FORMATION EN PHP

I- Généralité :

Blogs, réseaux sociaux, site e-commerce, API… Vous l'aurez compris avec PHP les développeurs sont en mesure de développer toutes sorte de site web, d'ailleurs nous parlerons d'application web dans la mesure où ils exposent souvent des centaines de fonctionnalités.

Le langage PHP a justement été conçu pour créer des sites « vivants » (on parle de sites dynamiques). Et si vous voulez apprendre à créer vous aussi des **sites web dynamiques**, c'est votre jour de chance : vous êtes sur un cours pour vrais débutants en PHP !
L'essentiel, c'est de lire en entier les chapitres dans l'ordre. Après, ça passe tout seul et vous vous étonnerez bientôt de ce que vous êtes capable de faire !

À la fin de ce cours, vous serez capable de :

- Installer les outils propres à PHP (serveur web, logiciel de gestion de base de données)
- Respecter les conventions d'écriture de code
- Programmer des fonctions
- Ecrire des instructions en PHP
- Enregistrer des données dans un fichier
- Stocker des données dans les sessions et les cookies
- Transmettre des données
- Ecrire des fonctions SQL basiques
- Communiquer avec la base de données
- Utiliser les expressions régulières en PHP

Ce qui fait le succès du Web aujourd'hui, c'est à la fois sa simplicité et sa facilité d'accès. Un internaute lambda n'a pas besoin de savoir « **comment ça fonctionne derrière** ». Et heureusement pour lui.

En revanche, un apprenti webmaster tel que vous doit, avant toute chose, connaître les bases du fonctionnement d'un site web. Qu'est-ce qu'un serveur et un client ? Comment rend-on son site dynamique ? Que signifient PHP et MySQL ?

Ce premier chapitre est là pour répondre à toutes ces questions et vous montrer que vous êtes capables d'apprendre à créer des sites web dynamiques. Tous les lecteurs seront à la fin rassurés de savoir qu'ils commencent au même niveau !

II- Les sites statiques et dynamiques :

On considère qu'il existe deux types de sites web : les sites **statiques** et les sites **dynamiques**.

1- Les sites statiques :

Ce sont des sites réalisés uniquement à l'aide des langages HTML

et CSS. Ils fonctionnent très bien mais leur contenu ne peut pas être mis à jour automatiquement : il faut que le propriétaire du site (le webmaster) modifie le code source pour y ajouter des nouveautés. Ce n'est pas très pratique quand on doit mettre à jour son site plusieurs fois dans la même journée ! Les sites statiques sont donc bien adaptés pour réaliser des sites « vitrine », pour présenter par exemple son entreprise, mais sans aller plus loin. Ce type de site se fait de plus en plus rare aujourd'hui, car dès que l'on rajoute un élément d'interaction (comme un formulaire de contact), on ne parle plus de site statique mais de site dynamique.

2- Les sites dynamiques :

Ils sont plus complexes, ils utilisent d'autres langages en plus de HTML et CSS, tels que PHP et MySQL. Le contenu de ces sites web est dit « dynamique » parce qu'il peut changer sans l'intervention du webmaster ! La plupart des sites web que vous visitez aujourd'hui, y compris OpenClassrooms, sont des sites dynamiques. Le seul prérequis pour apprendre à créer ce type de sites est de déjà savoir réaliser des sites statiques en HTML et CSS.

L'éléphant, la mascotte de **PHP**

L'objectif de ce cours est de vous rendre capables de réaliser des sites web dynamiques entièrement par vous-mêmes, pas à pas.
En effet, ceux-ci peuvent proposer des fonctionnalités bien plus excitantes que les sites statiques. Voici quelques éléments que vous serez en mesure de réaliser :

- **un espace membres** : vos visiteurs peuvent s'inscrire sur votre site et avoir accès à

des sections qui leur sont réservées ;

- **un forum** : il est courant aujourd'hui de voir les sites web proposer un forum de

discussion pour s'entraider ou simplement passer le temps ;

- **un compteur de visiteurs** : vous pouvez facilement compter le nombre de visiteurs

qui se sont connectés dans la journée sur votre site, ou même connaître le nombre de visiteurs en train d'y naviguer !

- **des actualités** : vous pouvez automatiser l'écriture d'actualités, en offrant à vos

visiteurs la possibilité d'en rédiger, de les commenter, etc. ;

- **une newsletter** : vous pouvez envoyer un e-mail à tous vos membres régulièrement

pour leur présenter les nouveautés et les inciter ainsi à revenir sur votre site.

Bien entendu, ce ne sont là que des exemples. Il est possible d'aller encore plus loin, tout dépend de vos besoins. Sachez par exemple que la quasi-totalité des sites de jeux en ligne sont dynamiques. On retrouve notamment des sites d'élevage virtuel d'animaux, des jeux de conquête spatiale, etc.

Mais… ne nous emportons pas. Avant de pouvoir en arriver là, vous avez de la lecture et bien des choses à apprendre ! Commençons par la base : savez-vous ce qui se passe lorsque vous consultez une page web ?

3- Fonctionnement d'un site web :

Lorsque vous voulez visiter un site web, vous tapez son adresse dans votre navigateur web, que ce soit Mozilla Firefox, Internet Explorer, Opera, Safari ou un autre. Mais ne vous êtes-vous jamais demandé comment faisait la page web pour arriver jusqu'à vous ?

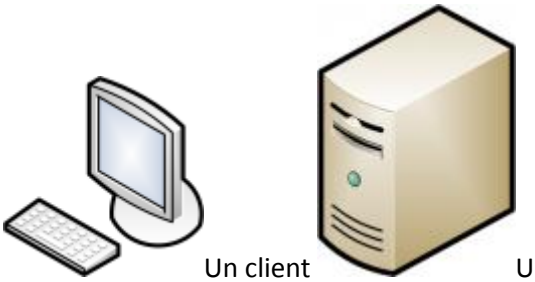
Il faut savoir qu'Internet est un réseau composé d'ordinateurs. Ceux-ci peuvent être classés en deux catégories.

- **Les clients** : ce sont les ordinateurs des internautes comme vous. Votre ordinateur fait

donc partie de la catégorie des clients. Chaque client représente un visiteur d'un site web. Dans les schémas qui vont suivre, l'ordinateur d'un client sera représenté par l'image suivante.

- **Les serveurs** : ce sont des ordinateurs puissants qui stockent et délivrent des sites web

aux internautes, c'est-à-dire aux clients. La plupart des internautes n'ont jamais vu un serveur de leur vie. Pourtant, les serveurs sont indispensables au bon fonctionnement du Web. Sur les prochains schémas, un serveur sera représenté par l'image de la figure suivante.



Un client Un serveur

La plupart du temps, le serveur est dépourvu d'écran : il reste allumé et travaille tout seul sans intervention humaine, 24 h/24, 7 j/7. Un vrai forçat du travail.

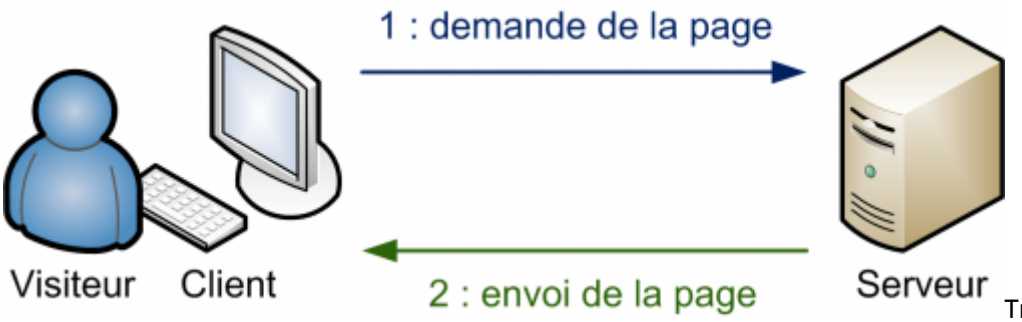
On résume : votre ordinateur est appelé **le client**, tandis que l'ordinateur qui détient le site web est appelé **le serveur**. Comment les deux communiquent-ils ?

C'est justement là que se fait la différence entre un site statique et un site dynamique. Voyons ensemble ce qui change.

Cas d'un site statique

Lorsque le site est statique, le schéma est très simple. Cela se passe en deux temps, ainsi que vous le schématise la figure suivante :

1. le client demande au serveur à voir une page web ;
2. le serveur lui répond en lui envoyant la page réclamée.



Transferts avec un site statique

La communication est donc plutôt basique :

- « Bonjour, je suis le client, je voudrais voir cette page web. »
- « Tiens, voilà la page que tu m'as demandée. »

Sur un site statique, il ne se passe rien d'autre. Le serveur stocke des pages web et les envoie aux clients qui les demandent sans les modifier.

Cas d'un site dynamique

Lorsque le site est dynamique, il y a une étape intermédiaire : la page est **générée** (fig. suivante).

- Le client demande au serveur à voir une page web ;
- le serveur prépare la page spécialement pour le client ;
- le serveur lui envoie la page qu'il vient de générer.

Transfert avec un site dynamique

La page web est générée à chaque fois qu'un client la réclame. C'est précisément ce qui rend les sites dynamiques vivants : le contenu d'une même page peut changer d'un instant à l'autre.

C'est comme cela que certains sites parviennent à afficher par exemple votre pseudonyme sur toutes les pages. Étant donné que le serveur génère une page à chaque fois qu'on lui en demande une, il peut la personnaliser en fonction des goûts et des préférences du visiteur (et afficher, entre autres, son pseudonyme).

Les langages du Web

Lorsqu'on crée un site web, on est amené à manipuler non pas un mais plusieurs langages. En tant que webmaster, il faut impérativement les connaître.

Certains programmes, appelés WYSIWYG (What You See Is What You Get), permettent d'aider les plus novices à créer un site web statique sans connaître les langages informatiques qui se cachent derrière… Mais pour réaliser un site dynamique comme nous le souhaitons, nous devrons absolument mettre les mains dans le cambouis.

Pour un site statique : HTML et CSS

De nombreux langages ont été créés pour produire des sites web. Deux d'entre eux constituent une base incontournable pour tous les webmasters.

- **HTML** : c'est le langage à la base des sites web. Simple à apprendre, il fonctionne à

partir de balises. Voici un exemple de code HTML :

```
<p>Bonjour, je suis un <em>paragraphe</em> de texte !</p>
```

- **CSS** : c'est le langage de mise en forme des sites web. Alors que le HTML permet

d'écrire le contenu de vos pages web et de les structurer, le langage CSS s'occupe de la mise en forme et de la mise en page. C'est en CSS que l'on choisit notamment la couleur, la taille des menus et bien d'autres choses encore. Voici un code CSS :

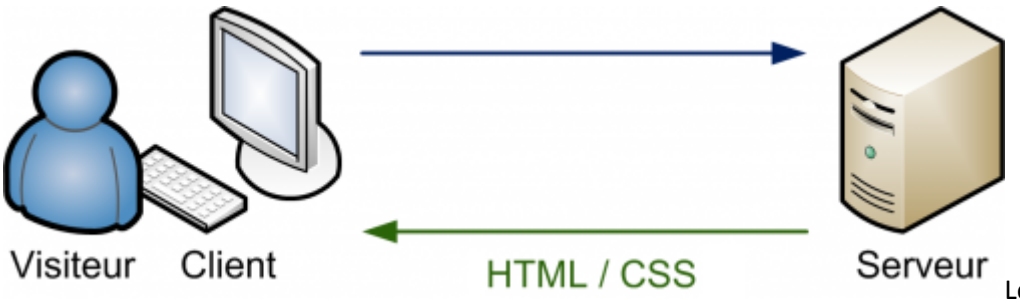
```
div.banner {

text-align: center;

font-weight: bold;

font-size: 120%;

}
```

Ces langages sont la base de tous les sites web. Lorsque le serveur envoie la page web au client, il envoie en fait du code en langage HTML et CSS.

Le problème, c'est que lorsqu'on connaît **seulement** HTML et CSS, on ne peut produire que des sites statiques… et non des sites dynamiques ! Pour ces derniers, il est nécessaire de manipuler d'autres langages en plus de HTML et CSS.

La question qu'il faut vous poser est donc : connaissez-vous HTML et CSS ?

Si oui, c'est parfait, vous pouvez continuer car nous en aurons besoin par la suite. Si la réponse est non, pas de panique. Ces langages ne sont pas bien difficiles, ils sont à la portée de tous. Vous pouvez les apprendre en lisant mon cours sur HTML et CSS.
Sachez qu'apprendre ces langages n'est l'affaire que de quelques petites semaines, voire moins si vous avez suffisamment de temps libre.



Le serveur envoie du HTML et du CSS au client

Pour un site dynamique : ajoutez PHP et MySQL

Quel que soit le site web que l'on souhaite créer, HTML et CSS sont donc indispensables. Cependant, ils ne suffisent pas pour réaliser des sites dynamiques. Il faut les compléter avec d'autres langages.

C'est justement tout l'objet de ce cours : vous allez apprendre à manipuler PHP et MySQL pour réaliser un site web dynamique.

- **PHP** : c'est un langage que seuls les serveurs comprennent et qui permet de rendre

votre site dynamique. C'est PHP qui « génère » la page web comme on l'a vu sur un des schémas précédents.

Ce sera le premier langage que nous découvrirons dans ce cours. Il peut fonctionner seul, mais il ne prend vraiment de l'intérêt que s'il est combiné à un outil tel que MySQL. Voici un code PHP :

```
<?php echo « Vous êtes le visiteur n° » . $nbre_visiteurs; ?>
```

- **MySQL** : c'est ce qu'on appelle un SGBD (Système de Gestion de Base de Données).

Pour faire simple, son rôle est d'enregistrer des données de manière organisée afin de vous aider à les retrouver facilement plus tard. C'est grâce à MySQL que vous pourrez enregistrer la liste des membres de votre site, les messages postés sur le forum, etc. Le langage qui permet de communiquer avec la base de données s'appelle le SQL. Voici un code en langage SQL :

```
SELECT id, auteur, message, datemsg FROM livreor ORDER BY datemsg DESC LIMIT 0, 10
```

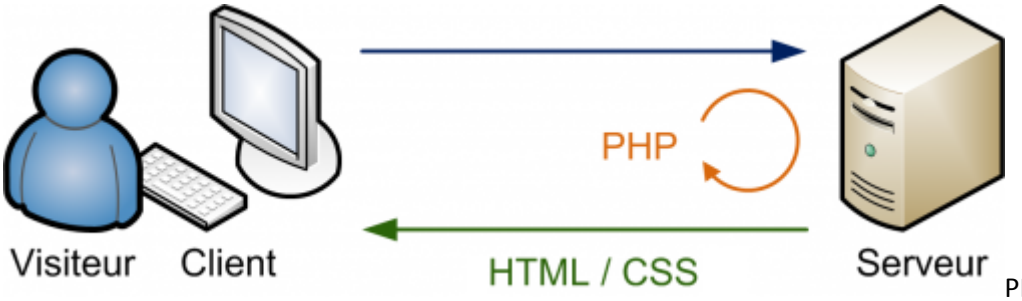
PHP et MySQL sont ce qu'on appelle des logiciels libres. Entre autres choses, cela vous donne des garanties de pérennité : tout le monde peut contribuer à leur développement, vous ne risquez donc pas de voir tous les webmasters se désintéresser de PHP et de MySQL du jour au lendemain, et ça c'est très important !
D'autre part, PHP et MySQL sont disponibles gratuitement. Cela signifie une chose essentielle : vous n'aurez pas à débourser un centime pour construire votre site web !

PHP peut fonctionner seul et suffit à créer un site dynamique, mais les choses deviennent réellement intéressantes lorsqu'on le combine à un SGBD tel que MySQL. Cependant pour simplifier, oublions pour le moment MySQL et concentrons-nous sur PHP.

PHP génère du HTML

Les clients sont incapables de comprendre le code PHP : ils ne connaissent que le HTML et le CSS. Seul le serveur est capable de lire du PHP.

Le rôle de PHP est justement de générer du code HTML (on peut aussi générer du CSS, mais c'est plus rare), code qui est ensuite envoyé au client de la même manière qu'un site statique, comme le montre la fig. suivante.



PHP décide ce qui va être affiché sur la page web envoyée au visiteur

PHP est un langage de programmation utilisé sur de nombreux serveurs pour prendre des décisions. C'est PHP qui décide du code HTML qui sera généré et envoyé au client à chaque fois.

