

Les types de données, les variables et les constantes

Table des matières

I. Contexte	3
II. Les variables PHP	3
III. Exercice : Appliquez la notion	5
IV. Les types de données	5
V. Exercice : Appliquez la notion	8
VI. Les opérations simples	8
VII. Exercice : Appliquez la notion	12
VIII. Les constantes en PHP	13
IX. Exercice : Appliquez la notion	14
X. Auto-évaluation	15
A. Exercice final	15
B. Exercice : Défi	17
Solutions des exercices	17

I. Contexte

Durée : 1 h

Environnement de travail : Repl.it

Pré-requis : Connaître les bases de PHP et HTML

Contexte

Une application PHP requiert des données de différents types (textes, nombres, etc.), qui seront stockées dans des variables pour pouvoir être manipulées. Ce cours aborde ces différents concepts afin de donner les bases du développement PHP.

II. Les variables PHP

Objectifs

- Rappeler ce qu'est une variable
- Apprendre à déclarer une variable PHP et à lui affecter une valeur
- Afficher une variable dans une page HTML

Mise en situation

Déclarer une variable PHP permet de réaliser différentes actions, comme effectuer des calculs, afficher du texte ou stocker une liste de valeurs.

Rappel Qu'est-ce qu'une variable ?

Une variable est représentée par un identifiant associé à une valeur.

Les variables peuvent changer de valeur en fonction des traitements effectués (calculs, modification de texte, modification de la liste de valeurs, etc.).

Méthode Comment déclarer une variable PHP ?

Pour déclarer une variable PHP, on précède le nom qu'on souhaite lui donner du symbole dollar (\$).

- Les noms des variables sont sensibles à la casse (prise en compte des majuscules/minuscules).
- Ils doivent commencer par une lettre ou le symbole souligné (`_`) et peuvent contenir des lettres, chiffres ou des soulignés.
- Une variable ne peut pas commencer par un chiffre (ex : `$1userName`) ou par un autre caractère spécial (ex : `$/userName`)
- La bonne pratique veut que le nom de la variable soit en anglais et assez descriptif (ex : `$userName` pour définir le nom de l'utilisateur).
- Pour lui affecter une valeur, on fait suivre le nom du symbole égal (=), suivi de la valeur désirée.
- Comme toutes les instructions en PHP, l'affectation doit être clôturée avec un point-virgule (;).

Exemple Déclaration de variables PHP

```
1 <?php
2
3 $userName = 'John'; //La variable appelée "userName" reçoit pour valeur "John"
4 $userAge = 36; //La variable appelée "userAge" reçoit la valeur 36
```

Rappel

« La balise fermante d'un bloc PHP à la fin d'un fichier est optionnelle, et parfois, il est utile de l'omettre. »
php.net¹

Méthode Afficher le contenu des variables dans une page HTML

Pour faire afficher la valeur d'une variable dans une page HTML, on utilise `echo` suivi du nom de la variable.

Exemple

L'exemple ci-dessous affiche John dans notre page HTML.

```
1 <html>
2   <head>
3     <title>Les variables PHP</title>
4   </head>
5   <body>
6     <?php
7       $userName = 'John';
8
9       echo $userName;
10    ?>
11  </body>
12 </html>
```

Syntaxe À retenir

On déclare une variable PHP en précédant son nom du symbole dollar \$.

Le nom de la variable doit :

- commencer par une lettre ou le symbole souligné,
- être en anglais,
- être descriptif de son contenu.

Pour l'afficher dans du code HTML, nous utilisons `echo` suivi du nom de la variable.

Complément

Les variables PHP²

`echo`³

1 <https://www.php.net/basic-syntax.instruction-separation>

2 <https://www.php.net/manual/fr/language.variables.basics.php>

3 <https://www.php.net/manual/fr/function.echo.php>

III. Exercice : Appliquez la notion

Soit le code PHP ci-après :

```
1 <?php
2 $login = 'John';
3 $messageCount = 0;
4 $unreadMessagesCount = 28;
5 $readMessageCount = 12;
6 $messageCount = $readMessageCount + $unreadMessagesCount;
7 echo 'Bonjour '.$login.'" ! Il y a ".$messageCount.'" messages dans votre boîte de réception !';
```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question 1

[solution n°1 p.19]

Exécutez ce code avec repl.it. Qu'affiche-t-il ?

