Introduction à PHP

Bases du langage

monnerat@u-pec.fr

5 avril 2019

IUT de Fontainebleau

Sommaire

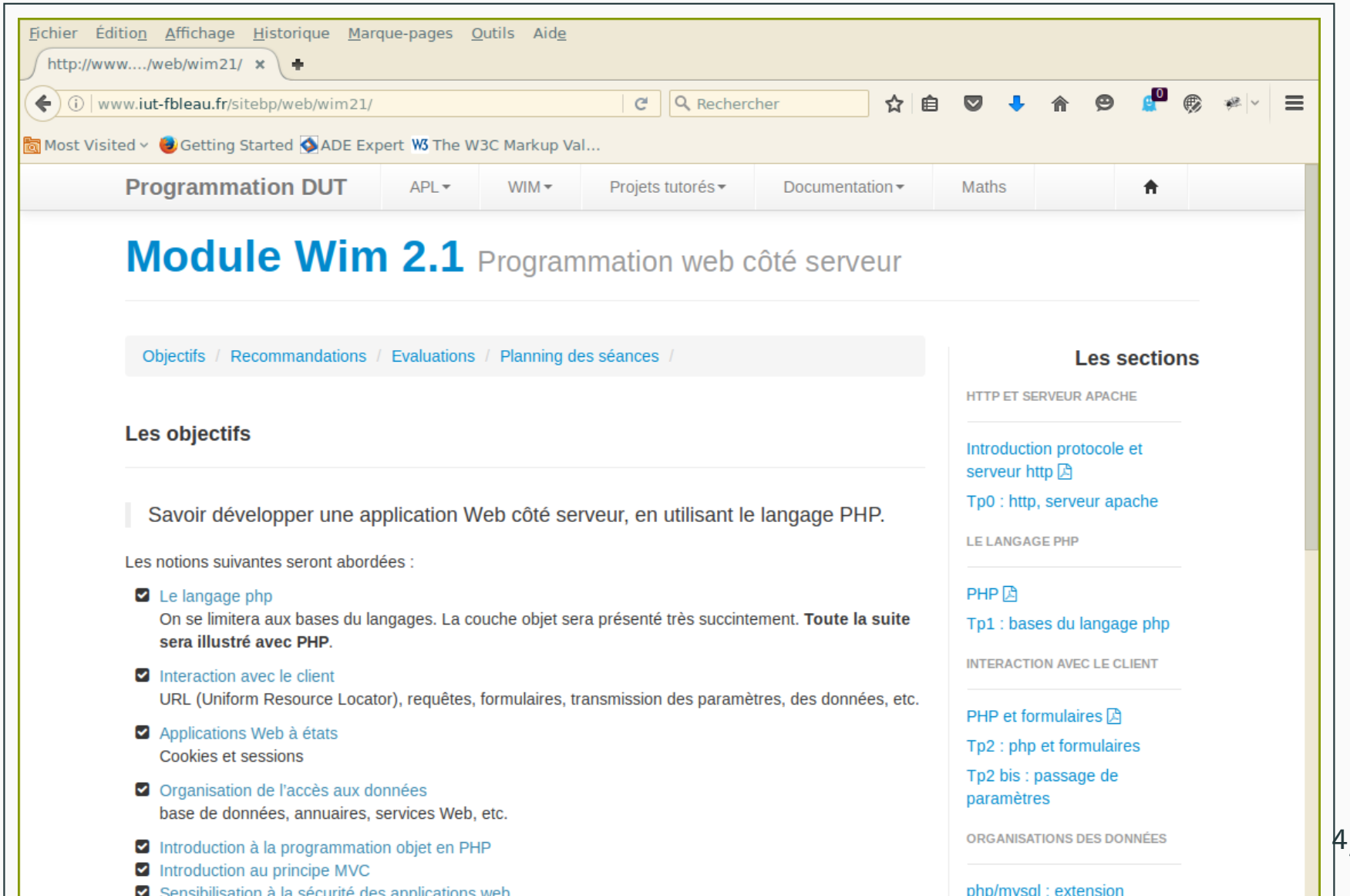
Introduction

Quelques liens

- PHP
  - http://www.php.net/.
  - http://www.phpinfo.net/.
  - http://www.phpfrance.com/.
  - http://www.developpez.com/php/
- MySQL
  - http://www.mysql.com
  - http://dev.nexen.net/docs/mysql
- HTML, CSS, Javascript, XML, etc.
  - https://developer.mozilla.org/fr/
  - http://www.w3schools.com

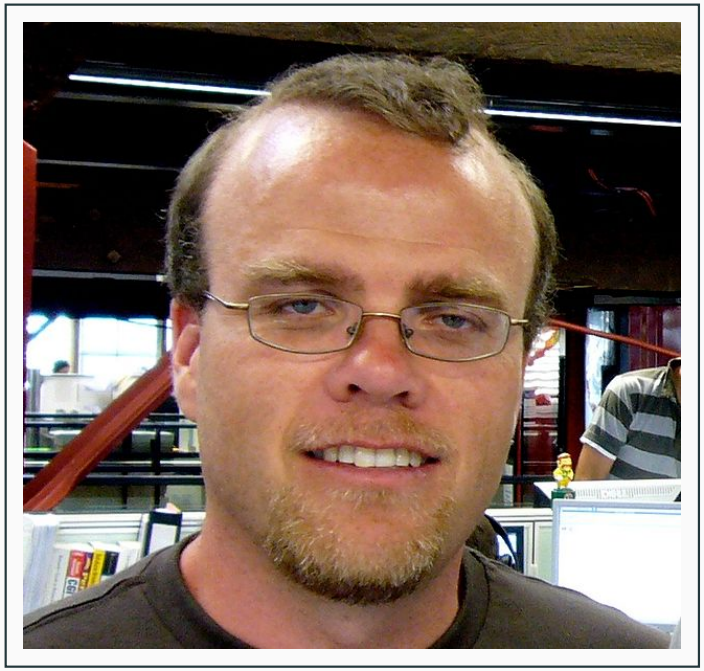
Cours et supports à l'iut

http://www.iut-fbleau.fr/sitebp/web/wim21/



Plan : 1 - Introduction

Un peu d'histoire



La première version été créée en 1994 par Rasmus Lerdorf pour les besoins des pages web personnelles (livre d'or, compteurs, etc.). A l'époque, PHP signifiait **Personnal Home Page**

C'est un langage incrusté au HTML et interpreté (PHP3) ou compilé (PHP4,5,7) côté serveur :

- Il dérive du `C` et du `Perl` dont il reprend la syntaxe.
- Il est extensible grâce à de nombreux modules et son code source est ouvert.
- Connection à la majorité des SGBD. (MySql, Oracle, postgres, ODBC, etc.)
- Prise en charge de nombreux protocoles et formats : Tcp, Http, Smtp, Ldap, Imap, Pop, SSl, Soap, XML, etc.
- Comme il supporte tous les standards du web et qu'il est gratuit, il s'est rapidement répandu sur la toile.

