PHP 5 – Structures de base

Fonctions – Fonctions Chaînes

PHP 5

Les fonctions

Structures de contrôle

- Les fonctions utilisateur
  - Déclaration

```
<?php
    function Nom_de_la-fonction($argument1, $argument2,
    …){
    //liste d'instructions
}
?>
```

  - Valeur par défaut

```
<?php
    function Nom_de_la-fonction($argument1='valeur_par_defaut'){
    //liste d'instructions
}
?>
```

  - Valeur de retour
    - La fonction peut renvoyer une valeur grâce au mot-clé : return
    - Une fonction peut contenir plusieurs instructions de retour, mais l'exécution s'arrêtera à la première mise en oeuvre

Structures de contrôle

- Les fonctions utilisateur
  - Exemple : fonction-return.php

```
<?php
function dire_texte($qui, $texte='Bonjour'){
    if(empty($qui)){
     return FALSE;
    }else{
        echo "$texte $qui";
        return TRUE;
    }
}
```

Structures de contrôle

- Appel
  - Nom_de_la_fonction(argument1, argument2, …)
- Exemple :

```
<?php
function dire_texte($qui, $texte='Bonjour'){
    if(empty($qui)){
     return FALSE;
    }else{
        echo "$texte $qui";
        return TRUE;
    }
}
dire_texte('cher phpeur', 'bienvenue');
//Utilisation de la valeur par défaut
dire_texte('cher phpeur');
?>
```

Structures de contrôle

- Appel
  - On peut aussi contrôler le retour

```
<?php
function dire_texte($qui, $texte='Bonjour'){
    if(empty($qui)){
     return FALSE;
    }else{
        echo "$texte $qui";
        return TRUE;
    }
}
if (dire_texte(""){
    echo "Erreur";
};
if (!dire_texte("cher phpeur")
//Affiche "Bonjour cher phpeur"
?>
```

Structures de contrôle

- Les fonctions utilisateur
  - Visibilité des variables
    - Les variables déclarées dans une fonction ne sont utilisables que dans celles-ci
    - Inversement, les variables déclarées dans votre script ne seront pas accessible dans une fonction : les deux espaces sont complètement indépendants
  - Exemple

```
<?php
    $param=3;
    function decremente($valeur){
            $valeur=$valeur-1;
            echo $param; //n'affiche rien
    }
}
decremente($param);
echo $param; //affiche 3
?>
```

Structures de contrôle

- Les fonctions utilisateur
  - Passage de paramètres par recopie
    - Par défaut, PHP fait un passage par recopie
    - La valeur utilisée par la fonction n'est donc pas celle donnée en argument mais une copie
      - Si vous la modifiez à l'intérieur de la fonction, cela n'aura pas d'influence dehors
  - Exemple

```
<?php
        function ajouter_cinq($nombre)
        {
            $nombre += 5; //équivalent de $nombre = $nombre + 5
            return $nombre;
        }

        $mon_entier = 15;
        echo ajouter_cinq($mon_entier); //affichera 20

        echo $mon_entier; //affichera 15
?>
```

Structures de contrôle

- Les fonctions utilisateur
  - Passage de paramètres par référence
    - On fait référence à la variable dans le programme appelant et tout ce qu'on fait sur la variable est reoportée au niveau du programme appelant
    - Pour cela, il faut accompagner le paramètre d'appel de "&"
  - Exemple

```
<?php
        function ajouter_cinq($nombre)
        {
            $nombre += 5; //équivalent de $nombre = $nombre + 5
            return $nombre;
        }

        $mon_entier = 15;
        echo ajouter_cinq(&$mon_entier); //affichera 20

        echo $mon_entier; //affichera 20
?>
```

