front-end developer_ /webdesigner

CMS : Wordpress

(Base & Theming)

Bruxelles Formation | CEPEGRA | Olivier Céréssia | 2018

webdesigner.cepegra-labs.be

PRÉAMBULE

Les CMS (Content Management System) sont des entités qui existent depuis un bon bout de temps.

Originellement, ils avaient été conçus pour créer des blogs et ils portent donc encore (pour les plus anciens d'entre eux) l'héritage d'un passé orienté « blogging ».

Mais voilà, depuis quelques temps maintenant, le monde du Web s'est rendu compte qu'il serait bon de détourner le système des blogs pour l'appliquer à des sites pour les rendre dynamiques.

Car, avec les bases que vous possédez à l'heure actuelle, vous êtes capables de répondre à toutes les demandes qui ne concernant pas des sites dynamiques.

Avec les CMS, vous allez pouvoir étendre votre champ d'action et produire vous-même, sans avoir forcément besoin de beaucoup de connaissance en développement, des sites entièrement dynamiques.

Ce chapitre de votre formation va dans un premier temps resituer ce qu'on entend par CMS puis par dynamique, ensuite nous verrons de quoi nous avons besoin en termes de connaissances mais aussi de logiciels ou de technologies pour mettre un CMS en place. Finalement, nous installerons un CMS très utilisé à l'heure actuelle (Wordpress), nous le découvrirons ensemble en profondeur puisqu'on va voir comment le paramétrer, choisir et appliquer des thèmes existants, des extensions, ajouter des contenus et, cerise sur le gâteau, vous verrez comment rendre dynamique un site statique que vous avez réalisé en HTML/CSS.

Une fois que nous aurons vu ensemble Wordpress, nous découvrirons quels sont les autres CMS du marché qui ont le vent en poupe : Drupal, PhpBB, Magento, Prestashop, … Nous verrons aussi si de nouveaux CMS essayent de prendre une belle place sur le marché et avec quels arguments ils espèrent renverser les géants actuels.

TABLE DES MATIÈRES

CMS : DÉFINITION ET UTILITÉ

• Définition

Voici ce qu'en dit Wikipedia :

Un système de gestion de contenu ou SGC ((en) Content Management System ou CMS) est une famille de logiciels destinés à la conception et à la mise à jour dynamique de sites Web ou d'applications multimédia. Ils partagent les fonctionnalités suivantes :

- ils permettent à plusieurs individus de travailler sur un même document ;
- ils fournissent une chaîne de publication (workflow) offrant par exemple la possibilité de mettre en ligne le contenu des documents ;
- ils permettent de séparer les opérations de gestion de la forme et du contenu ;
- ils permettent de structurer le contenu (utilisation de FAQ, de documents, de blogs, de forums de discussion, etc.) ;
- ils permettent de hiérarchiser les utilisateurs et de leur attribuer des rôles et des permissions (utilisateur anonyme, administrateur, contributeur, etc.) ;

• Utilités

Là où l'utilisation d'un CMS va commencer à devenir intéressante pour vous, c'est quand un client va typiquement venir vers vous en vous demandant : « je voudrais un site à 90% statique mais avec la possibilité de mettre des news quand je veux pour mettre en avant certains évènements de mon magasin ».

Dans un monde sans CMS (un site statique donc), le client vous appelle, vous transmet les informations à changer pour mettre un nouvel article sur son site. Vous devez ré-ouvrir vos fichiers .html et les modifier puis remettre tout sur le serveur et ce … à chaque fois que le client a une demi-virgule à changer sur son site. Vous comprenez que ça devient vite fastidieux.

C'est là que les CMS vous offrent une solution sur-mesure !

En effet, avec un CMS, vous définissez quel type de contenu devra être mis à jour régulièrement et vous en faites un contenu « dynamique » que le client va pouvoir créer et modifier lui-même sans passer par vous.

En terme de travail, pour vous, ça demande un peu plus de temps avant que le site ne soit mis en ligne, mais, gros avantage, une fois que tout est terminé et que vous avez formé votre client à l'utilisation du CMS que vous aurez choisi pour lui, normalement, vous n'entendrez plus trop parler de lui en ce qui concerne la mise à jour de son contenu.

Le client peut, à tout moment, rentrer dans son interface de gestion côté BackOffice et publier des contenus côté FrontOffice.

Mais au fond, savez-vous quelle est la différence entre back et FrontOffice ?

• BackOffice et FrontOffice

La différence majeure entre le back et le FrontOffice est que le premier n'est visible que par l'administrateur du site après avoir rentré un login et un mot de passe tandis que le FrontOffice est l'interface utilisateur que tous les visiteurs de votre site vont voir quand ils vont se connecter à l'URL de votre site.

Généralement, d'un CMS à l'autre, pour se connecter à une interface de BackOffice d'un CMS, vous devrez entrer une URL particulière liée au CMS que vous aurez choisi. De cette manière je peux déjà vous citer en exemple que pour un site développé avec Wordpress, l'adresse d'administration du BackOffice est la suivante :

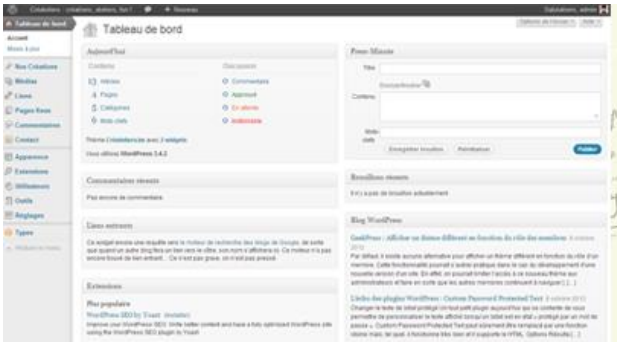
```
http://www.lenomdevotresite.com/wp-admin/
```

Peu importe le nom de votre site, dans Wordpress, pour se connecter à l'interface d'administration, il suffit donc de placer le chemin /wp-admin/ derrière votre URL pour pouvoir accéder au BackOffice.

Si vous avez bien entré le chemin d'accès au BackOffice de Wordpress, vous devriez tomber sur cet écran générique pour toutes les installations Wordpress :

C'est à partir de là que vous ou votre client entrerez vos login et mot de passe (que vous aurez choisis au préalable) pour pouvoir administrer votre site.

Une fois cette étape franchie, vous débarquez sur l'interface réservée au BackOffice qui est à opposer au résultat du FrontOffice :



UN SITE WEB DYNAMIQUE, C'EST QUOI ?

• Définition

Un site web dynamique, c'est un site web qui, dans sa grande majorité, est généré à la demande par opposition à un site web statique où tout est toujours présent, à n'importe quel moment.

Le contenu d'une page web dynamique peut varier en fonction d'informations (heure, nom de l'utilisateur, etc.) qui ne sont connues qu'au moment de sa consultation. À l'inverse, le contenu d'une page web statique est à priori identique à chaque consultation.

Lors de la consultation d'une page web statique, un serveur HTTP renvoie le contenu du fichier où la page est enregistrée.

Lors de la consultation d'une page web dynamique, un serveur HTTP transmet la requête au logiciel correspondant à la requête, et le logiciel se charge de générer et envoyer le contenu de la page. Les logiciels générant des pages web dynamiques sont fréquemment écrits avec les langages PHP, JavaServer Pages (JSP) ou Active Server Pages (ASP).

• Le trio Php-Apache-MySQL

Dans notre cas (mais aussi dans la majorité des autres cas sur la toile), un tri est indissociable quand on parle de sites web dynamiques : PHP, Apache et MySQL.

Php est le langage utilisé pour produire des pages Web dynamiques via un serveur HTTP.

Apache est le serveur web le plus répandu sur Internet permettant à des clients d'accéder à des pages web, c'est-à-dire en réalité des fichiers au format HTML à partir d'un navigateur (aussi appelé browser) installé sur leur ordinateur distant. C'est le support dynamique sur lequel on va se connecter pour demander nos fichiers PHP.

MySQL est un serveur de bases de données relationnelles SQL développé et diffusé sous une licence libre. C'est là qu'on va stocker toutes les données de notre site Web dynamique (articles, commentaires, utilisateurs, etc.). C'est en interrogeant et en extrayant le contenu de MySQL qu'on va pouvoir aller nourrir les fichiers PHP pour obtenir des pages HTML classiques et lisibles par nos navigateurs.

Vous l'aurez compris, ce trio indissociable est **essentiel** au déploiement d'un site Web dynamique. Vérifiez donc que ces 3 éléments sont présents chez l'hébergeur auprès duquel vous allez souscrire pour votre nom de domaine (par exemple). Mais je vous rassure, normalement, la grande majorité des hébergeurs sérieux disposent de ce trio magique.

• Travailler directement sur le serveur ou en local ?

Avec HTML et CSS, vous aviez l'habitude de travailler d'abord en local (sur vos disque-durs physiques) et ensuite de placer tout en ligne sur le serveur distant quand le résultat est fini.

Avec les sites dynamiques c'est un peu différent, puisque vous avez besoin d'une base de données (MySQL), d'un serveur dynamique distant (Apache) et d'une moulinette qui vous renvoie des beaux fichiers HTML lisible par vos navigateurs (PHP).

Ces technologies serveur ne fonctionnent à priori qu'à distance et donc, vos fichiers PHP ne seront efficaces qu'une fois placés sur votre serveur, en ligne.

Est-ce que pour autant il est recommandé de travailler directement en ligne (en uploadant chaque image, bout de code ou autre à chaque demi-changement) ?

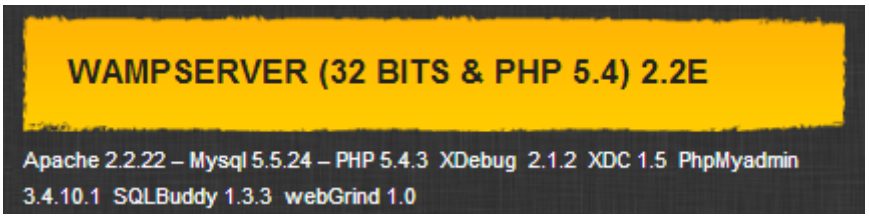
La réponse est **non**.

Et des personnes, avant nous, ont résolu ce problème en créant des logiciels qui simulent un serveur distant sur nos machines en local, vous offrant de cette façon la possibilité de lire du PHP sur votre machine, de créer des bases de données sur votre disque dur et de profiter du serveur Apache aussi en local !

Ces solutions tout-en-un s'appellent des plateformes de développement.

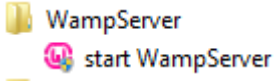
L'une des plus connues et utilisées est Wamp Server (Wamp = Windows Apache MySQL Php).
Le logiciel existe aussi en version Mac (Mamp) et Linux (Lamp).

L'installation est facile, vous allez d'abord sur la page du logiciel selon l'OS que vous utilisez (pour moi Windows donc Wamp Server) et vous téléchargez le fichier qui vous est proposé :

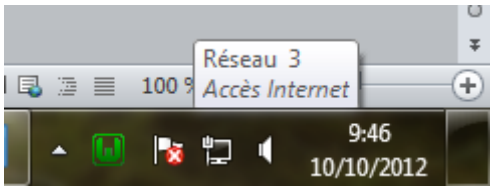


Vous l'installez en laissant les paramètres par défaut (il va, normalement, le placer sur votre disque dur c:/wamp).

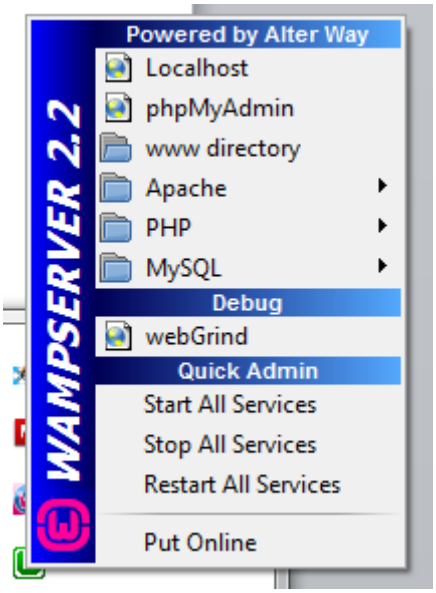
Une fois que c'est fait, il ne vous reste plus qu'à lancer Wamp.



