



Conception & Développement Informatique

PHP - Introduction

PHP, Command Line Interface, Variables d'URL, Pseudo-Frames

Mickaël DEVOLDÈRE

MD v1.0.1

20/03/2018



<http://www.arfp.asso.fr>

PHP - Introduction

PHP, Command Line Interface, Variables d'URL, Pseudo-Frames

Technologies et langages



HTML



PHP

Légende des icônes



Information complémentaire



Point d'attention particulier



Intervention du formateur possible



Lien vers une ressource externe

Sommaire

PHP - Historique	3
Caractéristiques du langage PHP	4
License.....	4
Portabilité.....	4
Exécution.....	4
Apprentissage de PHP	5
Richesse du langage PHP.....	5
PHP, un langage fiable et performant.....	5
PHP, un langage conçu pour les applications web dynamiques	6
PHP pour d'autres domaines d'applications.....	7
Les limites de PHP	8
PHP : documentation officielle	8
PHP, Un langage faiblement typé	9
PHP CLI : Interface en ligne de commande	9
Windows : Vérifier le PATH	9
Mon 1 ^{er} script PHP en ligne de commande.....	11
PHP : Dynamiser un site web	13
Les Paramètres d'url	13
Récupérer les paramètres d'url avec PHP : \$_GET	13
Historique du document	16
Crédits	16

PHP - Historique

Le langage PHP a été inventé par Rasmus LERDORF en 1995 pour son usage personnel (mise en ligne de son CV en l'occurrence). Autrefois abbréviation de Personal HomePage devenue aujourd'hui Hypertext Preprocessor, PHP s'impose comme un standard dans le monde de la programmation web par ses performances, sa fiabilité, sa souplesse et sa rapidité.

PHP a été inventé à l'origine pour le développement d'applications web dynamiques qui constituent encore le cas d'utilisation le plus courant et son point fort. Cependant, les évolutions qui lui ont été apportées jusqu'à aujourd'hui assurent à PHP une polyvalence non négligeable. PHP est par exemple capable d'interagir avec Java, de générer des fichiers PDF, d'exécuter des commandes Shell, de gérer des objets (au sens programmation orientée objet), de créer des images ou bien de fournir des interfaces graphiques au moyen de PHP GTK.

Dans cette présentation du langage, nous introduirons tout d'abord les caractéristiques de PHP, sa syntaxe de base et ses différentes possibilités et limites.

Caractéristiques du langage PHP

License

PHP est tout d'abord un langage de script interprété (en réalité précompilé en Opcode), gratuit, OpenSource et distribué sous une license autorisant la modification et la redistribution.

Portabilité

PHP est supporté sur plusieurs systèmes d'exploitation. C'est par exemple le cas des versions Microsoft Windows™, mais aussi des systèmes reposant sur une base UNIX (Apple MAC OS X™, distributions Linux ou encore Sun Solaris).

Il sera alors très facile de déplacer une application écrite en PHP d'un serveur Windows d'origine vers un serveur Linux sans avoir à la modifier (ou très peu).

Exécution

D'un point de vue exécution, PHP a besoin d'un serveur Web pour fonctionner. Toutes les pages demandées par un client seront construites par le serveur Web, en fonction des paramètres transmis, avant d'être retournées au client. Le schéma ci-dessous illustre le principe de fonctionnement de PHP.



Apprentissage de PHP

PHP est un langage dit de « haut niveau » dont la syntaxe est très proche du langage C. Cette syntaxe proche du langage naturel lui assure un apprentissage rapide et peu de contraintes d'écriture de code. Néanmoins, la maîtrise rapide de sa syntaxe ne signifie pas la maîtrise de ses fonctionnalités et de ses concepts. Une bonne connaissance et une utilisation avancée de la programmation PHP nécessite un temps d'apprentissage relativement long.

Richesse du langage PHP

Une des forces du langage PHP est sa richesse en terme de fonctionnalités. En effet, il dispose à l'origine de plus de 4000 fonctions natives prêtes à l'emploi garantissant aux développeurs de s'affranchir de temps de développement supplémentaires et parfois fastidieux. Ces fonctions permettent entre autre de traiter les chaînes de caractères, d'opérer mathématiquement sur des nombres, de convertir des dates, de se connecter à un système de base de données, de manipuler des fichiers présents sur le serveur...

