Initiation à la programmation

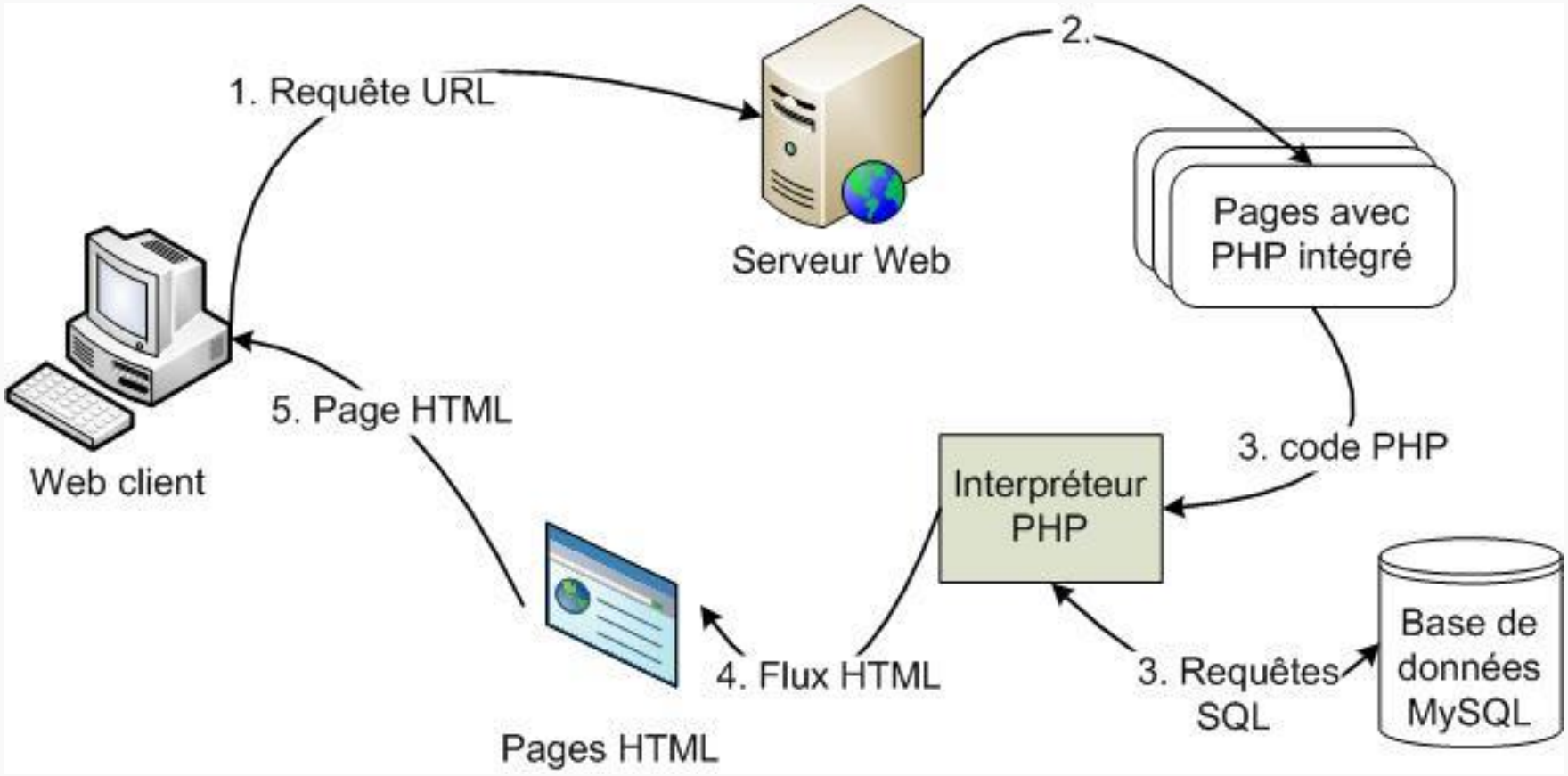
Introduction

Définition de la programmation
Interprétation des scripts
Outils
L’algorithme
Conventions de base

Le programme informatique

- Un programme informatique est composé d'une série d'instructions permettant à l'ordinateur d'exécuter certaines tâches.
- L'ensemble de ces instructions constitue le code source ou encore les scripts qui sont enregistrés dans un ou des fichiers.
- Afin d'écrire un programme, le développeur doit choisir un langage de programmation spécifique en fonction de l'objectif à atteindre et des spécificités de ce dernier.
- Pour être exécuté, le code source devra être traduit sous une forme compréhensible par l'ordinateur (binaire). C'est le rôle de l'interpréteur pour le PHP ou du compilateur pour le C++, Java…

Langage interprété (PHP)



Réalisation des scripts

- Pour écrire le programme vous devez disposer d'un logiciel de type éditeur de code. Il en existe de nombreux dont certains sont spécifiques au langage que vous avez choisi. Pour bénéficier de fonctionnalités avancées, il est préférable d'utiliser une plate-forme de développement appelée aussi IDE (Integrated Development Environment).

- Exemples:
  - PhpStorm
  - Visual Studio Code
  - Eclipse
  - NetBeans

Interprétation des scripts et affichage des pages web

- Pour que votre code source puisse être compris par la machine et transformé en page dans un navigateur, vous devez disposez de plusieurs applications (solution web).
- WampServeur: il s’agit d’un environnement indispensable à la création de sites web dynamiques contenant:
  - Un serveur Apache (communication entre les machines)
  - Le langage PHP et son interpréteur
  - Un serveur MySQL pour la gestion des bases de données

Les scripts PHP

- Vos fichiers contenant le code source doivent obligatoirement porter l'extension .php
- Pour être interprétés, ils seront enregistrés dans le dossier www qui représente le localhost.
- Pour afficher le résultat de votre code dans un navigateur, vous devez obligatoirement passer par le localhost. Chaque URL commencera donc par localhost/sous_dossier/fichier.php
- Le résultat affiché dans le navigateur (client) correspond à l'interprétation de votre code. Il s'agit pour la plupart du temps de code HTML, le code source étant bien entendu invisible.

