PROPULSEZ VOTRE SITE AVEC

WORDPRESS

Julien Chichignoud

OPENCLASSROOMS

DANS LA MÊME COLLECTION

Développez votre site web avec le framework Symfony 2

Alexandre Bacco

ISBN : 979-10-90085-42-8

Améliorez la visibilité de votre site grâce au référencement

Nassim Kebbani

ISBN : 979-10-90085-46-6

Apprenez le fonctionnement des réseaux TCP/IP

Eric Lalitte

ISBN : 979-10-90085-47-3

Structurez vos données avec XML

Ludovic Roland

ISBN : 979-10-90085-56-5

Programmez en ACTIONSCRIPT 3

Guillaume Chau & Guillaume Lapayre

ISBN : 979-10-90085-55-8

Rejoignez la communauté OpenClassrooms :

www.openclassrooms.com **www.facebook.com/openclassrooms** **@OpenClassrooms**

PROPULSEZ VOTRE SITE AVEC

WORDPRESS

Julien Chichignoud

OpenClassrooms 2014 - ISBN : 979-10-90085-74-9

Mentions légales :
Conception couverture : Amalgam Impression
Illustrations chapitres : Amalgam Impression

Avant-propos

Avec l'avènement d'Internet et du Web 2.0 depuis plusieurs années, de plus en plus d'utilisateurs décident de se lancer dans l'aventure de la création d'un site Internet personnel ou professionnel. Cette activité est accessible à tous, mais attention, il est important de partir avec les bons outils !

WordPress a été créé pour aider des utilisateurs sans connaissance technique particulière à mettre en place un blog à moindre coût, tout en ayant un aspect et une utilisation professionnels. Il est facile d'accès, polyvalent, extensible grâce à de nombreux modules. De plus, WordPress est fiable puisque que le logiciel existe depuis plus de dix ans. En dépit de la croyance populaire, WordPress n'est pas dédié qu'à la création de blogs, même si c'est là son orientation principale. Il est tout à fait possible de réaliser un site vitrine pour votre entreprise par exemple. L'outil est parfaitement adapté à cette utilisation.

Cette accessibilité se traduit par la simplicité d'ajout de nouvelles fonctionnalités pour les développeurs web. WordPress est construit de façon à pouvoir être personnalisé. Ainsi chaque site peut être complètement différent d'un autre et proposer son contenu sous une forme très éloignée de son voisin.

Si vous cherchez à créer votre site en un rien de temps, il y a de grandes chances que WordPress soit l'outil qu'il vous faut !

Que vous soyez un nouvel arrivant dans l'univers de WordPress, un utilisateur averti désirant aller plus loin ou bien tout simplement en soif de connaissance sur cet outil, ce livre est fait pour vous. La première partie de ce livre ne demande aucun prérequis particulier. Des connaissances basiques avec le langage de programmation PHP vous aideront à aborder sereinement les chapitres suivants.

Qu'allez-vous apprendre en lisant ce livre ?

Ce livre est composé de trois parties abordant chacune un aspect différent de Wordpress, afin de faire évoluer vos compétences au fil de votre lecture.

1. **Prendre en main WordPress :** entrez dans l'univers de WordPress facilement avec les premiers chapitres qui vous présenteront les fonctionnalités qui s'offrent à vous en tant que webmaster. Nous écrirons des articles, apprendrons à gérer

les contributions d'autres utilisateurs, puis nous personaliserons votre blog avec une nouvelle apparence et des modules créées par la communauté pour que ce site s'adapte à vos besoins.

2. **Développer votre thème :** la seconde partie du cours consiste à creuser le fonctionnement interne de WordPress pour que vous soyez capable de créer le thème que vous souhaitez sur votre site. Nous serons donc appelés à écrire du code PHP pour adapter WordPress à nos besoins et obtenir un site à votre image.
3. **Développer un plugin complet :** dans un troisième temps, nous verrons pas à pas comment développer un plugin à installer sous WordPress. Les plugins fournissent aussi bien des fonctionnalités d'affichage spécifiques à vos visiteurs, que de nouvelles options pour l'administrateur du site. À la fin de cette partie, vous serez capable de modifier ou de créer quasiment n'importe quel fonctionnement dans WordPress.
4. **Exploiter votre site :** une fois que votre site est prêt à recevoir ses premiers visiteurs, il reste une chose importante à faire : le mettre en ligne sur Internet. Dans cette partie, vous apprendrez comment vous y prendre pour que votre site soit visible sur la toile et pour que vos visiteurs soient toujours plus nombreux.

Comment lire ce livre ?

Suivez l'ordre des chapitres

Lisez ce cours comme on lit un roman. Il a été conçu pour cela.

Contrairement à beaucoup de livres techniques où il est courant de lire en diagonale et de sauter certains chapitres, il est ici très fortement recommandé de suivre l'ordre du cours, à moins que vous ne soyez déjà un peu expérimentés.

Pratiquez en même temps

Pratiquez régulièrement. N'attendez pas d'avoir fini de lire ce livre pour allumer votre ordinateur et faire vos propres essais.

Utilisez les codes web !

Afin de tirer parti d'OpenClassrooms dont ce livre est issu, celui-ci vous propose ce qu'on appelle des « codes web ». Ce sont des codes à six chiffres à saisir sur une page d'OpenClassrooms pour être automatiquement redirigé vers un site web sans avoir à en recopier l'adresse.

Pour utiliser les codes web, rendez-vous sur la page suivante :

`http://www.openclassrooms.com/codeweb`

Un formulaire vous invite à rentrer votre code web. Faites un premier essai avec le code ci-dessous :

▷ Tester le code web
Code web : 123456

Ces codes web ont deux intérêts :

— ils vous redirigent vers les sites web présentés tout au long du cours, vous permettant ainsi d'obtenir les logiciels dans leur toute dernière version ;
— ils vous permettent d'afficher les codes sources inclus dans ce livre, ce qui vous évitera d'avoir à recopier certains programmes un peu longs.

Ce système de redirection nous permet de tenir à jour le livre que vous avez entre les mains sans que vous ayez besoin d'acheter systématiquement chaque nouvelle édition. Si un site web change d'adresse, nous modifierons la redirection mais le code web à utiliser restera le même. Si un site web disparaît, nous vous redirigerons vers une page d'OpenClassrooms expliquant ce qui s'est passé et vous proposant une alternative.

En clair, c'est un moyen de nous assurer de la pérennité de cet ouvrage sans que vous ayez à faire quoi que ce soit !

Remerciements

Je tiens à adresser mes chaleureux remerciements aux personnes qui ont rendu possible la création de ce livre.

— Nazli Isikli et Jonathan Baudouin pour m'avoir accompagné au fil de la réalisation de cet ouvrage.
— Mes amis et proches qui ont participé à la relecture en me donnant leurs précieux avis.
— L'équipe d'OpenClassrooms, qui développe et maintient le site.
— Vous, lecteurs, qui êtes la raison d'être de ces pages.

Table des matières

Première partie

Prendre en main WordPress

Chapitre 1

Découvrir WordPress

Difficulté :

Ce premier chapitre introductif sera l'occasion de découvrir ce qu'est WordPress et ce qu'il nous permet de réaliser. Nous en profiterons aussi pour l'installer et faire un rapide survol des fonctionnalités dans son interface d'administration.

Présentation

Introduction

WordPress est un logiciel libre principalement spécialisé dans la création de blogs, dont la réalisation est très facile. C'est cependant loin d'être sa seule possibilité et rien ne vous empêchera de créer un site différent.

Ainsi, il est bon à savoir que WordPress est un CMS (Content Management System ou Système de Gestion de Contenu en français), c'est-à-dire qu'il permet à l'utilisateur (c'est-à-dire l'administrateur du site) de créer facilement des pages de contenu, comme par exemple :

— la page de présentation d'une entreprise ;
— des articles de blog ;
— une page de contact ;
— un portfolio pour un artiste ;
— bien encore des fiches produits sur un site de vente en ligne ;
— etc.

L'idée d'un CMS est de donner la possibilité de facilement créer du contenu sur le site, sans avoir à mettre les mains dans le code ni même avoir de connaissances techniques particulières.

Historique

WordPress est sorti pour la première fois en 2003 comme un projet dérivé de cafelog, lui-même étant un moteur de blog sorti en 2001. De nombreuses versions sont sorties depuis, chacune apportant son lot de nouveautés par rapports aux versions précédentes, comme les plugins, les widgets, les thèmes, une interface utilisateur améliorée... La version 3 est sortie en 2010 en apportant notamment la gestion des menus, de nouvelles fonctions pour gérer l'en-tête du site ainsi que le support du *multisite* (c'est ce qui permet d'avoir plusieurs sites sur une même instance de WordPress).

Fonctionnalités

WordPress est écrit en PHP, un langage de programmation spécialisé dans la création de sites Internet. Ce langage permet donc aux développeurs de rajouter des fonctionnalités qui pourront être réutilisées par d'autres utilisateurs. Il est donc facile à modifier si vous avez de bonnes bases dans ce langage.

L'une des grandes forces de WordPress est ainsi la multitude de plugins disponibles, développés par la communauté. Ce sont des modules permettant d'ajouter des fonctionnalités à WordPress, comme par exemple la création d'une galerie photo ou la gestion d'une newsletter. Il y en a aujourd'hui plus de 25 000 plugins sur le site officiel, wordpress.org. Il y a donc fort à parier que, si vous cherchez une fonctionnalité supplémentaire pour votre site, un plugin existe déjà pour cela ! Si cela n'était pas le cas,

vous pouvez bien entendu développer le votre et éventuellement le publier.

La renommée de WordPress dans le monde du blogging est telle qu'il existe même un site dédié, que vous retrouverez sur le code web suivant :

▷ Le site wordpress.com
Code web : 252527

À ne pas confondre avec le site officiel WordPress qui se termine en .org. Il vous permet de créer votre blog WordPress sans vous occuper de son hébergement ou de son installation. Tout est géré automatiquement, vous n'avez qu'à créer votre contenu. Vous n'avez en revanche pas directement la main sur le code et il sera nécessaire de passer par l'installation de thèmes ou de plugins pour faire des modifications fonctionnelles.

Mise en place du serveur web

Qu'est-ce qu'un serveur ?

Pour fonctionner, tout site Internet s'appuie sur un serveur qui se charge d'envoyer les pages à un visiteur qui en fait la demande. Le serveur est un programme qui exécute le code du site web (écrit en PHP dans le cas de WordPress) pour générer et envoyer la page au format HTML, lisible par un navigateur Web (Safari, Google Chrome, Mozilla Firefox, etc.). Pour utiliser WordPress, vous devrez donc obligatoirement en utiliser un ! Le serveur le plus utilisé pour PHP s'appelle Apache, il est totalement gratuit et peut fonctionner sur n'importe quel environnement (Windows, Linux ou Mac), mais il en existe d'autres, parfois payants.

De plus, le serveur web fonctionne le plus souvent conjointement à une base de données, qui permet de stocker les informations spécifiques à votre site : le contenu des pages, les utilisateurs inscrits, la configuration... Tout ce qui change d'un site à l'autre est potentiellement stocké à l'intérieur sous la forme de plusieurs tableaux de données. MySQL est un système de base de données très connu et répandu, car il est totalement gratuit et compatible avec une majorité d'applications, c'est pourquoi nous allons l'utiliser.

Un serveur web et la base de données peuvent s'installer sur n'importe quelle machine, même votre ordinateur personnel. Cela permet notamment de faire des tests localement, sans que votre site soit en ligne ; on parle alors **d'installation « locale »**. Il faut toujours commencer par une installation locale lorsque vous prenez en main un outil, afin de limiter la portée des erreurs que vous pourriez faire en débutant. Si vous faites des erreurs alors que votre site est en ligne, n'importe quel visiteur pourra le voir, tandis que vous êtes seul à y avoir accès tant que le site reste sur votre ordinateur personnel.

Sur Windows : WAMP

Si vous êtes sous Windows, vous pouvez utiliser WAMP. C'est un programme qui contient un serveur Apache ainsi qu'une base de données MySQL sur votre ordinateur.

Vous pouvez télécharger WAMP sur le code web suivant :

▷ Le site officiel WAMP
Code web : 659586

Ce site propose plusieurs versions différentes du logiciel. Si vous ne savez pas laquelle choisir pour votre ordinateur, vous pouvez choisir la première de la liste qui fera *a priori* parfaitement l'affaire.

Une fois le paquet téléchargé, lancez l'installation en conservant les options par défaut, ce qui devrait créer un dossier c :\wamp sur votre ordinateur. Vous pouvez vérifier que WAMP fonctionne correctement avec l'icône qui doit s'afficher dans la barre des tâches (voir la figure 1.1).

FIGURE 1.1 – Apparition de l'icône de WAMP dans la barre des tâches

Ouvrons maintenant un navigateur et tapons dans la barre d'adresse, l'url du site indiqué sur le code web suivant. Il correspond à l'adresse du serveur web présenté sur la figure 1.2).

▷ http ://localhost/
Code web : 820589

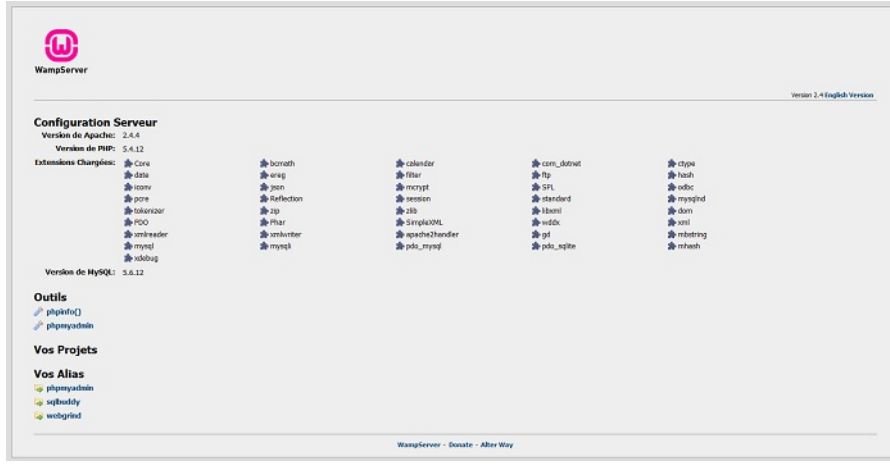


FIGURE 1.2 – La page d'accueil de WAMP

Si vous obtenez bien cette page, votre serveur web est opérationnel et prêt à faire tourner WordPress !

Sur Mac : MAMP

Pour les systèmes sous Mac OS X, il existe le logiciel MAMP, qui installe lui aussi tous les programmes qui nous seront nécessaires. Vous pouvez le télécharger directement en passant par le lien indiqué sur le code web suivant :

▷ Le logiciel MAMP
Code web : 983144

Commencez par extraire l'archive .zip puis exécutez le fichier .pkg pour lancer l'installation (voir la figure 1.3).

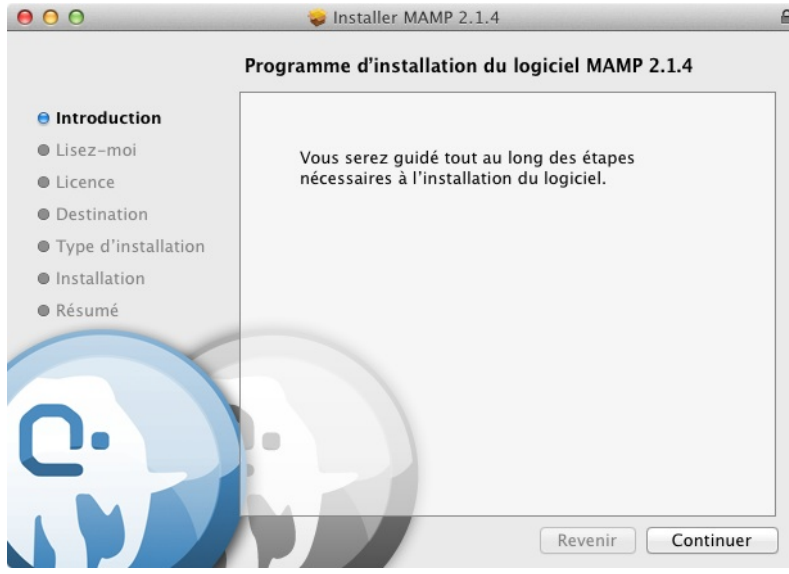


FIGURE 1.3 – L'installation MAMP sur Mac

Une fois MAMP installé, vous pouvez le lancer depuis le dossier « Applications > MAMP » de votre ordinateur. La fenêtre qui s'ouvre indique le statut des serveurs Apache et MySQL. S'il sont arrêtés, cliquez sur démarrer les serveurs pour voir le statut passer au vert (voir la figure 1.4).

Ouvrez ensuite la fenêtre « Préférences ». Dans l'onglet « Ports », cliquez sur le bouton « Ports par défaut d'Apache et de MySQL ». Puis, dans l'onglet « Apache », vous pouvez choisir le dossier dans lequel vous placerez les différents sites sur votre ordinateur (voir la figure 1.5).

Je vous conseille de choisir le dossier « Sites » qui doit déjà exister dans votre répertoire personnel

Vous pouvez valider les modifications, puis vous rendre sur la page indiquée sur le code web suivant :

▷ http ://localhost/MAMP/
Code web : 220398

FIGURE 1.4 – Démarrage des serveurs

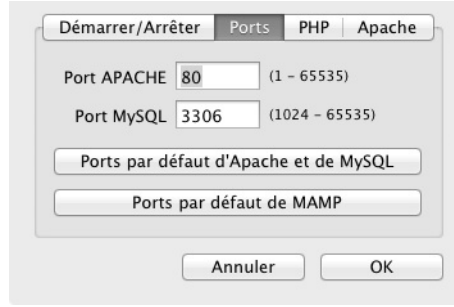


FIGURE 1.5 – Les ports d'Apache et MySQL sont mis à jour

La page d'accueil MAMP proposée sur la figure 1.6, vous permet de afin de vérifier le bon fonctionnement du serveur.

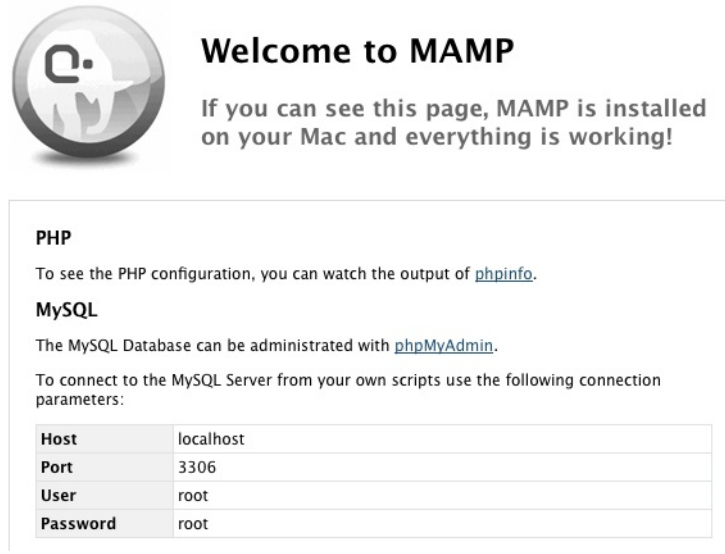


FIGURE 1.6 – La page d'accueil de MAMP

Sur Linux : LAMP

Si vous utilisez un système Linux comme Ubuntu ou Debian, le plus simple est d'installer les différents paquets contenant Apache, PHP et MySQL. Lancez donc la commande suivante dans une console :

```
sudo apt-get install apache2 php5 mysql-server libapache2-mod-
    php5 php5-mysql
```

L'ensemble des paquets doit s'installer sans problème. Lors de l'installation de MySQL, la console vous demandera d'entrer un mot de passe dont vous devrez vous souvenir pour vous connecter à la base de données, ne l'oubliez pas ! Installez ensuite phpMyAdmin avec la ligne de commande suivante.

```
sudo apt-get install phpmyadmin
```

L'invite de commande vous demandera dans un premier temps de choisir le serveur à configurer pour phpMyAdmin, choisissez Apache, puis validez la création d'une base de données à l'étape suivante. Enfin, vous devrez entrer le mot de passe choisi à l'installation de MySQL pour que le paquet puisse créer sa propre base de données.

Une fois l'installation terminée, rendez-vous à l'adresse indiquée sur le code web suivant :

▷ http ://localhost/
Code web : 820589

Une page de confirmation, comme la figure 1.7, doit s'afficher.

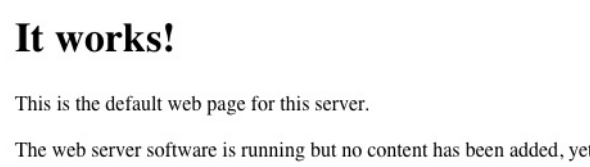


FIGURE 1.7 – La page d'accueil sur Linux

Installation

Cette partie vous concerne si vous souhaitez installer WordPress sur un environnement local (après avoir installé un serveur Apache sur votre ordinateur) ou bien sur un serveur dédié chez un hébergeur, ce qui vous permettrait de mettre votre site en ligne en gérant votre serveur seul. Si vous choisissez de créer votre blog sur le site officiel de WordPress, tout ce que vous aurez à faire est de vous inscrire, et tout sera installé automatiquement.

Création de la base de données

Avant de continuer, nous allons devoir créer la base de données pour que WordPress puisse stocker les informations spécifiques à notre site. Pour cela, nous devons accéder à l'interface de phpMyAdmin à l'adresse indiquée sur le code web suivant :

▷ L'interface de phpMyAdmin
Code web : 206573

Si vous utilisez MAMP, rendez-vous à l'adresse indiquée sur ce code web :

▷ L'interface de phpMyAdmin via MAMP
Code web : 792898

Identifiez-vous avec le login root et un mot de passe vide (si cela ne fonctionne pas, le mot de passe est probablement « root », notamment sur MAMP) afin d'arriver sur l'interface de gestion de la base de données. Cliquez en haut de la page sur l'onglet « Base de données », puis choisissez un nom de base de données (par exemple « WordPress ») sur la page qui s'affiche (voir la figure 1.8).

