

Développement PHP

Partie 2 : Bases de PHP

DENIS LOEUILLET – IFA - 2017

Partie 2 : Bases de PHP

- Variables, types de variables
- Les instructions conditionnelles
- Les boucles
- les types de données composés : tableaux et objets
- Les fonctions
- Les constantes

Partie 2 : Bases de PHP

Les variables

- Éléments indispensables dans tout langage de programmation
- Permettent de retenir temporairement des informations en mémoire
- Qu'est-ce qu'une variable ?
 - Petite information stockée en mémoire temporairement
 - Durée de vie courte
 - En PHP, la variable existe tant que la page est en cours de génération
 - Dès que la page PHP est générée, toutes les variables sont supprimées de la mémoire
 - Ce n'est pas un fichier qui est stocké sur le disque mais une petite information temporaire présente en mémoire vive
 - Les variables sont créées par le développeur
 - On peut en créer autant qu'on veut

Partie 2 : Bases de PHP

Les variables : un nom et une valeur

- Une variable est toujours constituée de 2 éléments :
 - Son nom : pour la reconnaître
 - Sa valeur : c'est l'information qu'elle contient, et qui peut changer
- On peut modifier sa valeur quand on veut, faire des opérations dessus, et, quand on en a besoin on peut l'appeler
- Indispensables pour un site PHP pour mémoriser des informations

NB :

- Il est de bon ton de ne coder qu'en anglais : le code que vous écrivez pourrait être relu / repris par n'importe qui, et il est possible que le (la) développeur(se) soit non francophone.
- Pour les noms de vos variables, prenez soin de ne les écrire qu'en anglais.

Partie 2 : Bases de PHP

Les types de données

- Les variables peuvent stocker différents types d'informations
- on parle de **type de données**
- PHP propose :
 - 4 types de données scalaires (ne pouvant contenir qu'une valeur)
 - 2 types composés (pouvant contenir plusieurs valeurs)
 - Et 2 types spéciaux
- Voici les principaux types de données à connaître

Partie 2 : Bases de PHP

Les types de données : les types scalaires

- Les type de données à connaître :
 - Les chaînes de caractères (string) :
 - ❖ <http://php.net/manual/fr/language.types.string.php>
 - ❖ C'est le nom informatique que l'on donne au texte
 - ❖ En PHP ce type de données s'appelle : string
 - ❖ On peut stocker des textes courts comme très longs
 - ❖ Permet de stocker toute séquence de caractères sur un octet (code ASCII compris entre 0 et 255), sans limite de taille
 - ❖ Exemple :

"Je suis un texte". Une chaîne de caractères est habituellement écrite entre guillemets ou entre apostrophes (on parle de guillemets simples) : 'Je suis un texte'. Les deux fonctionnent mais il y a une petite différence que l'on va découvrir plus loin.

Partie 2 : Bases de PHP

Les types de données : les types scalaires

- Les type de données à connaître :
 - Le type entier (integer ou int) :
 - ❖ <http://php.net/manual/fr/language.types.integer.php>
 - ❖ Ce sont les nombre de type 1, 2, 3, 4...
 - ❖ Mais également les entiers relatifs : -1, -2, -3, -4...
 - ❖ Le type entier (integer) permet de stocker un nombre entier signé sur 32 bits, soit des valeurs comprises entre -2 147 483 648 et +2 147 483 647.
 - ❖ Exemple : 42

Partie 2 : Bases de PHP

Les types de données : les types scalaires

- Les type de données à connaître :
 - Le type nombre à virgule flottante (float) :
 - ❖ <http://php.net/manual/fr/language.types.float.php>
 - ❖ Ce sont les nombre à virgule, comme 14.738
 - ❖ On peut stocker de nombreux chiffres après la virgule
 - ❖ Les nombres doivent être écrits avec un point à la place de la virgule (notation anglaise)
 - ❖ Le type nombre à virgule flottante (float) permet de stocker un nombre décimal sur une plage de valeurs dépendante de la plate-forme (généralement de l'ordre de 10^{-308} à 10^{+308}).
 - ❖ Exemple : 14.738

