

PHP



Prezi

FRIER Arnault

CDI 1 2014 - 2015

PHP Avancé !



Rasmus Lerdorf

1968 : 46 ans

à 26 ans création de PHP :
Personnal Home Page : 1994

Une bibliothèque de
fonctions en PERL ...



PHP / FI

Personal Home Page Tools / Form Interpreter



Andi Gutmans

Zeev Suraski

Z E N D
ev A i

From Tel-Aviv

En 1998 : PHP 3

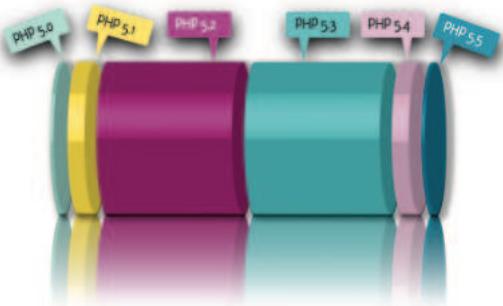
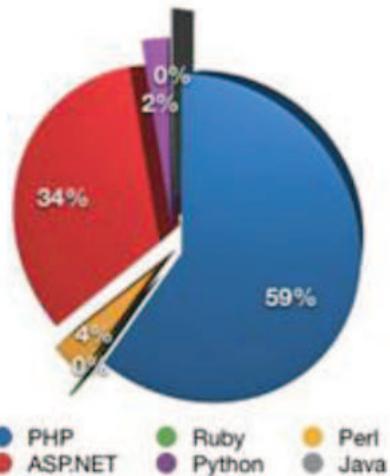
Avec Le ZEND ENGINE

En 2000 : PHP 4

Devient "Php : HyperText Preprocessor"

En 2004 : PHP 5

Avec Le ZEND ENGINE 2 => PHP Objet enfin Abouti !