Structures de contrôle

- Passage par référence (suite)
  - L'avantage de ce type d'opération est que vous travaillez directement sur la variable d'origine, il n'y a pas de recopie et donc les performances peuvent être meilleures
  - Vous n'avez d'ailleurs plus forcément besoin de retourner une valeur
  - Prenons cet exemple qui fait exactement la même chose que le précédent :

```
<?php
        function ajouter_cinq($nombre)
        {
            $nombre += 5; //équivalent de $nombre = $nombre + 5
        }

        $mon_entier = 15;
        ajouter_cinq(&$mon_entier);

        echo $mon_entier; //affichera 20
?>
```

PHP 5

Les chaînes

Chaînes de caractères

- Type String : formes-string.php
  - Une chaîne de caractères peut s'écrire de diverses manières en PHP, chacune utilisant un "délimiteur" bien précis :

```
<?php
//Délimitation par des guillemets :
echo "Hello World!";

//Délimitation par des apostrophes :
echo 'Hello World!';

//Délimitation par la syntaxe HereDoc :
$string = <<<END
Hello World!
END;
echo $string;

//Délimitation par la syntaxe NowDocs :
$string = <<<'END'
Hello World!
END;
echo $string;

//Caractère $ avec la syntaxe HereDoc :
$string = <<<END
Le signe \$ doit être échapé : \$var
END;
echo $string;

//Caractère $ avec la syntaxe NowDocs :
$string = <<<'END'
Le signe $ peut être utilisé : $var
END;
echo $string;
```

Les deux premières formes sont les plus communes. La 3° (HereDoc) est très largement moins utilisée à cause de sa complexité, ce qui est dommage car elle offre certains avantages. La 4° (NowDocs) est encore en discussion pour PHP 5.3.

Chaînes de caractères

- Les guillemets/apostrophes : guillemets.php
  - La syntaxe des guillemets permet d'utiliser sans crainte les apostrophes, mais tout se complique dès que l'on souhaite utiliser des guillemets :

    ```
    <?php echo "Voici un exemple d'apostrophe";
    echo "Voici un exemple de \"guillemets\"";
    ```

  - La syntaxe des apostrophes permet d'utiliser des guillemets dans le texte, mais nous ennuie avec les apostrophes :
    - `echo 'Voici un exemple de "guillemets"';`
    - `echo 'Voici un exemple d\'apostrophe';`

Chaînes de caractères

- Les guillemets/apostrophes
  - Si l'on souhaite pouvoir utiliser à la fois des guillemets et des apostrophes dans un même texte, plusieurs solutions s'offrent à nous :
    - Par échappement

```
<?php
echo "Voici un exemple d'apostrophe suivi de \"guillemets\"";
```

    - Par échappement

```
<?php
echo ‘Voici un exemple d'apostrophe suivi de \"guillemets\"';
```

    - HereDoc :

```
<?php
echo <<<EOT
Voici un exemple d'apostrophe suivi de "guillemets"
EOT; //pour écrire le « ; »
```

    - Par concaténation

```
echo "Voici un exemple d'apostrophe" . ' suivi de
    "guillemets"',"</br>\n";
```

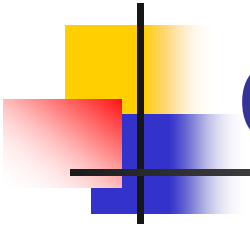
Chaînes de caractères

- Scanner une chaîne de caractères : sscanf.php
  - sscanf() permet de récupérer les variables à partir d'une chaîne de caractères
  - Exemple 1

```
<?php
// get author info and generate DocBook entry
$auth = "24\tLewis Carroll";
$n = sscanf($auth, "%d\t%s %s", $id, $first, $last);
echo "<author id='$id'>
    <firstname>$first</firstname>
    <surname>$last</surname>
</author>\n";
?>
```

  - Ceci donne :

```
Lewis Carroll
```



Chaînes de caractères

- Fonctions sur les chaînes
  - Nous allons donner dans la suite quelques exemples de fonctions sur les chaînes
  - Toutes les fonctions sur les chaînes se trouvent à l'adresse suivante :
    - http://www.w3schools.com/PHP/php_ref_string.asp