Question 2

[solution n°2 p.19]

En lisant le code, trouvez combien de messages lus et combien de messages non lus possède John, et vérifiez que la somme est correcte.

IV. Les types de données

Objectifs

- Connaître les types de données scalaires
- Savoir les utiliser en PHP

Mise en situation

PHP permet de manipuler plusieurs types de données scalaires :

- Une chaîne de caractères : `string`
- Un nombre entier : `int`
- Un nombre décimal : `float`
- Un booléen vrai/faux (*true/false*) : `bool`

Définition Type scalaire

Un **type scalaire** désigne un type simple, qui ne contient qu'une seule donnée, par exemple un nombre entier.

Remarque

Il existe des types plus complexes et qui peuvent contenir plusieurs valeurs. Les types scalaires sont la base sur laquelle ces types complexes sont construits.

1 <https://repl.it/>

Fondamental

Même si PHP ne l'impose pas, nous vous recommandons d'appliquer une bonne pratique, qui consiste à toujours conserver le même type pour une variable donnée. Par exemple, il ne faut pas réutiliser une variable dans laquelle est stockée une chaîne de caractères pour y stocker ensuite un booléen.

Définition Chaîne de caractères : string

Les *strings* sont des chaînes de caractères. Elles sont représentées par des données mises entre guillemets ou entre apostrophes.

Attention

Il existe une différence entre les guillemets et les apostrophes :

- Le texte entre "guillemets" est d'abord interprété avant d'être affiché, ce qui signifie que toute variable qu'il contient sera remplacée par sa valeur.
- Le texte entre 'apostrophes', lui, est affiché tel quel.

Exemple

```
1 <?php
2     $username = 'John';
3     echo $username.'<br>'; // Affiche "John"
4
5     $hello = "Bienvenue, $username";
6     echo $hello.'<br>'; // Affiche "Bienvenue, John" puisque, grâce aux guillemets, la
variable $username est interprétée
7
8     $hello = 'Bienvenue, $username';
9     echo $hello; // Affiche "Bienvenue, $username" puisque, grâce aux apostrophes, la chaîne
est affichée telle quelle.
```

Attention

Utiliser des guillemets doubles n'est pas considéré comme étant une bonne pratique. Pour l'instant, c'est la seule manière que nous connaissons pour afficher une variable dans une chaîne de caractères, donc c'est celle que nous allons utiliser, mais gardez en mémoire que ce n'est pas la meilleure. En effet, la différence syntaxique entre ces guillemets et les apostrophes est considérée comme trop faible pour la lisibilité du code.

Pour utiliser le caractère délimiteur (apostrophe ou guillemet) à l'intérieur d'une chaîne de caractères, il faut l'**échapper** avec le caractère \ suivi du symbole à échapper :

```
1 <?php
2     echo 'J\'ai un guillemet !<br>'; // Affiche : J'ai un guillemet
3     echo "Il a dit \"J'ai un guillemet\" !"; // Affiche : Il a dit "J'ai un guillemet" !
```

Définition Nombre entier : int

Les *int* sont les nombres entiers, qu'ils soient positifs ou négatifs. Ils permettent d'effectuer des calculs simples.

Exemple

```
1 <?php
2     $min = -100;
3     $max = 150;
4
5     echo $min; // Affiche -100
6     echo '<br>';
7     echo $max; // Affiche 150
```

Définition **Nombre décimal : float**

Les *float* permettent de manipuler des nombres décimaux, c'est-à-dire des nombres à virgule (avec un nombre fini de chiffres après la virgule).

Attention

Le séparateur des décimales n'est pas la virgule, mais le point : PHP utilise la notation décimale anglaise.

