PHP – Année 2010 / 2011
Licence MIW – IUT de Gap

2ème Séance

Rappels sur la syntaxe
Conventions de nommage

Rappels

- Les Types
- Les Variables, Références et Constantes
- Les Opérateurs
- Les Structures de contrôle
- Les Fonctions
- Fonctions essentielles

Rappels – Les Types

- Attention PHP → typage faible
- 8 types en PHP
  - Scalaires
    - Booléen (bool)
    - Entier (int)
    - Nombre a virgule flottante (float)
    - Chaine de caractère (string)

Rappels – Les Types

- Composés
  - Tableau (array)
  - Objet (object)
- Spéciaux
  - Ressource (ressource)
  - NULL

Rappels – Types - Bool

- Le Booléen
  - 2 états → 2 valeurs
    - true / TRUE
    - false / FALSE
    - $ma_var = true;
    - Beaucoup de fonction et structures de contrôle retournent un booléen

Rappels – Type - Int

- Nombre « entier », négatif ou positif, en base décimale, octale, héxadécimale
  - $ma_var = 15; //décimal
  - $ma_var = -36; //décimal négatif
  - $ma_var = 0x1A; //Hexadécimal (=26 décimal)

Rappels – Types - Float

- Nombres a virgule flottante
- Attention a la précision lors des calculs !
- Séparateur décimal : le point
  - $ma_var = 3.1415926535;

Rappels – Types - String

- Chaîne de caractère
- Délimitée par " ou ' ou syntaxe Heredoc / Nowdoc
  - $ma_var = 'Chaine de $car !';
  - $ma_var = "Chaine de $car !"; //$car sera interpreté
  - $ma_var = <<<MONTEXTE
  - Chaine de car
  - Chaine de $car
  - MONTEXTE;

Rappels – Types - Array

- Type « composé »
- Peut contenir « tout » autre type
- Pas de déclaration explicite de la taille

```
$ma_var = array();
$ma_var = array('1', 'bille', array(3, 'hibou'));
$ma_var[] = '8';
$ma_var['chou'] = 'hibou';
$ma_var = array('genou' => 'bijou');
```

Rappel – Types - Object

- Représente une instance d'une classe
  - Déclaration par : new ClassName();
  - Détails lors des séances sur la POO

Rappels – Types - Autres

- « Resource » → contient un « lien » vers une ressource externe. Par exemple une connexion à une BDD.

- NULL → valeur non connue (Inconnue ou Absence de valeur)

Rappels - Variables

- Chaque variable contient une valeur d'un type défini.
    - Le type peut changer en cours de script (implicitement ou par transtypage)
        - Transtypage se fait en faisant précéder la valeur à transtyper par :

(int), (bool), (float), (string), (array) …

    - Une variable se déclare par '$' suivi de '_' ou une lettre, puis caractères alphanumériques ou '_'.

Rappels - Variables

- Portée des variables
  - Locale (à l'intérieur d'une fonction)
  - « Script » (hors fonction)
  - Globale → préfixer la variables avec *global* à l'intérieur d'une fonction pour faire référence à la variable « générale » du script.

Rappels - Variables

- Dans une fonction, à chaque appel, la variable est re-définie. Mais il est possible de « conserver » la valeur d'une variable entre 2 appels.
    - Pour cela, utiliser *static* pour préfixer la variable

Rappels - Variables

- Certains noms de variables sont réservés.
- Attention si register_global est activé, certaines variables supplémentaires sont également réservées.

- Nom de variables réservés non abordés :

$php_errormsg, $argc, $argv, $HTTP_RAW_POST_DATA, $http_response_header

Rappels - Variables

- Variables "superglobales"
  - Accessibles en tout point du script
  - Les anciennes prédéfinies (ex : $HTTP_POST_VARS), ne sont pas superglobales. Leur usage est déprécié.
  - De la forme $_GET/POST/...
  - Tableaux de valeurs $_GET['clé'] = '…';
  - $_GET = array('clé1' => 'valeur1', 'clé2' => '…');