Initiation à la programmation en PHP

Introduction
Premiers scripts
Les variables
Affichage du contenu

Introduction

- PHP (Hypertext Preprocessor) est un langage de programmation spécifiquement orienté pour le web.
- Il fait partie de la famille des langages interprétés côté serveur.
- Il suit une évolution constante et reste le langage le plus utilisé pour la création de sites dynamiques. La dernière version majeure sortie est PHP 8 (en version majeure) qui intègre des notions avancées d'orienté objet, de typage une interprétation très rapide et une meilleure gestion de la mémoire.
- Un autre avantage est qu'il dispose d'une gigantesque communauté développant des outils (librairies, frameworks, services…) open source et proposant ses services.

Les tags PHP

- Sur un fichier PHP, le code doit être délimité par des tags (délimiteurs) permettant d'identifier le PHP qui devra être interprété.
- Ces délimiteurs sont: <?php et ?> On n'écrit pas le tag de fermeture si le code se termine par du PHP.
- Chaque instruction en PHP se termine par un point virgule.

Syntaxe

```
<?php
// Ici du PHP
?>
Ici du HTML ou du JavaScript
<?php
// Encore du PHP
```

Exemple

```
<?php

echo 'Je suis un script PHP';

?>

<h1>Je suis un titre de niveau 1 en HTML</h1>

<?php

echo 'Je suis la suite du script PHP';
```

Les commentaires

les commentaires sur plusieurs sont introduits par la séquence **/*** et se terminent par */. Par ailleurs, PHP utilise également les signes de commentaires //, qui permettent de commenter une ligne complète. Les commentaires ne sont pas interprétés et n'apparaissent pas dans le navigateur.

```
// Commentaire sur une ligne

/*
 * Commentaire
 * sur plusieurs lignes
 */
```

L’instruction echo

- L’instruction echo permet d’afficher un contenu ou un résultat dans le navigateur.
- Quand il s’agit de renvoyer une chaîne de caractères (string) vous devez l’encadrer par des délimiteurs. Ceux-ci sont soit l’apostrophe, soit les guillemets.

```
// délimiteurs simples
echo 'SITE IEPSCF';
echo '<h1>INTRODUCTION</h1>';
echo '<h2>L\'histoire d\'internet</h2>';

// délimiteurs doubles
echo "SITE IEPSCF";
echo "<h1>INTRODUCTION</h1>";
echo "Descartes à dit: \"je pense donc je suis\"";
```

L’instruction print

- Cette commande joue le même rôle que « echo », Elle retourne au navigateur un contenu.
- Exemples:

```php
// output avec print
print 'INTRODUCTION';
print "<h1>INTRODUCTION</h1>";
print("Introduction"); // parenthèses non obligatoires
```

- Echo et print ne sont pas de véritables fonctions du langage PHP mais plutôt des structures du langage. C’est pour cette raison que l’on ne met pas de parenthèses.

Les inclusions de fichiers (SSI)

Les instructions include et require permettent d’inclure et d’exécuter un fichier externe dans votre script.
Exemples:

- include 'partials/header.php’;
- require 'partials/header.php’;

La différence réside dans le fait que si le fichier à inclure contient des erreurs require arrête l’exécution du script tandis que include affiche le résultat du script.

Cette fonctionnalité permet de maintenir facilement un site web car si l’on modifie une donnée sur les fichiers inclus toutes les pages sont automatiquement mise à jour.

Les inclusions uniques

Pour éviter des inclusions multiples d’un même fichier PHP vous permet d’utiliser des instructions similaires qui sont include_once et require_once.
Exemples:

- include_once 'partials/header.php’;
- require_once 'partials/header.php’;

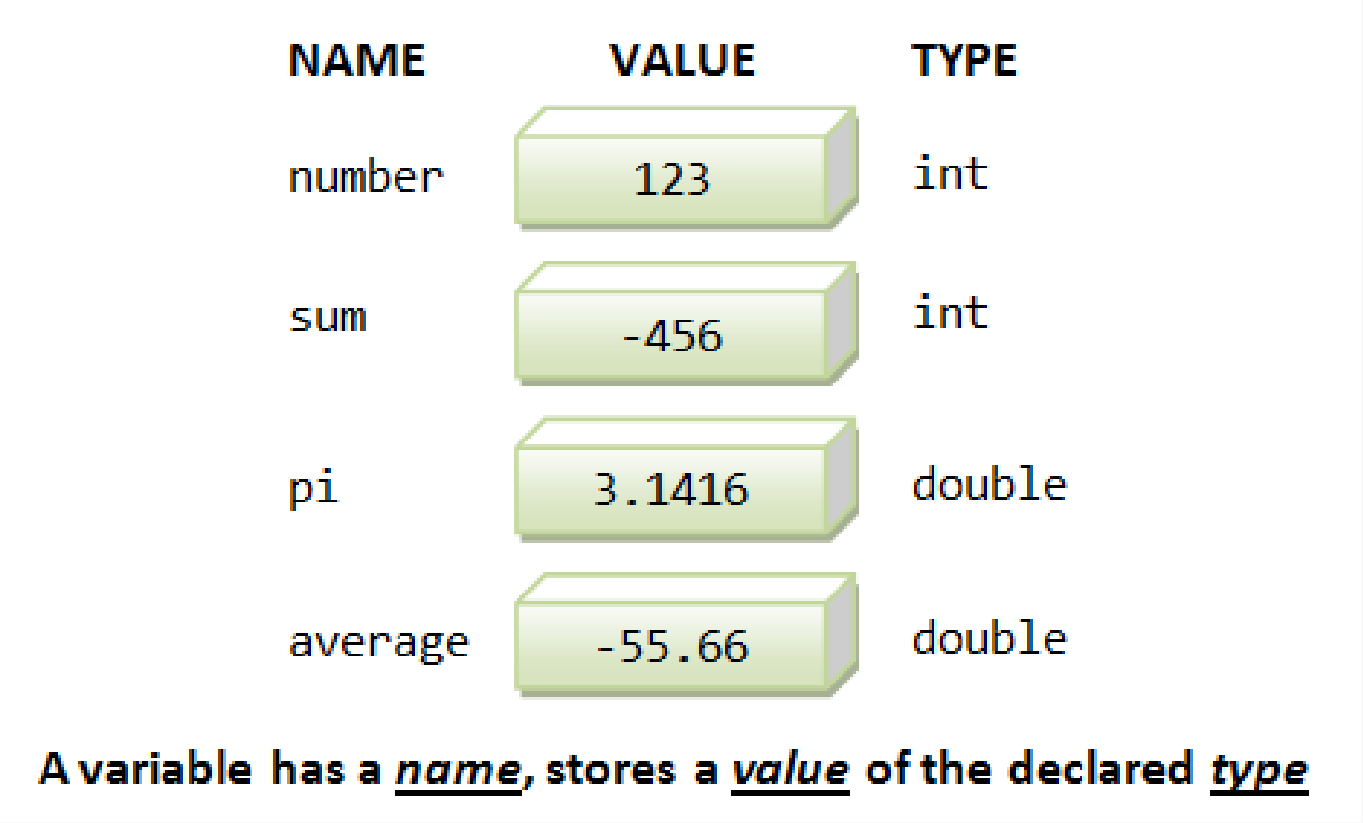
Cette façon de procéder se révèle efficace lors de l’utilisation de classes ou de fonctions pour éviter les redéfinitions.