PHP puise aussi sa richesse dans le dynamisme de sa communauté de développeurs. Celle-ci était estimée à 500 000 personnes environ en 2003 mais il est très probable qu'elle ait dépassé le million maintenant. Les profils de développeurs de la communauté sont très divers. Il y'a ceux qui apportent de nouvelles fonctionnalités et librairies de version en version, ceux qui traduisent la documentation en plusieurs langues ou encore les programmeurs ayant des compétences plus modestes qui réalisent des applications Opensources prêtes à l'emploi. Parmi les plus connues, nous pouvons citer les CMS (Joomla, SPIP, Dotclear, Wordpress...), les systèmes de ventes en ligne (OSCommerce), les forums (PHPBB, IPB, VBulletin), les frameworks (Zend Framework, Symfony, CakePHP, Jelix)...

PHP, un langage fiable et performant

Le langage est maintenant devenu un langage fiable, performant et viable. Il est capable de supporter des sites qui sollicitent des millions de requêtes journalières. De nombreuses entreprises de renommée nationale et internationale lui font confiance pour le développement de leur site Internet. Nous pouvons parmi elles citer TF1, IBM, Le Monde, Le Figaro, Club-Internet, Orange, Pages Jaunes... Un récent rapport daté de novembre 2006 indique que 87% des entreprises du CAC40 utilisent PHP. Lire l'interview de Perrick Penet (AFUP).

PHP, un langage conçu pour les applications web dynamiques

Le langage PHP a pour principale vocation la réalisation d'applications web dynamiques. Par définition, une « application (ou page) dynamique » est un programme capable de générer une page unique en fonction de paramètres qui lui sont transmis.

Un script PHP peut donc être intégré directement à l'intérieur d'un code html:

```
<html>
  <head>
    <title>Hello World en PHP</title>
  </head>
  <body>
    <p>
      <?php echo 'Hello World !'; ?>
    </p>
  </body>
</html>
```

Le script PHP, clairement identifié par les deux balises **<?php** (ouverture) et **?>** (fermeture), provoquera l'écriture de la chaîne de caractères Hello World entre les balises HTML après son exécution sur le serveur.

Nous aurions également pu obtenir le même résultat en utilisant le script php suivant :

```
<html>
  <head>
    <title>Hello World en PHP</title>
  </head>
  <body>
    <?php echo '<p>Hello World !</p>'; ?>
  </body>
</html>
```

On peut déduire du code précédent que PHP est capable de générer du code HTML (mais pas que !), ce qui fait tout son intérêt. La présentation du document HTML généré est alors complètement dépendante des conditions passées et des paramètres initiaux fournis.

PHP pour d'autres domaines d'applications

Fort de sa richesse, PHP ne se limite pas forcément à l'édition de pages web dynamiques. Il peut par exemple être utilisé en ligne de commande via l'utilisation de l'exécutable `php cli`. Ce cas d'utilisation permet alors d'exécuter des scripts directement sur les machines. Il est possible de coupler l'utilisation de PHP avec un gestionnaire de tâches tel qu'un serveur cron sous Linux.

PHP c'est également la possibilité de créer des applications lourdes fonctionnant sans serveur ni navigateur. Autrement dit des applications traditionnelles, autonomes et munies de fenêtres. Tout cela se réalise au moyen de la librairie PHP GTK disponible à l'adresse : <http://gtk.php.net>

Un autre point fort de PHP est sa capacité à s'interfacer très facilement avec de nombreux systèmes de gestion de bases de données relationnelles et noSQL. Parmi eux, nous pouvons retrouver MySQL, Oracle, SQLite, MS SQLServer, PostgreSQL, MongoDB...

Les possibilités offertes par PHP sont donc très nombreuses et nous n'allons pas les détailler toutes car cela vous donnerait des jours entiers de lecture.

Retenons néanmoins une liste de ses principales capacités :

- ✓ Manipulation d'un système de fichiers (création, édition, suppression, droits d'accès...)
- ✓ Gestion des sessions utilisateurs
- ✓ Génération et parsing de documents XML
- ✓ Génération d'images
- ✓ Génération de fichiers PDF
- ✓ Accès simplifié aux bases de données
- ✓ Exécution de commandes Shell
- ✓ Gestion des e-mails
- ✓ Compression et décompression d'archives ZIP
- ✓ Chiffrement de données
- ✓ Gestion d'annuaires LDAP
- ✓ Manipulation des dates
- ✓ Manipulation des URL
- ✓ Envoi et lecture de cookies
- ✓ Dialogue avec Java
- ✓ Utilisation d'Ajx
- ✓ ...