Rappels - Variables

- Les différentes superglobales
  - $_SERVER //variables de serveur et d'éxécution
  - $_GET //variables de l'url appelée (HTTP GET)
  - $_POST //variables passée par la méthode POST de HTTP
  - $_FILES //fichier uploadés par HTTP
  - $_REQUEST //contient valeurs de $_GET,$_POST,$_COOKIE
  - $_SESSION //variables contenues dans le fichier de session
  - $_ENV //variables de l'environnement d'éxécution
  - $_COOKIE //variables contenues dans le cookie

Rappels - Variables

- Les variables dynamiques → variables dont le **nom est variable**
- $a = 'miw';
- $$a = 'En cours !';
- echo $miw; //affiche En cours !
- Idem avec echo ${$a}

Rappels - Références

- La référence n'est pas un pointeur !
- La référence se déclare par '&'
  - $a = 3;
  - $b = &$a;
  - $b = 5;
  - echo $a; //affiche 5
- Les paramètres de fonctions peuvent dans la plupart des cas êtres passés par référence au lieu de déclarer une variable globale
  - function maFonction (&$ma_var) {
    - // traitements
  - }

Rappels - Constantes

- Valeur qui n'est pas amenée à changer au cours du script. ET qui ne peut l'être.
- Se définit par fonction define('NOM', 'valeur');
- Par convention nom en MAJ
- Nom débutant par lettre ou '_', puis '_' et alphanumérique
- echo MA_CONSTANTE; //affiche la constante

Rappels - Constantes

- A l'image des variables pré-définies, il existe des constantes "magiques". Préfixé et suffixé de 2 '_'.
- __LINE__, __FILE__, __DIR__, __FUNCTION__, __CLASS__, __METHOD__, __NAMESPACE__

Rappels - Opérateurs

- Les différents types d'opérateurs
  - Arithmétiques
  - Assignation
  - Sur les bits
  - Comparaison
  - Contrôle d'erreur
  - D'exécution
  - Incrémentation / Décrémentation
  - Logique
  - De chaînes
  - De tableaux
  - De types

Rappels - Opérateurs

- Arithmétiques
  - * //multiplication
  - / //division
  - % //modulo (reste de la division entière)
  - + //addition
  - - //soustraction ou négation

Rappels - Opérateurs

- Assignation
  - = //pour assigner une valeur à une variable
  - => //pour assigner une valeur à une clé de tableau

Rappels - Opérateurs

- Opérations sur les bits – Pour information
  - & //And
  - | //Or
  - ^ //Xor
  - ~ //Not
  - << //décalage à gauche
  - >> //décalage à droite

Rappels - Opérateurs

- Opérateurs de comparaison (renvoi de bool)
  - == //test de l'égalité
  - === //test de l'égalité et de même type
  - != et <> //différent de …
  - !== //différent ou de type différent
  - < //inférieur à
  - > //supérieur à
  - <= //inférieur ou égal à
  - >= //supérieur ou égal à

Rappels - Opérateurs

- Opérateur de contrôle d'erreur
  - Pour faire taire une erreur de fonction.
  - Attention a son usage, aucune erreur n'est affichée en cas de problème sur la fonction préfixée
  - utilisation en préfixant la fonction avec @
  - Par exemple @unlink('../mon_fichier'); n'affichera pas d'erreur si le fichier n'existe pas

Rappels – Opérateurs

- Incrémentation et décrémentation
  - ++$a; //incrémente puis retourne $a
  - $a++ //retourne $a puis incrémente
  - similaire, mais en décrément pour --$a et $a--
  - Ex : $a = 5;
    - echo $a++; //affiche 5
    - echo $a; //affiche 6
    - echo ++$a; //affiche 7

Rappels - Opérateurs

- Opérateurs logiques, similaires à ceux sur les bits
  - 'and' et '&&' //And
  - 'or' et '||' //Or
  - 'xor' //Xor (ou exclusif)
  - ! //Not
- Utilisation dans les conditions

Rappels - Opérateurs

- Opérateur de chaîne. Utilisation pour la concaténation
  - $ma_var = 'bonjour' . ' le monde';
  - echo $ma_var; //affiche bonjour le monde
  - $ma_var .= ' !';
  - echo $ma_var; //affiche bonjour le monde !