C'est un langage incrusté au HTML et interpreté (PHP3) ou compilé (PHP4,5,7) côté serveur :

- Il dérive du `C` et du `Perl` dont il reprend la syntaxe.
- Il est extensible grâce à de nombreux modules et son code source est ouvert.
- Connection à la majorité des SGBD. (MySql, Oracle, postgres, ODBC, etc.)
- Prise en charge de nombreux protocoles et formats : Tcp, Http, Smtp, Ldap, Imap, Pop, SSl, Soap, XML, etc.
- Comme il supporte tous les standards du web et qu'il est gratuit, il s'est rapidement répandu sur la toile.

Application WEB ?

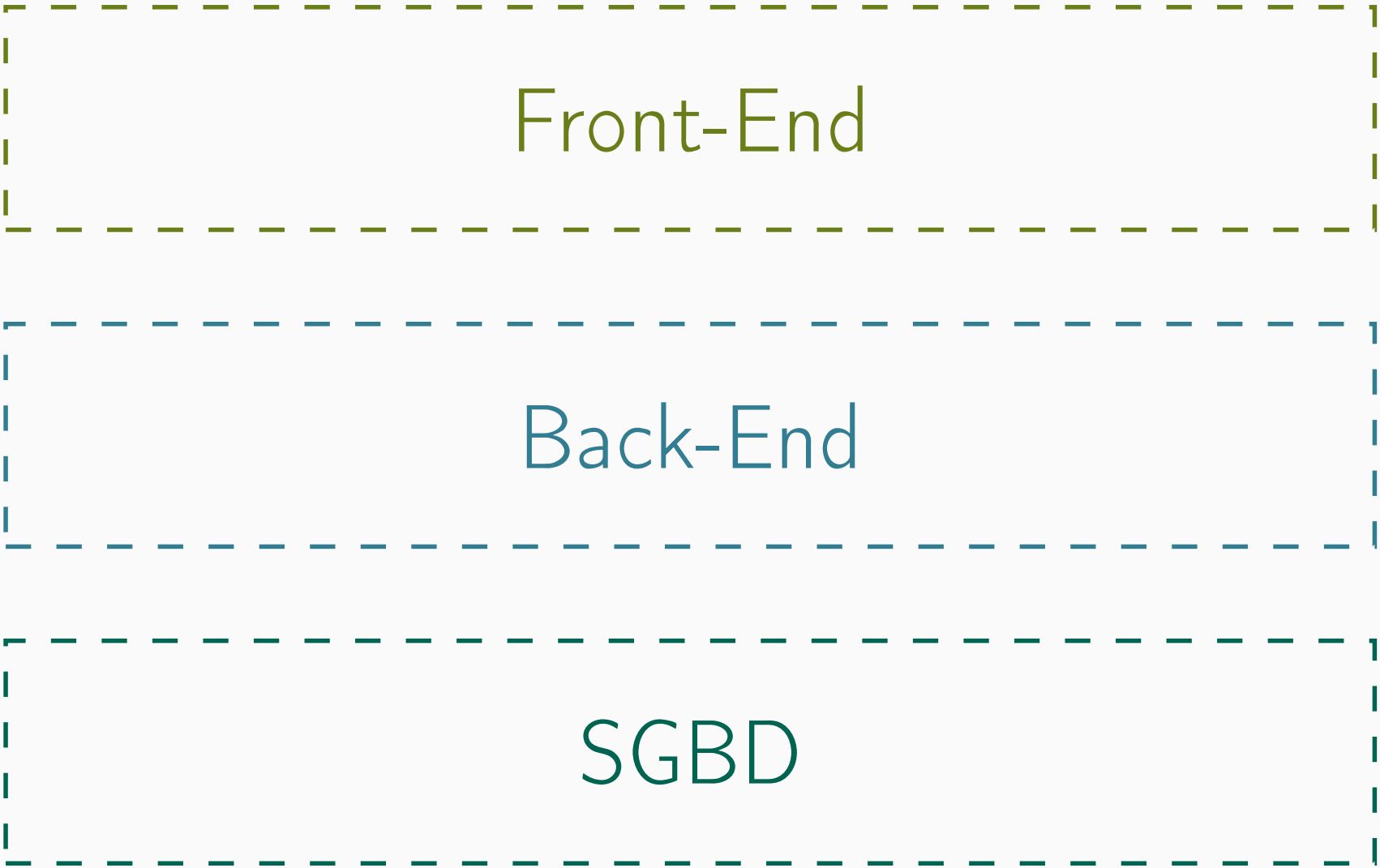
Avant de rentrer dans la syntaxe proprement dite du PHP, examinons le contexte !