Les variables et les constantes

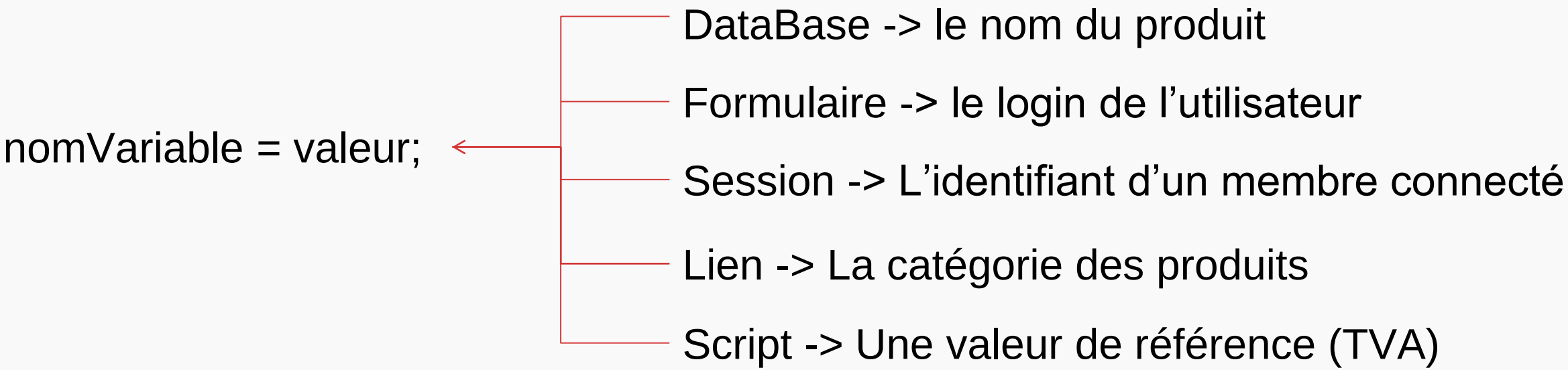
Introduction

- Les variables sont destinées à stocker les données qui seront utilisées et pourront être modifiées lors de l’exécution du programme. Il s’agit d’un espace mémoire (boîte) portant une étiquette (nom) et contenant une donnée.
- Il est important de noter que la plupart du temps les données des variables proviennent de l'extérieur du script (formulaire, base de données, sessions, cookies...)
- Contrairement à d'autres langages, le PHP n’exige pas la déclaration de type des variables avant leur utilisation. Une variable peut prendre une valeur de chaîne de caractères et, ensuite, être utilisée pour contenir un entier (transtypage automatique).

Principe de base



Syntaxe et exemples




```
$mail = 'john.doe@gmail.com';
$productName = 'Acer-Predator';
$price = 125.50;
```

Conventions de nommage

- Chaque variable possède un identifiant particulier, qui commence toujours par le caractère dollar ($) suivi du nom de la variable. Les règles de création des noms de variable sont les suivantes :
  - Le nom commence par un caractère alphabétique, pris dans les ensembles [a-z], [A-Z] ou par le caractère de soulignement.
  - Les chiffres peuvent être utilisés à l'exception du premier caractère du nom de la variable.
  - Les noms des variables sont sensibles à la casse.
  - Utilisez l’écriture camelCase pour les variables avec des noms composés: $userName ou encore $userFirstName

Exemples

```
$name // valide
$01name // invalide
$_name // valide
$NAME // valide (non conventionnel)
$first name // invalide
$first*name // invalide
$firstName // valide
$prénom // valide (non conventionnel)
```

Utilisation simple des variables

```php
// Declaration of variables
$firstName = 'John'; //string
$lastName = 'Doe'; //string
$age = 40; //integer
$price = 100; // integer
$quantity = 5; // integer

// Use of variables
echo $firstName; // John
echo '<br>'; //saut de ligne en html
echo $firstName.' - '.$lastName; // John - Doe
echo '<br>'; //saut de ligne en html
echo $firstName.' '.$lastName.' est âgé de '.$age.' ans.'; // John Doe est agé de 40 ans
echo '<br>Montant de la commande: '.$price * $quantity; // Montant de la commande: 500
```