Vous cliquez sur le lien, et une fois que c'est fait, une petite icône va venir apparaître dans votre barre des logiciels résidents (en bas à droite sur Windows) :

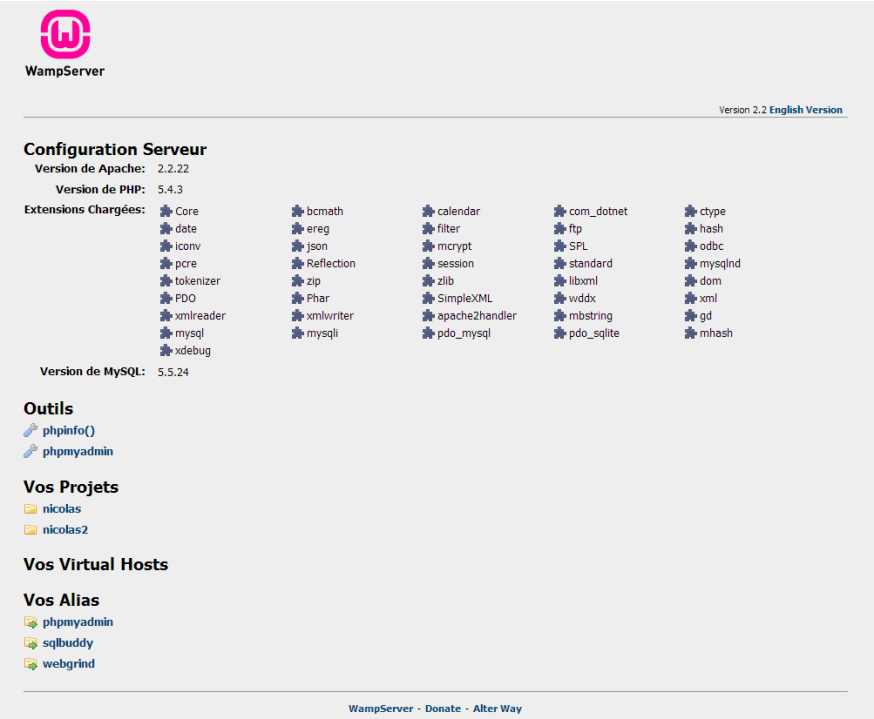


Si l'icône est en vert c'est bon signe, mais assurez-vous que le serveur est bien en ligne en cliquant du gauche sur l'icône et en cliquant le dernier lien « put online ».

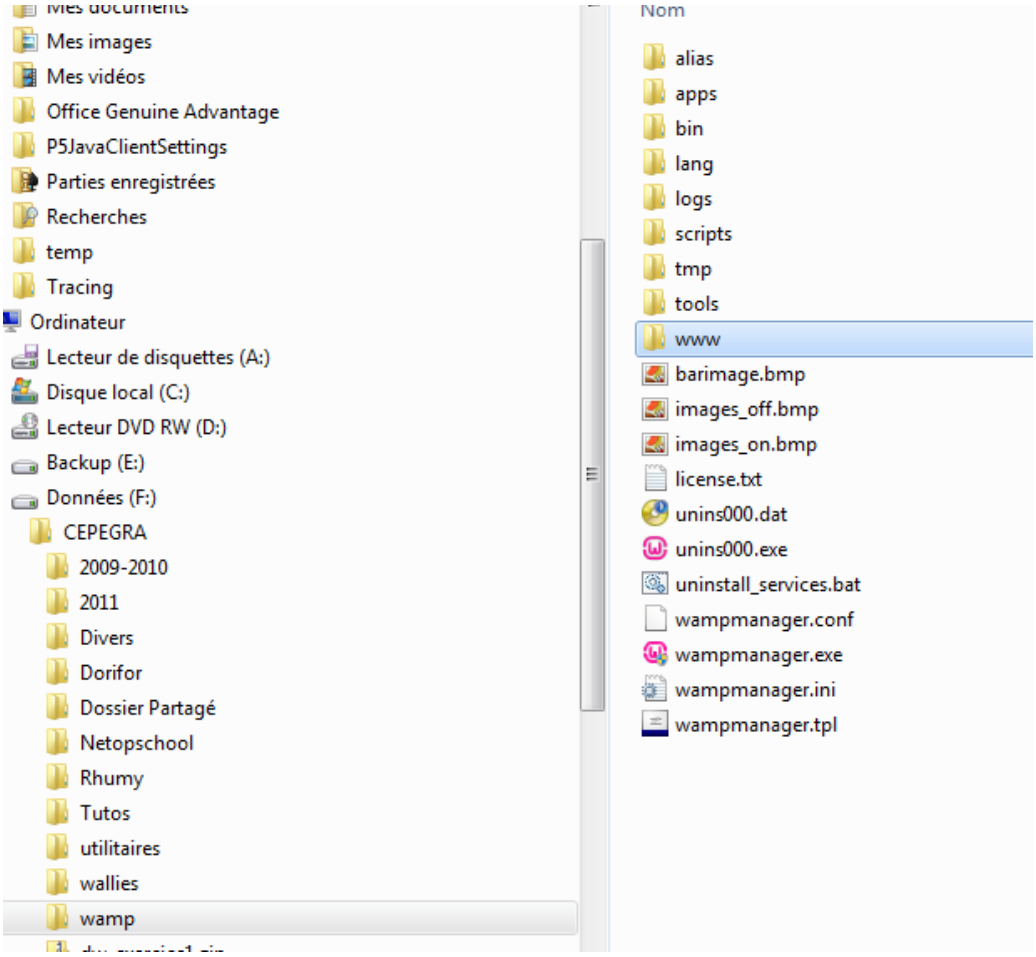


L'icône passe au rouge puis redevient verte et là, c'est ok, tout est prêt à fonctionner ! (youpie)

Pour vous assurer que tout ceci fonctionne, il vous suffit d'ouvrir un navigateur, de taper comme URL « **localhost** » (ou 127.0.0.1 – parfois il est nécessaire de passer par là si certaines choses dysfonctionnent) et là, vous devriez tomber sur une page de présentation de Wamp Server, si c'est le cas, c'est ok, sinon il faut voir d'où vient le problème (est-ce que Wamp a bien été « mis en ligne » ?).



Dirigez-vous maintenant vers votre répertoire où vous avez installé Wamp (à priori par défaut donc sur le c:/wamp – mais n'importe où ailleurs c'est correct également). Et regardons ensemble le seul dossier qui va nous intéresser, le dossier intitulé « www » :



C’est ce dossier qui fait office de serveur virtuel. Ça veut simplement dire que n’importe quel fichier dynamique (Php) inséré dans ce dossier sera interprété comme s’il était sur un serveur distant (plus besoin donc de placer vos fichiers Php en ligne pour les tester). (youpie²)

C’est donc dans ce dossier que nous allons pouvoir installer différents CMS et que nous allons aussi pouvoir jouer comme s’il s’agissait de notre serveur distant ! (youpie³)

J'AI BESOIN DE QUOI POUR INSTALLER UN CMS ?

Pour installer un CMS, en termes de logiciels ou de fichiers vous avez besoin :

- D'une technologie serveur dynamique mise en place en local ou en distant (on vient de le voir ensemble),
- d'une base de donnée installée et de connaître ses données techniques (nom de la base de données, utilisateur qui a le droit d'administration sur cette base, mot de passe de cet utilisateur et finalement l'adresse de la base de données),
- du fichier d'installation (souvent un .zip avec plein de fichiers à l'intérieur (wouaaaah)) du CMS que vous désirez installer,
- d'un client FTP uniquement si vous travaillez en distant (et des données de configuration pour vous connecter à votre serveur distant – ces données sont fournies par vos hébergeurs),
- … et c'est tout !

En termes de connaissances ou de savoir-faire, vous n'avez même pas besoin, pour installer un CMS d'avoir des connaissances en intégration ou autre, tout ce qu'il faut savoir faire c'est :

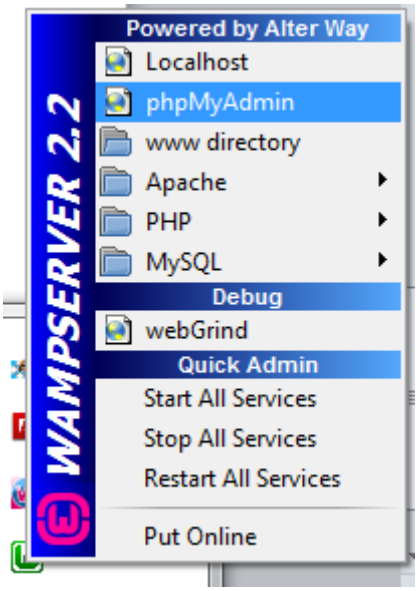
- Savoir lire (le fichier d'explication pour l'installation du CMS choisi)
- Savoir éditer un fichier avec n'importe quel éditeur de texte (bloc-note suffit) et y remplir 2 ou 3 informations données dans les étapes ci-dessus.
- … et c'est tout !

Je ne reviendrai pas sur la technologie serveur dynamique mise en place puisqu'on vient de le voir ensemble, penchons-nous d'abord sur l'installation d'une base de données.

• Installer une base de données (en local ou distant)

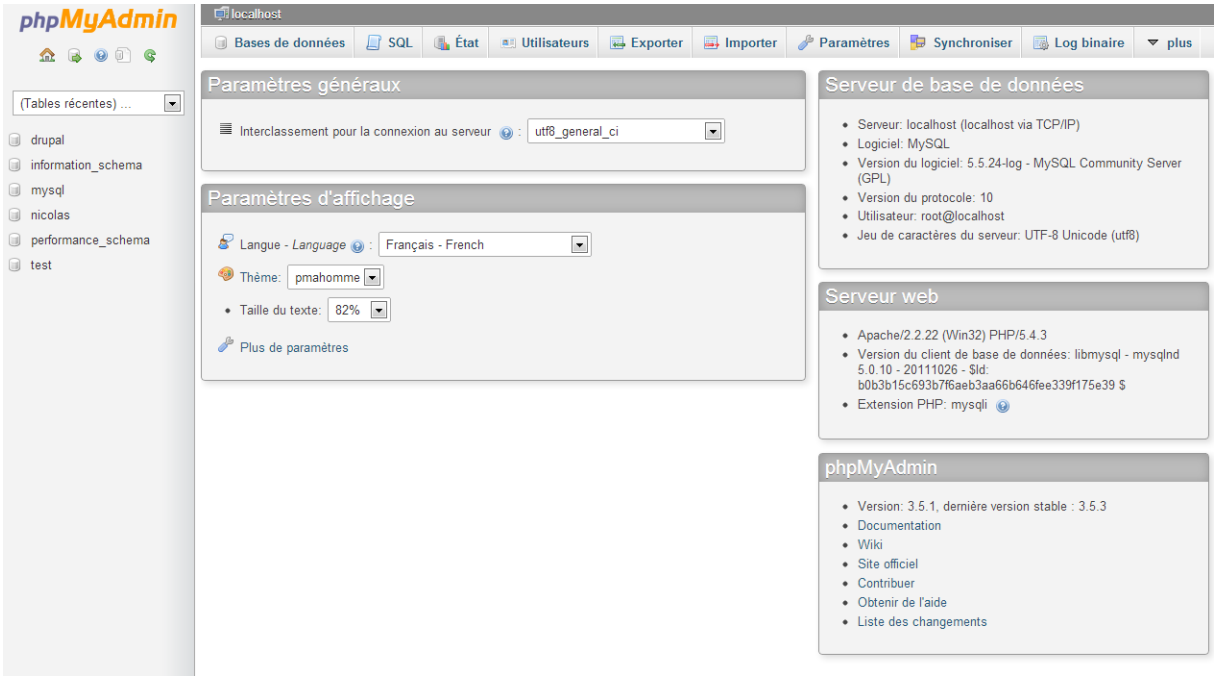
En local

Pour installer une base de données à partir de Wamp c'est facile, vous cliquez du gauche sur l'icône de Wamp et vous sélectionnez phpMyAdmin :



Si, pour des raisons obscures, cela ne fonctionne pas, passez par l'adresse 127.0.0.1 et, sur votre page d'administration de Wamp Server, cliquez sur le lien de PhpMyAdmin, en bas de page (sous « Outils » :

Vous arrivez ensuite sur l'interface d'administration de PhpMyAdmin :



Vous cliquez ensuite sur l'onglet « Base de données » et là, vous donnez un nom à votre base de données (qui aura un lien avec le thème du CMS que vous allez mettre en place par exemple, ou le nom du site pour lequel vous allez travailler, bref un nom cohérent !).

Vous appuyez ensuite sur créer et hop, c'est fait, votre base de données est prête à être utilisée (youpieuh !).

Retenez que, en local, quand vous utilisez Wamp Server, les données techniques (nom de la base de données, utilisateur qui a le droit d'administration sur cette base, mot de passe de cet utilisateur et

finalement l'adresse de la base de données) sont souvent les mêmes (sauf le nom de la base de données que vous avez choisi).

