


Intro **PHP : Hypertext Preprocessor** PHP 2

**Langage côté serveur**

↓

**Un serveur Web est indispensable pour interpréter le code PHP**



**Apache Server Status**

Server Version: Apache/2.4.18 (Ubuntu)  
 Server Built: May 11 2008 21:05:34  
 Current Time: Monday, 17-Jul-2010 12:00:00  
 Restart Time: Monday, 17-Jul-2010 12:00:00  
 Parent Server Generation: 0  
 Server uptime: 15 minutes 10 seconds  
 Total accesses: 9737 - Total Traffic: 487 KB  
 CPU Usage: u8.6 s10.64 c0.0 - 2.11% CPU load  
 10.7 requests/sec - 548 Bytes/sec - 51.83 requests currently being processed, 4 idle servers

| Developer | December 2014 | Percent | January 2015 | Percent | Change |
|-----------|---------------|---------|--------------|---------|--------|
| Apache    | 358,159,405   | 39.11%  | 348,460,753  | 39.74%  | 0.63   |
| Microsoft | 272,967,294   | 29.81%  | 241,276,347  | 27.52%  | -2.29  |
| nginx     | 132,467,763   | 14.47%  | 128,083,920  | 14.61%  | 0.14   |
| Google    | 20,011,260    | 2.19%   | 20,209,649   | 2.30%   | 0.12   |

http://news.netcraft.com/

```

6-0 13444/0/691/691 W 1.31 30 0 0.04 128.1.34.82 GET /news/HelloServer HTTP/1.0
7-0 13445/0/632/632 W 1.31 25 0 0.03 128.1.34.82 GET /news/HelloServer HTTP/1.0
8-0 13446/0/638/638 W 1.41 26 0 0.03 128.1.34.82 GET /news/HelloServer HTTP/1.0
9-0 13447/0/583/583 W 1.09 24 7 0.03 128.1.34.82 GET /news/HelloServer HTTP/1.0
10-0 - 0/0/542 - 1.19 27 5 0.02 128.1.34.82 GET /news/HelloServer HTTP/1.0
11-0 - 0/0/541 - 1.05 25 10 0.02 128.1.34.82 GET /news/HelloServer HTTP/1.0
  
```

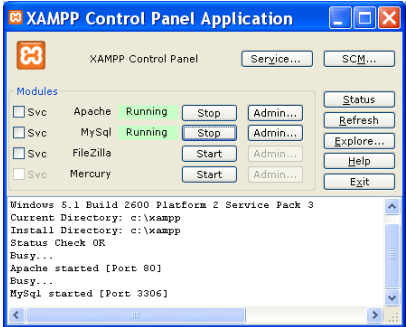
Srv Child Server number - generation  
 PID OS process ID  
 Acc Number of accesses this connection / this child / this slot