Validez par un clic sur « Créer », puis passez à l'étape suivante.

FIGURE 1.8 – Créer la base de données dans phpMyAdmin

Téléchargement

Les sources de WordPress sont récupérables sur le site fr.wordpress.org directement depuis la page d'accueil. Vous récupérez un fichier .zip (ou .tar.gz) contenant un dossier intitulé « wordpress » que vous devez décompresser dans le répertoire racine de votre serveur web. Par exemple :

- C :/wamp/www sous windows avec WAMP
- /var/www sur Ubuntu/Debian
- /Users/[nom]/Sites avec MAMP

Une fois le dossier en place, nous pouvons aller à l'adresse indiquée dans le code web, afin de passer à l'étape suivante.

▷ http ://localhost/wordpress
Code web : 107442

Paramétrage de l'installation

Si vous allez directement sur l'URL de votre site, vous obtenez une page d'erreur indiquant que WordPress a besoin d'un fichier wp-config.php pour fonctionner (voir la figure 1.9).

À partir de là, vous pouvez soit cliquer sur le bouton vous proposant de créer automatiquement un fichier de configuration en remplissant les informations de votre base de données, soit créer ce fichier à la main. Dans le premier cas, il faudra que votre serveur web ait les droits d'écriture à la racine de votre site pour pouvoir générer le fichier.

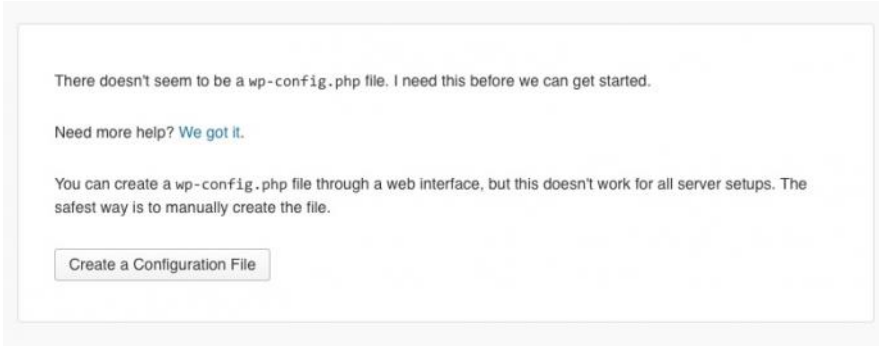


FIGURE 1.9 – Le fichier wp-config.php est manquant

Suivre l'installation automatique

Si vous suivez les étapes d'installation proposées par WordPress, il vous suffit de vous laisser guider et de renseigner les informations de connexion à votre base de données. Si vous n'avez fait aucune modification particulière, le login sera à nouveau root et il n'y aura pas de mot de passe (laissez le champ vide). Une fois les données correctement remplies, WordPress va créer pour vous le fichier de configuration ainsi que les tables nécessaires dans la base de données.

Si les informations de connexion sont correctes (comme sur la figure 1.10), vous devriez pouvoir procéder à la finalisation de l'installation.

FIGURE 1.10 – La configuration de la base de données est correcte

Créer le fichier wp-config.php

Il est aussi possible de créer le fichier de configuration à la main en se basant sur le fichier wp-config-sample.php situé à la racine de votre installation. Il suffit pour cela de copier le fichier en le renommant wp-config.php, puis de modifier les informations de connexion à l'intérieur du fichier.

Quelle que soit la méthode que vous avez choisie pour installer WordPress, toute modification ultérieure des informations de connexion à la base de données devra se faire en éditant le fichier wp-config.php.

Une fois cette étape franchie, vous pouvez choisir un titre pour votre site, définir votre nom d'utilisateur ainsi qu'un mot de passe et votre email. Validez. Votre site est prêt à recevoir ses premiers contenus (voir la figure 1.11) !

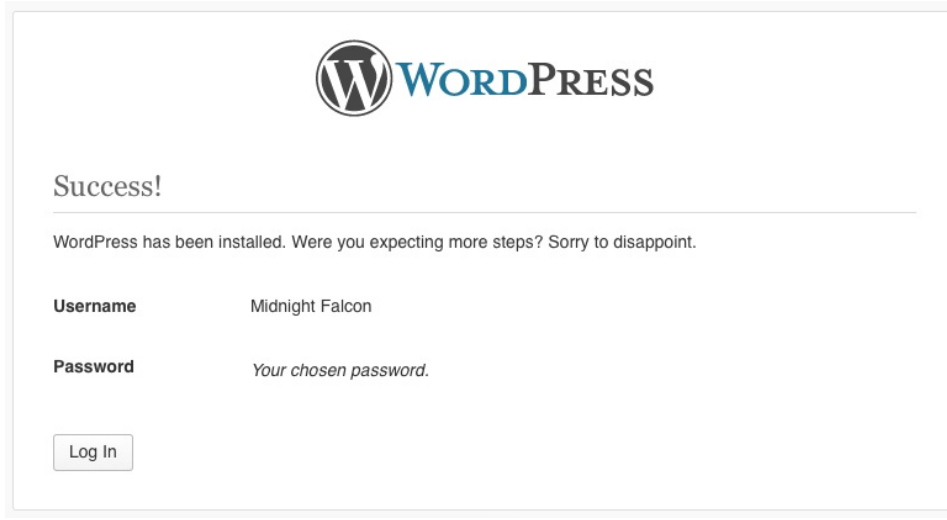


FIGURE 1.11 – L'installation est terminée

L'interface d'administration

Vous pouvez accéder à l'interface d'administration via l'URL indiquée sur le code web suivant :

▷ L'interface d'administration
Code web : 168651

Identifiez-vous avec le nom et le mot de passe fournis pendant la phase d'installation (voir la figure 1.12).

Avant de plonger dans l'ensemble des fonctionnalités offertes par votre CMS, il est bon de commencer par vous familiariser avec son interface.

Le menu est partagé en trois section :

— le tableau de bord ;
— les contenus ;
— les paramètres du site.

Le tableau de bord est une page affichant des informations générales sur l'ensemble du site. On peut y trouver des statistiques, un fil d'actualité sur WordPress, la liste

FIGURE 1.12 – L'interface de connexion à WordPress

des commentaires récents... C'est sur cette page (voir la figure 1.13) que vous arrivez lorsque vous vous connectez à l'interface.

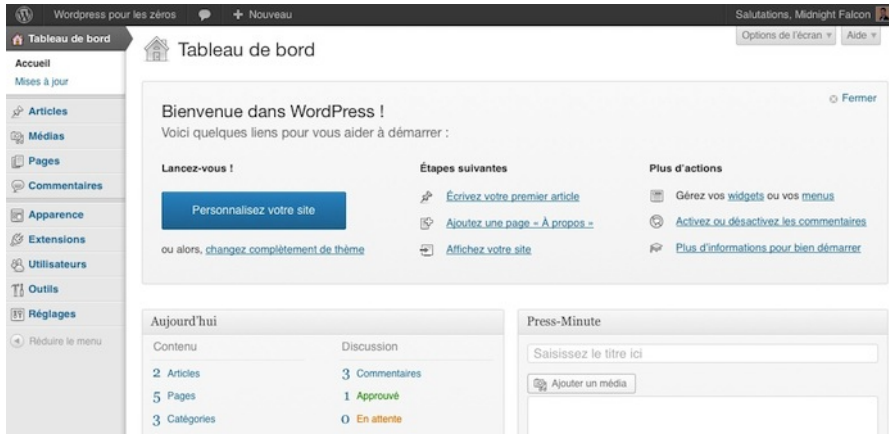


FIGURE 1.13 – L'accueil du panneau d'administration

La seconde section concerne l'édition des contenus que sont les pages, les articles, les médias ainsi que les commentaires. Ici, vous pourrez ajouter, supprimer ou éditer le contenu de votre site et, faire de la modération sur les commentaires. C'est la partie la plus utilisée pour produire du contenu sur le site.

Enfin, le bas du menu donne accès au paramétrage du site, comme le choix du thème et des plugins ou la gestion des utilisateurs. Il est aussi possible de gérer le fonctionnement de certaines fonctionnalités du site dans la partie « Settings ».

Si vous décidez par la suite d'installer des plugins (c'est-à-dire des modules apportant des fonctionnalités supplémentaires au site), ceux-ci pourront rajouter des informations sur le tableau de bord, comme par exemple :

— des statistiques sur vos visiteurs ;
— le nombre d'abonnés à votre newsletter ;
— la création de nouveaux menus à l'administration afin de donner accès à de nouvelles pages ;
— etc.

L'interface est donc susceptible d'évoluer.

En résumé

— WordPress est un CMS écrit en PHP créé en 2003.
— Un site Internet a besoin d'un serveur et d'une base de données pour fonctionner.
— Le serveur peut être installé sur votre ordinateur pour vos tests.
— L'interface d'administration permet de gérer votre site.

Chapitre 2

Les publications

Difficulté :

Votre site est dorénavant installé et prêt à l'emploi, mais il est encore terriblement vide ! Voyons comment remédier à cela en ajoutant du contenu pour les visiteurs.

Les articles

Sous WordPress, les contenus sont organisés en deux types : les articles et les pages. La différence entre les deux réside dans le type de contenu que vous allez placer à l'intérieur.

Un article sera généralement un contenu d'actualité, c'est-à-dire qu'il prend sa plus grande valeur au moment de sa publication. C'est typiquement le type de contenu utilisé pour les publications d'un blog ou sur un fil d'actualité.

Une page aura au contraire un contenu à valeur constante dans le temps sans avoir besoin d'être mise à jour. On peut l'utiliser pour présenter une société, une personne ou bien pour parler d'un sujet de fond.

Au niveau de la présentation, les articles peuvent être affichés en liste par ordre chronologique, puisque c'est ce qui fait leur sens, soit complètement soit avec un aperçu du contenu, tandis que les pages seront accessibles par un lien (le plus souvent dans le menu de navigation) vers leur contenu.

Gestion des articles

Pour créer un article, il suffit d'aller dans « Articles » et de choisir « Ajouter » dans le sous-menu.

L'éditeur de texte

Dans la page qui apparaît, vous avez un champ pour définir le titre de votre article ainsi qu'un éditeur de texte pour taper son contenu et le mettre en forme grâce à la barre d'outils dédiée (voir la figure 2.1).

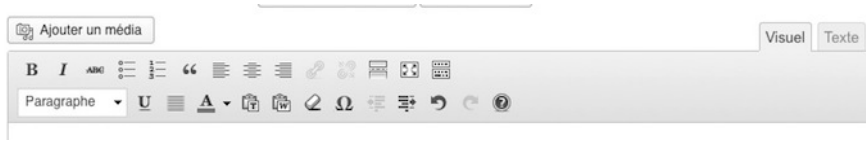


FIGURE 2.1 – L'éditeur de texte riche

Vous pouvez notamment mettre du texte en gras ou en italique, créer des listes à puces, changer l'alignement du texte ou sa couleur.

Pour avoir une description de ce que fait l'un des boutons de l'éditeur, vous pouvez le survoler pour afficher une petite bulle d'informations. Aussi, la dernière icône (en forme de point d'interrogation), affiche l'aide de l'éditeur dans une nouvelle fenêtre. Enfin, deux onglets intitulés «Visuel » et «Texte » permettent d'alterner la vue entre l'aperçu du rendu final et le code HTML généré. Si vous ne connaissez pas le HTML, il est inutile de basculer en mode texte, le mode visuel vous suffira amplement.

Publier l'article

À droite se trouvent les actions disponibles pour l'article en cours de création. Il est notamment possible d'avoir un aperçu de l'article avec le design du blog pour avoir une idée du rendu final. Vous pouvez aussi, comme dans la figure 2.2, gérer l'état de publication de votre article.

FIGURE 2.2 – Publication d'un article

Tant qu'un article n'est pas dans l'état « Publié », il n'apparaît pas sur le site. Vous pouvez donc commencer à le rédiger en tant que brouillon, sauvegarder puis y revenir plus tard pour le terminer. Le bouton « Publier » permet de valider l'article pour qu'il soit affiché, éventuellement à une date ultérieure.

Catégories et mots-clés

Il est possible d'associer une ou plusieurs catégories à un article, ainsi qu'un ensemble de mots-clés. Ceci permettra à vos visiteurs de se repérer plus facilement parmi la liste d'articles aux thèmes variés que vous pourrez écrire.

Les catégories

Pour gérer les catégories, il suffit de se déplacer dans le menu « Articles > Catégories ». La page correspondante permet d'ajouter une catégorie et d'éditer celles qui existent déjà. Par défaut, seule une catégorie est présente et il n'est pas possible de la supprimer, c'est la catégorie par défaut des articles si aucune autre assignation n'est choisie.

Pour créer une catégorie, il faut renseigner :

— le nom de la catégorie, qui sera affiché sur les pages ;
— un identifiant, typiquement utilisé dans l'url lors de l'affichage des articles d'une catégorie donnée ;
— un parent (facultatif), c'est-à-dire que chaque catégorie peut avoir une catégorie parente ;
— une description (facultative) qui sera éventuellement affichée si le thème le permet.

Une fois la catégorie créée, nous pouvons choisir d'y assigner un article. Pour cela, sur la page d'édition de l'article, il suffit de cocher la case correspondant à la catégorie à associer, puis de mettre à jour l'article. Si l'on affiche notre site à nouveau, la catégorie ajoutée devient visible dans le pied-de-page du site (ou sur le coté suivant le thème). En cliquant dessus, vous pouvez constater que l'on obtient sur une nouvelle page la liste des articles associés (voir la figure 2.3).

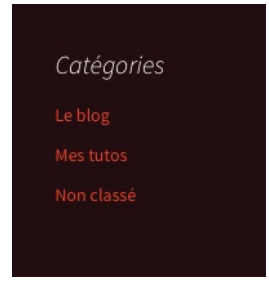


FIGURE 2.3 – La liste des catégories s'affiche sur le blog

Les mots-clés

Contrairement à la description thématique des catégories, les mots-clés ou tags permettent de caractériser un article de façon plus précise. Par exemple, un article sur le fonctionnement de WordPress, pourra être dans la catégorie « Mes tutos » sur le blog, mais pourra avoir comme mots-clés « cours », « informatique », « formation », « fonctionnement », « WordPress », etc. La description du contenu de l'article devient plus riche.

De même que pour les catégories, il existe une page spécifique dans l'administration pour gérer les mots-clés, dans le sous-menu du même nom. On préfère cependant créer les mots-clés directement sur la page de création d'un article, car il est commun d'en créer spécifiquement pour un article.

Pour cela, sous le formulaire permettant l'ajout de catégories, un champ vous invite à indiquer les mots-clés (séparés par des virgules). Il est aussi possible de faire un choix parmi les mots-clés existants les plus utilisés. Créez ou sélectionnez vos tags et mettez à jour l'article pour enregistrer les modifications (voir la figure 2.4).

Lors de la visualisation de l'article, vous aurez dorénavant une liste de mots-clés qui vous redirigeront vers la liste des articles associés, comme c'est le cas pour les catégories. En effet, si deux articles ont un mot-clé en commun (par exemple « informatique »), l'utilisateur pourra retrouver ces deux articles en tapant le mot-clé. Cela permet de faciliter la recherche de publications pour vos visiteurs en leur proposant un éventail des mots qui reviennent fréquemment sur votre site.

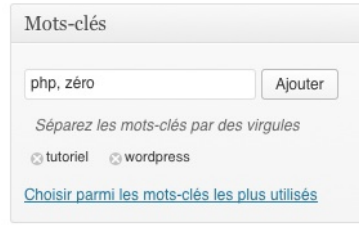


FIGURE 2.4 – Choix des mots-clés pour chaque article

Les pages

La création de pages est très similaire à celle des articles et se fait via le menu « Pages > Ajouter ». L'édition du contenu se fait par le même éditeur de texte et la publication suit le même procédé. En revanche, les options de la page sont différentes. Nous n'avons plus de catégories ou de mots-clés à associer, mais trois attributs nouveaux.

Les attributs

Sur le côté droit de la zone d'édition des pages, un cadre similaire à la figure 2.5 s'affiche.

FIGURE 2.5 – Les différents attributs de page

On peut associer une page parente à une autre, ce qui permet de définir une hiérarchie de pages. Le principal changement visuel apparaît dans le menu principal qui arborera alors des sous-menus pour afficher les pages enfants que vous aurez définies.

L'attribut « Modèle » (qui n'est pas toujours visible suivant les thèmes) sert à changer le format de la page, c'est-à-dire la façon dont elle est affichée. Par défaut, les pages sont le plus souvent affichées avec une colonne latérale (comme pour la liste des articles) qui affiche des widgets (liste de catégories, mots-clés, liens divers). Il est donc possible

sur certains thèmes de choisir de ne pas afficher cette barre latérale sur une page bien précise en choisissant un « modèle de page pleine largeur ».

Enfin, l'attribut « Ordre » définit l'ordre d'apparition de la page dans le menu par rapport aux autres. La page avec l'ordre le plus petit apparaitra à gauche, tandis que la page la plus à droite sera celle avec l'ordre le plus élevé.

Le menu

Un menu est l'élément essentiel de la navigation sur le site, car il donne un lien vers les principales pages de celui-ci. La gestion des menus s'effectue via le sous-menu « Réglages > Menus » dans l'interface d'administration.

WordPress me dit qu'il n'y a pas de menu existant ! Il y a pourtant bien un menu en haut de mes pages, non ?

En effet, en l'absence de menu, celui-ci est automatiquement généré par le thème lors de l'affichage d'une page. En l'occurrence, il affichera un lien vers la page d'accueil ainsi qu'un lien par page que vous aurez créée. Il vaut donc mieux avoir un menu personnalisé si vous souhaitez avoir la main sur l'affichage et les liens qui seront affichés !

Créer un menu

Pour créer un menu, rien de plus simple : il suffit de choisir un nom et de cliquer sur « Créer un menu ».

Une fois le menu créé, il ne manque plus qu'à lui ajouter des liens. Ceux-ci peuvent être de plusieurs types :

- un lien personnalisé pour lequel vous choisissez l'url exacte ;
- une page statique ;
- un lien vers une catégorie.

Lorsqu'un lien est ajouté au menu, il apparaît dans le cadre central qui résume les éléments de menu ajoutés. Il est alors possible, en dépliant le bloc, de supprimer le menu, de modifier le libellé ou de rajouter un attribut title au lien correspondant (voir la figure 2.6).

De cette façon, une bulle apparaît lorsque l'on survole le menu avec la souris (voir la figure 2.7).

Une fois que vous avez ajouté vos liens, il faut activer le menu dans le bloc « Emplacements du thème » où vous pouvez choisir le menu principal. Le menu sera alors utilisé comme navigation principale du site. Si le thème que vous utilisez le permet, il peut être possible d'avoir plus d'un emplacement de menu actif à la fois, vous pourriez alors en activer un deuxième affiché à un autre endroit de vos pages.

FIGURE 2.6 – Modification des attributs du menu

FIGURE 2.7 – L'attribut title s'affiche sous la souris

Je ne peux plus aller vers la page qui liste mes articles, on n'a pas pu l'ajouter au menu !

Vous l'avez vu, il n'est pas possible d'ajouter dans le menu un lien de type « page d'articles ». Vous êtes donc obligés de taper l'url de base du site pour avoir la liste de vos publications, ce qui n'est pas optimal... Deux solutions peuvent régler ce problème.

La première consiste à ajouter un lien personnalisé au menu, qui sera la page d'accueil du site et qui aura par exemple pour titre « Accueil ». Cette solution est néanmoins peu pratique et il faudra changer le menu si votre site change d'adresse.

La deuxième option consiste à associer la liste des articles à une page. Dans « Réglages > Lecture », vous devez alors choisir d'afficher une page statique en tant que page d'accueil (choisissez donc une page de contenu que vous avez créée), puis une page pour lister les articles. Notez que si elle est utilisée pour lister les articles, le contenu de cette page ne sera jamais affiché, il peut donc être vide. Il suffit ensuite d'ajouter ces deux pages à votre menu, la première sera la page d'accueil et la seconde fera le lien vers la liste des articles.

La seconde option ne permet pas de conserver la page des articles comme page d'accueil. Il faut pour cela faire un développement spécifique dans le code PHP de WordPress, ou utiliser un plugin qui le permettrait.

Organiser les menus

L'ordre d'affichage des liens se fait par défaut dans l'ordre d'ajout au menu, mais il est tout à fait possible de modifier cela comme bon vous semble. En glissant-déposant

les menus choisis, vous pouvez les ordonner comme il vous plaît, mais aussi créer des sous-menus. Pour cela, il faut simplement déplacer l'élément de menu légèrement vers la droite, en-dessous de celui qui sera le parent (voir la figure 2.8).

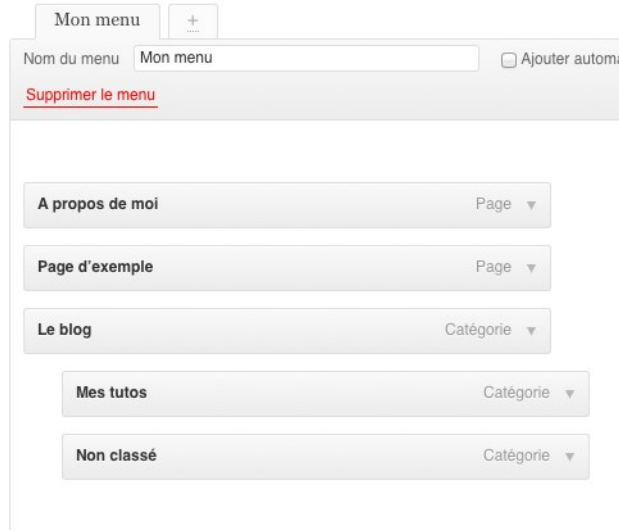


FIGURE 2.8 – L'interface de gestion des menus

Dorénavant, un survol sur le menu parent affichera le sous-menu comme sur la figure 2.9 !

FIGURE 2.9 – Affichage du sous-menu au survol

Ceci permet de reproduire une navigation similaire à ce que l'on avait obtenu en définissant des pages parentes d'autres pages, mais c'est ici applicable à tous les types d'éléments de menu, comme les liens personnalisés et les catégories.