Exemple

```
1 <?php
2     $pi = 3.14159265359; // On remarque l'utilisation du point au lieu de la virgule
3     echo $pi;
```

Méthode **Un booléen vrai/faux (true/false) : bool**

Un booléen est une valeur qui est :

- soit vraie (*true*)
- soit fausse (*false*).

Ces mots-clés doivent être écrits sans apostrophes ni guillemets afin de ne pas être considérés comme des chaînes de caractères.

Attention

Un booléen n'est pas fait pour être affiché tel quel.

Utiliser *echo* sur un booléen produit des résultats surprenants :

- n'affiche rien pour la valeur *false*,
- et 1 pour la valeur *true*.

Exemple

```
1 <?php
2     $isLogged = true;
3     echo $isLogged; // Affiche 1
4     echo '<br>';
5     $isLogged = false;
6     echo $isLogged; // N'affiche rien du tout
```

Syntaxe À retenir

Les quatre types scalaires en PHP :

- Une chaîne de caractères : `string` (entre guillemets ou apostrophes)
- Un nombre entier : `int`
- Un nombre décimal : `float` (utiliser le point et non la virgule comme séparateur des décimales)
- Un booléen vrai/faux (`true/false`) : `bool`

On associe chaque variable à un seul type.

Complément

Les types (documentation PHP)¹

V. Exercice : Appliquez la notion

Considérez le code suivant :

```
1 <?php
2 $userName = 'D'Artagnan';
3 echo "Je suis $userName";
```

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question 1

[solution n°3 p.19]

Exécutez-le dans repl.it et observez le retour de la console (l'écran noir en dessous de l'affichage HTML). Que constatez-vous ?

Question 2

[solution n°4 p.19]

Avec ce que nous avons vu dans le cours et en vous aidant de cette erreur, corrigez ce script PHP pour qu'il affiche "Je suis D'Artagnan".

Indice :

"**Syntax Error**" (ou "**erreur de syntaxe**") signifie que le code PHP n'est pas valide. Il y a une erreur quelque part qui fait que le serveur n'arrive pas à exécuter ce code PHP.

Indice :

Il faut faire très attention lorsque l'on utilise certains caractères dans les chaînes de caractères.

VI. Les opérations simples

Objectifs

- Effectuer les opérations mathématiques simples (addition, soustraction, etc.)
- Savoir utiliser l'opérateur de concaténation pour les chaînes de caractères
- Connaître la priorité des opérateurs
- Découvrir les différents opérateurs d'affectation

¹ <https://www.php.net/manual/fr/language.types.php>

² <https://repl.it/>

Mise en situation

Nous allons voir dans ce cours comment effectuer des calculs mathématiques simples, comme des additions ou des concaténations (c'est-à-dire la mise bout à bout de chaînes de caractères), et nous allons étudier les opérateurs d'affectation.

Méthode Opérations sur les nombres

En PHP, nous pouvons effectuer toutes sortes d'opérations simples sur les nombres en utilisant :

- Les signes + et - pour les additions et soustractions.
- Les signes * (étoile) et / (barre oblique ou slash) pour les multiplications et les divisions.

Exemple Additions et soustractions

Dans l'exemple ci-dessous, nous voulons connaître notre solde bancaire afin de savoir s'il est débiteur ou créditeur. Pour cela :

- Nous calculons nos dépenses dans la variable `$expenses`, qui est égale au résultat de l'addition effectuée.
- Puis nous calculons nos recettes dans la variables `$revenues`.
- Nous calculons ensuite le solde en faisant la soustraction "recettes moins dépenses", dans la variable `$bankBalance`.
- Puis avec `echo` nous affichons le résultat calculé, qui vaut donc 48 654.

```
1 <html>
2 <head>
3   <title>Les opérations simples PHP</title>
4 </head>
5 <body>
6   <?php
7     $expenses = 23995 + 4550;
8     $revenues = 76110 + 1089;
9
10    $bankBalance = $revenues - $expenses;
11
12    echo $bankBalance;
13  ?>
14 </body>
15 </html>
```

Exemple Multiplications et divisions

Dans l'exemple ci-dessous, nous voulons calculer le pourcentage de parts que nous possédons sur un bien immobilier.

