Langage de Script PHP

Partie 1 : Les bases du PHP

Enseignants de l'EMSI Rabat

3éme Année - Ingénierie en Informatique et Réseaux

27 octobre 2023

Table of Contents

Table of Contents

https ://server/path/file

Quand un client saisit et valide un URL dans le navigateur :

1. L'ordinateur récupère l'adresse IP address du serveur à l'aide d'un DNS (Domain Name Server).
2. Le navigateur se connecte à cette adresse IP et demande le fichier.
3. Le serveur (par exemple, Apache) récupère le fichier sur son disque et envoie son contenu au navigateur du client.

Certains URLs spécifient des programmes que le serveur exécute, récupère le résultat de cette exécution et l'envoie au navigateur :

L'URL ci-dessous permet de contacter le serveur ***www.emsi.ma*** et de lui demander d'envoyer le résultat du programme ***formations/certifications/index.php*** après l'avoir exécuté.

http://www.emsi.ma/formations/certifications/index.php

La programmation côté serveur s'effectue avec des langages de programmation (ou des framework) comme :

- **PHP, Java/JSP, Ruby on Rails, ASP.NET, Python, Perl, NodeJS**

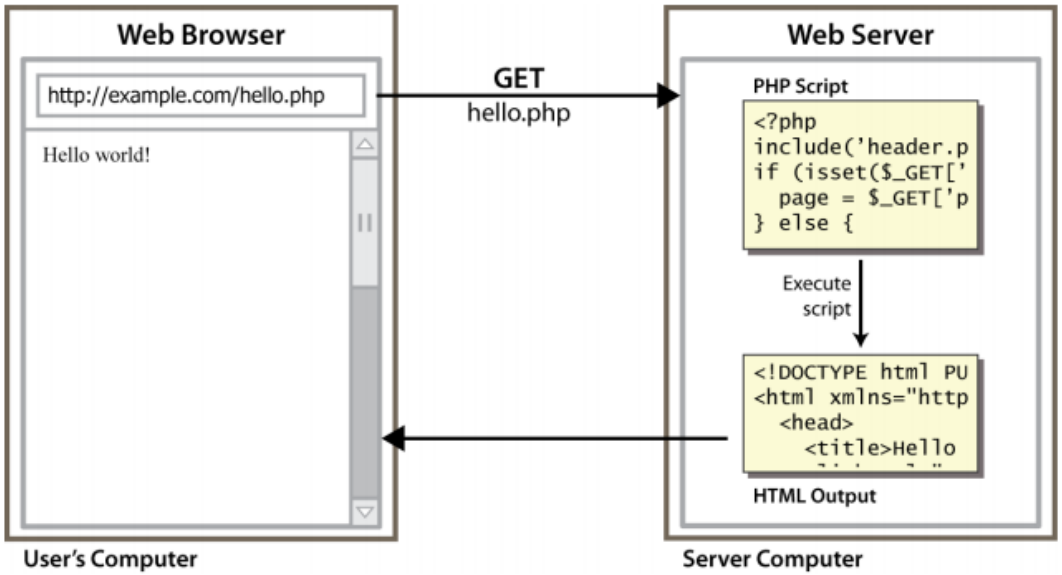
Les serveurs Web sont capables d'exécuter des programmes écrits dans ces langages et peuvent envoyer les résultats de ces programmes au navigateur du client.

Chaque langage/framework a ses avantages et ses inconvénients

La popularité des langages/frameworks varie avec le temps :

- Programming Language Popularity
- Google Trends pour les frameworks populaires et aussi pour PHP.

On utilise PHP dans ce cours à cause de ses avantages : simplicité, popularité, interprété, bien documenté, ...



La requête du navigateur est un fichier **.html** (contenu statique) : le serveur envoie simplement le contenu du fichier.

La requête du navigateur est un script **.php** (contenu dynamique) : le serveur exécute le script et envoie le résultat.

PHP (modèle)

```
print "text";
echo "text";
```

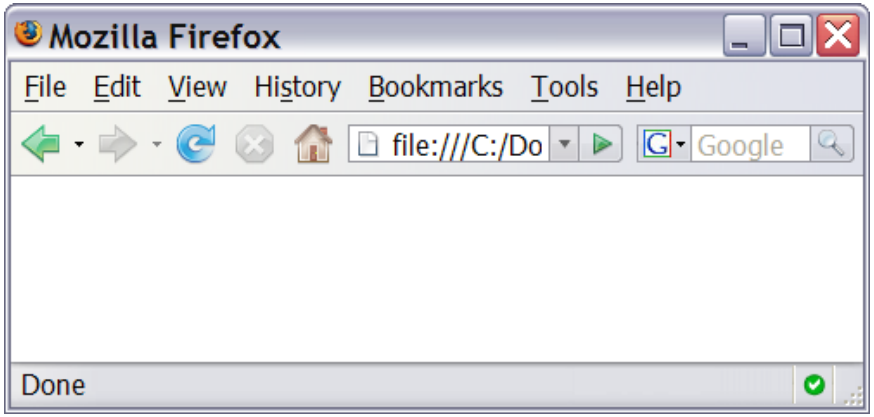
PHP (exemple)

