Le langage PHP

Sommaire

Les principaux aspects du langage PHP

- généralités
- littéraux, variables et constantes
- tableaux associatifs et indicés
- boucles et tests
- Quelques exemples

Qu'est-ce qu'un « script » PHP

- C'est un fichier d'extension `.php` contenant du code PHP.
- Tout code PHP doit être inclus dans une balise `<?php ... ?>`.
- Le code PHP produit du texte HTML avec la commande `echo`

$\Rightarrow$ on obtient un document HTML produit *dynamiquement*.

Exemple : un script très simple (ExPHPa.php)

```
<HTML> <HEAD>
<TITLE>HTML avec PHP</TITLE></HEAD>
<BODY bgcolor='white'>

Ici c'est du HTML statique.

<P>

<?php
  echo "Ici c'est du HTML dynamique";
?>
</P>
</BODY></HTML>
```

Traitement d'un script PHP

- On appelle avec le navigateur une URL de la forme *http://serveur/script.php*
- Le serveur web accède au script et exécute le code PHP qu'il contient.
- L'exécution du code produit un document HTML qui est renvoyé au navigateur

Attention: PHP est exécuté *sur le serveur*: le navigateur reçoit du HTML (dynamique).

Syntaxe de PHP

PHP est un langage de programmation comme le C ou Java, mais

- beaucoup plus simple à apprendre et à utiliser (en particulier, pas de types)
- très adapté à la programmation Web (production de texte)
- énormément de fonctions et de scripts prêts à l'emploi (disponibles sur le Web)

Très utilisé pour la réalisation de sites web.

Syntaxe – les bases

- Rappel: le code PHP doit être inclus dans une balise `<?php ... ?>` (sinon pas interprété).
- Le code PHP se compose d'instructions séparées par des point-virgule;
- trois manières d'inclure des commentaires :
  1. entre les signes « /* » et « */ » ;
  2. en commençant une ligne par « // » :
  3. en commençant une ligne par « # ».

Littéraux et constantes

Littéral = valeur « en dur », non modifiable

- littéral numérique: 1 ou 3.14
- littéral chaîne de caractères: “Tintin en Amérique” encadré par des guillemets doubles.

Constantes = valeur référencée par un symbole non modifiable

- Créé avec la commande `define`

```
define (’PI’, ’3.14116’)
```

- Permet d’éviter les fautes de frappe ou de les corriger facilement

Variables

Variable = symbole référençant une valeur.

- syntaxe d'une variable: un $ suivi du nom. Exemple: `$adresse`
- une variable peut référencer des valeurs différentes au cours de l'exécution
- pas de variable typée en PHP: une variable peut référencer un nombre, puis une chaîne, ...
- pas de déclaration de variable en PHP !
- Attention: minuscules et majuscules. (`$adresse` et `$Adresse`: deux variables différentes)

Les types PHP

- Les entiers: 1, 2, 3, 12980
- Les flottants: 3.14, 1.23, 2093.2988
- Les booléens (TRUE ou FALSE)
- Les chaînes de caractères (entre guillemets doubles)
- Les tableaux et les objets

Typage très souple: PHP convertit le type en fonction de l'opération effectuée.

Rôle des types PHP

Essentiellement, PHP convertit le type d'une valeur en fonction de l'opération effectuée.

- Si j'écris `$a + $b`, PHP convertira `$a` et `$b` en numériques.

Quelle que soit le type d'une valeur, on peut la transformer en chaîne de caractères.

- Si j'écris `"Valeur 1=$a, Valeur 2=$b"`, PHP convertit `$a` et `$b` en chaînes et les inclut dans la chaîne principale.

PHP est conçu pour produire du texte!

Interpolation et concaténation

- **Interpolation**: pouvoir inclure la valeur d'une variable directement dans une chaîne de caractère.
- **Concaténation**: assemblage de deux chaînes de caractères avec l'opérateur « . ».

```
$a = 2;
$b = 3;
echo "$a + $b = " .  $a + $b .  "<BR>".
```

On obtient la chaîne "2 + 3 = 5
" dans le document HTML produit.

Les tableaux

Tableau = suite de valeurs référencées

- par un indice (tableaux indicés)
- ou par une clé (tableaux associatifs)

La paire (indice, valeur) ou (clé, valeur) est un élément du tableau.

La taille des tableaux est *dynamique*: on ajoute des éléments à volonté.

Tableaux indicés

Syntaxe pour désigner un élément: le nom du tableau, puis des crochets indiquant la clé ou l'indice.

```
$tab[0] = "élément 1 ";
$tab[1] = "élément 2 ";
$tab[2] = 120;
```

PHP peut affecter automatiquement les indices.

```
$tab[] = "élément 1 "; // $tab[0] !
$tab[] = "élément 2 "; // $tab[1] !
$tab[] = 120; // $tab[2] !
```

NB: les indices automatiques commencent à 0.