- Nous initialisons la variable `$totalNumberShares` avec le nombre total de parts.
- Nous créons `$myShares` avec le nombre de parts que nous possédons.
- Nous calculons le pourcentage dans `$percentage` en multipliant nos nombres de parts par 100, puis en divisant par le nombre total de parts.
- Ensuite nous affichons le résultat, qui vaut 39,971803278689 (soit presque 40 % de parts).

```
1 <html>
2 <head>
3   <title>Les opérations simples PHP</title>
4 </head>
5 <body>
```

```

6  <?php
7      $totalNumberShares = 1525;
8      $myShares = 609.57;
9
10     echo $myShares * 100 / $totalNumberShares;
11     ?>
12 </body>
13 </html>

```

Méthode Concaténer des chaînes de caractères

Pour joindre une chaîne de caractères à une autre, nous devons utiliser l'opérateur de concaténation **point** (.).

Exemple

Dans l'exemple ci-dessous, nous avons concaténé la première partie du message avec ' : ' (avec des espaces avant et après afin que les textes ne soient pas collés), et enfin avec le nom d'utilisateur.

```

1 <html>
2 <head>
3     <title>Les opérations simples PHP</title>
4 </head>
5 <body>
6     <?php
7         $hello = 'Bienvenue sur notre site';
8         $userName = 'John';
9
10        $message = $hello . ' : ' . $userName; // Affiche "Bienvenue sur notre site : John"
11
12        echo $message;
13    ?>
14 </body>
15 </html>

```

Méthode Priorité des opérateurs

Comme pour les calculs mathématiques, il y a une priorité par défaut dans les opérateurs (la multiplication est prioritaire sur l'addition, par exemple) et l'utilisation de parenthèses permet de modifier cette priorité.

Exemple

```

1 <html>
2 <head>
3     <title>Les opérations simples PHP</title>
4 </head>
5 <body>
6     <?php
7         $calcul = 10 + 2 * 3; // La multiplication est prioritaire sur l'addition
8         $otherCalcul = (10 + 2) * 3; // Les parenthèses sont prioritaires sur la
9         multiplication
10        echo $calcul; // Affiche 16
11        echo '<br>';
12        echo $otherCalcul; // Affiche 36
13    ?>
14 </body>

```

```
15 </html>
```

Attention Priorités

L'opération de concaténation a la même priorité que l'addition, il faut donc être vigilant en concaténant des opérations directement.

```
1 <?php
2
3     echo 'Mon frère a '. 30-1 .' ans'; // Affiche 'Mon frère a 29 ans'
4     echo 'Mon frère a '.(30-1).' ans'; // Affiche 'Mon frère a 29 ans', les parenthèses
    permettent de faire la soustraction en premier.
5     echo 'Mon père a '. 30*2 .' ans'; // Affiche 'Mon père a 60 ans', puisque la
    multiplication est prioritaire ici.
```

Attention Utiliser les espaces autour du caractère de concaténation

Étant donné que le point est également le délimiteur des nombres flottants et qu'il est possible d'écrire .1 pour 0.1, alors il ne faut surtout pas oublier l'espace entre la chaîne de caractères et le nombre.

```
1 <?php
2     echo 'Né en '.1980; // Provoque une erreur puisque PHP interprète 'Né en '.1980 comme une
    string placée à côté du nombre flottant 0.1980, ce qui est une erreur de syntaxe.
```

Conseil

D'une manière générale, il est fortement conseillé de stocker le résultat des calculs dans une variable puis de la concaténer, plutôt que de faire les calculs en même temps que la concaténation.

Méthode Les différents opérateurs d'affectation

En PHP, il existe différents opérateurs permettant d'affecter une valeur à une variable.

Le plus simple est le signe égal (=), qui fait une simple affectation de valeur à une variable.

Mais il y a en d'autres qui sont une combinaison d'un symbole (+, -, *, /, etc.) avec le signe égal (=) et qui permettent de manipuler la variable de façon plus synthétique.

