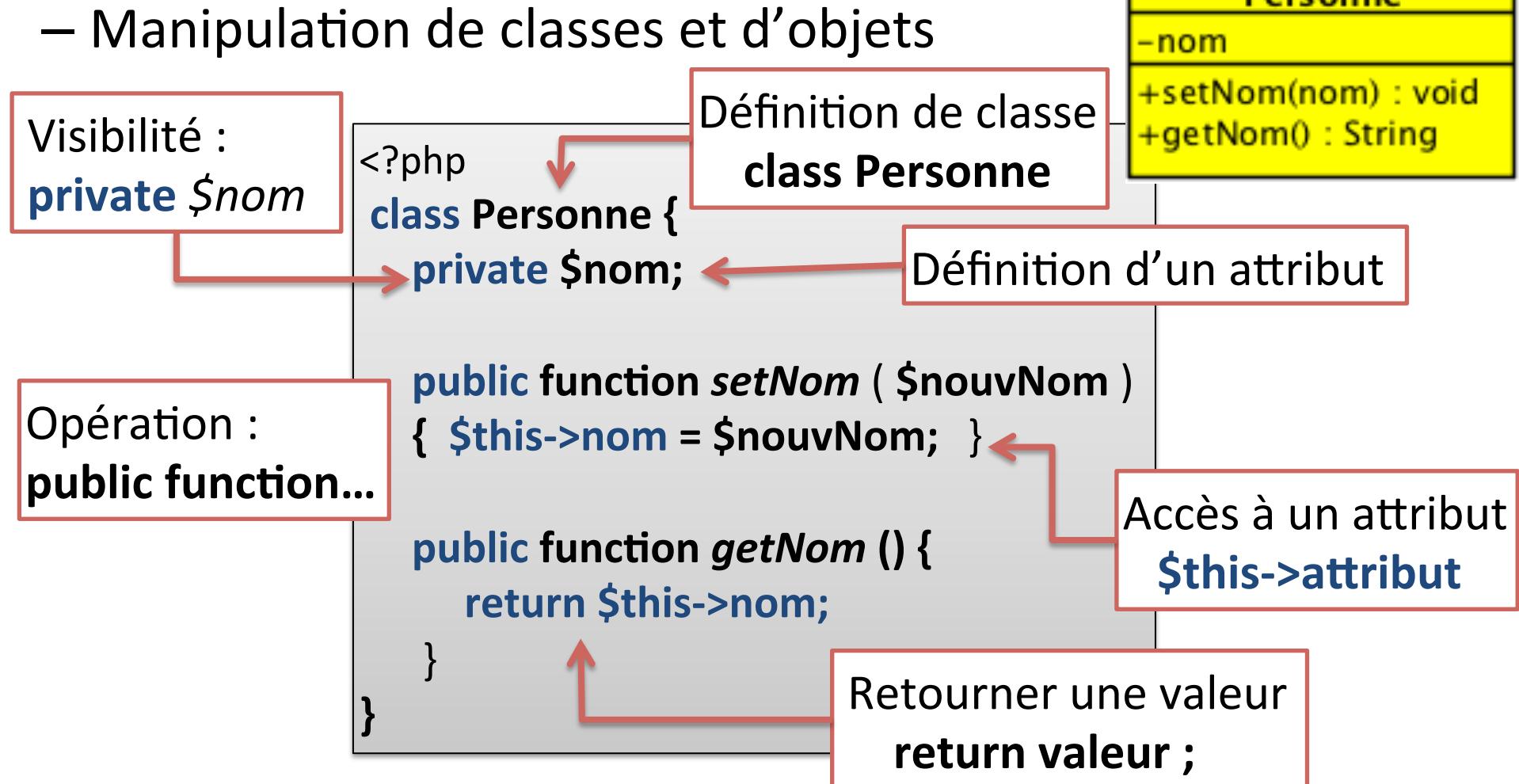
ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Informatique Modélisation UML

Objectifs de la séance :
Classes & Objets

PHP orienté objets

- PHP 5 est un langage « orienté objets »
 - Manipulation de classes et d'objets



PHP orienté objets

- **Classes & Objets**

- Création d'un objet : **\$obj = new classe();**

```
...  
<?php  
$toto = new Personne();  
  
$toto->setNom("Toto");  
  
echo "<p> ... " . $toto->getNom() . "</p>";  
  
$toto->nom = "blablabla";  
echo " <p> " . $toto->nom . " </p> ";  
?>
```

Création d'un objet
\$toto = new Personne ()

Accès aux opérations publiques
\$toto->setNom("Toto")
\$toto -> getNom ()

Impossible d'accéder aux attributs privés



PHP orienté objets

```
<html> <head> ...
<?php
    class Personne {
        private $nom;
        ...
    } //fin classe Personne
?>
</head>
<body> ...
<?php
    $toto = new Personne();
    $toto->setNom("Toto");
    echo "<p> Objet <i>Personne</i> : " . $toto->getNom() . "</p> ";
    ...
    echo " <p> " . $toto->nom . " </p> ";
?>
</body> </html>
```

Classes et Objets en PHP

localhost:8888/exemplesPHP/coursPH Lecteur

Classes et objets en PHP

Objet *Personne* : Toto

Teste de l'accès (ça ne doit pas marcher) :

Fatal error: Cannot access private property Personne::\$nom in /Users/kirsch/Documents/Sites/exemplesPHP/coursPHP-2013-1.php on line 35

Erreur car l'attribut « nom » est privé !!

PHP orienté objets

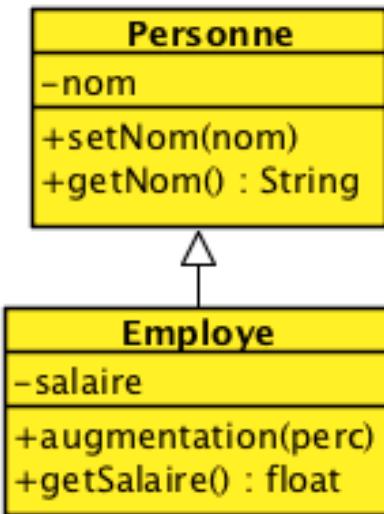
- **Classes & objets :**
 - Héritage : class *SousClasse* extends *SuperClasse*

```

class Employe extends Personne {
  private $salaire = 1000;

  public function augmentation ($perc) {
    if ($perc > 0) {
      $this->salaire = $this->salaire +
      $this->salaire*$perc;
    }
    }
    public function getSalaire()
    { return $this->salaire; }
}
  
```

Fichier Employe.php



La classe ***Employe*** hérite de la classe ***Personne***

class Employe extends Personne

On rajoute un nouvel attribut
private \$salaire
 Et des nouvelles opérations
public function augmentation
public function getSalaire

PHP orienté objets

- **Classes & objets**

```
<?php  
include "Employe.php" ;
```

On importe la définition des classes
Employe et **Personne**

```
$toto = new Employe();
```

Toto est un **Employé**, il est donc
une **Personne**

```
$toto->setNom("Toto");  
$toto->augmentation(0.10);
```

Toto possède un **salaire (Employe)**,
mais aussi un **nom (Personne)**

```
echo "<i> nom </i> : ". $toto->getNom() ;  
echo " <i> salaire </i> : " . $toto->getSalaire() . " € </p>";  
?>
```

La classe **Employe** hérite tous les
attributs et **opérations** de **Personne**

PHP orienté objets

• Classes & objets

```
<html> <head> . . . </head>
<body> <h1>Objets en PHP</h1>
<?php
    include "Employe.php" ;

    $toto = new Employe();

    $toto->setNom("Toto");
    $toto->augmentation(0.10);

    echo "<p>Objet Employe : </p> <ul>" ;
    echo "<li> <i> nom </i> : ". $toto->getNom() . "</li>";
    echo "<li> <i> salaire </i> : " . $toto->getSalaire() . " € </li>";
    echo "</ul>" ;
?>
</body> </html>
```

Objets en PHP

Objet Employe :

- *nom* : Toto
- *salaire* : 1100 €

PHP orienté objets

- Classes & objets
 - Méthode **constructeur** : **__construct**
 - Redéfinition d'une opération

```
class Manager extends Employe {  
    private $bonus ;  
    function __construct ($bon)  
    { $this->bonus = $bon; }  
  
    public function getSalaire() {  
        return parent::getSalaire() + $this->bonus;  
    }  
    public function setBonus ($nouvBon) { ... }  
    public function getBonus () { ... }  
}
```