Les médias

Si vous êtes dorénavant capables de créer des contenus sur votre site, il vous manque un élément important pour rendre celles-ci plus attrayantes : les médias. Images, sons

ou vidéos, il est tout à fait possible de les insérer dans vos pages et vos articles pour illustrer vos propos.

Insertion dans un article

Toutes les informations qui suivent sont valables sur les pages ainsi que sur les articles.

Sur la page d'édition d'un article, vous trouverez un bouton « Ajouter un média » (voir la figure 2.10).

FIGURE 2.10 – L'ajouter d'un média

Après avoir cliqué dessus, une nouvelle fenêtre d'envoi de fichier apparaît (voir la figure 2.11).

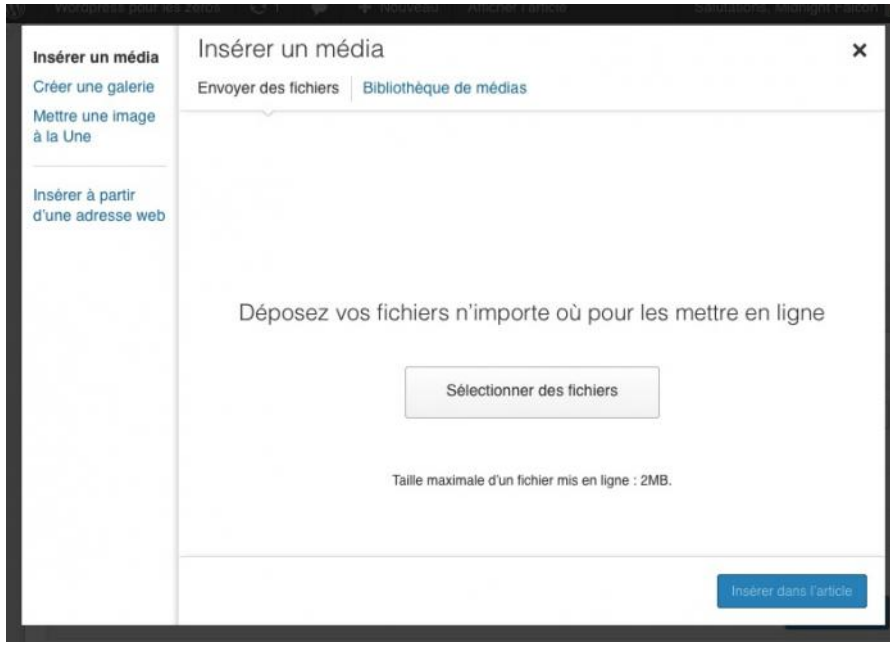


FIGURE 2.11 – L'interface pour l'envoi des médias

Vous pouvez glisser-déposer, comme dans la figure 2.12, un fichier dans la fenêtre ou bien cliquer sur le bouton « Sélectionner les fichiers » pour utiliser l'explorateur de fichiers. Une fois le fichier sélectionné, celui-ci est directement envoyé sur le serveur web et un aperçu est visible dans l'onglet « Bibliothèque de média » de la fenêtre.

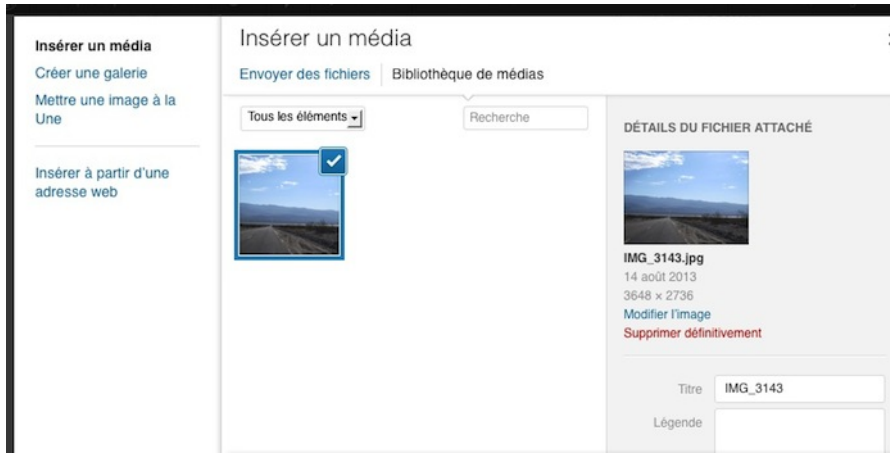


FIGURE 2.12 – Envoie de l'image sur le serveur

Il est possible d'envoyer tout type de fichier, tant que celui-ci à une taille inférieure à 2Mo. Vous pouvez donc sans hésiter insérer un fichier de musique ou même un PDF, WordPress se chargera de proposer un affichage adéquat lors de la visualisation de l'article par un visiteur (voir la figure 2.13).

FIGURE 2.13 – Apparence d'un média audio sur un article

Lorsque le média que vous voulez utiliser dans votre article est envoyé, vous n'avez plus qu'à le sélectionner et à cliquer sur le bouton d'insertion dans l'article en bas à droite. Avant cela, vous pouvez éditer les propriétés du média sur la droite de la fenêtre :

— Titre : le texte à afficher au survol du média ;
— Légende : une légende qui apparaît sous le média ;
— Description : un texte de description plus long que la légende (cela dépend du

thème, elle n'apparait pas sur le thème par défaut par exemple) ;
— Lier à : permet de choisir le comportement lorsque l'on clique sur le média.

Il y a aussi quelques attributs spécifiques aux images :

— Texte alternatif : le texte à afficher si l'image n'a pu être affichée ;
— Alignement : si vous choisissez « gauche » ou « droite », l'image sera bordée par le texte de l'article ;
— Taille : la taille de la miniature à afficher dans l'article.

Enfin, lorsque les médias voulus ont été insérés, n'oubliez pas de mettre à jour votre article pour que les modifications soient prises en compte !

Sur la page « Médias > Ajouter », vous pouvez, de même que sur un article, ajouter de nouveaux médias à la bibliothèque. Notez cependant que pour apparaître sur le site, il faudra obligatoirement ajouter ce média à un article ou une page.

Gérer les médias non utilisés

Lorsque vous supprimez un média d'un article, celui-ci n'est pas totalement supprimé de WordPress, le fichier envoyé reste sur le serveur pour pouvoir être éventuellement utilisé sur d'autres articles.

Pour effacer complètement un média, vous devrez, comme dans la figure 2.14, passer par la page « Médias > Bibliothèque » qui répertorie l'ensemble des fichiers envoyés sur le site. Il suffit alors de cliquer sur le lien « Supprimer définitivement » pour l'effacer de la bibliothèque.

FIGURE 2.14 – Suppression des médias envoyés

Si vous supprimez un média qui est toujours utilisé dans un article, vous aurez alors un lien mort affiché sur votre site, ce qui est très frustrant pour vos visiteurs. Soyez donc certain que le média n'est plus du tout utilisé avant de le supprimer !

En résumé

— Les articles sont des publications régulières dépendant de l'actualité.
— Les pages sont destinées à des présentations.
— Le menu peut être géré automatiquement ou manuellement.
— Tout type de média peut être ajouté dans une publication.

Chapitre 3

Gérer un site participatif

Difficulté :

La vie d'un site, et en particulier d'un blog, ne se limite pas nécessairement à l'action d'une seule personne (en l'occurrence vous). Elle passe aussi par la participation des visiteurs qui peuvent apporter leur réflexion à vos articles, ainsi qu'à d'autres rédacteurs qui peuvent s'investir dans l'enrichissement de votre contenu.

Les commentaires

Si votre site a du succès, il y a fort à parier que des visiteurs voudront laisser des commentaires sur certains articles qui les auront inspiré, voire sur certaines pages.

Activer ou non les commentaires

Tout d'abord, voyons comment faire pour activer (ou désactiver) les commentaires sur les pages et les articles du site.

Dans l'interface d'administration, allez dans « Réglages > Discussion ». En haut de la page, une case vous demande si vous souhaitez « Autoriser les visiteurs à publier des commentaires sur les derniers articles ». Si cette case est cochée, tous les articles et les pages que vous créerez pourront recevoir des commentaires, et inversement si vous la décochez. En revanche, cette option ne permet pas d'activer ou désactiver les commentaires sur les contenus qui sont déjà créés. Pour cela, il faut aller dans le menu « Articles », puis choisir « Modification rapide » sous l'article dont vous souhaitez changer le statut (voir la figure 3.1).

FIGURE 3.1 – Plusieurs actions sont disponibles pour les articles

Vous pouvez alors, comme sur la figure 3.2, modifier l'autorisation des commentaires via la case à cocher « Autoriser les commentaires ».

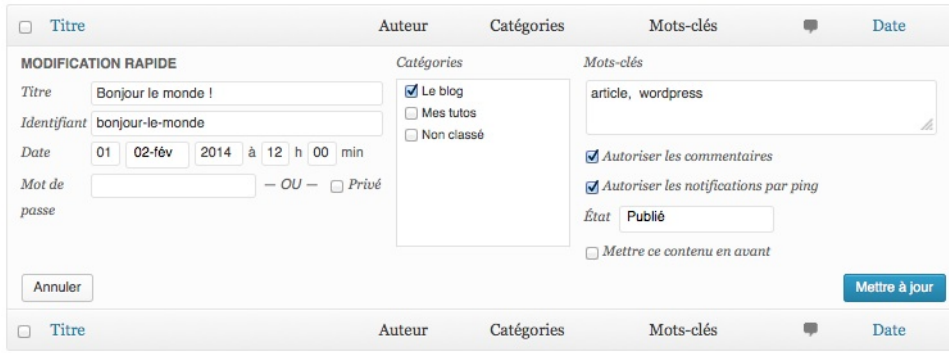


FIGURE 3.2 – Autorisation des commentaires

Pour faire ce changement sur la totalité de vos articles, vous devrez sélectionner « Modifier » dans la liste des actions groupées, sélectionner tous les articles pour lesquels

vous souhaitez désactiver les commentaires, et cliquer sur « Appliquer ». Un formulaire similaire au précédent s'affiche alors et permet entre autres d'activer (ou désactiver) les commentaires pour tous les articles (voir la figure 3.3).

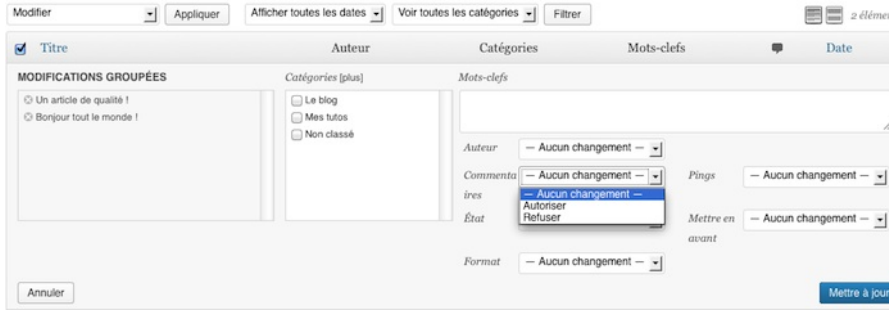


FIGURE 3.3 – L'activation des commentaires peut être groupée

Les actions décrites ici sont aussi valables pour les pages en passant par le menu listant les pages du site.

Les options

Revenons à la page « Réglages > Discussion ». De nombreux paramètres sont disponibles ici pour vous aider à gérer les commentaires, il est par exemple possible de demander à ce que tous les commentaires soient validés par un administrateur ou bien de désactiver les commentaires pour les vieux articles. En bas de la page il est possible de choisir si les commentaires peuvent avoir un avatar à coté du nom de l'utilisateur l'ayant posté. Si oui, vous devez alors choisir un type d'avatar :

— Homme mystère : un logo d'utilisateur anonyme est affiché ;
— Vide : une image blanche est affichée ;
— Gravatar : l'image est récupérée sur Gravatar, un service permettant d'associer une image à une adresse email ;
— Génération automatique : l'avatar sera une image aléatoire dans le style visible sur l'aperçu.

Si votre choix se porte sur Gravatar, il vous faut alors choisir un niveau maximal pour la censure de l'image, c'est-à-dire qu'une image réservée aux adultes ne sera par exemple pas affichée si vous choisissez un niveau parmi G, PG ou R. Si vous ne savez pas lequel sélectionner, laissez « G – Visible pour tous », qui est celui utilisé par défaut.

Modérer les commentaires

Les statuts

Une fois un commentaire créé, il peut se trouver dans plusieurs statuts différents.

- Approuvé : le commentaire est visible sur le site ;
- En attente : le commentaire est en attente de validation, seul vous ou un autre administrateur du site peuvent le voir ;
- Indésirable : le commentaire est masqué car il a été désapprouvé par un administrateur.

Enfin, il est possible d'avoir un commentaire dans la corbeille si vous avez décidé de le supprimer. Voyons tout de suite comment procéder à ces changements de statut.

Édition et suppression

La liste des commentaires présents sur le site est affichable par le menu « Commentaires » du panneau d'administration. En survolant un commentaire et en cliquant sur « Modifier », vous accédez à la page d'édition d'un commentaire.

Lorsque vous êtes connecté comme administrateur, il est aussi possible d'accéder directement à la fiche d'édition d'un commentaire avec le lien « Modifier » qui apparaît sur le front-office lors de la visualisation d'un commentaire.

Au centre vous pouvez modifier le contenu du commentaire ainsi que le nom de l'utilisateur ayant posté celui-ci. Un historique des actions sur ce commentaire (par exemple un changement de statut) est aussi affiché en bas de page.

Il est déconseillé d'éditer le contenu d'un commentaire car l'utilisateur pourra penser que vous cherchez à modifier sa déclaration en votre faveur. N'utilisez cette fonctionnalité que pour supprimer une phrase offensante par exemple, bien que dans la plupart des cas la suppression pure et simple du commentaire soit préférable.

Sur la droite, nous retrouvons la liste des statuts présentés plus haut, ainsi que la date de création du commentaire et un lien pour le déplacement dans la corbeille.

Si nous choisissons de placer le commentaire comme « En attente », vous pouvez voir qu'il est immédiatement retiré de l'affichage public sous l'article auquel il se rattachait. D'autre part, un petit « 1 » est apparu dans le menu d'administration pour indiquer qu'un commentaire était en attente de validation par un administrateur (voir la figure 3.4).

Si le commentaire est marqué comme « Indésirable » ou bien placé dans la corbeille, il n'apparaît plus dans la liste des commentaires. Il faut alors choisir le filtre « Indésirable » (ou corbeille) en haut de la page pour y accéder (voir la figure 3.5).

FIGURE 3.4 – Un commentaire en attente de validation

FIGURE 3.5 – La corbeille regroupe les messages supprimés

Vous pouvez alors choisir d'annuler ce statut et de restaurer le commentaire pour qu'il apparaisse à nouveau, ce qui est utile en cas d'erreur par exemple (voir la figure 3.6).

FIGURE 3.6 – Les commentaires indésirables peuvent être conservés ou effacés

Réglages

En plus des paramètres s'appliquant à tous les commentaires que nous avons déjà parcouru, il est possible de filtrer certains textes contenus dans les commentaires pour les empêcher d'être publiés sans votre approbation. Dans le champ « Modération de commentaires », vous pouvez choisir un ensemble de mots qui déclencheront la mise attente d'un commentaire ayant ce contenu dans son texte. Il n'est pas obligatoire de mettre des mots complets, c'est-à-dire que « Press » bloquera les commentaires contenant « WordPress », « Pression » ou encore « Presse ».

Faites bien attention à ne mettre qu'un mot bloquant par ligne dans la zone de texte.

Vous pouvez faire de même dans le champ « Liste noire pour les commentaires », à la différence que les commentaires correspondant à l'un des mots choisis seront immédiatement marqués comme « Indésirables », sans avoir à être modérés.

Les utilisateurs

Gestion des utilisateurs

Toute la gestion des utilisateurs s'effectue par le menu « Utilisateurs » de l'administration qui vous permet de gérer l'ensemble des utilisateurs existant, d'en créer de nouveaux et d'éditer votre propre profil. Vous pouvez donc lister tous les utilisateurs de votre site (voir la figure 3.7).

FIGURE 3.7 – Liste des utilisateurs inscrits sur votre site

Les rôles

À chaque utilisateur est associé un rôle définissant l'ensemble des droits qui lui sont accordés, c'est-à-dire la liste des actions qu'il pourra effectuer dans l'interface d'administration du site.

- Administrateur : c'est le rôle qui accorde tous les droits et que vous avez par défaut en tant que créateur du site ;
- Éditeur : il permet de publier et éditer l'ensemble des articles du site, même celles qui ne lui appartiennent pas ;
- Auteur : l'auteur peut publier et éditer ses propres contenus ;
- Contributeur : il peut créer ou éditer des articles et des pages lui appartenant, mais ne peut pas les publier (il lui faut demander à un éditeur ou un administrateur de le faire pour lui) ;
- Abonné : il peut uniquement accéder au panneau d'accueil de l'administration et gérer son profil.

Pour modifier le rôle d'un utilisateur, cliquez sur son nom dans la liste des utilisateurs. Vous pouvez alors faire votre choix grâce à une liste déroulante. Notez que vous ne pouvez pas changer votre propre rôle (voir la figure 3.8) !

FIGURE 3.8 – Attribution des rôles aux utilisateurs

La gestion des droits est un sujet important pour la sécurité de votre site. Il faut toujours faire attention à n'accorder que les droits strictement nécessaires à un utilisateur. Évitez donc de créer un profil administrateur si la personne qui l'utilisera n'a besoin que d'écrire des articles.

Création d'utilisateur

Vous pouvez créer un nouvel utilisateur en allant dans le sous-menu « Ajouter ». Cette interface permet de définir le nom du nouvel utilisateur, son rôle et quelques informations personnelles optionnelles.

Il est aussi possible d'autoriser la création de comptes utilisateur pour n'importe quel visiteur du site. Pour cela, dans le menu « Réglages > Général », il suffit de cocher la case « Tout le monde peut s'enregistrer » et de définir le rôle par défaut de ces utilisateurs.

Il est peu probable que vous ayez besoin de cette option, mais il est bon de savoir qu'elle existe. En revanche, si vous l'utilisez, faites attention à ce que le rôle par défaut des utilisateurs soit le plus faible possible (abonné), sinon vous vous exposez à des modifications du contenu de votre site par n'importe qui voulant s'enregistrer !

En résumé

— Les commentaires permettent aux visiteurs de s'exprimer sur les publications.
— L'administrateur peut modérer les commentaires avant et après leur publication.
— Plusieurs personnes peuvent contribuer au site, éventuellement avec des droits différents.

Chapitre 4

Modifier l'apparence

Difficulté :

L'identité de votre site passe avant tout par le thème graphique, c'est-à-dire l'ensemble des caractéristiques visuelles de votre site, comme les couleurs, la taille des textes, les images de fond ou la disposition des blocs sur la page. Lorsque vous avez installé WordPress, vous n'avez pas eu la possibilité de choisir le thème graphique utilisé, et pour cause : il est installé par défaut et c'est ensuite à vous d'en trouver un autre et de le changer si vous le voulez. Voyons tout de suite comment procéder.

Changer de thème

Le choix d'un thème se fait lui aussi dans l'interface d'administration, on y accède par le menu « Apparence > Thèmes ». Ici vous aurez probablement un ou deux thèmes préinstallés et que vous pouvez activer ou désactiver à loisir (voir la figure 4.1).

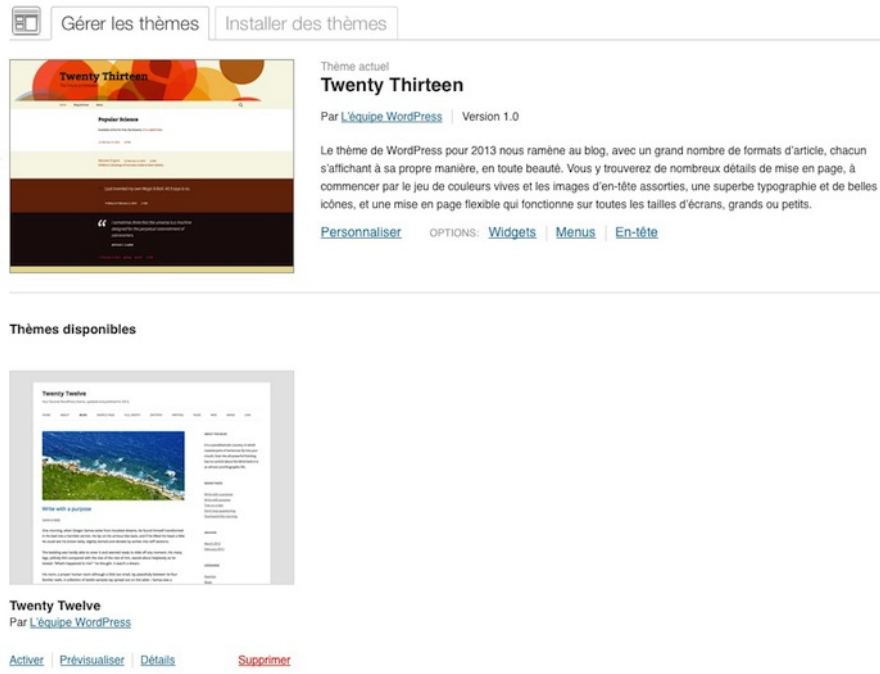


FIGURE 4.1 – Liste des thèmes installés

Essayons de modifier le thème actuellement utilisé (ici, « Twenty Thirteen ») par le thème « Twenty Twelve ». Cliquez sur le lien « Activer », puis rafraîchissez votre site : la nouvelle apparence est d'ores et déjà mise en place, il n'y a rien de plus à faire (voir la figure 4.2) !

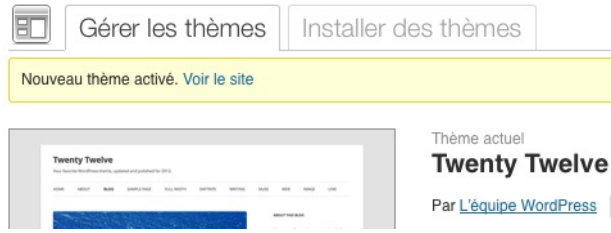


FIGURE 4.2 – Le thème actif a été modifié

Et si je veux essayer de nouveaux thèmes non fournis avec WordPress ?