WORDPRESS



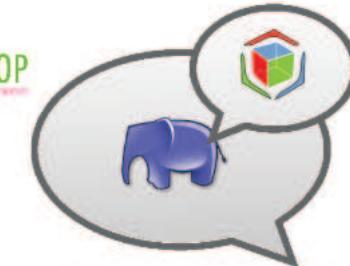
Drupal™



 Symfony



PRESTASHOP
The Simple E-commerce Experience



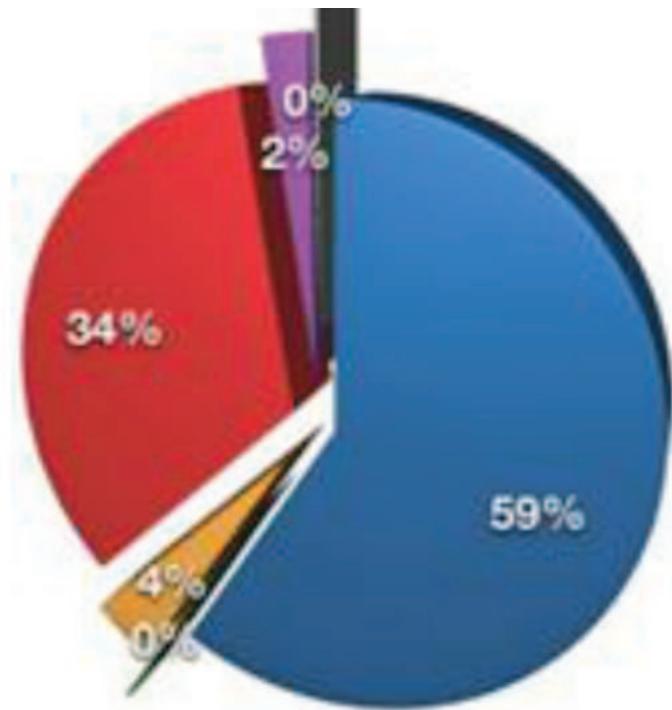
phpBB
creating communities

ZF ZEND
FRAMEWORK



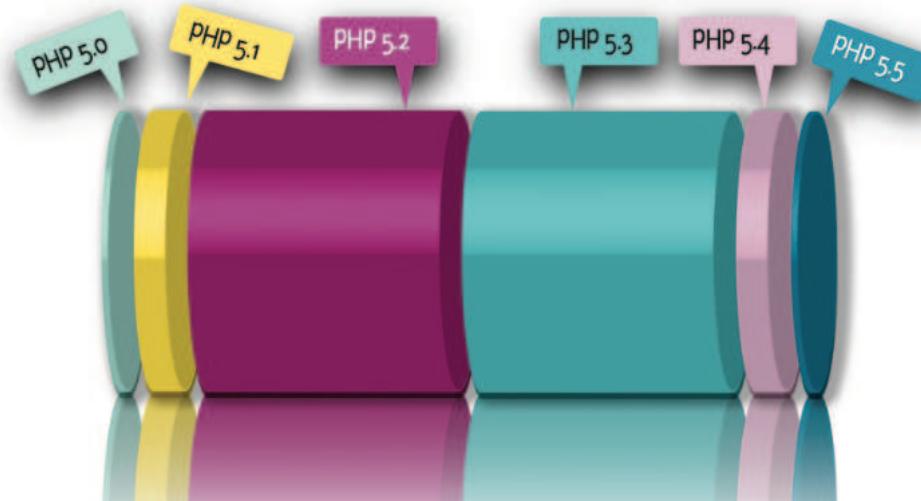
Joomla!



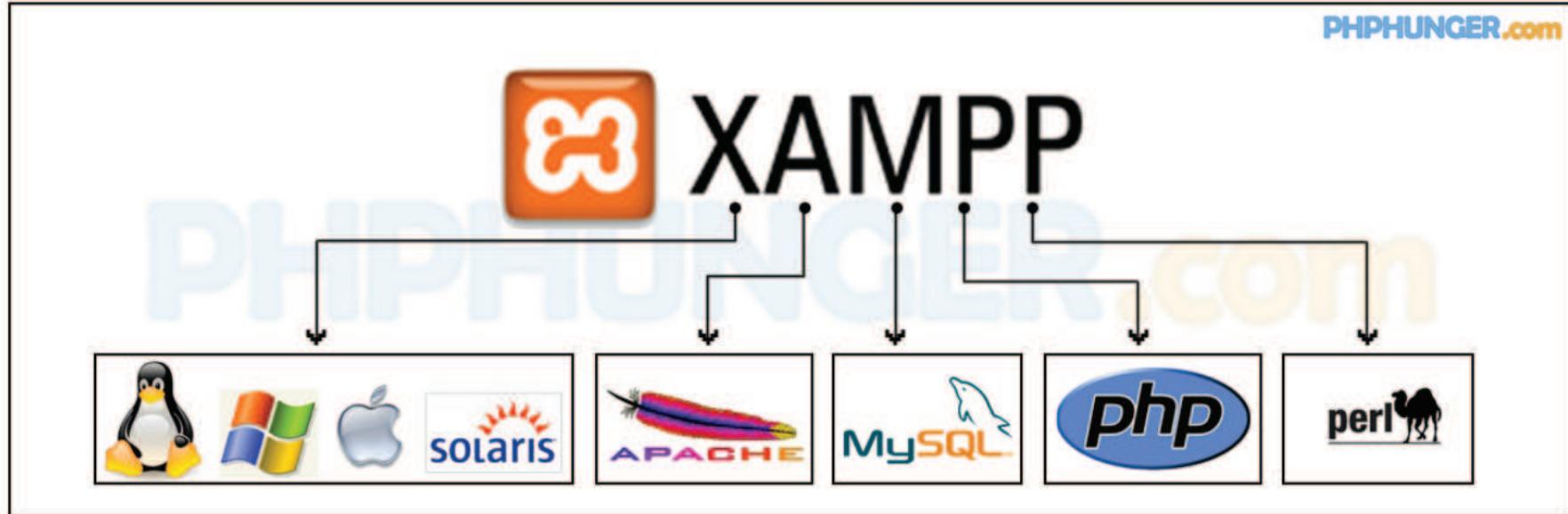


Legend for the pie chart:

- PHP (Blue)
- ASP.NET (Red)
- Ruby (Green)
- Python (Purple)
- Perl (Yellow)
- Java (Grey)







Cheatography

PHP Syntaxe & Fondamentaux - FR Cheat Sheet by Zetura (Zetura) via cheatography.com/18426/cs/1752/