- `$myVar += $otherVar` : affecte à `$myVar` le résultat de : `$myVar + $otherVar`. C'est l'équivalent de `$myVar = $myVar + $otherVar`
- `$myVar -= $otherVar` : effectue la même chose, mais avec une soustraction : `$myVar - $otherVar`.
- `$myVar *= $otherVar` : équivaut à : `$myVar * $otherVar`.
- `$myVar /= $otherVar` : équivaut à : `$myVar / $otherVar`.
- `$myVar .= $otherVar` : équivaut à une concaténation des variables : `$myVar . $otherVar`.

Il existe également deux autres opérateurs pour ajouter ou enlever 1 rapidement à une variable : `$myVar++` et `$myVar--` (qui équivalent respectivement à `$myVar += 1` et `$myVar -= 1`).

Exemple

```
1 <html>
2   <head>
3     <title>Les opérations simples PHP</title>
4   </head>
5   <body>
6     <?php
7       $x = 10;
```

```

8     $y = 15;
9
10    $x += $y; //est similaire à $x = $x + $y
11
12    echo $x; // Affiche 25 (x = 10 + 15)
13    echo '<br>';
14
15    $hello = 'Bienvenue';
16    $userName = 'John';
17
18    $hello .= ' : ' . $userName;
19
20    echo $hello; // Bienvenue : John ($hello . ' : ' . $userName)
21    ?>
22 </body>
23 </html>

```

Syntaxe À retenir

- Les opérateurs arithmétiques : addition et soustraction (+ et -) ; multiplication et division (* et /).
- Les opérateurs de chaînes : concaténation de chaînes de caractères à l'aide du point (.).
- La priorité des opérateurs : la multiplication et la division prioritaires sur addition, soustraction et concaténation. Possibilité d'utiliser des parenthèses pour définir une priorité.
- Les opérateurs d'affectation : permettent d'affecter directement une valeur, le résultat d'une opération, une concaténation de chaînes de caractères, etc.

Complément

Les opérateurs arithmétiques (documentation PHP)¹

Opérateurs de chaînes²

La priorité des opérateurs³

Les opérateurs d'affectation⁴

Exercice : Appliquez la notion

[solution n°5 p.19]

Exercice

Pour chaque morceau de code ci-dessous, essayez de deviner le résultat affiché, puis jouez-le dans repl.it pour vous corriger. Saisissez la réponse affichée par repl.it pour confirmer la bonne exécution du code.

```

1 <?php
2     $amount = 50 + 2 * 3;
3     $amount /= 2;
4     echo $amount;

```

Exercice

1 <https://www.php.net/manual/fr/language.operators.arithmetic.php>

2 <https://www.php.net/manual/fr/language.operators.string.php>

3 <https://www.php.net/manual/fr/language.operators.precedence.php>

4 <https://www.php.net/manual/fr/language.operators.assignment.php>

```
1 <?php
2     $userName = 'John';
3     $greeting = 'Bonjour';
4     $sentence = $greeting;
5     $sentence .= $userName;
6     echo $sentence;
```

Exercice

Qu'affiche le code ci-dessous ?

```
1 <?php
2     $counter = 5;
3     $counter++;
4     $counter++;
5     $counter++;
6     $counter++;
7     $counter++;
8     $counter++;
9     $counter *= 2;
10    echo $counter;
```

VIII. Les constantes en PHP

Objectif

- Apprendre ce qu'est une constante, quand et comment l'utiliser

Mise en situation

Les constantes permettent de stocker des valeurs en les fixant de manière à ce qu'elles ne puissent jamais être modifiées.

Un taux peut en être un exemple, si nous ne voulons pas que la valeur de ce taux puisse être modifiée et si nous voulons réutiliser exactement la même valeur à plusieurs endroits de notre code.

Méthode Déclarer une constante avec `define`

```
1 define('NAME', value);
```

Pour déclarer une constante, il faut utiliser la fonction `define()`, au sein de laquelle nous précisons en premier son nom (en majuscules), puis ensuite sa valeur.

Fondamental

Déclarer une constante est différent que déclarer une variable : pas de symbole dollar (\$) ici, que ce soit pour la créer ou pour l'appeler.