Il est en effet possible d'installer des thèmes créés par la communauté de diverses façons : en utilisant l'interface d'administration ou en téléchargeant les fichiers vous-même.

Ajouter un thème via l'administration

Pour ajouter un thème directement depuis l'interface, choisissez l'onglet « Installer des thèmes » en haut de la page de sélection des thèmes. Ici, vous pouvez rechercher des thèmes par mots-clés, envoyer un fichier .zip contenant un thème préalablement téléchargé, ou encore parcourir les derniers thèmes ajoutés par les créateurs (voir la figure 4.3).

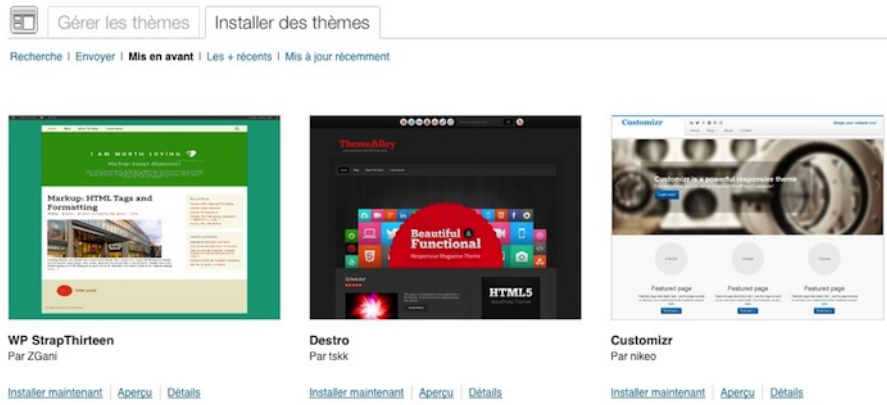


FIGURE 4.3 – Nouveau thème à télécharger

Lorsque vous parcourez une galerie de thèmes (c'est-à-dire en faisant une recherche ou en parcourant les derniers thèmes mis à jour) vous avez la possibilité d'avoir un aperçu du résultat final du thème par le lien « Aperçu » en dessous de chaque miniature. En cliquant sur le bouton « Installer maintenant », WordPress va télécharger automatiquement le thème demandé et celui-ci sera ensuite disponible dans la liste des thèmes.

Utiliser un thème téléchargé

Dans le cas où vous auriez téléchargé un thème séparément, par exemple directement sur le site officiel de WordPress, à l'onglet « Themes » ou sur un site proposant des thèmes WordPress, vous pouvez installer ce thème manuellement en plaçant les fichiers dans l'arborescence du CMS.

Essayez, par exemple, d'installer le thème proposé sur le code web suivant :

▷ Le thème Expound
Code web : 254578

Décompressez le fichier .zip récupéré et placez le dossier expound dans le répertoire wordpress/wp-content/themes/ de votre installation. Si vous affichez à nouveau la liste des thèmes dans l'interface d'administration, Expound est maintenant sélectionnable pour une utilisation sur votre site (voir la figure 4.4) !

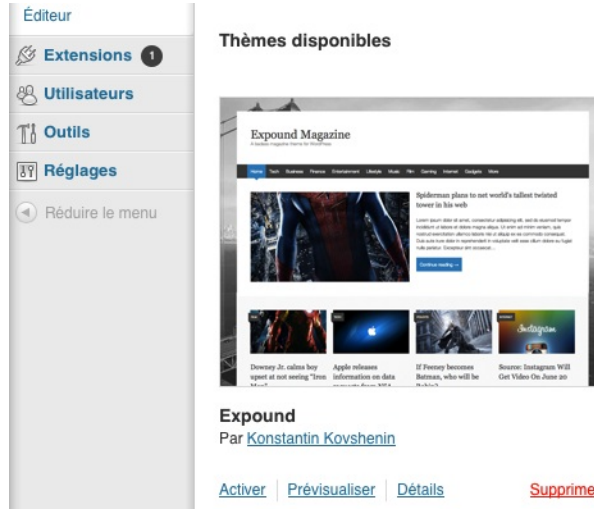


FIGURE 4.4 – Les thèmes peuvent être installés manuellement

Ajouter des widgets

Les widgets (ou composant d'interface graphique en français) sont des petits blocs à ajouter sur les pages de votre site. Ils fournissent une information ou une fonctionnalité spécifique aux visiteurs. Par exemple, il peut s'agir de l'heure, d'un calendrier, du bulletin météo, de mini-jeux... C'est un moyen simple de rajouter du contenu dynamique, toujours sans avoir besoin de modifier le code du site.

Placer un nouveau widget

Dans le menu Apparence > Widgets, on retrouve au centre de la page l'ensemble de tous les widgets disponibles sur le site, et sur la droite, les différentes zones de votre thème pouvant contenir lesdits widgets. En effet, les widgets ne peuvent pas être placés n'importe où dans une page, ils ont besoin qu'une (ou plusieurs) zone du thème graphique utilisé soit conçue pour en recevoir. Par défaut, WordPress a un certain nombre de widgets livrés sans que vous ayez besoin de les installer. Il est

néanmoins toujours possible d'en rajouter au travers de plugins, que nous verrons plus tard comment récupérer.

Pour ajouter un widget dans une zone spécifique du site (qui est déterminée par le thème utilisé), il vous suffit de le glisser-déposer de la zone centrale vers la « Zone principale » (ce nom peut varier en fonction du thème utilisé) sur la droite. Ajoutons par exemple le widget « Nuage de mots-clés » par cette méthode (voir la figure 4.5).

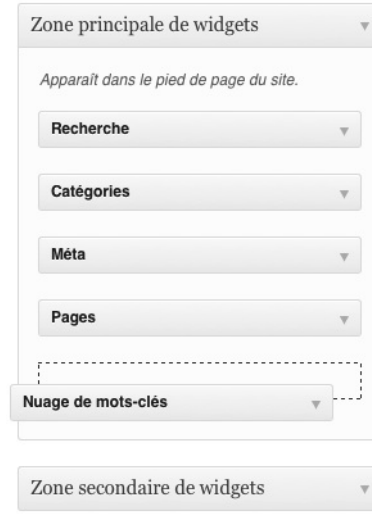


FIGURE 4.5 – Assigner le widget à une zone spécifique

Une fois qu'un widget est placé dans une zone de widgets, vous pouvez éditer ses propriétés, qui déterminent son comportement sur les pages de votre site. Ajoutons donc le titre « Mots-clés » et choisissons la taxinomie « Mots-clés », qui permet d'afficher une courte liste des mots-clés les plus utilisés pour marquer les articles que vous avez publiés. L'autre option, « Catégories », propose le même affichage mais pour les catégories d'articles. Cliquez donc sur « Enregistrer », puis rafraîchissez la page d'accueil du site ; le nouveau widget doit apparaître dans la zone choisie (avec le thème « Twenty Thirteen » et la zone « Principale », ce sera le pied-de-page), à coté de ceux déjà présents (voir la figure 4.6).

Désactiver un widget

Pour retirer un widget, rien de plus simple : il suffit de cliquer sur le lien « Supprimer ». Cela retirera le widget de la zone dans laquelle il se trouvait et supprimera les réglages appliqués (voir la figure 4.7).

En revanche, vous pouvez aussi choisir de ne plus afficher un widget, tout en conservant les options que vous aviez choisies, comme le titre par exemple. Pour cela, vous pouvez déplacer le widget en question dans la zone « Widgets désactivés » située en bas de la

FIGURE 4.6 – Le pied de page affiche le widget fraichement ajouté

FIGURE 4.7 – Suppression du widget

page de gestion des widgets. Vous pourrez alors replacer le widget sur votre site plus tard avec les mêmes réglages.

En résumé

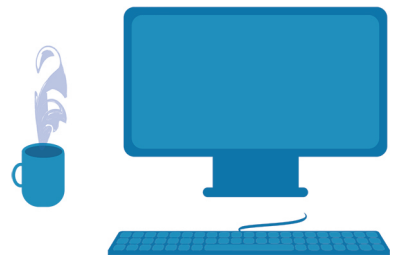
— L'apparence du site est déterminée par le thème choisi.
— De nouveaux thèmes peuvent être téléchargés depuis la bibliothèque officielle.
— Les widgets offrent des fonctionnalités supplémentaires dans les zones périphériques de chaque page.

Chapitre 5

TP : Créez vos premières pages

Difficulté :

Afin de pratiquer les notions présentées jusqu'ici, je vous propose un exercice d'application consistant à créer différents contenus au travers de l'interface d'administration. Nous allons donc créer l'ébauche d'un blog avec quelques éléments de base. Ce sera ensuite à vous de poursuivre ce travail au gré de votre imagination !



Présentation de l'exercice

Consignes

Ce TP ne devrait pas vous poser beaucoup de problèmes. Je vous donne ici les lignes directrices afin de parcourir à nouveau les fonctionnalités principales que nous avons vues jusque là, pour réaliser le contenu de votre site.

L'objectif sera donc de créer votre premier article. Je vous propose de suivre, dans l'ordre les éléments suivants :

— installer un thème ;
— créer un article ;
— créer des catégories, dont « Le blog » ;
— créer une page de présentation personnelle ;
— créer un menu comprenant trois éléments ;
— installer des widgets.

Tout d'abord, commencez par **installer un thème spécifique**. Dans mon cas, j'ai choisi « Base WP », mais n'hésitez pas à faire un choix différent selon vos goûts. Vous pourrez ensuite **créer un premier article** de présentation de votre blog, similaire à la figure 5.1.

FIGURE 5.1 – Votre premier article doit attirer l'oeil !

Tous les éléments de mise en forme sont disponibles dans l'éditeur de texte riche. N'hésitez pas à explorer les possibilités qu'il offre et à rajouter du contenu supplémentaire ! Vous devrez au minimum intégrer une liste à puces, un lien vers une autre page du site, une image et le formatage du texte (par exemple avec des éléments en gras ou italique).

Ensuite, vous devez créer des catégories, dont « Le blog ».

Une fois les catégorie créées, vous devrez **créer une page de présentation personnelle** qui sera rattachée à la catégorie « Le blog ». Rattachez aussi votre premier article à cette catégorie.

Le menu doit donc comprendre (comme sur la figure 5.2) :

— la page d'accueil faisant le listing des articles publiés ;
— la page de présentation personnelle ;
— un lien vers la catégorie « Le blog ».

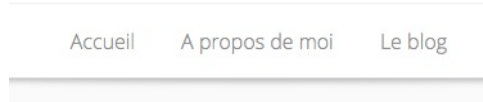


FIGURE 5.2 – Le menu du blog

Enfin, je vous demande d'utiliser, comme sur la figure 5.3, **les widgets** suivants :

— le calendrier ;
— la barre de recherche ;
— les catégories ;
— les mots clés.

Vous aurez probablement des widgets déjà installés avec votre thème, il faudra donc les supprimer s'ils sont inutiles. N'oubliez pas que vous pouvez toujours les rajouter plus tard si vous en éprouvez le besoin.

Avant de passer à la correction, vérifiez bien que vous avez ajouté tous les éléments demandés ci-dessus. N'hésitez pas à vous référer aux chapitres précédents si vous ne retrouvez pas une fonctionnalité précise.

À tout de suite pour la correction !

Correction

Si vous avez bien suivi le cours jusqu'ici, vous devriez n'avoir rencontré que peu de problèmes pour débuter sur votre blog. Nous allons reprendre ensemble les points principaux qui ont pu vous poser quelques difficultés.

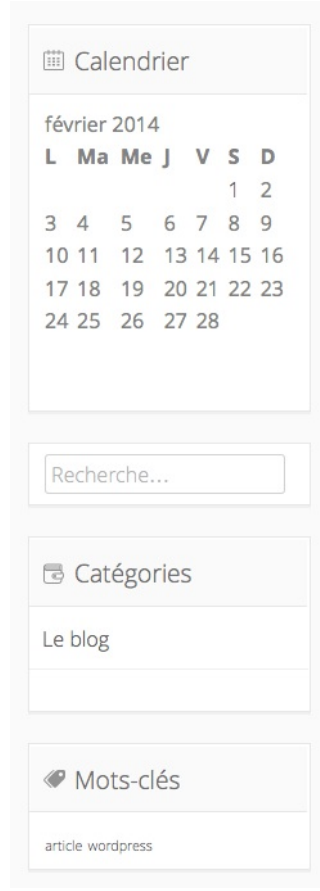


FIGURE 5.3 – La barre latérale contenant les widgets

Le thème

Le thème est généralement le point de départ pour votre site puisqu'il détermine l'identité de celui-ci. Les images de présentation du TP on été prises sur le thème « Base WP », que vous pouviez utiliser en utilisant la fonctionnalité de recherche de thèmes présentée dans la chapitre dédié à ce sujet.

La page de présentation personnelle

C'est le premier contenu par lequel vous pouviez commencer, étant donné qu'il fallait ensuite créer un lien vers cette page sur votre premier article. Je n'ai pas posé de critère de réalisation particulier ici, vous étiez donc assez libre. Pour ajouter cette page à la catégorie « Le blog », vous pouviez passer par le formulaire sur la droite de l'éditeur qui permet de créer et assigner la catégorie (voir la figure 5.4).

FIGURE 5.4 – Créez et sélectionnez la catégorie à assigner à la page

L'article d'introduction

Comme je vous l'ai dit, cet article contenait plusieurs éléments à insérer avec l'éditeur de texte de WordPress : une liste à puces, une image et un lien.

Pour insérer une liste à puces, il suffit de cliquer sur cette petite icône (voir la figure 5.5).

FIGURE 5.5 – Bouton pour les listes à puces

Ensuite, ajouter les puces une par une, en tapant le texte et en passant à la ligne. Un nouveau clic sur l'icône permet d'arrêter la liste.

Notez que vous pouviez aussi créer des listes numérotées avec le bouton suivant (voir la figure 5.6).

L'ajout d'une image se fait avec le bouton « Ajout des médias » au dessus de l'éditeur de texte. Il vous suffit ensuite de choisir une image comme nous l'avions fait dans le chapitre sur les médias, puis de valider (voir la figure 5.7).

FIGURE 5.6 – Bouton pour les listes numérotées

FIGURE 5.7 – Bouton pour l'ajout d'image

Il fallait ensuite créer un lien vers la page de présentation créée précédemment. Le bouton en forme de chaîne, comme sur la figure 5.8, permet de réaliser cette tâche.

FIGURE 5.8 – Bouton pour les liens

En cliquant dessus, une fenêtre s'ouvre. Vous deviez donc taper l'adresse d'un lien quelconque (vers l'extérieur de votre site) ou bien un lien vers une autre page du blog en sélectionnant celle-ci dans la liste affichée (comme sur la figure 5.9).

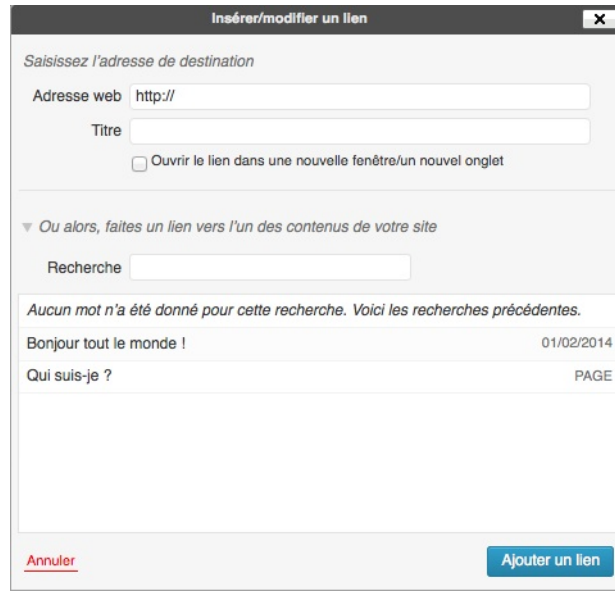


FIGURE 5.9 – Édition du lien à insérer

Enfin, avant de publier l'article, assignez-lui la catégorie créée plus tôt grâce au formulaire sur la droite de la page.

Le menu

Pour ajouter les liens demandés à votre menu, il fallait commencer par créer un nouveau menu en passant par l'écran « Apparence > Menu ». Vous pouviez ensuite rajouter les pages qui vous intéressent en les sélectionnant manuellement, puis en cliquant sur « Ajouter au menu ». Faites ensuite de même pour la catégorie « Le blog » (voir la figure 5.10).

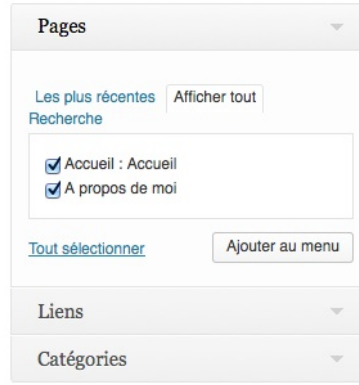


FIGURE 5.10 – Ajoutez les pages dans le menu

N'oubliez pas ensuite de lier le menu à votre thème dans l'onglet « Gérer les menus ». Choisissez le menu dans la liste déroulante avant d'enregistrer ce changement (voir la figure 5.11).

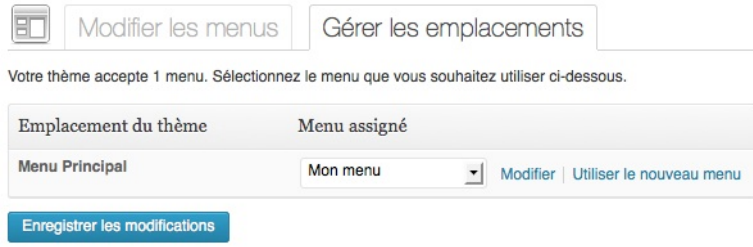


FIGURE 5.11 – Sélectionnez le menu à utiliser avec ce thème

Les widgets

Pour terminer, vous deviez ajouter quelques widgets à votre thème. Sur l'illustration présentée au début de l'exercice, j'ai utilisé quatre widgets :

— le calendrier ;
— la recherche ;
— la liste de catégories ;

— le nuage de mots-clés.

Pour les intégrer, il vous suffit d'aller sur l'écran de gestion des widgets et de les faire glisser un par un dans la zone de widgets voulue. Par exemple ici, la barre latérale (voir la figure 5.12).

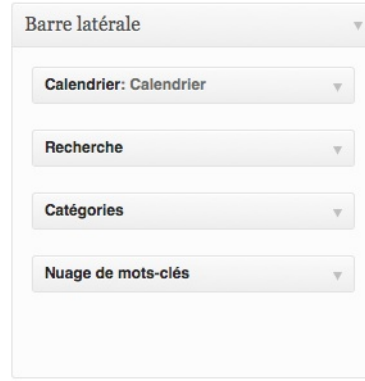


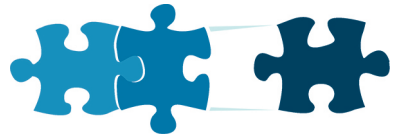
FIGURE 5.12 – Les widgets utilisés pour ce TP

Chapitre 6

Ajouter des plugins

Difficulté :

Vous l'avez vu avec les thèmes, WordPress propose une grande souplesse dans la personnalisation de votre site, et ceci avec des processus simplifiés au maximum. De même que nous avons pu modifier l'apparence de notre site sans difficulté, nous pourrons aussi lui ajouter de nouvelles fonctionnalités grâce aux plugins. Les plugins sont des modules à installer individuellement dont le but est d'étendre les fonctionnalités d'un logiciel, ici WordPress. Par exemples, intégrer des boutons Facebook, ajouter les statistiques de visites dans le panneau d'administration ou encore avec une galerie de photos sur vos publications...



Gérer les plugins

Installer de nouveaux plugins

Tout comme les thèmes, les plugins peuvent être installés directement par l'interface d'administration ou manuellement en téléchargeant les fichiers nécessaires et en les plaçant dans le dossier approprié.

Installation automatique

Nous accédons au moteur de recherche des plugins par le menu « Extensions > Ajouter », qui propose une méthode de recherche similaire à celle des thèmes. Vous pourrez chercher des plugins par mots-clés, par date d'ajout ou bien simplement parcourir les plus utilisés par la communauté. Cherchons parmi les extensions les plus populaires en cliquant sur le lien correspondant (voir la figure 6.1).

La grande majorité des plugins est créée à destination du monde entier, par conséquent leur description mais aussi la documentation sont généralement écrites en anglais. Ne soyez donc pas surpris d'avoir des résultats de recherche dans la langue de Shakespeare !

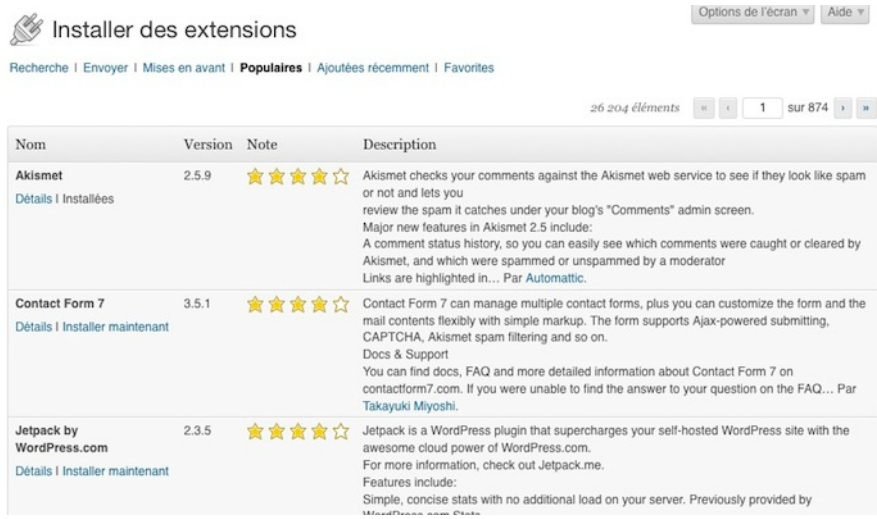


FIGURE 6.1 – La bibliothèque de plugins de WordPress

Dans la liste qui s'affiche, nous pouvons voir pour chaque plugin son nom, la version actuelle, une note des utilisateurs et une courte description de ses fonctionnalités. Pour lancer l'installation, cliquez simplement sur « Installer maintenant » et WordPress se chargera de télécharger les fichiers nécessaires au fonctionnement, là encore tout se passe selon le même procédé que pour les thèmes.