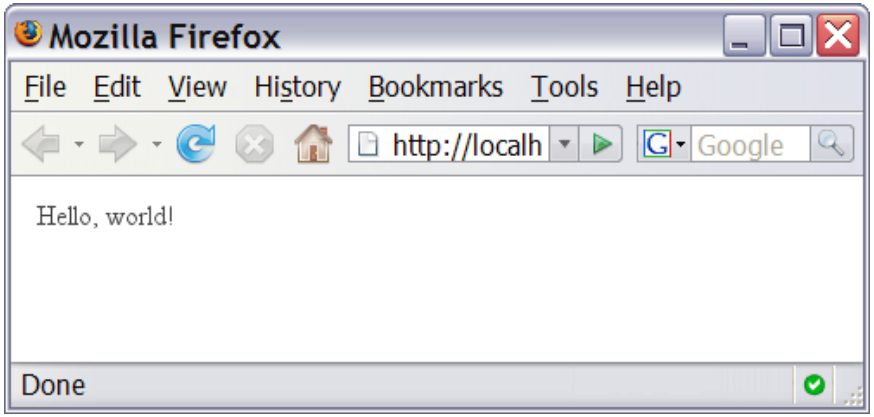
```
echo "Hello, World!";
echo "Escape \“chars\” are the SAME as in Java!";
echo "you can have
line breaks in a string.";

echo ’A string can use "single-quotes". It’s cool!’;
```

Résultat

```
Hello, world!
Escape "chars" are the SAME as in Java!
You can have line breaks in a string.
A string can use "single-quotes". It's cool!
```

Les fonctions **print** et **echo** sont presque identiques.





- On ne peut pas visualizer un script **PHP** avec le protocole **file** : rien ne s'affiche ou bien on voit le source du fichier.
- Il faut placer le script **PHP** sur le serveur local (localhost) : on peut alors activer le script via la barre d'adresse du navigateur.

PHP (code)

```php
<?php
//PHP code goes here
echo "This is a test !";
?>
```

Résultat

This is a test !

Le code PHP est toujours contenu dans les balises `<?php` et `?>`

PHP et HTML

PHP (code)

HTML content
<?php
PHP code
?>
HTML content
<?php
PHP code
?>
HTML content ...

- Le code (PHP) contenu dans `<?php` et `?>` est exécuté et le résultat transmis au client.
- Le code (HTML) non contenu dans `<?php` et `?>` est transmis tel quel au client.

Table of Contents

PHP (exemple)

```php
$user_name = "Peacetachful";
$age = 33;
$age_in_dog_years = $age / 7;
$this_class_rocks = TRUE;
```

- Les noms de variable sont case-sensitive. On peut utiliser le `_` si le nom est long et composé comme `$user_name)`.
- Les noms de variables commencent toujours avec le caractère `$`.
- Les variables sont déclarées implicitement à la première affectation. Une variable peut contenir des valeurs successives de types différents.

Les types de base : **integer**, **float**, **boolean**, **string**, **array**, **object**, **NULL**,

La fonction `is_type` teste le type d'une valeur (i.e **is_string**).

La fonction **gettype** retourne le type d'une valeur sous la forme d'une chaîne de caractères (pas très utilisée).

PHP **convertit automatiquement** très souvent :

- `string` vers `int` conversion automatique avec `+` pour : `("1" + 1 == 2)`.
- `int` vers `float` conversion automatique avec `/` pour : `(3 / 2 == 1.5)`.

Conversion explicite avec (type) : `$age = (int) "21";`

- Opérateurs Arithmétiques : `+, -, *, /, %`
- Opérateurs d'Incrémentation & Décrémentation : `++, − −`
- Opérateurs d'Affectation : `=, +=, -=, *=, /=, %=, .=`
- Opérateurs de concaténation : `.`
- Opérateurs de comparaison :
  `==, <, >, <=, >=, !=, <>, ===, !==`

Beaucoup d'opérateurs convertissent automatiquement leurs opérandes : `5 + "7"` donne `12`.