Ces 4 données capitales sont donc :

- **Nom de la base de données** : Nomdevotrebasededonnées
- **Nom de l'utilisateur** : root
- **Mot de passe de l'utilisateur** : (rien – néant)
- **Adresse de la base de données** : localhost (proposé par défaut)

Sur un serveur distant

Ici, ça va dépendre de l'hébergeur que vous aurez choisi mais, peu importe votre choix d'hébergeur, si celui-ci vous a promis lors de la vente une base de données disponible, c'est suffisant pour installer plusieurs CMS sur la même base.

Vous devez donc, dans l'interface d'administration de votre hébergeur, trouver l'endroit où vous pouvez créer une base de données. En général, cette création est faite sans que vous deviez passer par phpmyadmin, c'est simplifié pour vous par un champ de formulaire qu'il vous suffit de remplir.

Une fois que c'est fait, vous devriez récupérer les 4 informations capitales habituelles soit sur votre interface d'administration, soit par mail.

Ces 4 données capitales sont donc :

- **Nom de la base de données** : Nomdevotrebasededonnées (parfois avec les hébergeurs distants vous n'avez pas le choix du nom de votre base, il vous sera imposé).
- **Nom de l'utilisateur** : Fourni par votre hébergeur
- **Mot de passe de l'utilisateur** : Fourni par votre hébergeur
- **Adresse de la base de données** : localhost (proposé par défaut) ou une autre adresse fournie par votre hébergeur

INSTALLONS WORDPRESS PAS À PAS

L'installation de Wordpress se vante d'être complète en 5 minutes chrono, si vous avez bien toutes vos données techniques liées à l'installation de votre base de données, nous allons voir qu'ils ne mentent pas !

Voici les différentes étapes à franchir pour arriver à une installation complète de Wordpress (que ce soit en local ou en distant) en supposant que l'étape de la création de la base de données soit ok :

- Téléchargez le pack d'installation de Wordpress
- Décompressez-le sur votre disque dur
- Éditez le fichier de configuration avec les **4 données importantes** vues précédemment
- Placez tout le contenu de votre dossier décompressé soit en local (dans wamp/www) ou n'importe où sur votre serveur distant (avec un client FTP)
- Accédez ensuite à l'endroit (en local ou distant) où vous avez placé votre dossier et remplissez quelques informations sur votre site
- C'est prêt !

• Téléchargement de Wordpress

Pas trop moyen de se tromper à cette étape-ci : vous allez sur le site officiel de Wordpress francophone (http://www.wordpress-fr.net) et, sur le homepage, vous cliquez sur l'énorme bouton :

Vous l'enregistrez où vous voulez sur votre disque dur, il s'agit d'un fichier .zip qui contient la dernière version (ici 3.4) de Wordpress en version française.

De manière générale, n'importe quel CMS fonctionne de cette façon-là.

• Décompressez l'archive sur votre disque dur

Retrouvez le fichier que vous venez de télécharger et décompressez tous les fichiers qui se trouvent dans l'archive dans un dossier au nom de votre projet CMS (par exemple).

• Éditez le fichier wp-config-sample.php

Dans la liste des fichiers que vous venez de décompresser, à la racine, se trouve un fichier qui porte le nom wp-config-sample.php.

Renommez-le en « wp-config.php » puis éditez-le avec votre éditeur de code favori.

De la ligne 22 à la ligne 31 se trouvent les seules informations que vous allez devoir changer pour que le paramétrage de votre Wordpress soit correct :

```
20  // ** Réglages MySQL - Votre hébergeur doit vous fournir ces informations. ** //
21  /** Nom de la base de données de WordPress. */
22  define('DB_NAME', 'votre_nom_de_bdd');
23
24  /** Utilisateur de la base de données MySQL. */
25  define('DB_USER', 'votre_utilisateur_de_bdd');
26
27  /** Mot de passe de la base de données MySQL. */
28  define('DB_PASSWORD', 'votre_mdp_de_bdd');
29
30  /** Adresse de l'hébergement MySQL. */
31  define('DB_HOST', 'localhost');
```

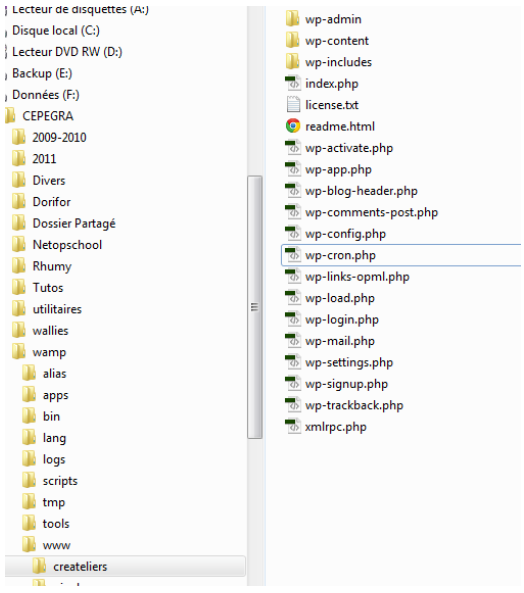
Les 4 lignes qui ne sont pas en commentaire ci-dessus sont les 4 fameuses données importantes dont on parle depuis presque le début de ce cours.

- À la ligne 22, vous devez remplacer 'votre_nom_de_bdd' par le vrai nom de la base de données que vous avez créée précédemment.
- À la ligne 25, vous devez remplacer 'votre_utilisateur_de_bdd' par le nom de l'administrateur de la base de données.
- À la ligne 28, vous devez remplacer 'votre_mdp_de_bdd' par le mot de passe de la base de données.
- À la ligne 31, vous devez remplacer 'localhost' par l'adresse de l'hébergement MySQL s'il est différent de localhost, ce n'est pas toujours le cas.

Une fois ces 4 paramètres changés et quand vous êtes certain des données que vous avez rentrées, vous pouvez sauvegarder et fermer votre fichier « wp-config-sample.php ».

• Placez votre dossier sur votre serveur (local ou distant)

Il suffit ensuite de placer votre dossier complet soit sur votre serveur local (dans c:/wamp/www) ou n'importe où sur votre serveur distant (en passant par un client FTP – ce sera forcément beaucoup plus long).



- # Accédez à votre dossier (local ou distant) et remplissez quelques informations à propos de votre site

Ouvrez votre navigateur et accédez au dossier que vous avez placé soit :

- Localhost/votrenomdedossier/ (ou 127.0.0.1/votrenomdedossier/)
- http://www.votrenomdedomaine.be/votrenomdedossier/

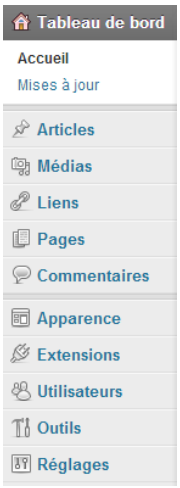
Si vous avez bien fait votre boulot, vous devriez arriver sur cette page où vous seront demandés :

- Le titre de votre site
- L'identifiant que vous désirez utiliser pour vous logger à la partie BackOffice de votre site
- Votre mot de passe pour vous logger
- Votre adresse e-mail
- La possibilité ou non d'indexer votre site sur les moteurs de recherche
- … et finalement un bouton installer Wordpress qui fait tout le travail pour vous !

WORDPRESS : INTERFACE ET POSSIBILITÉS

Nous allons maintenant découvrir ensemble quelles fonctionnalités de base Wordpress permet dès son installation. Pour ce faire, nous allons parcourir ensemble les différents menus présents par défaut dans la colonne de gauche de votre interface de BackOffice de Wordpress :

- Tableau de bord
- Articles
- Médias
- Liens
- Pages
- Commentaires
- Apparence
- Extensions
- Utilisateurs
- Outils
- Réglages



Mais commençons par la disposition générale de l'interface de votre zone d'administration de Wordpress.

• L'interface de Wordpress

L'interface de base de Wordpress se scinde en 3 parties principales :

- Une toolbar en top
- Un menu de navigation à gauche
- La zone du contenu dans la partie de droite

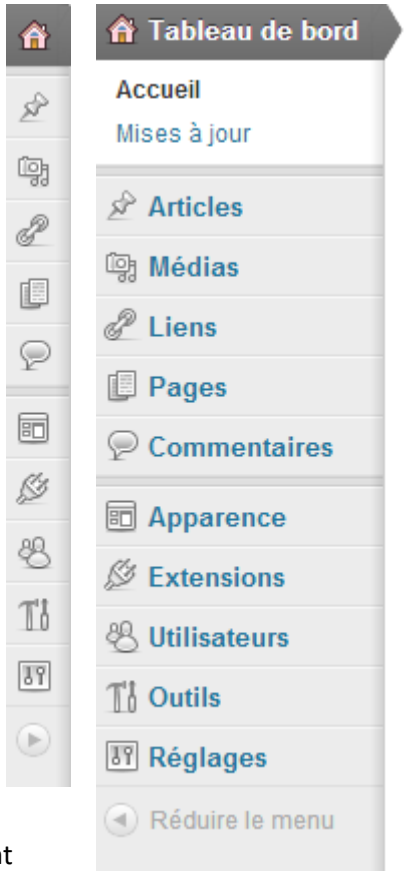
La Toolbar

Cette toolbar reprend 5 boutons (de gauche à droite) :

- Un bouton avec le logo de Wordpress qui rapporte à des liens directs pour une aide liée à Wordpress (forum, aide, etc.)
- Un bouton menant directement à votre site côté FrontOffice
- Un accès rapide à la modération des commentaires (le phylactère)
- Un bouton pour ajouter rapidement un des types de contenu : un article, un média, un lien, une page ou un utilisateur
- Et finalement tout à droite, un bouton pour se déconnecter ou pour modifier son profil, accompagné d'une représentation de vous liée à un compte Gravatar (qui permet, à l'aide d'une seule adresse e-mail, d'avoir toujours la même photo de profil sur tous les sites qui ont souscrit à Gravatar).

Notons finalement que, si vous êtes loggé sur Wordpress et que vous lancez votre site en aperçu, vous garderez la toolbar active en surfant sur votre site, pour pouvoir retourner plus facilement côté BackOffice.

Le menu de gauche



Le menu de gauche vous ramène à tout ce que vous pouvez faire dans Wordpress.

Quand une catégorie est active, le fond est grisé et ses sous-catégories (aussi dispo en roll-over) apparaissent dans une zone blanche sous lui.

Grâce au bouton « Réduire le menu » tout en bas, vous pouvez replier le menu pour qu'il ne soit plus représenté que par ses icônes.

Une légère séparation est présente entre les premiers éléments et ceux du bas (à partir de « Apparence »).

C'est simplement pour marquer une séparation entre les éléments typiquement connotés « contenu » et les éléments qui vont servir à paramétrer votre site Wordpress.

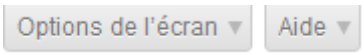
Plus tard, quand vous installerez des extensions, il est probable que ce menu comporte de nouveaux choix, liés directement à l'extension que vous aurez installée.

Parfois même, ces extensions vont se placer dans les sous-menus (souvent sous « réglages »).

La zone de contenu de droite

Dans la partie de droite va venir s'affiche le résultat des clics effectués soit dans la toolbar, soit dans le menu.

De plus, 2 petits onglets inversés sont placés dans le coin supérieur droit :



Ces petits onglets vous permettent d'affiche les options de l'écran en cours. Attention, ces options varient à chaque type d'écran sur lequel vous êtes. De cette manière vous aurez + ou – d'options selon que vous êtes en train d'écrire un article ou de paramétrer vos mots-clés.

Le bouton aide vous explique, un peu comme je le fais à l'heure actuelle, le détail de chacune des rubriques que vous parcourez, au cas où ce que j'écris n'étais pas clair, n'hésitez pas à aller y jeter un œil.

• Le tableau de bord

Le tableau de bord, comme son nom l'indique, est un ensemble de panneaux, fonctionnalités, informations les plus utiles et les plus couramment utilisées dans Wordpress qui sont regroupées

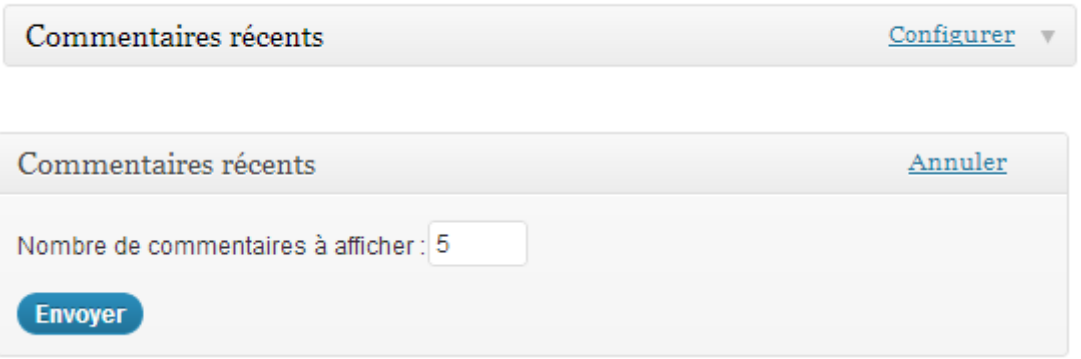
ensemble de manière condensée pour vous permettre de les modifier rapidement et d'y avoir accès d'un coup d'œil dès votre connexion au BackOffice.