Syntaxe de base			Tableaux			Chaines de caractères (string)		
Code PHP	<?php //Contenu ?>		Création	\$tableau = array();		Délimités entre	"chaine de caractères"	
Commentaire sur une ligne	// Commentaire		Ajout	\$tableau[] = "valeur";		guillemet ou apostrophe	ou 'chaine de caractères'	
Commentaires sur plusieurs lignes	/* Commentaire */		Ajout sur un index	\$tableau[4] = "valeur";		Entre guillement, les variables sont interprétées	\$var = caractères; echo "chaine de \$var";	
Fin d'instruction	;		Ajout sur une clé	\$tableau["cle"] = "valeur";				
Inclusion de fichier	require_once('nom_fichier.php');		Création numérique	\$tableau = array('valeur1', 'valeur2');		Caractère d'échappement	\	
Variables et Constantes			Création associative	\$tableau = array('cle1' => 'valeur1', 'cle2' => 'valeur2');		Retour à la ligne	\n	
\$nomVariable = "Chaine de caractère";			Écriture depuis numérique	echo \$tableau[4];		Retour chariot	\r	
\$nomVariable = 'Chaine de caractère';			Écriture depuis association	echo \$tableau['cle1'];		Tabulation	\t	
\$nomVariable = 5;			Tableaux numériques	Les clés sont des chiffres numériques		Fonction utilisateur		
\$nomVariable = "Une (\$autreVariable) info";			Tableaux associatifs	Les clés sont des chaînes de caractères		function multiplier(\$arg1, \$arg2){		
echo \$nomVariable;			Matrice (tableau multi-dimensions)	\$matrice[2][3] = "valeur";)	return \$arg1 * \$arg2;		
global \$varGlobale;						}		
echo \$GLOBALS['varGlobale']			Classes and Objects			\$param1 = 4;		
define('NOMCONSTANTE', 'valeur');			Types			\$param2 = 8;		
echo NOMCONSTANTE;			Booléen	boolean	true OU false	\$resultat = multiplier(\$param1, \$param2);		
Fonctions sur les variables			Entier	integer	nombre positif ou négatif	class SomeClass {		
Vérifier l'existence de la variable	isset(\$var);		Nombre flottant	float	nombre à virgule positif ou négatif	private \$property;		
Détruire une variable	unset(\$var);		Nombre flottant	double	nombre à virgule positif uniquement	public \$anotherProperty;		
Connaitre le type	gettype(\$var);		Chaîne de caractères	string	chaîne de caractères	protected \$yetAnotherProperty = null;		
Vérifier un type	is_[type](\$var); Ex : is_string(\$var);					public function __construct(\$arg=null)		
Cast (changement de type)	\$var = (string) \$var;					{		
Conversion de valeur	[type]val(\$var); Ex : intval(\$var); \$floatval(\$var);					\$this->property = \$arg;		
Test si la variable est vide	empty(\$var);					}		
						public function someMethod()		
						{		
						echo "Hi";		
						}		
						public function getProperty()		
						{		
						return \$this->property;		
						}		



By Zetura (Zetura)
cheatography.com/zetura/
webmaster-alsace.fr

Published 19th February, 2014.
Last updated 2nd June, 2014.
Page 1 of 3.

Sponsored by [Readability-Score.com](https://readability-score.com)
Measure your website readability!
<https://readability-score.com>

Cheatography

Syntaxe de base

Code PHP

<?php //Contenu ?>

Commentaire sur une
ligne

// Commentaire

Commentaires sur
plusieurs lignes

/* Commentaire */

Fin d'instruction

;

Inclusion de fichier

require_once('nom_fic
hier.php');

Variables et Constantes

```
$nomVariable = "Chaine de caractere";
```

```
$nomVariable = 'Chaine de caractere';
```

```
$nomVariable = 5;
```

```
$nomVariable = "Une {$autreVariable} info";
```

```
echo $nomVariable;
```

```
global $varGlobale;
```

```
echo $GLOBALS['varGlobale']
```

```
define('NOMCONSTANTE', 'valeur');
```

```
echo NOMCONSTANTE;
```

Ecrit

num

Ecrit

assc

Tabl

num

Tabl

assc

Matr

mult

)

Fonctions sur les variables

Vérifier

`isset($var);`

l'existence de la
variable

Détruire une
variable

`unset($var);`

Connaître le type

`gettype($var);`

Vérifier un type

`is_[type]($var);` Ex :
`is_string($var);`

`is_string($var);`

Cast
(changement de
type)