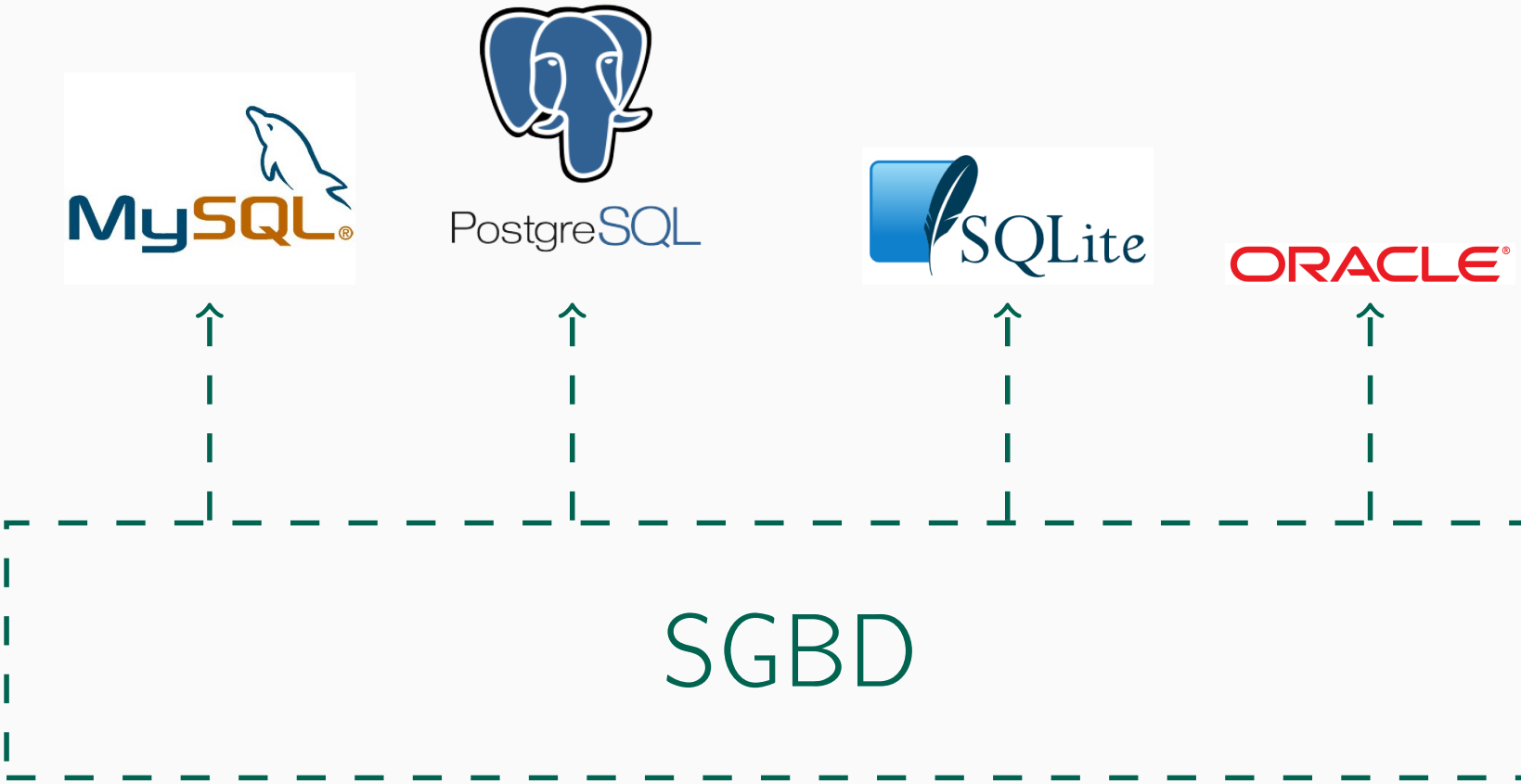
Plan : 1 - Introduction

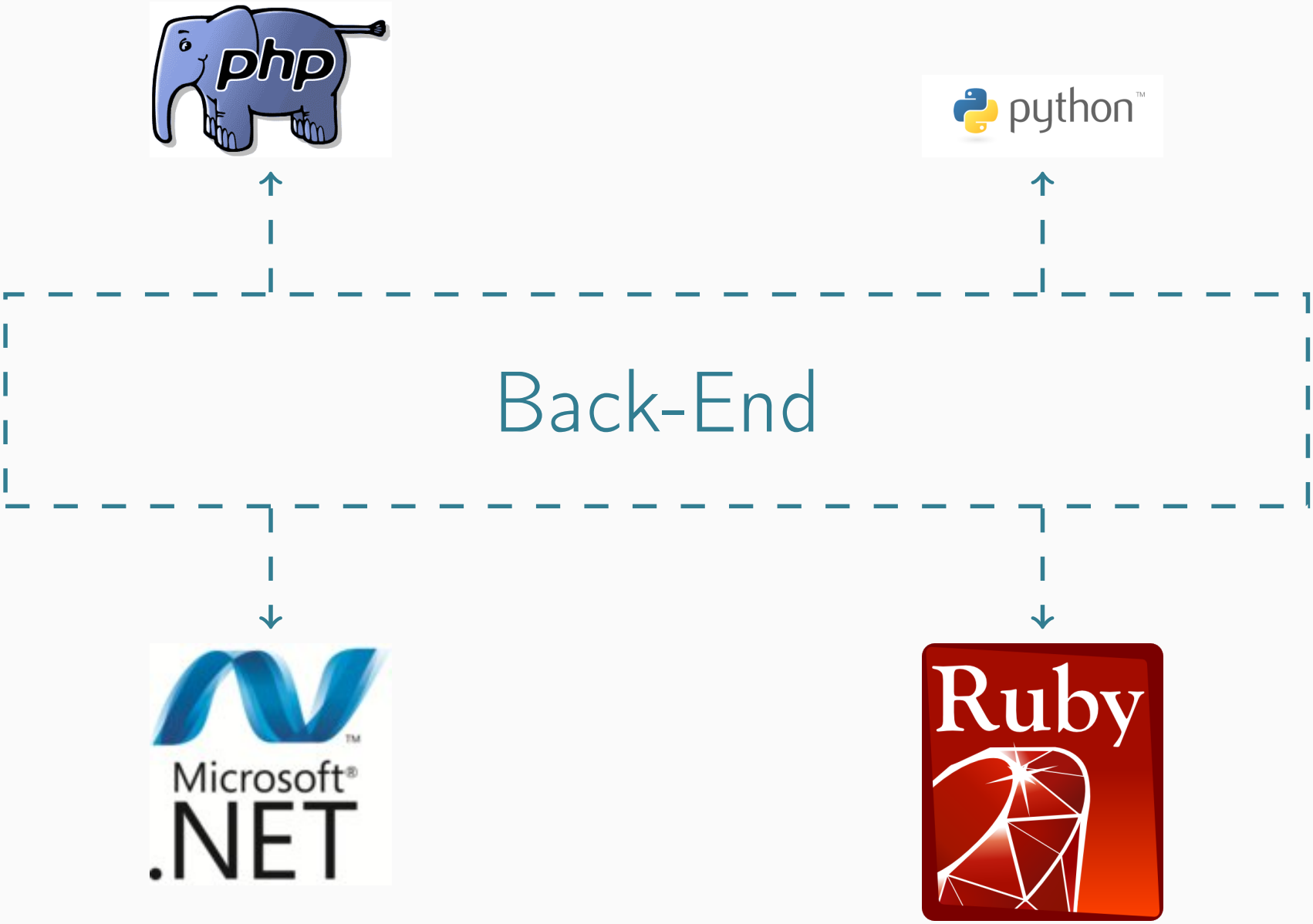
Une application WEB ?

