

# Le langage PHP

Mohammad Ghoniem

Université de Bretagne Sud

# Plan

- PHP et HTML
- La syntaxe du langage
- Les variables
- Les types de données
- Les expressions et les opérateurs
- Les structures de contrôle
- Les variables relatives au web

# Pourquoi utiliser PHP ?

- Contenu dynamique vs. contenu statique
- Applications web
- Script côté serveur, libre et opensource
  - Diminution du trafic réseau
  - Élimination des problèmes de compatibilité
  - Amélioration de la sécurité en comparaison aux scripts s'exécutant sur le client

# PHP et HTML

- Incorporation du code PHP dans une page HTML
  - Exemple
- Inclusion de fichiers : `include "nomdefichier";`
  - Exemple
- Interaction avec l'utilisateur (formulaires)
  - Exemple

# Syntaxe du langage

- Les noms de variables :
  - sensibles à la casse,
  - débutent par un \$ suivi d'un \_ ou d'une lettre,
  - constitués d'une séquence alphanumérique et de \_.
- Des noms de fonctions insensibles à la casse
- Les espaces sont ignorés
  - Espacer le code pour le rendre plus lisible.
- Les commentaires /\*...\*/, // et #

# Les types de données

- Trois types primitifs
  - entiers entre  $-2^{31}$  et  $2^{31}$
  - flottants (éventuellement en notation scientifique)
  - chaînes de caractères (délimitées par des " ou des ')
- Trois types composés
  - tableaux
    - indexés numériquement ou par des clés
    - index facultatif pour un ajout en fin de tableau
  - objets

# Les valeurs booléennes

- Mots-clés : *true* et *false*
- PHP évalue la valeur booléenne des données :
  - *false* si la variable entière vaut 0, *true* sinon.
  - *false* si la chaîne de caractères est vide, *true* sinon.
  - *false* si le tableau est vide, *true* sinon.
  - *false* si l'objet n'a pas aucune variable ou fonction définie, *true* sinon.

# Les variables

- Les variables sont non typées.
- Conversion de type
  - (int),
  - (float), (double), (real),
  - (string),
  - (array),
  - (object).
- `gettype()`, `is_long()`, `is_double()`, `is_string()`, `is_array()` et `is_object()`.
- `intval()`, `doubleval()`, `strval()`.

# Variables spéciales

## ■ Variables variables

- Des variables dont le nom est variable
- Exemple : `$field = "IDProduit"; $$field = "432BB";`
- Confusion possible

## ■ Variables statiques

- définies dans une fonction avec le mot-clé *static*
- leur valeur est persistante d'un appel à l'autre

# La portée des variables

```
$poste = "a";  
function change_pos() {  
    $poste = "b";  
}  
change_pos();  
echo ("$poste");  
// imprime a
```

```
$poste = "a";  
function change_pos() {  
    global $poste;  
    $poste = "b";  
}  
change_pos();  
echo ("$poste");  
// imprime b
```

# Les constantes

## ■ Définition

- `define("JAUNE" , "#FFFF00");`

## ■ Vérification

- `defined()`

- Exemple :

```
if(defined("JAUNE")){  
    echo("<BODY BGCOLOR=" . JAUNE . ">\n");  
}
```

# Expressions et opérateurs

## ■ Les expressions

- 5
- 5+5
- \$a
- \$a==5
- Sqrt(9)

## ■ Les opérateurs

- similaires à ceux de Java ou C

# Les opérateurs

- Opérateurs arithmétiques
  - +, -, \*, /, %
- Opérateurs de comparaison
  - <, >, <=, >=, ==, !=, <>
- Opérateurs logiques
  - &&, and, ||, or, xor, !
- Opérateur de concaténation de chaînes
  - .