Traitement des chaînes de caractères

- Exemple 2 : sscanf2.php

```
<?php
// Lecture d'un numéro de série
list($serial) = sscanf("SN/2350001", "SN/%d");
// et la date de fabrication
$mandate = "January 01 2000";
//list() permet d’assigner des valeurs à plusieurs variables
list($month, $day, $year) = sscanf($mandate, "%s %d %d");
echo "Le produit $serial a été fabriqué le : $year-" .
     substr($month, 0, 3) . "-$day\n";
?>
```

- Ceci donne :

```
Le produit 2350001 a été fabriqué le : 2000-Jan-1
```

Traitement des chaînes de caractères

- Accéder à un caractère précis

```
<?php
    $texte = ‘PHP';
    echo $texte[1]; //Affiche H
?>
```

- Valeur ASCII d’un caractère

```
<?php
    echo ord(‘a’); //renvoie 97
    echo chr(97); //renvoie a
?>
```

- Longueur d’une chaîne

```
<?php
    $livre = ‘PHP 5 avancé’;
    echo strlen($livre); //renvoie 12
?>
```

Traitement des chaînes de caractères

- Calculer le nombre de mots d'une chaîne

```
<?php
    $livre = 'PHP 5 avancé';
    echo str_word_count($livre); //affiche 3
?>
```

- Lister les mots d'une chaîne
  - En ajoutant un argument à str_word_count(), elle envoie la liste des mots dans un tableau

```
<?php
    $livre = 'PHP 5 avancé';
    $tab= str_word_count($livre, 1);
    Var_dump($tab) //affiche : [0]=> "PHP" [1]=> 5 "avancé" [2]=> avancé
?>
```

Traitement des chaînes de caractères

- Comptage de mots
  - Autre exemple

```
<?php
        print_r(str_word_count("Hello world!",1));
?>
```

  - donne

```
Array
        (
        [0] => Hello
        [1] => world
        )
```

  - Autre exemple

```
<?php
        print_r(str_word_count("Hello world!",2));
?>
```

  - donne

```
(
        [0] => Hello
        [6] => world
)
```

Traitement des chaînes de caractères

- Comptage de mots : str_word_count.php
  - Autre exemple

```
<?php
        print_r(str_word_count("Hello world & good morning!",1));
        print_r(str_word_count("Hello world & good morning!",1,"&"));
?>
```

  - donne

```
Array
        (
        [0] => Hello
        [1] => world
        [2] => good
        [3] => morning
        )
Array
        (
        [0] => Hello
        [1] => world
        [2] => &
        [3] => good
        [4] => morning
        )
```

Traitement des chaînes de caractères

- Position d'une sous-chaîne :
  - strpos(chaine, sous-chaine)
- Présence de caractères dans une chaîne
  - strspn()
    - retourne la longueur de la première sous chaîne trouvée contenant uniquement la première sous chaîne trouvée
  - strcspn()
    - fait l'opération inverse : retourne la longueur de la première sous chaîne ne contenant aucun des caractères
  - Exemple

```
<?php
    $livre = 'chaîne à vérifier';
    $masque ="'";
    if(strcspn($chaine, $masque) == strlen($chaine)){
        echo 'il n y a pas d\'apostrophes';
    }else {
     echo 'il y a des apostrophes';
    }
?>
```

Traitement des chaînes de caractères

- Protections
  - addslashes() : protège certains caractères, comme les guillemets, les apostrophes et barres obliques, en les préfixant automatiquement
  - addcslashes() : convertit les fins de lignes et les retours chariot, ainsi que les caractères dont le code ASCII est inférieur à 32 ou supérieur à 126

```php
<?php
    $texte = "texte\n\r\"\'texte";
    //Affiche texte et "'texte, sur deux lignes
    echo $texte;

    //affiche texte et \"\'texte, sur deux lignes
    echo addslashes($texte);

    //affiche texte\n\r\"\'texte, sur une ligne
    echo addcslashes($texte, "\"\n\r");
?>
```