PHP

```
$favorite_food = "Asian";
echo $favorite_food[2];                    # i
```

C'est identique à un tableau de caractères, utilisable avec les `[]`

L'opérateur de concatenation est `.` (point) et non `+`

- `5 + "2 turtle doves"` retourne 7.
- `5 . "2 turtle doves"` retourne "52 turtle doves".

Peut être délimitée par " " ou ''.

PHP

```php
$name = "Emsi - 3iir"
$length = strlen($name);                    # 11
$cmp = strcmp($name, "Emsi");               # > 0
$index = strpos($name, "i");                # 3
$last = substr($name, 7, 4);                # "3iir"
$name = strtoupper($name);                  # "EMSI - 3IIR"
```

Nom	Fonctions équivalentes en JavaScript
strlen	length
strpos	indexOf
substr	substring
strtolower, strtoupper	toLowerCase, toUpperCase
trim	trim
explode, implode	split, join

PHP (exemple)

```php
$age = 16;
echo "You are " . $age . " years old."
print "You are $age years old.";              # You are 16 years old.
```

- Les variables dans les chaînes délimitées par `" "` sont interprétées.
- Les variables dans ces chaînes sont remplacées par leur valeur.

- Les variables dans les chaînes délimitées par `' '` ne sont pas interprétées :

PHP (exemple)

```php
echo 'You are $age years old.';          # You are $age years old.
```

- Pour lever toute ambiguité, on peut placer le nom de la variable entre :

PHP (exemple)

```php
echo "Today is your $ageth birthday."; # ageth not found
echo "This's your {$age}th birthday."; # This's your 16th birthday.
```

Expressions régulières
String : filtrage

Les expressions régulières constituent un système très puissant et très rapide pour faire des recherches dans des chaînes de caractères. C'est un outil efficace pour le filtrage et la vérification du contenue des Strings
Les principales fonctions qui utilisent les "regex" (Regular Expressions) sont :

- **preg_match($patern, $subject)** : Effectue une recherche de correspondance avec une expression régulière.
- **preg_replace($patern, $replace, $subject)** : Remplace le texte qui correspond à une expression régulière.
- **preg_split($patern, $replace, $subject)** : Éclate une chaîne par expression régulière.

Une regex est toujours entourée de caractères spéciaux appelés délimiteurs (/)
Il existe d'autres fonctions PHP qui utilisent les regex.
Il existe aussi des fonctions PHP qui permettent le filtrage sans regex comme filte

- **preg_match($patern, $subject)**

On peut utiliser cette fonction pour vérifier si l'information donnée par l'utilisateur correspond bien à ce qui est demandé. Par exemple, pour vérifier une variable qui doit contenir seulement les valeurs 'O' ou 'N', on peut écrire :

PHP (exemple)

```
if ( !preg_match("/[ON]$/",$rep)) {
  echo "Veuillez saisir une réponse valide: O ou bien N ! <br>";
}
```

PHP

```php
$feels_like_summer = FALSE;
$php_is_rad = TRUE;
$student_count = 217;
$nonzero = (bool) $student_count;                    # TRUE
```

Les valeurs suivantes sont vues comme **FALSE** (toutes les autres valeurs sont vues comme **TRUE**) :

- 0, 0.0 et Tableaux avec 0 element.
- "", "0", et NULL (y compris les variables non initialisées).

(bool) permet de convertir explicitement une valeur vers un booléen.

PHP (Modèle)

```
if (condition) {
   statements ;
} else if (condition) {
   statements ;
} else {
   statements ;
}
```

On peut utiliser **elseif** (sans espace) au lieu de **else if** .

PHP (Modèle)

```
for (initialization ; condition ; update) {
   statements ;
}
```

PHP (exemple)

```php
for ($i = 0; $i < 10; $i++) {
  echo "$i squared is " . $i * $i . " <br>";
}
```

PHP (Modèle 1)

```
while (condition) {
    statements;
}
```

PHP (modèle 2)

```
do {
    statements;
} while (condition);
```

Les instructions **break** et **continue** existent et fonctionnent comme en Java.

PHP (Modèle)

```
$name = array();                          # Create
$name = [];                          # Create
$name = array(value0, ..., valueN);
$name = [value0,...];                          # Create
$name[index]                          # Get element value
$name[index] = value;                 # Set element value
$name[] = value;                      # Add to end
```

PHP (Exemple)

```
$a = array();                          # Empty array (length 0)
$a[0] = 23;                            # Stores 23 at index 0 (length 1)
$a2 = array("some", "strings", "in", "an", "array");
$a2[] = "Ooh!";                       # Add string to end (at index 5)
```

NB : Un tableau peut contenir des éléments de différents types.