`$var = (string) $var;`

Conversion de
valeur

`[type]val($var); Ex :`
`intval($var);`
`$floatval($var);`

Test si la variable
est vide

`empty($var);`

Tableaux

Création

```
$tableau = array();
```

Ajout

```
$tableau[] = "valeur";
```

Ajout sur un
index

```
$tableau[4] = "valeur";
```

Ajout sur une clé

```
$tableau["cle"] = "valeur";
```

Création
numérique

```
$tableau = array('valeur1',  
'valeur2');
```

Création

```
$tableau = array('cle1' =>
```



Prezi

```
function
```

```
{
```

```
return $a
```

```
}
```

```
$param1
```

```
$param2
```

```
$resultat
```

info";

Création associative

```
$tableau = array('cle1' => 'valeur1', 'cle2' => 'valeur2');
```

Ecriture depuis numérique

```
echo $tableau[4];
```

Ecriture depuis association

```
echo $tableau['cle1'];
```

Tableaux numériques

Les clés sont des chiffres

Tableaux associatifs

Les clés sont des chaînes de caractères

Matrice (tableau multi-dimensions)

```
$matrice[2][3] = "valeur";
```

Types

Booléen

boolean

true OU false

Entier

integer

nombre positif ou
négatif

Nombre
flottant

float

nombre à virgule
positif ou négatif

Nombre
flottant

double

nombre à virgule
positif uniquement

Chaîne de
caractère

string

chaîne de
caractères



Chaines de caractères (string)

Délimités entre "chaine de caractères"
guillement ou ou 'chaine de
apostrophe caractères'

Entre guillement, les \$var = caractères;
variables sont echo "chaine de \$var";
interprétés

Caractère \ d'échappement

Retour à la ligne \n

Retour chariot \r

\t



Fonction utilisateur

```
function multiplier($arg1, $arg2)
{
    return $arg1 * $arg2;
}

$param1 = 4;
$param2 = 8;
$resultat = multiplier($param1, $param2);
```

Classes and Objects

```
class SomeClass {  
    private $property;  
    public $anotherProperty;  
    protected $yetAnotherProperty = null;  
    public function __construct($arg=null)  
    {  
        $this->property = $arg;  
    }  
    public function someMethod()  
    {  
        echo "Hi";  
    }  
    public function getProperty()  
    {  
        return $this->property;  
    }  
}
```

Classes and Objects (cont)

```
public function setProperty( $p )  
{  
    $this->property = $p;  
}  
}  
  
$myObject = new SomeClass( "123" );  
echo $myObject->getProperty(); // 123  
$myObject->property; // ERROR:private
```

Operateurs

Affectation = \$var = 5;

Affectation par référence &= \$nouvelVar
&= \$var;

Addition + \$var = \$var + 5;

Soustraction - \$var = \$var - 5;

Multiplication

*

```
$var = $var *  
5;
```

Division

1

```
$var = $var /  
5;
```

Modulo

%

```
$var = $var %  
5;
```

Incrémentation

+

\$var =
\$var++

Décrémation

—

`$var = $var--;`

`$var++;`

Décrémentation

`--`

`$var = $var--;`

Opérateurs
combinés

[opérateur]
`=`

`$var += 5;`
`$var *= 5;`

Concaténation

`.`

`echo $var."`
`chaine";`

Concaténation et
assignation

`.=`

`$var .= "`
`chaine";`

assignation

chaine";

Operateurs de comparaison

Egalité == if(\$var == 5)

Inférieur à < if(\$var < 5)

Operateurs de comparaison (cont)

Inférieur ou égal à \leq if(\$var \leq 5)

Supérieur à $>$ if(\$var > 5)

Supérieur ou égal à \geq if(\$var \geq 5)

Différent de \neq if(\$var \neq 5)

5)

Différent de

`!=` `if($var !=
5)`

Strictement égal (valeur et
type)

`==` `if($var ==
5)`

Différent en valeur ou en
type

`!==` `if($var !==
5)`

UtilisÃ©s pour les structures conditionnelles

Operateurs logiques

Inversion	!	Retourne true si false, et inversement
ET	&&	Retourne true si 2 conditions à true
Un seul	^	Retourne true si une seule des conditions à true

OU		Retourne true si une condition à true
ET non prioritaire	AND	Similaire à && mais moins prioritaire
Un seul non prioritaire	XOR	Similaire à ^ mais moins prioritaire
OU non prioritaire	OR	Similaire à mais moins prioritaire