Exemple

Dans l'exemple ci-dessous, nous avons créé une constante contenant notre taux bancaire `BANK_RATE` avec la valeur `2.05`.

Nous l'utilisons ensuite pour l'afficher, puis pour effectuer un calcul dont on affiche le résultat.

```

1 <html>
2   <head>
3     <title>Les constantes PHP</title>
4   </head>
5   <body>
6     <?php
7       define('BANK_RATE', 2.05);
8
9       echo BANK_RATE; // 2.05
10      echo '<br>';
11
12      $amount = 10000;
13      $result = $amount * BANK_RATE / 100;
14      echo $result; // 205
15    ?>
16  </body>
17 </html>

```

Fondamental

Une constante ne peut pas être redéfinie.

Exemple

```

1 <?php
2   define('BANK_RATE', 2.05);
3   define('BANK_RATE', 3.05);
4
5   echo BANK_RATE; // 2.05

```

Ce code retourne un avertissement et ignore la seconde définition de `BANK_RATE`.

```

1 PHP Notice: Constant BANK_RATE already defined in
  /home/runner/JovialHatefulMicrokernel/main.php on line 3
2 2.05

```

Syntaxe À retenir

- Une constante est définie avec la fonction `define`.
- Sa valeur ne peut pas être modifiée.

Complément

Les constantes¹

Exercice : Appliquez la notion

[solution n°6 p.20]

Exercice

¹ <https://www.php.net/manual/fr/language.constants.php>

Le code ci-dessous est valide en PHP.

```
1 <?php
2     $pi = 3.14;
3     $circleRadius = 8;
4     $pi = 6;
5     $circleArea = $pi * $circleRadius * $circleRadius;
6     echo $circleArea;
```

- ☐ Vrai
- ☐ Faux

Exercice

Le code ci-dessous est valide en PHP.

```
1 <?php
2     define('PI', 3.14);
3     $circleRadius = 8;
4     PI = 6;
5     $circleArea = PI * $circleRadius * $circleRadius;
6     echo $circleArea;
```

- ☐ Vrai
- ☐ Faux

X. Auto-évaluation

A. Exercice final

Exercice 1

[solution n°7 p.21]

Exercice

La variable `$maVariable` est une variable valide en PHP.

- ☐ Vrai
- ☐ Faux

Exercice

La variable `$maVariable` respecte les bonnes pratiques de développement.

- ☐ Vrai
- ☐ Faux

Exercice

La variable `$_maVariable` est une variable valide en PHP.

- ☐ Vrai
- ☐ Faux

Exercice

La variable `$1variable` est une variable valide en PHP.

- ☐ Vrai
- ☐ Faux

Exercice

La variable `$-variable` est une variable valide en PHP.

- ☐ Vrai
- ☐ Faux

Exercice

Qu'est-ce que le code suivant affiche ?

```
1 <?php
2
3 $clientName = 'Henry';
4 $clientName = 'Robert';
5
6 echo $clientName;
```

- ☐ Henry
- ☐ Robert

Exercice 8

[solution n°8 p.22]

Exercice

Quels sont les types scalaires de PHP ?

- ☐ string
- ☐ variable
- ☐ bool
- ☐ PHP
- ☐ int
- ☐ float
- ☐ type
- ☐ array
- ☐ \$

Exercice

Le code suivant est valide en PHP :

```
1 <?php
2 $userName = 'John';
3 $userAge = 30;
4 $userIsConnected = true;
5 $userBankAmount = 1324,20;
```

- ☐ Vrai
- ☐ Faux

Exercice

Qu'est-ce que le code suivant affiche ?

```
1 <?php
2     $userAge = 30;
3     $userName = 'John';
4     echo 'Je suis $userName et je suis né il y a $userAge ans !';
```

Exercice

Qu'est-ce que le code suivant affiche ?

```
1 <?php
2     $userAge = 30;
3     $userName = 'John';
4     echo "Je suis $userAge et je suis né il y a $userAge ans !";
```

B. Exercice : Défi

Pour ce défi, vous allez écrire un script de conversion de monnaie.