Vous avez la possibilité de replier les panneaux pas utiles pour vous en cliquant sur le petit triangle dans le coin supérieur droit de chaque boite :

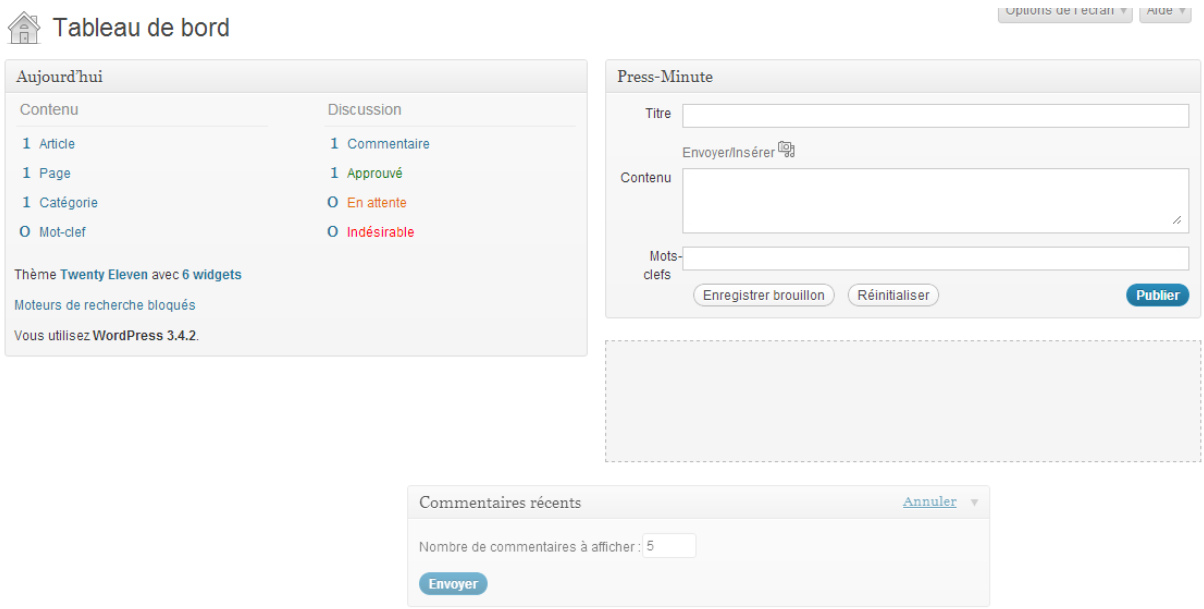
Par défaut, le tableau de bord affiche les boites suivantes :

- Aujourd'hui : indique le nombre d'article publiés aujourd'hui, le nombre de commentaires reçu et leur état de modération ainsi que le thème installé, le nombre de widgets et la version de Wordpress.
- Commentaires récents : affiche le contenu des derniers commentaires et permet de filtrer les commentaires selon leur statut (en attente de relecture, approuvé, indésirable, corbeille ou tous).
- Liens entrants : Ce widget envoie une requête vers le moteur de recherche des blogs de Google, de sorte que quand un autre blog fera un lien vers le vôtre, son nom s'affichera ici. Ce moteur n'a pas encore trouvé de lien entrant… Ce n'est pas grave, on n'est pas pressé.
- Extensions : vous propose des extensions populaires ou nouvelles, tous les jours.
- Press-Minute : vous permet d'écrire un article rapide avec un titre, du contenu et même des images et des mots-clés !
- Brouillons récents : affiche les derniers brouillons (articles non parus) que vous avez créés.
- Blog Wordpress : les dernière news en direct des sites officiels de Wordpress
- Autres actus : pareil que pour le Blog Wordpress mais uniquement en français.

En passant votre curseur sur la barre de certains blocs, vous aurez la possibilité d'utiliser un bouton « configurer » qui permet de changer certaines choses à certains blocs présents sur le tableau de bord (le nombre de commentaires affichés par exemple pour le bloc commentaires) :



Vous pouvez également ré-organiser les blocs en utilisant le drag and drop, ils s'articuleront selon vos envies :



LE THEMING AVEC WORDPRESS

• Préambule

Apprendre à créer un thème utilisable dans Wordpress pourra vous aider à créer des solutions « sur mesure » pour vos clients à l'avenir.

Maîtriser cette technique vous amènera à un niveau peu maîtrisé par la plupart des utilisateurs et créateurs de sites avec Wordpress qui se contentent généralement d'installer Wordpress, de la paramétrer, d'y installer un thème tout-fait (qui ne répond pas forcément à toutes les attentes de votre client) et finalement de bidouiller pas mal pour arriver à un résultat final peu satisfaisant.

Pour parvenir à cette maitrise, il vous suffit d'avoir un site statique parfaitement codé et fini en html/css mais aussi quelques petites notions de PHP essentielles à la mise en place des fonctions de Wordpress.

Rassurez-vous tout de suite, vous allez peu écrire du PHP, la majorité des codes PHP que vous allez insérer sont des « Wordpress Tags » qui, un peu à la façon de jQuery, font déjà, à eux-seuls, des choses sans rien ajouter. Il suffit de les places au bon endroit et d'éventuellement leur ajouter quelques paramètres pour qu'ils s'articulent dans votre site comme vous le désirez.

• Pourquoi créer son propre thème Wordpress ?

Il existe déjà beaucoup de thèmes existants, dont de nombreux de très bonne qualité. De plus, se lancer dans une telle aventure demande pas mal de temps et d'énergie… Oui, mais il y a aussi pas mal d'avantages à créer son propre thème Wordpress et ceux-ci ne sont pas négligeables.

Selon les développeurs de Wordpress, créer son propre thème :

- C'est créer son propre univers, avoir un site avec le look que l'on veut,
- D'un point de vue technique, mettre les mains dans le cambouis, c'est mieux comprendre les templates, les template tags ou encore le Wordpress Loop et en tirer profit. Ainsi, vous pourrez customiser votre site comme vous le souhaitez, que ce soit au niveau du look mais aussi des fonctionnalités,
- C'est aussi la possibilité de créer des environnements qui n'existent pas encore chez les thèmes proposés par Wordpress,
- Possibilité aussi de faire des "releases" publiques du ou des thèmes créés,
- C'est une excellente opportunité pour mieux connaître et comprendre le XHTML, CSS et PHP,
- C'est un très bon travail de créativité et c'est fun (sisi ! ☺)!

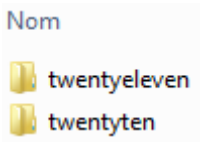
C'est vrai que c'est très intéressant. On comprend vraiment beaucoup de choses et on peut utiliser son site au maximum. Pour cela, pas forcément besoin de beaucoup de connaissances techniques, mais une bonne compréhension du fonctionnement d'un site et le minimum de connaissances en XHTML, CSS et PHP sont nécessaires pour ne pas être perdu très rapidement, ni frustré de ne pas arriver à ce qu'on veut.

Un thème c'est quoi ?

Un thème c'est tout d'abord un dossier qui va séjourner, au niveau de l'arborescence des fichiers de Wordpress dans le dossier `wp-content/themes` de votre installation.

Si vous avez 5 dossiers présents dans `wp-content/themes`, vous aurez 5 thèmes disponibles à l'activation dans votre menu apparences/thèmes.

Par défaut, quand vous installez Wordpress, vous n'avez que 2 thèmes installés :



Un thème est composé de plusieurs genres de fichiers :

- **style.css** : c'est la feuille de style de votre site web, elle doit obligatoirement porter ce nom-là pour être reconnue par Wordpress
- **screenshot.jpg ou .png** : c'est la capture d'écran de votre thème que vous retrouverez dans votre administration au moment d'activer votre thème. C'est donc sa représentation graphique côté administration.
- **Des fichiers de templates en php** : vous aurez plein de fichiers php différents qui ont tous une utilité spécifique. Par exemple, index.php est utilisé pour la page d'accueil, tandis que single.php représente un article, etc.
- **Tous vos autres fichiers ou dossiers** : c'est aussi là que vous allez placer vos autres fichiers (images, fonts exotiques, autres feuilles de styles, javascript, etc.)

Un thème c'est une association de tous les fichiers ci-dessus dont certains sont obligatoires, d'autres pas.

Nous allons voir l'utilité de chacun de ces fichiers un par un.

Style.css : la feuille de styles

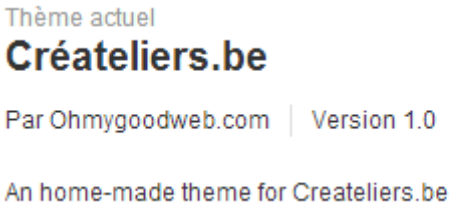
Elle est essentielle dans Wordpress et doit obligatoirement porter le nom « style.css » (au singulier).

Donc, si, au préalable, vous avez déjà réalisé l'intégration de votre site et de ses différentes pages, vous reprendrez votre feuille de style générale et vous la renommerez en « style.css ».

Ensuite, étape importante, vous indiquerez ce bloc d'information directement dans les premières lignes de votre fichier CSS en commentaires :

```
/*
Theme Name: bbxpress
Theme URI: http://wordpress.bbxdesign.com/
Description: WordPress tutorial theme
Version: 1.0
Author: bbx
Author URI: http://bbxdesign.com/
Tags: blue, full-width, simple
*/
```

Les informations présentes dans ce fichier vont venir s'inscrire directement à côté de la représentation du screenshot de votre thème, dans l'interface d'administration de Wordpress, au niveau du menu Apparence/thèmes comme vous pouvez le voir ci-dessous :



Les différentes informations fournies dans le fichier CSS sont les suivantes :

- **Theme Name** : Le nom qui apparaîtra pour représenter votre thème (ici : Créateliers.be)
- **Theme URI** : Un lien qui va accompagner le nom de votre thème
- **Description** : Une description, courte ou longue de votre thème et de ce qu'il permettra (si vous désirez le mettre à la disposition des autres plus tard). (ici : « An home-made theme for Createliers.be)
- **Version** : Le numéro de version de votre thème, si vous en faites plusieurs versions (ici : 1.0)
- **Author** : Qui a réalisé le thème (ici : Par Ohmygoodweb.com)
- **Author URI** : L'URL liée à l'auteur pour ramener vers son site officiel s'il en a un.
- **Tags** : des mots clés de classification pour le thème si vous le mettez à disposition de la communauté, vous pourrez le retrouver grâce à ces tags.

• Screenshot : l'aperçu de votre thème côté administration

Ce fichier, à placer au même niveau que la feuille de style (directement à la racine de votre thème), doit porter le nom « screenshot » avec l'extension d'image « png » ou « jpg » selon votre choix.

Pour être optimal et remplir tout l'espace du screenshot réservé à cet effet dans Wordpress, il doit mesurer 300x225.

Voici un exemple pour Créateliers.be :

• Les templates PHP

Les fichiers de template sont des fichiers PHP utilisés par Wordpress pour générer du code HTML (qui est, pour rappel, le seul code que vos navigateurs peuvent comprendre).

Il existe des différences entre les fichiers de template PHP. Certains sont des fichiers de template complets et d'autres ne sont que des parties qui vont composer ces templates complets.

Commençons par les fichiers communs qui composent vos templates complets :

- **Header.php** : définit la portion de code qui est commune à toutes vos pages et qui regroupe tout le code qui va de la déclaration du `doctype` jusqu'à la dernière balise commune à toutes les parties supérieures de vos pages.
- **Footer.php** : définit la portion de code qui est commune à toutes vos pages et qui regroupe tout le code termine chacune de vos pages (et ce inclus les balises `</body>` et `</html>`).
- **Sidebar.php** : un bloc de page supplémentaire qui est facultatif. Ce n'est pas parce que son nom veut dire « barre latérale » qu'il doit d'office représenter une barre sur le côté de votre site. Ça pourrait être un bloc présent sur certaines pages et pas d'autres. Souvent, c'est dans ces blocs-là qu'on place les widgets mais on peut aussi placer les widgets ailleurs (dans le header et le footer par exemple).
- **Searchform.php :** si vous désirez personnaliser le champ de recherche de base de Wordpress, retrouvez le fichier searchform.php dans un autre thème et personnalisez-le. Vous aurez un champ de recherche personnalisé.