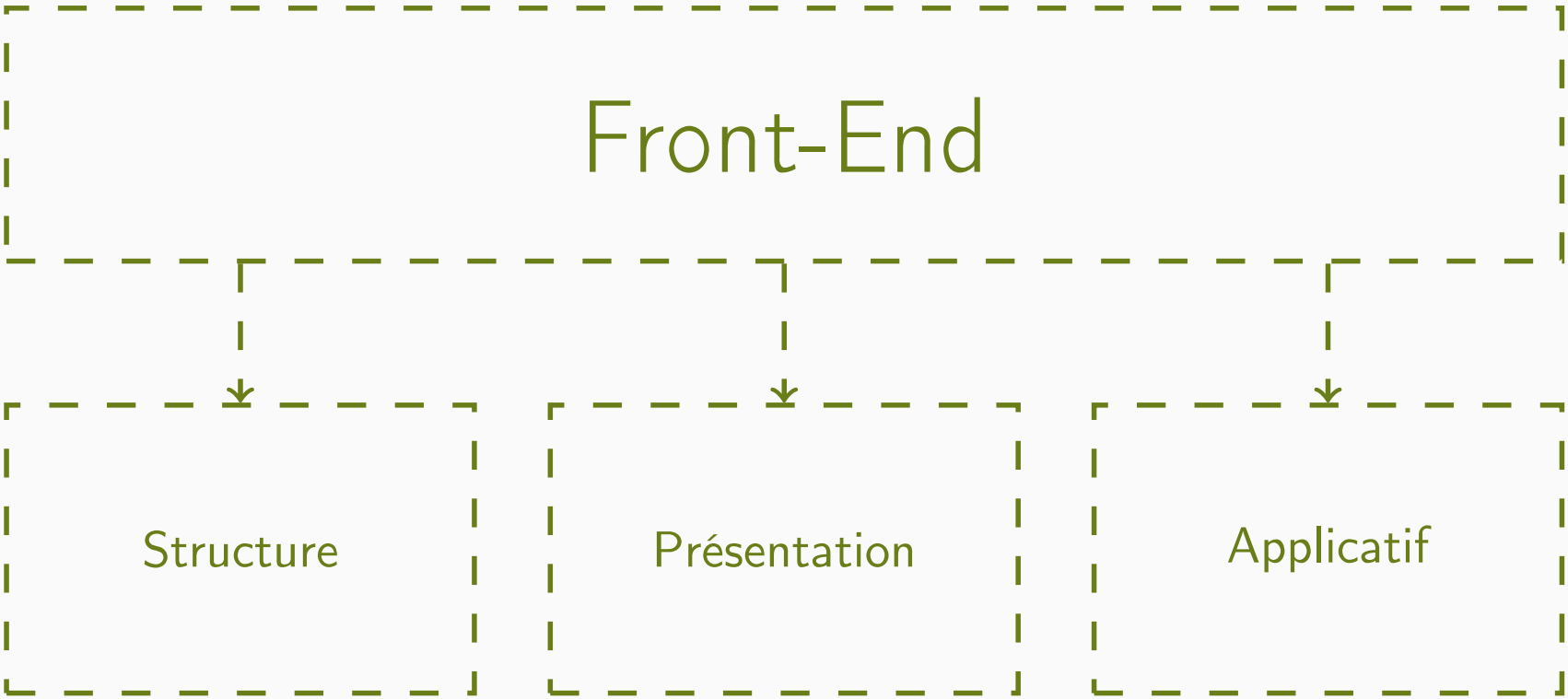
Une application Web est une application clients/serveur(s) sans états.

On peut la voir en trois couches ...


Front-End
Back-End
SGBD


MySQL
PostgreSQL
SQLite
ORACLE
SGBD


php
python
Back-End
Microsoft
.NET
Ruby


Front-End
Structure
Présentation
Applicatif

Front-End

Présentation

Applicatif

Front-End

Applicatif

Front-End

Client requête sur `http://www.arda/bonjour.php`

Client requête sur `http://www.arda/bonjour.php`

Serveur

Client requête sur `http://www.arda/bonjour.php`

Serveur

Client requête sur `http://www.arda/bonjour.php`

Serveur

fichier

```
<html>
  <body>
<?php
echo "<b>bonjour</b>";
?>
  </body>
</html>
```

Client requête sur `http://www.arda/bonjour.php`

Serveur

fichier

```
<html>
  <body>
<?php
echo "<b>bonjour</b>";
?>
  </body>
</html>
```