Opérations sur les variables: les opérateurs d'affectation

- Le processus de création d'une variable s'appelle une déclaration. En PHP la déclaration est accompagnée de l'affectation. Il s'agit d'assigner une valeur à la variable.

```php
$firstName = 'John';
$lastName = 'Doe';
$name = $firstName.$lastName; // JohnDoe
$name = $firstName.' '.$lastName; // John Doe
$price = 10;
$price = $price + 5; // 15
$price += 5; // 20
$rate = 21;
$tva = $price * $rate / 100; // 4.2
```

Opérations sur les variables: opérateurs d’affectation

- Opérateurs d'affectation
  - Simple

    $a = 10;
  - Opérateurs combinés

    En plus du simple opérateur d'affectation, il existe des "opérateurs combinés" pour tous les opérateurs arithmétiques.

    $a = 10;

    $b = 5;

    $c = 'Hello';

    $b += $a; // 15

    $c .= 'World !'; // HelloWorld !

Opérations sur les variables: opérateurs arithmétiques

A condition d’être de type integer ou float, vous pouvez réaliser les opérations classiques sur vos variables. Voici un récapitulatif des opérateurs arithmétiques utilisés en PHP.

-$a	Négation	(Opposé de $a)
$a + $b	Addition	(Somme de $a et $b)
$a - $b	Soustraction	(Différence de $a et $b)
$a * $b	Multiplication	(Produit de $a et $b)
$a / $b	Division	(Quotient de $a et $b)
$a % $b	Modulo	(Reste de $a divisé par $b)
$a ** $b	Exponentielle	(Résultat de l'élévation de $a à la puissance $b)

Les types de variables

Comme déjà signalé, le PHP ne nécessite pas la déclaration de type pour les variables. Pourtant ces types existent bien et sont déclarés lors de l'affectation de la variable typage dynamique).
Cependant, si vous le souhaitez, vous pouvez typer vos variables manuellement.
Les types existants:

boolean: C'est le type le plus simple. Un booléen représente une valeur de vérité. Il peut valoir TRUE ou FALSE.

integer: Un entier est un nombre appartenant à la classe $\mathbb{Z}$ = {..., -2, -1, 0, 1, 2, ...}.

float: Les nombres à virgule flottante (1.5 | 1.785 | -100.20 | …)

string: Une chaîne de caractères est une série de caractères alphanumériques ou numériques

Les types de variables

- array: Un tableau en PHP est en fait série de valeurs ordonnées enregistrées dans une même variable ($courses = ['PHP', 'HTML', 'CSS'])
- null: La valeur spéciale NULL représente une variable sans valeur
- object: Un objet (POO)
- callable: Une fonction
- iterable: Pseudo-type (PHP 7.1) Variable parcourue par la boucle foreach
- resource: Une ressource est une variable spéciale, contenant une référence vers une ressource externe

Exemples

```php
$driversLicense = true; // boolean
$postalCode = 5000; // integer
$price = 20.50; // float
$name = 'John Doe'; // string
$courses = ['PHP', 'HTML', 'CSS', 'JavaScript']; // array
$old = null; // null
$dbh = new DateTime(); // object
$cut = cutString(); // callable
$content = fopen('list.php', 'W'); // resource
```

Fonctions pour manipuler les variables

- PHP propose une série de fonctions permettant de manipuler ou d'obtenir des informations sur les variables. La plupart du temps, elles s'utilisent dans le cadre d'une conditionnelle (if). Nous aborderons ce cas de figure mais sans entrer maintenant dans les détails.
- Bien que PHP n'oblige pas le typage des variables, les types existent et peuvent être manipulés (vérification, validation,…).

gettype(): retourne le type de la variable

```
Exemple: echo gettype($name); // string
```

settype(): Affecte un type à une variable

```
Exemple: settype($number, 'string'); // var de $number de type string
```

Les fonctions des variables

- is_string(): Détermine si une variable est de type chaîne de caractères. Il existe une fonction semblable par type de données. Dans ce cas, retourne TRUE si la variable est une chaîne de caractères, FALSE sinon.

```
if(is_string($name)) {
    echo 'Votre nom est bien une chaîne de caractères';
}
```

- is_int() ou is_integer(): Détermine si la variable donnée est de type nombre entier.
- is_numeric(): Détermine si une variable est un type numérique
- is_bool(): Détermine si une variable est un booléen
- is_float(): Détermine si la variable donnée est de type nombre décimal
- Référez-vous à la documentation officielle pour les autres types

Les fonctions des variables

empty(): Détermine si une variable est vide

```
$login = '';
if(empty($login)){
    echo 'Votre login est obligatoire';
}
```

isset(): Détermine si une variable est déclarée et est différente de NULL

```
if(!isset($password)){
    echo 'Vous devez vous identifier';
}
```

Les fonctions des variables

unset(): Détruit une ou plusieurs variables: unset(var01, var02,…)

```
$login = 'John';
unset($login);
echo $login; // Undefined variable: login
```

var_dump(): affiche les informations structurées d'une variable, y compris son type et sa valeur. Cette fonction est utilisée uniquement dans le cadre d’un débogage.

```
$price = 15.5;
var_dump($price); // float(15.5)
```

Les constantes

- Une constante est un identifiant (un nom) qui représente une valeur simple. Comme son nom le suggère, cette valeur ne peut jamais être modifiée durant l'exécution du script. Par défaut, le nom d'une constante est sensible à la casse. Par convention, les constantes sont toujours en majuscules.
- Dans un site web, les constantes permettront de stocker des valeurs de paramétrage (settings) sur lesquelles vous pourrez réaliser les mêmes opérations que pour les variables.
- Définition d’une constante:

```
define('CONST', ‘valeur’); // syntaxe
define('DB_CLIENT', 'catalogue'); // exemple

echo DB_CLIENT; // catalogue
```

Les structures du langage

Les conditionnelles
Les boucles
Les tableaux

Les structures de contrôle

La conditionnelle if()
La boucle for()

La structure conditionnelle – if()

Introduction

- Dans vos pages web vous serez souvent amenés à utiliser des conditions. Elles servent à afficher ou traiter des informations en fonction de certaines conditions ou certaines valeurs reçues.