La version 5 de PHP a apporté un grand vent de fraîcheur au langage et aux professionnels. Le principal manque jusque là était son modèle objet trop succinct. Les développeurs se sont alors penchés sur cette problématique et ont finalement implémenté un modèle de programmation objet proche du langage Java. Grâce à un tel modèle, les professionnels encore hésitants à utiliser PHP deviennent de plus en plus nombreux. Des applications complètement objet voient également le jour et intègrent un design pattern MVC. Les plus connues aujourd'hui sont les frameworks Zend et Symfony des sociétés respectives Zend Technologies et Sensio Labs.

Les limites de PHP

Malgré toutes les qualités que nous pouvons attribuer au langage PHP, subsistent quelques défauts. Par exemple, dans le cas de très grosses applications, il peut présenter quelques faiblesses et devenir inadapté. Un langage tel que PERL deviendrait alors plus adéquat. Cet argument reste toutefois très subjectif dans la mesure où les développeurs de PHP améliorent la qualité et la robustesse du langage.

Le second défaut (mais qui paradoxalement fait son succès et sa qualité) que nous pouvons lui reprocher est sa grande simplicité d'utilisation. Cela a beaucoup terni l'image de PHP parceque tout webmaster (même très peu expérimenté) devient capable de créer du code et des applications facilement. Cependant, la plupart des codes produits par des développeurs amateurs n'est pas forcément bien conçu, souvent peu sécurisé, peu maintenable et même non optimisé. Les autres langages comme C++, .Net, Java, ASP, Perl, Python ou Ruby ne subissent pas cette mauvaise image dans la mesure où leur apprentissage est plus fastidieux.

Enfin, le dernier défaut reprochable à PHP est son manque de rigueur dans la nomenclature des fonctions et de la syntaxe. Par exemple, un manque de standardisation des noms des fonctions :

Utilisation du séparateur underscore :

- `str_replace()`, `preg_match()`, `mysql_real_escape_string()` ...

Fonctions composées de plusieurs mots écrite en un seul :

- `wordwrap()`, `htmlspecialchars()` ...

Traduction du to en littéral ou numéraire :

- `bin2hex()`, `strtotime()`...

PHP : documentation officielle

PHP héberge sa documentation sur <http://www.php.net> . Vous y trouverez toutes les informations nécessaires pour bien utiliser PHP.

PHP, Un langage faiblement typé

Contrairement à d'autres langages tels que C# ou Java, PHP est ce qu'on appelle un langage faiblement typé. Il permet au programmeur de ne pas se soucier du type de variables qu'il manipule. Si cela offre certaines facilités, l'approche comporte toutefois certains pièges dont il vaut mieux être conscient.

PHP CLI : Interface en ligne de commande

Le suite de ce document fait référence à une installation de WampServer sur une machine Windows.



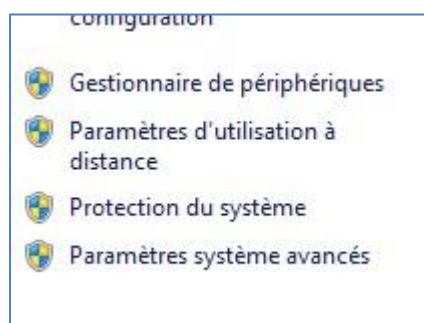
WampServer devrait être présent sur votre machine. Si tel n'est pas le cas, votre formateur peut vous accompagner dans la démarche d'installation.

Windows : Vérifier le PATH

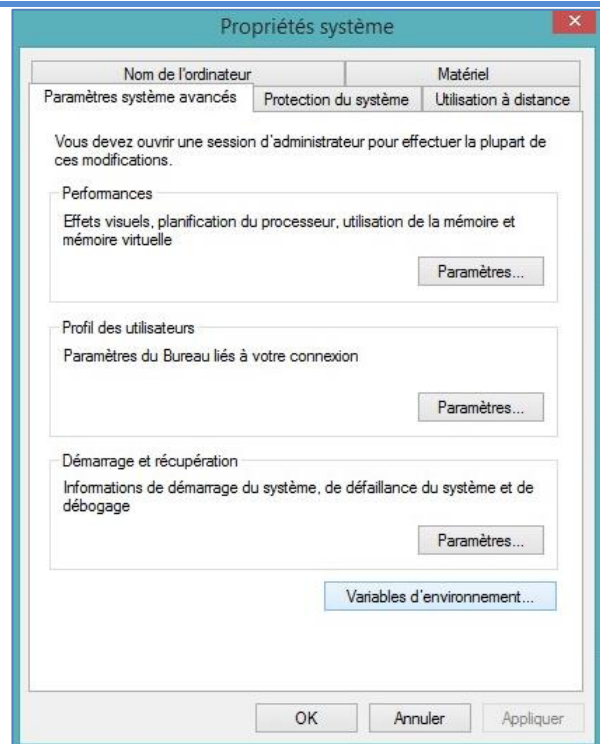
Lorsque vous installez PHP que ce soit avec WAMP, XAMPP, ou tout simplement en installant le binaire de php pour windows, il ne vous est pas possible d'utiliser php en ligne de commande .