Pour bien comprendre l'intérêt de tout cela, prenons un exemple. On peut écrire en PHP : « **Si le visiteur est membre de mon site et qu'il s'appelle Jonathan, affiche**Bienvenue Jonathan **sur la page web. En revanche, si ce n'est pas un membre de mon site, affiche** Bienvenue **à la place et propose au visiteur de s'inscrire.** »
C'est un exemple très basique de site dynamique : selon que vous êtes un membre enregistré ou non, vous ne verrez pas les mêmes choses et n'aurez peut-être pas accès à toutes les sections.

Et la concurrence ?

HTML et CSS n'ont pas de concurrents car ce sont des standards. Tout le monde est censé les connaître et les utiliser sur tous les sites web.

En revanche, pour ce qui est des sites dynamiques, PHP et MySQL sont loin d'être les seuls sur le coup. Je ne peux pas vous faire une liste complète de leurs concurrents, ce serait bien trop long (et ennuyeux !). Cependant, pour votre culture générale, il faut au moins connaître quelques autres grands noms.

Tout d'abord, si on a souvent tendance à combiner PHP et MySQL pour réaliser de puissants sites dynamiques, il ne faut pas mélanger les deux. Le premier a des concurrents différents du second.

Les concurrents de PHP

Parmi les concurrents de PHP, on peut citer les suivants :

- **ASP .NET** : conçu par Microsoft, il exploite le framework (c'est-à-dire un ensemble

de bibliothèques qui fournissent des services pour les développeurs) .NET bien connu des développeurs C#. Ce langage peut être intéressant si vous avez l'habitude de développer en C# .NET et que vous ne voulez pas être dépaysés.

- **Ruby on Rails** : très actif, ce framework s'utilise avec le langage Ruby et permet de

réaliser des sites dynamiques rapidement en suivant certaines conventions.

- **Django** : il est similaire à Ruby on Rails, mais il s'utilise en langage Python.
- **Java et les JSP (Java Server Pages)** : plus couramment appelé « **JEE** » ou « **Java**

EE », il est particulièrement utilisé dans le monde professionnel. Il demande une certaine rigueur. La mise en place d'un projet JEE est traditionnellement un peu plus longue et plus lourde mais le système est apprécié des professionnels et des institutions. C'est ce qui est utilisé sur le site des impôts français, par exemple.

Ruby on Rails

Je ne peux pas présenter ici tous les concurrents, mais cela devrait déjà vous donner une bonne idée. Pour information, il est aussi possible d'utiliser par exemple le langage C ou le C++, bien que ce soit plus complexe et pas forcément toujours très adapté (en clair, je ne le recommande pas du tout).

Lequel choisir dans le lot ? Lequel est le **meilleur** ?

Étant donné l'objet de ce cours, vous vous attendez à ce que je vous réponde instantanément « PHP ! ». Mais non. En fait, tout dépend de vos connaissances en programmation. Si vous avez déjà manipulé le Java, vous serez plus rapidement à l'aise avec les JSP. Si vous connaissez Python, Django semble tout indiqué.

Quant à PHP, il se démarque de ses concurrents par une importante communauté qui peut vous aider rapidement sur Internet si vous avez des problèmes. C'est un langage facile à utiliser, idéal pour les débutants comme pour les professionnels : Wikipédia et Facebook sont des exemples de sites célèbres et très fréquentés qui fonctionnent grâce à PHP.

Bref, il n'y a pas de meilleur choix. Je vous recommande le langage pour lequel vous serez certains d'avoir quelqu'un pour vous aider. PHP en ce sens est souvent un très bon choix.

Les concurrents de MySQL

En ce qui concerne les bases de données, le choix est là encore très vaste.
Cependant, alors que PHP et ses concurrents sont la plupart du temps libres et gratuits, ce n'est pas le cas de la plupart des SGBD.

Parmi les concurrents de MySQL, je vous conseille de connaître (au moins de nom) les suivants :

- **Oracle** : c'est le SGBD le plus célèbre, le plus complet et le plus puissant. Il est

malheureusement payant (et cher), ce qui le réserve plutôt aux entreprises qui l'utilisent déjà massivement. Il existe cependant des versions gratuites d'Oracle, notamment pour ceux qui veulent apprendre à s'en servir.

- **MariaDB** : variante libre de MySQL, qui a été créée depuis que ce dernier a été

racheté par… Oracle. Oui, je sais, on peut penser que ce monde est fou ! MySQL appartient à Oracle, mais reste bien une base de données à part. MariaDB est une copie (fork) de MySQL qui a la volonté de rester libre et indépendante.

- **Microsoft SQL Server** : édité par Microsoft, on l'utilise souvent en combinaison avec

ASP .NET, bien qu'on puisse l'utiliser avec n'importe quel autre langage. Il est payant, mais il existe des versions gratuites limitées.

- **PostgreSQL** : il s'agit d'un SGBD libre et gratuit comme MySQL, qui propose des

fonctionnalités plus avancées. Parfois comparé à Oracle, il lui reste cependant du chemin à parcourir. Il dispose d'une communauté un peu moins importante que MySQL et Oracle. OpenClassrooms utilise PostgreSQL.

Oracle

Là encore, cette liste est loin d'être exhaustive mais vous présente au moins quelques grands noms.
Pour information, MySQL reste de loin le SGBD libre et gratuit le plus utilisé. Parmi les solutions professionnelles payantes, Oracle est le plus avancé et le plus répandu mais son utilisation est surtout réservée aux grosses entreprises.

En fin de compte, si vos moyens sont limités, vous n'avez pas beaucoup de choix pour le SGBD. MySQL est le plus indiqué car il est libre, gratuit, performant et utilisé par de nombreuses personnes qui sont susceptibles de vous aider.

Plusieurs combinaisons sont possibles

Comme vous avez pu le constater, vous avez le choix entre de nombreux outils pour réaliser un site dynamique. La plupart d'entre eux sont gratuits.

Sachez que vous pouvez a priori combiner ces outils comme bon vous semble. Par exemple, on peut très bien utiliser PHP avec une autre base de données que MySQL, telle que Oracle ou PostgreSQL. De même, MySQL peut être utilisé avec n'importe quel autre langage : Java, Python, Ruby, etc.

Cependant, la combinaison « **PHP + MySQL** » est probablement la plus courante. Ce n'est pas par hasard si ce cours traite de ces deux outils qui ont fait leurs preuves.

En résumé :

- Il existe deux types de sites web :
  - les sites **statiques** : réalisés en HTML et CSS, leur contenu ne peut être mis à jour que par le webmaster ;

- les sites **dynamiques** : réalisés avec d'autres outils comme PHP et MySQL en

plus de HTML et CSS, ils permettent aux visiteurs de participer à la vie du site, de poster des messages… bref, de rendre le site vivant !

- Les visiteurs du site sont appelés les clients. Ils demandent au serveur qui héberge le

site de leur transmettre les pages web.

- PHP est un langage exécuté par le serveur. Il permet de personnaliser la page en

fonction du visiteur, de traiter ses messages, d'effectuer des calculs, etc. Il génère une page HTML.

- MySQL est un système de gestion de bases de données. Il se charge du stockage des

informations (liste des messages, des membres…).

III-Préparation de l'environnement de travail :

Nous savons désormais que PHP s'exécute sur le serveur et que son rôle est de générer des pages web. Cependant, seul un serveur peut lire du PHP ; or votre ordinateur n'est pas un serveur. Comment diable allez-vous pouvoir créer un site dynamique si PHP ne fonctionne pas chez vous ?

Qu'à cela ne tienne : nous allons temporairement transformer votre ordinateur en serveur pour que vous puissiez exécuter du PHP et travailler sur votre site dynamique. Vous serez fin prêts à programmer après avoir lu ce chapitre !

De quels programmes a-t-on besoin ?

Selon que l'on crée un site statique ou un site dynamique, on a besoin de logiciels différents. En fait, faire un site dynamique nécessite malheureusement pour nous quelques logiciels supplémentaires !

Avec un site statique

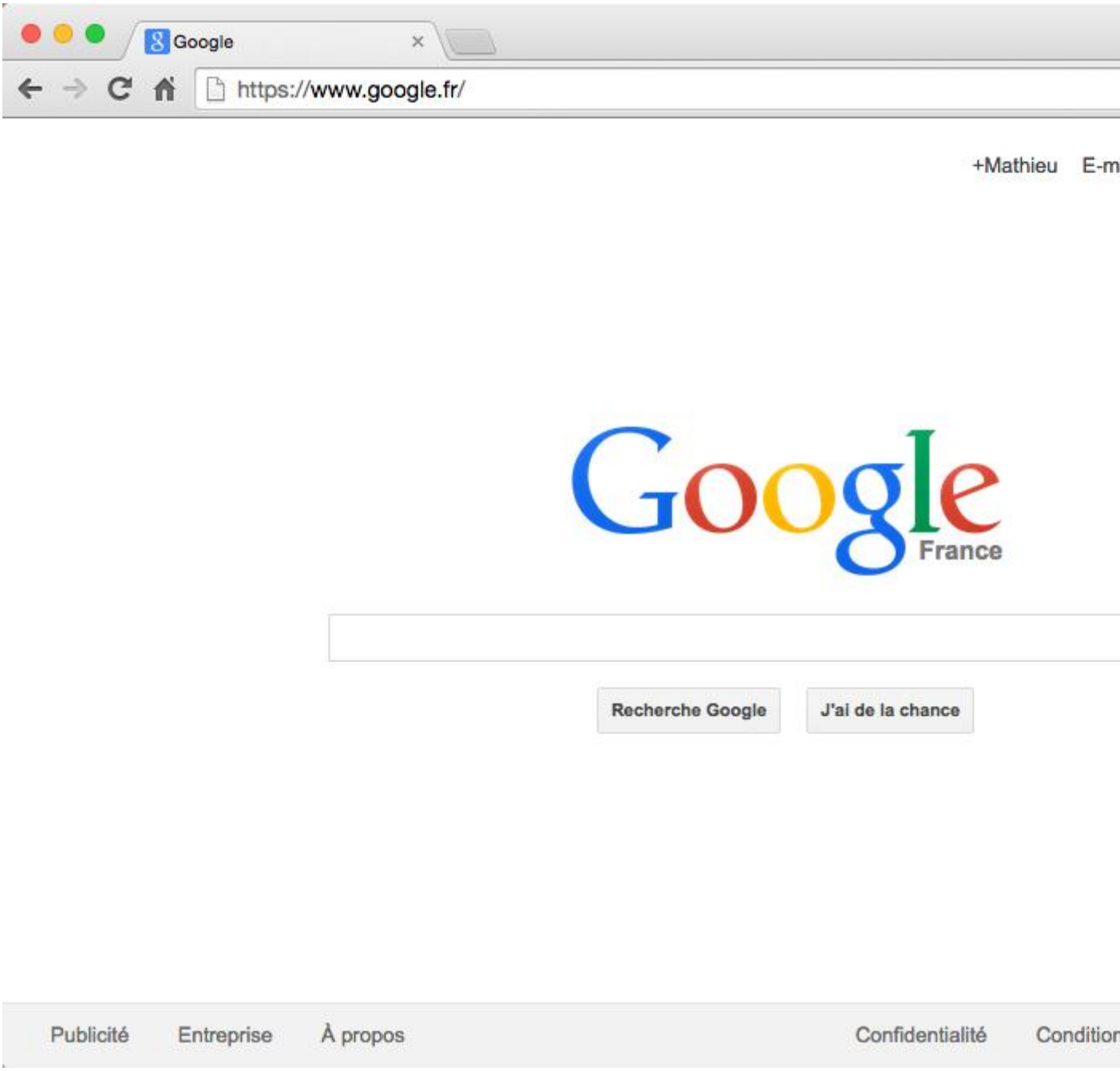
Les webmasters qui créent des sites statiques avec HTML et CSS ont de la chance, ils ont en général déjà tous les programmes dont ils ont besoin.

- **Un éditeur de texte** : en théorie, un programme tel que le Bloc-notes livré avec

Windows suffit, bien qu'il soit recommandé d'utiliser un outil un peu plus évolué comme Notepad++. Nous reparlerons du choix de l'éditeur à la fin de ce chapitre.

- **Un navigateur web** : il permet de tester la page web. Vous pouvez utiliser par

exemple Mozilla Firefox, Internet Explorer, Google Chrome, Opera, Safari, ou tout autre navigateur auquel vous êtes habitués pour aller sur le web. Il est conseillé de tester son site régulièrement sur différents navigateurs.



Google Chrome

Cependant, pour ceux qui comme nous travaillent sur des sites dynamiques, ces outils ne suffisent pas. Il est nécessaire d'installer des programmes supplémentaires.

Avec un site dynamique

Pour que votre ordinateur puisse lire du PHP, il faut qu'il se comporte comme un serveur. Rassurez-vous, vous n'avez pas besoin d'acheter une machine spéciale pour cela : il suffit simplement d'installer les mêmes programmes que ceux que l'on trouve sur les serveurs qui délivrent les sites web aux internautes.

Ces programmes dont nous allons avoir besoin, quels sont-ils ?

- **Apache** : c'est ce qu'on appelle un serveur web. Il s'agit du plus important de tous les

programmes, car c'est lui qui est chargé de délivrer les pages web aux visiteurs. Cependant, Apache ne gère que les sites web statiques (il ne peut traiter que des pages HTML). Il faut donc le compléter avec d'autres programmes.

- **PHP** : c'est un plug-in pour Apache qui le rend capable de traiter des pages web

dynamiques en PHP. En clair, en combinant Apache et PHP, notre ordinateur sera capable de lire des pages web en PHP.

- **MySQL** : c'est le logiciel de gestion de bases de données dont je vous ai parlé en

introduction. Il permet d'enregistrer des données de manière organisée (comme la liste des membres de votre site). Nous n'en aurons pas besoin immédiatement, mais autant l'installer de suite.

Logo d'Apache

Tous ces éléments qui vont nous aider à créer notre site dynamique sont libres et gratuits. Certes, il en existe d'autres (parfois payants), mais la combinaison Apache + PHP + MySQL est la plus courante sur les serveurs web, à tel point qu'on a créé des « packs » tout prêts qui contiennent tous ces éléments. Il est possible de les installer un à un mais cela prend plus de temps et vous n'allez rien y gagner (sauf si vous êtes administrateur de serveur, ce qui ne devrait pas être votre cas).

Dans la suite de ce chapitre, nous allons voir comment installer le « pack » qui convient en fonction de votre système d'exploitation.

Sous Windows : WAMP

Il existe plusieurs paquetages tout prêts pour Windows. Je vous propose d'utiliser WAMP Server qui a l'avantage d'être régulièrement mis à jour et disponible en français.

Commencez par télécharger WAMP ici

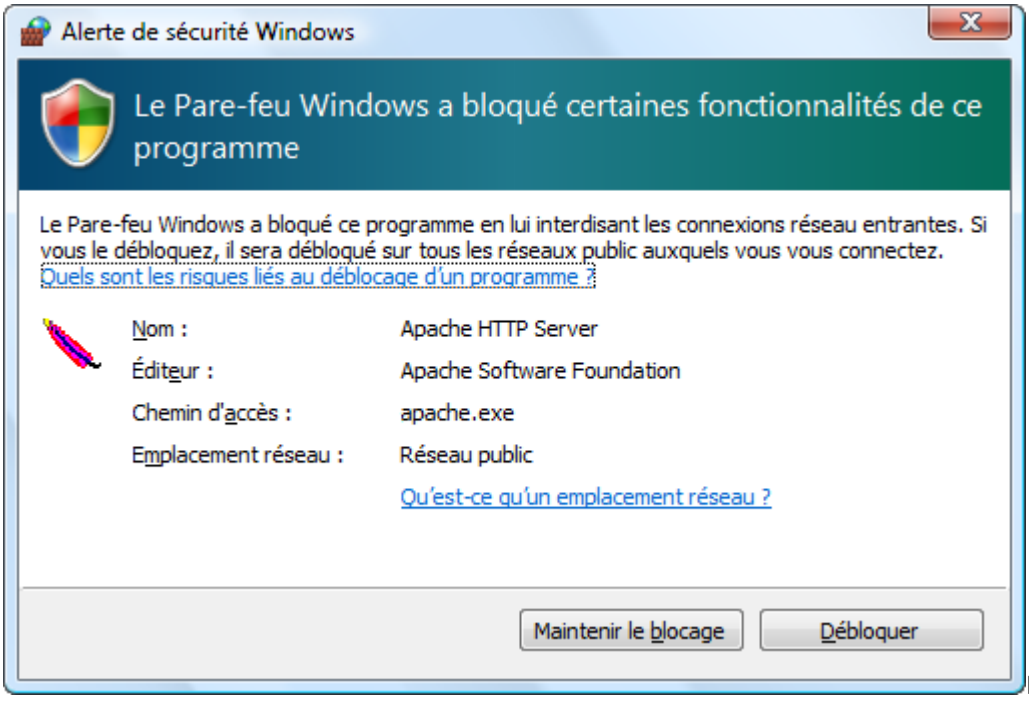
Vérifiez que vous prenez une version de WAMP avec PHP 5.4 minimum. Les fonctionnalités de PHP changent d'une version à l'autre et ce cours est basé sur les versions de PHP >= 5.4. De préférence, basez-vous désormais sur PHP 7.