Exemple:

- Si le visiteur a un rôle d'administrateur, il pourra accéder et modifier toutes les données du site. Par contre s'il possède un rôle d'utilisateur, il ne pourra que consulter les données.
- Afin d'utiliser les conditionnelles en programmation, vous devez d'abord connaître certains symboles (opérateurs de comparaison).

Opérateurs de comparaison

- **Opérateurs de comparaison**

$a **==** $b	Egal
$a **===** $b	Identique
$a **!=** $b	Différent
$a **<** $b	Plus petit que
$a **>** $b	Plus grand
$a **<=** $b	Inférieur ou égal
$a **>=** $b	Supérieur ou égal

Schéma logique if()

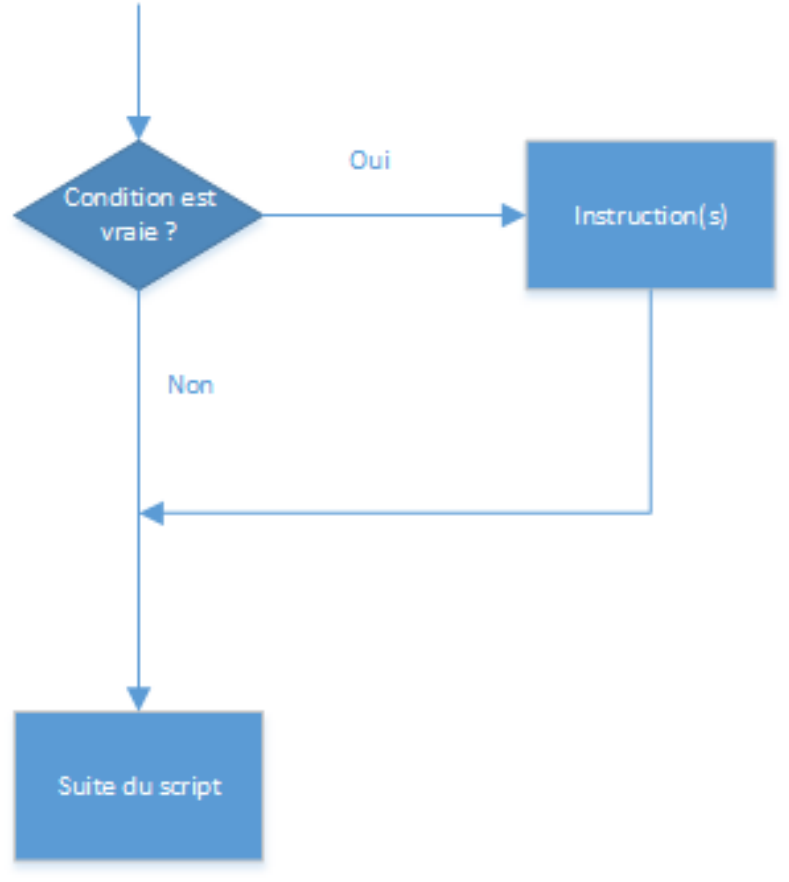
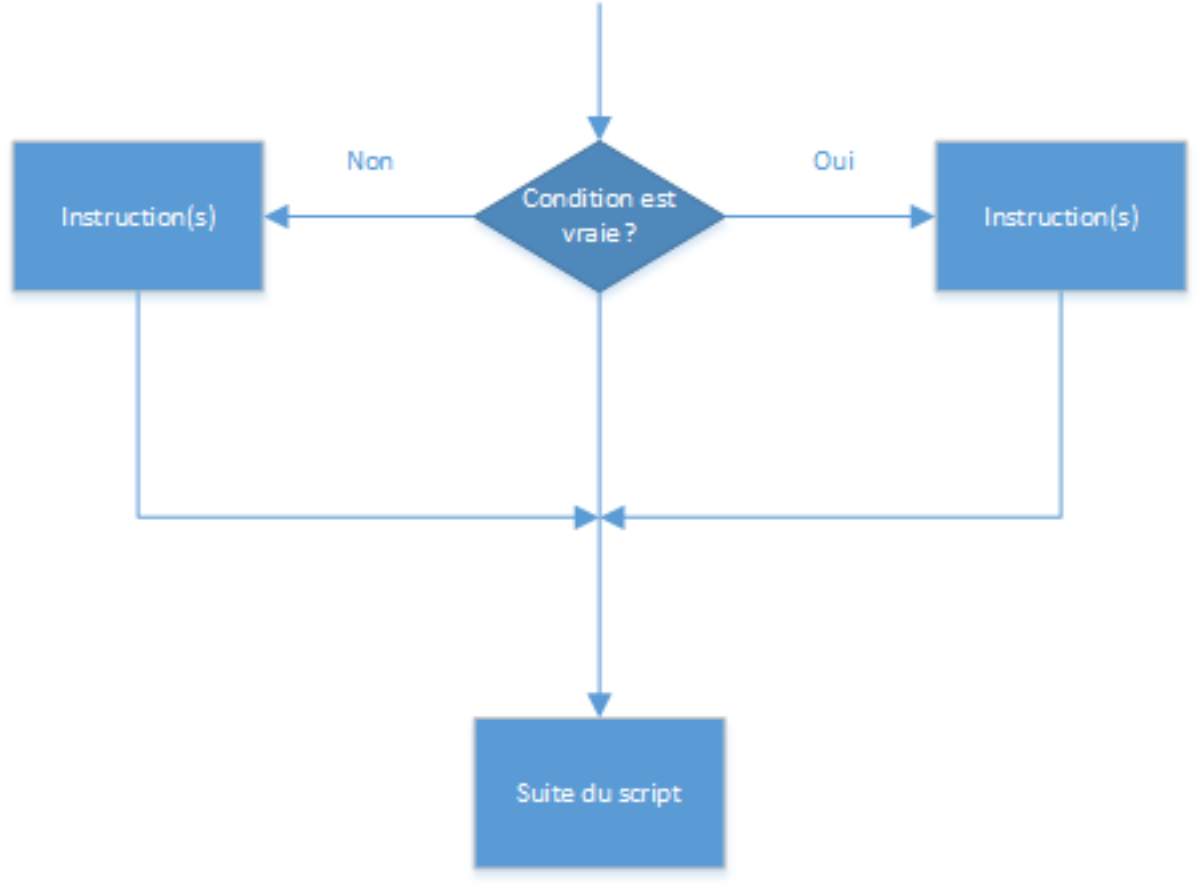