Le **constructeur** est appelé chaque fois qu'un objet est créé (**new**)

Redéfinition de l'opération **getSalaire**
parent::getSalaire correspond à l'opération **getSalaire** définie par la super-classe (**Employe**)

PHP orienté objets

- Classes & objets

```
<?php
require "Manager.php" ;
$toto = new Manager(400);

$toto->setNom("Toto");
$toto->augmentation(0.10);

echo "<p><i>Manager</i> : ". $toto->getNom()
    . ", salaire ". $toto->getSalaire() . " € "
    . ", bonus " . $toto->getBonus() . "</p>";

?>
```

Appel au **constructeur** :
function __construct (\$bon)
{ \$this->bonus = \$bon; }

Appel à l'opération **getSalaire**
de la classe **Manager**



Objets en PHP

Manager : Toto, salaire 1500 € , bonus 400



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Informatique Modélisation UML

Objectifs de la séance :
Formulaires HTML & PHP

PHP est un langage pour le Web

- **Communication entre le client (navigateur) et le serveur (php)**
 - Les **formulaires en HTML** permettent de recueillir des données auprès de l'utilisateur
 - Les données sont ensuite **communiquées à un programme**
 - Le navigateur **envoie les données** récoltées par les formulaires au serveur
 - Le programme (page PHP) récupère les données grâce à des **variables**



Formulaires HTML

- Un **formulaire HTML** est défini par la balise
<form ...> ... </form>
 - Tous les éléments sont à l'intérieur de la balise
**<form name="*nomFormulaire*"
action="*page.php*" method="**get | post**" > </form>**
- Les **champs du formulaire** sont introduits par différents balises :
 - **<input type="..." name="..." value="..." id="..." />**
 - **<textarea name="..." id="..." cols="..." rows="..." > ... </textarea>**
 - **<select name="..." id="..." size="..." >
<option value="..." > ... </option> </select>**

action : à qui on envoie les données

method: comment on envoie les données



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Formulaires HTML

The screenshot shows a web browser window with the URL <http://localhost:8888/exem>. The form contains fields for 'Nom' (Name), 'Produit' (Product), and 'Opinion' (Opinion). The 'Nom' field has the value 'votre nom'. The 'Produit' dropdown menu is set to 'Super Kdo'. The 'Opinion' area contains the text 'Votre opinion sur nos produits'. Below the form are two buttons: 'Envoyer' (Send) and 'Nettoyer' (Clean).

Form Elements and Associated HTML Code:

- Nom:** (Associated code block)
- Produit:**
Super Kdo (Associated code block)
- Opinion:** (Associated code block)
- Buttons:**
 - (Associated code block)
 - (Associated code block)

HTML Code Examples:

- <input type="text" name="nomClient" value="votre nom" size="40" maxlength="150" />
- <select name="produit">
 <option value="SuperKdo">
 Super Kdo </option>
 ...
</select>
- <textarea name="opinionClient" cols="40" rows="5" >
 Votre opinion sur nos produits
</textarea>
- <input type="submit" value="Envoyer" />
- <input type="reset" value="Nettoyer" />

Text Box:

input type="submit"
se charge d'envoyer les données du formulaire

Formulaires HTML

```
<form name="formClient" action="coursPHP-7.php" method="POST" >
```

```
    <label for="nom">Nom</label>
```

```
    <input type="text" id="nom" name="nomClient"
```

```
        value="votre nom" size="40" maxlength="150" /> <br/>
```

```
    <label>Produit</label>
```

```
    <select name="produit"
```

```
        <option value="SuperKdo"
```

```
        <option value="MegaTruc">Mega Truc</option>
```

```
        <option value="BabyFun">Baby Fun</option>
```

```
    </select> <br/>
```

```
    <label>Opinion</label>
```

```
    <textarea name="opinionClient" cols="40" rows="5" >
```

Votre opinion sur nos produits </textarea>


```
    <input type="submit" value="Envoyer" class="bouton" />
```

```
    <input type="reset" value="Nettoyer" class="bouton" />
```

```
</form>
```

À qui les données sont envoyées

input type="text"
Zone de saisie

select ... option
Liste de sélection d'options

textarea
Zone de texte

input type="submit"
Input type="reset"
Boutons d'envoi et de reset du formulaire

Formulaires HTML & PHP

- Les données recueillies dans le formulaire sont transmises au programme indiqué dans **action=...**
- Dans **PHP**, on récupère ces données grâce à deux **tableaux associatifs** spéciaux
 - **\$_GET** → *<form action="..." method="get">*
 - **\$_GET["nom"]** *<input ... name="nom" />*
 - **\$_POST** → *<form action="..." method="post">*
 - **\$_POST["nom"]** *<input ... name="nom" />*



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Formulaires HTML & PHP

Formulaire

localhost:8888/exemplesPHP/formClient.html

Nom	<input type="text" value="Manuele"/>
Produit	<input type="text" value="Baby Fun"/> <input type="button" value="bla bla bla bla bla"/>
Opinion	<input type="button" value="Envoyer"/> <input type="button" value="Nettoyer"/>

```
<form name="formClient" action="coursPHP-7.php"
      method="POST">
<label for="nom">Nom</label>
<input type="text" id="nom"
       name="nomClient" value="votre nom"
       size="40" maxlength="150" /> <br/>
```

Formulaire Client

localhost:8888/exemplesPHP/formClient.php

Données enquête

Merci de votre participation, Manuele !

Votre produit est : BabyFun

Votre opinion est : bla bla bla bla bla

```
...
<?php
$nom = $_POST["nomClient"];
$op = $_POST["opinionClient"];
$prod = $_POST["produit"];
echo "<p>Merci de votre participation, $nom ! </p>";
echo "<p>Votre produit est : <i> $prod </i> </p>";
echo "<p> Votre opinion est : <i> $op </i> </p>";
?>
```

Formulaires HTML & PHP

• Méthode GET

- Les données sont envoyées dans l'**URL** du programme
- Limitée à 256 octets
- **Déconseillé**



```
<form name="formGet"
      action="coursPHP-8.php"
      method="GET">

<label>Nom</label>
<input type="text" name="client" size="20" /> <br/>
<label>Mot de passe </label>
<input type="password" name="mdp" size="10"/>
<br/>
<input type="submit" value="Envoyer" />
</form>
```

Données reçues :

Bienvenue, *toto* !

Formulaire Get

```
<?php
echo "<p>Bienvenue, <i>" . $_GET["client"] . "</i> ! </p>";
?>
```

Formulaires HTML & PHP

• Exemple

The screenshot shows a web form with the following fields:

- Vos données**:
 - Nom :** Manuele
 - Email :** mkirschpin@univ-paris1.fr
 - Sexe :** Femme
- Vos produits**:
 - Produit préféré :** Mega Truc
 - Suggestions :** bla bla bla bla
- Envoyer** | **Nettoyer**

```
<form name="..." action="coursPHP-9.php" method="POST">
<fieldset>
<legend>Vos données </legend>
<label>...</label> <input type="text" name="nom" ... /><br/>
<label>...</label> <input type="email" name="email" ... /><br/>
<input type="radio" name="sexe" value="Homme" />Homme
<input type="radio" name="sexe" value="Femme" /> Femme<br/>
</fieldset>
```

```
<fieldset> <legend>Vos produits </legend>
<label>...</label>
<select name="produit">
  <option value="SuperKdo">...</option>
<option value="MegaTruc"> Mega Truc</option>
  <option value="BabyFun"> ... </option>
</select> <br/>
<label>...</label>
<textarea name="opinion" ... > ... </textarea>
</fieldset>
```

Formulaires HTML & PHP

- Exemple



Récapitulatif

- *Nom : Manuele*
- *Email : mkirschpin@univ-paris1.fr*
- *Sexe : Femme*
- *Produit préféré : MegaTruc*
- *Suggestion : bla bla bla bla*

```
<body>
<h1>Récapitulatif </h1>
<ul>
<?php
    echo "<li> Nom : " . $_POST["nom"] . "</li>" ;
    echo "<li> Email : " . $_POST["email"] . "</li>" ;
    echo "<li> Sexe : " . $_POST["sexe"] . "</li>" ;
    echo "<li> Produit préféré : " . $_POST["produit"] . "</li>" ;
    echo "<li> Suggestion : " . $_POST["opinion"] . "</li>" ;
?>
</ul>
</body>
```



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Informatique Modélisation UML

Objectifs de la séance :

Instructions de contrôle en PHP

Fonctions

- **Instructions de contrôle**

- Instructions pour gérer le flot d'exécution
- **Instructions conditionnelles**
 - Elles conditionnent l'exécution
 - Semblables à un **nœud de Décision** (diagramme activités)
 - *if... else ..., switch ... case ...*
- **Instructions de boucle**
 - Elles permettent la **répétition** d'un bloc d'instructions
 - *for ... , foreach ... , while ... , do... while*

- Instructions conditionnelles if ... else...