La méthode qui suit permet d'ajouter un programme à la ligne de commande de Windows. Pour cela, il est nécessaire de renseigner le chemin vers l'exécutable PHP dans les variables d'environnements de Windows.

Allez dans **Panneau de configuration** puis **Système et sécurité** puis **Système** (Panneau de configuration\Système et sécurité\Système). Sur la gauche de votre fenêtre devrait se trouver un bouton **Paramètres système avancé**.



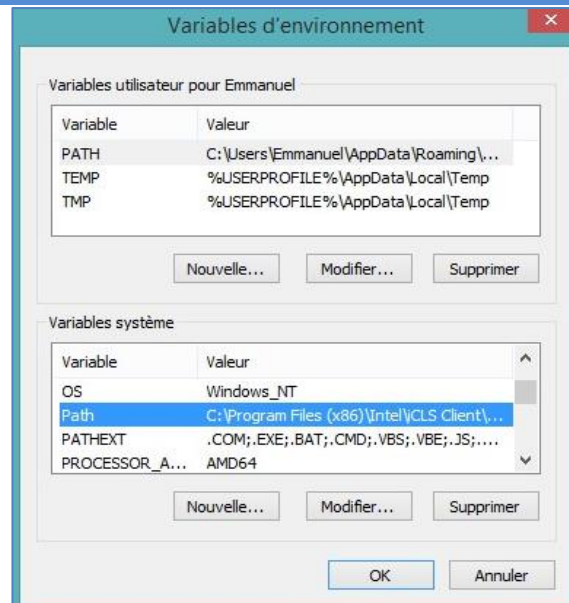
Dans la fenêtre suivante, Cliquez sur **variables d'environnement**



Il vous faut maintenant rajouter le chemin d'accès vers votre exécutable PHP à la variable d'environnement PATH.

Dans **variables système**, sélectionnez l'entrée « Path » puis cliquez sur « modifier »

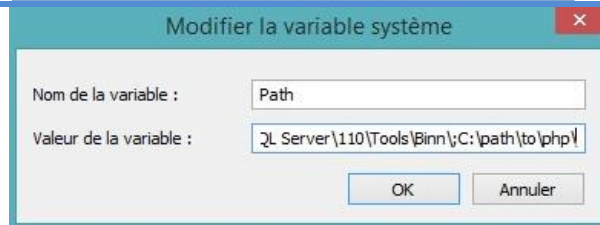
J'attire votre attention sur le fait que les chemins contenus dans cette variable sont séparés par un point-virgule.



Ajoutez un point-virgule à la fin du champ « valeur de la variable » puis le chemin vers le dossier contenant l'exécutable PHP.

Pour wamp64 version 3, par défaut :

`C:\wamp64\bin\php\php7.1.9`



Une fois l'ajout effectué, il ne vous reste plus qu'à tester en tapant la commande `php -v`.

```
C:\Users\CRM>php -v
PHP 7.1.9 (cli) (built: Aug 30 2017 18:34:46) ( ZTS MSVC14 (Visual C++ 2015) x64 )
Copyright (c) 1997-2017 The PHP Group
Zend Engine v3.1.0, Copyright (c) 1998-2017 Zend Technologies
```

Si tout fonctionne, le numéro de version de votre installation de php devrait s'afficher.

Si tel n'est pas le cas, il faudra redémarrer l'ordinateur pour que les modifications soient prises en compte.

Mon 1^{er} script PHP en ligne de commande

Créez un nouveau projet PHP dans le répertoire **C:\wamp64\www\exercices_php**

Créez ensuite un dossier « cli » à l'intérieur du dossier « exercices_php ».