Une fois téléchargé, installez-le en laissant toutes les options par défaut. Il devrait s'installer dans un répertoire comme`C:\wamp`et créer un raccourci dans le menu *Démarrer*.

Lorsque vous lancez WAMP, une icône doit apparaître en bas à droite de la barre des tâches, à côté de l'horloge, comme sur la figure suivante.

Icône de WAMP

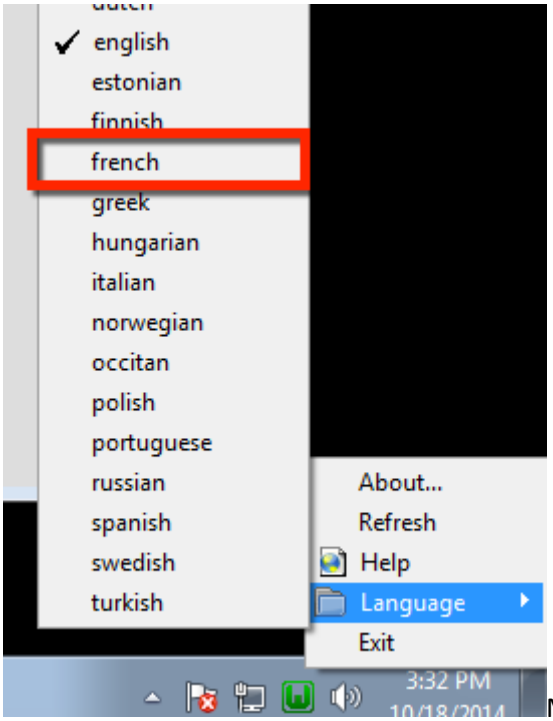
Si une fenêtre apparaît pour vous indiquer que le pare-feu bloque Apache, cliquez sur *Autoriser l'accès* (fig. suivante). Vous n'avez aucune raison de vous inquiéter, c'est parfaitement normal.



Firewall et Apache

Si WAMP ne se lance pas correctement malgré tout, vérifiez que vous n'avez pas Skype ouvert en même temps. Les deux programmes ne peuvent pas tourner en parallèle (ils utilisent les mêmes ports de communication sur votre machine). Dans ce cas, fermez Skype pendant que vous utilisez WAMP.

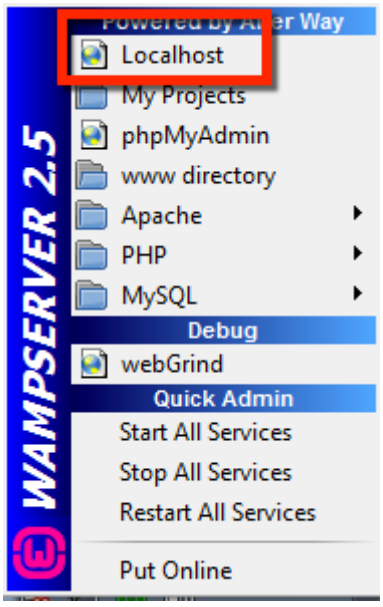
Par défaut, WAMP est en anglais. Vous pouvez facilement le passer en français en faisant un clic droit sur l'icône de WAMP dans la barre des tâches, puis en allant dans le menu *Language / french* (fig. suivante).



Modification de la langue

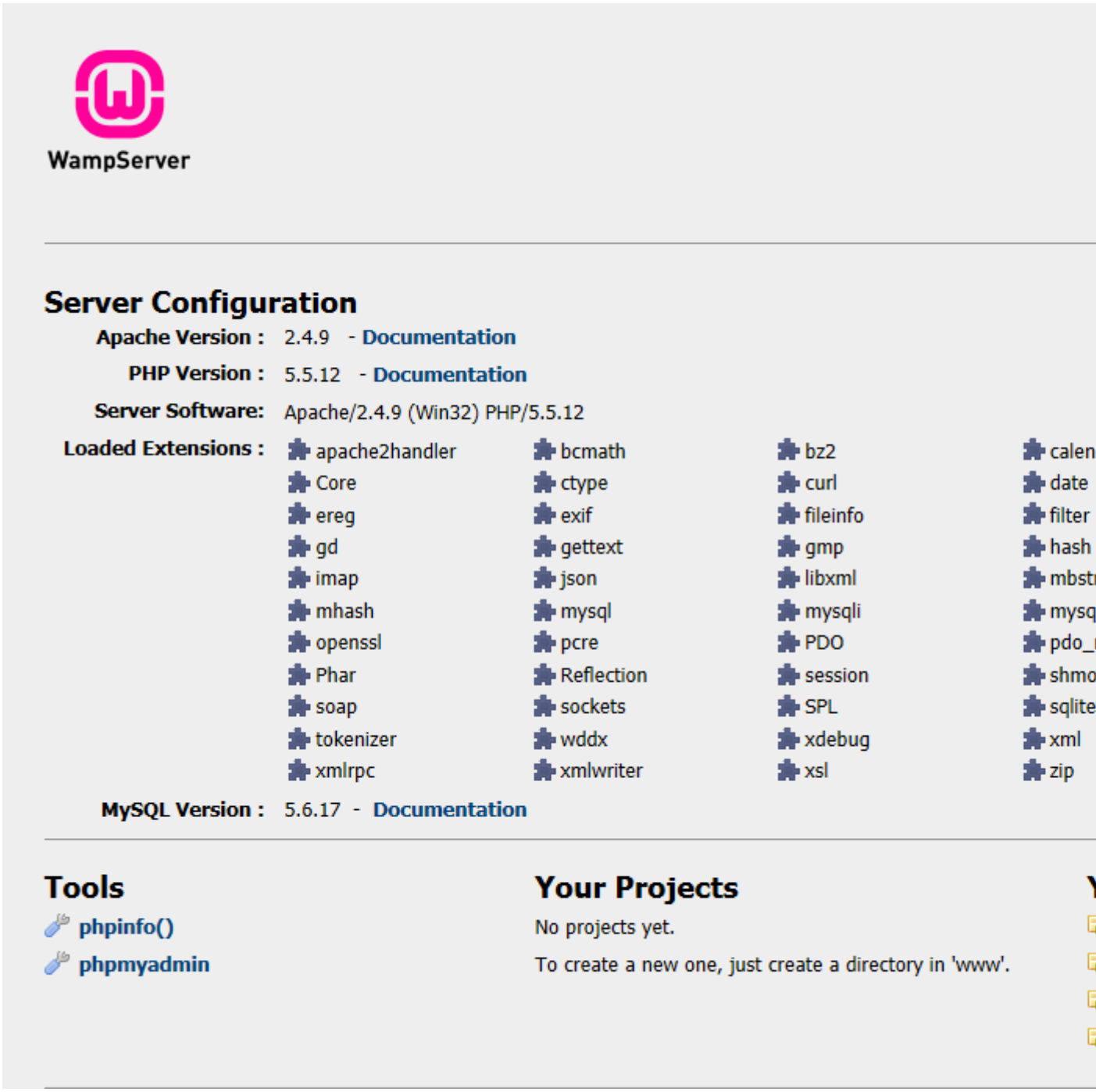
WAMP est maintenant en français !

Vous pouvez alors lancer la page d'accueil de WAMP. Faites un clic gauche sur l'icône de WAMP (attention, j'ai bien dit un **clic gauche** cette fois), puis cliquez sur *Localhost*, comme le montre la figure suivante.



Menu localhost de WAMP

Une page web similaire à la capture de la figure suivante devrait s'ouvrir dans votre navigateur favori (Firefox, par exemple). Si la page s'affiche chez vous, cela signifie qu'Apache fonctionne.



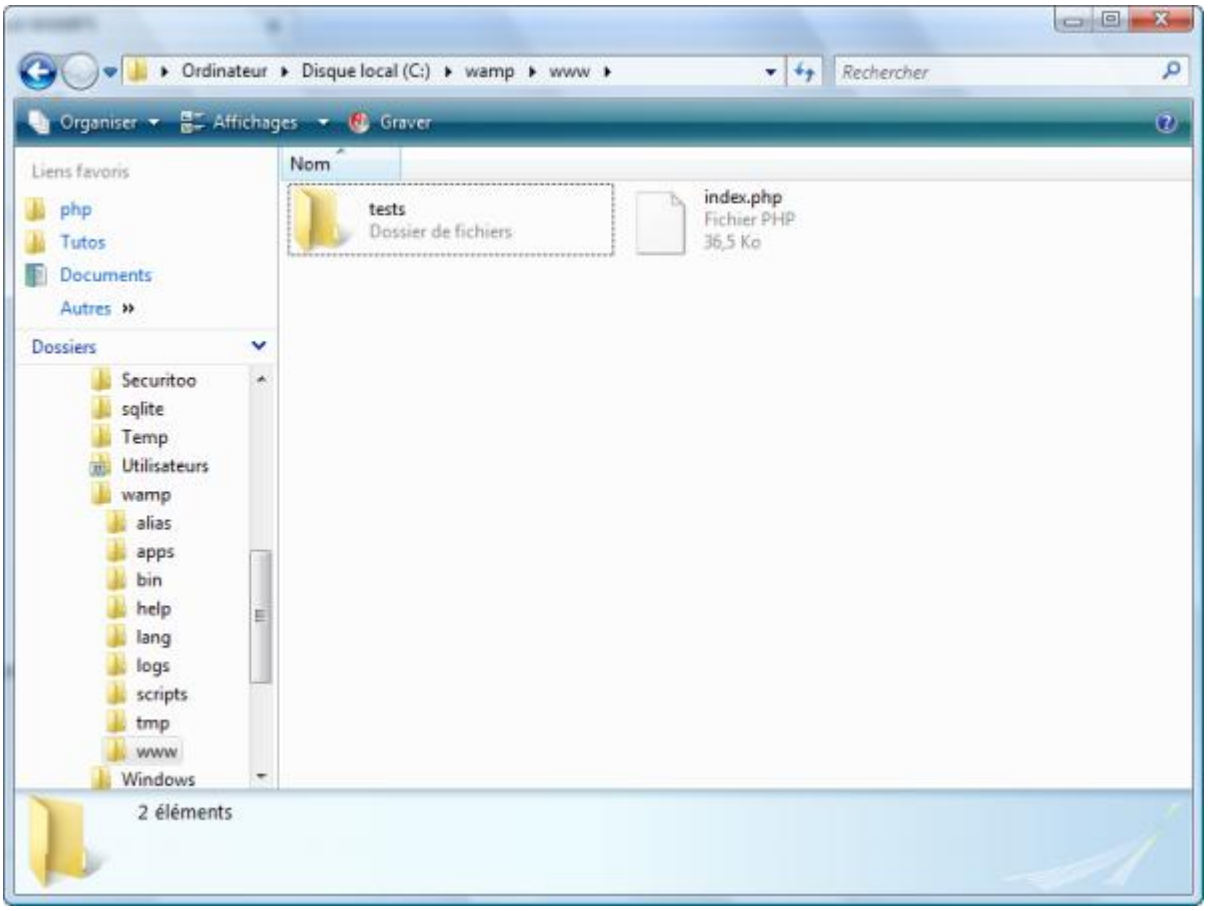
Page d'accueil de WAMP

La page web que vous voyez à l'écran vous a été envoyée par votre propre serveur Apache que vous avez installé en même temps que WAMP. Vous êtes en train de simuler le fonctionnement d'un serveur web sur votre propre machine. Pour le moment, vous êtes le seul internaute à pouvoir y accéder. On dit que l'on travaille « **en local** ». Notez que l'URL affichée par le navigateur dans la barre d'adresse est`http://localhost/`, ce qui signifie que vous naviguez sur un site web situé sur votre propre ordinateur.

La section « Vos projets » de la page d'accueil de WAMP doit indiquer qu'aucun projet n'existe pour le moment. Considérez que chaque site web que vous entreprenez de faire est un nouveau projet.

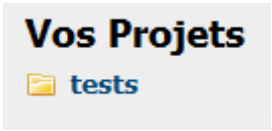
Nous allons créer un projet de test que nous appellerons`tests`. Pour ce faire, ouvrez l'explorateur Windows et rendez-vous dans le dossier où WAMP a été installé, puis dans le sous-dossier intitulé`www`. Par exemple :`C:\wamp\www`.

Une fois dans ce dossier, créez un nouveau sous-dossier que vous appellerez`tests`, comme le suggère l'image suivante.



Dossier créé pour WAMP

Retournez sur la page d'accueil de WAMP et actualisez-la (vous pouvez appuyer sur la touche F5). La section « Vos projets » devrait maintenant afficher « tests » car WAMP a détecté que vous avez créé un nouveau dossier (fig. suivante).



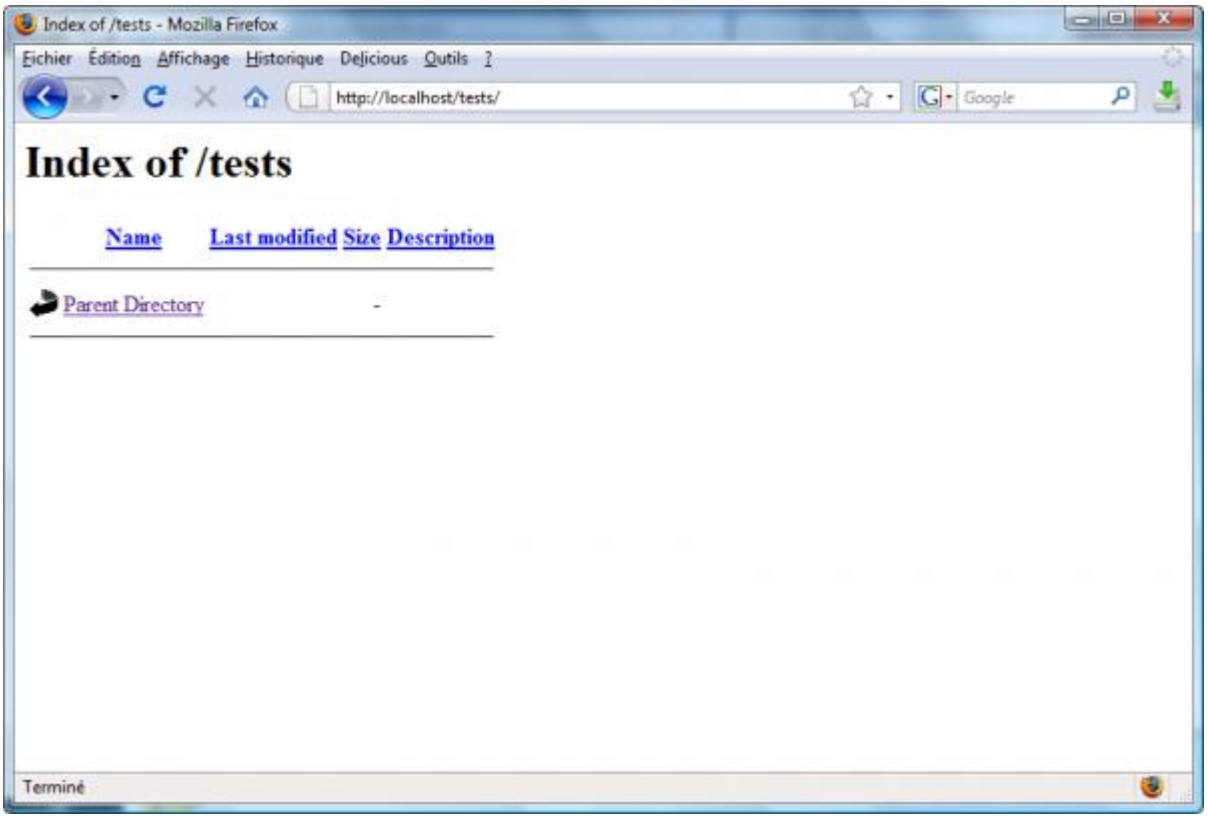
Projets dans WAMP

Vous créerez là-dedans vos premières pages web en PHP.