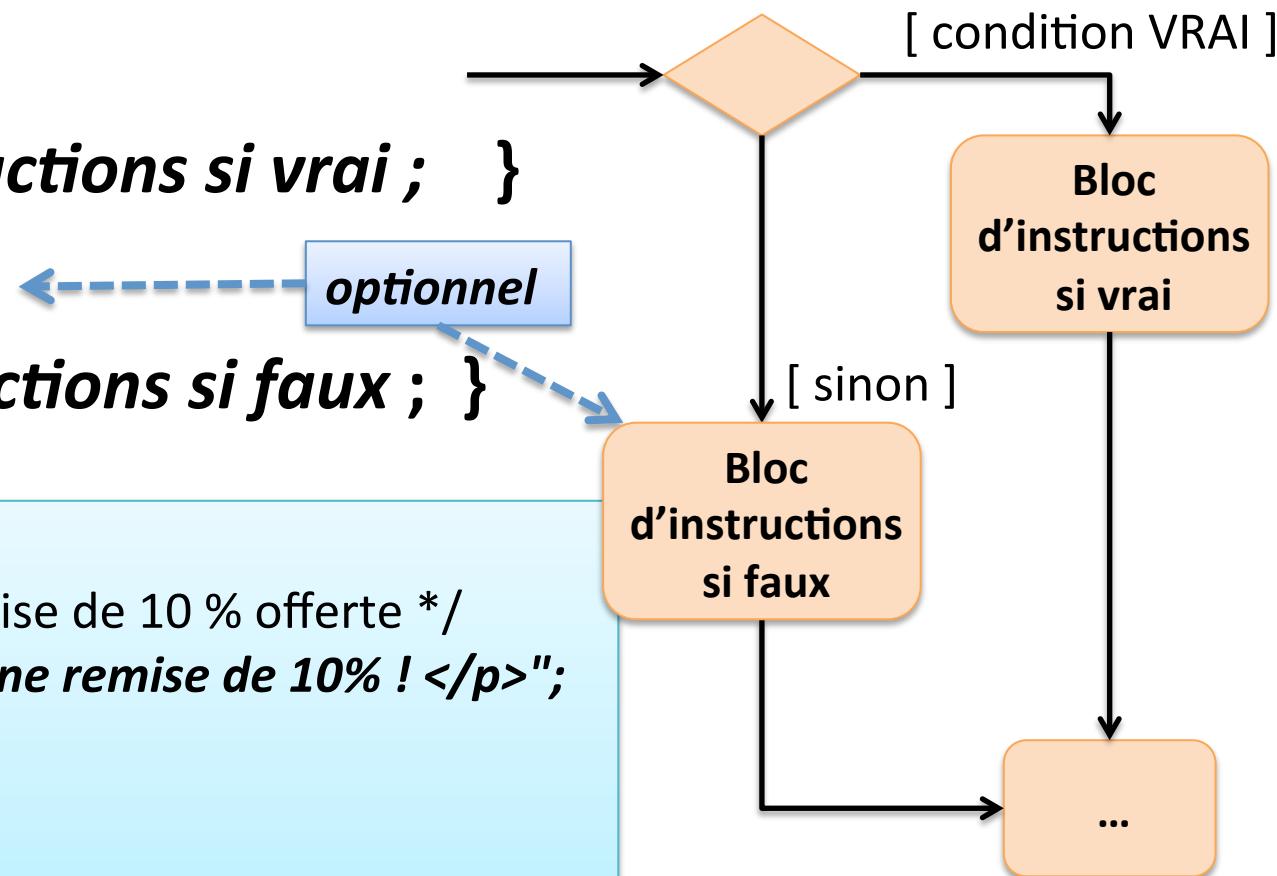
if (*condition*)

{ *bloc d'instructions si vrai ;* }

else

{ *bloc d'instructions si faux ;* }

```
if ( $qte >= 100 )
{
    $remise = 0.10; /* remise de 10 % offerte */
    echo "<p>Vous avez une remise de 10% ! </p>";
}
else {
    $remise = 0.05;
    echo "<p>Vous avez une remise de 5% </p>";
}
```





UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- **Instructions conditionnelles if ... else...**

- Les données pour la condition peuvent venir d'un formulaire

formExemple11.html

```
<form name="..." method="POST"
      action="coursPHP-11.php" >
...
<select name="prix">
    <option value="10">
        Super Kdo - 10€ </option>
    ...
</select>
...
<input type="number" size="10"
       name="qte" />
...
<input type="submit" value="Devis" />
</form>
```

<?php

```
$qte = $_POST["qte"];
$prixunit = $_POST["prix"];
$remise = 0;

if ( $qte >= 100 )
{
    $remise = 0.10; /* remise de 10 % offerte */
    echo "<p>Vous avez une remise de 10% ! </p>";
}

$prix = $prixunit * $qte
      - ($prixunit * $qte * $remise);
echo "<p> Pour un prix de <i> $prixunit </i>
l'unité et <i> $qte </i> unités, vous avez à
régler <i> $prix </i></p>";

?>
```

coursPHP-11.php



PHP

Préparer votre devis

Produit : Super Kdo - 10€

Quantité : 100

Devis

```
<form name="..." method="POST"  
      action="coursPHP-11.php" >
```

```
    <label>Produit : </label>
```

```
    <select name="prix">
```

```
        <option value="10" >Super ... </option>
```

```
        ... </select> <br/>
```

```
    <label>Quantité : </label>
```

```
    <input name="qte" type="number"
```

```
                  size="10" /> <br/>
```

```
    <input type="submit" value="Devis" />
```

```
</form>
```

Devis

Vous avez droit à une remise de 10% !

Pour un prix de 10 l'unité et 100 unités, vous avez à régler 900

```
<?php  
$qte = $_POST["qte"];  
$prixunit = $_POST["prix"];  
$remise = 0;  
....  
if ( $qte >= 100)  
{ $remise = 0.10;  
echo "<p>Vous avez .... </p>";  
}  
... ?>
```



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- **Instructions conditionnelles if ... else...**
 - Les blocs if... else ... peuvent contenir n'importe quelle instruction, y compris d'autres blocs if... else ...

```
if ( condition1 )  
{   bloc d'instructions si condition1 vraie ;  }  
elseif (condition2)  
{   bloc d'instructions si condition2 vraie ;  }  
else  
{   bloc d'instructions si les conditions sont fausses ;  }
```



Préparer votre devis

Produit : Super Kdo - 10€

Quantité : 100

Devis

```
<form name="..." method="POST"
      action="coursPHP-12.php" >
...
<select name="prix"> ... </select>
...
<input type="number" ... name="qte"/>
...
<input type="submit" value="Devis" />
</form>
```