Puis, créez un fichier « welcome.php » qui sera notre 1^{er} script PHP en ligne de commande.

Insérez le code suivant :

Ligne 1 : Balise d'ouverture de PHP

Ligne 3 : on vérifie que le tableau \$argv contient plus d'un élément

Ligne 5 : Si le tableau \$argv contient plus d'un élément, on stocke le 2^{ème} élément dans la variable \$arg.

Ligne 9 et 10 : Si \$argv contient moins de 2 éléments, on demande à l'utilisateur de saisir son nom. La fonction « fgets(STDIN) » invite l'utilisateur à taper un texte et récupère sa saisie dans la variable \$arg.

Ligne 13 : Affichage du résultat

```

1 <?php
2
3     if(count($argv) > 1)
4     {
5         $arg = $argv[1];
6     }
7     else
8     {
9         echo 'Type your name: ';
10        $arg = fgets(STDIN);
11    }
12
13    echo PHP_EOL . "Welcome ".trim($arg). PHP_EOL;
14
15    ?>

```

Ouvrez maintenant un terminal puis dirigez vous vers le répertoire contenant le script à l'aide de la commande « cd ».

```

C:\Users\CRM>cd c:\wamp64\www\exercices_php\cli
c:\wamp64\www\exercices_php\cli>

```

Puis pour exécuter le script « welcome.php » :

```

c:\wamp64\www\exercices_php\cli>php welcome.php
Type your name:

```

Entrez votre prénom et observez le résultat.

Explications :

Lorsqu'un script est exécuté en ligne de commande, 2 variables sont automatiquement créées et utilisables en lecture : \$argc et \$argv.

Variable	Type	Description
\$argc	INT	Nombre de paramètres de la commande
\$argv	ARRAY	Tableau contenant les paramètres de la commande

Lorsque vous appelez un script PHP depuis la ligne de commande, au moins 1 paramètre est fourni : le nom du fichier qui est exécuté. La variable \$argc aura toujours la valeur 1 au minimum et le tableau \$argv contiendra donc au minimum 1 paramètre (le nom du fichier).

Il est possible de spécifier plus de paramètres à l'appel d'un script en ligne de commande. Il suffit de renseigner chaque paramètre à la suite de la commande (chaque paramètre est séparé par un espace).

A l'exécution, le 2^{ème} paramètre est stocké dans \$argv[1], le 3^{ème} dans \$argv[2] etc...

Le script précédent « welcome.php » accepte un second paramètre qui, s'il est spécifié, permet de directement afficher le prénom de l'utilisateur sans avoir besoin de lui demander.

Exécutez à nouveau le script « welcome.php » en y ajoutant votre prénom en second paramètre.

```
c:\wamp64\www\exercices_php\cli>php welcome.php mickael  
Welcome mickael
```

Vous remarquerez que l'utilisateur n'est pas invité à entrer son prénom qui est directement affiché.

Exercez vous en créant des scripts en ligne de commandes permettant :

- de calculer une addition
- de calculer l'âge d'une personne selon une date de naissance donnée
- d'afficher les fichiers d'un répertoire

PHP : Dynamiser un site web

Le plus gros intérêt de PHP est de pouvoir dynamiser le contenu d'un site web. C'est-à-dire que le contenu va s'adapter selon les paramètres envoyés à PHP.

Les Paramètres d'url

Un URL est un chemin vers une ressources accessible par un réseau.

Exemple d'URL simple :

```
http://mondomaine.fr/contact.html
```

Avez-vous déjà observé des adresses de sites web du type :

```
http://mondomaine.fr/index.php?page=articles&article_id=1
```

Cet URL est ce qu'on appelle un URL paramétré. C'est-à-dire qu'il contient des paramètres que l'on va pouvoir exploiter.

L'URL peut être décomposés en 2 parties :

Avant le « ? »	http://mondomaine.fr/index.php	URL menant au script PHP à exécuter
Après le « ? »	page=articles&article_id=1	Paramètres d'url séparés par un « & »

Chaque paramètre se trouvant après le « ? » est une paire « clé=valeur ».

Les paramètres sont séparés entre eux par le caractère « & » (ou « & » si encodé en HTML).

Dans l'url précédente, nous pouvons donc identifier 2 paramètres

Nom (clé) du paramètre	=	Valeur du paramètre
page	=	articles
article_id	=	1

Récupérer les paramètres d'url avec PHP : \$_GET

PHP permet de récupérer et exploiter les paramètres d'URL.