Ces bouts de code forment des templates complets (on verra comment juste après) qui sont les suivants :

- **Index.php** : c'est le fichier de votre page d'accueil !
- **Single.php** : c'est le fichier de template dans lequel Wordpress puisera pour produire le code HTML de chacun de vos articles de votre site.
- **Page.php** : c'est le fichier de template dans lequel Wordpress puisera pour produire le code HTML de chacun de vos « pages statiques » de votre site.
- **Archive.php** : c'est le fichier de template dans lequel Wordpress puisera pour produire le code HTML de vos pages d'archives quand vous cliquez sur une information de type : date, taxonomie, auteur de l'article, catégorie.
- **Search.php** : c'est le fichier de template dans lequel Wordpress puisera pour produire le code HTML de vos résultats de recherche (à partir d'un champ de recherche).
- **404.php** : c'est le fichier de template dans lequel Wordpress puisera pour produrie le code HTML de votre page d'erreur 404.

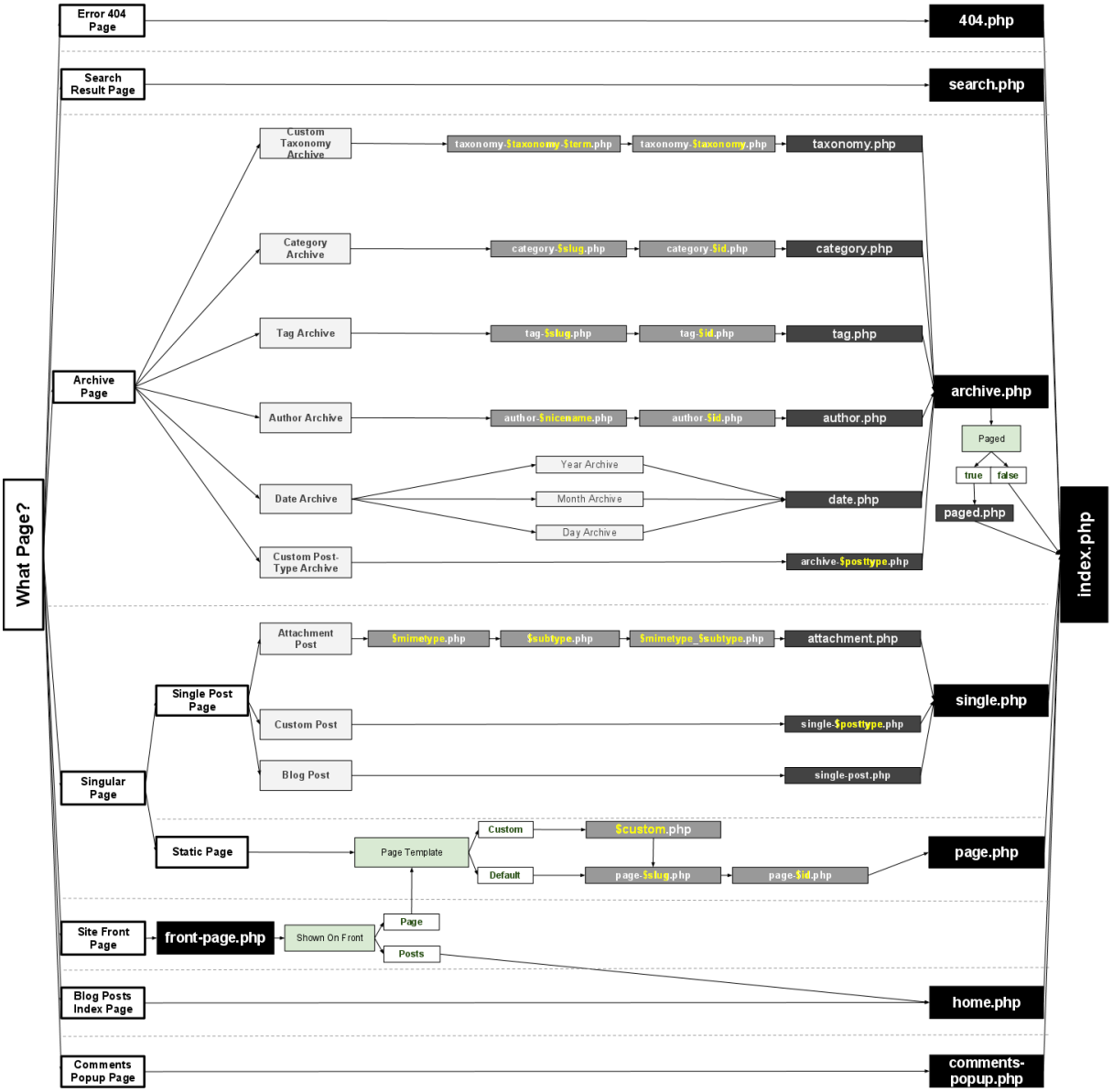
Il existe encore d'autres templates de pages complets, nous les verrons un peu plus tard, car il en existe une infinité que vous pouvez décliner à partir de vos pages single.php ou page.php.

Faites bien attention aux noms de ces fichiers, il est très important de les respecter pour que Wordpress puisse les reconnaître.

Hiérarchie des fichiers

Voici la hiérarchie des fichiers les uns par rapport aux autres. Il faut bien comprendre quel fichier Wordpress « charge » quand vous faites une requête pour une page ou l'autre.

http://wordpress.bbxdesign.com/wp-content/uploads/2012/04/hierarchie-templates-wordpress-full.png



Il est capital de bien comprendre ce schéma.

Par exemple, si vous rentrez une URL invalide sur votre Wordpress, il va chercher le fichier 404.php, si celui-ci n'existe pas, alors il utilise le template de la page index.php pour sortir le résultat.

On voit ici plus clairement qu'un clic sur un élément de type taxonomie, catégorie, mot clé, auteur ou date redirige vers le fichier archive.php. Si celui-ci n'existe pas, c'est la page d'accueil qui est à nouveau employée.

Bien comprendre ce schéma, c'est comprendre le fonctionnement général de Wordpress et donc pouvoir mieux appréhender les prochaines étapes qui vont arriver.

• Le codex de Wordpress

Le codex de Wordpress (http://codex.wordpress.org/) est un peu le dictionnaire d'utilisation de Wordpress, il reprend toutes les fonctions de Wordpress, comment les utiliser, quels sont les paramètres que nous pouvons leur ajouter et aussi des exemples de mise en place au cas où tout ce qui précède n'aurait pas suffi à votre compréhension.

On y retrouve entre autre la liste de tous les Template Tags qui sont le cœur-même de Wordpress.

• Les Template Tags

Qu'est-ce qu'un template tag ?

Un template tag, c'est un code qui ordonne à Wordpress de faire quelque chose ou d'aller chercher quelque chose quelque part.

Comment écrit-on un template tag ?

L'écriture d'un template tag doit toujours être faite en PHP sinon Wordpress ne peut pas la comprendre et c'est surtout la façon d'ajouter du dynamisme à vos sites.

```
<?php template_tag_name('parameters'); ?>
```

Il y a trois zones décrites dans le bout de code ci-dessus :

- En rouge : la syntaxe PHP pour que la lecture dynamique de ce code ait lieu
- En bleu : le nom du template tag qui doit toujours être accompagné de parenthèses même s'il n'y a pas de paramètre à l'intérieur
- En vert : les paramètres liés au template tag en question, ils peuvent être plusieurs et sont généralement encadrés par des simple quotes.

Comme vous pouvez le voir, ce bout de code est précédé de `<?php` (qui est la marque d'une insertion de code PHP dans votre code html) et se termine par `?>` (qui est la marque de la fermeture de votre insertion PHP).

Un template tag peut se retrouver dans n'importe quelle balise HTML : <title>, <head>, <h1>, , etc.

Si vous laissez les paramètres vides, alors Wordpress utilise les paramètres par défaut du template tag. Il est donc souvent important de placer des paramètres à l'intérieur des parenthèses.

Les paramètres d'un template tag

Les paramètres sont importants pour les template tags, ils permettent de spécifier comment le template tag doit se comporter, comment il doit afficher les résultats, ce sur quoi il doit vraiment agir, etc.

Syntaxiquement, vous reconnaîtrez les paramètres car ils sont placés à l'intérieur des parenthèses et des simples quotes ('parameters').

• Créons notre propre thème

0. Étapes préliminaires

Avant d'envisager commencer à vraiment créer votre thème, il vous faut avoir rempli certains critères :

- Avoir fini l'intégration complète de chacune des pages structurellement différentes de votre site.
- Avoir installé Wordpress « de base » à un endroit (ce qui implique que vous avez une base de donnée dédiée et tous les accès qui vont avec).
- Avoir une structure bien commentée et claire tant au niveau de votre code HTML que dans vos règles CSS

Quand vous avez réuni tous ces points-là alors vous pouvez commencer le theming.

1. Créez le dossier pour votre thème

La toute première chose à faire, qui ne vous prendra pas beaucoup de temps, c'est de créer, dans le répertoire wp-content/themes/ un dossier qui portera le nom de votre thème.

Dans l'exemple ci-dessus, le thème que nous allons créer portera le nom « Biodimestica ».

Dès que vous aurez créé ce dossier vide, vous pouvez déjà aller voir côté back-office (dans Apparence-Thèmes). Et vous allez voir qu'un nouveau thème est sur le point d'être installé mais qu'il reste un problème :

C'est tout à fait normal, tant que vous ne serez pas passé à l'étape suivante, ce message restera visible.

2. Importez vos feuilles de style et créez un screenshot de votre thème

Il est temps d'importer votre feuille de style ! En commençant par la feuille principale. Vous allez pouvoir la copier de votre répertoire où vous aviez réalisé votre intégration HTML initiale et la coller directement dans le nouveau répertoire que vous venez de créer et qui porte le nom de votre thème.

screenshot.jpg	22/10/2012 12:02	Fichier JPG	16 Ko
style.css	29/10/2012 15:32	Document de feui...	6 Ko

Dès que vous aurez collé la feuille de style au bon endroit, il vous faudra encore créer une vignette qui représentera votre thème dans le back-office.

Pour cela, comme indiqué dans le cours précédemment, celle-ci doit absolument porter le nom screenshot et l'extension .png ou .jpg et ce fichier doit également avoir les dimensions **300x225**. Vous devez ensuite la placer au même endroit que votre fichier style.css, c'est-à-dire directement dans la racine de votre site.

3. Importez vos images

Prenez votre dossier avec toutes vos images et déplacez-le au même endroit que vos fichiers de style et screenshot :

img	7/11/2012 13:58	Dossier de fichiers	
screenshot.jpg	7/11/2012 13:50	Fichier JPG	61 Ko
style.css	29/10/2012 15:32	Document de feui...	6 Ko

4. Importez vos scripts Js (jQuery, etc.)

Prenez votre dossier avec tous vos fichiers scripts (si vous en avez, sinon passez cette étape ☺) et collez-le au même endroit que le dossier img :

img	7/11/2012 13:58	Dossier de fichiers	
js	7/11/2012 13:59	Dossier de fichiers	
screenshot.jpg	7/11/2012 13:50	Fichier JPG	61 Ko
style.css	29/10/2012 15:32	Document de feui...	6 Ko

5. Créez vos fichiers de template en PHP

Il est maintenant temps de créer vos fichiers PHP vides, vous aurez besoin au minimum des fichiers suivants :

img	7/11/2012 13:58	Dossier de fichiers	
js	7/11/2012 13:59	Dossier de fichiers	
archive.php	7/11/2012 14:01	PHP Script	0 Ko
footer.php	7/11/2012 14:01	PHP Script	0 Ko
functions.php	7/11/2012 14:01	PHP Script	0 Ko
header.php	7/11/2012 14:01	PHP Script	0 Ko
index.php	7/11/2012 14:01	PHP Script	0 Ko
page.php	7/11/2012 14:01	PHP Script	0 Ko
screenshot.jpg	7/11/2012 13:50	Fichier JPG	61 Ko
search.php	7/11/2012 14:01	PHP Script	0 Ko
single.php	7/11/2012 14:01	PHP Script	0 Ko
style.css	29/10/2012 15:32	Document de feui...	6 Ko

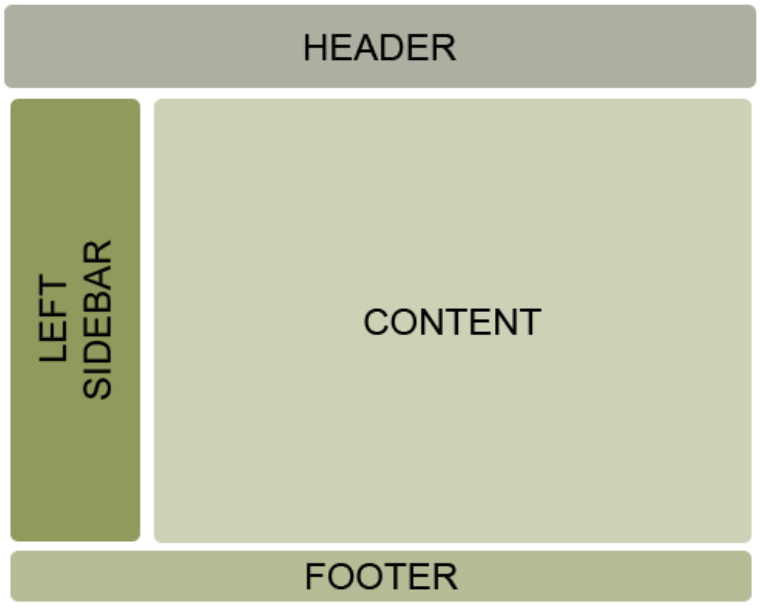
Soit 8 fichiers PHP différents avec EXACTEMENT ces noms-là sinon Wordpress ne les reconnaîtra pas.