Intéprétation

Client requête sur `http://www.arda/bonjour.php`

Serveur

fichier

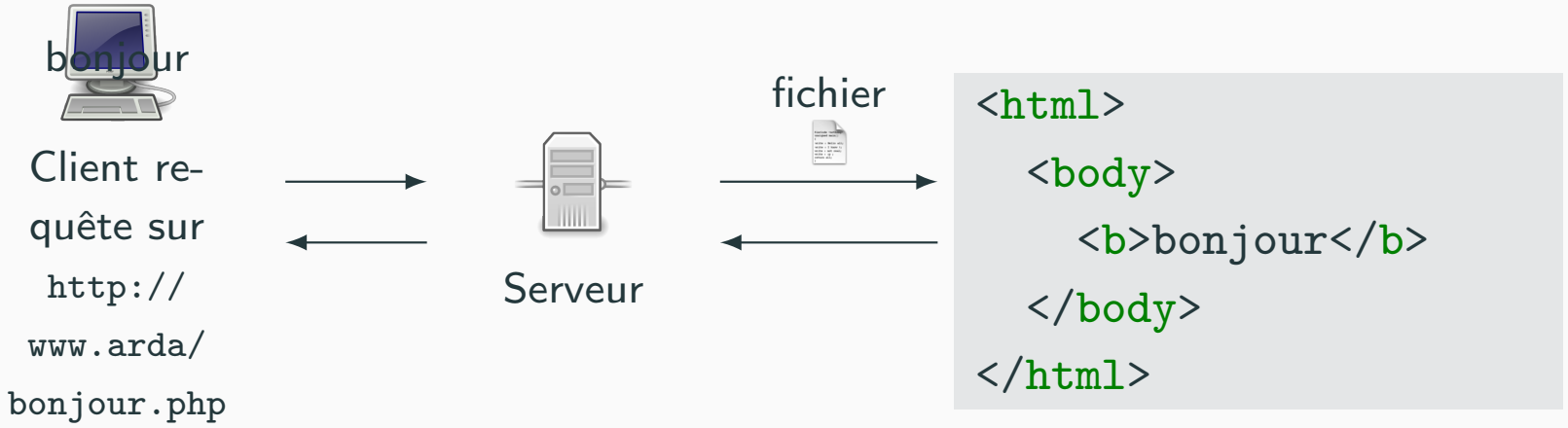
```
<html>
  <body>
    <b>bonjour</b>
  </body>
</html>
```

Client requête sur `http://www.arda/bonjour.php`

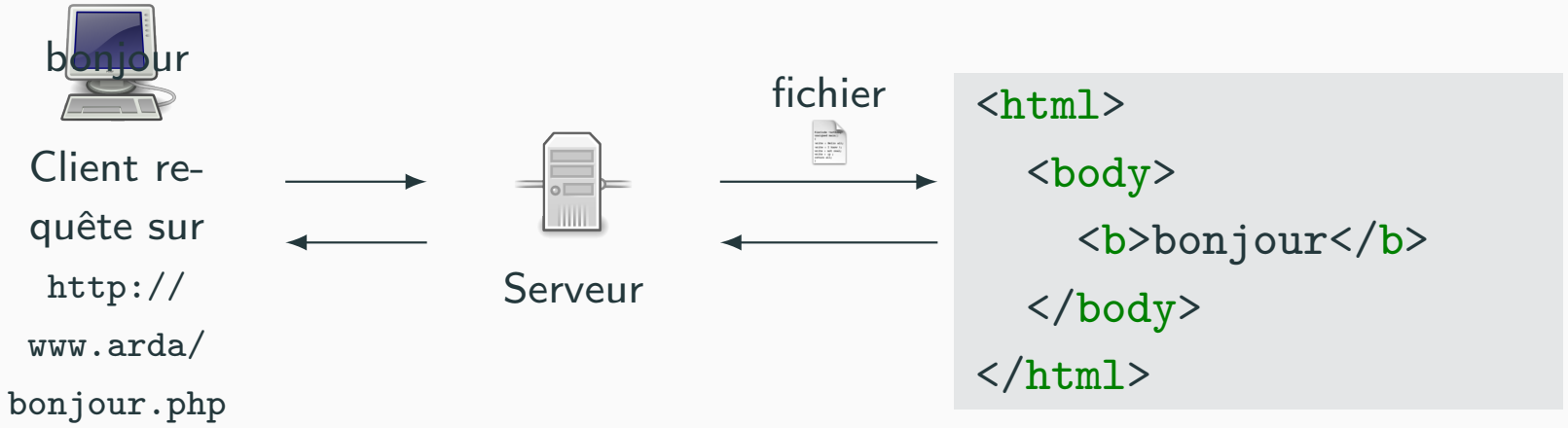
Serveur

fichier

```
<html>
  <body>
    <b>bonjour</b>
  </body>
</html>
```

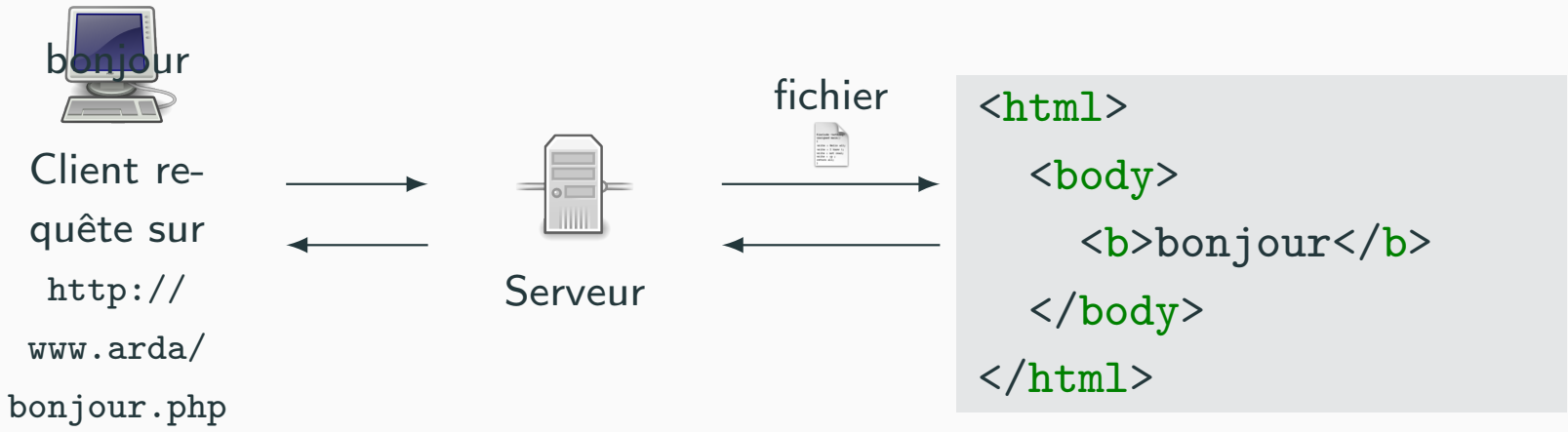


L'intégration de l'interpréteur PHP dans le serveur Web peut se faire de deux manières :