Nom	Description
count	Nombre d'éléments dans un tableau
print_r	Affiche le contenu d'un tableau
array_pop, array_push, array_shift, array_unshift	Pour utiliser un tableau comme une pile/file
in_array, array_search, array_reverse, sort, rsort, shuffle	Recherche et tri
array_fill, array_merge, array_intersect, array_diff, array_slice, range	Création, initialisation et manpulation
array_sum, array_product, array_unique, array_filter, array_reduce	Traitement des éléments

PHP (Exemple)

```php
$tas = array("CA", "MJ", "MG", "SK", "KC");
for ($i = 0; $i < count($tas); $i++) {
$tas[$i] = strtolower($tas[$i]);
}                                  # ("ca", "mj", "mg", "sk", "kc")

$tas[] = "zn";                   # ("ca", "mj", "mg", "sk", "kc", "zn")
array_unshift($tas,"os");         # ("os", "ca", "mj", "mg", "sk", "kc", "zn")
$c = array_shift($tas);           # ("ca", "mj", "mg", "sk", "kc", "zn")
array_pop($tas);                     # ("ca", "mj", "mg", "sk", "kc")
array_push($tas, "kt");             # ("ca", "mj", "mg", "sk", "kc", "kt")
array_reverse($tas);                 # ("tk", "kc", "sk", "mg", "mj", "ca")
sort($tas);                          # ("ca", "kc", "mg", "mj", "sk", "tk")
$ks = array_slice($tas, 1, 2);    # ("kc", "mg")
```

Les tableaux en PHP remplacent la plupart des structures de données de Java : **list**, **stack**, **queue**, **set**, **map**, ...

PHP (Modèle)

```
foreach ($array as $variableName) {
   statements ;
}
```

PHP (exemple)

```php
$stooges = array ("Larray", "Moe", "Curly", "Shemp");
foreach ($stooges as $stooge) {
  echo "Moe slaps $stooge\n";                    # Even himself
}
```

La boucle **foreach** permet d'itérer sur les éléments d'un tableau sans utiliser les indices.

PHP (Modèle)

```
function name(parameter1, ..., parameterN) {
   statements;
}
```

PHP (exemple)

```
function bmi($weight, $height) {
  $result = 703 * $weight / $height / $height;
  return $result;
}
```

- Les types des paramètres et de la valeur de retour ne sont pas spécifiés.

PHP (Modèle)

name(expression1,..., expressionN);

PHP (exemple)

```
$w = 163;                          # pounds
$h = 70;                           # inches
$my_bmi = bmi($w, $h);
```

Erreur

Appeler une fonction avec le mauvais nombre d'argument provoque une erreur.

PHP (Modèle)

```
name (parameter1=value, ..., parameterN=value) {
   statements ;
}
```

PHP (exemple)

```php
function print_separated($str, $separator=", ") {
  for ($i = 0; $i < strlen($str); $i++) {
    echo $str[$i] . $separator;
  }
}
print_separated("hello");                    # h, e, l, l, o,
print_separated("hello", "-");               # h-e-l-l-o-
```

PHP (Exemple)

```php
$school = "EMSI";                                    # global
...

function downgrade() {
  global $school;
  $suffix = "(3IIR)";                                  # local

  $school = "$school $suffix";
  echo "$school\n";
}
```

- Les variables déclarées dans une fonction sont **locales** à cette fonction.
- Les variables déclarées à l'extérieur d'une fonction sont **globales** pour cette fonction.

PHP (Modèle)

```
# commentaire sur une ligne

// commentaire sur une ligne

/*
block de commentaires
block de commentaires
...
*/
```

Comme en Java, avec `#` en plus :

- Beaucoup de développeurs PHP utilisent `#` au lieu de `//`

Table of Contents

PHP (exemple)

```
URL?name1=value1&name2=value2...


http://www.google.com/search?q=Aya
http://example.com/student_login.php?username=aya&id=1234
```

- Il est possible de passer des paramètres lors d'une requête à un serveur :
  - Des couples (paramètre,valeur) placés à la fin de l'URL, par exemple, (`username` , `aya`), et (`id` , `1234`).
- Un script PHP peut lire et utiliser ces paramètres.
- Les paramètres sont toujours des chaînes de caractères.

- Deux types de requête vers le serveur :

1. Une requête GET est utilisée pour obtenir des données (HTML) du serveur sans changer son état. La requête peut avoir des paramètres.

2. Une requête POST est utilisée pour transmettre des données au serveur et éventuellement changer son état. La requête peut avoir des paramètres.