En revanche, si vous cliquez sur ce lien « tests », une erreur s'affiche… Je ne vais pas rentrer dans les détails du comment du pourquoi ( ), je vais plutôt vous demander de saisir l'adresse suivante dans la barre d'adresse de votre navigateur et tout ira bien :

http://localhost/tests/

Vous devriez voir une page vide comme dans la figure suivante.



Le contenu est pour le moment vide

Si vous avez le même résultat, cela signifie que tout fonctionne. Bravo, vous avez installé WAMP et il fonctionne correctement. Vous êtes prêts à programmer en PHP !

Vous pouvez passer les sections suivantes qui ne concernent que les utilisateurs sous Mac OS X et Linux.

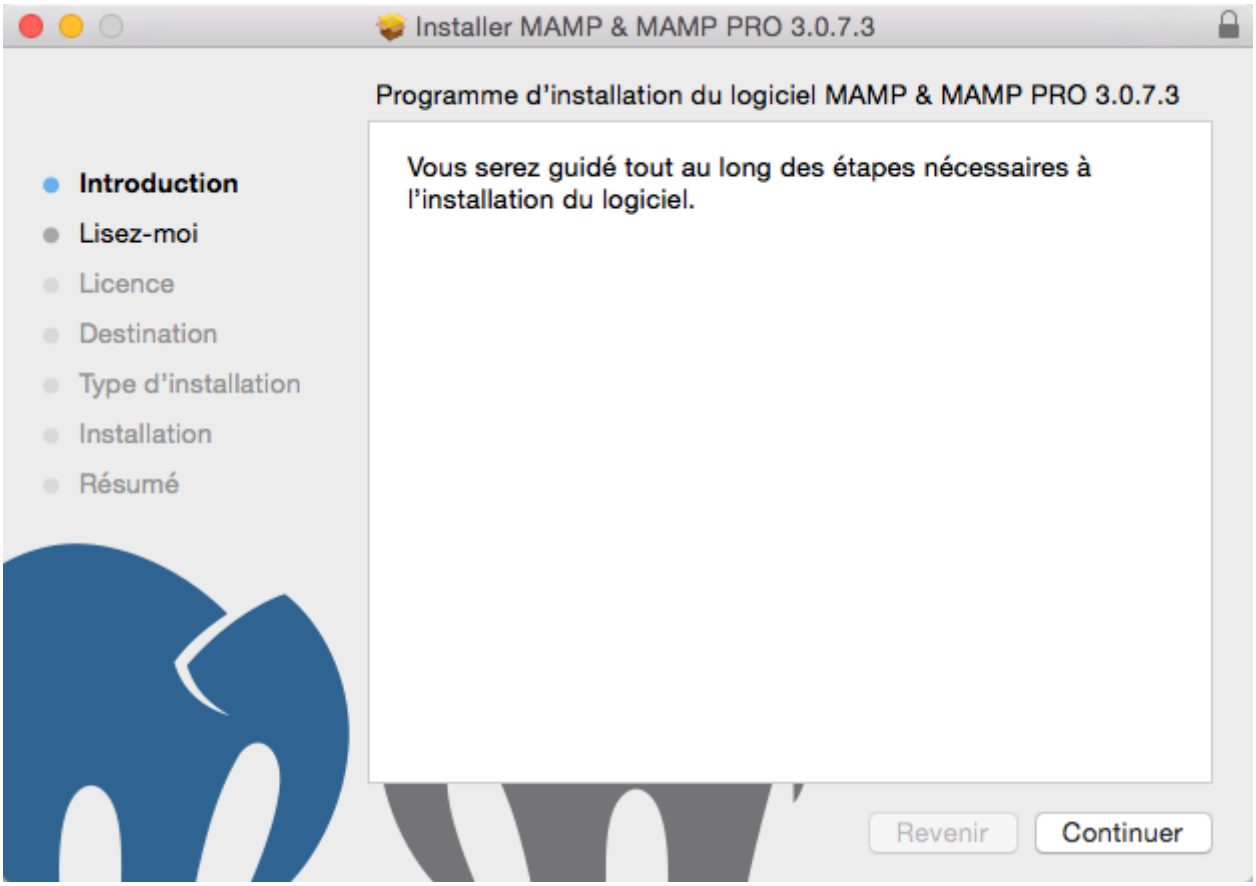
Sous Mac OS X : MAMP

Pour ceux qui ont un Mac sous Mac OS X, je vous conseille le programme MAMP (Mac Apache MySQL PHP). Il est vraiment très simple à installer et à utiliser.
Vous pouvez le télécharger ici.

Vérifiez que vous prenez une version de MAMP avec PHP 5.4 minimum. Les fonctionnalités de PHP changent d'une version à l'autre et ce cours est basé sur les versions de PHP >= 5.4. La version PHP 7 est la plus récente : utilisez-la autant que possible, elle est plus rapide et sera compatible avec le cours.

Normalement, c'est le cas si vous suivez bien le lien de téléchargement proposé.

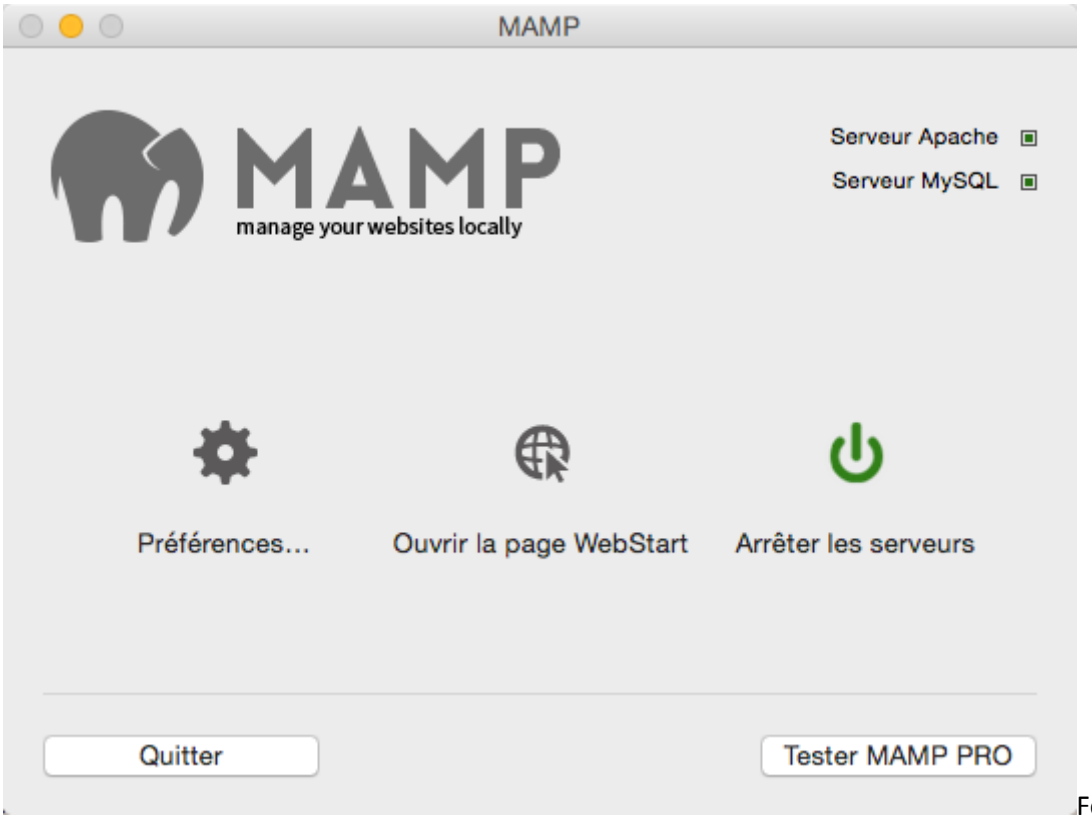
Vous devriez avoir téléchargé une archive un installeur .pkg. Il n'y a qu'à se laisser guider :



Installation de MAMP

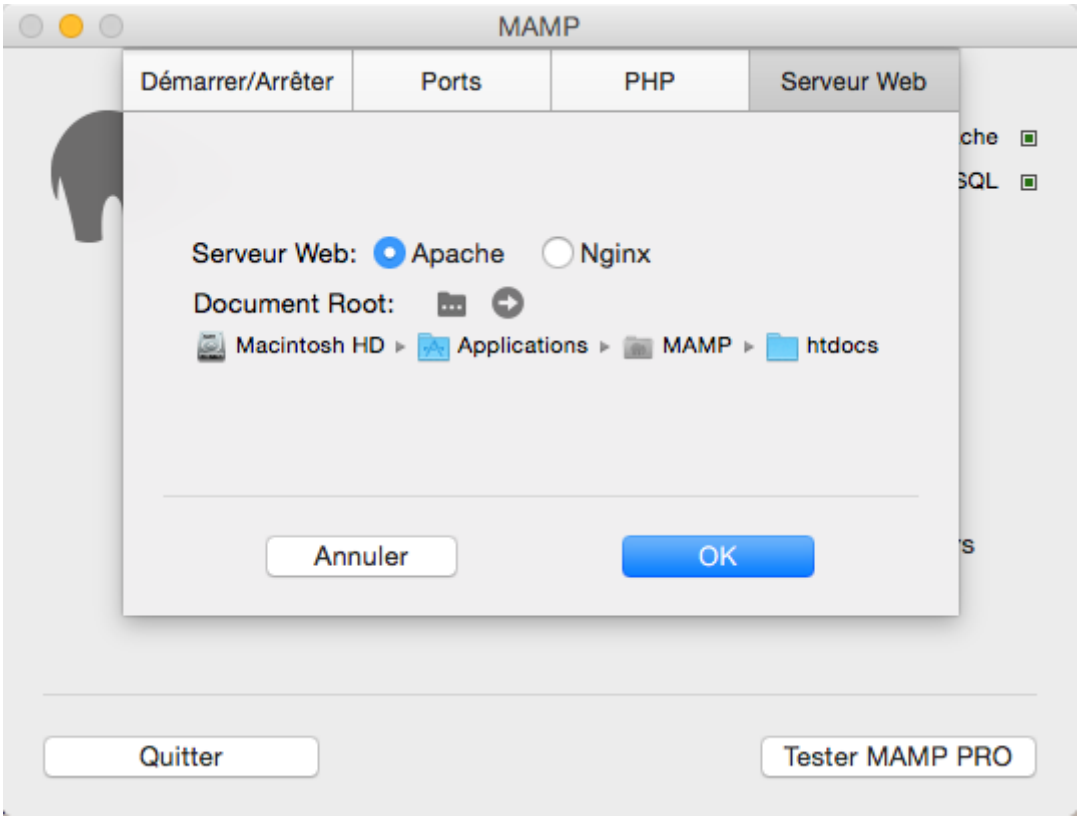
Vous devez tout simplement faire glisser le dossier MAMP en bas à gauche vers le dossier « Applications » au-dessus.

MAMP est maintenant installé. Vous le trouverez dans votre dossier « Applications ». La fenêtre principale de MAMP indique que les serveurs Apache et MySQL ont été correctement démarrés. Les cases des serveurs en haut à droite doivent être cochées comme sur la figure suivante :



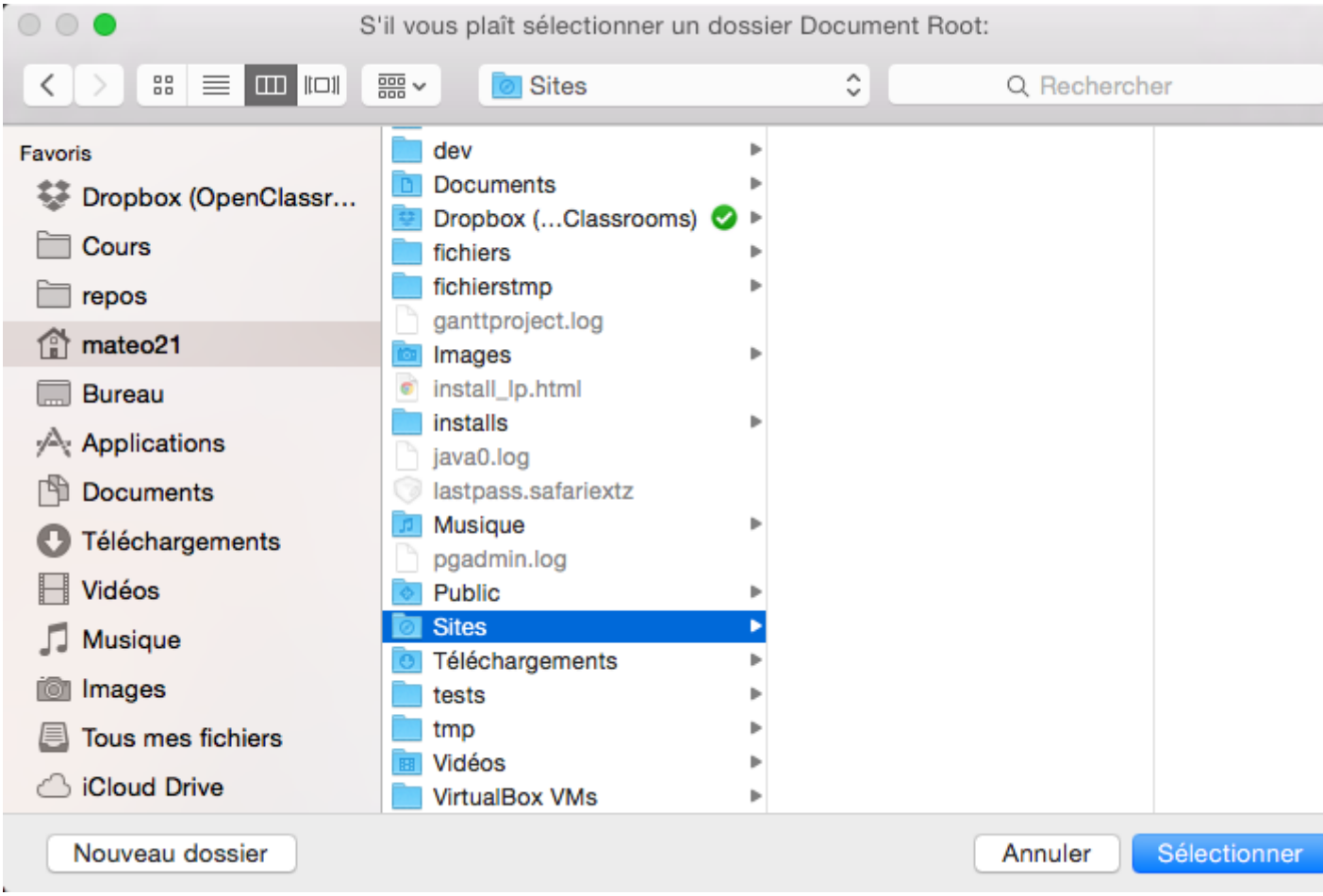
Fenêtre principale de MAMP

Je vous invite à configurer le répertoire dans lequel Apache ira chercher les fichiers PHP de votre site web. Pour cela, cliquez sur le bouton « Préférences » de la fenêtre principale. Une boîte de dialogue de configuration s'ouvre (figure suivante). Cliquez sur l'onglet Serveur Web en haut.



Configuration de MAMP

Cliquez sur l'icône « … » pour sélectionner le dossier dans lequel vous placerez les fichiers de votre site web. Sous Mac OS, un dossier est déjà créé : il s'agit de « Sites », dans votre répertoire personnel (fig. suivante).