Schéma logique if() else



Syntaxe de la condition if()

- Instruction IF()

```
if (condition) {
  commandes
}
```

- Instruction IF() ELSE

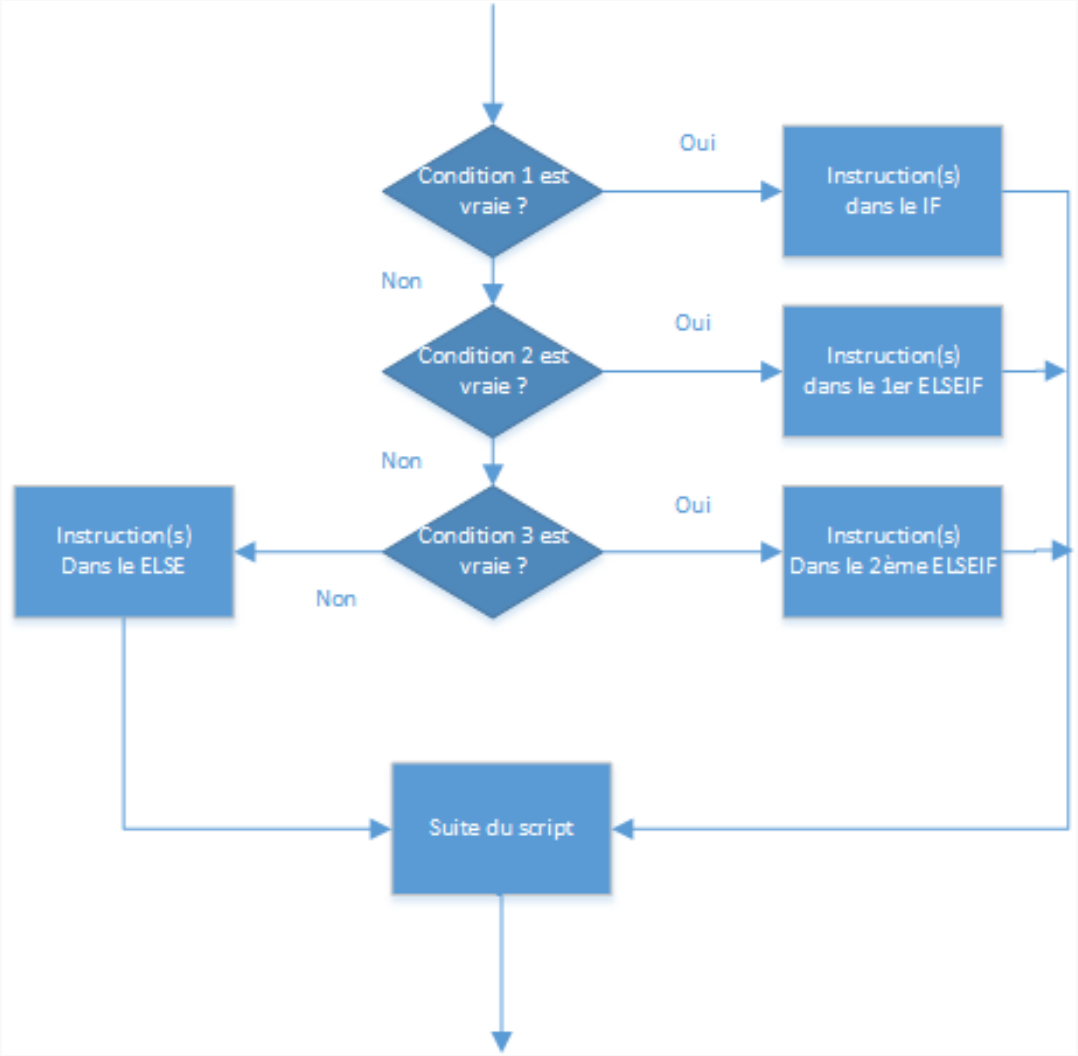
```
if (condition) {
  commandes
} else {
commandes
}
```

```php
$points = 12;

if($points >= 10) {
    echo 'Réussite';
}
```

```php
if($points >= 10) {
    echo 'Réussite';
} else {
    echo 'Echec';
}
```

Schéma logique if() elseif() else



Syntaxe de la condition if() elseif()

- Instruction IF() ELSEIF()

```
if (condition 1) {

  commandes

} elseif (condition 2) {

commandes

}
```

```php
if($points >= 14) {
    echo 'Bonne moyenne';
} elseif ($points >= 10) {
    echo 'Moyenne';
}
```