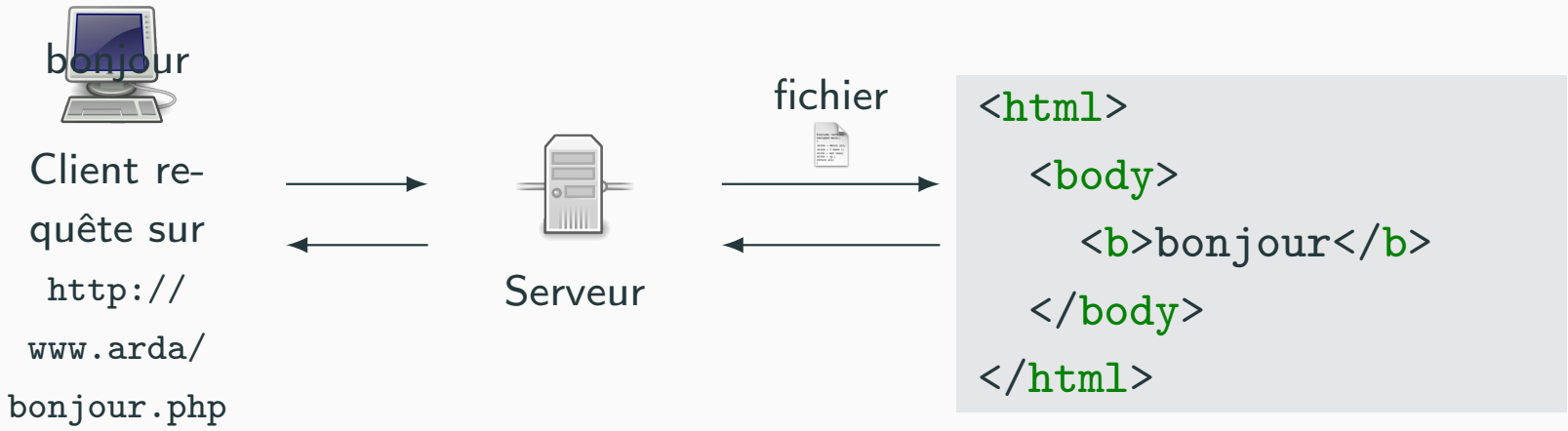
L'intégration de l'interpréteur PHP dans le serveur Web peut se faire de deux manières :

- comme module du serveur : l'interpréteur est directement intégré dans le serveur web. C'est le processus serveur qui interpréte le code.



L'intégration de l'interpréteur PHP dans le serveur Web peut se faire de deux manières :

- comme module du serveur : l'interpréteur est directement intégré dans le serveur web. C'est le processus serveur qui interpréte le code.
- comme un programme CGI (Common Gateway Interface) : le serveur lance un processus annexe, qui de façon externe interprète le code php. Le serveur récupère alors le résulat.



L'intégration de l'interpréteur PHP dans le serveur Web peut se faire de deux manières :

- comme module du serveur : l'interpréteur est directement intégré dans le serveur web. C'est le processus serveur qui interpréte le code.
- comme un programme CGI (Common Gateway Interface) : le serveur lance un processus annexe, qui de façon externe interprète le code php. Le serveur récupère alors le résulat.
- L'exemple précédent est crétin....

Adressage des documents

Solution

- URL : *Uniform Resource Locator* = adressage universelle de ressources.
- 3 parties : le protocole (comment), le nom (où) et le nom du document (quoi).
- URL$\subset$ URI *Universal Resource Identifier*.

Exemples

- `http://www.iut-fbleau.fr/sitebp`
- `https://www.google.fr:8888/img/plan.php?x=12&y=20`
- `ftp://user:password@www.iut-fbleau.fr/account/`

Composition d'une URL

`protocole://hostname:port/path/extrapath?arguments`

- La racine / de path est définie par la configuration du serveur Web. (rien à voir a priori avec la racine du système de fichier)
- path peut contenir un point d'ancrage.
- extrapath et arguments permettent de passer des informations à un programme qui s'exécutent sur le serveur (script php par exemple).

Clients/Serveurs

- Client (souvent) : un navigateur qui fait des requêtes sur un serveur qui interprète le php.

Clients/Serveurs

- Client (souvent) : un navigateur qui fait des requêtes sur un serveur qui interprète le php.

- Serveur(s) :

HTTP(S)	pour les pages HTML et PHP : `Apache`, `nginx`, `hiawatha`
SGBD	serveur sql : `mysql`, `oracle`, `Postgres`
LDAP	OpenLdap (authentification)

On peut aussi utiliser php indépendamment d'un serveur web, comme un langage de script à part entière :

interpréteur `php`

```
php -f helloworld.php
```

fichier exécutable

```
#!/usr/bin/php
<?php
echo "hello world !!";
?>
```

A l'iut ...

- Un serveur http(s) (Apache) sur la machine `dwarves.arda` (dwarves.iut-fbleau.fr)
  qui sert le répertoire `public_html` à votre racine, via l'url

  `http://dwarves.arda/~votre_login`

  Ou bien

  `http://dwarves.iut-fbleau.fr/~votre_login`

  Ou bien

  `https://dwarves.iut-fbleau.fr/~votre_login`

  dans lequel se trouve un fichier `.htaccess` qui demande une authentification via ldap.

- Des serveurs de bases de données :
  - `mysql` sur dwarves,
  - `Oracle` sur lorien.

Le langage

Intégration de php à html

Les pages web sont au format html/xhtml. Les pages web dynamiques sont au format php sur le serveur. Le code source php est directement insérer dans le flux html grâce au conteneur de la norme XML : `<?php ... ?>`

```
<!DOCTYPE html>
<html>
  <head>
    <title>hello world en php !</title>
  </head>
  <body>
    <?php
    echo "<b>hello world !<b>"; /* original !!  */
    ?>
  </body>
</html>
```

Autre syntaxe d'intégration :

- `<? ... ?>`
- `<script language="php"> ... </script>`
- `<% ... %>`

Plan : 2 - Le langage

Variables et types

- Le typage implicite et dynamique en php. Il n'est donc pas nécessaire de déclarer leur type au préalable ni même de les initialiser avant leur utilisation.