Le dossier Sites de Mac OS X

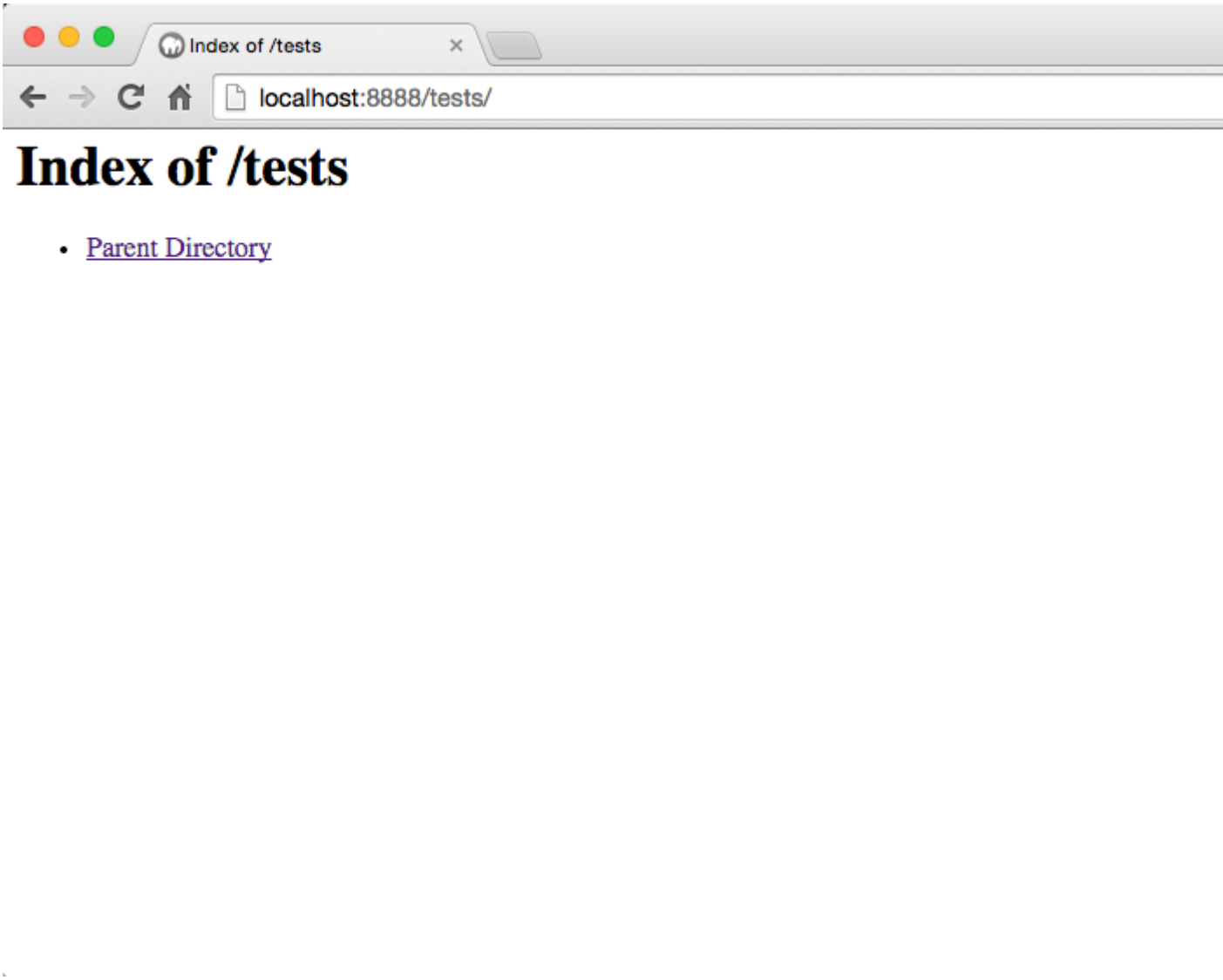
Sélectionnez ce répertoire, qui devrait être de la forme`/Users/pseudo/Sites`. Notez que ce n'est pas une obligation : vous pouvez utiliser n'importe quel autre répertoire si vous le désirez.

Validez les changements et retournez sur la fenêtre principale de MAMP. Là, cliquez sur « Ouvrir la page WebStart ». Votre navigateur (Firefox ou Safari, par exemple) devrait alors s'ouvrir et afficher une page web.

Pour vous préparer pour la suite, je vous invite à créer un dossier « tests » dans votre répertoire « Sites ». Nous placerons nos premiers fichiers PHP de test à l'intérieur.

Si le dossier « tests » a été correctement créé, vous pouvez visualiser son contenu en vous rendant à l'adresse`http://localhost:8888/tests/`.

Si tout va bien, une page vide devrait s'afficher (figure suivante).



Un dossier vide s'affiche dans le navigateur grâce à MAMP

MAMP est correctement installé et configuré. Vous êtes maintenant prêts à travailler en PHP pour le chapitre suivant !

Sous Linux : XAMPP

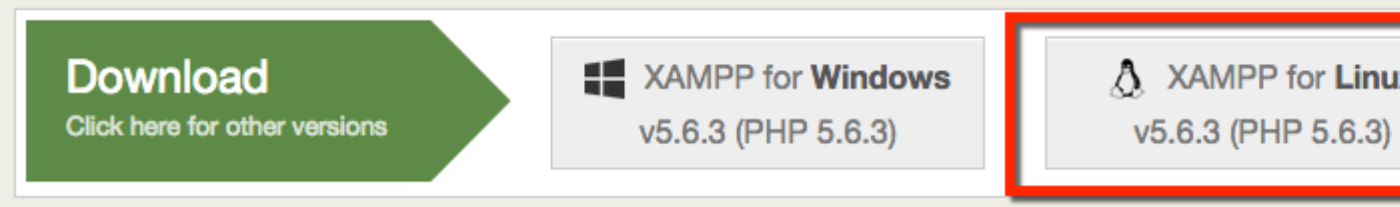
Sous Linux, il est courant d'installer Apache, PHP et MySQL séparément. Toutefois, il existe aussi des packs tout prêts comme XAMPP (X Apache MySQL Perl PHP), anciennement connu sous le nom de LAMPP.

Ce pack est plus complet que WAMP pour Windows ou MAMP pour Mac OS X. Nous n'utiliserons toutefois qu'une partie des éléments installés.

Vérifiez que vous prenez une version de XAMPP avec PHP 5.4 minimum. Les fonctionnalités de PHP changent d'une version à l'autre et ce cours est basé sur les versions de PHP >= 5.4.

Ca devrait être le cas si vous prenez la dernière version proposée sur le site.

Sur le site officiel de XAMPP, recherchez le lien XAMPP pour Linux.



Téléchargement de XAMPP pour Linux

XAMPP est aussi disponible pour Windows et Mac OS X comme vous pourrez le constater sur le site. La méthode d'installation est sensiblement différente, mais vous pouvez l'essayer si vous avez déjà testé WAMP (pour Windows) ou MAMP (pour Mac OS X) et qu'il ne vous convient pas.

Une fois le téléchargement terminé, ouvrez une console. L'installation et le lancement de XAMPP se font en effet uniquement en console (allons, allons, pas de chichis, vous n'allez pas me faire avaler que c'est la première fois que vous l'ouvrez, la console !).

Rendez-vous dans le dossier dans lequel vous avez téléchargé XAMPP. Par exemple, dans mon cas, le fichier se trouve sur le bureau :

```
cd  /Desktop
```

Vous devez passer`root`pour installer et lancer XAMPP.
`root`est le compte administrateur de la machine qui a notamment le droit d'installer des programmes. Normalement, il suffit de taper`su`et de rentrer le mot de passe`root`. Sous Ubuntu, il faudra taper`sudo su`et taper votre mot de passe habituel.

Si comme moi vous utilisez Ubuntu, tapez donc :

```
sudo su
```

Donnez les droits d'exécution au fichier que vous venez de télécharger :

```
chmod 755 xampp-linux-*-installer.run
```

Puis lancez le programme d'installation :

```
./xampp-linux-*-installer.run
```

Pensez à remplacer l'étoile dans la commande par le numéro de version du fichier téléchargé.

Et voilà ! XAMPP est maintenant installé.

Pour démarrer XAMPP (et donc Apache, PHP et MySQL), tapez la commande suivante :

```
/opt/lampp/lampp start
```

Si vous désirez plus tard arrêter XAMPP, tapez :

```
/opt/lampp/lampp stop
```

N’oubliez pas que vous devez être `root` lorsque vous démarrez ou arrêtez XAMPP.

Ce n’est pas bien compliqué, comme vous pouvez le voir !

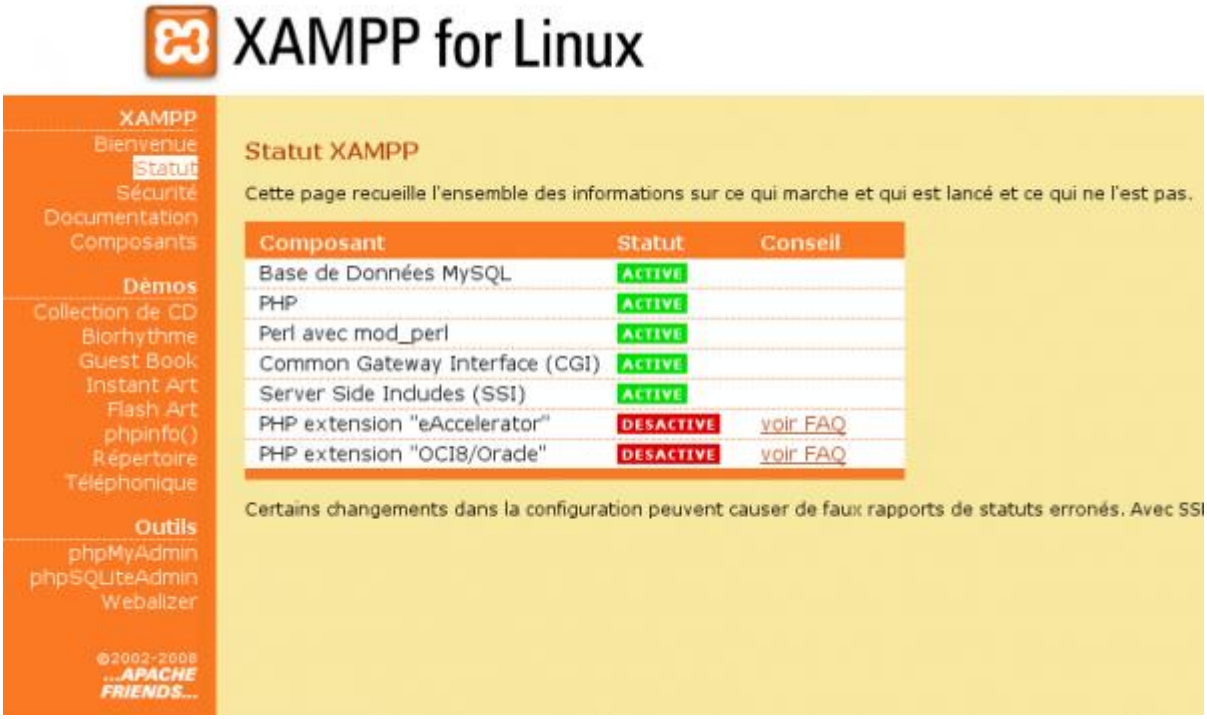
Vous pouvez maintenant tester XAMPP en ouvrant votre navigateur favori et en tapant l’adresse suivante : `http://localhost`.

Vous devriez voir la page de sélection de la langue de XAMPP. Cliquez sur « Français », comme la figure suivante vous y invite.

Choix de la langue dans XAMPP

La page principale de configuration de XAMPP s’affiche ensuite. Elle est plus complète que ses homologues WAMP et MAMP, notamment parce que XAMPP contient plus de logiciels et propose donc plus de fonctionnalités (beaucoup plus).

Vous pouvez vérifier que tout fonctionne correctement en allant dans le menu `Statut`, comme dans la figure suivante.



Statut des composants de XAMPP

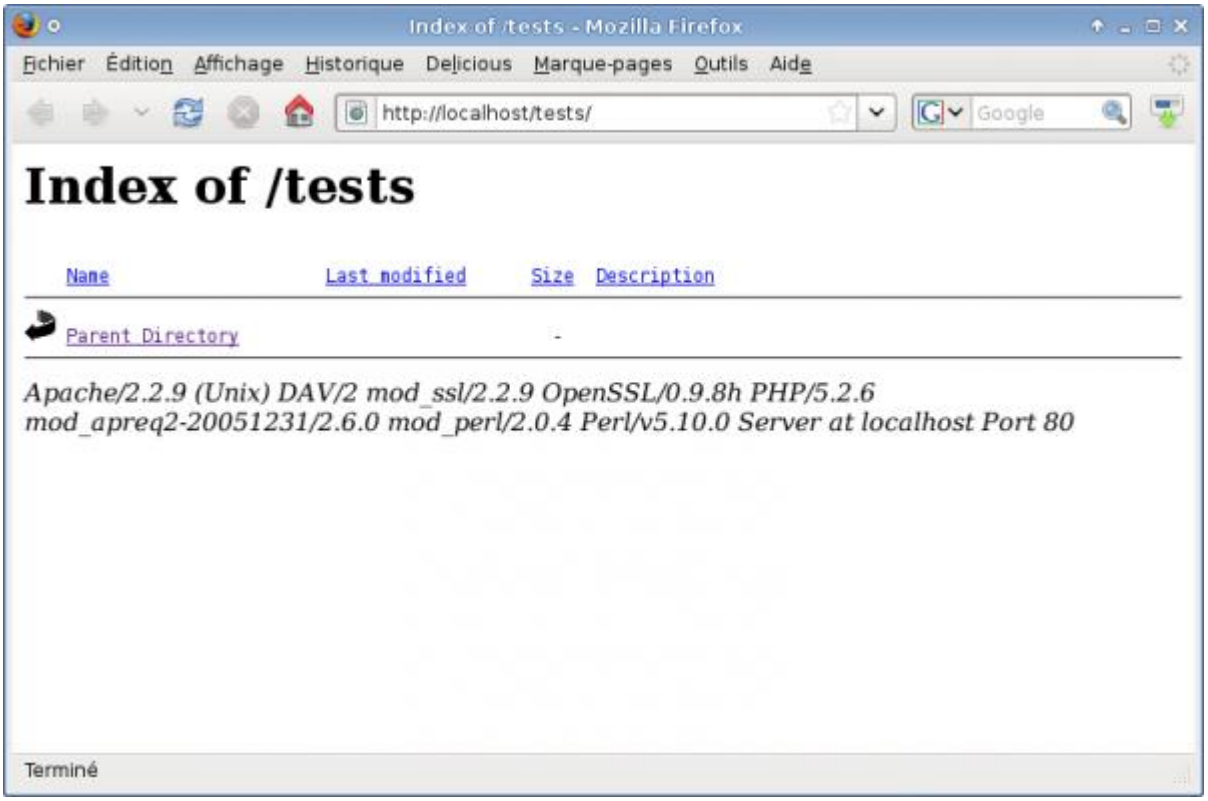
Au minimum, les modules MySQL et PHP doivent être en vert. Quant aux autres, nous ne les utiliserons pas, donc peu importe.

Les fichiers PHP devront être placés dans le répertoire `/opt/lampp/htdocs`. Vous pouvez y créer un sous-répertoire `tests` pour vos premiers tests.

```
cd /opt/lampp/htdocs
mkdir tests
```

Une fois le dossier créé, vous pouvez y accéder depuis votre navigateur à l'adresse suivante : `http://localhost/tests`.

Vous devriez voir une page similaire à la figure suivante.



Le dossier tests est actuellement vide dans XAMPP

Vous êtes prêts à travailler en PHP !

Utiliser un bon éditeur de fichiers

Comme vous devez déjà le savoir, pour éditer le code d'une page web vous avez plusieurs solutions :

- utiliser un éditeur de texte tout simple que vous avez déjà, comme **Bloc-notes**. Pour l'ouvrir, faites `Démarrer / Programmes / Accessoires / Bloc-notes`. Ce logiciel suffit normalement à écrire des pages web en HTML et même en PHP, mais…

- le mieux reste d'utiliser un **logiciel spécialisé** qui colore votre code (très pratique) et

qui numérote vos lignes (très pratique aussi). Il existe des centaines et des centaines de logiciels gratuits faits pour les développeurs comme vous. Je vais vous en présenter ici deux : un gratuit (Sublime Text) et un payant (PHPStorm).