Partie 2 : Bases de PHP

Les types de données : les types scalaires

- Les type de données à connaître :
 - Le type booléen (bool) :
 - ❖ Ref : <http://php.net/manual/fr/language.types.boolean.php>
 - ❖ Type très important qui permet de stoker la valeur vrai (true) ou faux (false)
 - ❖ Le type booléen (boolean) peut prendre deux valeurs : TRUE (ou true) et FALSE (ou false).
 - ❖ Exemple : true

Partie 2 : Bases de PHP




Les types de données : les types spéciaux

- Les type de données à connaître :
 - Le type NULL :
 - ❖ <http://php.net/manual/fr/language.types.null.php>
 - ❖ Le « type » NULL est un peu particulier et correspond au type d'une variable utilisée sans jamais avoir été initialisée.
 - ❖ Une variable avec la valeur NULL n'a pas de valeur
 - ❖ Une variable est considérée comme NULL si
 - Elle a eu d'assigner la constante NULL
 - Elle n'a été définie à aucune valeur
 - Elle a été détruite par la fonction unset
 - ❖ Syntaxe :
`$var = NULL;`

Partie 2 : Bases de PHP

Les types de données

- Résumé :

Type de données	Exemple de valeur
string	"Du texte"
int	42
float	14.738
bool	true  false 
NULL	

Partie 2 : Bases de PHP

Affecter une valeur à une variable

- Premières manipulations de variables :

```
1 <?php
2 $age_du_visiteur = 17;
3 ?>
```

Qu'est-ce qu'on vient de faire ?



Avec ce code PHP on vient de créer une variable

- Son nom : \$age_du_visiteur
- Sa valeur : 17

NB :

- On ne peut pas mettre d'espace dans un nom de variable. À la place, utilisez un underscore «_»
- évitez aussi les accents, les cédilles et tout autre symbole : PHP ne les apprécie pas trop...

Partie 2 : Bases de PHP

Affecter une valeur à une variable

- Premières manipulations de variables :

```
1 <?php
2 $age_du_visiteur = 17;
3 ?>
```

Syntaxe :

- ❖ D'abord, on écrit le symbole « dollar » (\$) : il précède toujours le nom d'une variable, ça permet de dire à PHP « J'utilise une variable ». Vous reconnaîtrez toujours qu'il y a une variable par la présence du symbole « dollar » (\$).
- ❖ Ensuite, il y a le signe « égal » (=) : pour dire que \$age_du_visiteur est égal à...
- ❖ À la suite, il y a la valeur de la variable, ici 17.
- ❖ Enfin, il y a l'incontournable point-virgule (;) qui permet de terminer l'instruction.

Qu'est-ce que ce code va afficher?

Partie 2 : Bases de PHP

Affecter une valeur à une variable

- Que se passe-t-il dans le cas suivant ?

```
1 <?php
2 $age_du_visiteur = 17; // La variable est créée et vaut 17
3 $age_du_visiteur = 23; // La variable est modifiée et vaut 23
4 $age_du_visiteur = 55; // La variable est modifiée et vaut 55
5 ?>
```

- La variable \$age_du_visiteur va être créée
- Elle va prendre pour valeur 17 puis 23 puis 55
- Rien ne s'affiche
- Dans la mémoire de l'ordinateur, une zone nommée age_du_visiteur vient de prendre la valeur 17 puis 23 puis 55

Partie 2 : Bases de PHP

Affecter une valeur à une variable

- Le type string (chaîne de caractères)
 - Ce type permet de stocker du texte : il faut entourer votre texte avec des guillemets ou des apostrophes

Quelle est la différence entre ces 2 exemples ?

```
1 <?php
2 $nom_du_visiteur = "Mateo21";
3 $nom_du_visiteur = 'Mateo21';
4 ?>
```