Traitement des chaînes de caractères

- Protections pour HTML
  - Quand on envoie des balises à l'impression, les caractères sont interprétés
  - Pour éviter cette interprétation, il faut en convertir les caractères spéciaux (<,> et &) en entités (<, >, &) HTML équivalents
  - La fonction htmlspecialchars() permet d'effectuer cette conversion

Traitement des chaînes de caractères

- Exemple

```php
<?php
    $texte = "valeur avec & <br> et avec \" et '";
    //Ne convertit rien, tout est interprété
    echo $texte, "<br>\n";

    //Convertit les caractères &, >, < et "
    echo htmlspecialchars($texte), "<br>\n";
    echo htmlspecialchars($texte, ENT_COMPAT),  "<br>\n ";

    //Convertit les caractères &, >, <, " et '
    echo htmlspecialchars($texte, ENT_QUOTES),  "<br>\n ";

    //Convertit les caractères &, > et < uniquement
    echo htmlspecialchars($texte, ENT_NOQUOTES),  "<br>\n ";
?>
```

Traitement des chaînes de caractères

- Affichage d'une fin de ligne
  - nl2br() : permet d'interpréter le « \n » en un retour chariot
  - Exemple

```
<?php
    $texte = "valeur avec \n et ";
    echo nl2br($texte);
?>
```

  - Affiche

```
valeur avec
et
```

Traitement des chaînes de caractères

- Manipulations sur les chaînes
  - Recherche une sous chaîne : strstr()

```
<?php
    strstr('eric.daspet@dreams4net.com', '@');
    //renvoie  @dreams4net.com
?>
```

  - Récupère une sous chaîne : substr()

```
<?php
    $texte = 'PHP 5 avancé';
    echo substr($texte, 6, 2) //renvoie  av
?>
```

Traitement des chaînes de caractères

- Manipulations sur les chaînes
  - Remplacer un motif : str_replace()

```
<?php
    $texte = ‘PHP 5 avancé’;
    $cherche = ‘4’;
    $remplace = ‘5’;
    echo str_replace($cherche, $remplace, $texte) //renvoie PHP 5
?>
```

  - Remplissage : str_pad

```
<?php
    echo str_pad(‘PHP’, 10) //complète jusqu’à 10 caractères avec des espaces
?>
```

Traitement des chaînes de caractères

- addcslashes.php
  - Accompagne d'un \

```
<?php
    $str = "Hello, my name is Kai Jim.";
    echo $str."<br />";
    echo addcslashes($str,'m')."<br />";
    echo addcslashes($str,'K')."<br />";
?>
```

  - Retourne

```
Hello, my name is Kai Jim.
Hello, \my na\me is Kai Ji\m.
Hello, my name is \Kai Jim.
```

Traitement des chaînes de caractères

- addcslashes.php
  - Ajoute un \ à un ensemble de caractères

```
<?php
    $str = "Hello, my name is Kai Jim.";
    echo $str."<br />";
    echo addcslashes($str,'A..Z')."<br />";
    echo addcslashes($str,'a..z')."<br />";
    echo addcslashes($str,'a..h');
?>
```

  - Retourne

```
Hello, my name is Kai Jim.
\Hello, my name is \Kai \Jim.
H\e\l\l\o, \m\y \n\a\m\e \i\s K\a\i J\i\m.
H\ello, my n\am\e is K\ai Jim.
```

Traitement des chaînes de caractères

- Répétition

```
<?php
    echo str_repeat(".",13);
?>
```

  - donne

```
.............
```

- Comptage de mots

```
<?php
    echo str_word_count("Hello world!");
?>
```

  - donne
    - 2

Traitement des chaînes de caractères

- Comparaison par strcasecmp()
  - Retourne
    - 0 : en cas d'égalité
    - <0 si str1 < str2
    - >0 : si str1 > str2
  - Exemple

```
<?php
    echo strcasecmp("Hello world!","HELLO WORLD!");
?>
```

  - retourne

```
0
```