Devis

Prix unitaire : 10 , Quantité : 100 , Remise : 10 %

Total à régler : 900

PHP

```
<?php
$qte = $_POST["qte"];
$prixunit = $_POST["prix"];

if ( $qte >= 100 )
    { $remise = 0.10 ; }
elseif ( $qte >= 50 )
    { $remise = 0.05 ; }
else
    { $remise = 0 ; }

$prix = $prixunit * $qte
      - ($prixunit * $qte * $remise);

echo "<p> Prix unitaire : <i> $prixunit </i>,
      Quantité : <i> $qte </i>,
      Remise : <i>" . $remise*100 . "</i> % </p>";

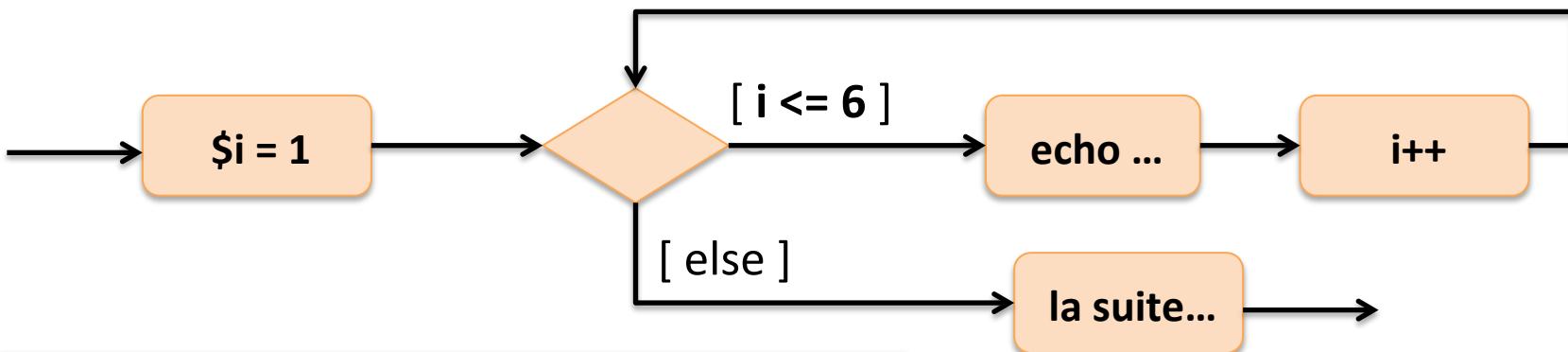
echo "<p><i>Total à régler : </i>
      <b> $prix </b></p>";

?>
```

- **Instructions de boucle : for**

- La boucle **for** permet de **répéter** (un certain nombre de fois) l'exécution d'un bloc d'instructions

for (*initialisation* ; *condition* ; *incrémantation*)
{ *bloc d'instructions à répéter* ; }



```
for ( $i = 1 ; $i <= 6 ; $i++ )
{
    echo "<h$i> Titre niveau $i </h$i>";
}
```

$\$i++ \rightarrow \$i = \$i + 1$



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- Instructions de boucle : for

```
<?php
for ( $i = 1 ; $i <= 6 ; $i++)
{
    echo "<h$i> Titre niveau $i </h$i>";
}
?>
```

Titre niveau 1

Titre niveau 2

Titre niveau 3

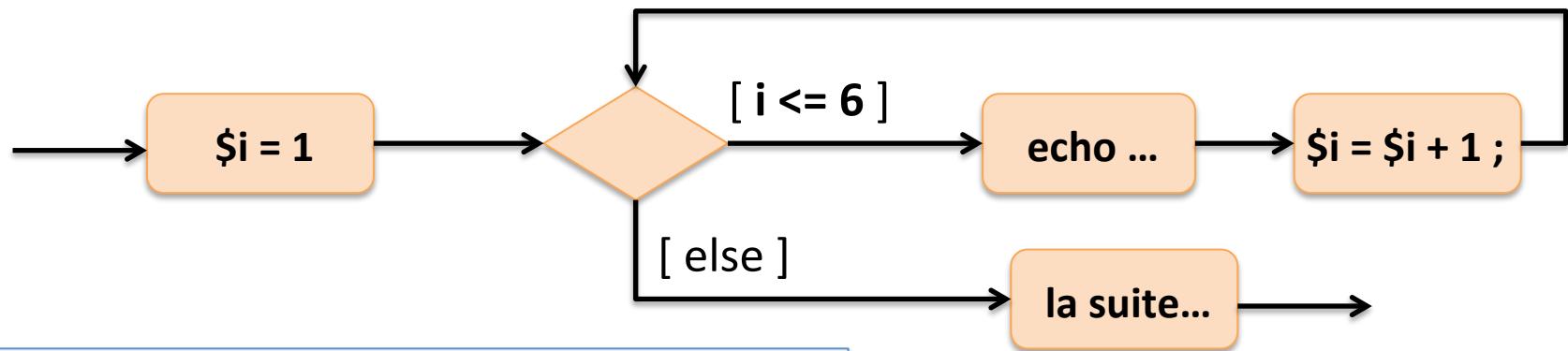
Titre niveau 4

Titre niveau 5

Titre niveau 6

- Instructions de boucle : **while**

- La boucle **while** permet de continuer à réaliser un bloc d'opérations **tant qu'une condition soit vraie**



```
$i = 1;  
while ( $i <= 6 ) {  
    echo "<h$i> Titre niveau $i </h$i>";  
    $i = $i + 1;  
}
```



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- Instructions de boucle : while

```
<?php
```

```
    $i = 1;
```

```
    while ( $i <= 6 ) {
```

```
        echo "<h$i> Titre niveau $i </h$i>";
```

```
        $i = $i + 1;
```

```
}
```

```
?>
```

On donne une **valeur initiale** à la variable **\$i**

Tant que **\$i** ne dépasse pas la valeur 6

On met à jour la **valeur de la variable \$i**

Titre niveau 1

Titre niveau 2

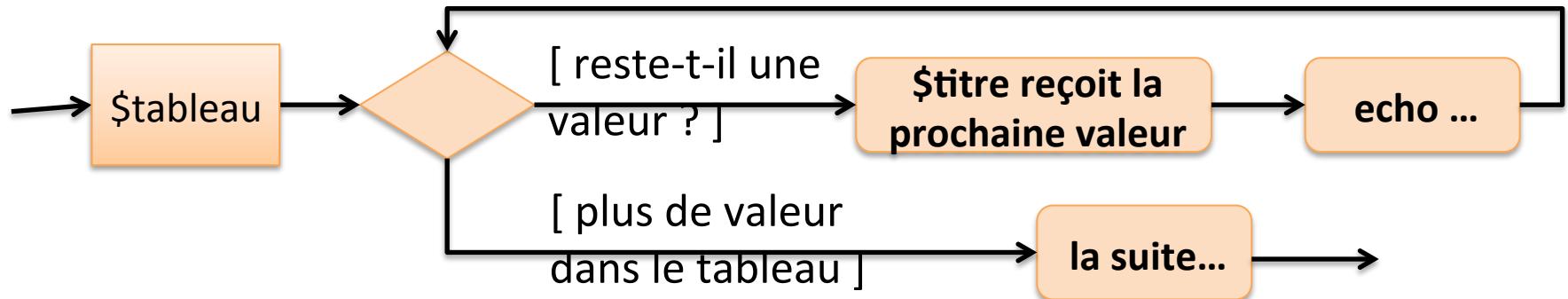
Titre niveau 3

Titre niveau 4

Titre niveau 5

Titre niveau 6

- **Instructions de boucle : foreach**
 - La boucle **foreach** permet de **répéter** un bloc d'instructions pour **chaque valeur dans un tableau**



```
foreach ($tableau as $titre) {  
    echo "<$titre> Titre $titre  
          </$titre>";  
}
```



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- Instructions de boucle : `foreach`

```
<?php  
    On définit un tableau  
    $tableau = array("h1", "h2", "h3",  
                    "h4", "h5", "h6");  
foreach ($tableau as $titre ) {  
    echo "<$titre> Titre $titre  
          </$titre>";  
}  
?>
```

Pour chaque valeur
dans le tableau

Titre h1
Titre h2
Titre h3
Titre h4
Titre h5
Titre h6

- Instructions de boucle : **foreach**
 - ça fonctionne aussi pour les tableaux associatifs

```
<?php  
  
$tableau = array ("nom" => "Dupont" ,  
                  "prenom" => "Jean" ,  
                  "adresse" => "qq part à Paris" ) ;  
  
foreach ($tableau as $cle=>$valeur) {  
    echo "<li> $cle : $valeur </li>" ;  
}  
?>
```

On définit un tableau associatif : clé => valeur

Pour chaque pair
\$clé => \$valeur
dans \$tableau

- *nom : Dupont*
- *prenom : Jean*
- *adresse : qq part à Paris*



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- **Instructions de boucle : boucles imbriquées**
 - Il est possible d'imbriquer des boucles les unes dans les autres

```
<table>
<?php
    for ( $lin = 1 ; $lin <= 9 ; $lin++ ) {
        echo "<tr> ";
        for ( $col = 1 ; $col <= 9 ; $col++ ) {
            echo "<td> "
                . ($col * $lin) . "</td> ";
        }
        echo "</tr> ";
    }
?>
</table>
```