6. Scindez votre code HTML en plusieurs sections

Conventionnellement, une page Web comporte plusieurs « sections » en elle :

- Un header dans lequel on retrouve toute l'en-tête de votre document HTML (doctype, head, liens vers les feuilles de style, etc.) mais aussi souvent votre logo, votre menu de navigation, parfois aussi des liens vers les réseaux sociaux ou d'autres choses du genre.
- Une zone de contenu qui présente le contenu de votre site (c'est cette partie-là qui est amenée à changer de page en page alors que le header et le footer restent, eux, souvent semblables).
- Parfois une barre latérale dans laquelle on va venir placer des éléments de navigation, des outils (champ de recherche, articles récents, etc.) ou d'autres choses de ce genre.
- Un footer qui comporte le footer de votre site et qui est présent sur toutes les pages et qui se termine à la balise de fermeture </html>.

On pourrait représenter ça schématiquement sous cette forme :



Avant de commencer à remplir les fichiers PHP, vous allez devoir délimiter les différentes sections de votre site : son header, sa zone de contenu, son éventuelle sidebar et son footer.

Si vous avez bien codé proprement votre HTML, ceci devrait aller très rapidement.

Attention
Tous les sites n'ont pas forcément un footer très bien garni et dans ce cas-là on serait tenté de croire que se passer du fichier footer.php est une bonne idée mais je vous recommande de toujours au moins y placer vos balises de fermeture `</body>` et `</html>` et parfois même aussi vos balises de fermeture de vos conteneurs qui englobent votre site.

7. Les includes

Les includes en PHP sont des portions de code que l'on va inclure aux pages complètes car ces portions de code sont les mêmes sur toutes les pages.

Les includes principalement utilisés dans Wordpress sont :

- **header.php** : le début du code, avec la doctype, le <head>, le début du <body>
- **footer.php** : la fin du code, avec le pied de page et la fermeture des balises </body> et </html>
- **sidebar.php** : une colonne qui contient souvent le formulaire de recherche, un bouton RSS, la liste des catégories, un nuage de tags…
- **searchform.php** : le formulaire de recherche, tout seul
- **comments.php** : le module de commentaires, présent dans single.php

8. Le fichier header.php

C'est le premier fichier que nous allons manipuler.

Vous allez tout d'abord y coller tout le contenu de la zone de votre site que vous aurez identifiée comme étant le header de votre site. Cette partie sera à priori visible sur toutes les pages de votre site.

On y retrouvera communément :

- Le DocType,

- la balise `<head>` et tout ce qu'elle contient :

    - les `<meta>`,

    - les liens vers les feuilles de style,

    - les liens vers les feuilles de scripts,

    - la balise `<title>`,

- les balises d'ouverture de vos conteneurs généraux (qui englobent votre site),

- les menus de navigation qui sont communs à toutes vos pages.

Une fois que vous avez bien copié ce qu'il faut dans le fichier header.php, le travail de dynamisation peut commencer.

Dynamisation de header.php

Prenons le template d'un fichier header classique :

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Virginia Quiroga - Graphic Design and Web</title>
<link href="reset.css" rel="stylesheet" type="text/css">
<link href="styles.css" rel="stylesheet" type="text/css">
</head>
<body>
<nav>
      <h1><a href="#">Virginia Quiroga</a></h1>
      <ul>
      <li><a href="#">Portfolio</a></li>
        <li><a href="#">Curriculum</a></li>
    </ul>
</nav>
```

Ce document en HTML5 comporte classiquement une balise `<title>`, deux liens vers des feuilles de style (la feuille de style de base et un reset), une balise `<nav>` avec un menu en `<ul>` et `<li>` tout aussi classique et, finalement, une balise `<h1>` à l'intérieur du menu `<nav>` qui reprend le nom du site.

Nous allons dynamiser ce bout de code à l'aide des ***template tags***.

Le premier template tag capital pour dynamiser notre header est :

```
<?php bloginfo(); ?>
```

Ce tag, pour être efficace, doit être accompagné de paramètres. Grâce aux paramètres que vous allez placer à l'intérieur des parenthèses, vous allez pouvoir dynamiser :

- La balise `<title>` de votre site,
- L'adresse de vos feuilles de styles,
- Le contenu de votre balise `<h1>`,
- Le menu complet de navigation de votre site.

Commençons par dynamiser la balise `<title>` de votre site. Pour le moment, elle contient le contenu suivant : `Virginia Quiroga - Graphic Design and Web` et nous allons faire en sorte que ces deux informations soient dynamiques.

Pour Wordpress, ces deux informations correspondent, dans l'interface d'administration (dans réglages/général) à « Titre du site » et à « Slogan ».

Pour afficher le titre de votre site avec Wordpress, vous devez simplement insérer le paramètre `name` :

```
<?php bloginfo('name'); ?>
```

Pour afficher le sous-titre (ou la description) de votre site avec Wordpress, vous devez simplement insérer le paramètre description :

```
<?php bloginfo('description'); ?>
```

Il vous suffit donc de remplacer vos données statiques de votre title par ces deux nouveaux bouts de code :

```
<title><?php bloginfo('name'); ?> - <?php bloginfo('description');
?></title>
```

Vous pouvez toujours séparer ces deux informations par un élément non dynamique (ici un trait d'union).

Dès que vous aurez fait ça, retournez côté administration, dans réglages/général et, si vous changez la valeur des champs « titre du site » et « slogan », votre balise `<title>` s'adaptera automatiquement, c'est magique ! :}

C'est cool, on a réussi à dynamiser le titre et le sous-titre de ma page web !

Mais il y a encore quelques petites choses à dynamiser dans mon fichier header.php : les feuilles de styles, et, éventuellement, les fichiers javascript que vous avez insérés.

Commençons par personnaliser les chemins d'accès à nos feuilles de styles.

Il faut savoir que, par défaut, pour Wordpress, il est nécessaire que votre thème possède une feuille de style qui s'appellera exactement « style.css ». Cette feuille de styles doit absolument se trouver au même endroit que vos fichiers PHP (index, footer, header, etc.).

Pour la dynamiser, il vous faut utiliser de nouveau le tempalte tag bloginfo avec un autre paramètre :

```
<?php bloginfo('stylesheet_url'); ?>
```

En indiquant ce bout de code, votre navigateur va comprendre que la feuille de style que vous allez dynamiser est bien la feuille de style qui s'appelle style.css et son chemin complet sera correct, vous pouvez donc supprimer tout ce qui est dans l'attribut href et le remplacer par ce bout de code PHP complet comme ceci :

```
<head>
<meta charset="utf-8">
<title>Virginia Quiroga - Graphic Design and Web</title>
<link href="reset.css" rel="stylesheet" type="text/css">
<link href="<?php bloginfo('stylesheet_url'); ?>" rel="stylesheet"
type="text/css">
</head>
```

C'est bien, mais en faisant cela, vous n'aurez dynamisé que la feuille de style qui portait exactement le nom « style.css », il reste à dynamiser le chemin de la feuille de style qui s'occupe de faire un reset sur votre site, et cette feuille porte justement le nom « reset.css ».

Pour dynamiser son chemin, il s'agit encore d'un autre paramètre pour le template tag bloginfo :

```
<?php bloginfo('stylesheet_directory'); ?>
```

Cette commande va ressortir tout le chemin complet de l'endroit où est hébergée votre feuille de style de base « style.css » mais sans indiquer « style.css » au bout puisqu'il s'agit ici d'une « directory », donc d'un chemin.

Si votre fichier reset se trouve au même endroit que votre fichier style.css, alors vous n'aurez qu'à faire ceci :

```
<link href="="<?php bloginfo('stylesheet_directory'); ?>/reset.css"
rel="stylesheet" type="text/css">
```

Ici, bloginfo renvoie le chemin complet jusqu'à la feuille de style de base, après, il faut rajouter une « / » et directement le nom du fichier si celui-ci se trouve au même endroit que la feuille de style « style.css », mais si cette feuille s'était trouvée dans un dossier « css » il aurait fallu spécifier aussi le nom du dossier comme ceci : /css/reset.css.

Un des derniers paramètres importants du template tag « bloginfo » est le paramètre template_directory. Il vous sera utile, comme pour stylesheet_directory, pour renvoyer le chemin complet de theme jusqu'au dossier de votre thème. On l'utilise donc pour spécifier les chemins d'éventuelles feuilles javascript (.js) qui seraient liées dans le <head> de vos documents.

Par exemple :

```
<script type="text/javascript" src="<?php bloginfo('template_directory');
?>/js/jquery.js"></script>
```

Attention
Une dernière chose importante à savoir si vous désirez injecter du javascript dans votre balise <head>, c'est que pour Wordpress, vous devez l'activer avec cette instruction simple qu'il vous suffit de mettre avant la balise de fermeture de votre </head> :

```
<?php wp_head(); ?>
```

Cette remarque est aussi valable pour le footer, nous y reviendrons dans le point suivant.

9. Le fichier footer.php

Alors que le contenu du fichier header.php est souvent bien fourni, il est en revanche possible que le fichier footer.php soit un peu plus léger.

Néanmoins, il devrait toujours au moins comprendre 2 balises de fermeture (que vous avez ouvertes dans le header.php) : `</body>` et `</html>`.

Si vous avez une partie commune à toutes vos pages, alors il est évident qu'elle doit aussi figurer dans le footer.php ! Puisque ce fichier footer.php sera chargé à chaque fois en bas de chaque page où vous l'aurez appelé !

Attention
Comme pour les fichiers javascript qui doivent être liés dans le <head>, pour que le javascript qui doit être lancé dans le footer fonctionner, il faut l'activer à l'aide du template tag suivant :

```
<?php wp_footer(); ?>
```

Grâce à ce template tag correctement placé dans le fichier header.php et dans le fichier footer.php, normalement, toutes les extensions que vous installerez par la suite via l'interface de Wordpress fonctionneront correctement ! Youpie :} !

10. Le fichier sidebar.php

Le fichier sidebar.php va définir une colonne latérale que l'on pourrait appeler dans tous les templates. Elle aurait du contenu général, non spécifique à une page précisément.

En général, les sidebar accueillent quelque chose de particulier qui est inhérent à Wordpress : l'insertion de Widgets.

Les Widgets sont des petits modules (un bloc de texte, une liste de catégories, un formulaire de recherche, un calendrier, une liste des derniers articles écrits, les derniers commentaires ajoutés, …) que l'on peut insérer, ordonner et administrer à partir de backoffice de Wordpress.

C'est surtout intéressant si votre site est destiné à être transmis à quelqu'un qui ne s'y connait pas du tout en écriture de code PHP ou en theming avec Wordpress, mais pour vous ça n'a pas presque pas d'utilité car vous êtes assez bons pour pouvoir placer votre code où vous le désirez !

Nous verrons plus tard comme rendre notre thème « Widget-ready ».

11. Le fichier searchform.php

Pour personnaliser votre champ de recherche (formulaire), il vous suffit de créer un fichier searchform.php. Si vous ne le faites pas, il va reprendre le formulaire de Wordpress par défaut. Donc cette étape n'est pas du tout obligatoire.

Mais si vous désirez quand même le personnaliser parce que c'est vraiment plus joli (sisi !), alors il y a des règles à respecter :

- la balise <form> doit avoir comme paramètres action= »/ » (ou la racine du site) et method='GET'
- l'input texte de recherche doit avoir paramètre id='s'

On pourrait avoir un formulaire simplifié comme suit :