Je vous propose donc d'installer un logiciel qui va vous permettre d'éditer vos fichiers source de manière efficace. Vous en avez probablement déjà installé un si vous avez appris à programmer en HTML / CSS, mais comme on n'est jamais trop prudent, je vais rapidement vous en présenter quelques-uns en fonction de votre système d'exploitation.

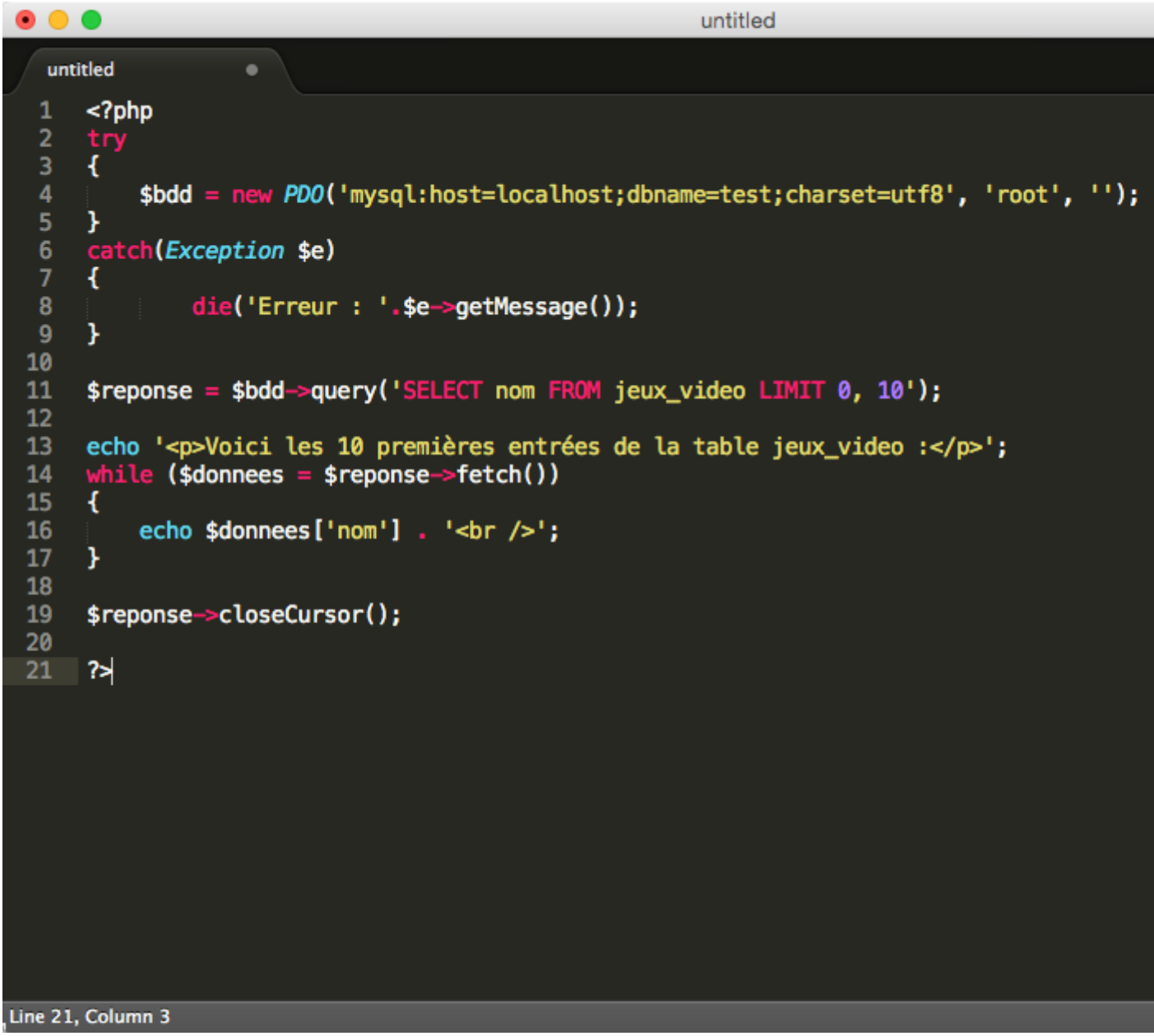
Voici le code source HTML que nous allons utiliser pour commencer en terrain connu. Copiez-collez ce code dans l'éditeur de texte que je vais vous faire installer :

```
<!DOCTYPE html>

<html>

<head>

<meta charset=« utf-8 » />

<title>Ceci est une page HTML de test</title>

</head>

<body>

<h2>Page de test</h2>

<p>

Cette page contient <strong>uniquement</strong> du code HTML.<br />

Voici quelques petits tests :

</p>

<ul>

<li style=« color: blue; »>Texte en bleu</li>

<li style=« color: red; »>Texte en rouge</li>

<li style=« color: green; »>Texte en vert</li>

</ul>

</body>

</html>
```

Il n'y a pas de PHP pour l'instant afin de commencer en douceur. Nous allons simplement essayer d'enregistrer un fichier HTML avec ce code pour nous échauffer.

Sublime Text (gratuit)

Que vous soyez sous Windows, Mac ou Linux, je vous recommande de commencer avec l'éditeur Sublime Text qui est suffisamment léger et simple pour nous qui débutons :



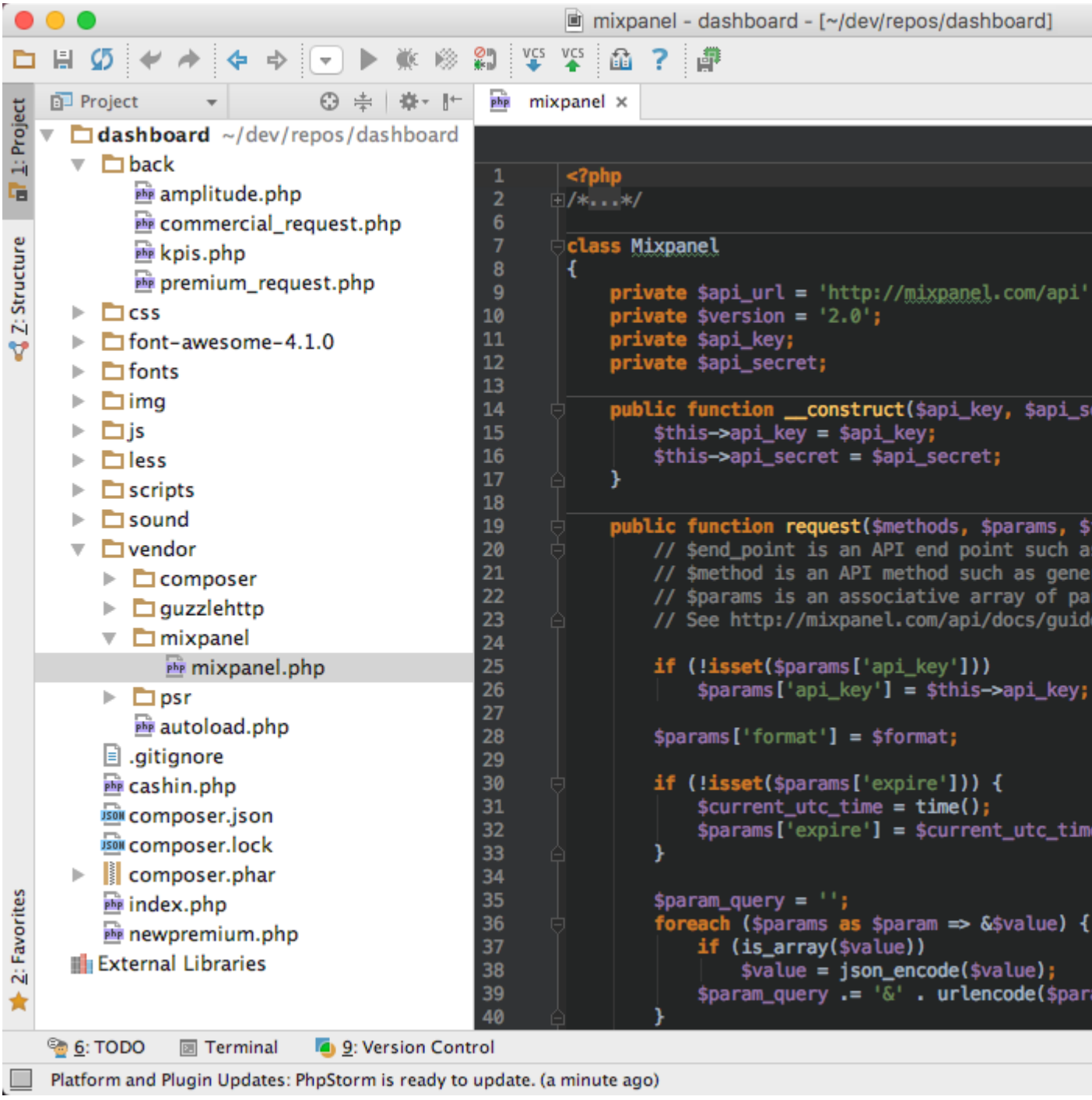
L'éditeur de texte Sublime Text

Ne vous fiez pas à son apparente simplicité : Sublime Text est en effet rapide et simple à la base, mais il est possible d'étendre ses fonctionnalités avec d'innombrables plugins !

Sublime Text est un très bon éditeur, utilisé par de nombreux développeurs (y compris des pros). Il voit en revanche ses limites sur de gros projets, où certains pros lui préfèrent PHPStorm.

PHPStorm (payant)

PHPStorm ressemble un peu plus à une « machine de guerre » que Sublime Text. Et pour cause : c'est un IDE, un environnement de travail de développeur. Il est utilisé par de nombreux développeurs PHP professionnels de ma connaissance.



L'éditeur PHPStorm, utilisé par de nombreux professionnels

PHPStorm est plus « costaud » que Sublime Text. Il met plus de temps à charger, il peut avoir de nombreuses fonctionnalités avancées grâce à ses plugins… Par ailleurs, PHPStorm est **payant**.

Vous ne commencerez peut-être pas de suite avec PHPStorm, mais gardez-le sur votre radar car c'est un outil très utilisé que vous essaierez sûrement un jour.

En résumé :

- Pour créer des sites web dynamiques, nous devons installer des outils qui

transformeront notre ordinateur en serveur afin de pouvoir tester notre site.

- Les principaux outils dont nous avons besoin sont :
  - Apache : le serveur web ;
  - PHP : le programme qui permet au serveur web d'exécuter des pages PHP ;
  - MySQL : le logiciel de gestion de bases de données.
- Bien qu'il soit possible d'installer ces outils séparément, il est plus simple pour nous

d'installer un paquetage tout prêt : WAMP sous Windows, MAMP sous Mac OS X ou XAMPP sous Linux.

- Il est conseillé d'utiliser un éditeur de texte qui colore le code source comme Sublime

Text pour programmer convenablement en PHP.
Pour les personnes plus expérimentées qui travaillent sur de gros projets, je recommande PHPStorm.

IV- Écrivez votre premier script :

Dans le premier chapitre, nous avons découvert le principe de fonctionnement du PHP. Ici, nous allons passer au concret et réaliser notre toute première page web en PHP.

Ne vous attendez pas à un résultat extraordinaire, mais cela va nous permettre de prendre nos marques. Vous allez en particulier comprendre comment on sépare le code HTML classique du code PHP.

Vous êtes prêts ? Allons-y !

Les balises PHP

Vous savez donc que le code source d'une page HTML est constitué de **balises** (aussi appelées **tags**). Par exemple,`<ul>`est une balise.

Le code PHP vient s'insérer au milieu du code HTML. On va progressivement placer dans nos pages web des morceaux de code PHP à l'intérieur du HTML. Ces bouts de code PHP

seront les parties dynamiques de la page, c'est-à-dire les parties qui peuvent changer toutes seules (c'est pour cela qu'on dit qu'elles sont **dynamiques**).

La figure suivante illustre cela.

Insertion de code PHP

Comme vous pouvez le voir, on retrouve le code HTML que l'on connaît bien… et on insère en plus des données dynamiques au milieu. Ici, par exemple, c'est le pseudonyme : il change en fonction du visiteur. La partie surlignée peut donc changer selon les visiteurs.

OpenClassrooms fait la même chose pour ses membres inscrits. Votre pseudonyme est affiché en haut des pages lorsque vous êtes connectés.

La forme d'une balise PHP

Si je vous parle de cela, ce n'est pas par hasard. Pour utiliser du PHP, on va devoir introduire une nouvelle balise… et celle-ci est un peu spéciale. Elle commence par`<?php`et se termine par`?>`; c'est à l'intérieur que l'on mettra du code PHP, ce que je vais vous apprendre tout au long de ce cours.

Voici une balise PHP vide :

```
<?php ?>
```

À l'intérieur, on écrira donc du code source PHP :

```
<?php /* Le code PHP se met ici */ ?>
```

On peut sans problème écrire la balise PHP sur plusieurs lignes. En fait, c'est même indispensable car la plupart du temps le code PHP fera plusieurs lignes. Cela donnera quelque chose comme :

```
<?php
```

```
/* Le code PHP se met ici

Et ici

Et encore ici */

?>
```

Il existe d'autres balises pour utiliser du PHP, par exemple`<? ?>`,`<% %>`, etc. Ne soyez donc pas étonnés si vous en voyez. Néanmoins,`<?php ?>`est la forme la plus correcte, vous apprendrez donc à vous servir de cette balise et non pas des autres.

Insérer une balise PHP au milieu du code HTML

La balise PHP que nous venons de découvrir s'insère au milieu du code HTML comme je vous l'ai dit plus tôt. Pour reprendre l'exemple que l'on a vu au chapitre précédent :

```
<!DOCTYPE html>

<html>

<head>

<title>Ceci est une page de test avec des balises PHP</title>

<meta charset=« utf-8 » />

</head>

<body>

<h2>Page de test</h2>

<p>

Cette page contient du code HTML avec des balises PHP.<br />

<?php /* Insérer du code PHP ici */ ?>

Voici quelques petits tests :

</p>

<ul>

<li style=« color: blue; »>Texte en bleu</li>

<li style=« color: red; »>Texte en rouge</li>

<li style=« color: green; »>Texte en vert</li>

</ul>
```

```
<?php

/* Encore du PHP

Toujours du PHP */

?>

</body>

</html>
```

Peut-on placer une balise PHP n'importe où dans le code ?

Oui ! Vraiment n'importe où. Pas seulement dans le corps de la page d'ailleurs : vous pouvez placer une balise PHP dans l'en-tête de la page (regardez la ligne 4 de l'exemple ci-dessous).

```
<!DOCTYPE html>

<html>

<head>

<title>Ceci est une page de test <?php /* Code PHP */ ?></title>

<meta charset=« utf-8 » />

</head>
```

Plus fort encore, vous pouvez même insérer une balise PHP au milieu d'une balise HTML, comme le montre la ligne 5 de l'exemple ci-dessous (bon, ce n'est pas très joli, je vous l'accorde) :

```
<!DOCTYPE html>

<html>

<head>

<title>Ceci est une page de test</title>

<meta <?php /* Code PHP */ ?> charset=« utf-8 » />

</head>
```

Comment ça fonctionne ? À quoi ça peut servir ?

Il faut se rappeler que PHP génère du code HTML. Nous allons mieux comprendre le fonctionnement en apprenant à afficher du texte en PHP.

Afficher du texte

Bon, tout ça c'est bien beau, mais il serait temps de commencer à écrire du code PHP, non ? Grande nouvelle : c'est maintenant que vous allez apprendre votre première instruction en PHP.

Ne vous attendez pas à quelque chose d'extraordinaire, votre PC ne va pas se mettre à danser

la samba tout seul.

Vous allez cependant un peu mieux comprendre comment le PHP fonctionne, c'est-à-dire comment il génère du code HTML. Il est indispensable de bien comprendre cela, soyez donc attentifs !

L'instruction`echo`

Le PHP est un langage de programmation, ce qui n'était pas le cas du HTML (on parle plutôt de langage de description, car il permet de décrire une page web). Si vous avez déjà programmé dans d'autres langages comme le C ou le Java, cela ne devrait pas vous surprendre. Néanmoins, dans ce cours, nous partons de Zéro donc je vais supposer que vous n'avez jamais fait de programmation auparavant.

Tout langage de programmation contient ce qu'on appelle des **instructions**. On en écrit une par ligne en général, et elles se terminent toutes par un point-virgule. Une instruction commande à l'ordinateur d'effectuer une action précise.