Type	Description	Exemple
`NULL`	une seule valeur	`NULL`
`boolean`	booléen	`True/False`
`integer`	entier	`1234,056,0xaf23c`
`double`	réel	`1.234,1.2e3`
`string`	chaîne de caractères	`"toto"`
`array`	tableau	`[1234,"denis"]`
`object`	objet	

- Les identificateurs de variable sont précédés du symbole $ (dollars). Exemple : `$toto`.

- C'est le "contexte" qui détermine le type d'une variable ou d'une expression.

Types et variables

```php
<?php
$a;          // a est de type NULL
$a=1;        // a est un entier
$a=1.2;      // a est un double
$a="toto";   // a est une chaine
?>
```

- On peut caster comme en C.
- Les fonctions
  - `var_dump` affiche les informations structurées d'une variable.
  - `print_r` affiche des informations lisibles pour une variable.
- Les nombres entiers, comme en C, peuvent être représentés en décimal (`$a=153`), en octal (`$a=015`) ou en hexadécimal (`$a=0xa3`).

```php
$a=array(1.5,"aaaa",true,10=>0x14);
print_r($a);
var_dump($a);
```

var_dump

```
array(4) {
[0]=>
float(1.5)
[1]=>
string(4) "aaaa"
[2]=>
bool(true)
[10]=>
int(20)
}
```

print_r

```
Array
(
[0] => 1.5
[1] => aaaa
[2] => 1
[10] => 20
)
```

Variables et types

Fonction	Description
`empty($var)`	renvoie vrai si la variable est vide
`isset($var)`	renvoie vrai si la variable existe
`unset($var)`	détruit une variable
`gettype($var)`	retourne le type
`settype($var,"type")`	convertit la variable (cast)

- On a aussi les fonctions (d'interrogations) : `is_long()`, `is_double()`, `is_string()`, `is_array()`, `is_object`, `etc.`
- Chaîne comme identificateur `${$var}=valeur`

```php
$toto="annee";
${$toto}=2007;
echo $annee; // $annee vaut 2007
```

Portée

Pour la majorité des variables, la portée concerne la totalité d'un script PHP. Mais

- Une variable locale à une fonction n'est pas connue dans le reste du programme. Tout comme une variable du programme n'est pas connue dans une fonction.
- Il est possible néanmoins d'avoir recours à des variables globales (cf fonctions)

Les variables ont une **vie temporaire**. Une fois "la page affichée" (la réponse calculée), elles cessent d'exister. (pas tout à fait vrai avec la notion de variables de sessions)

Plan : 2 - Le langage

Chaînes de caractères

Elles sont généralement délimitées par des guillemets.

```php
<?php
echo  "Super le <b>php</b> !!!";
?>
```

Comme en Perl (ou Shell), ce qui est précédé par $ est expansé ; c'est à dire remplacé par sa valeur.

```php
<?php
$a=10;$b=20;$c=$a+$b;
echo "la somme de $a et de  $b vaut $c";
?>
```

Comme en C, on peut protéger certains caractères avec le caractère d'échappement \.

```php
<?php
echo "je mets des  \$var et des guillemets \"";
echo "et  des antislashs \\";
?>
```

On peut protéger l'ensemble de la chaîne en utilisant des quotes (il faut alors protéger la quote elle-même si besoin).

```php
<?php
echo 'je mets ce que je veux : \  \n  $var';
?>
```

Traitement des chaînes

Afficher une chaîne :

- `echo`
- `printf` : même syntaxe qu'en C

```php
printf('%03d %s %03.4f',1,'bonjour',1.0);
```

```php
$texte='Super le PHP';
echo $texte{2};
echo $texte[0]; // depuis php5
echo ord('A');
echo chr(68);
```

Fonctions sur les chaînes

Il en existe beaucoup. Se réferer à la documentation du langage.

Protection et échappement (Injection)

- ⇝ SQL :
  - La fonction `addslashes` permet de protéger automatiquement les guillemets, quote et antislashes.
  - La fonction `stripslashes` fait l'opération inverse.

  Il existe cependant de meilleures alternatives suivant la base et l'api utilisée :
  - `mysqli_escape_string` pour mysql,
  - `pg_escape_string` pour postgres, etc.
- ⇝ HTML : Les fonctions `htmlentities` et `htmlspecialchars` permettent de convertir les caractères spéciaux en entités HTML.

Plan : 2 - Le langage

Les tableaux

Deux types d'indexages : numérique ou alphabétique.

```php
$tableau=array(10,12,13,$variable,10=>"bonjour",1.2);
echo $tableau[1];
```

Quel est l'indice de `1.2` ?

```php
$individu=array(
  "prenom"=>"Michel",
  "ville"=>"Fontainebleau",
  "code postale"=>"77100");
echo $individu['prenom'];
```

On peut mixer les deux.

PHP dispose d'instructions pour parcourir les éléments d'un tableau.

```php
/* seulement les valeurs */
foreach ($arr as $value) {
  echo "Valeur : $value<br />\n";
}
/* valeurs et clefs      */
foreach ($arr as $key => $value) {
  echo "Clef : $key Valeur : $value<br />\n";
}
/* modification des valeurs */
foreach ($arr as &$value) {
  $value=$value+2;
}
```

Il existe de nombreuses fonctions dédiées à la gestion des tableaux.

Plan : 2 - Le langage

Constantes

La fonction `define` permet de définir des chaînes constantes.

`define("chaineconstante",valeurconstante)`

```php
define ("PI", 3.14);
define ("JAUNE", "#FFFF00");
define ("UPEC", "Université Paris Est Créteil");
//exemple d'utilisation
echo ("Le perimetre du cercle est :" . 2*PI*$rayon);
echo ("<Body Bgcolor=" . JAUNE . ">");
```

- Les constantes ne sont préfixées par $.
- Il est fréquent de construire des fichiers qui ne contiennent que des constantes pour gérer des paramètres de configuration ou de traduction de manière centralisée.