Partie 2 : Bases de PHP

Affecter une valeur à une variable

- Le type string (chaîne de caractères)

D'autres exemples :

```
1 <?php
2 $variable = "Mon \"nom\" est Mateo21";
3 $variable = 'Je m\'appelle Mateo21';
4 ?>
```

```
1 <?php
2 $variable = 'Mon "nom" est Mateo21';
3 $variable = "Je m'appelle Mateo21";
4 ?>
```

Partie 2 : Bases de PHP

Affecter une valeur à une variable

- Le type int (nombre entier)
 - Il suffit tout simplement d'écrire le nombre que vous voulez stocker, sans guillemets.

```
1 <?php
2 $age_du_visiteur = 17;
3 ?>
```

Partie 2 : Bases de PHP

Utiliser les types de données

- Le type float (nombre décimal)
 - Il faut écrire votre nombre avec un point au lieu d'une virgule. C'est la notation anglaise .

```
1 <?php
2 $poids = 57.3;
3 ?>
```


Partie 2 : Bases de PHP

Utiliser les types de données

- Le type bool (booléen)
 - Pour dire si une variable vaut vrai ou faux, vous devez écrire le mot true ou false sans guillemets autour (ce n'est pas une chaîne de caractères !).
 - Il faut bien choisir le nom de votre variable pour que l'on comprenne ce que ça signifie

```
1 <?php
2 $je_suis_un_zero = true;
3 $je_suis_bon_en_php = false;
4 ?>
```

Partie 2 : Bases de PHP

Utiliser les types de données

- Une variable vide avec NULL
 - Si vous voulez créer une variable qui ne contient rien, vous devez lui passer le mot-clé NULL ou null.
 - Cela sert simplement à indiquer que la variable ne contient rien, tout du moins pour le moment.

```
1 <?php
2 $pas_de_valeur = NULL;
3 ?>
```

Partie 2 : Bases de PHP

Afficher et concaténer des variables

Nous avons appris :

- à créer des variables
- à stocker des informations à l'intérieur.

Mais pour le moment, aucun de nos codes source n'affiche quoi que ce soit.

Partie 2 : Bases de PHP

Afficher et concaténer des variables

- Afficher le contenu d'une variable
 - « echo » peut aussi servir pour afficher la valeur d'une variable :
<http://php.net/manual/fr/function.echo.php>

```
1 <?php
2 $age_du_visiteur = 17;
3 echo $age_du_visiteur;
4 ?>
```

- Il suffit d'écrire le nom de la variable que vous voulez afficher
- On ne met pas de guillemets autour du nom de la variable

Partie 2 : Bases de PHP

Afficher et concaténer des variables

- La concaténation (assemblage) :
<http://php.net/manual/fr/language.operators.string.php>
 - Comment faire, par exemple, pour écrire : « Le visiteur à 17 ans » : 17 étant la valeur de la variable `$age_du_visiteur`
 - 1^{ère} solution

```
1 <?php
2 $age_du_visiteur = 17;
3 echo "Le visiteur a ";
4 echo $age_du_visiteur;
5 echo " ans";
6 ?>
```


Partie 2 : Bases de PHP

Afficher et concaténer des variables

- La concaténation (assemblage)
 - Il y a une autre solution qui s'écrit sur une seule ligne
 - 2 méthodes pour cette solution
 - ✓ Une méthode qui utilise les guillemets
 - ✓ Une méthode qui utilise les apostrophes

Partie 2 : Bases de PHP

Afficher et concaténer des variables

- La concaténation (assemblage)
 - Concaténer avec des guillemets
 - ✓ C'est plus simple
 - ✓ On peut écrire le nom de la variable au milieu des guillemets
 - ✓ Le nom de la variable sera remplacé par sa valeur

```
1 <?php
2 $age_du_visiteur = 17;
3 echo "Le visiteur a $age_du_visiteur ans";
4 ?>
```

- ✓ Avec des guillemets les variables à l'intérieure sont remplacées par leur valeur