http://w3techs.com/technologies/overview/web\_server/all

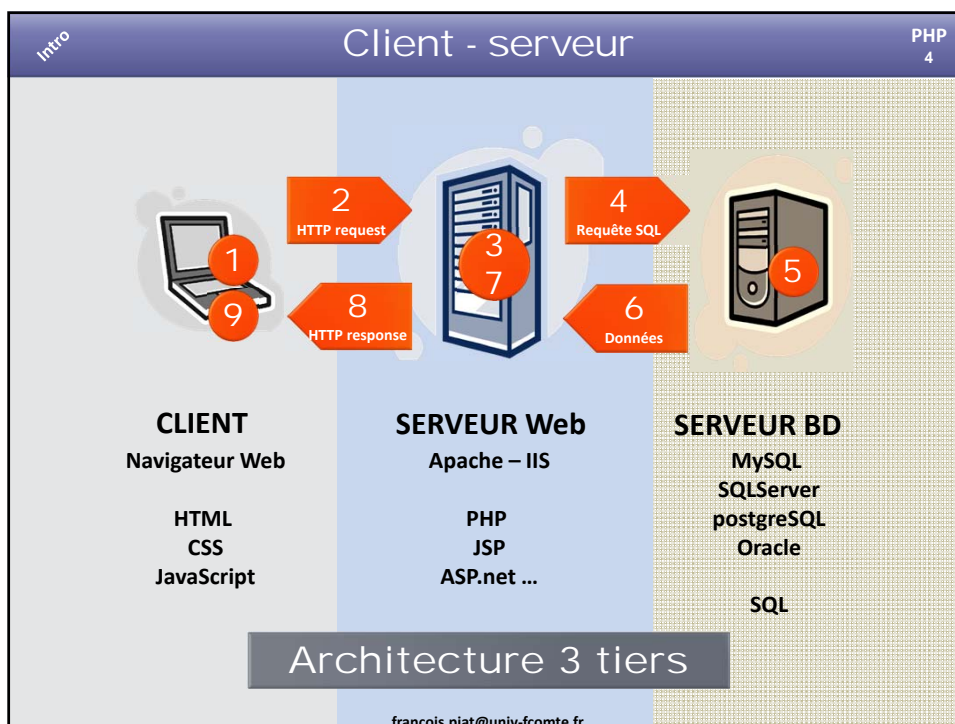
francois.piat@univ-fcomte.fr

Intro
Environnement
PHP 3

- **L**inux
- A**pache
- M**ySQL
- P**HP
  
- **W**indows
- A**pache
- M**ySQL
- P**HP
  
- <http://www.easyphp.org/>
- <http://www.wampserver.com/>
- <http://www.apachefriends.org/fr/xampp.html>



francois.piat@univ-fcomte.fr



**Client - serveur**

1. Client sends an HTTP request.

2. Server receives the HTTP request.

3. Server processes the request.

4. Server sends a SQL query.

5. Database server receives the SQL query.

6. Database server returns data.

7. Server receives the data.

8. Server sends an HTTP response.

9. Client receives the HTTP response.

**HTTP request:**

```
GET /index.php HTTP/1.1
Host: www.univ-fcomte.fr
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_7_2; rv:1.9.2.2) Gecko/20100302 Firefox/3.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: fr
Accept-Encoding: gzip, deflate
Accept-Charset: utf-8,utf-16,utf-32;q=0.7,*;q=0.3
Cookie: PHPSESSID=...
```

**SQL query:**

```
SELECT * FROM modules, diplomes
WHERE moUniversite = 'Lons-le-Saunier'
AND moInscripti
AND moType = '$1'
```

**Website Screenshot:**

UFC Université de Franche-Comté

Actualités

La physique quantique

Le chercheur

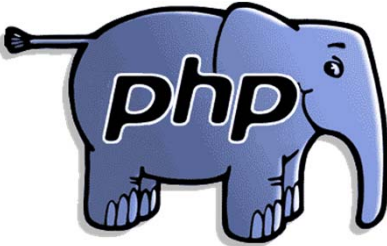
Emplois

Intro

Histoire

PHP6

- **PHP : Hypertext Preprocessor**
- **Langage Open Source**
  - Version 4 en 2000. Vrai démarrage de PHP
  - Version 5 en 2004. Programmation objet
  - <http://www.php.net/manual/fr/history.php.php>
- **Bibliothèques de fonctions très fournies**
  - Pas de règles de nommage
  - Utiliser l'aide de PHP
  - <http://www.php.net/manual/fr/>