Installation manuelle

Les plugins sont aussi téléchargeables manuellement sur le site officiel de WordPress, dans la section « Plugins ». Lorsque vous avez récupéré les fichiers du plugin, placez-les dans le dossier wordpress/wp-content/plugins, cela suffit pour terminer l'installation !

Activer ou supprimer le plugin

L'administration propose un panneau de gestion des plugins accessible via « Extensions > Extensions installées ». Vous pourrez ici gérer les différents plugins installés sur votre site.

Une fois qu'un plugin est installé, il est nécessaire de l'activer pour que ses fonctionnalités soient appliquées. Cela se fait simplement en cliquant sur le lien « Activer » sous la description du plugin. De même, si vous n'avez plus besoin d'un plugin, vous pouvez le désactiver de la même façon. Enfin, pour désinstaller un plugin, c'est-à-dire pour supprimer l'ensemble de ses fichiers et les informations qu'il a pu stocker dans votre base de données, il faut cliquer sur le lien « Supprimer ».

La suppression est irréversible ! Vous devrez réinstaller le plugin si vous voulez le réutiliser plus tard.

Pour pouvoir supprimer un plugin, il faut d'abord que celui-ci soit désactivé.

Mise à jour d'un plugin

Afin de bénéficier de nouvelles fonctionnalités ou simplement de corrections de bugs, les plugins ont la possibilité d'être mis à jour lorsque l'équipe en charge du développement en sort une nouvelle version. Ceci vous est signalé dans le menu principal avec une petite vignette indiquant le nombre de plugins pouvant bénéficier d'une mise à jour (voir la figure 6.2).

FIGURE 6.2 – Une mise à jour est disponible

Avant de mettre à jour le plugin, consultez les détails de la mise à jour. Vous y trouverez une liste décrivant l'ensemble de ces mises à jour et des risques éventuels. Lorsque vous êtes prêt pour récupérer la dernière version, cliquez sur « Mettre à jour automatiquement », WordPress téléchargera automatiquement les nouveaux fichiers.

Si vous êtes plutôt adepte du téléchargement manuel de l'archive contenant le code du plugin, vous pouvez effectuer la mise à jour d'un plugin en récupérant la nouvelle version du code et en remplaçant l'ancien dossier du plugin à mettre à jour, de la même façon que pour l'installation.

Exemples de plugins

Vous savez maintenant rechercher et installer des plugins, en voici quelques-uns parmi les plus populaires qui pourraient vous être utiles.

qTranslate

Par défaut, un site utilisant WordPress ne peut être affiché qu'en une seule langue inscrite dans la configuration. De plus, vos articles et pages sont rédigés dans la langue de votre choix, mais il n'est pas possible d'en avoir plusieurs versions dans des langues différentes.

qTranslate permet de contourner cette limitation en proposant une interface de rédaction des publications modifiée pour que celles-ci soient rédigées en plusieurs langues. Il est aussi possible aux utilisateurs de choisir la langue dans laquelle ils souhaitent parcourir votre site. Une fois le plugin installé, un nouveau menu fait son apparition dans « Réglages > Langues » et vous permet notamment de choisir les langues que vous souhaitez utiliser sur votre site (voir la figure 6.3).

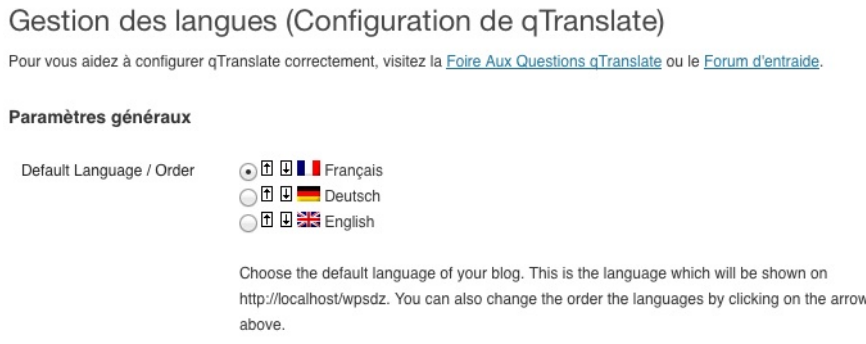


FIGURE 6.3 – Gestion des langues

Il est très probable que lors de l'activation du plugin, votre interface passe en anglais. Vous devez alors ajouter le français à la liste des langues activées et en faire la langue par défaut.

Sur la page d'édition des articles, un champ de texte par langue a été rajouté pour le titre de la publication. De plus, un système d'onglets permet de choisir la langue dans laquelle vous éditez le contenu, vous avez donc toutes les versions d'une même publication sur une seule page. C'est ensuite l'utilisateur qui choisira la langue dans laquelle il souhaite parcourir le contenu (voir la figure 6.4).

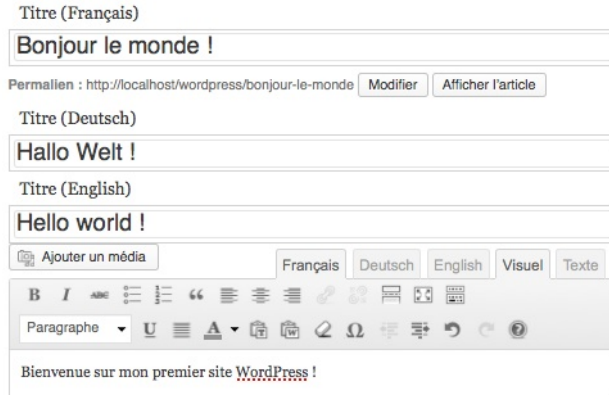


FIGURE 6.4 – L'interface modifiée pour la rédaction

Enfin, pour que les utilisateurs puissent changer de langue comme bon leur semble, le plugin fournit aussi un widget pour choisir la langue dans laquelle les pages doivent s'afficher. L'affichage peut varier du lien à la liste déroulante, avec éventuellement l'affichage des drapeaux correspondant aux pays (voir la figure 6.5). Vous verrez, c'est très pratique !

FIGURE 6.5 – Le widget de qTranslate

Hupso Share Buttons

Ce second plugin se concentre sur les réseaux sociaux en proposant d'ajouter des boutons de partage pour une vaste sélection d'entre eux. Vos visiteurs peuvent ainsi partager vos publications d'un simple clic vers les réseaux sociaux. Les boutons peuvent être ajoutés en dessous de vos articles, pages et même comme widget suivant la configuration que vous choisissez (voir la figure 6.6).

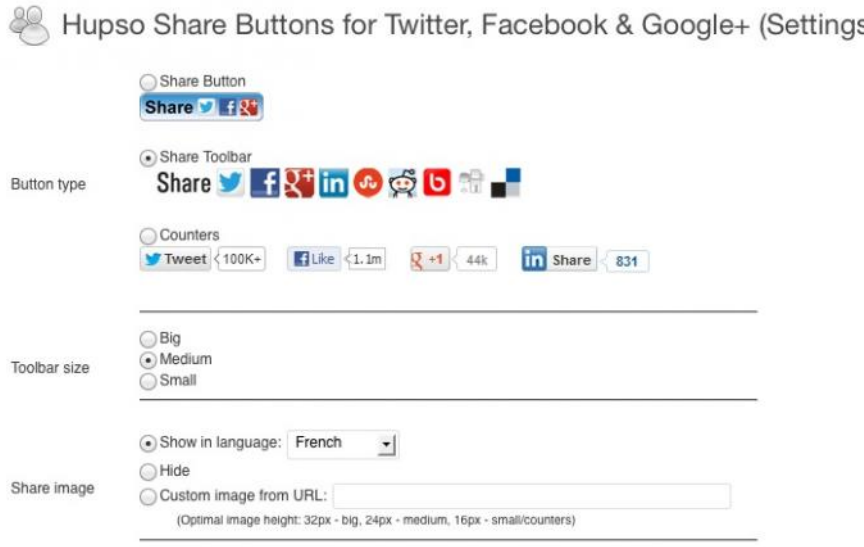


FIGURE 6.6 – Quelques options pour la configuration des boutons

De nombreuses options sont par ailleurs disponibles, vous pouvez ainsi choisir la position des boutons et la façon dont le texte de partage est déterminé, exclure des catégories d'articles du partage ou encore changer le type de bouton. C'est donc un plugin très utile qui vous évite les étapes fastidieuses d'intégration de ces boutons pour chaque réseau social, et le résultat est plutôt réussi (voir la figure 6.7).

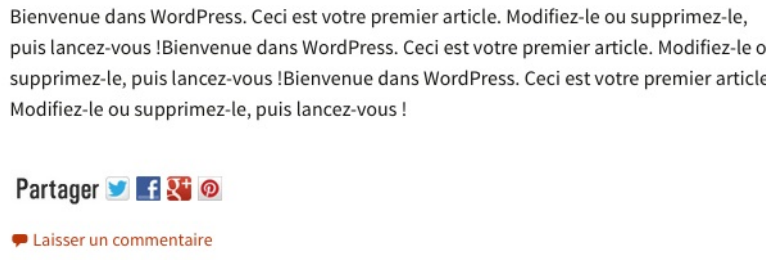


FIGURE 6.7 – Le rendu sous les articles

Lorsque ce plugin est utilisé sur une installation locale, les fonctionnalités de partage peuvent être incomplètes voire ne pas fonctionner. C'est un comportement normal dû au fait que les réseaux sociaux essayent de se connecter au site en partageant un lien dans le but d'y récupérer des informations de partage. Cette étape n'étant pas réalisable sur un site local, le partage ne peut fonctionner normalement.

NextGEN Gallery

Pour terminer cette sélection, regardons de plus près le plugin NextGEN Gallery, qui permet la création de galeries d'images à afficher dans vos publications.

Vous commencez par créer un album dans lequel vous ajoutez des images une par une ou par groupe de façon très simplifiée, vous les classez, puis vous pouvez afficher celles-ci à divers endroits : dans un article, une page ou un widget. L'affichage peut aussi prendre la forme d'un carrousel afin de faire défiler les images de façon automatique, c'est à vous de choisir.

Ici aussi, de nombreuses options permettent d'adapter le plugin à tous vos besoins, que ce soit en changeant la taille, le style, les durées de transitions des images, ou encore en permettant une sélection précise des images à afficher en fonction du contexte de la page visualisée (voir la figure 6.8).

FIGURE 6.8 – L'affichage sous forme de galerie

En résumé

— WordPress est personnalisable à l'aide de plugins. Ces modules permettent d'ajouter de nombreuses fonctionnalités.
— Les plugins sont développés par la communauté.
— Il existe plus de 25 000 plugins, n'hésitez pas à parcourir la bibliothèque !

Deuxième partie

Développer votre thème

Chapitre 7

Premiers pas dans le code

Difficulté :

Avant de nous attaquer au développement sous WordPress, je propose une entrée en matière générale afin de prendre nos premières marques avec la documentation, ainsi que la structure de WordPress et de ses concepts. Une fois que nous aurons dégrossi le sujet, vous serez prêts à plonger au cœur de votre site !

Utiliser la documentation

Si vous avez déjà développé des sites web en utilisant un CMS ou un framework, vous avez probablement déjà eu besoin de rechercher des informations dans la documentation dédiée. Dans le cas contraire, sachez que c'est une mine d'informations pour les développeurs qui décrit l'ensemble des fonctionnalités fournies par l'application pour vous permettre de les exploiter.

Sur WordPress, la documentation s'appelle Codex. Vous le retrouverez sur le code web suivant :

▷ La documentation Codex
Code web : 894247

Elle regroupe tout ce qu'il faut savoir sur la création de thèmes, de plugins ou encore de widgets. Elle décrit aussi le fonctionnement interne de WordPress, n'hésitez donc pas à la parcourir si vous souhaitez approfondir vos connaissances ou simplement détailler un point précis.

La structure de WordPress

Système de fichiers

La plupart des fichiers situés directement à la racine de WordPress sont soit des points d'entrée de l'application (c'est-à-dire appelés directement par le client lors d'une requête vers le site) tels que index.php, wp-login.php ou wp-signup.php, soit des fichiers permettant l'initialisation de l'application, comme wp-config.php ou wp-settings.php.

Vous avez ensuite trois dossiers qui regroupent le cœur de WordPress :

— wp-includes regroupe toute la logique de WordPress, pour toutes les fonctionnalités natives (hors administration).
— wp-admin permet de regrouper les pages et fonctionnalités de l'interface d'administration.
— wp-content contient les plugins et les thèmes installés dans votre application (respectivement dans les sous-dossiers « Plugins » et « Themes »), c'est dans ce dossier que nous développerons de nouvelles fonctionnalités.

J'insiste lourdement sur le fait que tout le code que vous écrivez devra se trouver dans un plugin ou bien dans un thème, et nulle part ailleurs ! Vous ne devez jamais modifier les fichiers natifs de WordPress pour des raisons de propreté du code : tout ce qui est spécifique à votre site doit être cloisonné pour être facilement réutilisable. De plus, lorsque WordPress est mis à jour, tous les fichiers extérieurs aux plugins et aux thèmes sont susceptibles d'être modifiés. Vos éventuelles modifications seraient donc perdues !

Rien ne vous empêche en revanche d'ouvrir les fichiers de WordPress, par exemple pour chercher ce que fait exactement une fonction que vous utilisez. Il est ainsi très commun d'avoir à regarder le contenu du dossier wp-includes lorsque l'on commence à rajouter des fonctionnalités personnalisées.

La base de données

Maintenant que vous avez fait connaissance avec les fichiers de WordPress, nous allons jeter un coup d'œil à ce qui se cache dans la base de données. N'ayez pas peur, c'est vraiment très léger !

WordPress ne contient que 11 tables par défaut, c'est-à-dire si vous ne faites pas de modification particulière. Bien entendu, des plugins peuvent créer de nouvelles tables si les fonctionnalités qu'ils rajoutent le nécessitent.

— wp_commentmeta et wp_comments permettent la sauvegarde des commentaires sur les publications du site.
— wp_links était utilisée dans les anciennes versions de WordPress et est toujours présente pour des raisons de compatibilité.
— wp_options contient les valeurs des paramètres de configuration du site, comme le slogan, qui peuvent être éditables dans l'interface d'administration.
— wp_postmeta et wp_posts stockent le contenu des publications du site, que ce soit des articles ou des pages, mais aussi les menus. Il faut noter qu'il existe un système de version des pages et des articles et que les différentes versions sauvegardées sont représentées chacune par une ligne dans la table wp_posts.
— wp_term_relationships, wp_term_taxonomy et wp_terms contiennent les informations relatives aux catégories, aux tags, ainsi que leur lien avec les différents articles et pages du site.
— wp_usermeta et wp_users maintiennent les données utilisateurs, que ce soit leur nom ou bien les droits accordés à chacun.

Les noms des tables présentées tiennent compte du préfixe par défaut wp_ ajouté lors d'une installation standard, le début des noms peut donc varier si vous avez choisi un préfixe différent. Dans la suite du cours, les tables seront notées avec le préfixe par défaut.

Le principe des actions

Théorie

Dans WordPress, tout ou presque a été prévu dans le but de pouvoir être adaptable par des développeurs tiers voulant personnaliser l'application. Pour cela, il faut que celle-ci fournisse un moyen de rajouter du code spécifique sans pour autant que le développeur ait besoin de modifier les fichiers natifs de WordPress. De cette façon, le code rajouté peut être partagé sur un autre site en ajoutant simplement de nouveaux fichiers et sans modifier les existants. Pour offrir cette possibilité, le concept des actions a été introduit.

Une action est une fonction qui est appelée lorsqu'un événement particulier se produit, par exemple la sauvegarde d'un article par un contributeur. L'idée générale est que, dans notre exemple, la fonction qui gère la sauvegarde de l'article va déclencher un

événement intitulé save_post, qu'un certain nombre de fonctions peuvent observer, pour s'exécuter à cet instant.

Le nombre de fonctions observant un événement n'est pas limité, c'est là tout l'intérêt de ce principe. Si vous avez besoin de faire un traitement spécifique lorsqu'un article est sauvegardé, il vous suffit de créer la fonction effectuant ce traitement et de la brancher sur l'événement save_post, et WordPress s'occupera de faire l'appel au moment opportun.

Les fonctions utilisées

Concrètement, l'enregistrement d'une action se fait à l'aide de la fonction add_action(), qui prend en paramètre le nom de l'événement déclencheur et le nom de la fonction à appeler.

Un troisième argument peut-être ajouté à cet appel, c'est la priorité. Étant donné que les actions peuvent être multiples sur un événement donné, vous pourriez avoir besoin de déterminer si votre action sera exécutée avant ou après une autre. C'est l'utilité de la priorité : les actions ayant la priorité de plus faible niveau seront exécutées en premier, la valeur par défaut étant 10.

Ainsi, si vous voulez exécuter un fonction nommée ma_fonction() après les autres actions observant l'événement save_post, vous pouvez faire l'appel qui suit.

```
1 <?php
2 add_action('save_post', 'ma_fonction', 20);
```

L'appel aux actions connectées à un événement donné se fait lorsque la fonction do_action() est exécutée, celle-ci prenant en argument le nom de l'événement et les éventuels arguments à passer aux actions exécutées.

L'appel à l'événement save_post, situé dans le fichier wp-includes/post.php, suit bien entendu ce format :

```
1 <?php
2 do_action('save_post', $post->ID, $post);
```

WordPress et la programmation orientée objet

Retour en arrière

Comme il a été dit en introduction, la première version de WordPress est sortie au cours de l'année 2003. De son côté, PHP5, qui inclut un support avancé des concepts objets dans le langage, est sorti en 2004, soit peu de temps après.

Par conséquent, le développement de WordPress s'est à l'origine purement organisé sur du code procédural, c'est-à-dire sans notions de programmation orientée objet. Au fil des versions, de nouveaux composants on été ajoutés à WordPress, et ceux-ci ont été construits dans une mesure de plus en plus importante autour des objets.

Cependant, afin de maintenir la plus grande compatibilité possible pour les plugins et les thèmes existants, créés par la communauté, mais aussi pour ne pas avoir a réécrire complétement le CMS, une grande partie du code est restée sous forme procédurale au fil des ans. Il est donc très courant d'utiliser des fonctions de l'espace global, tout en ayant parfois affaire à des objets pour les fonctionnalités les plus récentes de WordPress. Ne soyez donc pas étonné d'avoir à manier ces deux aspects au cours de vos développements, c'est tout à fait normal d'après la conception du moteur.

Et votre code dans tout ça ?

En ce qui concerne le code que vous écrirez, il n'y a pas de norme particulière à suivre : de nombreux plugins sont encore écrits sans avoir une structure objet, d'autres au contraire y sont très attachés. Dans la suite de ce cours, les exemples seront le plus souvent orientés objet, mais il reste toujours certains cas particuliers (notamment les thèmes) pour lesquels la structure de WordPress n'est pas adaptée. Nous devrons donc aussi, suivant les cas, concilier les deux.

Gardez toutefois à l'esprit que le choix reste le vôtre et qu'aucune obligation n'existe. Si la programmation objet devient de plus en plus utilisée sur les applications PHP, le fonctionnement du CMS n'est pas toujours le choix le plus évident.

En résumé

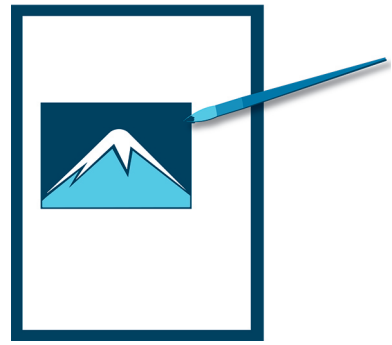
- Le codex renferme toutes les informations techniques sur WordPress.
- Il existe 11 tables permettant de stocker l'ensemble des informations spécifiques à votre site.
- Les actions permettent d'exécuter des fonctions lors du déclenchement d'événements spécifiques.
- WordPress contient du code procédural et orienté objet dans son coeur, il vous appartient de choisir comment vous souhaitez organiser votre propre code.

Chapitre 8

Les thèmes

Difficulté :

Nous avions vu dans la première partie de ce cours comment installer un nouveau thème WordPress. Cela nous a permis de modifier l'apparence de notre site, mais il est certain que vous préféreriez avoir la possibilité de créer votre propre thème ou bien de modifier un thème existant. Il est maintenant temps de nous y plonger !



Structure d'un thème

Pour créer un thème, il faut commencer par ajouter un dossier dans le répertoire wp-content/themes. Ce dossier porte généralement le nom du thème (sans espaces, ni caractères spéciaux), par exemple « themezero ».

Mais que mettre dans ce nouveau dossier ?

Dans WordPress, un thème a une structure assez simple, car il ne nécessite qu'au minimum, deux fichiers !

Le fichier styles.css

Tout d'abord, un fichier style.css permet de déclarer le thème auprès de WordPress ; on y renseignera notamment le nom du thème, l'auteur et le site Internet, ou encore un numéro de version. S'il est préférable de remplir le plus d'informations possible, notamment dans le cas où vous décidiez de publier votre thème, seul le nom de celui-ci est obligatoire.

Pour déclarer le thème, il suffit de mettre les informations demandées dans un commentaire au début de votre fichier style.css.

```
/*
Theme Name : Le thème des zéros
Author : Midnight Falcon
Author URI : http://monsiteweb.com
Description : Notre premier thème WordPress !
*/
```

Ce fichier style.css pourra aussi contenir des règles de mise en forme CSS (c'est tout même le but de ce type de fichier !). La seule obligation est d'avoir un en-tête de déclaration du thème. Rien qu'avec ce fichier, WordPress est capable de détecter votre nouveau thème, celui-ci doit ainsi être visible dans la liste des thèmes (voir la figure 8.1).