Partie 2 : Bases de PHP

Afficher et concaténer des variables

- La concaténation (assemblage)
 - Concaténer avec des apostrophes
 - ✓ Écrivez le même code avec des apostrophes

```
1 <?php
2 $age_du_visiteur = 17;
3 echo 'Le visiteur a $age_du_visiteur ans'; // Ne marche pas
4 ?>
```

- ✓ Avec des apostrophes les variables à l'intérieure ne sont pas remplacées par leur valeur
- ✓ Il faut séparer les variables et le texte

```
1 <?php
2 $age_du_visiteur = 17;
3 echo 'Le visiteur a ' . $age_du_visiteur . ' ans';
4 ?>
```

Partie 2 : Bases de PHP

Afficher et concaténer des variables

- La concaténation (assemblage)
 - Concaténer avec des apostrophes
 - ✓ Ça à l'air plus compliqué
 - ✓ Mais code plus lisible
 - ✓ On repère plus facilement les variables
 - ✓ L'éditeur de texte devrait colorer la variable
 - ✓ Méthode plus rapide car PHP n'a pas besoin de chercher les variables au milieu du texte

Partie 2 : Bases de PHP

Faire des calculs simples

- Les opérations de base : addition, soustraction...
<http://php.net/manual/fr/language.operators.arithmetic.php>
 - Les signes à connaître pour faire les 4 opérations de base

Symbole	Signification
+	Addition
-	Soustraction
*	Multiplication
/	Division

Partie 2 : Bases de PHP

Faire des calculs simples

- Les opérations de base : addition, soustraction...
 - Quelques exemples

```
1 <?php
2 $nombre = 2 + 4; // $nombre prend la valeur 6
3 $nombre = 5 - 1; // $nombre prend la valeur 4
4 $nombre = 3 * 5; // $nombre prend la valeur 15
5 $nombre = 10 / 2; // $nombre prend la valeur 5
6
7 // Allez on rajoute un peu de difficulté
8 $nombre = 3 * 5 + 1; // $nombre prend la valeur 16
9 $nombre = (1 + 2) * 2; // $nombre prend la valeur 6
10 ?>
```

Partie 2 : Bases de PHP

Faire des calculs simples

- Les opérations de base : addition, soustraction...

- Quelques exemples

```
1 <?php
2 $nombre = 10;
3 $resultat = ($nombre + 5) * $nombre; // $resultat prend la valeur 150
4 ?>
```

- La précedence des opérateurs :

La priorité des opérateurs spécifie l'ordre dans lequel les valeurs doivent être analysées.

<http://php.net/manual/fr/language.operators.precedence.php>

Partie 2 : Bases de PHP

Faire des calculs simples

- Le modulo
<http://php.net/manual/fr/language.operators.arithmetic.php>
 - Le modulo renvoie le reste d'une division entière
 - Symbole : %

```
1 <?php
2 $nombre = 10 % 5; // $nombre prend la valeur 0 car la division tombe juste
3 $nombre = 10 % 3; // $nombre prend la valeur 1 car il reste 1
4 ?>
```

Partie 2 : Bases de PHP

Faire des calculs simples

- Les autres opérations
<http://php.net/manual/fr/book.math.php>

Les opérations plus complexes peuvent être réalisées en PHP mais il faudra passer par ce qu'on appelle des fonctions, une notion que l'on découvrira plus tard. Les opérations basiques que l'on vient de voir sont amplement suffisantes pour la programmation PHP de tous les jours.

Partie 2 : Bases de PHP

Les variables : résumé

- Une variable est une petite information qui reste stockée en mémoire le temps de la génération de la page PHP. Elle a un nom et une valeur.
- Plusieurs types de variables pour stocker différents types d'informations :
 - du texte (string)
 - des nombres entiers (int)
 - des nombres décimaux (float)
 - des booléens pour stocker vrai ou faux (bool), etc.
- En PHP, un nom de variable commence par le symbole \$
- La valeur d'une variable peut être affichée avec l'instruction « echo ».
- Il est possible de faire des calculs mathématiques entre plusieurs variables : addition, soustraction, multiplication...