```
<form method="get" id="form" action="<?php bloginfo('url'); ?>/">
  <input type="text" value="<?php the_search_query(); ?>" name="s" id="s">
  <input type="submit" id="submit">
</form>
```

En validant le formulaire, c'est en revanche à partir du template de page search.php qu'il va fournir le résultat !

12. Le fichier comments.php

Le fichier comments.php ne peut être inclus uniquement dans single.php, et plus précisément dans la boucle de single.php.

Il a 2 rôles :

- Afficher les commentaires avec pour chacun l'auteur, l'avatar, la date et le contenu. Les commentaires peuvent être imbriqués, c'est-à-dire avoir plusieurs niveaux de réponse.
- Afficher le formulaire de nouveau commentaire avec le nom, l'email et le message.

Le fichier comments.php est le plus dur à modifier dans un thème WordPress. C'est pourquoi je pars toujours du fichier présent dans le thème par défaut (Twenty Ten par exemple) et le modifie à partir de là. C'est le seul auquel j'applique ce processus.

La première difficulté réside dans les possibilités d'affichage des commentaires :

- Si le Post est protégé par mot de passe, il n'affiche pas de commentaire, ni de formulaire.
- S'il n'est pas protégé, il regarde si les commentaires sont autorisés.
- Si oui, il regarde si il y a des commentaires à afficher.
- Si leur nombre dépasse le nombre / page autorisé, il affiche la pagination.
- Enfin, il affiche le formulaire de nouveau commentaire si les commentaires sont ouverts.

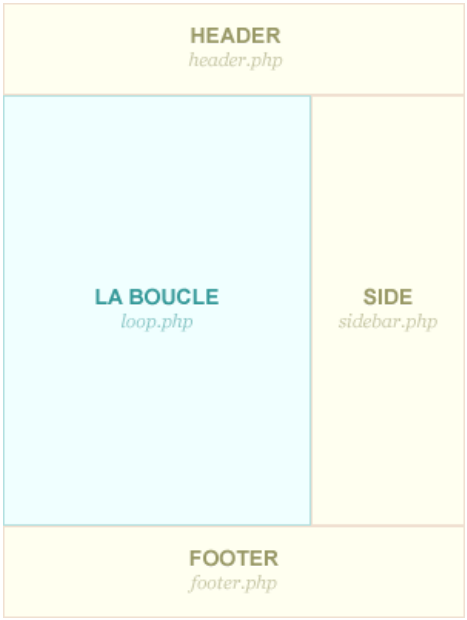
Avec toutes ces conditions, il est facile de se perdre dans les méandres de comments.php…

Une deuxième difficulté, c'est la customisation du code. En effet, depuis WordPress 3, l'affichage des commentaires se fait en appelant uniquement la fonction wp_list_comments(). Du coup, le code HTML généré n'est pas modifiable… enfin, pas facilement.

13. La boucle Wordpress

La boucle, ou son nom original « The Loop », est le noyau dur de WordPress. Il s'agit d'une simple boucle PHP par laquelle WordPress va passer pour afficher chaque Post. Depuis WordPress 3, cette boucle est généralement définie à part, dans un fichier loop.php. Ce n'est pas obligatoire mais pratique.

Voici un schéma de la page d'accueil, avec les includes et la boucle :



Alors que les fichiers header.php, sidebar.php et footer.php ne s'afficheront qu'une seule et unique fois, le fichier loop.php, lui s'affichera autant de fois qu'il y a de posts à afficher.

Par défaut, le nombre d'articles affichés au maximum par la boucle est réglé sur 10, une valeur que vous pouvez modifier dans les paramètres de votre Wordpress (dans réglages : lecture) en vis-à-vis de « Les pages du site doivent afficher au plus … », mais rassurez-vous, vous pouvez toujours accéder aux articles suivants avec un système d'archives des pages (mais il faut placer un p'tit bout de code pour cela).

Attention cette règle (qui affiche x post sur la page) n'est valable que pour la page index.php.

En effet, pour les pages single.php et page.php, par exemple, un seul et unique post ou une seule et unique page ne seront affichés à la fois.

La structure de la boucle Wordpress est assez simple, voyons d'abord sa syntaxe :

```
<?php if (have_posts()) : ?>
  <!-- Si j'ai des Posts, j'affiche cette partie -->
  <?php while (have_posts()) : the_post(); ?>
    <!-- Pour CHAQUE Post, j'affiche ça -->
  <?php endwhile; ?>
<?php else : ?>
  <!-- Si il n'y a pas de Post, j'affiche cette partie -->
<?php endif; ?>
```

Avec les commentaires, cela devrait déjà être suffisamment éloquent pour vous.

On a donc 3 parties :

- Une qui sert à afficher les posts et toute la structure d'un post s'il y a des posts (if (have posts()),

- Une qui sert à afficher le code qui sera répété pour chaque post,

- Une qui sert à afficher quelque chose s'il n'y a pas de post.

Les template tags de la boucle Wordpress

Plusieurs template tags vous seront utiles pour ressortir les informations de chaque post que vous allez créer. Ces tags, les voici :

`<?php the_title(); ?>`	Affiche le titre du post

<?php the_excerpt(); ?>	Affiche un résumé du post
<?php the_content(); ?>	Affiche tout le contenu du post
<?php the_thumbnail(); ?>	Affiche l'image à la une du post
<?php the_category(); ?>	Affiche la/les catégorie(s) choisie(s) pour le post
<?php the_tags(); ?>	Affiche le(s) mot(s) clé choisi(s) pour le post
<?php the_date(); ?>	Affiche le jour auquel le post a été créé
<?php the_time(); ?>	Affiche l'heure à laquelle le post a été créé
<?php the_author(); ?>	Affiche qui a créé le post
<?php the_permalink(); ?>	Crée un lien vers le post complet

Attention
Parmi les tags ci-dessus, chacun renvoie un peu le résultat qui a été prévu par Wordpress.Par exemple, the_title renvoie du texte brut, tandis que the_category renvoie un lien cliquable menant vers tous les posts de cette catégorie, the_tags renvoie une suite de liens qui sont tous cliquables et qui mènent vers une liste des posts liés à ces mots, etc.
Soyez donc vigilants quant à l'utilisation des template tags ci-dessus !

14. Le fichier index.php

Maintenant que nous avons définis les includes les plus importants (header, loop, sidebar et footer), utilisons-les dans 1 template : index.php.

Ce fichier est celui qui sera lancé en toutes circonstances si Wordpress ne trouve pas un template de fichier quand vous cliquez sur un lien (pour autant qu'il n'y ait pas de fichier 404.php – qu'il est possible de créer aussi ☺).

Il existe un autre fichier qui est dédié à la homepage de votre site web, il s'agit de home.php. Je l'utilise personnellement peu, mais c'est une question d'habitude.

Il vous reste donc maintenant à appeler chaque portion de code que vous avez isolée pour reformer votre code complet à l'intérieur du fichier index.php.

Pour ce faire, il va vous falloir utiliser la fonction d'include créée pour Wordpress :

- `<?php get_header(); ?>` : pour appeler le fichier header.php
- `<?php get_footer(); ?>` : pour appeler le fichier footer.php
- `<?php get_sidebar(); ?>` : pour appeler le fichier sidebar.php

Attention
il est important de les appeler dans le bon ordre : celui qui correspond exactement à votre structure de page avant sa découpe !

Évidemment, entre chaque include, vous pouvez ajouter des portions de code qui vont faire appel à la boucle de Wordpress et ses template tags (pour faire revenir des informations d'un type de contenu structurées d'une certaine façon).

Le fichier index.php est aussi « simple » que ça et c'est souvent à partir de sa structure que vous allez pouvoir décliner vos autres templates de page, car dans chaque page il est plutôt probable qu'on retrouve au moins le header et le footer que vous avez

isolés ☺. Mais retenez quand même que ce n'est pas forcément obligatoire, tout dépend de votre façon de découper votre code original !

15. Le fichier single.php

Il est important de comprendre quand est appelé le fichier single.php. Pour y parvenir, n'hésitez pas à consulter à nouveau le diagramme inséré quelques pages plus haut, il est capital de bien le comprendre.

Single.php, c'est le template pour la représentation d'un article de type « post » (et pas « page ») de base. Nous verrons plus tard qu'on peut le décliner pour les types de « posts » personnalisés également.

Ça veut simplement dire que c'est à partir de la structure de ce fichier que Wordpress va puiser pour afficher à vos utilisateurs la représentation d'un article et de son contenu à partir de n'importe quel lien qui pointe vers cet article sur votre site.

Ces articles sont accessibles, côté back-office, sous l'appellation « articles » en haut de la colonne de gauche, c'est un peu le « cœur » de Wordpress.

Le fichier single.php est donc utile pour les « posts » de base alors que le fichier page.php sera utile pour les « pages » de base de Wordpress.

Avec ces deux template-là, vous avez déjà assez pour produire un site de petite taille avec peu de types de contenus différents, mais si vous désirez aller plus loin, il vous faudra maitriser aussi les « types de contenu » que nous découvrirons dans un chapitre ultérieur.

Attention
Pour faire apparaître les informations d'un article à l'intérieur du fichier single.php, il est capital d'entourer les informations dynamiques (titre, contenu, tags, …) d'une boucle, sinon ces informations ne s'affichent pas.

16. Le fichier page.php

Il fonctionne exactement de la même façon que le fichier single.php mais lui il s'adresse à toutes les pages de base que vous allez créer à partir de l'onglet « pages » dans Wordpress.

C'est-à-dire que quand vous allez cliquer sur un lien qui mène vers une page de votre site, c'est à partir du fichier page.php que la structure est puisée et appliquée.

Pour le reste, comme pour le fichier single.php, il faut bien entourer d'une boucle les informations dynamique de vos pages (titre, contenu, image, …) pour les voir apparaître.

17. Le fichier archive.php

C'est simplement le fichier de template qui sera choisi par Wordpress quand vous cliquez sur un de ces éléments sur votre site :

- Un mot clé
- Une catégorie de votre site
- L'auteur d'un article
- La date d'un article

18. Le fichier functions.php

C'est probablement le fichier le plus important de Wordpress mais c'est aussi le fichier le moins « sexy » de Wordpress car il ne va à priori comporter que du code PHP.

Il comporte toutes les fonctions (et surtout tous les paramètres des fonctions) nécessaires au bon fonctionnement de certaines fonctionnalités de Wordpress :

- Image à la une
- Menu dynamique
- Widgets
- Pagination
- Nouveau type de contenu
- Nouveau type de taxonomie
- etc.

Il est souvent assez fourni et donc il est plus que conseillé de bien le documenter et le commenter pour vous y retrouver et savoir ce que fait chaque bout de code que vous aurez inscrit dans le fichier functions.php.

Voici déjà quelques portions de codes qui pourraient vous aider :

```
add_theme_support( 'post-thumbnails' ); // Activation des images à la une

add_theme_support('nav-menus'); // Activation des menus

if ( function_exists( 'add_image_size' ) ) {
      add_image_size( 'image-homepage', 380, 240, true );
} // Création d'un nouveau format d'image et création de son nom
```

Vous trouverez beaucoup d'autres fonctions sur le web ou même des générateurs de fichiers function.php (http://www.wpfunction.me/).

19. Les boucles modifiées wp_query et query_posts

En dehors de la boucle de base explicitée dans un précédent chapitre, nous allons voir ici une variante de cette boucle grâce à laquelle vous allez avoir beaucoup plus de souplesse dans ce que vous voulez afficher.

Cette boucle, elle utilise la fonction wp_query. Cette fonction va vous servir dans beaucoup de circonstances (à priori elle pourrait même vous servir dans tous les cas ☺) de la même manière que query_posts.