Tableaux associatifs

On doit donner explicitement la clé. Elle désigne de manière unique l'élément.

```
$mes["Vertigo"] = "Hitchcock";
$mes["Sacrifice"] = "Tarkovski";
$mes["Alien"] = "Scott";
```

Pour initialiser un tableau, on peut utiliser la commande `array`.

```
$tab = array ( "élément 1 ", "élément 2 ", 120)
$mes = array ("Vertigo" => "Hitchcock",
    "Sacrifice" => "Tarkovski",
    "Alien" => "Scott");
```

Tableaux multi-dimensionnels

Un élément est désigné par un ou plusieurs indices ou clés.

```
$tab[0][0] = "En haut à gauche";
$tab[0][1] = "En haut à droite";
$tab[1][0] = "En bas à gauche";
$tab[1][1] = "En bas à droite";
```

Valeur de `$tab[0]`: un tableau à une dimension.

```
$mes = array (
  "Vertigo" => array ( "Alfred", "Hitchcock"),
  "Sacrifice" => array ( "Andrei", "Tarkovski"
  "Alien" => array ( "Ridley", "Scott"));
```

Expression et affectation

Expression: toute instruction qui produit une valeur.

```
$a + 3;
“Mon nom est “ .  $nom
```

À peu près toutes les instructions sont des expressions en PHP.

Affectation: on affecte le résultat d’un expression à une variable.

```
$b = $a + 3;
$c = “Mon nom est “ .  $nom
```

Opérateurs arithmétiques

Classique : +, -, /, *, et % pour le modulo.

```
$a = 3;
$b = 8;
$c = $a + 2 * $b;
```

Priorités: voir le poly. Mais le plus simple est d'utiliser des parenthèses.

```
$a = 3;
$b = 8;
$c = $a + (2 * $b);
```

Concaténation de chaînes

C’est le point «.».

```
$c1 = "Bonjour "’;
$c2 = "Dominique";
// Affichage de la chaîne
// "Bonjour cher Dominique"
echo $c1 .  " cher " .  $c2;
```

Pour ajouter un fragment à une chaîne:

```
$c = "Bonjour cher";
$c = $c .  " Dominique"
```

ou encore la construction équivalente:

```
$c .= " Dominique"
```

Opérateurs de comparaison

Classique: $<$, $>$, $<=$, $>=$, sauf la comparaison: $==$ (deux « = »)

Une erreur très courante:

```
$i = 1;
$j = 2;

// Renvoie FALSE: i est différent de j.
if ($i == $j) ...
// Renvoie la valeur de $j, soit TRUE !
if ($i = $j) ...
```

Structures de contrôle

Permettent de diriger le flux d'exécution vers un ensemble d'instruction, ou *bloc*.

- Les tests: `if-else`, `switch`.
- Les boucles: `while`, `for`, `do`, `foreach`.

Dans un premier temps on peut se limiter à `if-else`, `while` et `foreach`.
Voir le polycopié pour les autres.

Les tests `if-then-else`

```
if (expression)
{
  // Bloc si expression est vraie.
}
else
{
  // Bloc si expression est fausse.
}
// Ici le script continue.
```

Exemple de test

```
if ($a == 2) {
  echo “La variable a vaut 2”
}
else {
  if ($a == 3) {
    echo “La variable a vaut 3”
  }
  else {
    echo “La variable a ne vaut ni 2 ni 3”
  }
}
```

Le `while`

Permet d'exécuter un bloc d'insructions tant qu'une condition est remplie.

```
while (expression)
{
  // Ici, expression est vraie.
}
```

Exemple:

```
$a = 0;
while ($a < 10) {
  echo “a vaut $a”;
  $a = $a + 1;
}
```

Le `foreach`

Permet de parcourir un tableau.