Partie 2 : Bases de PHP

Les instructions conditionnelles

A quoi servent les conditions ?



Afficher des choses différentes en fonctions de certaines conditions

Exemple :

- Dire « Bonjour » au visiteur si c'est le matin et « Bonsoir » si c'est le soir

Dans PHP, il existe plusieurs structures conditionnelles

- La structure de base : if ... else
- Une alternative pratique : switch
- Les ternaires : des conditions condensées

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

1. Les symboles à connaître : retenir quelques symboles de comparaison
2. La structure else ... if : comment fonctionne une condition else ... if
3. Le cas des booléens : une utilisation particulière
4. Des conditions multiples : on peut utiliser plusieurs conditions à la fois
5. Petite astuce
6. Alternative à l'instruction if...else
7. L'opérateur ternaire

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

1. Les symboles à connaître

Symbole	Signification
==	Est égal à
>	Est supérieur à
<	Est inférieur à
>=	Est supérieur ou égal à
<=	Est inférieur ou égal à
!=	Est différent de

- Ne pas confondre = et ==

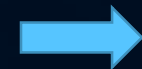
➤ = : symbole d'affectation

➤ == : symbole de test d'égalité

- On peut aussi ajouter :

➤ ===

➤ !==



compare également les types des variables

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

2. La structure else ... if

```
1 <?php
2 $age = 8;
3
4 if ($age <= 12)
5 {
6     echo "Salut gamin !";
7 }
8 ?>
```

- On commence par le mot « if »
- Entre parenthèses la condition
- Entre accolades les instructions à exécuter si la condition est remplie

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

2. La structure else ... if

```
1 <?php
2 $age = 8;
3
4 if ($age <= 12) // SI l'âge est inférieur ou égal à 12
5 {
6     echo "Salut gamin ! Bienvenue sur mon site !<br />";
7     $autorisation_entrer = "Oui";
8 }
9 else // SINON
10 {
11     echo "Ceci est un site pour enfants, vous êtes trop vieux pour pouvoir entrer. Au revoir
    !<br />";
12     $autorisation_entrer = "Non";
13 }
14
15 echo "Avez-vous l'autorisation d'entrer ? La réponse est : $autorisation_entrer";
16 ?>
```

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

2. La structure else ... if

```
1 <?php
2 if ($autorisation_entrer == "Oui") // SI on a l'autorisation d'entrer
3 {
4     // instructions à exécuter quand on est autorisé à entrer
5 }
6 elseif ($autorisation_entrer == "Non") // SINON SI on n'a pas l'autorisation d'entrer
7 {
8     // instructions à exécuter quand on n'est pas autorisé à entrer
9 }
10 else // SINON (la variable ne contient ni Oui ni Non, on ne peut pas agir)
11 {
12     echo "Euh, je ne connais pas ton âge, tu peux me le rappeler s'il te plaît ?";
13 }
14 ?>
```

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

2. La structure else ... if

Dans l'ordre, PHP rencontre les conditions suivantes :

- Si \$autorisation_entrer est égale à « Oui », tu exécutes ces instructions...
- sinon si \$autorisation_entrer est égale à « Non », tu exécutes ces autres instructions...
- sinon, tu redemandes l'âge pour savoir si on a ou non l'autorisation d'entrer.

Remarque :

Au départ, une variable ne contient rien.

Sa valeur est vide, on dit qu'elle vaut NULL, c'est-à-dire rien du tout.

Pour vérifier si la variable est vide, vous pouvez taper :

if (\$variable == NULL)...