francois.niet@univ-lyon2.fr

Intro

Langage

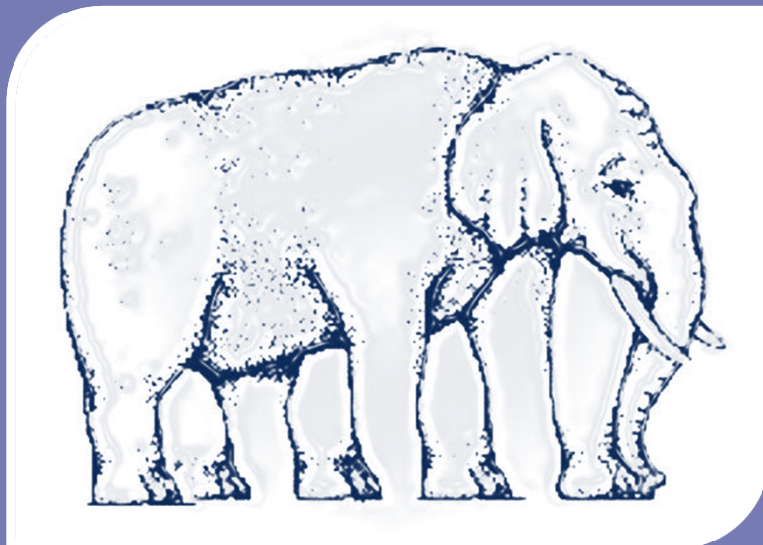
PHP  
7

- Fichier texte avec extension **.php**
  - Indique au serveur qu'il doit interpréter le code du fichier
  - Code PHP inclus entre les tags **<?php** et **?>**
- Pas de compilation : langage interprété
- Structure et syntaxe comme **C** ou **Java**
  - La casse n'a pas d'importance, sauf dans le nom des variables
  - Les instructions doivent se terminer par **;**
- Typage dynamique des variables



francois.piat@univ-fcomte.fr

# Types et variables



Variables
Types de données
PHP 9

- **Scalaires**
  - Chaînes de caractères
  - Entiers (+- 2 147 483)
  - Virgule flottante (+- 1.7<sup>E</sup>-308)
  - Booléen **FALSE** / **TRUE**
- **Composés**
  - Tableaux
  - Objets
- **Spéciaux**
  - Ressource
  - **null**

**FALSE**

0

0.0

''

'0'

tableau vide

null

objet vide

**TRUE**

tout ce qui  
n'est pas  
évalué à  
**FALSE**

francois.piat@univ-fcomte.fr

Variables
Nommer une variable
PHP 10

\$var
\$\_vaR
\$v10
\$\_2

- **Le nom d'une variable**
  - commence par le caractère **\$**
  - suivi d'un **caractère alphabétique** ou du caractère **\_** (underscore)
- **Le reste du nom peut être composé de n'importe quel caractère alphanumérique.**

\$1var
\$\_[v
\$v-10
\$a+b

- **Attention au respect de la casse (majuscules/minuscules)**

\$Var
<>
\$var
<>
\$VAR

francois.piat@univ-fcomte.fr

Variables
Initialiser une variable
PHP 11

- **Pas de déclaration** mais directement une affectation.

```
$nom = 'François';
$taille = 1.8;
$prof = true;
```

```
$cours = getCours();
$moi = $nom.' Piat';
$a = $b = $c = 123;
```

- **Typage dynamique** suivant le type de valeur contenue dans la variable (Type juggling).

```
010 $taille = 1.8;
020 ...
030 $taille = 'un mètre quatre-vingt';
040 ...
050 $taille = array('François', 1.8);
060 ...
```

francois.piat@univ-fcomte.fr

Variables
Connaître le type d'une variable
PHP 12

- **gettype(\$var)**      "boolean" "integer" "double" "string"  
                              "array" "object" "resource" "NULL"

- **is\_bool(\$var)**
- is\_int(\$var)**
- is\_float(\$var)**
- is\_string(\$var)**
- is\_array(\$var)**

- is\_object(\$var)**
- is\_resource(\$var)**
- is\_null(\$var)**
- is\_numeric(\$var)**
- is\_scalar(\$var)**

francois.piat@univ-fcomte.fr

Variables
Transtypage (casting)
PHP 13

- Forcer le type d'une variable.
- `$var = (type) $var;`

↓  
 (string)  
 (int)  
 (float)  
 (array)  
 (object)
- `settype($a, 'type');`