Ici, la première instruction que nous allons découvrir permet d'insérer du texte dans la page web. Il s'agit de l'instruction`echo`, la plus simple et la plus basique de toutes les instructions que vous devez connaître.

Voici un exemple d'utilisation de cette instruction :

```
<?php echo « Ceci est du texte »; ?>
```

Comme vous le voyez, à l'intérieur de la balise PHP on écrit l'instruction`echo`suivie du texte à afficher entre guillemets. Les guillemets permettent de délimiter le début et la fin du texte, ce qui aide l'ordinateur à se repérer. Enfin, l'instruction se termine par un point-virgule comme je vous l'avais annoncé, ce qui signifie **Fin de l'instruction**.

Notez qu'il existe une instruction identique à`echo`appelée`print`, qui fait la même chose. Cependant,`echo`est plus couramment utilisée.

Il faut savoir qu'on a aussi le droit de demander d'afficher des balises. Par exemple, le code suivant fonctionne :

```
<?php echo « Ceci est du <strong>texte</strong> »; ?>
```

Le mot « texte » sera affiché en gras grâce à la présence des balises`<strong>`et`</strong>`.

Comment faire pour afficher un guillemet ?

Bonne question. Si vous mettez un guillemet, ça veut dire pour l'ordinateur que le texte à afficher s'arrête là. Vous risquez au mieux de faire planter votre beau code et d'avoir une terrible « Parse error ».

La solution consiste à faire précéder le guillemet d'un antislash\:

```
<?php echo « Cette ligne a été écrite \ »uniquement\ » en PHP. »; ?>
```

Vous savez que le code PHP s'insère au milieu du code HTML. Alors allons-y, prenons une page basique en HTML et plaçons-y du code PHP (ligne 12) :

```
<!DOCTYPE html>

<html>

<head>

<title>Notre première instruction : echo</title>

<meta charset=« utf-8 » />

</head>

<body>

<h2>Affichage de texte avec PHP</h2>

<p>

Cette ligne a été écrite entièrement en HTML.<br />

<?php echo « Celle-ci a été écrite entièrement en PHP. »; ?>

</p>

</body>

</html>
```

Je vous propose de copier-coller ce code source dans votre éditeur de texte et d'enregistrer la page. Nous allons l'essayer et voir ce qu'elle produit comme résultat.

Mais au fait, vous rappelez-vous comment vous devez enregistrer votre page PHP ?

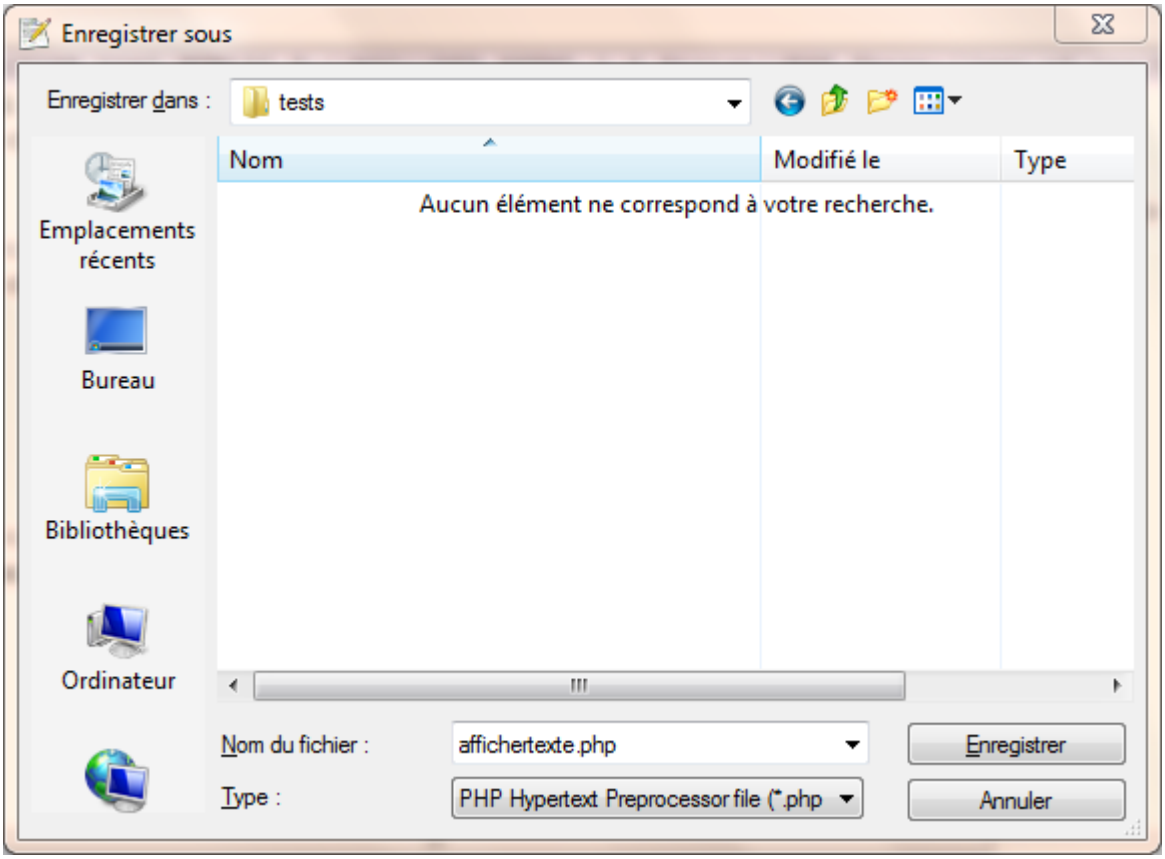
Enregistrer une page PHP

Je vous ai expliqué comment faire dans le chapitre précédent mais un petit rappel ne peut pas faire de mal.

Enregistrez la page avec l'extension `.php`, par exemple `affichertexte.php`, dans le dossier `tests` que je vous ai fait créer. Il doit se trouver dans `C:\wamp\www\tests` sous Windows.

L'essentiel, quel que soit votre système d'exploitation, est que le fichier soit enregistré dans le dossier `www` (ou un de ses sous-dossiers) sinon le fichier PHP ne pourra pas s'exécuter !

Si vous utilisez Notepad++, sélectionnez `PHP Hypertext Preprocessor file (*.php)` dans la fenêtre pour enregistrer, comme le montre la figure suivante.



Sauvegarde d'une page PHP

Une fois la page enregistrée, il faut maintenant la tester.

Tester la page PHP

Pour tester votre page PHP, cela dépend de votre système d'exploitation mais la manoeuvre est dans les grandes lignes la même.

Sous Windows, démarrez WAMP si ce n'est pas déjà fait. Allez dans le menu `Localhost`, la page d'accueil s'ouvre. Là, si vous avez bien créé le dossier `tests` dans le répertoire `www` comme indiqué au chapitre précédent, vous devriez voir un lien vers le dossier `tests`. Cliquez dessus (nous avons déjà fait cela dans le chapitre précédent).

Une page web s'ouvre indiquant tous les fichiers qui se trouvent dans le dossier`tests`. Vous devriez avoir le fichier`affichertexte.php`. Cliquez dessus : votre ordinateur génère alors le code PHP puis ouvre la page. Vous avez le résultat devant vos yeux.

Le même résultat peut être obtenu dans votre navigateur en allant directement à l'adresse`http://localhost/tests/affichertexte.php`. La méthode devrait être quasiment la même que vous soyez sous Windows, Mac OS X ou Linux.

Alors, que voyez-vous ?
Je pense que vous êtes étonnés et surpris de ce que je vous ai fait faire : ça a l'air d'être inutile, et ce n'est pas tout à fait faux. Le code PHP a « écrit » une ligne à l'écran, tout simplement.

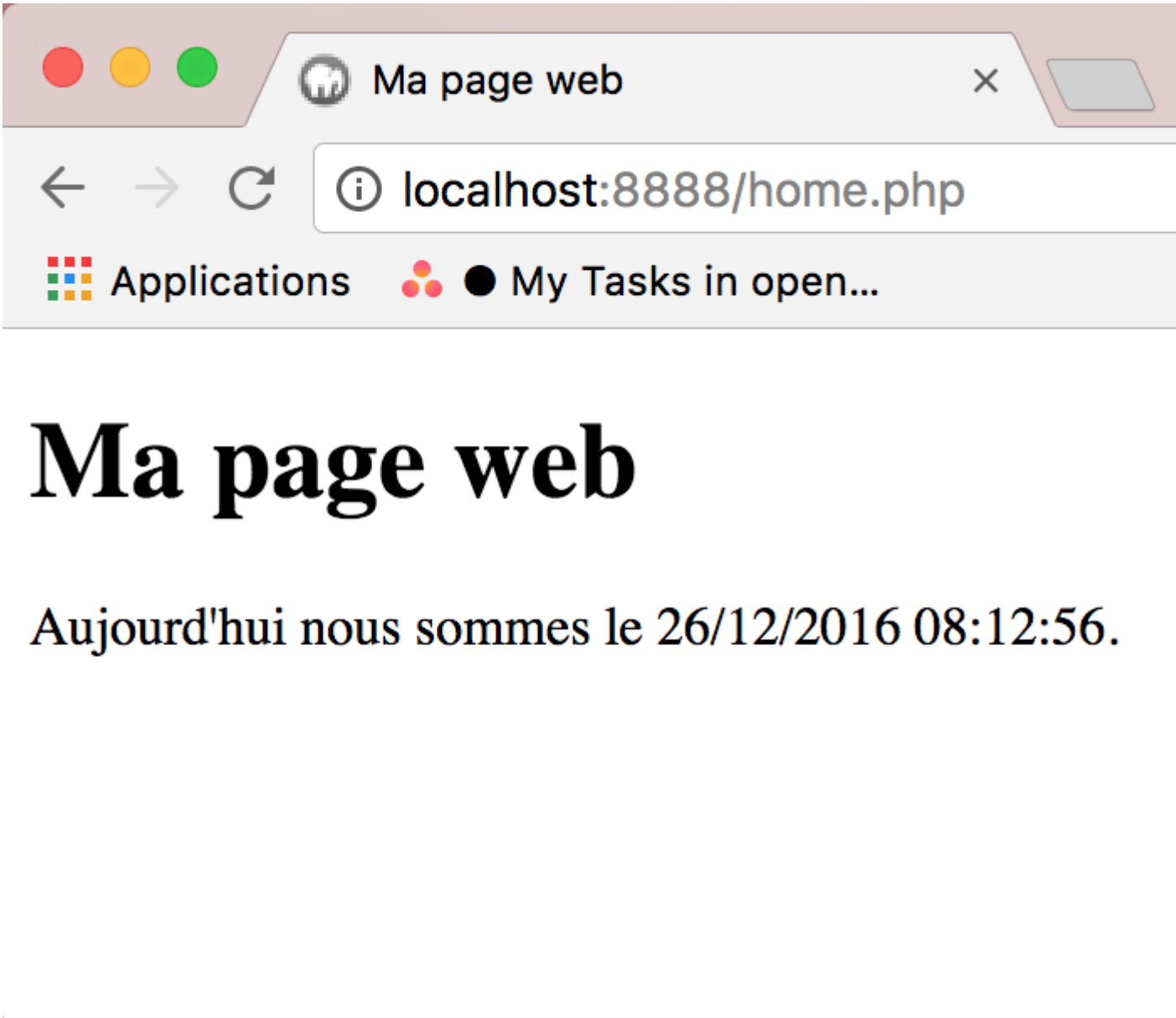
Mais euh, c'est pas plus simple de l'écrire en HTML ?

Si !
Mais vous verrez bientôt l'intérêt de cette fonction. Pour le moment, on constate juste que ça écrit du texte.

En attendant, pour vous amuser et comprendre la force de PHP, essayez juste le code suivant (qu'on expliquera plus tard dans le cours) :

```
<!DOCTYPE html>

<html>

<head>

<meta charset=« utf-8 » />

<title>Ma page web</title>

</head>

<body>

<h1>Ma page web</h1>

<p>Aujourd'hui nous sommes le <?php echo date('d/m/Y h:i:s'); ?>.</p>

</body>

</html>
```

Testez ce code et admirez



Affichage dynamique de la date et l'heure en PHP

La date et l'heure s'affichent automatiquement sur la page !

Maintenant actualisez la page : l'heure s'est mise à jour toute seule !

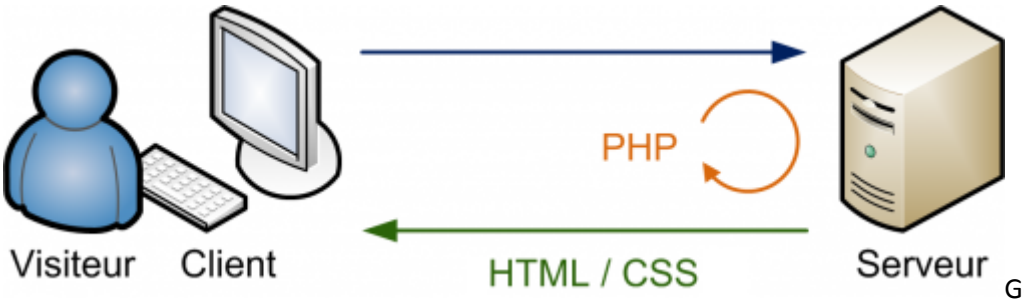
Regardez le code source de la page générée dans le navigateur : vous verrez qu'il n'y a pas de code PHP. L'heure a directement été envoyée dans le code HTML. Pourquoi ?

Comment PHP génère du code HTML

L'instruction`echo`demande à PHP d'insérer à cet endroit le texte que vous demandez. Si on traduit l'instruction que je vous ai présentée plus haut en français, ça donne : Insérer le texte : **« Celle-ci a été écrite entièrement en PHP.** ».

Il ne faut jamais oublier le point-virgule à la fin d'une instruction. Si jamais ça arrive, vous aurez le message d'erreur : « Parse Error ».
Notez que ça plante uniquement si votre code PHP fait plus d'une ligne (ça sera tout le temps le cas). Prenez donc l'habitude de toujours mettre un « ; » à la fin des instructions.

Je vous ai expliqué dans le tout premier chapitre que le PHP générait du code HTML et renvoyait au visiteur uniquement du code HTML (accompagné éventuellement de sa feuille de style CSS), comme le montre la figure suivante.

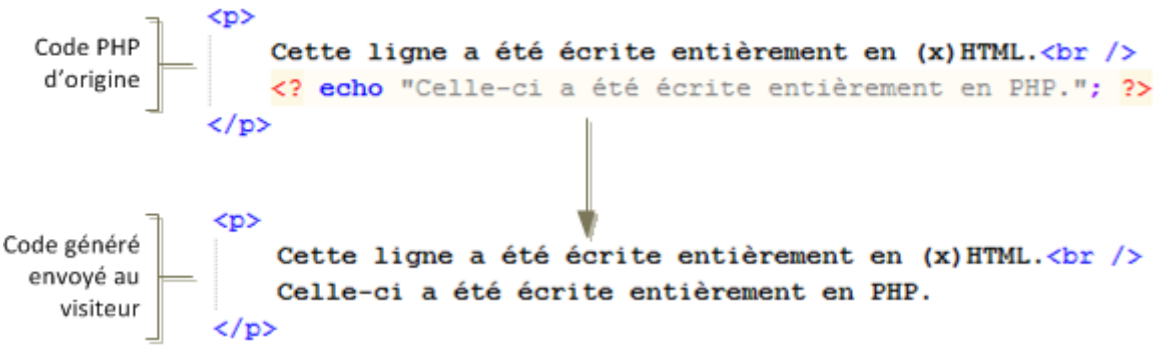


Génération de HTML par PHP

Sur la figure suivante, vous découvrez concrètement ce qu'il se passe avec notre code source.

Le code PHP est exécuté en premier et l'ordinateur fait ce qu'on lui demande. Ici on lui a dit « Affiche ce texte ici ».

Une fois toutes les instructions PHP exécutées (ici c'était simple, il n'y en avait qu'une !), la page qui sort est une page qui ne contient que du HTML ! C'est cette page de « résultat » qui est envoyée au visiteur, car celui-ci ne sait lire que le HTML.