1	2	3	4	5	6	7	8	9
2	4	6	8	10	12	14	16	18
3	6	9	12	15	18	21	24	27
4	8	12	16	20	24	28	32	36
5	10	15	20	25	30	35	40	45
6	12	18	24	30	36	42	48	54
7	14	21	28	35	42	49	56	63
8	16	24	32	40	48	56	64	72
9	18	27	36	45	54	63	72	81

- ## Fonctions

- PHP offre une large panoplie de fonctions
 - Exemple : **isset(\$var)** → TRUE si \$var est connue
 - Exemple : **empty(\$var)** → TRUE si \$var est vide (ou vaut 0)
- On peut aussi écrire les nôtres
(même en dehors des classes)
 - **function nomFonction (\$paramètre , ...) { instructions }**

```
function salutation ( $nom ) {  
    echo "<h1>Bienvenue, $nom ! </h1>";  
    echo "<p class=droite>Aujourd'hui, nous sommes le " .date('d / m / Y'). "</p>";  
}
```



```
...  
<form name="..." method="POST"  
      action="coursPHP-15.php" >  
<label>Nom : </label>  
<input type="text" name="client"  
      size="25"/>  
...  
<input type="submit" value="OK" />  
</form>
```

Identification

Nom : manuele

Mot de passe :

OK

Nom :

Mot de passe :

OK

DID

```
<?php  
function salutation ( $nom ) {  
    date_default_timezone_set("Europe/Paris");  
    echo "<h1>Bienvenue, $nom ! </h1>";  
    echo "<p class=droite>Aujourd'hui..."  
        . date('d / m / Y'). "</p>" ;  
}  
  
if ( isset ($_POST["client"]) AND  
    ! empty ($_POST["client"]) ) {  
    salutation ( $_POST["client"] ) ;  
}  
else { salutation ("cher client") ; }  
?>
```

Bienvenue, manuele !

Aujourd'hui, nous sommes le 04 / 03 / 2012

Bienvenue, cher client !

Aujourd'hui, nous sommes le 04 / 03 / 2012

Manu





UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- **Importation des fichiers**

- Incorporer le contenu d'un fichier dans une page PHP
- But : réutilisation des fichiers, uniformisation du site

- **include "fichier" et include_once "fichier"**

- **include** remplace la ligne par le contenu du fichier
- **include_once** fait ça **une seule fois** (même dans une boucle)

- **require "fichier" et require_once "fichier"**

- idem **include**, mais si le fichier n'existe pas, on a une **erreur**



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

```
<meta charset="UTF-8" />
...
<title>Mon site</title>
<link rel="stylesheet"
      href="css/blocs.css" />
```

```
<header>
  <h1>Mon site</h1>
</header>
<nav>
  <h2>Exemples </h2>
  <ul>
    <li>...</li>
  ...
  </ul>
</nav>
```

```
<head> <?php
      include_once "head.html";
      require "mesfonctions.php" ; ?>
</head> <body>
```

```
<?php include_once "headerNav.html"; ?>
```

```
...
```

```
<?php
      salutation ("cher client");
?>
```

```
<article>
  <h2> News </h2>
  <p> ... </p>
</article>
...
```

```
<?php
function salutation ( $nom ) {
  echo "<p class=droite><b>Bienvenue,
        $nom ! </b></p>";
  echo "<p class=droite>Aujourd'hui,
        nous sommes le "
  .date('d / m / Y'). " </p>";
}
?>
```



include_once "head.html"

```
<head> ...
  <title>Mon site</title>
  <link rel="stylesheet"
        href="css/blocs.css" />
```

```
</head>
<body>
```

```
<header> <h1>Mon site</h1> </header>
```

```
<nav>
```

```
<h2>Exemples </h2>
```

include_once "headerNav.html";

```
<ul>
```

```
  <li>...</li>
```

```
...
```

```
  </ul>
```

```
</nav>
```

```
<section>
```

```
<p class=droite><b>Bienvenue, cher client ! </b></p><p
class=droite>Aujourd'hui, nous sommes le 22/ 03 / 2014 </p>
```

```
...
```

Mon site

Bienvenue, cher client !

Aujourd'hui, nous sommes le 22 / 03 / 2014

News

bla blabla bla blabla bla blabla bla blablabla bla bla
blablabla bla blabla bla blabla bla blabla bla blabla bla bla
bla blabla bla blabla bla blabla bla blabla bla blabla bla

```
require "mesfonctions.php" ;
salutation ("cher client") ;
```



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Informatique Modélisation UML

Objectifs de la séance :
PHP & les bases de données

PHP

- **Accès aux bases de données à partir de PHP**
 - PHP-MySQL sont très utilisés pour les sites Web
 - Différents « bibliothèques » disponibles
 - **mysqli** et PDO
- **Etapes pour l'utiliser une base des données**
 - 1) connexion au serveur MySQL
 - 2) envoi des requêtes SQL (select, insert into...)
 - 3) récupération des résultats
 - 4) fermeture de la connexion

- **Connexion à un serveur MySQL à travers mysqli**
 - Toute la communication avec la BdD passe par un **objet** de la classe « **mysqli** »
 - La **connexion** s'effectue à la **création de cet objet (new)**

\$idcon = new mysqli (\$host, \$user, \$mdp, \$bdd);

The code '\$idcon = new mysqli (\$host, \$user, \$mdp, \$bdd);' is annotated with arrows pointing from descriptive text to the corresponding parameters:

- An arrow points from the text 'objet identifiant de la connexion' to the variable '\$host'.
- An arrow points from the text 'nom du serveur' to the variable '\$user'.
- An arrow points from the text 'mot de passe' to the variable '\$mdp'.
- An arrow points from the text 'utilisateur autorisé à accéder à la base' to the variable '\$bdd'.
- An arrow points from the text 'base de données' to the variable '\$bdd'.

- **Toute connexion ouverte doit être fermée**

\$bool = \$idcon->close () ;

The code '\$bool = \$idcon->close () ;' is annotated with a bracket and text:

on demande à l'objet mysqli de fermer la connexion



PHP

```
<?php  
$host = "localhost";  
$user = "root";  
$mdp = "root";  
$bdd = "clientsBD";
```



Astuce : placer ces informations dans un fichier et faire **require (ou include)** "fichier"

```
$mysqli = new mysqli ($host, $user, $mdp, $bdd);
```

Création de l'objet connexion

```
if ( $mysqli->connect_errno ) {  
    die ("<p> Impossible de connecter à $bdd : "  
        . $mysqli->connect_error . "</p>" );
```

```
}  
else {  
    echo "<p> Connecté au serveur $host,  
        à la base $bdd </p>";
```

L'attribut **connect_errno** de indique si la connexion a bien été établie

```
    $mysqli->close();
```

En cas de problème, on arrête avec la fonction **die**.

```
}
```

Fermeture de la connexion

- Envoie de requêtes à une base de données

`$result = $mysqli->query ($sql) ;`

Résultat de la requête

*exécution de la requête
sur l'objet connexion*

*Requête SQL à
exécuter*

- Requête SQL :

- S'il s'agit d'un **SELECT**, le **résultat** correspond aux **données** fournies par la requête (objet **mysqli_result**)
- Sinon (**INSERT, UPDATE, DELETE...**), le résultat sera **TRUE** si la requête est **bien exécutée**, **FALSE** sinon



```
<form name="formNouveauClient"
      action="coursPHP-18.php"
      method="POST">
  ...
  <input type="text" name="nom" ... />
  ...
  <input type="text" name="email" ... />
  ...
  <input type="submit" value="Envoyer" />
</form>
```

id	nom	email	adresse
1	Manuele Kirsch Pinheiro	mkirschpin@univ-paris1.fr	90 rue Tolbiac Paris

Selection : [Modifier](#) [Effacer](#) [Exporter](#)

Nombre de lignes: En-têtes à chaque

PHP

Formulaire

Nouveau client

Nom	<input type="text" value="Manuele Kirsch Pinheiro"/>
Email	<input type="text" value="mkirschpin@univ-paris1.fr"/>
Adresse	<input type="text" value="90 rue Tolbiac, 75013 Paris"/>

Connexion BdD

Insertion de données

connecté !