Un premier fichier PHP

Ensuite, vous aurez besoin d'un fichier PHP pour générer le code HTML de la page. Un fichier index.php est le minimum pour démarrer et peut théoriquement suffire pour un thème, même s'il sera difficile d'aller bien loin avec si peu !

Ça paraît trop simple ! Comment peut-on gérer tous les cas proprement avec seulement un seul fichier PHP ? J'ai vu des thèmes avec plus d'une dizaine de fichiers !

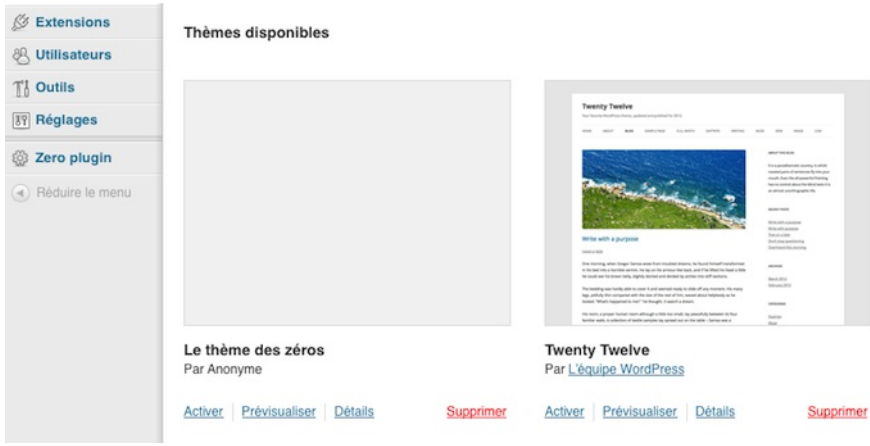


FIGURE 8.1 – Le nouveau thème dans l'administration

Effectivement, à moins de gérer tous les cas d'affichage dans votre fichier index.php et d'arriver à un code illisible, votre thème sera forcément incomplet. Pour cette raison, WordPress va lire un fichier différent du thème suivant le contexte de la page courante. Par exemple, pour afficher la liste des articles associés à une catégorie, le fichier category.php sera appelé. Cependant, si celui-ci n'existe pas, l'application applique le comportement par défaut et c'est le fichier index.php qui sera utilisé pour le rendu.

Ce fonctionnement de WordPress est le **mécanisme de fallback**, il vous permet de ne pas avoir à créer autant de fichiers que de types de pages sur le site et de mutualiser le code pour les pages ayant un affichage semblable. Quand le fichier censé être utilisé pour une page n'existe pas, on retombe sur un fichier par défaut.

Aussi, cette hiérarchie de fichier comporte en général plusieurs niveaux. Par exemple, si la catégorie que vous souhaitez afficher s'appelle « zéro », WordPress ira tout d'abord chercher un fichier nommé category-zero.php, puis category.php si le premier est inexistant. Puis, il ira chercher index.php.

Il y a en réalité cinq niveaux de hiérarchie pour les templates d'une catégorie, j'ai volontairement simplifié le système pour que vous compreniez.

Nous pouvons d'ores et déjà créer le fichier index.php et écrire une simple ligne dedans.

```
1 <?php
2 echo 'Bonjour les zéros';
```

Activez maintenant le thème et rafraîchissez une page pour constater que le fichier index.php est bien appelé : le texte « Bonjour les zéros » s'affiche sur la page.

Un fichier courant : functions.php

Il reste un dernier fichier qui revient très souvent dans les thèmes, c'est le fichier functions.php qui contient des fonctions aidant à l'affichage dans un thème donné. Il ne s'occupe pas du rendu à proprement parler mais vous permet principalement de définir des traitements particuliers à appeler depuis vos templates, dans le but d'éviter d'alourdir ces derniers avec trop de code PHP.

Son inclusion est faite automatiquement par WordPress s'il est présent dans le dossier du thème courant. Les fonctions qui y sont définies sont alors accessibles depuis tous les fichiers du thème.

Par exemple, créez une fonction display_hello() dans ce fichier :

```
1 <?php
2 function display_hello()
3 {
4     echo 'Bonjour les zéros';
5 }
```

Nous pouvons maintenant appeler cette fonction dans n'importe quel template de notre thème, nous allons donc le faire dans index.php :

```
1 <?php
2 display_hello();
```

Vous devez donc obtenir le même résultat que précédemment, le code a simplement été déplacé dans une fonction.

Héritage de thème

Avant de nous lancer tête baissée dans la création d'un nouveau thème à partir de rien, commençons simplement : modifier un thème existant.

En revanche, pas question de modifier directement les fichiers du thème original ! En effet, si jamais celui-ci venait à être mis à jour, vous perdriez vos modifications en téléchargeant les dernières nouveautés.

Pour concilier ces deux points, l'héritage de thème est exactement ce dont vous avez besoin. Il s'agit de définir un thème dit « enfant » qui pourra alors ne surcharger que les éléments nécessaires du thème dit « parent ». Tous les fichiers qui ne sont pas redéfinis dans le thème enfant seront pris dans le thème parent. Vous êtes donc libre de modifier ce que bon vous semble en copiant certains fichiers du thème parent dans le thème enfant afin de les adapter à l'apparence que vous souhaitez obtenir.

Déclaration du thème enfant

Comme pour la déclaration d'un thème classique, un thème hérité consiste à créer un nouveau dossier (par exemple enfant) placé dans le répertoire themes. Celui-ci doit

contenir un fichier style.css indiquant des informations de base. En plus du nom du nouveau thème, il faut indiquer quel thème sera le parent avec la ligne « Template ».

```
/*
Theme Name : Mon template hérité
Template : twentythirteen
*/
```

Ici, nous avons créé un template enfant du thème par défaut « Twenty Thirteen ». Notez bien que la directive « Template » doit contenir le nom du **dossier** du thème parent, et non le nom du thème déclaré dans le fichier style.css de ce dernier.

Si vous activez ce nouveau thème, vous vous rendrez compte que le contenu de la page est bon, mais la mise en forme est complètement détruite. Malgré l'héritage de thème, le fichier style.css du thème parent n'est pas pris en compte. C'est une exception à l'héritage des fichiers de thème.

Le fichier functions.php est une autre exception : si vous en créez un dans le thème enfant, il sera inclus en plus de celui du thème parent, mais ne l'écrasera pas. Vous pouvez donc rajouter de nouvelles fonctions sans risquer de rendre inutilisables celles qui étaient créées par défaut.

En revanche, rien ne vous empêche de faire explicitement appel au fichier CSS du parent à partir de votre propre fichier style.css afin de récupérer le design original.

```
@import url("../twentythirteen/style.css");
```

Le site a maintenant dû reprendre son apparence originale. Partant de cette étape, nous pouvons commencer à rajouter de nouvelles règles CSS pour modifier le thème d'origine. Essayons par exemple de changer la couleur du nom du site.

```
.site-header h1 {
    color:green;
}
```

Si l'on rafraîchit l'une des pages du site, le titre est bien devenu vert (voir la figure 8.2).

Surcharge de fichiers

Nous pouvons à présent modifier le style d'un thème, mais pourquoi ne pas aller plus loin et modifier le contenu des pages ? Par exemple, en bas de toutes les pages, vous pouvez voir que la signature de l'équipe WordPress apparaît. Vous souhaitez la retirer ? Aucun problème, il suffit de surcharger le template footer.php pour retirer le lien qui y est généré. Vous pouvez même en profiter pour rajouter votre propre mention, comme un copyright (voir la figure 8.3).

```
<footer id="colophon" role="contentinfo">
    <div class="site-info">
```

FIGURE 8.2 – Nouvelle couleur du titre

```
3           Thème du zéro, reproduction interdite.
4       </div>
5   </footer>
```

FIGURE 8.3 – Le nouveau pied-de-page

Vous avez donc remplacé le pied-de-page original par votre contenu personnel. Ce système de surcharge fonctionne avec tous les fichiers de templates, n'hésitez donc pas à l'utiliser pour modifier des thèmes existants.

Ajouter une zone de widgets

La majorité des thèmes sous WordPress permettent l'ajout de widgets dans une ou plusieurs zones des pages du site. Il est obligatoire qu'un thème supporte au moins une zone de widgets pour que ceux-ci puissent être ajoutés via le menu « Apparence > Widgets » de l'administration, nous allons donc détailler la création d'une telle zone.

Enregistrer la zone

Pour pouvoir afficher une zone de widgets, il faut commencer par l'enregistrer avec la fonction register_sidebar(), qui prend un tableau en paramètre. Celui-ci peut avoir jusqu'à sept clés :

— id : un id unique qui servira à afficher la zone ;
— name : le nom de la zone qui sera affiché dans l'administration ;
— description : le texte à afficher sur la page de gestion des widgets ;
— before_widget : code HTML à afficher avant chaque widget ;
— after_widget : code HTML à afficher avant chaque widget ;
— before_title : code HTML à afficher avant chaque titre de widget ;
— after_title : code HTML à afficher après chaque titre de widget.

Aucune des clés n'est obligatoire mais mieux vaut toutes les définir vous-même, sinon WordPress choisira des valeurs par défaut (entre autres, l'id sera généré de façon à être unique). De plus, la fonction doit être exécutée lors de l'action widget_init, il faudra donc passer par un appel à la fonction add_action(). Dans le fichier functions.php du thème, nous ajoutons donc le code suivant :

```
1  <?php
2  add_action('widgets_init','zero_add_sidebar');
3  function zero_add_sidebar()
4  {
5      register_sidebar(array(
6          'id' => 'my_custom_zone',
7          'name' => 'Zone supérieure',
8          'description' => 'Apparait en haut du site',
9          'before_widget' => '<aside>',
10         'after_widget' => '</aside>',
11         'before_title' => '<h1>',
12         'after_title' => '</h1>'
13     ));
14 }
```

Vous devriez maintenant voir apparaître, comme sur la figure 8.4, la nouvelle zone sur la page d'édition des widgets dans l'administration.

Afficher les widgets

Une fois la zone de widgets créée, vous pouvez l'afficher où bon vous semble dans votre thème avec un appel à dynamic_sidebar(), prenant en paramètre l'id de la zone choisie.

```
1  <div><?php dynamic_sidebar('my_custom_zone');?></div>
```

Une méthode alternative : des widgets sans zone

Il est aussi possible d'afficher un widget dans un thème sans passer par une zone dédiée. Mais si vous procédez ainsi, le widget ne sera pas modifiable dans le menu « Apparence

FIGURE 8.4 – Apparition de la nouvelle zone dans l'administration

> Widgets », c'est donc une méthode moins commune d'afficher les widgets.

Pour procéder, nous devons appeler la fonction the_widget(), pouvant prendre trois paramètres :

```
1 <?php the_widget( $widget, $instance, $args ); ?>
```

Le paramètre $widget correspond au nom du widget à ajouter, c'est-à-dire à la classe PHP décrivant le widget. La liste des widget natifs de WordPress est disponible cette page de la documentation que vous retrouverez sur le code web suivant :

▷ Liste des widgets natifs
Code web : 213568

Le paramètre $instance doit contenir les paramètres (sous forme d'un tableau) du widget, c'est-à-dire ceux que vous auriez définis dans l'administration lors de l'ajout du widget à une zone, par exemple le titre. De nouveau, les valeurs possibles sont indiquées dans la documentation.

Enfin, $args (à nouveau un tableau) permet de choisir la valeur des variables before_widget, after_widget, before_title, after_title, que nous avons vues plus haut. Seul le premier argument est obligatoire, il est donc très simple d'ajouter un widget avec cette fonction.

```
1 <?php the_widget( 'WP_Widget_Calendar'); ?>
```

Ajouter un menu

Déclaration du menu

De même que pour les widgets, l'affichage d'un menu passe par une zone dédiée qui doit être au préalable déclarée par le thème pour être utilisée lors de l'initialisation de WordPress. L'enregistrement se fait au travers de la fonction register_nav_menu(), prenant deux paramètres : **l'identifiant du menu**, qui doit être unique pour un thème

donné (un thème peut utiliser plusieurs menus), et le **libellé du menu**, qui sera affiché dans le panneau d'administration pour l'identifier sur l'écran de gestion des menus.

Il vous suffit donc d'utiliser une action sur l'événement init comme dans l'exemple suivant.

```
<?php
add_action('init', 'zero_add_menu');
function zero_add_menu()
{
    register_nav_menu('main_menu', 'Menu principal');
}
```

Cet emplacement doit maintenant apparaître dans l'interface d'administration (la gestion des menus) dès lors que votre thème est actif (voir la figure 8.5).

FIGURE 8.5 – Apparition de la nouvelle zone de menu

L'affichage

Pour afficher le menu, il vous suffit d'appeler la **fonction wp_nav_menu()** qui nécessite un tableau comme unique paramètre. Celui-ci peut contenir les différentes clés suivantes :

- theme_location : c'est la zone de menu que vous souhaitez afficher. Ce sera le plus souvent la seule clé vraiment nécessaire à préciser ;
- menu : si vous souhaitez insérer un menu précis, spécifiez son nom ici ;
- menu_class : ceci est la classe CSS à appliquer au menu, la valeur par défaut étant 'menu' ;
- container : la balise HTML pour définir le conteneur du menu (par exemple 'div', qui est la valeur par défaut) ;
- container_class : la classe CSS à appliquer au conteneur du menu ;
- before, after : du code HTML à insérer avant (ou après) chaque lien du menu.

Vous devez donc fournir vos paramètres sous la forme d'un tableau avec des paires clé-valeur. Pour rappel, la clé pour indiquer le nom « Menu » est : theme_location. L'appel le plus simple du menu tient donc simplement en une ligne :

```
<?php wp_nav_menu(array('theme_location' => 'main_menu'));
```

Pour que votre menu s'affiche sur le site, n'oubliez pas de l'assigner à la zone voulue dans le panneau d'administration.

En résumé

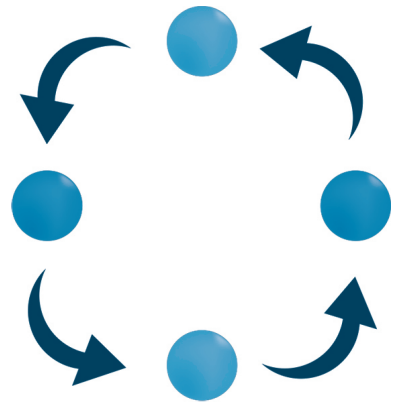
— Un thème a besoin au minimum d'un fichier style.css et index.php pour fonctionner.
— Faire hériter un thème « enfant » permet de personnaliser un design existant sans modifier le thème « parent »
— Les widgets et les menus peuvent être ajoutés dans un thème. Pour cela, il faut déclarer au moins une zone d'accueil pouvant les accueillir.

Chapitre 9

Le processus de rendu

Difficulté :

Après avoir créé un thème pour personnaliser l'apparence du site avec notre propre code HTML et CSS, nous allons détailler le processus de rendu des pages dans WordPress, afin de savoir quelles fonctionnalités sont disponibles pour modifier le comportement de l'application à un point donné.



La boucle de rendu

Les templates tags

Présentation

L'ensemble des fonctions fournies et conçues pour être appelées directement dans les fichiers de thème, sont appelées des **template tags**. Ces fonctions de WordPress peuvent avoir plusieurs utilisations :

— afficher une contenu sur une page ;
— vérifier une condition ;
— récupérer une information globale comme le titre du site ou son adresse.

Par exemple, la fonction bloginfo() permet de récupérer toutes sortes de valeurs dans la base de données en fonction du paramètre qu'on lui fournit : le nom du blog, le répertoire du thème utilisé, la version de WordPress... Aussi, la fonction wp_title() se charge de calculer la valeur du texte présent dans la balise `<title>` de votre page HTML en fonction de la page vue, ce qui nous évite de faire divers tests pour le déterminer (à moins que vous ne vouliez afficher une valeur spécifique).

Le but des template tags est donc d'afficher les éléments ajoutés par l'administrateur du site (par exemple les articles) grâce à l'utilisation de fonctions PHP spécifiques.

Structurer le site

Les template tags permettent aussi d'appeler des portions de code affichées sur de nombreuses pages, comme c'est le cas de l'en-tête et du pied de page, présents sur l'ensemble du site. Pour ces éléments, les fonctions get_header() et get_footer() ont été créées et se chargent de l'inclusion dans votre page des fichiers header.php et footer.php respectivement.

Ainsi, en créant ces deux fichiers avec les contenus pour l'en-tête et le pied de page, le fichier index.php peut être écrit ainsi :

```
1 <?php get_header() ?>
2
3 <!-- ici s'insère le contenu principal de la page -->
4
5 <?php get_footer() ?>
```

Rendu d'un contenu

Les pages et articles

Le rendu d'une page ou d'un article passe donc lui aussi par l'utilisation d'un template tag particulier. Cependant, il doit suivre un processus prévu par WordPress qui est appelé **la boucle de rendu**, consistant en un appel successif de fonctions qui permettent

la récupération des informations au sein d'une boucle.

La boucle de rendu la plus simple fait appel à au moins trois fonctions différentes de WordPress afin de parcourir et afficher les articles récupérés.

```
1 <?php
2 while (have_posts()) :
3     the_post();
4     the_content();
5 endwhile; ?>
```

Pour la simplicité du code, il n'y a pas de HTML dans les templates, mais vous en aurez très probablement besoin dans un vrai thème pour obtenir le design de votre choix.

La fonction have_posts() renvoie un booléen pour savoir s'il reste des objets à afficher sur la page en cours. Par exemple, lorsque l'on cherche à lister les derniers articles d'un blog, la fonction renverra `true` tant que tous les articles récupérés dans la base de données n'auront pas été affichés.

À l'intérieur de la boucle, **la fonction the_post()** effectue la récupération d'un article. À chaque appel de cette fonction, un curseur interne à WordPress sélectionne l'article suivant dans la liste de ceux que l'on a récupérés. Lorsque le curseur arrivera au dernier article, alors la méthode have_posts() renverra `false` et la boucle se terminera.

Une fois que le curseur est positionné sur un article, il faut appeler **la fonction the_content()** qui se charge de l'affichage du contenu de l'article en lui même. Cette fonction est donc conçue pour fonctionner à l'intérieur de la boucle de rendu, car elle est associée au rendu d'un article donné. De même, l'utilisation de **la fonction the_title()** au sein de la boucle permettra d'afficher le titre de l'article en cours.

Pourquoi utiliser une boucle lorsque l'on sait que l'on aura un seul objet, par exemple une page ?

En effet, utiliser la boucle de rendu pour une page peut paraître lourd alors que l'on sait à l'avance qu'il n'y aura qu'un seul passage. N'oubliez cependant pas que WordPress a un système de fallback (c'est-à-dire une procédure par défaut) pour l'utilisation des templates. Il est donc possible (et c'est souvent le cas) qu'un template donné soit utilisé pour le rendu des pages et des listes d'articles, qui seront quant à eux plusieurs sur une même page HTML. Utiliser la boucle permet donc d'avoir des templates plus génériques et qui fonctionneront quel que soit l'entité que l'on cherche à afficher.

Les commentaires

Avec l'affichage des publications viennent aussi les commentaires de vos visiteurs, que vous aurez besoin d'intégrer dans votre thème. À nouveau, tout se joue avec l'utilisation

des bons template tags à l'endroit où vous désirez afficher les commentaires. Tout d'abord, la fonction comments_template(), appelée dans la boucle de rendu, inclut le fichier comments.php que nous verrons plus bas et qui se charge de la mise en forme des commentaires. Ensuite, la fonction comment_form() crée le formulaire pour les utilisateurs voulant ajouter un nouveau commentaire à une publication. La boucle de rendu se complète donc encore un peu avec ces nouvelles fonctions.

```
<?php
while (have_posts()) :
  the_post();
  the_content();

  comments_template();
  comment_form();
endwhile; ?>
```

Le fichier comments.php, si l'on met de côté la mise en forme, n'a besoin que d'appeler wp_list_comments() pour afficher les commentaires associés à l'article. Cette fonction se charge de l'itération au travers de la liste des commentaires associés la publication actuellement parcourue par la boucle de rendu.

```
<ol>
  <?php wp_list_comments(); ?>
</ol>
```

Les filtres

Même si nous avons à notre disposition toutes les fonctions nécessaires pour récupérer les contenus du site, il y a fort à parier pour que vous vouliez, à un moment ou un autre, modifier la valeur de retour d'une fonction native de WordPress.

Prenons la fonction the_title(), qui permet d'obtenir le titre de l'article ou de la page en cours. Supposons que vous ne vouliez afficher au maximum que les 50 premiers caractères du titre si celui-ci est trop long. Regardons le prototype de la fonction (dans le fichier wp-includes/post-template.php) :

```
<?php function the_title($before = '', $after = '', $echo =
    true)
```

Le fonction permet de rajouter du code HTML avant et après le titre avec les deux premiers paramètres, tandis que le troisième indique si elle doit retourner la valeur du titre ou l'afficher elle-même via un echo.

Ce dernier argument pourrait être utilisé (en choisissant la valeur false) pour faire un traitement à la suite de la récupération du titre. En revanche, ce traitement particulier devrait être exécuté pour chaque template qui utilise cette fonction : c'est lourd et cela charge les templates de traitements supplémentaires.

Pour éviter cela, WordPress propose un système de filtres sur lesquels vous pouvez brancher des fonctions effectuant un traitement comme celui que l'on désire obtenir.

Appeler un filtre

Pour bien comprendre comment WordPress vous permet d'ajouter un traitement particulier à un endroit du code, il faut voir comment les filtres sont appelés.

La fonction apply_filters() déclenche l'application de toutes les fonctions rattachées à une clé de filtre donnée. Cette clé de filtre est le premier argument de la fonction, suivi par les paramètres envoyés.

Si nous revenons à la fonction the_title(), nous voyons qu'elle fait elle-même un appel à la fonction get_the_title() située dans le même fichier (wp-includes/post-template.php). C'est cette dernière qui exécute apply_filters() avant de renvoyer la valeur du titre le l'article que l'on désire obtenir. Cet appel contient trois arguments :

```
1 <?php apply_filters( 'the_title', $title, $id );
```

La clé du filtre est donc the_title, tandis que deux arguments sont passés aux fonctions de retour : le titre de l'article et son ID dans la base de données. À partir de ces données qui seront transférées au filtre, vous avez les moyens d'appliquer vos traitements.