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

3. Le cas des booléens

Les booléens sont des variables qui valent soit « true » soit « false »

Ils sont très bien adaptés pour les structures conditionnelles

```
1 <?php
2 if ($autorisation_entrer == true)
3 {
4     echo "Bienvenue petit nouveau. :o)";
5 }
6 elseif ($autorisation_entrer == false)
7 {
8     echo "T'as pas le droit d'entrer !";
9 }
10 ?>
```

Remarque : pas de guillemets autour de « true » et « false »

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

3. Le cas des booléens

```
1 <?php
2 if ($autorisation_entrer)
3 {
4     echo "Bienvenue petit nouveau. :o)";
5 }
6 else
7 {
8     echo "T'as pas le droit d'entrer !";
9 }
10 ?>
```

- PHP comprend qu'il faut vérifier si \$autorisation_entrer vaut true
- Avantages :
 - Plus rapide à écrire
 - Ça se comprend mieux

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

3. Le cas des booléens

```
1 <?php
2 if (! $autorisation_entrer)
3 {
4
5 }
6 ?>
```

- Le symbole « ! » permet de vérifier si la variable vaut false
- C'est pareil que d'écrire : if (\$autorisation_entrer == false)
- La méthode courte est plus lisible

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.logical.php>

4. Les conditions multiples

➡ permet de poser plusieurs conditions à la fois

➡ pour cela, on a besoin de nouveaux mots-clés

Mot-clé	Signification	Symbole équivalent
AND	Et	&&
OR	Ou	

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.logical.php>

4. Les conditions multiples

```
1 <?php
2 if ($age <= 12 AND $langue == "français")
3 {
4     echo "Bienvenue sur mon site !";
5 }
6 elseif ($age <= 12 AND $langue == "anglais")
7 {
8     echo "Welcome to my website!";
9 }
10 ?>
```

```
1 <?php
2 if ($sexe == "fille" OR $sexe == "garçon")
3 {
4     echo "Salut Terrien !";
5 }
6 else
7 {
8     echo "Euh, si t'es ni une fille ni un garçon, t'es quoi alors ?";
9 }
10 ?>
```

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.logical.php>

5. Petite astuce

```
1 <?php
2 if ($variable == 23)
3 {
4     echo '<strong>Bravo !</strong> Vous avez trouvé le nombre mystère !';
5 }
6 ?>
```

```
1 <?php
2 if ($variable == 23)
3 {
4     ?>
5 <strong>Bravo !</strong> Vous avez trouvé le nombre mystère !
6 <?php
7 }
8 ?>
```


Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/control-structures.switch.php>

6. Alternative à if...else : switch

- Les structures à base de if... elseif ... else suffisent pour traiter n'importe quelle condition
- Dans certains cas on peut vouloir utiliser « switch »

```

1 <?php
2 if ($note == 0)
3 {
4     echo "Tu es vraiment un gros nul !!!";
5 }
6
7 elseif ($note == 5)
8 {
9     echo "Tu es très mauvais";
10 }
11
12 elseif ($note == 7)
13 {
14     echo "Tu es mauvais";
15 }
16
17 elseif ($note == 10)
18 {
19     echo "Tu as pile poil la moyenne, c'est un peu juste...";
20 }
21
22 elseif ($note == 12)
23 {
24     echo "Tu es assez bon";
25 }
26
27 elseif ($note == 16)
28 {
29     echo "Tu te débrouilles très bien !";
30 }
31
32 elseif ($note == 20)
33 {
34     echo "Excellent travail, c'est parfait !";
35 }
36
37 else
38 {
39     echo "Désolé, je n'ai pas de message à afficher pour cette note";
40 }
41 ?>

```

```

1 <?php
2 $note = 10;
3
4 switch ($note) // on indique sur quelle variable on travaille
5 {
6     case 0: // dans le cas où $note vaut 0
7         echo "Tu es vraiment un gros nul !!!";
8         break;
9
10    case 5: // dans le cas où $note vaut 5
11        echo "Tu es très mauvais";
12        break;
13
14    case 7: // dans le cas où $note vaut 7
15        echo "Tu es mauvais";
16        break;
17
18    case 10: // etc. etc.
19        echo "Tu as pile poil la moyenne, c'est un peu juste...";
20        break;
21
22    case 12:
23        echo "Tu es assez bon";
24        break;
25
26    case 16:
27        echo "Tu te débrouilles très bien !";
28        break;
29
30    case 20:
31        echo "Excellent travail, c'est parfait !";
32        break;
33
34    default:
35        echo "Désolé, je n'ai pas de message à afficher pour cette note";
36 }
37 ?>