Vous êtes le client numéro 1

Manuele Kirsch Pinheiro

UFR06 Gestion

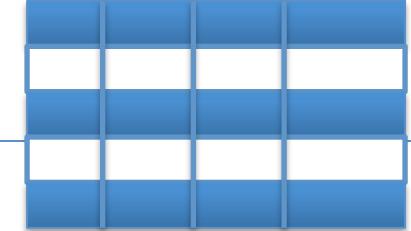


coursPHP-18.php

```
<?php  
if ( ! empty($_POST["nom"]) AND ! empty($_POST["email"]) ) {  
    require "connexion.php";  
    $mysqli = connexion();  
  
    $nom = $_POST["nom"];  
    $email = $_POST["email"];  
    $adr = $_POST["adresse"];  
    $id = '\N'; /* auto-increment */  
  
    $sql = "INSERT INTO client (id, nom, email, adresse)  
           VALUES ( '$id', '$nom', '$email', '$adr' ) ";  
  
    $result = $mysqli->query ($sql) ;  
  
    if ( !$result ) { echo "<p>Désolée, ... </p>"; }  
    else {  
        echo "<p> Vous êtes le client numéro <i>" .  
              $mysqli->insert_id . "</i></p>";  
    }  
    $mysqli->close();  
}  
... ?>
```

connexion.php

```
<?php  
function connexion() {  
    $host = "localhost";  
    $user = "uml";  
    $mdp = "uml";  
    $bdd = "clientsBD";  
  
    $mysqli = new mysqli ($host,  
                         $user, $mdp, $bdd) ;  
  
    if ( $mysqli->connect_errno ) {  
        die ("<p> Impossible ..." .  
              $mysqli->connect_error . "</p>");  
    }  
    return $mysqli ;  
} ?>
```



- **Récupération des données**

`$result = $mysqli->query ("SELECT * FROM table");`

- Les requêtes SELECT fournissent des données
- On récupère le résultat (ligne à ligne) à l'aide des opérations `fetch_*`
- Chaque appel à `fetch_*` retourne la **prochaine ligne**
 - Ligne dans un tableau à indice : `$result->fetch_row () ;`
 - Ligne dans un tableau associatif : `$result->fetch_assoc () ;`
 - **Ligne dans un objet** : `$result->fetch_object () ;`

coursPHP-19.php

```
<?php
require "connexion.php";
$mysqli = connexion();

$sql = "SELECT id, nom, email, adresse
        FROM client ORDER BY nom ";
$result = $mysqli->query ($sql);

if ( ! $result ) { echo "<p> Desolée ... </p>" ; }
else {
    while ( $ligne = $result->fetch_object() ) {
        ...
        echo "<td>" . $ligne->id . "</td>";
        echo "<td>" . $ligne->nom . "</td>";
        echo "<td>" . $ligne->email . "</td>";
        echo "<td>" . $ligne->adresse . "</td>";
        ...
    }
} ?>
```

Clients			
	Nom	Email	Adresse
1	Manuele Kirsch Pinheiro	mkirschpin@univ-paris1.fr	90 rue Tolbiac, 75013 Paris

PHP

connexion.php

```
<?php
function connexion() {
    ...
    $mysqli = new mysqli ($host,
    $user, $mdp, $bdd);
    ...
    return $mysqli ;
} ?>
```

On exécute la requête avec l'opération \$mysqli->query

L'opération \$result->fetch_object récupère la prochaine ligne, FAUX s'il n'y reste plus de lignes.

Chaque attribut de la requête devient un attribut de l'objet \$ligne



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

La même requête avec récupération des informations

- ... par tableau à indice

```
...  
$sql = "SELECT id, nom, email, adresse  
        FROM client ORDER BY nom " ;
```

```
$result = $mysqli->query ($sql) ;
```

```
...  
while ( $ligne = $result->fetch_row() ) {
```

```
...  
    echo "<td>" . $ligne[0] . "</td>";  
    echo "<td>" . $ligne[1] . "</td>";  
    echo "<td>" . $ligne[2] . "</td>";  
    echo "<td>" . $ligne[3] . "</td>";  
...  
}
```

ça commence toujours par **0**

- ... par tableau associatif

```
...  
$sql = "SELECT id, nom, email, adresse  
        FROM client ORDER BY nom " ;
```

```
$result = $mysqli->query ($sql) ;
```

```
...  
while ( $ligne = $result->fetch_assoc () ) {
```

```
...  
    echo "<td>" . $ligne['id'] . "</td>";  
    echo "<td>" . $ligne['nom'] . "</td>";  
    echo "<td>" . $ligne['email'] . "</td>";  
    echo "<td>" . $ligne['adresse'] . "</td>";  
...  
}
```

chaque attribut est accessible
par son **nom**

- Autres informations peuvent être récupérées d'un objet **mysqli_result** (**\$result = \$mysqli->query (...)**)
 - **Combien de lignes et colonnes** on peut récupérer
 - **\$nblignes = \$result->num_rows ;**
 - **\$nbcol = \$result->field_count ;**
 - **Les noms des colonnes (attributs)** dans le résultat
 - **\$colonnes = \$result->fetch_fields();**



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

```
...  
$sql = "SELECT id, nom, email, adresse  
        FROM client ORDER BY nom " ;  
$result = $mysqli->query ($sql) ;  
...  
echo "<p> Nous avons " . $result->num_rows . " clients. </p>" ;  
echo "<p> Il y a " . $result->field_count . " attributs par client. </p>" ;  
...  
u
```

A partir de l'objet **\$result**, on peut récupérer le nombre de lignes (attribut **num_rows**) et de colonnes par ligne (attribut **field_count**).

On peut aussi récupérer les **colonnes**. Chaque colonne est un **objet** et l'attribut **name** donne son nom.

La ligne aussi est un objet dont les **attributs** correspondent aux **colonnes**. On peut utiliser un **foreach** pour accéder à la **valeur des attributs**.

```
...  
$titres = $result->fetch_fields() ;  
foreach ($titres as $colonne) {  
    echo "<th> " . $colonne->name . " </th>" ;  
}  
while ( $ligne = $result->fetch_object() ) {  
    echo "<tr>" ;  
    foreach ( $ligne as $colonne=>$val ) {  
        echo "<td> " . $val . " </td>" ;  
    }  
    echo "</tr>" ;  
} ...  
u
```



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

Informatique Modélisation UML

Objectifs de la séance :
Mécanismes de sessions



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

- **Mécanisme de sessions**
 - Chaque visite à un site / page est indépendante
 - Les **sessions** permettent de conserver les informations des visiteurs **entre les pages**
 - Les informations sur les sessions sont stockées sur le **serveur**
- Fonctionnement général
 - 1) Ouverture de session : **session_start ()**
 - Chaque utilisateur reçoit un identifiant transmis entre les pages
 - 2) Définition des variables de sessions (données)
 - Les variables de session sont transmises de page à page
 - **`$_SESSION["variable"] = valeur ;`**
 - 3) Fermeture de session : **session_destroy()**



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

localhost:8888/exem

Identification

Login :

Mot de passe :

Login & mdp
différents de uml

localhost:8888/exemp

Desolé !

Page accessible uniquement aux membres.

Login & mdp corrects
(uml /uml)

localhost:8888/exemplesPHP/coursPHP-23.php

- [Accueil membre](#)
- [Deconnexion](#)

Bienvenue, cher uml

```
<form name="..."  
action="coursPHP-23.php"  
method="POST">  
    <label>Login : </label>  
    <input type="text" name="login" maxlength="15" /> <br/>  
    <label>Mot de passe : </label>  
    <input type="password" name="mdp" maxlength="15" />  
    <br/>  
    <input type="submit" value="OK" />  
</form>
```



PHP

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

Ouverture d'une session
(au début de chaque page)

```
<html>  
  <head> ... </head>  
  <body>  
    <?php  
    ...
```

```
    $login = $_POST["login"] ;  
    $mdp = $_POST["mdp"];
```

Définition des variables de session
 $$_SESSION["var"]$

```
    if ( $login == "uml" AND $mdp == "uml") {  
        $_SESSION["login"] = $login ;  
        ...
```

```
        echo "<h1>Bienvenue, cher $login </h1>" ;  
    }
```

```
    else { echo "<h1>Desolé ! </h1>";  
          echo "<p> Page accessible uniquement aux membres. </p>";  
    }
```

```
?>
```

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

Les variables de session contiennent les informations qui passeront de page en page.