Brancher un filtre

Pour brancher une fonction de filtre, nous utilisons la fonction add_filter() prenant deux paramètres : le nom du filtre et une fonction de rappel lorsque le filtre doit être exécuté.

Pour cela, vous devrez utiliser le fichier functions.php (créez-le si vous ne l'avez pas encore) de notre thème et qui est inclus automatiquement par WordPress s'il existe.

```
1 <?php add_filter('the_title', 'truncate_long_title');
```

Bien entendu il faut maintenant déclarer la fonction truncate_long_title() pour que le filtre fonctionne. Faites cela à la suite, toujours dans functions.php.

```
1 <?php
2 function truncate_long_title($title)
3 {
4     if (strlen($title) > 50) {
5         $title = substr($title, 0, 50).'...';
6     }
7     return $title;
8 }
```

Maintenant, si vous créez un article avec plus de 50 caractères dans son titre, il sera coupé.

Ce qu'il faut vraiment retenir ici, c'est que nous avons pu étendre une fonctionnalité de WordPress sans pour autant avoir modifié le code du cœur de l'application : tout le code supplémentaire se trouve uniquement dans le thème que nous avons créé. C'est un principe de développement auquel il faut se tenir si l'on veut proposer du code maintenable et pouvant rester fonctionnel même avec de nouvelles versions du moteur.

Ajouter des templates personnalisés

Nous avons un dernier point important à éclaircir pour conclure cet aperçu des thèmes de WordPress. Jusque là, vous savez comment déclarer des fichiers de thème dont le nom est défini par WordPress (comme content.php ou footer.php) et comment les surcharger grâce aux thèmes enfants. Mais comment utiliser des fichiers totalement personnalisés qui ne sont utilisables que sur votre site et dont le choix du nom sera arbitraire ?

On ne peut pas simplement utiliser la fonction require de PHP ?

C'est une solution envisageable, mais qui manque beaucoup de souplesse, car vous devrez indiquer le chemin du fichier lors de l'appel.

```
1 <?php
2 require __DIR__ . '/mon-template.php';
3 // ou mieux :
4 require STYLESHEETPATH . '/mon-template.php';
```

La constante STYLESHEETPATH correspond au chemin du répertoire de votre thème, c'est-à-dire le dossier dans lequel se situe le fichier style.css. Il n'est donc pas nécessaire de connaître le nom du thème pour avoir le chemin des fichiers.

Malheureusement avec ce fonctionnement, nous perdons tout le mécanisme de fallback de WordPress. En effet, si le fichier n'existe pas, nous aurons une erreur d'exécution lors de la tentative d'inclusion. De plus, dans le premier cas, il n'est pas possible pour un thème enfant de surcharger uniquement le fichier mon-template.php, il faudra aussi surcharger les appels. Et dans le second cas d'inclusion, le fichier devrait obligatoirement être redéfini dans le thème enfant, ce qui va à l'encontre du système d'héritage de thème.

Pour résoudre ce problème, il existe la fonction get_template_part() définie dans le fichier wp-includes/general-template.php comme suit :

```
1 <?php function get_template_part( $slug, $name = null )
```

Cette fonction inclut le fichier de template de la forme $slug-$name.php, ou $slug.php si $name vaut null ou que le premier n'existe pas, tout en conservant le mécanisme de fallback.

Ainsi, l'appel à notre fichier spécifique devient tout simplement :

```
1 <?php
2 get_template_part( 'mon-template.php' );
```

N'hésitez pas à vous servir de l'argument $name pour cette fonction si la fonctionnalité spécifique que vous désirez coder doit comporter plusieurs templates. De cette façon, vous pouvez regrouper sous le même préfixe (l'argument $slug) plusieurs templates similaires.

En résumé

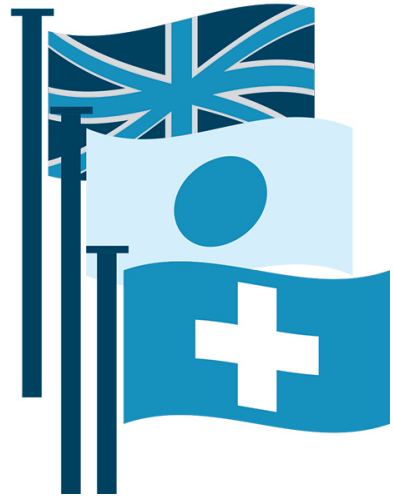
— La boucle de rendu est chargée d'afficher l'ensemble des articles correspondant à une page donnée du site.
— Les fonctions de filtres permettent de modifier le contenu de la valeur de retour de certaines fonctions, comme le titre de la page ou le contenu d'un article.
— Vous pouvez inclure des fichiers spécifiques avec la fonction get_template_part().

Chapitre 10

L'internationalisation

Difficulté :

Lorsqu'un site peut être visible de tous ou que l'on développe des fonctionnalités (thèmes ou plugins) pouvant être réutilisées par n'importe qui dans le monde, il est essentiel de penser à la traduction des textes qui seront affichés sur l'écran des utilisateurs. Pour cela, il faut préparer la traduction des chaînes de caractères en utilisant des fonctions de traduction, puis réaliser les traductions de vos textes dans les langues que vous souhaitez proposer.



Les fonctions de traduction

Traduire un texte

La traduction d'un texte passe toujours par une fonction de traduction dont le travail est de trouver la correspondance entre une chaîne de caractères dans une langue donnée (généralement l'anglais) et la chaîne traduite dans une autre langue. La chaîne originale est appelée **clé de traduction**. C'est elle qui sera écrite dans le code PHP et que l'on traduira vers le langage désiré dans des fichiers annexes contenant l'ensemble des traductions possibles.

Comment WordPress détermine-t-il la langue de l'utilisateur ?

La langue dans laquelle traduire le texte est déterminée par la **locale** de l'application. C'est un code permettant de déterminer le pays et la région parlant la langue choisie. Par exemple, la locale de la France est fr_FR, celle des États-Unis est us_EN. La locale de votre application WordPress est inscrite dans le fichier wp-config.php, lorsque la constante WPLANG est définie.

```
1 <?php
2 define('WPLANG', 'fr_FR');
```

Pour traduire une clé donnée, WordPress utilise une fonction nommée __() (deux underscores successifs) qui prend comme paramètre la clé de traduction. Sa valeur de retour est le texte traduit dans la langue souhaitée. Il est aussi possible d'utiliser la fonction _e(), qui fonctionne exactement de la même façon que la précédente, mais qui utilise un echo pour afficher directement le résultat de la traduction, au lieu de la renvoyer à l'aide d'un return. Elle est donc très adaptée à une utilisation dans les fichiers de thèmes.

En reprenant le fichier footer.php que nous avions modifié plus tôt dans notre thème, nous pouvons placer le texte (en anglais maintenant) spécifique à notre thème dans une fonction de traduction.

```
1 <?php _e('Zero theme, copy is forbidden.'); ?>
```

Le texte n'est pas traduit quand j'affiche la page, c'est normal ?

C'est parfaitement normal : nous n'avons indiqué nulle part comment traduire le texte. Dans le cas où la fonction de traduction ne trouve pas de correspondance entre la clé de traduction et la locale demandée, elle renvoie directement la clé. Ainsi, si l'on demande la traduction d'une clé qui n'a pas été intégrée dans les fichiers de traductions, aucune modification ne sera effectuée. Il faudra donc absolument rajouter la correspondance dans la langue choisie !

Le domaine de traduction

Les fonctions de traduction __() et _e() prennent en plus de la clé un second paramètre optionnel correspondant au domaine de traduction. Le domaine de traduction permet de spécifier plus précisément l'origine de la traduction que vous utilisez, notamment si celle-ci est spécifique à un thème ou un plugin donné. Dans ce cas, vous pourrez placer toutes les traductions du domaine dans un fichier unique, sans avoir à modifier d'autres fichiers de traductions.

Par exemple, dans le thème TwentyThirteen, le texte « Laisser un commentaire » visible sous les articles est déclaré ainsi :

```
1 <?php
2 __( 'Leave a comment', 'twentythirteen' )
```

Pour être utilisé, le domaine doit auparavant être déclaré, par exemple dans le fichier functions.php de votre thème, à l'aide de la fonction load_theme_textdomain(). Cette fonction attend de recevoir le domaine à déclarer et le chemin vers les fichiers de traduction pour ce domaine. Par convention, le chemin choisi est un dossier languages dans le dossier du thème. Le thème TwentytThirteen procède donc lui aussi de cette manière :

```
1 <?php
2 load_theme_textdomain( 'twentythirteen', get_template_directory
    () . '/languages' );
```

Si les fichiers de traduction ne sont pas trouvés dans le dossier déclaré, comme c'est le cas pour TwentyThirteen, alors WordPress ira les chercher dans le dossier wp-content/languages/themes.

Ajouter des traductions

Pour ajouter de nouvelles traductions, nous devons générer les fichiers qui contiendront les associations entre les clés de traduction et les textes traduits dans chaque langue. Les fichiers dont WordPress a besoin sont des fichiers MO (Machine Object) écrits en langage binaire et donc illisibles pour nous. Heureusement, nous pouvons écrire les fichiers de traduction dans un format lisible puis les convertir en fichiers MO pour WordPress.

Utiliser Poedit

Nous allons utiliser le logiciel Poedit pour gérer les traductions de notre thème car il a l'avantage d'être disponible sous Windows, Mac et Linux. De plus, il est relativement simple à l'utilisation. Vous pouvez le télécharger sur le lien indiqué dans le code web suivant :

▷ Télécharger Poedit
Code web : 702504.

Une fois téléchargé et installé, lancez Poedit. Vous devez, comme sur la figure 10.1, arriver sur une interface très minimaliste.

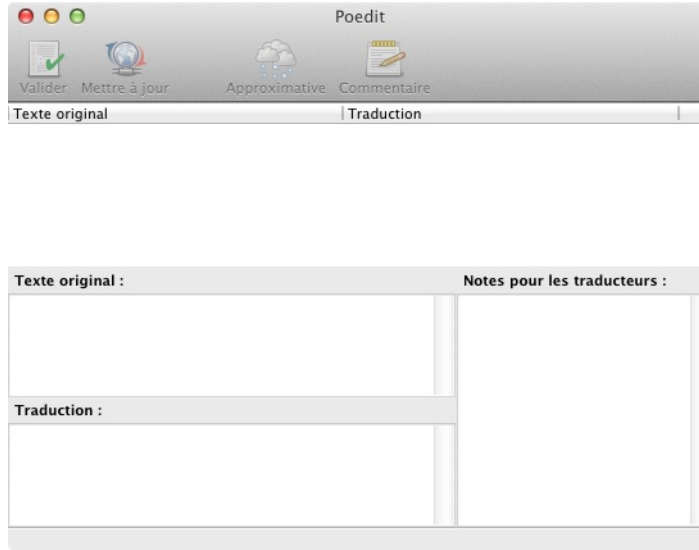


FIGURE 10.1 – L'accueil de Poedit

La première chose à faire est de créer un « Catalogue », c'est ce qui correspondra à un domaine dans WordPress. Ouvrez le menu « Fichier > Nouveau catalogue », puis ouvrez l'onglet « Chemins des sources » dans la fenêtre qui s'ouvre. Poedit est capable de lire les fichiers PHP à la recherche de fonctions de traduction. Il vous propose donc d'indiquer le chemin de vos fichiers PHP pour les lire à la recherche de vos clés de traduction. Cliquez donc sur le bouton « Nouvel élément », puis écrivez le chemin complet vers votre thème dans la zone de texte (voir la figure 10.2).

Allez ensuite dans « Mots-clés source » et ajoutez le nom des deux fonctions permettant d'effectuer des traductions dans WordPress : _e (underscore puis « e ») et __ (deux underscores). Cela permet à Poedit de savoir quelle fonction chercher dans le code pour récupérer les clés de traduction (voir la figure 10.3).

Choisissez ensuite « Accepter » et vous devriez voir la liste des textes traduisibles dans le thème choisi (voir la figure 10.4).

Il ne vous reste plus qu'à sélectionner la ligne à traduire puis à écrire la traduction de chaque clé dans la case « Traduction » en bas. Une fois terminé, allez dans « Fichier > Enregistrer sous... » pour sauvegarder les traductions au format PO ainsi qu'une copie au format MO.

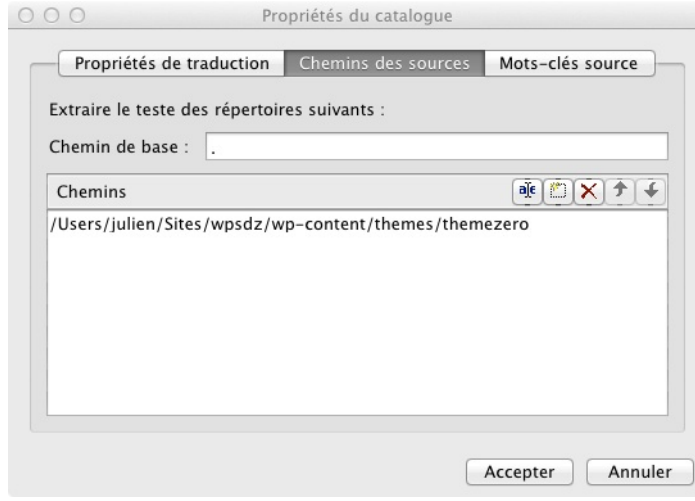


FIGURE 10.2 – Le chemin vers vos fichiers sources

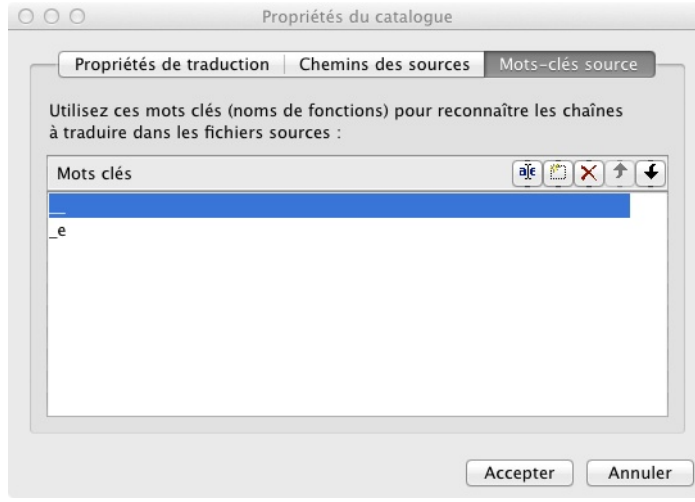


FIGURE 10.3 – Les noms des fonctions de traductions

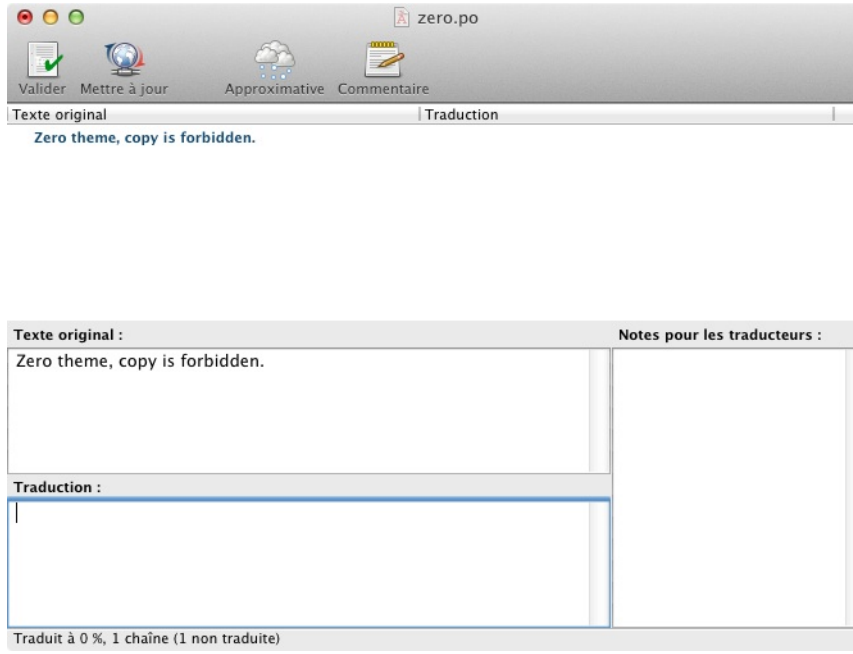


FIGURE 10.4 – Affichage de la liste des clés de traduction

Si vous souhaitez modifier des traductions après avoir déjà exporté les fichiers, vous pouvez rouvrir le fichier PO puis le sauvegarder avec vos changements, le fichier MO sera lui aussi mis à jour.

Utiliser les traductions dans un thème

Pour que le fichier de traductions soit utilisé dans le thème, nous devons placer le fichier MO dans le dossier wp-content/themes/zero/languages. Le nom du fichier MO doit être de la forme locale.mo, par exemple fr_FR.mo.

Il ne manque plus que la déclaration du domaine dans le fichier functions.php, et l'appel de celui-ci dans les fonctions de traduction.

```
1 <?php
2 load_theme_textdomain( 'zero', get_stylesheet_directory() . '/
    languages' );
```

```
1 <?php _e('Zero theme, copy is forbidden.', 'zero'); ?>
```

En résumé

— La traduction est traitée par les fonctions __() et _e().
— Les valeurs d'une clé de traduction dans une langue donnée doivent être écrites dans des fichiers PO compilés au format MO.
— Le domaine de traduction permet de regrouper les traductions, notamment pour les thèmes et les plugins.

Chapitre 11

TP : Personnalisez votre thème

Difficulté :

Nous avons d'ores et déjà parcouru de nombreuses fonctionnalités de WordPress pour lesquelles il vous est possible de faire des développements spécifiques. L'objectif étant d'adapter l'application au design que vous désirez. Ce TP est l'occasion de mettre en pratique ces notions, en créant votre premier thème original !

Présentation de l'exercice

Consignes

Pour cet exercice, vous devez créer un thème original sans base de départ et sans utiliser l'héritage de thème. De plus, votre thème devra inclure les éléments suivants :

— au moins un emplacement pour le menu ;
— au moins une zone de widget ;
— le support des pages, articles et commentaires ;
— les différents éléments doivent être judicieusement séparés en utilisant les template tags ;
— les portions de texte faisant partie du design doivent pouvoir être traduites.

En plus de ces points, vous pouvez, si vous vous sentez à l'aise, utiliser le système de fallback de WordPress en créant un affichage spécifique pour une page catégorie (avec le fichier category.php), ou bien pour les publications de type « Page » (avec un fichier page.php).

Si vous êtes en mal d'inspiration, voici (dans la figure 11.1) la page d'accueil du thème réalisé pour cet exercice et qui inclut les éléments demandés.

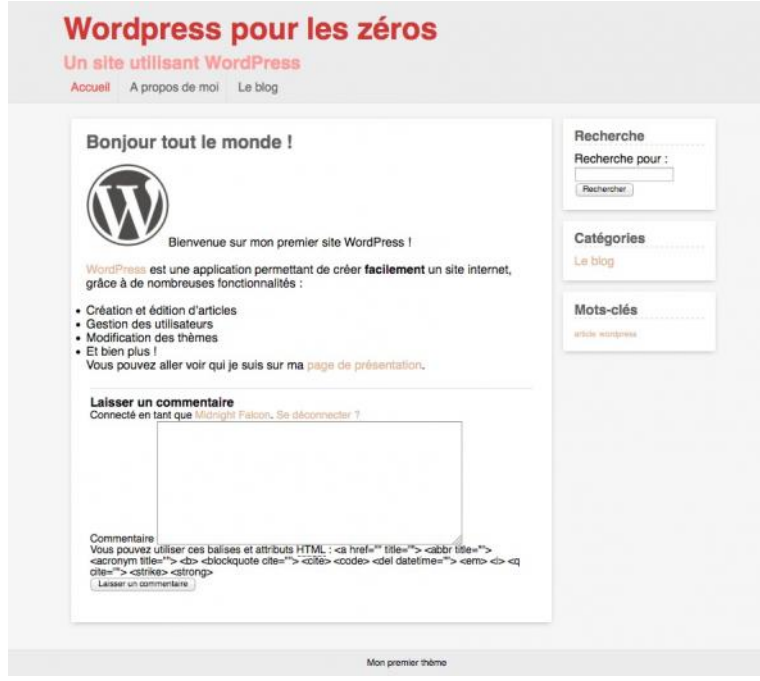


FIGURE 11.1 – Exemple de réalisation

Indications

Vous ne savez pas par où commencer ? Voici un exemple de marche à suivre, que vous pouvez appliquer :

— déclarer le nouveau thème avec les fichiers index.php et style.css ;
— mettre en place les différents fichiers PHP dont vous aurez besoin ;
— construire le fichier index.php petit à petit en testant et vérifiant régulièrement que les éléments se comportent comme prévu.

Enfin, prenez le temps d'implémenter chaque fonctionnalité demandée en y allant progressivement. Rien ne presse, l'important étant d'arriver au bout !

Correction

Déclarer les emplacements du menu et du widget

Commençons par le fichier functions.php, afin de déclarer tout de suite un emplacement pour le menu et une zone de widget, que nous utiliserons par la suite dans le thème.

```
1  <?php
2  add_action('widgets_init','tp_add_sidebar');
3  function tp_add_sidebar()
4  {
5    register_sidebar(array(
6      'id' => 'zone_widget_droite',
7      'name' => 'Zone latérale droite',
8      'description' => 'Apparait sur la droite site',
9      'before_widget' => '<aside>',
10     'after_widget' => '</aside>',
11     'before_title' => '<h1>',
12     'after_title' => '</h1>'
13   ));
14 }
15
16 add_action('init', 'tp_add_menu');
17 function tp_add_menu()
18 {
19   register_nav_menu('main_menu', 'Menu principal');
20 }
```

Gestion des éléments affichés

Nous continuons ensuite avec index.php, dont le rôle est de rassembler la gestion de tous les éléments affichés sur le site. Nous devons appeler l'en-tête, la barre latérale pour les widgets et le pied de page. Au milieu, nous ajoutons la boucle de rendu qui se charge d'afficher les publications en fonction de la page sur laquelle nous nous trouvons.

```
<?php get_header(); ?>
<div class="container">
<div class="content">
  <?php
  while (have_posts()) {
    the_post();
    get_template_part('content');
  }?>
</div>
<?php get_sidebar() ?>
</div>
<?php get_footer();
```

Vous pouvez noter l'utilisation d'un template spécifique pour l'affichage du contenu des publications grâce à la fonction get_template_part().