Pour réaliser cet exercice, vous aurez besoin de travailler sur l'environnement de travail :



Question

[solution n°9 p.23]

Sachant qu'aujourd'hui un euro vaut :

- 1,08 dollar américain
- 120,34 yens
- 0.00016 bitcoins

Codez un script qui va afficher la valeur d'une variable donnée en euros dans ces trois monnaies.

La solution doit apparaître sous cette forme (ici un exemple avec 3,12 €) :

« 3.12 euros = 3.3696 dollars, 375.4608 yens et 0.0004992 bitcoins. »

Indice :

Pour convertir une monnaie en une autre, il suffit de multiplier la quantité donnée par le taux de change de la monnaie cible.

Indice :

Attention : les taux de change ne doivent pas pouvoir être modifiés dans le code, sinon gare aux escroqueries !

Solutions des exercices

1 <https://repl.it/>

p. 5 Solution n°1

Bonjour John ! Il y a 40 messages dans votre boîte de réception !

p. 5 Solution n°2

Messages non lus (variable `unreadMessagesCount`) : 28

Messages lus (`readMessageCount`) : 12

Total = 40

p. 8 Solution n°3

Il y a une erreur qui s'affiche en rouge :

```
[Wed Apr 1 16:40:19 2020] 172.18.0.1:45962 [500]: / - syntax error, unexpected
'Artagnan' (T_STRING) in /home/runner/MonthlyImprobableComputationalscience/inde
x.php on line 2
```

p. 8 Solution n°4

L'apostrophe de "D'Artagnan" ferme prématurément la chaîne de caractères ! Il y a deux réponses possibles pour résoudre le problème. Soit on échappe le caractère qui pose problème :

```
1 <?php
2     $userName = 'D\'Artagnan';
3     echo "Je suis $userName";
```

Soit on utilise les guillemets pour définir la chaîne de caractères :


```
1 <?php
2     $userName = "D'Artagnan";
3     echo "Je suis $userName";
```

Exercice p. 12 Solution n°5**Exercice**

Pour chaque morceau de code ci-dessous, essayez de deviner le résultat affiché, puis jouez-le dans repl.it pour vous corriger. Saisissez la réponse affichée par repl.it pour confirmer la bonne exécution du code.

```
1 <?php
2     $amount = 50 + 2 * 3;
3     $amount /= 2;
4     echo $amount;
```

28

 La multiplication est prioritaire par rapport à l'addition, donc $50 + 2 * 3 = 56$. On divise ensuite ce nombre par 2, ce qui donne 28.

Exercice

```
1 <?php
2     $userName = 'John';
3     $greeting = 'Bonjour';
4     $sentence = $greeting;
5     $sentence .= $userName;
6     echo $sentence;
```

BonjourJohn

Q \$sentence prend la valeur de \$greeting, qui vaut "Bonjour".

\$sentence .= \$userName; est l'équivalent de \$sentence = \$sentence.\$userName;, ce qui donne "BonjourJohn". On remarque une erreur courante avec la concaténation : l'oubli d'espace ! Faites-y très attention lorsque vous manipulez des chaînes de caractères.

Exercice

Qu'affiche le code ci-dessous ?

```
1 <?php
2     $counter = 5;
3     $counter++;
4     $counter++;
5     $counter++;
6     $counter++;
7     $counter++;
8     $counter++;
9     $counter *= 2;
10    echo $counter;
```

22

Q On incrémente 6 fois la valeur 5, qui donne 11. On la multiplie ensuite par 2 pour faire 22.

Exercice p. 14 Solution n°6

Exercice

Le code ci-dessous est valide en PHP.

```
1 <?php
2     $pi = 3.14;
3     $circleRadius = 8;
4     $pi = 6;
5     $circleArea = $pi * $circleRadius * $circleRadius;
6     echo $circleArea;
```

☒ Vrai

☐ Faux

Q Ce code est malheureusement valide en PHP ! En définissant une variable, \$pi peut être modifiée dans le code sans forcément que l'on s'en rende compte, et cela mène à des résultats incohérents.