Lorsqu'un script PHP est appelé depuis un URL paramétré, les différents paramètres sont stockés dans un tableau (array).

Ce tableau est référencé par la variable **\$_GET** qui est une variable « super-globale ».



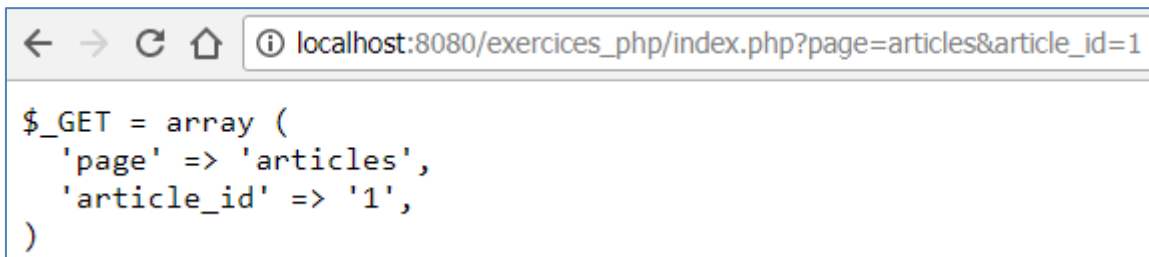
Plus d'informations concernant les variables superglobales sur le site officiel de PHP :
<http://php.net/manual/fr/language.variables.superglobals.php>

Par défaut, la variable `$_GET` est un tableau vide (lorsqu'aucun paramètre d'URL n'est fourni).

Reprenons l'URL précédente qui contenait un paramètre « `page` » et un 2nd paramètre « `article_id` » :

http://mondomaine.fr/index.php?page=articles&article_id=1

Lorsque cet URL est appelé, PHP va alimenter le tableau `$_GET` qui contiendra donc les 2 paramètres tel que présenté ci-dessous :



```
$_GET = array (
    'page' => 'articles',
    'article_id' => '1',
)
```

L'accès aux paramètres contenus dans le tableau `$_GET` se réalise comme pour un tableau associatif classique.

Par exemple, pour récupérer la valeur du paramètre « `page` », le code PHP à écrire est le suivant :

Ligne 2 : On vérifie que la clé « `page` » existe dans le tableau `$_GET` .

Ligne 3 : Si la clé « `page` » existe, on attribue sa valeur à la variable `$page`.

Ligne 8 : Sinon, on attribue une valeur par défaut.

Ligne 13 à 15 : équivalent au if/else précédent utilisant l'opérateur ternaire.

```
<?php
2  if(!empty($_GET['page']))
3  {
4      $page = $_GET['page'];
5  }
6  else
7  {
8      $page = 'valeur par défaut';
9  }
10
11  // Que l'on pourrait également écrire :
12
13  $page = !empty($_GET['page'])
14      ? $_GET['page']
15      : 'Valeur par défaut';
16
```

Les pseudo-Frames : Cas d'utilisation des paramètres d'URL

Les « pseudo frames » sont un cas d'utilisation spécial des paramètres d'URL (\$_GET).

On désigne un paramètre qui fera référence à un fichier (seulement son nom). Si le fichier existe, on affiche son contenu. Sinon on affiche un contenu par défaut.

Le code suivant illustre cette utilisation :

```
1  <?php
2  // Si le paramètre "page" existe et n'est pas vide
3  if(!empty($_GET['page']))
4  {
5      $page = $_GET['page'];
6  }
7  else
8  {
9      $page = 'home';
10 }
11
12 // On nettoie la variable
13 // http://php.net/manual/fr/function.basename.php
14 $page = basename($page);
15
16 // On définit le chemin vers le fichier.
17 // Nos fichiers sont stockés dans le répertoire Views.
18 $chemin_page = __DIR__ . '/Views/' . $page . '.php';
19
20 // Si le fichier n'existe pas
21 // On définit l'entête "erreur 404"
22 // On termine l'exécution du script avec un message d'erreur
23 if(!is_file($chemin_page))
24 {
25     header("HTTP/1.0 404 Not Found");
26     exit('Erreur 404: La page demandée n\'existe pas !');
27 }
28
29 // Sinon, tout est OK, on peut inclure le fichier.
30
31 require $chemin_page;
```

Suite à venir...

--- Fin du document ---

Historique du document

Auteur	Date	Observations
Mickaël DEVOLDÈRE	19/03/2018	Création du document

Crédits