```

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/control-structures.switch.php>

6. Alternative à if...else : switch

- Il y a beaucoup moins d'accolades : uniquement au début et à la fin de « switch »
- Case signifie : cas
- On indique au début sur quelle variable on travaille : switch (\$note)
- On dit à PHP : je vais analyser la valeur de \$note
- Ensuite on utilise des « case » pour analyser chaque cas : case 0, case 10
Cela signifie : Dans le cas où la valeur est 0 ... Dans le cas où la valeur est 10
- Avantage : on n'a plus besoin de mettre le double égal : ==
- Défaut : ça ne marche pas avec les autres symboles : < > <= >= !=
 on ne peut tester que l'égalité
- « default » à la fin est l'équivalent du « else » : cas par défaut
- Ne pas oublier le mot clé « break » à la fin de chaque « case » car sinon PHP continue à analyser tous les cas :
 - « break » demande à PHP de sortir du switch
 - PHP sort des accolades et ne lit pas les autres « case »

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/control-structures.switch.php>

6. Alternative à if...else : switch

- Quand choisir « if », et quand choisir « switch »?



C'est surtout une question de présentation et de clarté

- Pour une condition simple et courte : « if »
- Pour une série de conditions à analyser : « switch » pour rendre le code plus clair

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

6. L'opérateur ternaire

- C'est une condition condensée qui fait 2 chose sur une seule ligne
 - Il teste la valeur d'une variable
 - Il affecte une valeur à une variable

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

6. L'opérateur ternaire

```
1 <?php
2 $age = 24;
3
4 if ($age >= 18)
5 {
6     $majeur = true;
7 }
8 else
9 {
10    $majeur = false;
11 }
12 ?>
```

on peut faire la même chose en une seule ligne :

```
1 <?php
2 $age = 24;
3
4 $majeur = ($age >= 18) ? true : false;
5 ?>
```

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

6. L'opérateur ternaire

```
1 <?php
2 $age = 24;
3
4 $majeur = ($age >= 18) ? true : false;
5 ?>
```

- Le test précédent est fait en une seule ligne
 - La condition testée est `$age >= 18`
 - Si vrai : la valeur indiquée après le « ? » sera affectée à la variable `$majeur`
 - Sinon : la valeur qui suit le symbole « : » sera affectée à la variable `$majeur`

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else

<http://php.net/manual/fr/language.operators.comparison.php>

6. L'opérateur ternaire

```
1 <?php
2 $age = 24;
3
4 $majeur = ($age >= 18) ? true : false;
5 ?>
```

- Le test précédent est fait en une seule ligne
 - La condition testée est `$age >= 18`
 - Si vrai : la valeur indiquée après le « ? » sera affectée à la variable `$majeur`
 - Sinon : la valeur qui suit le symbole « : » sera affectée à la variable `$majeur`

Partie 2 : Bases de PHP

Les instructions conditionnelles : if ... else : résumé

- Les conditions permettent à PHP de prendre des décisions en fonction de la valeur des variables.
- La forme de condition la plus courante est if...elseif...else qui signifie « si »... « sinon si »... « sinon ».
- On peut combiner des conditions avec les mots-clés AND (« et ») et OR (« ou »).
- Si une condition comporte de nombreux elseif, il peut être plus pratique d'utiliser switch, une autre forme de condition.
- Les ternaires sont des conditions condensées qui font un test sur une variable, et en fonction des résultats de ce test donnent une valeur à une autre variable. Elles sont cependant plus rarement utilisées.