```
<?php
$query = new WP_Query($args);
if($query->have_posts()) : while($query->have_posts()) : $query-
>the_post();
?>

Votre contenu à dynamiser

<?php endwhile; endif; ?>
```

Voilà la structure de base d'une boucle utilisant wp_query. Rien de fort différent, la grande nuance se situe en fait dans le remplacement des paramètres à passer à la place de $args.

Et là, ça tombe bien, vous avez beaucoup de choix pour effectuer des requêtes précises.

Par exemple ceci :

```
$query = new WP_Query(array('post_type' => 'post', 'posts_per_page' => 1));
```

Va définir une boucle qui va chercher le type de contenu « post » (articles de base) et n'affiche qu'un seul de ces articles à cet endroit-là.

Avec une autre requête on peut aussi aller chercher les articles et les pages en une seule fois (ou plusieurs types de post types par exemple) :

```
<?php $query = new WP_Query(array('post_type' => array('post', 'page')));
?>
```

Récupérer les 4 derniers articles d'une ou plusieurs catégories :

```
<?php $query = new WP_Query(array('posts_per_page' => 4, 'category_name' =>
'nom-de-votre-categorie')) ; ?>
```

Récupérer le custom post type « film » trié par titre dans l'ordre décroissant en n'affichant pas les 3 premiers résultats :

```
<?php $query = new WP_Query(array('post_type' => 'film', 'orderby' =>
'title', 'order' => 'DESC', 'offset' => '3')); ?>
```

Des centaines d'autres possibilités sont dispo ici :
http://codex.wordpress.org/Class_Reference/WP_Query

Et concernant la boucle query_posts ça se passe ici (elle fonctionne globalement comme wp_query mais utilise les fonctions de Wordpress donc elle pourrait vous être utile dans certains cas) :
http://codex.wordpress.org/Function_Reference/query_posts

N'hésitez pas à abuser de ces boucles ultra-pratiques où vous le désirez !

20. Créer un nouveau type de contenu et une nouvelle taxonomie

Si on devait s'en tenir aux 2 types de contenus offerts par Wordpress, on serait quand même assez limités dans nos conceptions de thèmes. C'est pourquoi, depuis assez peu de temps, Wordpress propose à ses utilisateurs de pouvoir créer de nouveaux types de contenu.

En plus des articles et des pages de base, vous allez pouvoir créer des types de contenu particuliers qui pourront répondre directement à vos besoins.

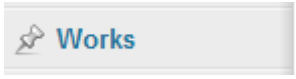
Et en plus de créer des types de contenus, vous allez aussi pouvoir leur attribuer des catégories et/ou des mots clés.

Pour déclarer un nouveau type de contenu, il n'y a rien de plus simple. Il suffit de vous rendre dans votre fichier functions.php et d'y coller le code suivant que vous allez adapter :

```
add_action('init', 'my_custom_init');
function my_custom_init()
{
register_post_type('works', array(
  'label' => __('Works'),
  'singular label' =>   ('Work'),
  'public' => true,
  'show_ui' => true,
  'capability_type' => 'post',
  'hierarchical' => false,
  'supports' => array('title', 'excerpt', 'thumbnail')
));}
```

- La fonction register_post_type sert ici à déclarer un nouveau type de contenu avec toute une série de paramètres que je vous propose de découvrir ci-dessous.
- Works correspond au nom système que vous donnez à votre nouveau type de contenu
- Label est le nom qu'on retrouvera côté administration dans la barre du menu de gauche
- Singular_label est l'équivalent du label mais au singulier
- Public définit si ce nouveau type de contenu doit être publié ou pas
- Show_ui fait apparaître l'item works dans le menu si sa valeur est réglée sur true
- Capability_type définit si ce type de contenu est plutôt un « post » ou une « page » (au niveau structurel)
- Hierarchical définit s'il peut y avoir une hiérarchie ou pas entre les éléments
- Supports est une table qui définit quels éléments d'un post on va avoir côté administration. Voici ce que vous pouvez décider d'afficher :
    - Title : boite pour entrer le titre du contenu
    - Editor : fait apparaitre la boite d'édition du contenu de votre type de contenu
    - Author
    - Thumbnail : permet de mettre une image à la une
    - Excerpt : une boite pour un résumé de votre contenu
    - Etc. (les autres sont moins importants mais vous pouvez les retrouver ici : http://codex.wordpress.org/Function_Reference/post_type_supports)

Dès que vous aurez sauvegardé votre fichier functions.php, une entrée « Works » apparaîtra dans votre interface d'administration.



Avec la possibilité d'ajouter de nouveaux « works » comme pour les articles classiques.

Pour faire apparaître du contenu issu de ce nouveau post-type à n'importe quel endroit dans Wordpress, il suffit d'indiquer dans votre boucle qu'elle doit aller chercher le contenu de post-type « works » (nom sysème – càd sans majuscule, espace blanc, accentuation) et de l'afficher avec plus ou moins de paramètres. Vous pouvez le faire avec la boucle wp_query ou avec query_posts :

Avec wp_query :

```
<?php
$query = new WP_Query(array('post_type' => 'works', 'posts_per_page' =>
1));
if($query->have_posts()) : while($query->have_posts()) : $query-
>the_post();
?>

Votre contenu à dynamiser issu du type de contenu « works ».

<?php endwhile; endif; ?>
```

Avec query_posts :

```
<?php wp_reset_postdata(); ?>
  <?php query_posts('posts_per_page=-1&post_type=works'); ?>
  <?php if (have_posts()) : ?>
    <?php while (have_posts()) : the_post(); ?>

      Votre contenu à dynamiser issu du type de contenu « works ».

    <?php endwhile; ?>
  <?php endif; ?>
```

Pour ce nouveau type de contenu « works », nous allons voir également comment lui attacher des mots-clés et/ou des catégories.

Ça se passe de nouveau du côté du fichier functions.php.

Juste après la déclaration du nouveau type de contenu, vous allez ajouter ces 2 lignes :

```
register_taxonomy( 'type', 'works', array( 'hierarchical' => true, 'label'
=> 'Type', 'query_var' => true, 'rewrite' => true ) );

register_taxonomy( 'couleur', 'works', array( 'hierarchical' => false,
'label' => 'Couleur', 'query_var' => true, 'rewrite' => true ) );
```

La première ligne va déclarer un type de taxonomie :

- Qui aura le nom « type » (de quel genre de travail s'agit-il ..)
- Qui concerne le type de contenu « works »
- Hierarchical réglé sur « true » veut dire qu'il s'agit d'un type de taxonomie « catégories » qui peut avoir des sous-catégories etc.
- Query_var sert à pouvoir utiliser cette taxonomie dans nos templates
- Rewrite c'est la chaîne de caractères présente dans les permaliens. Si on met "true", ça prend la valeur par défaut, c'est à dire le nom de la taxonomie.

La deuxième ligne va déclarer un type de taxonomie :

- Qui aura le nom « couleur » (Possibilité de classer les travaux par couleur)
- Qui concerne le type de contenu « works »
- Hierarchical réglé sur « false » veut dire qu'il s'agit d'un type de taxonomie « mots-clé » qui ne peut pas avoir de sous-catégorie.
- Query_var sert à pouvoir utiliser cette taxonomie dans nos templates
- Rewrite c'est la chaîne de caractères présente dans les permaliens. Si on met "true", ça prend la valeur par défaut, c'est à dire le nom de la taxonomie.

Avec ces 2 lignes de déclaration, vous verrez directement apparaître, en sous-menu du nouveau type de contenu que vous avez créé, le nom des 2 taxonomies créées et, surtout, quand vous créerez un « work », vous allez pouvoir le mettre dans une catégorie (type) et lui attribuer des mots-clé (couleur). C'est trop cool non ? ☺

Pour afficher ces éléments taxonomiques sur une page, voici comment faire (à l'intérieur d'une boucle évidemment) :

```
<?php echo get_the_term_list( $post->ID, 'type', '<p>Type de travail : ',
', ', '</p>' ) ?>
<?php echo get_the_term_list( $post->ID, 'couleur', '<p>Couleurs : ', ', ',
'</p>' ) ?>
```

C'est aussi simple que ça ☺ Vous indiquez par son nom système le type de taxonomie que vous voulez afficher et vous précédez cet élément de n'importe quelle portion de code html désirée.

21. Créer un nouveau Template de page pour vos pages personnalisées

Les templates de page sont super-importants et très facile à prendre en main !

Ils vont, entre autre, vous permettre de créer des pages où vous pouvez mettre le code PHP que vous voulez, sans vous soucier de rien du tout. Ce code sera affiché à travers une page (que vous devrez donc créer dans l'interface de Wordpress, sous « page »). Il vous suffit ensuite de dire que la page que vous venez de créer doit puiser sa structure dans le fichier de template PHP.

3 étapes donc pour créer un template de page :

- Créer un template de page
- Créer une page dans l'admin Wordpress
- Lier la page créée dans Wordpress au template que vous avez créé

Un template de page, c'est juste un fichier PHP que vous allez mettre au même niveau que vos autres fichiers PHP. On peut lui donner le nom qu'on veut pour autant qu'il ne s'agisse pas d'un des noms pris par Wordpress (pas de page.php, post.php ou autre chose du genre, utilisez plutôt des noms très précis du genre works.php par exemple).

La première chose à faire est donc de créer un fichier PHP.

Ce fichier doit absolument commencer par ce bout de code que vous allez adapter :

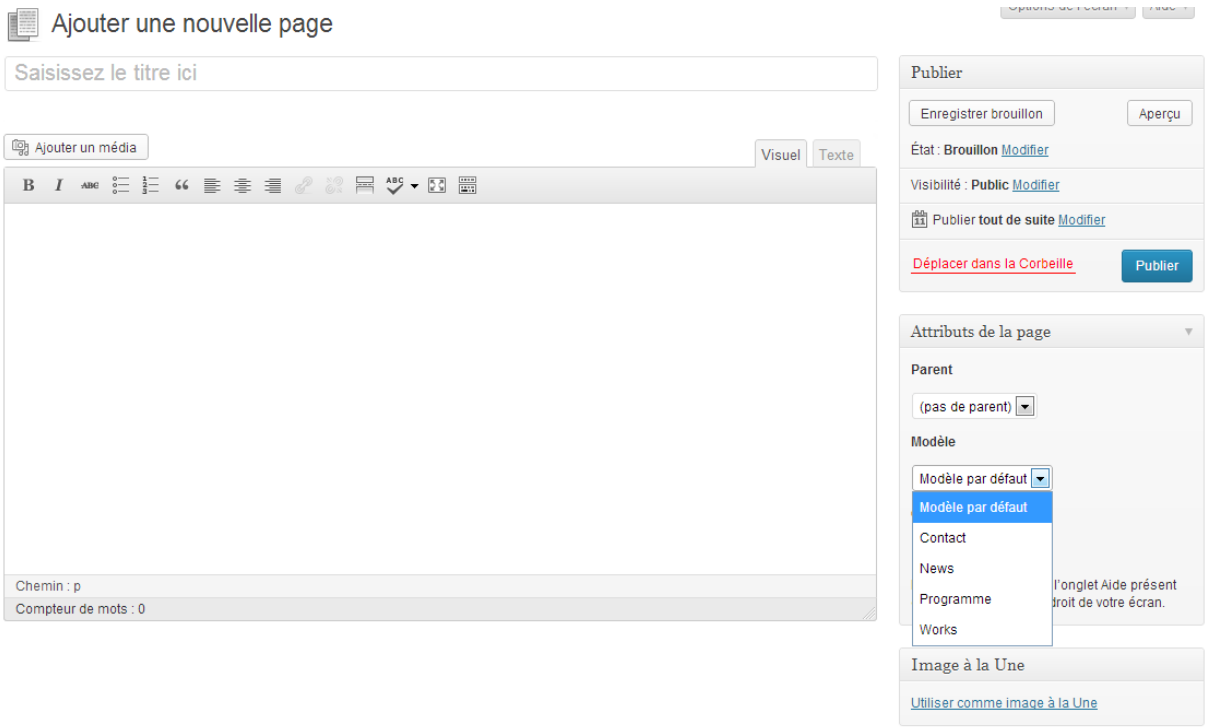
```
<?php
/*
Template Name: Works
*/
?>
```

Cette portion de code qui introduit votre fichier de template sert à lui donner un nom qui sera celui présent lors de la liaison de votre page, dans l'interface de Wordpress.

Ensuite, vous mettez tout le code Html/PHP nécessaire pour avoir le résultat que vous voulez (vous pouvez appeler à nouveau votre header/footer/sidebar etc.).

La deuxième étape consiste à créer une page dans l'admin de Wordpress. Page que je vous conseille d'appeler de la même façon que votre fichier PHP (pour la cohérence).

Ensuite, dans la boite « Attributs de la page », dans la colonne de droite, vous allez sélectionner sous « modèle » le nom de modèle à appliquer sur cette nouvelle page statique. Dans notre exemple, il y a 4 modèles de page différent, on choisira « works ».



Et voilà, c'est tout ! Ces template de page sont capitaux et très utiles, vous vous en rendrez compte assez vite !

22. Conclusion

Avec ce petit syllabus, nous avons parcourus une belle partie des fonctionnalités de base et avancées de Wordpress.

Mais Wordpress permet encore bien d'autres choses, et c'est en allant dans le détail et étant confronté à des problèmes que vous trouverez des solutions parfaites pour tous vos projets Wordpress.

Gardez en tête que Wordpress évolue aussi au fil de ses versions et que de nouvelles fonctionnalités vont voir le jour dans les mois à venir.

Finissons ce cours en énumérant certaines extensions intéressantes à installer pour Wordpress :

- Types (pour ajouter des champs personnalisés)
- Contact Form 7 (pour vos formulaires)
- Google XML sitemap (pour le référencement des pages de votre site)
- Menu editor (pour éditer le menu de votre page d'administration)
- Wp Page Navi (pour faciliter la gestion de la pagination sous WP)
- Crop Thumbnails (pour avoir des crop de vos images exactement aux bonnes dimensions)