Rappels - Opérateurs

- Opérateurs sur les tableaux
  - Pour la plupart similaires au opérateurs de comparaison
  - == //mêmes paires clés / valeurs
  - === //mêmes paires, dans le même ordre, et même type
  - != et <> //non égalité → contraire de ==
  - !== //contraire de ===
  - Un autre opérateur : opérateur d'union
  - + //ajoute les éléments d'un tableau à l'autre, sans écraser les clés. Donc attention à l'ordre des opérandes.

Rappels – Structures de contrôle

- De différents types
  - Instructions de langage (break, include, require ...)
  - Tests et conditions
  - Boucles

Rappels – Structures de contrôle

- Les tests et conditions
  - if
  - if / else
  - if / elseif/else if / else
  - "Ternaire"

Rappels – Structures de contrôle

- Le "If" (et les autres)
  - if(ma_condition) { //sous entendu est TRUE
    - //traitement
  - }
  - Syntaxe alternative :
    - if($ma_var == '28') :
      - //traitement
    - endif;

Rappels – Structures de contrôle

- Il est possible de faire un "if" sur une ligne
  - if($truc != $chose) $ma_var = false;
- Mais attention à la lisibilité et à l'ajout d'instructions !
- Pour compléter le if, on peut utiliser "else" et les "else if" et "elseif"

Rappels – Structures de contrôle

- if($ma_var == true) {
  - //traitement
- } else { //ou else : en syntaxe alternative
  - //traitement
- }
- if($ma_var == false) {
  - //traitement
- } elseif (ma_condition_2) { //ou else if
  - //traitement
- } else { ....

Rappels – Structures de contrôle

- Le switch
  - Eviter d'enchainer des else if …
  - switch($ma_var) { //ou switch($ma_var) :
    - case 'miaou' :
      - echo 'C\'est un chat !';
      - break;
    - case 'ouaf' :
      - echo 'C\'est un chien !';
      - break;
    - default :
      - echo 'Animal inconnu !';
      - break;
  - } // ou endswicth en alternatif

Rappels – Structures de contrôle

- While → tant que la condition est vrai alors …
- $a = 1;
- while($a < 10) { //ou while(...) :
  - echo $a++ //affichera 1, puis 2 … 9
- } //ou endwhile; en alternatif
- do … while → même principe, sauf que la condition est testée aprés, donc toujours au moins une itération

Rappels – Structures de contrôle

- La boucle "for" → pour la variable allant de ... à ..., par pas de ..., faire …
- for($i = 1; $i < 11; $i++) { //ou for(...) :
  - if($i % 2 == 0) {
    - echo $i; //affiche donc les chiffres pairs de 2 à 10
  - }
- } //ou endfor; si syntaxe alternative

Rappels – Structures de contrôle

- La boucle foreach → pour parcourir en entier un tableau
- foreach($mon_tableau as $valeur) { //ou :
  - ....
- } //ou endforeach;
- foreach($mon_tableau as $cle => $valeur) { ...

Rappels - Fonctions

- Une fonction permet de factoriser le code, et de s'en servir plusieurs fois, de manière récursive, dans plusieurs lieux, et de commencer a bâtir des briques.
- Déclaration par le mot clé 'function', et retourne une valeur par 'return'
- function maFonction($arguments) {
  - //traitements
  - return $resultat;
- }
- puis utilisation :
  - echo maFonction('28'); //affichera le retour

Rappels – Fonctions récursives

- C'est une fonction qui s'appelle elle-même. Sert lorsque l'on ne connait pas d'avance le nombre d'itération. (menu multi-niveau …)
- Ex : un compte a rebours
- function reverseCount($start) {
  - if($start > 0) {
    - echo $start;
    - reverseCount($start – 1);
  - } else {
    - return;
  - }
- }

Rappels – Fonctions essentielles

- Fonctions is_... (isset, is_numeric, …)
  - Pour tester l'existence ou l'appartenance d'une variable/constante à un type
- Manipulations de tableau avec les array_...
- Manipulations de chaînes avec les str_...

Conv. nommage et présentation

- Des conventions pour :
  - Produire un code standard, compréhensible
    - Travail en équipe facilité
    - Relecture aisée
  - Par exemple la convention « Zend » :
    - http://framework.zend.com/manual/fr/coding-standard.coding-style.html
    - http://framework.zend.com/manual/fr/coding-standard.naming-conventions.html