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

localhost:8888/exemplesPHP/coursPHP-23.php

localhost:8888/exemplesPHP/coursPHP-24.php

Bienvenue, cher uml

• [Accueil membre](#)
• [Deconnexion](#)

Mon site

Client uml : Ceci est une page pour les abonnés

<?php session_start(); ?>
<html> <head>... </head>
<body>
<?php
if (isset(\$_SESSION["login"]) AND ! empty(\$_SESSION["login"])) {
 \$login = \$_SESSION["login"] ;
...
 echo "<p>Client \$login : Ceci est un abonné." ;
}
else {
 echo "<h1>Desolé ! </h1>" ;
 echo "<p> Il s'agit d'une page privée !! Il faut être membre. </p>" ;
}
?>
...

Usage des variables de session
\$_SESSION["var"]



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

The diagram illustrates a user session flow. It starts with a screenshot of a browser window titled "localhost:8888/exemplesPHP/coursPHP-23.php". The page displays a navigation menu with links to "Accueil membre" and "Deconnexion". A red arrow points from this menu to another browser window titled "localhost:8888/exemplesPHP/coursPHP-25.php", which shows a "Deconnexion" page with the message "Au revoir et à bientôt, cher uml". Below the first browser window, the corresponding PHP code is shown:

```
<?php session_start(); ?>
<html> <head>... </head>
<body>
<?php
    if ( isset( $_SESSION["login"] ) AND ! empty( $_SESSION["login"] ) ) {
...
        unset($_SESSION["login"]);
        session_destroy();
    }
    else { echo "<h1>Desolé ! </h1>";
        echo "<p> Pas de connexion active. </p>";
    }
?>
...

```

Two callout boxes provide additional information:

- A blue box labeled "Fermeture de la session **session_destroy()**" points to the line `session_destroy();`.
- A blue box labeled "Ne pas oublier de vider les variables de session **unset(\$_SESSION["var"])**" points to the line `unset($_SESSION["login"]);`.

- **Mécanisme de sessions**

- Base pour la gestion de **panier** dans les sites de e-commerce
- Les produits choisis par le client sont enregistrés en tant que **variables de session**
- On peut y garder des **objets SIMPLES**

```
class LigneProduit {  
    public $nom ;  
    public $qte ;  
  
    /* constructeur */  
    function __construct( $nom ) {  
        $this->nom = $nom;  
        $this->qte = 1;  
    }  
}
```

Contenu du panier est gardé dans les variables de session.
Tableau contenant des objets LigneProduit.
Chaque **\$_SESSION[\$produit]** contient un objet.

PHP

```
function ajouterProduit($produit) {  
    $qte = 0;  
  
    if ( ! isset ( $_SESSION[$produit] ) ) {  
        $_SESSION[$produit] = new LigneProduit($produit);  
        $qte = $_SESSION[$produit]->qte  
    }  
  
    else { // produit déjà là, augmenter alors sa quantité  
        $objet = $_SESSION[$produit] ;  
        $objet->qte = $objet->qte + 1;  
        $qte = $objet->qte ;  
    }  
    return $qte;  
}
```

Chaque produit choisi est identifié par un « id » (ici le nom).
`$_SESSION[$produit]` va contenir un objet **LigneProduit**

S'il n'y a aucun **`$_SESSION[$produit]`**, on va créer un nouveau objet **LigneProduit**

S'il y a déjà un **`$_SESSION[$produit]`**, on va juste augmenter la valeur de l'attribut « qte » dans l'objet **LigneProduit**

PHP

```
function supprimerProduit($produit) {  
    $qte = 0 ;  
  
    if ( isset( $_SESSION[$produit] ) ) {  
        $objet = $_SESSION[$produit] ;  
        $objet->qte = $objet->qte - 1;  
        $qte = $objet->qte;  
  
        if ( $qte <= 0) { //on supprime le produit  
            unset($_SESSION[$produit]);  
        }  
    }  
  
    return $qte;  
}
```

Lorsqu'on veut supprimer un produit, on va réduire sa quantité dans l'objet **LigneProduit**

On récupère l'objet **LigneProduit** gardé dans **\$_SESSION[\$produit]**

On réduit sa quantité d'une unité

S'il n'en reste plus (la **quantité** a atteint **0 unités**), on **supprime** le produit de la session



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

On peut récupérer le contenu du panier en récupérant le contenu de la variable de session

`$_SESSION`

```
function afficherPanier() {  
    echo "<table>" ;  
    foreach($_SESSION as $objet) {  
        echo "<tr><td> " . $objet->nom . " </td> <td> "  
            . $objet->qte . " </td> </tr> " ;  
    }  
    echo "</table>" ;  
}
```

Pour chaque **objet**
LigneProduit gardé dans
`$_SESSION`



UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP

Nos Produits

[Terminer](#)[Voir panier](#)

Produit	Prix		
Autocollant Autocollant au symbole du club	15.00 €	<input style="width: 20px; height: 20px; border: none; background-color: #f0f0f0; border-radius: 50%;" type="button" value="+"/>	<input style="width: 20px; height: 20px; border: none; background-color: #f0f0f0; border-radius: 50%;" type="button" value="-"/>
T-shirt Official T-shirt official du club	35.00 €	<input style="width: 20px; height: 20px; border: none; background-color: #f0f0f0; border-radius: 50%;" type="button" value="+"/>	<input style="width: 20px; height: 20px; border: none; background-color: #f0f0f0; border-radius: 50%;" type="button" value="-"/>
T-shirt baby-look T-shirt au symbole du club	25.00 €	<input style="width: 20px; height: 20px; border: none; background-color: #f0f0f0; border-radius: 50%;" type="button" value="+"/>	<input style="width: 20px; height: 20px; border: none; background-color: #f0f0f0; border-radius: 50%;" type="button" value="-"/>

Exemple de Panier

[Produits](#)

Ajouter T-shirt Official au panier : quantité 1

[Terminer](#)

T-shirt Official

1

[Voir panier](#)

Exemple de Panier

[Produits](#)

Ajouter autocollant au panier : quantité 1

[Terminer](#)

T-shirt Official

1

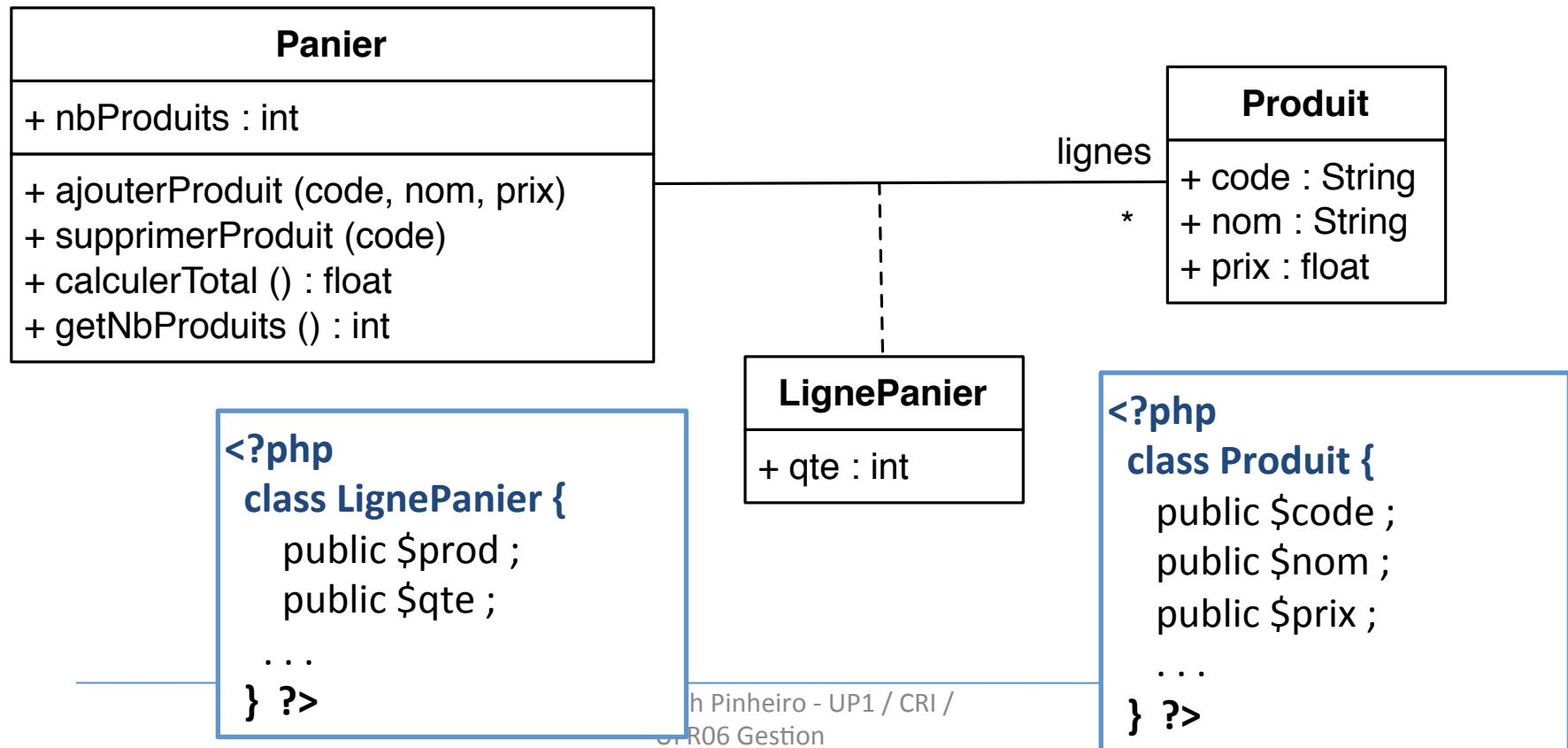
[Voir panier](#)