PHP (exemple)

```php
$user_name = $_GET["username"];
$id_number = (int) $_GET["id"];
$eats_meat = FALSE;
if (isset($_GET["meat"])) {
    $eats_meat = TRUE;
}
```

- `$_GET["parameter"]` ou `$_POST["parameter"]` contient la valeur d'un paramètre **GET** ou **POST** (une chaîne de caractères).
- les paramètres passés directement dans l'URL sont des paramètres **GET** : `http ://domain/script.php ?name1=value1&name2=value2`

PHP (script)

```php
$base = $_GET["base"];
$exp = $_GET["exponent"];
$result = pow($base, $exp);
echo "$base ^$exp = $result";
```

PHP (URL)

puissance.php ?base=3&exponent=4

PHP (Résultat)

```
3 ^4 = 81
```

PHP (Modèle)

```
<form action="destination URL" method="get OR post">
  form controls
</form>
```

- L'élément de base en HTML pour transmettre des données au serveur.
- L'attribut **action** contient l'URL du scrip (PHP) qui va traîter les données du formulaire.
- Quand tous les éléments de formulaires ont été renseignés, le formulaire est soumis et les données contenues dans ces éléments sont transmises au script.
- Si l'attribut **action** n'est pas présent, le script qui va traîter les données est le ***même*** qui contient ce formulaire !

PHP (Modèle)

```
<input type="text" name="q" value="type in here" >
<input type="submit" value="Send" >
```

- L'élément **input** permet de créer beaucoup d'éléments de formulaire différents.
- L'attribut **name** est le nom du paramètre envoyé au serveur.
- L'attribut **type** détermine le type d'élément : **button**, **file**, **checkbox**, **hidden**, **password**, **radio**, **reset**, **submit**, **text**,...
- L'attribut **value** contient le texte initial de l'élément.

PHP (Modèle)

```
<textarea rows="4" cols="20">
  Type your comments here.
</textarea>
```

- L'élément input permet de créer beaucoup d'éléments de formulaire différents.
- Une zone de saisie de texte sur plusieurs lignes.
- Les attributs **rows** et **cols** spécifient la hauteur et la largeur en nombre de caractères.
- L'attribut **readonly** rend le contenu de l'élément non modifiable.

PHP (Exemple)

```
<input type="checkbox" name="lettuce"> Lettuce
<input type="checkbox" name="tomato" checked="checked"> Tomato
<input type="checkbox" name="pickles" checked="checked"> Pickles
```

- 0, 1, ou plusieurs checkboxes peuvent être sélectionnées en même temps.

- Lors de l'envoi au serveur :
  - La valeur de l'attribut **name** de chaque checkbox sélectionnée est le nom d'un paramètre transmis au serveur, et la valeur de ce paramètre est **on**.

http ://domain/script.php ?tomato=on&pickles=on

PHP (Exemple)

```
<input type="checkbox" name="lettuce" checked="checked" >
```

- L'attribut **checked** permet de sélectionner des checkboxes par défaut :

PHP (Exemple)

```
<input type="radio" name="cc" value="visa" checked="checked"> Visa
<input type="radio" name="cc" value="mastercard"> MasterCard
<input type="radio" name="cc" value="amex"> American Express
```

- Choix mutuellement exclusifs.
- Un groupe est déterminé par la même valeur de l'attribut **name**.

- Lors de l'envoi au serveur :
  - La valeur de l'attribut **name** d'un groupe est le nom du paramètre transmis au serveur.
  - La valeur de l'attribut **value** du bouton sélectionné dans ce groupe est la valeur de ce paramètre.

http ://domain/script.php ?cc=visa

PHP (Modèle)

```
<input type="text" name="username"> Name <br>
<input type="text" name="sid"> SID <br>
<input type="hidden" name="school" value="EMSI">
<input type="hidden" name="year" value="2019">
```

- Paramètre invisible à l'utilisateur dont la valeur (de l'attribut **value**) est transmise au serveur lorsque le formulaire est soumis.
- Très utile pour échanger des données entre plusieurs appels à des scripts.
- Du point de vue de PHP, identique à un **input** de type **text**.

PHP (Modèle)

```
<select name="favoritecharacter">
  <option value="brad">Brad Pitt</option>
  <option value="george">George Clooney</option>
  <option value="johnny" selected>Johnny Depp </option>
</select>
```

- Le nom du paramètre transmis au serveur est la valeur de l'attribut **name** de l'élément **select**.
- La valeur de l'attribut **value** de l'élément **option** sélectionné est la valeur du paramètre transmis au serveur.
- **select** possède les attributs : **disabled**, **multiple**, **size**.
- L'attribut **selected** permet de sélectionner un élément **option** par défaut.