# Les opérateurs (2)

- Opérateurs sur les bits
  - $\&$ ,  $|$ ,  $\wedge$ ,  $\gg$ ,  $\ll$ ,  $\sim$
- Opérateur d'affectation
  - $=$
- Opérateurs de variables
  - $\$$ ,  $\&$
- Opérateur d'objet
  - $->$
- Opérateur de suppression d'erreur
  - $@$

# Les structures de contrôle

```
if(expression) {  
    instructions  
}  
elseif(expression) {  
    instructions  
}  
else {  
    instructions  
}
```

```
if(expression):  
    instructions  
elseif(expression):  
    instructions  
else:  
    instructions  
endif;
```

# Les structures de contrôle (2)

```
switch(expression) {  
    case expression:  
        instructions  
        break;  
    default:  
        instructions  
        break;  
}
```

```
switch(expression):  
    case expression:  
        instructions  
        break;  
    default:  
        instructions  
        break;  
endswitch;
```

# Les boucles

```
while(expression) {  
    instructions  
}
```

```
while(expression):  
    instructions  
endwhile;
```

```
do {  
    instructions  
} while (expression)
```

```
for(init; cond; itérative) {  
    instructions  
}
```

```
for(init; cond; itérative):  
    instructions  
endfor;
```

# Les fonctions

- Fonctions internes du langage vs. fonctions externes
- Les fonctions exécutent des instructions et/ou retournent un résultat
- Exemple :

```
function triple($x) {  
    $x = $x * 3;  
    return $x;  
}
```
- Passage d'arguments :
  - Par valeur : `$triplevar = triple($var)`; (le contenu de `$var` est inchangé)
  - Par référence : `triple(&$var)`; (le contenu de `$var` est triplé)

# Fonctions imbriquées et récursives

- Imbrication de fonctions
  - Pour des raisons d'organisation du code
  - Les fonctions imbriquées sont accessibles de partout

- Fonctions récursives

- Exemple :

```
function puissance($base, $exp) {  
    if($exp) {  
        return $base * puissance ($base, $exp -1);  
    }  
    return 1;  
}
```

# Affectation d'une fonction à une variable

- Lorsque le choix de la fonction dépend du contexte
- Passage d'arguments entre parenthèses

```
switch($type_nav) {  
    case "NN" :  
        $fonction_charge = "load_nn";  
        break;  
    default :  
        $fonction_charge = "load_generic";  
}  
$fonction_charge($URL);
```

# Les variables relatives au web

- Création automatique de variables pour les données reçues par HTTP
- Accès aux variables d'environnement du serveur
  - `<? phpinfo() ?>`
- Gestionnaire de formulaire générique
  - Tableau `$HTTP_GET_VARS`
  - Tableau `$HTTP_POST_VARS`
  - Tableau `$HTTP_COOKIE_VARS`

# Les tableaux

- Peuvent contenir des éléments de différents types
- Fonctionnent comme une table de hachage
- Initialisation
  - `$pays[] = "fr"; $pays[] = "de"; $pays[] = "us";`
  - `$pays[0] = "fr"; $pays[1] = "de"; $pays[2] = "us";`
  - `$pays[50] = "fr"; $pays[20] = "de"; $pays[10] = "us";`
  - `$pays[] = array("fr", "de", "us");`
  - `$pays[] = array(1=>"fr", "de", "us");`
- Peuvent être multi-dimensionnels

# Parcourir un tableau

- Tableaux indexés séquentiellement
  - Usage de la fonction `count()` et d'une boucle
- Tableaux indexés non séquentiellement
  - Usage de `key()` et `current()`
  - Usage de `each()` et `list()` (méthode préférable)
  - Usage de `next()` et `prev()`
- Tableaux indexés par une chaîne de caractères
  - `while(list($cle, $val) = each($pays)) {echo "...";};`

# Fonctions de tri

- `sort($pays)` et `rsort($pays)`
- `asort($pays)` et `arsort($pays)` pour les indices non numériques
- `ksort($pays)` et `krsort($pays)` tri sur les clés
- `usort($pays, $comparateur)`, `uasort(...)` et `uksort(...)`
- Fonctions de mélange : `shuffle()`

# Références complémentaires

- <http://www.php.net/manual/fr/>
- <http://www.commentcamarche.net/php/phpintro.php3>
- <http://www.phpdebutant.org/>