autocollant

1

PHP : Panier avancé

- Voici un exemple avancé de Panier qui utilise les classes en PHP et la notion de session
- Le panier est modélisé par une **classe Panier**



PHP : Panier avancé

```
class Panier {  
    public $lignes ; ←  
    public $nbProduits ;  
  
    function __construct() {  
        $this->nbProduits = 0 ;  
    }  
    ...  
    function ajouterProduit($code, $nom, $prix) {  
        if ( $this->nbProduits == 0 ) {  
            $prod = new Produit($code, $nom, $prix);  
            $lp = new LignePanier($prod);  
            $this->lignes[$code] = $lp;  
            $this->nbProduits = 1;  
        }  
        ...  
    }  
}
```

Chaque Ligne de Panier est gardée dans un tableau associatif
`$this->lignes[$code] => $LignePanier`

On commence avec zéro produits dans le panier

On va créer le tableau lors de l'ajout du premier produit au panier

PHP : Panier avancé

```
function ajouterProduit($code, $nom, $prix) {  
    if ( $this->nbProduits == 0) { . . . }  
    else {  
        if ( isset ( $this->lignes[$code] ) ) {  
            $lp = $this->lignes[$code] ;  
            $qte = $lp->qte;  
            $lp->qte = $qte + 1;  
        }  
        else {  
            $prod = new Produit($code, $nom, $prix);  
            $lp = new LignePanier($prod);  
  
            $this->lignes[$code] = $lp;  
            $this->nbProduits = $this->nbProduits + 1;  
        }  
    }  
}
```

Pour ajouter, on vérifie si le produit est déjà dans le panier

S'il y est, on le récupère et on met à jour la quantité

S'il n'y est pas, on va y ajouter une nouvelle ligne de panier

PHP : Panier avancé

```
function ajouterProduit($code, $nom, $prix) {
```

```
    if ( isset ( $this->lignes[$code] ) ) {
```

On ne supprime que si le produit est dans le panier

```
        $lp = $this->lignes[$code] ;  
        $lp->qte = $lp->qte - 1 ;
```

S'il y est, on met à jour la quantité,
en **supprimant 1 unité**

```
        if ( $lp->qte < 1 ) {  
            unset($this->lignes[$code]);  
            $this->nbProduits = $this->nbProduits - 1;
```

Par contre, s'il ne reste plus rien (qte < 1), on **supprime la ligne de panier** du tableau

```
    }
```



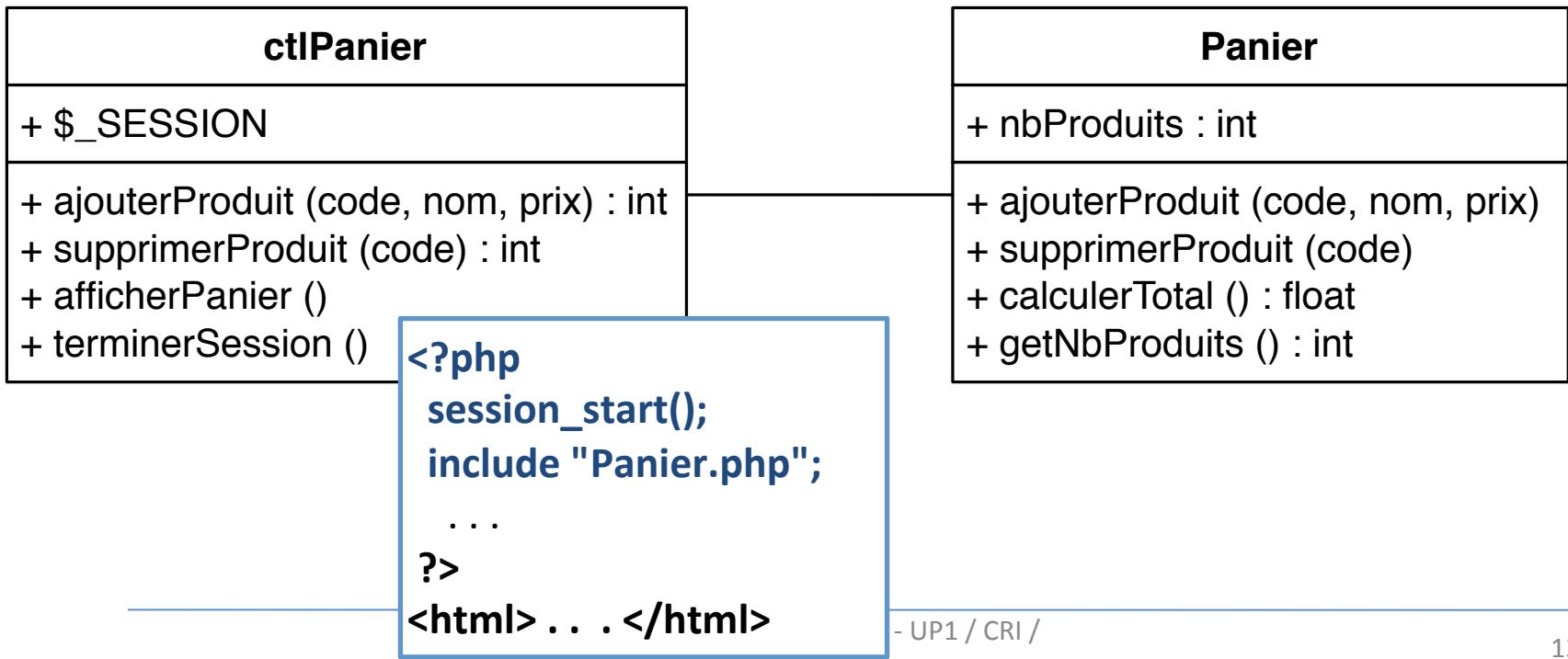
UNIVERSITÉ PARIS 1

PANTHÉON SORBONNE

ÉCOLE DE MANAGEMENT
DE LA SORBONNE

PHP : Panier avancé

- C'est un objet **Panier** que notre site va manipuler
- Une page « **ctlPanier.php** » va ainsi gérer le panier
- Pour cela, elle va devoir garder un objet **Panier** dans **\$_SESSION**



PHP : Panier avancé

- Or un objet **Panier** est un objet **complexe**
- Pour le garder dans **\$_SESSION**, il va falloir le « compacter » : c'est la **sérialisation**
 - **\$_SESSION["panier"] = serialize(\$panier)**
 - **unserialize (\$_SESSION["panier"])**

```
function ajouterProduit($produit, $nom, $prix) {  
...  
$panier = unserialize($_SESSION["panier"]);  
$panier->ajouterProduit($produit, $nom, $prix);  
$_SESSION["panier"] = serialize($panier);  
...}
```

Pour ajouter ou supprimer un produit au panier, on va le récupérer, le modifier puis le remettre dans la session

```
function supprimerProduit($produit) {  
...  
$panier = unserialize($_SESSION["panier"]);  
$panier->supprimerProduit($produit);  
$_SESSION["panier"] = serialize($panier);  
...}
```