`$a = 1.5;`  
`$b = (string) $a;`  
➔ `$b = '1.5'`

`$a = 1.5;`  
`$b = (int) $a;`  
➔ `$b = 1`

`$a = '100 euros';`  
`$b = (int) $a;`  
➔ `$b = 100`

`$a = '100 euros';`  
`$b = (bool) $a;`  
➔ `$b = TRUE`

francois.piat@univ-fcomte.fr

Variables
Portée (scope)
PHP 14

- 1 **Portée locale** : une variable définie dans une fonction est connue uniquement à l'intérieur de cette fonction.
- 2 **Portée globale** : une variable définie à l'extérieur d'une fonction est connue dans tout le script (ie le fichier php), sauf dans les fonctions.
- 3 **Pas de portée de bloc.**

francois.piat@univ-fcomte.fr

Variables PHP 15

## Portée locale

Une variable définie dans une fonction est connue **uniquement à l'intérieur de la fonction**.

```
<?php

function quiEstLa() {
    $nom = 'François';
    echo $nom;
}

quiEstLa();

echo $nom;

... instructions ...

?>
```

→ François

→

francois.piat@univ-fcomte.fr

Variables PHP 16

## Portée globale

Une variable définie à l'extérieur d'une fonction est connue dans tout le script (ie le fichier php), **sauf dans les fonctions**.

```
<?php

function quiEstLa() {
    echo $nom;
}

$nom = 'François';

echo $nom;

quiEstLa();

echo $nom;

?>
```

→

→ François

→ François

francois.piat@univ-fcomte.fr



Variables

## Locale vs globale

PHP 17

```
<?php
function quiEstLa() {
    $nom = 'Pierre';
    echo $nom;
}

$nom = 'François';
echo $nom;
quiEstLa();
echo $nom;
?>
```

→ Pierre

→ François

→ François

francois.piat@univ-fcomte.fr

Variables

## Mot clé global (1)

PHP 18

Une variable globale peut être connue dans une fonction avec le mot clé **global**.

```
<?php
function quiEstLa() {
    global $nom;
    echo $nom;
}

$nom = 'François';
echo $nom;
quiEstLa();
echo $nom;
?>
```

→ François

→ François

→ François

francois.piat@univ-fcomte.fr

Variables
Mot clé global (2)
PHP 19

**Une variable globale peut être connue dans une fonction avec le mot clé **global**.**

```
<?php

function quiEstLa() {
    global $nom;
    echo $nom;
    $nom = 'Pierre';
}

$nom = 'François';

echo $nom;

quiEstLa();

echo $nom;
?>
```

→ François

→ François

→ Pierre

francois.piat@univ-fcomte.fr

Variables
Variables super - globales
PHP 20

- Tableaux associatifs **prédéfinis** et **gérés** par PHP.

|                   |                  |
|-------------------|------------------|
| <b>\$GLOBALS</b>  | <b>\$_COOKIE</b> |
| <b>\$_POST</b>    | <b>\$_FILES</b>  |
| <b>\$_GET</b>     | <b>\$_SERVER</b> |
| <b>\$_SESSION</b> | <b>\$_ENV</b>    |

- Connues dans le script **ET** dans les fonctions.

francois.piat@univ-fcomte.fr

Variables
Variables globales
PHP 21

**Plutôt qu'utiliser le mot clé `global`,  
utilisez le tableau super-global `$GLOBALS`**

```
<?php

function quiEstLa() {
    global $nom;
    echo $nom;
    $nom = 'Pierre';
}

$nom = 'François';

echo $nom;

quiEstLa();

echo $nom;

?>
```

```
<?php

function quiEstLa() {
    echo $GLOBALS['nom'];
    $GLOBALS['nom'] = 'Pierre';
}

$GLOBALS['nom'] = 'François';

echo $GLOBALS['nom'];

quiEstLa();

echo $GLOBALS['nom'];

?>
```

francois.piat@univ-fcomte.fr

Variables
Constantes
PHP 22

- La valeur d'une constante n'est **pas modifiable** le temps de l'existence du script.