Plan : 2 - Le langage

Opérateurs arithmétiques

Ce sont les mêmes qu'en langage C.

`$a + $b`	addition
`$a - $b`	soustraction
`$a * $b`	produit
`$a / $b`	division
`$a % $b`	division
`$a ++ , ++$a`	post,pre incrémentation
`$a-- , --$a`	décrémentation

```php
echo '<table>';
for ($i=0;$i<10;$i++){
  $class = ($i % 2) ? "impair" : "pair" ;
  echo "<tr class='$class'><td>$i</td></tr>";
}
echo '</table>';
```

Opéateurs logiques et binaires

Opérateurs booléens

`||` **ou** `OR`	OU logique
`&&` **ou** `AND`	ET logique
`XOR`	OU exclusif

Opérateurs bit à bit

`|`	OU bit à bit
`&`	ET bit à bit
`^`	OU exclusif bit à bit
`~`	Négation bit à bit
`«, »`	Décalages

Concaténation

Il s'agit de l'opérateur _._. Il permet de concaténer deux chaînes de caractères.

```php
<?php
$a="Hello";
$b="world !!";
echo $a.' '.$b;

$table="etudiant";
$login="toto";
$req="select * from ".$table." where login='$login'";

echo  "Nous sommes le ".date('G \h i').".";
?>
```

Relation de comparaisons

==	relation d'égalité (après transtypage=
<,<=	inferiorité stricte, infériorité
>,>=	supériorité stricte, supériorité
!=	différence
===	identité : mêmes valeurs et mêmes types
!==	valeurs différentes, ou types différents

Depuis PHP7, on a un nouvel opérateur <=> (space ship)

```php
<?php
// Entiers
echo 1 <=> 1; // 0
echo 1 <=> 2; // -1
echo 2 <=> 1; // 1
?>
```

Opérateurs d'assignation

L'opérateur d'affectation est "=".

```php
$s="bonjour";
$a = ($b = 4) + 5;
// $a est maintenant égal à 9, et $b vaut 4.
```

Tout comme en langage C, si $op \in \{+, -, *, /, \&, \%, |, .\}$:

```php
$a = 3;
$a += 5;
$b = "Bonjour ";
$b .= " tout le monde!";
```

$a op =$b équivaut à $a=$a op $b

Plan : 2 - Le langage

Instructions

Les instructions d'un script php se termine (généralement) par un `;`.

```php
<?php
    echo 'Ceci est un test';
?>
```

De plus, plusieurs instructions peuvent être regroupées en bloc, délimité par des accolades ("{}"). Un bloc est considéré comme une instruction.

```php
{
  $b = "Bonjour ";
  $b .= " tout le monde!";
}
```

Conditionnelles

Simple

```
if (booléen) instruction ou {instructions}
```

Double

```
if (booléen) instruction ou {instructions}
else instruction ou {instructions}
```

Multiple

```php
$nombre=mt_rand(0,4);
switch($nombre){
case 0,2,4 :  echo "nombre pair";break;
default    :  echo "nombre impair";
}
```

Itératives

Comme en langage C

- `while` et `do while`
- `for`
- On peut sortir inconditionnellement avec break,e t passer à l'itération suivante avec continue

Comme en Perl

- `foreach` permet de parcourir un à un les éléments d'un tableau.
- Deux syntaxes possibles :
- `foreach($array as $element)` *instructions*
- `foreach($array as $key=>$element)` *instructions*

Syntaxe alternative pour les blocs

PHP propose une autre manière de rassembler des instructions à l'intérieur d'un bloc pour les structures : `if`, `while`, `for`, `foreach` et `switch`

avec `if`

```php
<?php if ($a == 5): ?>
<p>a égale 5</p>
<?php endif; ?>
```

avec `foreach`

```php
<ul>
  <?php foreach($this->list as $contact) : ?>
    <li><?php echo $contact->nom;?></li>
  <?php endforeach;?>
</ul>
```

Plan : 2 - Le langage

Fonctions

Le pgcd

```php
<?php
function pgcd($a,$b=0){
  while($b!=0){
    $r=$a%$b;
    $a=$b;
    $b=$r;
  }
  return $a;
}
echo "le pgcd de 1520 et 448 est  ".pgcd(1520,448)."<br/>";
?>
```

Le passage par défaut se fait par valeur mais on peut déclarer un paramètre comme devant être passé par référence

Passage par référence

```php
<?php
function swap(&$a,&$b)
{
  $aux=$a;
  $a=$b;
  $b=$aux;
}
$i=2;
$j=3;
swap($i,$j);
echo "i=$i et j=$j";
?>
```

Variable globale

On peut modifier la portée des variables locales à une fonction.

global permet de travailler sur une variable de portée globale au programme. Le tableau associatif `$GLOBALS` permet d'accéder aux variables globales du script :

Portée globale

```php
<?
function change() {
  global $var;            // definit $var comme globale
  $GLOBALS["toto"] ++;    // incremente la variable globale $toto
  $var++;                  // cela sera repercute dans
}                          // le reste du programme
?>
```

Variable statique

static permet de conserver la valeur d'une variable locale à une fonction.

Variable statique

```
<?
function change() {
  static $var=0;  // definit $var comme statique
  $var++;          // sa valeur sera conservee
          // jusqu'au prochain appel
?>
```

Réutilisation de code

Il est pratique et propre d'écrire des fonctions dans un fichier à part (bibliothèque) de manière à les réutiliser suivant les besoins [fonctions de connexion à une base, authentification, etc.]

- `include` et `include_once` : inclut et évalue le code dans un script.

```php
<?php
include_once("mesfonctions.php");
?>
```

- `require` et `require_once` : idem
- Lequel utiliser : `include` ou `require` ? : il n'y a plus de différence depuis PHP 4.0.2. en cas d'erreur, include génére seulement un warning, tandis que require génére une erreur fatale.

On peut inclure aussi du code html par ce biais.

- Les fichiers sont inclus suivant le chemin du fichier fourni ; si aucun n'est fourni, l'`include_path` sera vérifié avant.
- Il est fréquent d'inclure un fichier par rapport au fichier en cours :

Inclusion à partir du répertorie courant

```php
<?php
include_once dirname(__FILE__).'/menu.php';
?>
```