Syntaxe de la condition if() elseif() else

- Instruction IF() ELSEIF() ELSE

```
if (condition 1) {
   commandes
} elseif (condition 2) {
   commandes
} else {
   commandes
}
```

```php
if($points >= 14) {
    echo 'Bonne moyenne';
} elseif ($points >= 10) {
    echo 'Moyenne';
} else {
    echo 'Echec';
}
```

Conditions multiples

- Dans ce cas de figure, on va poser plusieurs conditions à la fois. Pour cela, on aura besoin de nouveaux opérateurs. Voici les principaux à connaître :
- Opérateurs logiques

! $a	Not (Non)
$a && $b	And (Et)
$a || $b	Or (Ou)
$a xor $b	XOR (si $a OU $b est TRUE, mais pas les deux en même temps)

- Exemple

```
if($age >= 18 && $country == 'Belgique') {
    echo 'Bienvenu sur mon site';
} else {
    echo 'Vous ne remplissez pas les conditions';
}
```

Les deux conditions doivent être remplies

La structure d’itération – for()

Introduction

- La boucle est aussi une structure de contrôle indispensable en programmation. Vous disposez d'une base de données contenant les produits que vous souhaiteriez afficher sur votre page. La seule façon de procéder, est l'utilisation d'une boucle qui vous permettra en une seule opération l'affichage de tous les produits.
- Une boucle est une structure répétitive (itération) dont le fonctionnement sera toujours le même. On pose une condition qui sera généralement liée à la valeur d'une variable et on exécute le code de la boucle « en boucle » tant que la condition est vérifiée.
- Pour éviter de rester bloqué à l'infini dans une boucle, il faudra que la condition donnée soit fausse à un moment donné (pour pouvoir sortir de la boucle).

Opérateurs d’incrémentation et de décrémentation

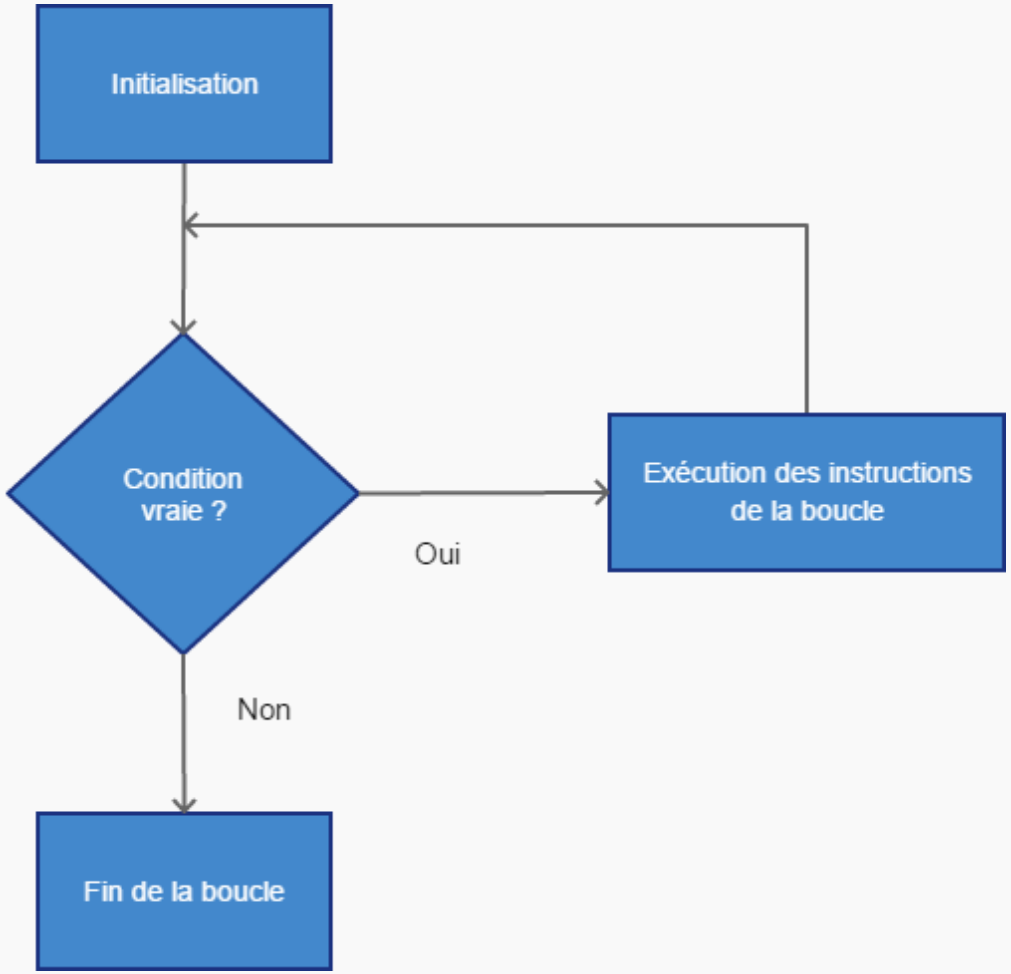
- Pour que notre condition devienne fausse à un moment, on incrémentera ou on décrémentera la valeur de notre variable à chaque nouveau passage dans la boucle.

- Opérateurs d'incrémentation et décrémentation
  - ++$a Pré-incrémente
  - $a++ Post-incrémente
  - --$a Pré-décrémente
  - $a-- Post-décrémente

exemples

```php
$a = 1;
$b = 1;
$c = 3;
echo $a++; // 1
echo '<br>';
echo $a; // 2
echo '<br>';
echo ++$b; // 2
echo '<br>';
echo $c--; // 3
echo '<br>';
echo $c; // 2
```

Schéma logique du for()



Syntaxe

- Boucle FOR()

for (initialisation; condition; incrémentation) {

commandes;

}

```php
for($i = 1; $i <= 10; $i++) {
    echo $i.'<br>';
}
```

Cette boucle ne sert qu’a afficher des chiffres de 1 à 10 mais elle illustre bien la syntaxe de base.