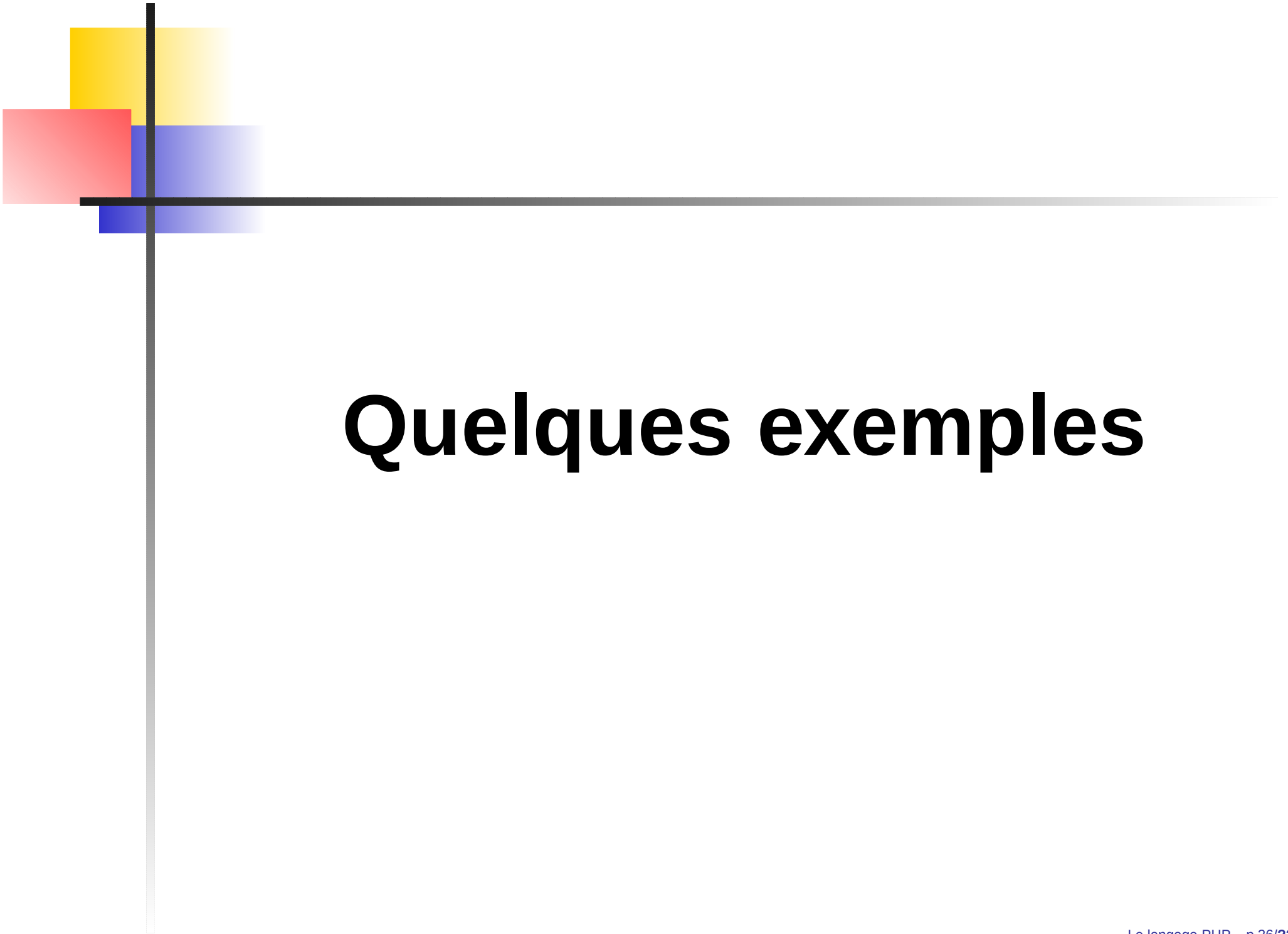
- Tableau indicé:

```
foreach ($tableau as $valeur)
  { /* bloc */ }
```

- Tableau associatif:

```
foreach ($tableau as $cle => $valeur)
  { /* bloc */ }
```

À chaque passage dans le bloc, `$valeur` contient la valeur de l'élément courant (`$cle` contient la clé pour les tableaux associatifs).



Quelques exemples

Quelques exemples

PHP permet de récupérer automatiquement les paramètres transmis par HTTP. Ces paramètres sont stockés dans des tableaux prédéfinis.

- le tableau `$_SERVER` contient les paramètres CGI sur le contexte de la demande.
- le tableau `$_POST` contient les paramètres transmis en mode POST.
- le tableau `$_GET` contient les paramètres transmis en mode GET.
- le tableau `$_REQUEST` contient tous les paramètres!

Premier exemple (SERVER.php)

```
<HTML><HEAD>
  <TITLE>Paramètres CGI</TITLE>
</HEAD>
<BODY>

<?php
// Script affichant les paramètres CGI.

$serveur = $_SERVER['SERVER_SOFTWARE'];
$client = $_SERVER['HTTP_USER_AGENT'];

echo "<TABLE BORDER=1>";
echo "<TR><TD>Le serveur<TD>$serveur";
echo "<TR><TD>Le client<TD>$client";
echo "</TABLE><P>";
?>
```

Deuxième exemple (GET.php)

```
<HTML><HEAD>
  <TITLE>Paramètres GET</TITLE>
</HEAD>
<BODY>

<TABLE border=2>
<?php
// Script affichant les paramètres GET.

foreach ($_GET as $cle => $valeur) {
  echo "<TR><TD>$cle<TD>$valeur";
}
?>
</TABLE></BODY></HTML>
```

Troisième exemple (Bonjour.php)

```
<HTML><HEAD>
  <TITLE>Dis bonjour</TITLE>
</HEAD><BODY>

<?php
if (isSet($_REQUEST['nom'])
    and isSet($_REQUEST['prenom'])) {
  $nom = $_REQUEST['nom'];
  $prenom = $_REQUEST['prenom'];

  echo "Bonjour $prenom $nom";
}
else
echo "Mais qui êtes-vous?";
?>

</BODY></HTML>
```