`define('NOM_CONSTANTE', valeur);`

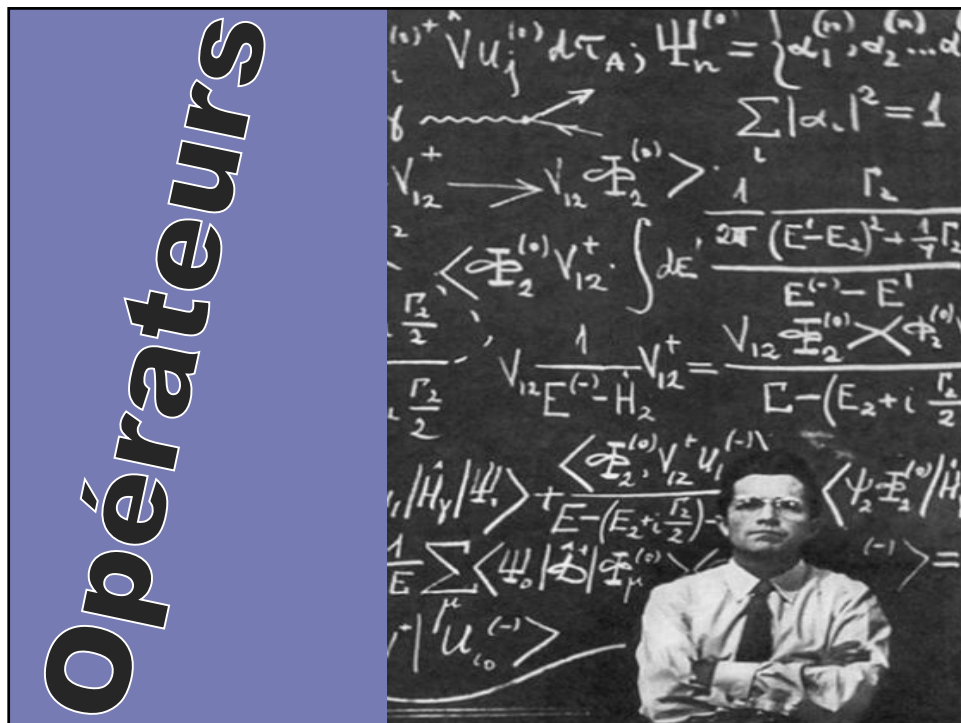
`define('NOM_BD', 'maBase');`  
`mysqli_select_db($conx, NOM_BD);`

`define('DEBUG', TRUE);`  
`if (DEBUG) {`  
 `affiche_erreur();`  
`}`

`define('REPERTOIRE_DU_SITE', realpath(__FILE__));`

- Une constante est connue dans le script **ET** dans les fonctions.

francois.piat@univ-fcomte.fr



Opérateurs
Assignment
PHP 24

**=**

**+=    -=    /=    \*=    %=    . =**

```
$d = 100;
$d += 10;
echo $d;
```

→ 110

`$d += 10;`  
équivalent à  
`$d = $d + 10;`

```
$a = 'Je suis ';
$a .= 'étudiant';
echo $a;
```