Commentaires

- Pour exécuter une action plusieurs fois, nous avons besoin d’une variable que l’on initialise à 1.
- Ensuite nous devons décrire la condition d’arrêt de la boucle.
- Enfin, la variable devra lors de chaque passage être incrémentée (augmentée) de 1. Sinon la boucle est infinie.
- Entre les accolades nous indiquons le traitement à réaliser lors de chaque passage dans la boucle.

La structure tableau (array)

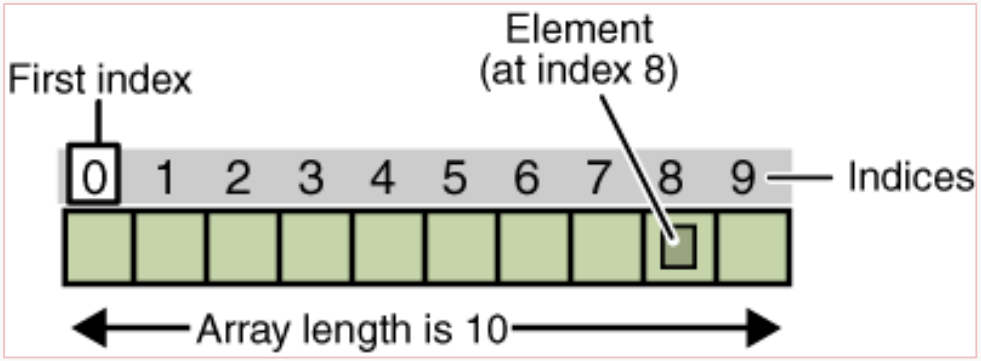
Introduction

- Contrairement à une variable classique qui ne stocke qu'une seule valeur, un tableau, en revanche, est une variable stockant un ensemble ou une série de valeurs. Un même tableau peut contenir de nombreux éléments, chacun d'eux pouvant être une valeur unique, comme un texte ou un nombre, ou un autre tableau. Un tableau comprenant d'autres tableaux est dit "multidimensionnel".

- Dès lors que des informations sont enregistrées dans un tableau, elles peuvent être soumises à diverses manipulations d'affichage et de traitement (tris, recherches, parcours, calculs, affichages groupés…) L'ensemble des informations enregistrées dans un tableau peut être manipulé comme s'il s'agissait d'une seule entité.

Tableaux indexés (numérotés)

- Il existe deux types de tableaux: les tableaux indexés ou associatifs. Nous commencerons par les premiers. Il s'agit de la structure la plus simple ou chaque donnée est accessible par sa position dans le tableau.



- Dans un tableau, les valeurs sont rangées dans des « cases » différentes. Ici, nous travaillons sur un array numéroté, c'est-à-dire que chaque case est identifiée par un numéro. Ce numéro est appelé clé et la première est toujours 0

Déclaration et affichage du tableau

- Déclaration et initialisation

```php
$courses = ['HTML', 'CSS', 'PHP', 'JavaScript', 'Network'];
```

- Affichage

```php
echo $courses[0]; // HTML
echo $courses[1]; // CSS
echo $courses[2]; // PHP
echo $courses[3]; // JavaScript
echo $courses[4]; // Network
```

Parcourir le tableau avec la boucle foreach()

- Afficher une à une les données du tableau est répétitif et fastidieux. En PHP, il existe une boucle spécifique pour les tableaux: la boucle foreach().
- Elle va passer en revue chaque donnée du tableau, et lors de chaque passage, elle va mettre la valeur dans une variable temporaire.
- Syntaxe:

```
foreach(array as variable) {

    echo $variable;

}
```

```php
foreach($courses as $course) {
    echo $course.'<br>';
}
```

Tableaux associatifs

- Un tableau associatif est un tableau qui va utiliser des clefs textuelles qu'on va associer à chaque valeur.
- Les tableaux associatifs vont s'avérer intéressant lorsqu'on voudra donner du sens à nos clefs, c'est-à-dire créer une association forte entre les clefs et les valeurs d'un tableau.
- Imaginons par exemple qu'on souhaite stocker les périodes de nos différents cours dans un tableau. C'est impossible dans le tableau indexé car chaque valeur reçoit automatiquement une clé de 0 à n. Par contre avec le tableau associatif je vais pouvoir associer à une clé de mon choix une valeur. Le nom du cours comme clé et les périodes comme valeurs.

Déclaration et affichage

- Déclaration

```php
$courses = [
    'HTML' => 80,
    'CSS' => 100,
    'PHP' => 160,
    'JavaScript' => 120,
    'Network' => 80
];
```

L'opérateur **=>** sert ici à associer la clé à une valeur

- Affichage

```php
foreach($courses as $course => $time) {
    echo $course.' - '.$time.'<br>';
}
```

1. d'abord le nom du tableau;
2. ensuite le mot-clé **as** (qui signifie « en tant que »);
3. enfin, le nom des variables séparées par l'opérateur **=>** qui vont contenir tour à tour chacun des éléments du tableau.

Trier un tableau

Php met à votre disposition une série de fonctions permettant de trier un tableau selon qu'il soit indexé ou associatif. Commencez par trier le tableau avant la boucle d'affichage.

Tableaux indexés

```
$courses = ['HTML', 'CSS', 'PHP', 'JavaScript', 'Network'];

sort($courses); // Du plus petit au plus grand ou ordre alphabétique

rsort($courses); // Du plus grans au plus petit ou ordre alphabétique inverse
```

Trier un tableau

Tableaux associatifs

```
$coursesTime = [ 'HTML' => 80, 'CSS' => 100, 'PHP' => 160, 'JavaScript' => 120, 'Network' => 80];

ksort($coursesTime);  // Trie le tableau sur les clés par ordre croissant

krsort($coursesTime);  // Trie le tableau sur les clés par ordre décroissant
```