Génération de HTML avec echo

Rappelez-vous, seul le serveur peut exécuter du PHP. Le PHP n'est**jamais** envoyé au visiteur. Pour que nous puissions exécuter du PHP sur notre ordinateur (afin de faire nos tests), nous avons dû le transformer en mini-serveur en installant un programme tel que WAMP.

Les commentaires

Bon, mine de rien je viens de vous apprendre pas mal de choses d'un coup, ça doit vous faire un choc. D'accord ce n'était pas extraordinaire, mais vous n'allez pas tarder à comprendre toute la subtilité de la chose.

Avant de terminer ce chapitre, je tiens à vous parler de quelque chose qui à mes yeux a une très grande importance en PHP, comme dans tout langage de programmation : les commentaires.

Un **commentaire** est un texte que vous mettez pour vous dans le code PHP. Ce texte est ignoré, c'est-à-dire qu'il disparaît complètement lors de la génération de la page. Il n'y a que vous qui voyez ce texte.

Mais alors, à quoi sert un commentaire ?

C'est pour vous. Cela permet de vous y retrouver dans votre code PHP, parce que si vous n'y touchez pas pendant des semaines et que vous y revenez, vous risquez d'être un peu perdus. Vous pouvez écrire tout et n'importe quoi, le tout est de s'en servir à bon escient.

Il existe deux types de commentaires :

- les commentaires monolignes ;
- les commentaires multilignes.

Tout dépend de la longueur de votre commentaire. Je vais vous présenter les deux.

Les commentaires monolignes

Pour indiquer que vous écrivez un commentaire sur une seule ligne, vous devez taper deux slashs : «//». Tapez ensuite votre commentaire.
Un exemple ?

```
<?php

echo « J'habite en Chine. »; // Cette ligne indique où j'habite

// La ligne suivante indique mon âge

echo « J'ai 92 ans. »;

?>
```

Je vous ai mis deux commentaires à des endroits différents :

- le premier est à la fin d'une ligne ;
- le second est sur toute une ligne.

À vous de voir où vous placez vos commentaires : si vous commentez une ligne précise, mieux vaut mettre le commentaire à la fin de cette ligne.

Les commentaires multilignes

Ce sont les plus pratiques si vous pensez écrire un commentaire sur plusieurs lignes, mais on peut aussi s'en servir pour écrire des commentaires d'une seule ligne. Il faut commencer par écrire `/*` puis refermer par `*/` :

```
<?php

/* La ligne suivante indique mon âge

Si vous ne me croyez pas…

… vous avez raison ;o) */

echo « J'ai 92 ans. »;

?>
```

Ici, les commentaires n'ont pas grande utilité, mais vous verrez de quelle façon je les utilise dans les prochains chapitres pour vous décrire le code PHP.

En résumé :

- Les pages web contenant du PHP ont l'extension `.php`.
- Une page PHP est en fait une simple page HTML qui contient des instructions en langage PHP.
- Les instructions PHP sont placées dans une balise `<?php ?>`.
- Pour afficher du texte en PHP, on utilise l'instruction `echo`.
- Il est possible d'ajouter des commentaires en PHP pour décrire le fonctionnement du code. On utilise pour cela les symboles `//` ou `/* */`.

V- Configuration de PHP et visualisation des erreurs :

Avant d'aller plus loin, il est important de faire un petit stop par ce que vous allez sans doute beaucoup rencontrer (et ce n'est absolument pas grave !) : les erreurs.

En effet, lorsque un script PHP plante, le comportement par défaut de PHP est de n'afficher qu'une page blanche (une page de navigateur sans contenu).

Pour faciliter notre vie de développeur, il va falloir faire en sorte que les erreurs PHP s'affichent. Sinon, nous aurons de grosses difficultés par la suite pour comprendre pourquoi nos pages ne marchent pas.

Nous allons donc changer la configuration de PHP.

Configurer PHP pour afficher les erreurs

Et oui, PHP est configurable !

Si les erreurs s'affichent déjà bien dans votre navigateur, inutile de faire les manipulations qui vont suivre !

Par défaut, PHP n'affiche pas les erreurs pour éviter de donner trop d'indications aux utilisateurs pour des raisons de sécurité (un mantra à vous répéter : « moins l'utilisateur en sait sur mon application, mieux mon application se portera ! »).

La configuration de PHP se fait dans un fichier appelé « php.ini ». Encore faut-il savoir où il se trouve !

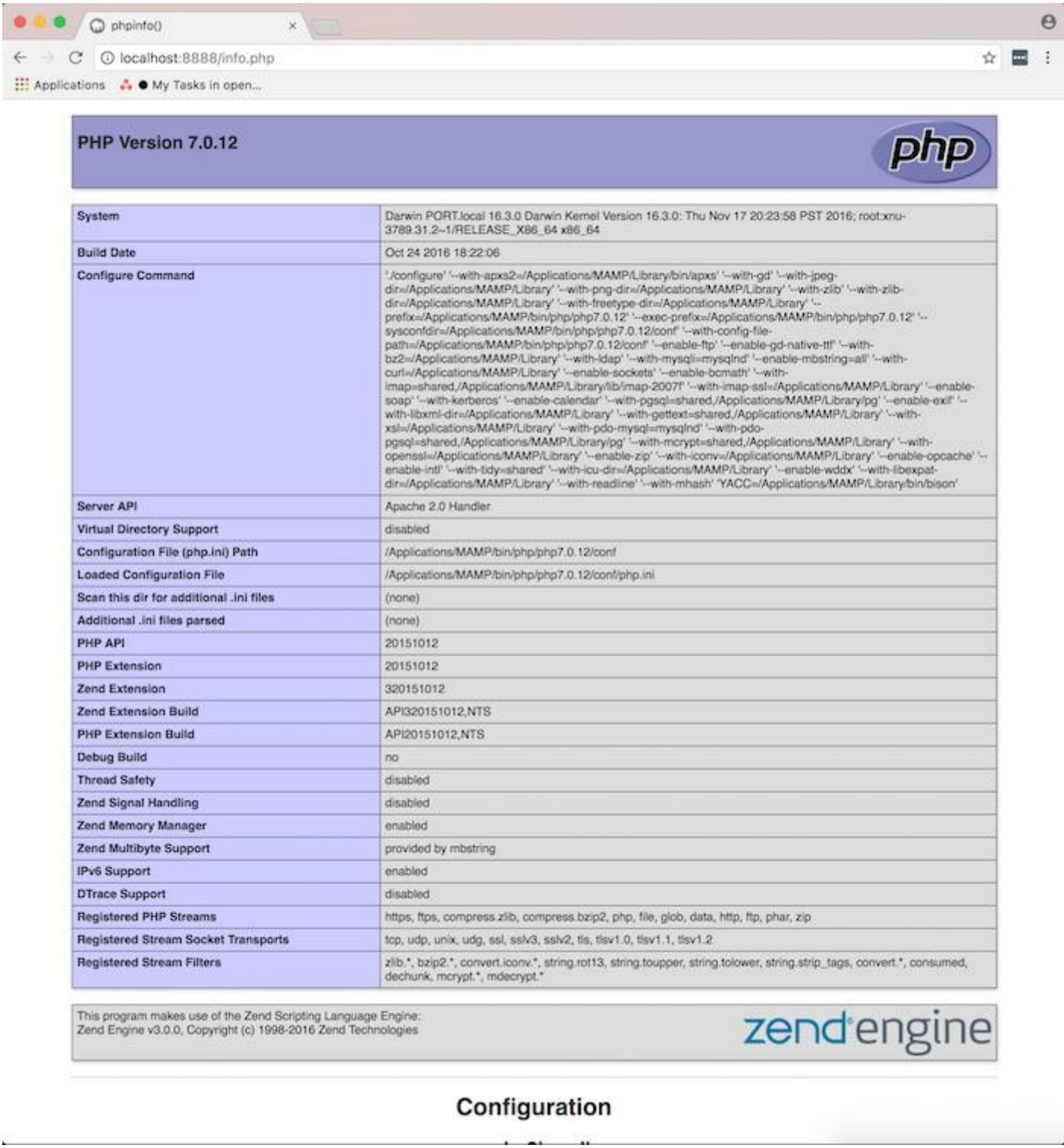
Localiser le fichier de configuration PHP du serveur web

Pour connaitre l'ensemble des informations relatives au PHP utilisé par le serveur web, il existe une commande PHP `phpinfo()` (on parle de fonction, on y reviendra). Nous allons l'utiliser pour localiser le fichier de configuration pour que nous puissions le modifier.

Je vous invite donc à créer un fichier PHP avec simplement le code qui suit :

```
<?php

phpinfo();
```

Enregistrez le sous le nom `info.php` dans le dossier qui est lu par votre serveur web. Et enfin, affichez la page. Vous devriez obtenir le résultat suivant :



Résultat du script contenant l'instruction phpinfo()

Cette page contient toute une flopée d'informations dont, la version de PHP utilisée (pour moi il s'agit de PHP 7.0.12) le type de serveur web (ici Apache)… Et la localisation du (ou des fichiers) de configuration pour PHP.

Retrouvez la ligne « Loaded Configuration File » (ce qui signifie « fichier de configuration chargé » en anglais), et regardez la valeur. Dans mon cas, on peut voir ceci :

Chemin du fichier de configuration de PHP chargé par le serveur web

Je vais donc ouvrir ce fichier et le modifier.

Il faut s'assurer que les clés de configuration `error_reporting` et `display_errors` ont respectivement les valeurs `E_ALL` & `on`.

Allons-y étape par étape :

1. Effectuez une recherche dans le fichier avec le terme `error_reporting`. S'il n'y a pas écrit `error_reporting = E_ALL` , remplacez la par la bonne valeur.
2. Ensuite, effectuez une nouvelle recherche dans le fichier avec le terme `display_errors` . S'il n'y a pas écrit `display_errors = On` , remplacez la par la bonne valeur.
3. Enregistrez le fichier.
4. Relancez le serveur pour qu'il prenne en compte vos modifications. Il suffit de relancer WAMP ou MAMP par exemple.

Dans le fichier de configuration, le point-virgule (`;`) en début de ligne signifie que tout ce qui suit est un commentaire. Si l'une de ces lignes (ou les deux) sont commentées, il suffit de retirer le point-virgule en début de ligne.

Prenez soin de faire bien attention à ce que ces lignes de configuration n'existent qu'une seule fois dans le fichier : en effet, ne créez pas ces lignes si elles existaient déjà  .

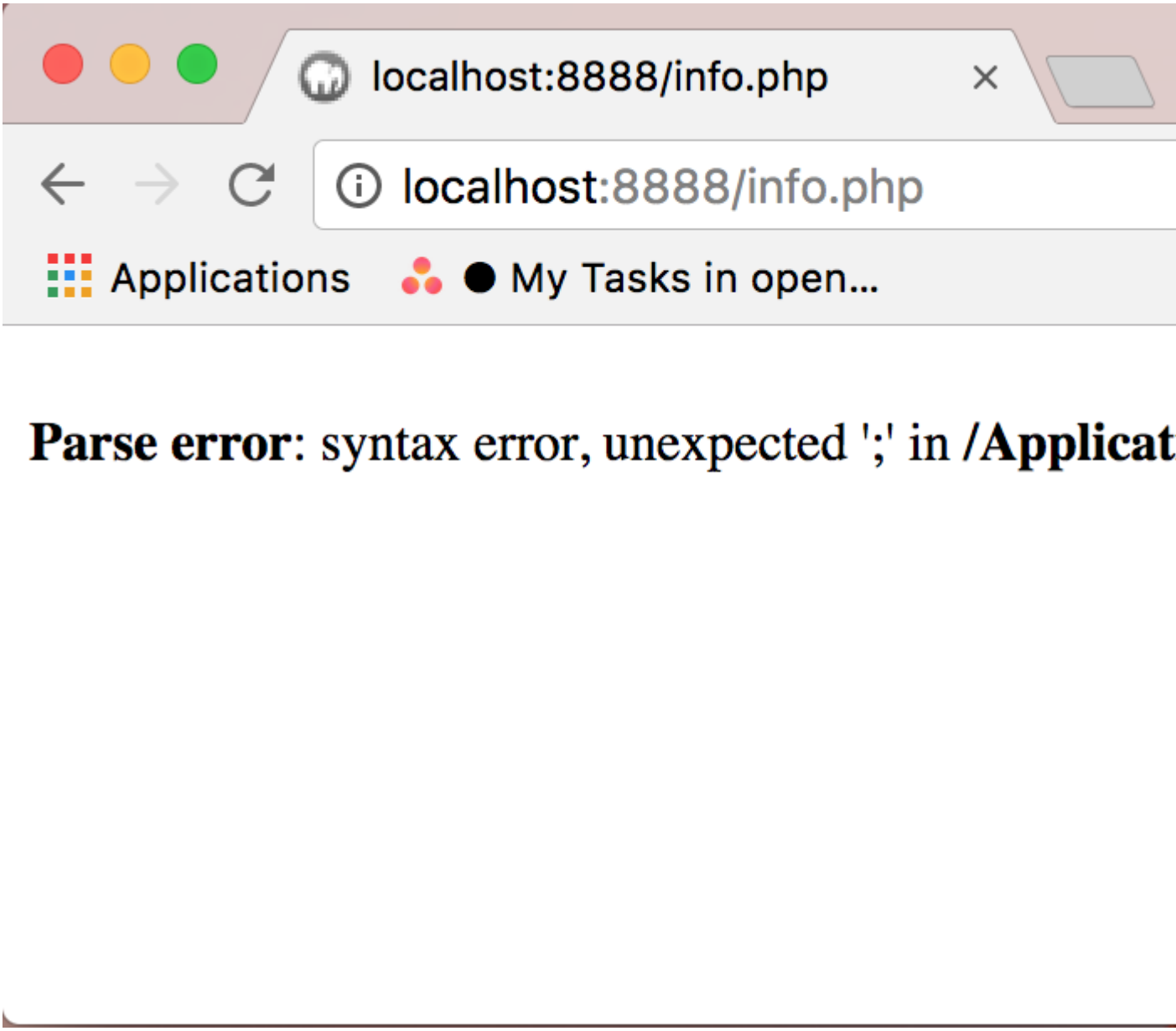
Tester l'affichage des erreurs

Nous allons maintenant créer une erreur dans un script PHP pour nous assurer que l'erreur s'affiche dans le navigateur : dans le script que nous avons créé pour afficher les informations relatives à PHP pour le serveur web (nous l'avions appelé `info.php`), retirez une parenthèse, puis enregistrez le fichier. Ca devrait donner ceci :

```
<?php

phpinfo(;
```

Oui je sais, il manque une parenthèse, c'est une erreur, on le fait exprès.

Maintenant, affichez la page à l'aide votre navigateur web.



Affichage des erreurs dans le navigateur web

Et voilà ! Si vous voyez bien cette erreur, c'est que PHP est configuré pour afficher le détail des erreurs. Ouf ! Ca nous fera gagner beaucoup de temps pour comprendre nos problèmes par la suite.

Evaluation :

- Installer les outils propres à PHP (serveur web, logiciel de gestion de base de données)

Description

Qu’avez-vous retenu de cette première partie ?

Question 1

Qu’est-ce qui est indispensable au bon fonctionnement d’un site web écrit avec PHP afin qu’il s’affiche correctement dans un navigateur ?

- Un serveur web (Apache, Nginx…)
- Un IDE
- Un terminal (ligne de commande)

Question 2

PHP permet de créer des sites…

- Statiques
- Dynamiques

Question 3

Sélectionnez des technologies concurrentes de PHP

Attention, plusieurs réponses sont possibles.

- [ ] MySQL
- [ ] Django
- [ ] HTML
- [ ] Ruby on Rails

Question 4

Sélectionnez des éditeurs de texte dans lesquels on peut écrire du code PHP :

Attention, plusieurs réponses sont possibles.

- [ ] Apache
- [ ] Sublime Text

- [ ] PHPStorm
- [ ] Google Chrome

Question 5

A quoi ressemble une balise PHP ?

- ( ) `<?/ ?>`
- ( ) `<php ?>`
- ( ) `<?php ?>`

Question 6

Pourquoi faut-il configurer PHP pour afficher des erreurs sur sa machine ?

- ( ) Cela nous permet de comprendre nos erreurs et de les résoudre plus vite
- ( ) Cela accélère le temps d'affichage d'une page
- ( ) Cela nous permet d'éviter un bug de PHP qui empêche les pages de s'afficher correctement