→ Je suis étudiant

`$a .= 'étudiant;`  
équivalent à  
`$a = $a . 'étudiant';`

```
$a = $b = $c = 0;
$d = $e = false;
$t1 = $t2 = $t3 = array();
```

francois.piat@univ-fcomte.fr

Opérateurs

Mathématiques

PHP 25

**+   -   \*   /   %**

`$a = 10 + 20 - 2;` → 28 : integer

`$b = $a / 3;` → 9.3333... : float

**Incrément et décrément : ++   --**

`$a ++;` → `$a = $a + 1;`

`$a --;` → `$a = $a - 1;`

francois.piat@univ-fcomte.fr

Opérateurs

Comparaison

PHP 26

**==   !=   <>**

**<   >   <=   >=**

**===   !==**

`if ($nom == 'François') {  
    $type = 'Prof';  
}`

`if ($note > 15) {  
    echo 'Très bonne note';  
}`

francois.piat@univ-fcomte.fr

Opérateurs == et === != et !== PHP 27

|   |   |
|---|---|
| <pre>\$a = 100; \$b = '100'; if (\$a == \$b) {     echo 'égaux'; } else {     echo 'différents'; }</pre> <p>→ égaux</p>       | <pre>\$a = 100; \$b = '100'; if (\$a === \$b) {     echo 'égaux'; } else {     echo 'différents'; }</pre> <p>→ différents</p>       |
| <pre>\$a = 100; \$b = '100 euros'; if (\$a == \$b) {     echo 'égaux'; } else {     echo 'différents'; }</pre> <p>→ égaux</p> | <pre>\$a = 100; \$b = '100 euros'; if (\$a === \$b) {     echo 'égaux'; } else {     echo 'différents'; }</pre> <p>→ différents</p> |

francois.piat@univ-fcomte.fr

Opérateurs == et === != et !== PHP 28

|   |  |
|---|--|
| <pre>\$a = 0; if (\$a == FALSE) {     echo 'perdu'; } else {     echo 'gagné'; }</pre> <p>→ perdu</p> | <pre>\$a = 0; if (\$a === FALSE) {     echo 'perdu'; } else {     echo 'gagné'; }</pre> <p>→ gagné</p> |
|---|--|

Particulièrement important pour tester le résultat de fonctions qui peuvent renvoyer FALSE ou un autre type de données (cf travail sur les chaînes de caractères).

francois.piat@univ-fcomte.fr

Opérateurs

Logique

PHP 29

AND    &&    OR    ||    !

```
if ($a == $b && $c == $d || $c != $f) {
    $texte = "Test trop compliqué";
}
```

```
if (! $b) {
    $texte = 'b est évalué faux';
}
```

```
($a == 10) && uneFonction();
($a != $b) && $c = 12345;
```

francois.piat@univ-fcomte.fr

Opérateurs

Concaténation

PHP 30

Mettre bout à bout des chaînes de caractères : •

```
$a = 'Je suis ';
$b = 'étudiant';
$c = $a.$b;
echo $c;
```

→ Je suis étudiant

```
$a = 123;
$b = 456;
$c = $a.$b;
echo $c;
```

→ 123456

francois.piat@univ-fcomte.fr

Opérateurs

Conditionnel

PHP 31

**(expression) ? FaireSiVrai : FaireSiFaux**

|   |  |
|---|--|
| <code>(\$b == 100) ? foncUn() : foncDeux();</code>        | <pre>if (\$b == 100) {     foncUn(); } else {     foncDeux(); }</pre>                  |
| <code>\$admis = (\$note &gt;= 10) ? 'oui' : 'non';</code> | <pre>if (\$note &gt;= 10) {     \$admis = 'oui'; } else {     \$admis = 'non'; }</pre> |
| <code>\$prof = (\$nom == 'Piat');</code>                  | <pre>if (\$nom == 'Piat') {     \$prof = TRUE; } else {     \$prof = FALSE; }</pre>    |

francois.piat@univ-fcomte.fr

Opérateurs

[ ] et ( )

PHP 32

- Sélectionner un élément dans un tableau avec [ et ].

```
$element = $t[1];
$element = $t['nom'];
```

- Invoquer d'une fonction avec ( et ).

```
$nb = count($t);
maFonction();
```

```
$a = 'maFonction';
$a();
```

francois.piat@univ-fcomte.fr