Fonctions PHP utiles

Récupérer une partie
d'une chaîne

`substr($string, start,
length);`

Transformer une chaine
en tableau

`explode(',', $string);`

Concaténer un tableau
en chaine

`implode(',',
$tableau);`

Fonctions PHP utiles (cont)

Retirer les espaces
au début et à la fin
d'une chaîne

`trim($string);`

Remplacer à par b
dans une chaîne

`str_replace('a', 'b',
$string);`

Vérifier une
expression régulière

`preg_match('regex',
$string);`

Remplacer une
expression régulière

`preg_replace('regex',
'b' $string);`

expression régulière

\$string),

Remplacer une
expression régulière
par b

```
preg_replace('regex',  
'b', $string);
```

Arrêter le script PHP

```
exit();
```

Envoyer un mail

```
mail($mailDest, $sujet,  
$message, 'From:  
' . $mailEnvoi);
```



Expression régulière

Expression régulière

^

Début de chaîne

\d

Chiffre entre 0 et 9

\w

Caractère alphanumérique [0-
9A-Za-z]

\s

Espace

.

N'importe quelle lettre, chiffre ou
espace

()

Groupe

[]

Classe de caractères

{x} {x,}

Quantité = x | Supérieur ou égal à x |

{x,y}

Entre x et y

*

Quantité de 0 ou plus

?

Quantité de 0 ou 1

+

Quantité de 1 ou plus

|

OU

\

Caractère d'échappement

Exemple pour une syntaxe de mail :

`^[\w{.-\+}]+@[\\w.-]+\.[a-zA-Z]{2,6}\$`



Structure conditionnelle : IF

```
if (condition) {  
    // Instructions  
}  
  
elseif (condition) {  
    // Instructions  
}  
  
else {  
    // Instructions
```

```
}
```

```
if( $something == true ) {  
    // Si $something vaut true  
    doSomething();  
}  
elseif ( $something == false ) {  
    // Si $something vaut false  
    doSomethingElse();  
}  
else {  
    // sinon, exécuter doNothing();  
    doNothing();
```

```
doNothing();  
}  
  
if(condition):  
    // Instructions  
endif;  
  
(condition)? instructions si true : instructions si  
false;
```

Structure conditionnelle : SWITCH

Structure conditionnelle : SWITCH

```
switch ($var) {  
    case 1:  
        // Instructions  
        break;  
    case "test":  
        // Instructions  
        break;  
    default:  
        // Instructions  
        break;  
}
```

Peut ãtre utilisÃ© avec des chiffres ou chaÃ®nes de caractÃ“res

Boucle WHILE

Tant que la condition est vrai, l'instruction est exécutée

```
while(condition){
```

```
// Instructions
```

```
}
```

```
$i = 1;
```

```
while($i < 10){
```

```
    echo $i;
```

```
    $i++;
```

```
$i++;
```

```
}
```

Exécution au moins une première fois

```
$i = 1;
```

```
do{
```

```
echo $i;
```

```
$i++;
```

```
}
```

```
while($i < 10);
```

Attention aux boucles infinies

Boucle FOR

Exécute la première expression lors de l'initialisation, puis tant que la condition est valide, exécute le contenu de la boucle et fini en exécutant la dernière expression

```
for(expression1; condition; expression2){
```

```
// Instructions
```

```
}
```

```
for($i = 1; $i < 10; $i++){
```

```
// Instructions
```

Boucle FOREACH

A chaque itération dans la boucle assigne la valeur de l'élément courant à la variable et le pointeur interne du tableau est avancé d'un élément.

```
foreach($tableau as $element){  
    // Instructions
```

Boucle FOREACH (cont)

```
}

foreach($tableau as $key => $value){

    // Instructions

}

$tableau = array(
    'prenom' => 'Obi-wan',
    'nom' => 'Kenobi',
    'metier' => 'Jedi'
);

foreach($tableau as $contenu){

    echo "Valeur : $contenu<br/>";
```



```
echo "Valeur : $contenu<br/>";  
}  
  
foreach($tableau as $cle => $valeur){  
echo "Clé : $cle -> Valeur : $valeur<br/>";  
}
```

Attention, la boucle fonctionne sur une copie du tableau spécifiée, pas sur le tableau lui-même



Prévi

CONTINUE

CONTINUE

```
for ($i = 0; $i < 5; ++$i) {  
    if ($i == 2)  
        continue;  
    print "$i , ";  
}
```

produces the following output:

0 , 1 , 3 , 4