Exercice

Le code ci-dessous est valide en PHP.

```
1 <?php
2     define('PI', 3.14);
3     $circleRadius = 8;
4     PI = 6;
5     $circleArea = PI * $circleRadius * $circleRadius;
6     echo $circleArea;
```

☐ Vrai

☒ Faux

 Maintenant, PI est une constante, il n'est plus possible de la modifier.

Exercice p. 15 Solution n°7

Exercice

La variable `$maVariable` est une variable valide en PHP.

☒ Vrai

☐ Faux


 Le nom "maVariable" est complètement valide pour PHP.

Exercice

La variable `$maVariable` respecte les bonnes pratiques de développement.

☐ Vrai

☒ Faux


 Les bonnes pratiques de code indiquent que le code doit être en anglais, donc ce nom français ne respecte pas ce standard. De plus, il est très peu probable que le nom "maVariable" renseigne vraiment sur son contenu. Appeler une variable "variable" est l'équivalent développeur de répondre "une voiture" à la question "qu'est-ce que vous avez comme voiture ?".

Exercice

La variable `$_maVariable` est une variable valide en PHP.

☒ Vrai

☐ Faux


 Une variable peut commencer par une lettre ou par un symbole "souligné", donc le nom "`_maVariable`" est valide.

Exercice

La variable `$1variable` est une variable valide en PHP.

☐ Vrai

☒ Faux


 Un nom de variable ne peut pas commencer par un chiffre ! Elle peut en contenir cependant, donc `$variable1` aurait été un nom valide.

Exercice

La variable `$-variable` est une variable valide en PHP.

☐ Vrai

☒ Faux

 Seul le caractère souligné "_" est autorisé dans le nom d'une variable, le tiret "-" n'est pas autorisé.


Exercice

Qu'est-ce que le code suivant affiche ?

```
1 <?php
2
3 $clientName = 'Henry';
4 $clientName = 'Robert';
5
6 echo $clientName;
```

☐ Henry

☒ Robert

 La valeur initiale "Henry" de `$clientName` est écrasée par la valeur "Robert", donc c'est cette dernière qui est affichée.

Exercice p. 16 Solution n°8

Exercice

Quels sont les types scalaires de PHP ?

☒ string

☐ variable

☒ bool

☐ PHP


☒ int

☒ float

☐ type

☐ array

☐ \$

 PHP possède 4 types scalaires : *int*, *float*, *string* et *bool*.


Exercice

Le code suivant est valide en PHP :

```
1 <?php
2     $userName = 'John';
3     $userAge = 30;
4     $userIsConnected = true;
5     $userBankAmount = 1324,20;
```

☐ Vrai

☒ Faux


 Le séparateur pour les *float* est le point, et non la virgule.

Exercice

Qu'est-ce que le code suivant affiche ?

```
1 <?php
2     $userAge = 30;
3     $userName = 'John';
4     echo 'Je suis $userName et je suis né il y a $userAge ans !';
```

Je suis \$userName et je suis né il y a \$userAge ans !

 L'utilisation des guillemets simples fait que la chaîne de caractères n'est pas interprétée : les variables ne sont pas remplacées par leurs valeurs.

Exercice

Qu'est-ce que le code suivant affiche ?

```
1 <?php
2     $userAge = 30;
3     $userName = 'John';
4     echo "Je suis $userName et je suis né il y a $userAge ans !";
```

Je suis John et je suis né il y a 30 ans !

 Les apostrophes permettent d'interpréter les variables.

p. 17 Solution n°9

```
1 <?php
2     define('DOLLAR_RATE', 1.08);
3     define('YEN_RATE', 120.34);
4     define('BITCOIN_RATE', 0.00016);
5     $eurosPrice = 3.12;
6     $dollarPrice = $eurosPrice * DOLLAR_RATE;
7     $yenPrice = $eurosPrice * YEN_RATE;
8     $bitcoinPrice = $eurosPrice * BITCOIN_RATE;
9     echo $eurosPrice.' euros = '.$dollarPrice.' dollars, '.$yenPrice.' yens et
10    '.$bitcoinPrice.' bitcoins.';
```