Dans le fichier header.php, nous ajoutons le début du code HTML pour nos pages, avec l'inclusion du fichier CSS, le titre du blog ainsi que son slogan avec la fonction bloginfo(), et le menu qui a été déclaré plus haut.

```
<!doctype html>
<html>
  <head>
    <title><?php echo wp_title() ?></title>
    <link rel="stylesheet" href="wp-content/themes/themetp/
        style.css" type="text/css"/>
  </head>
<body>
<header class="header">
  <div class="container">
    <h1><?php bloginfo() ?></h1>
    <h2><?php bloginfo( 'description' ); ?></h2>
    <?php wp_nav_menu(array('theme_location' => 'main_menu'))
        ?>
  </div>
</header>
```

Le pied de page est ici très simple. Vous pouvez y ajouter une mention de l'auteur et le nom de votre thème par exemple. Pensez à donner la possibilité de traduire le texte avec la fonction _e() !

```
<footer>
  <p><?php _e('Mon premier thème') ?></p>
</footer>
</body>
</html>
```

Pour afficher les widgets, nous passons par le fichier sidebar.php, appelé grâce à la fonction get_sidebar() depuis index.php.

```
<div class="widgets">
  <?php dynamic_sidebar('zone_widget_droite'); ?>
```

```
</div>
```

Il nous reste maintenant à afficher le contenu des publications en lui-même. Faisons cela dans le fichier content.php et affichons le titre de chaque article, son contenu puis les commentaires associés, ces derniers étant traités dans comments.php.

```
<article class="post">
  <h1><?php the_title() ?></h1>
  <div>
    <?php the_content(); ?>
  </div>
  <?php comments_template() ?>
  <?php comment_form(); ?>
</article>
```

```
<ol>
  <?php wp_list_comments(); ?>
</ol>
```

Vous êtes maintenant capables de mettre en place (ou de modifier) un thème WordPress pour l'adapter aux besoin d'un site. N'oubliez pas que l'apparence est la première chose que l'on voit lorsque l'on arrive sur une page web. Il est donc important d'y apporter du soin, notamment avec les règles de formatage CSS !

Corrigé type

Vous trouverez donc ci-dessous le fichier CSS utilisé pour réaliser l'illustration d'exemple présentée avec les consignes de ce TP.

N'hésitez pas à aller sur le code web suivant pour l'affichage de ce code :

▷ Corrigé type
Code web : 207051

```
/*
Theme Name: Thème TP
*/

* {
  margin:0;
  padding:0;
  text-decoration:none;
  color:black;
  font-family:sans-serif;
}
body {background:#F5F5F5;}
h1,h2 {color:#555;}
a {color:#DDAA88}

.container {
```

```
17    margin:0 auto;
18    width:900px;
19  }
20  .header {
21    background:#E9E9E9;
22    border-bottom:1px solid #E0E0E0;
23    margin-bottom:20px;
24    padding-top:10px;
25  }
26  .header h1 {
27    color:#CC3030;
28    font-size:40px;
29    margin:10px 0;
30  }
31  .header h2 {
32    color:#FF9090;
33    font-size:24px;
34  }
35  .content {
36    display:inline-block;
37    width:650px;
38  }
39  .menu li {
40    border-right:1px solid #DDD;
41    display:inline-block;
42    padding:10px;
43    list-style:none;
44  }
45  .menu li:last-child {border:none;}
46  .menu a {color:#555;}
47  .current-menu-item a {color:#E44;}
48
49  .widgets {
50    display:inline-block;
51    width:200px;
52    vertical-align:top;
53  }
54  .widgets li {list-style:none;}
55  .widgets aside {
56    background:#FFF;
57    border-radius:3px;
58    box-shadow:0 2px 6px -2px #999;
59    margin:0 0 15px;
60    padding:15px;
61  }
62  .widgets aside h1 {
63    border-bottom: 1px dashed #CCC;
64    color:#555;
65    margin-bottom:10px;
66    font-size:18px;
```

```
67 }
68
69 .post {
70   background:#FFF;
71   border-radius:3px;
72   box-shadow:0 2px 6px -2px #999;
73   margin:0 10px 15px;
74   padding:20px;
75 }
76 .post h1 {margin-bottom: 15px;}
77 .comment-respond {
78   border-top:1px solid #DDD;
79   font-size:14px;
80   margin:20px 5px;
81   padding-top:10px;
82 }
83 footer {
84   text-align:center;
85   font-size:12px;
86   background:#E9E9E9;
87   border-top:1px solid #E0E0E0;
88   margin-top:20px;
89   padding:10px 0;
90 }
```

Troisième partie

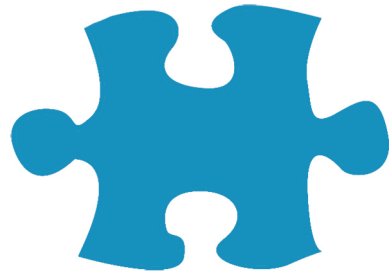
Développer un plugin complet

Chapitre 12

Créer des plugins

Difficulté : 

Après avoir vu thèmes, nous aborderons ici le second point majeur de WordPress : les plugins. Le support de ces modules additionnels a pour objectif de vous offrir un moyen sans limite de rajouter de nouvelles fonctionnalités à WordPress, qu'elles soient spécifiques à votre site ou bien réutilisables les autres. Grâce aux plugins, vous pourrez ajouter tout ce dont vous avez besoin. Après avoir abordé les principes de la création d'un plugin sous WordPress, nous suivrons pas à pas les étapes de création à travers une étude de cas.



Déclarer le plugin

De même que les thèmes, les plugins ont une structure de base très simple. Seul un minimum d'actions doit être exécuté pour pouvoir les déclarer, et c'est ensuite à vous d'y rajouter des fonctions pour compléter leur structure.

Les plugins doivent tous être situés dans le dossier wp-content/plugins, et peuvent être constitués d'un ou plusieurs fichiers.

Tout d'abord, créons un dossier zero dans lequel seront placés les fichiers PHP du plugin, ainsi qu'un premier fichier zero.php.

Ce premier fichier ne doit pas obligatoirement être nommé comme le dossier parent, mais c'est préférable par convention.

Pour que le plugin soit reconnu par WordPress, il faut nécessairement déclarer au moins son nom dans le fichier principal. Nous faisons cela via un commentaire au début du fichier.

```
1 <?php
2 /*
3 Plugin Name: Nom du Plugin
4 */
```

Si vous souhaitez donner plus d'informations à propos de votre plugin, notamment si vous souhaitez le distribuer, il existe d'autres informations qu'il est recommandé d'indiquer à la suite du nom.

- Plugin URI : l'adresse du site de votre plugin décrivant son fonctionnement et donnant des informations complémentaires ;
- Description : un paragraphe de description du plugin, affiché en parcourant la liste des plugins ;
- Version : la version du plugin ;
- Author : votre nom ;
- Author URI : l'adresse de votre site ;
- License : un nom de licence pour le code du plugin.

Nous pouvons donc avoir un en-tête comme celui-ci :

```
1  <?php
2  /*
3  Plugin Name: Zero plugin
4  Plugin URI: http://zero-plugin.com
5  Description: Un plugin d'introduction pour le développement
       sous WordPress
6  Version: 0.1
7  Author: Midnight Falcon
8  Author URI: http://votre-site.com
9  License: GPL2
10 */
```

Votre plugin est maintenant reconnu par WordPress ! Pour le vérifier, allez dans l'interface d'administration. Vous devez le voir apparaître dans la liste des extensions. Si c'est le cas, activez-le (voir la figure 12.1) !

FIGURE 12.1 – Le plugin est reconnu par WordPress

Nos premières fonctions

Écrivons maintenant notre première fonction dans ce plugin. Tout ce qui est déclaré dans le plugin peut être appelé depuis n'importe quel autre endroit du code, que ce soit depuis un thème ou un autre plugin. Il est donc tout à fait possible pour un plugin d'utiliser les fonctions d'un autre plugin, si celui-ci a été activé sur votre site.

Malgré cela, il est préférable d'utiliser au maximum le système modulaire de WordPress pour appeler vos fonctions, comme nous l'avons fait plus tôt avec les filtres et la fonction add_filter(). De cette façon, il n'est pas nécessaire d'appeler les fonctions du plugin en dehors de celui-ci, ce qui limite les dépendances et les besoins de créer des appels depuis l'extérieur. En particulier, utiliser les filtres vous permet de rendre votre plugin compatible avec n'importe quel thème de votre site, puisqu'il enregistra de lui-même les actions qu'il doit effectuer.

Rajouter un filtre simple

Notre première tâche sera de modifier sur toutes les pages l'attribut `<title>` pour indiquer que le plugin est activé. Pour ce faire, le filtre wp_title, appelé avec la fonction the_title(), est idéal pour nous.

```
1 <?php
2 add_filter('wp_title', 'zero_modify_page_title', 20) ;
3 function zero_modify_page_title($title) {
4     return $title . ' | Avec le plugin des zéros !' ;
5 }
```

Pour éviter au maximum les conflits dans les noms de fonctions avec un autre plugin, il est conseillé de les préfixer, ici avec zero.

Dorénavant, tant que le plugin sera activé, toutes les pages verront leur titre mis à jour comme écrit dans la fonction. En allant par exemple sur une page intitulée « À propos de nous », nous trouvons le code HTML suivant :

```
<head>
    <!-- ... -->
    <title>À propos de nous | Wordpress pour les zéros | Avec
        le plugin des zéros !</title>
```

Utiliser une structure objet

Afin d'utiliser toute la puissance de PHP dans notre plugin, nous allons structurer notre plugin à l'aide de classes. Ce n'est bien entendu pas obligatoire, un plugin peut tout à fait fonctionner avec des fonctions dans l'espace global, mais la programmation objet permet de mieux organiser une application de grande ampleur, nous allons donc tout de suite restructurer le code de notre plugin en ce sens.

Nous créons donc dans le fichier zero.php une classe Zero_Plugin avec pour méthode le filtre que nous avions utilisé précédemment.

```
<?php
class Zero_Plugin
{
    public function __construct()
    {
        add_filter('wp_title', array($this, 'modify_page_title'
            ), 20) ;
    }

    public function modify_page_title($title)
    {
        return $title . ' | Avec le plugin des zéros !' ;
    }
}

new Zero_Plugin();
```

Comme vous le voyez, l'utilisation de la fonction add_filter() a été légèrement modifiée pour la spécification de la fonction de rappel. Étant donné que nous avons déplacé la fonction modify_page_title() dans un objet, il faut aussi fournir l'instance de l'objet pour pouvoir appeler la fonction ! Ceci se fait en passant comme argument un tableau avec l'instance de l'objet à utiliser et le nom de la méthode de classe à appeler.

Une structure multifichiers

Toujours dans le but de structurer votre plugin, vous aurez probablement besoin d'écrire du code dans différents fichiers, qui contiendront par exemple différentes classes si vous

choisissez d'utiliser des objets.

Dans une application PHP, l'utilisation de plusieurs fichiers passe par l'inclusion de ceux-ci au travers d'une directive include ou include_once. Au sein de WordPress, seul le fichier principal d'un plugin activé (celui qui déclare le nom du plugin) est inclus automatiquement au chargement de l'application. Tous les autres fichiers doivent donc être inclus par le développeur s'il désire les utiliser. Dans le cas contraire, les éventuelles classes et méthodes contenues ne seront pas utilisables.

La tâche de notre fichier zero.php sera donc plus particulièrement de charger les fichiers nécessaires au chargement du plugin. Pour illustrer cela, nous allons une dernière fois réorganiser la structure de notre code : une nouvelle classe Zero_Page_Title, dans un fichier pagetitle.php, contiendra le filtre de titre que nous avons créé et la classe Zero_Plugin se chargera d'instancier Zero_Page_Title (et toute autre classe que nous pourrons ajouter ultérieurement). De cette façon, nous avons un point d'entrée qui prend en charge les créations d'objets, eux-mêmes effectuant des traitements spécifiques, ce qui facilitera l'ajout de fonctionnalités.

Le code final des deux classes est donc le suivant.

```php
<?php
class Zero_Plugin
{
    public function __construct()
    {
        include_once plugin_dir_path( __FILE__ ).'/page_title.
            php';
        new Zero_Page_Title();
    }
}
```

```php
<?php
class Zero_Page_Title
{
    public function __construct()
    {
        add_filter('wp_title', array($this, 'modify_page_title'
            ), 20) ;
    }

    public function modify_page_title($title)
    {
        return $title . ' | Avec le plugin des zéros !' ;
    }
}
```

Un plugin complet

Il est maintenant temps d'aller plus loin dans la création de plugin en exploitant au maximum les possibilités offertes par WordPress. Nous allons pouvoir commencer à rentrer dans les sujets avancés de la création de plugin autour d'un objectif précis : la mise en place d'un plugin de newsletter, que nous créerons au fil des chapitres suivants.

Les objectifs

Le plugin que nous allons mettre en place est relativement simple. Il consiste à proposer aux visiteurs de s'inscrire sur le site en fournissant leur adresse email afin de recevoir une newsletter concernant l'actualité du site.

Plusieurs sujets entrent en œuvre, à commencer par la création d'un widget, affiché sur toutes les pages, dans le but de récupérer les adresses email. La base de données devra comporter une table supplémentaire afin de stocker la liste des adresses à qui il faut envoyer la newsletter. Enfin, l'interface d'administration permettra à l'administrateur de définir le contenu de cet email, son objet, l'adresse email de l'expéditeur et un bouton pour déclencher l'envoi.

Le chantier couvrira donc les aspects majeurs de WordPress, il est temps de s'y attaquer sans plus tarder !

La classe Zero_Newsletter

Afin de garder une structure cohérente dans notre module, nos allons créer une nouvelle classe Zero_Newsletter qui se chargera de tout le code concernant la fonctionnalité de newsletter. Plaçons-la dans un fichier newsletter.php, qui sera pour l'instant assez simple.

```
1 <?php
2 class Zero_Newsletter
3 {
4     public function __construct()
5     {
6     }
7 }
```

N'oubliez pas de modifier le constructeur de la classe Zero_Plugin pour instancier un nouvel objet Zero_Newsletter.

```
1 <?php
2 include_once plugin_dir_path( __FILE__ ).'/newsletter.php';
3 new Zero_Newsletter();
```

En résumé

— Tous les plugins sont placés dans le répertoire wp-content/plugins.
— Le plugin est déclaré avec un commentaire dans l'en-tête du fichier principal.
— Un plugin doit être de préférence découpé en fichiers suivant les fonctionnalités ajoutées.

Chapitre 13

Créer des widgets

Difficulté :

Comme nous avons pu le voir au début du cours, les widgets sont des éléments de base bien pratiques de l'interface graphique. Ils permettent de déplacer des blocs de fonctionnalités sur différentes zones de vos pages, mais pour être utilisables, encore faut-il en avoir défini le comportement dans le code auparavant !

Déclarer un widget

Commençons tout de suite par le widget à afficher sur le site pour l'inscription des emails.

Une nouvelle classe

Tous les widgets sont des objets qui doivent hériter de la classe WP_Widget (déclarée dans le fichier wp-includes/widgets.php) et surcharger au moins deux méthodes.

Avant tout, le constructeur de la classe WP_Widget doit être appelé par le constructeur de la classe fille afin de définir :

- un identifiant pour le widget ;
- un titre à afficher dans l'administration ;
- éventuellement des paramètres supplémentaires comme la description du widget (elle aussi affichée dans le panneau « widget » de l'administration).

```
<?php function __construct( $id_base, $name, $widget_options =
    array(), $control_options = array() )
```

La seconde fonction à surcharger est la méthode widget(), définie ainsi :

```
<?php function widget($args, $instance)
```

Le premier argument est un tableau contenant des paramètres d'affichage que nous détaillerons lors de l'implémentation de la méthode. Ils permettent notamment d'obtenir un rendu du widget qui correspond au thème graphique utilisé. Le second argument contient les paramètres du widget sauvegardés dans la base de données, c'est-à-dire les paramètres que l'administrateur a définis lors de l'ajout du widget à la zone dédiée.

```
<?php
public function __construct()
{
    parent::__construct('zero_newsletter', 'Newsletter', array(
        'description' => 'Un formulaire d\'inscription à la
        newsletter.'));
}

public function widget($args, $instance)
{
    echo 'widget newsletter';
}
```

La classe de notre widget est maintenant créée, il faut donc dire à WordPress que nous voulons l'utiliser, c'est-à-dire la déclarer en tant que widget.

Pour cela, on utilise la fonction register_widget(), qui prend en paramètre le nom de la classe du widget. Dans notre cas, l'appel sera le suivant :

```
<?php
register_widget('Zero_Newsletter_Widget');
```

Cependant, nous ne pouvons pas l'appeler à n'importe quel moment, il faut attendre que WordPress soit prêt à enregistrer l'existence des widgets. Ceci se fait grâce à une action widget_init ; nous devons donc faire un appel à add_action() pour la déclaration du widget.

Ainsi, en utilisant les fonctions anonymes de PHP, l'enregistrement du widget devient :

```
1 <?php
2 add_action('widgets_init', function(){register_widget('
    Zero_Newsletter_Widget');});
```

Afin de garder une structure cohérente dans notre module, nos allons créer une nouvelle classe Zero_Newsletter qui se chargera de tout le code concernant la fonctionnalité de newsletter. Plaçons-la dans un fichier newsletter.php, qui sera pour l'instant assez simple.

```
1  <?php
2  include_once plugin_dir_path( __FILE__ ).'/newsletterwidget.php
       ';
3
4  class Zero_Newsletter
5  {
6      public function __construct()
7      {
8          add_action('widgets_init', function(){register_widget('
               Zero_Newsletter_Widget');});
9      }
10 }
```

N'oubliez pas de modifier le constructeur de la classe Zero_Plugin pour créer un nouvel objet Zero_Newsletter.

```
1 <?php
2 include_once plugin_dir_path( __FILE__ ).'/newsletter.php';
3 new Zero_Newsletter();
```

Le widget est dorénavant déclaré et se retrouve visible dans la liste proposée par WordPress lorsque l'on se rend dans le menu « Apparence > Widgets » (voir la figure 13.1).

FIGURE 13.1 – Le widget Newsletter

Les paramètres

Vous avez probablement remarqué que le widget n'a aucun paramètre disponible dans l'interface d'administration. Pourtant, il faudrait au minimum pouvoir définir un titre à afficher en haut du widget pour qu'il soit identifiable parmi les autres.

L'affichage du paramétrage du widget est complètement délégué à notre classe en surchargeant la méthode form(), qui prend en paramètre un tableau contenant les valeurs précédemment enregistrées.

Dans le formulaire que nous allons créer, il est important d'utiliser, pour la génération des attributs id et name de vos champs, deux méthodes définies par WP_Widget qui sont respectivement get_field_id() et get_field_name(). Ces deux méthodes vont générer un identifiant et un nom unique pour chaque champ qui utilisés par WordPress lors de la sauvegarde des valeurs, il est donc très important de les utiliser, sans quoi l'enregistrement des paramètres ne pourra pas fonctionner.

Voici donc comment nous allons redéfinir la méthode form().

```
1  <?php
2  public function form($instance)
3  {
4      $title = isset($instance['title']) ? $instance['title'] : '
           ';
5      ?>
6      <p>
7          <label for="<?php echo $this->get_field_name( 'title' )
               ; ?>"><?php _e( 'Title:' ); ?></label>
8          <input class="widefat" id="<?php echo $this->
               get_field_id( 'title' ); ?>" name="<?php echo $this
               ->get_field_name( 'title' ); ?>" type="text" value="
               <?php echo  $title; ?>" />
9      </p>
10     <?php
11 }
```

Vous pouvez maintenant définir une valeur pour le titre, et vérifier que celle-ci est bien sauvegardée (voir la figure 13.2).

Le rendu final

Pour terminer notre widget, il faut maintenant modifier la méthode widget() afin d'afficher le paramètre titre et le champ d'enregistrement des emails.

La récupération d'un paramètre se fait au travers de la variable $instance contenant les paramètres du widget. Le titre est donc récupéré par l'instruction $instance['title']. Cela n'est cependant pas tout : il existe un filtre intitulé widget_title auquel peuvent se brancher des fonctions appliquant un traitement particulier pour les titres des widgets.

Il n'est bien sûr pas obligatoire de l'utiliser pour obtenir l'affichage du paramètre, mais

FIGURE 13.2 – L'ajout du widget dans une zone dédiée

gardez à l'esprit qu'un utilisateur de votre plugin pourrait vouloir utiliser ce filtre pour modifier l'affichage des widgets sur son thème. Il faut donc l'utiliser !

```
<?php
echo apply_filters('widget_title', $instance['title']);
```

Le formulaire d'enregistrement d'adresse sera très simple, car nous n'avons besoin que d'un seul champ.

```
<form action="" method="post">
    <p>
        <label for="zero_newsletter_email">Votre email :</label
            >
        <input id="zero_newsletter_email" name="
            zero_newsletter_email" type="email"/>
    </p>
    <input type="submit"/>
</form>
```

J'ai volontairement laissé le champ action du formulaire vide, afin que l'utilisateur reste sur la même page lors de son inscription. Nous verrons plus bas comment récupérer les informations postées.

```
<?php
public function widget($args, $instance)
{
    echo $args['before_widget'];
    echo $args['before_title'];
    echo apply_filters('widget_title', $instance['title']);
    echo $args['after_title'];
    ?>
    <form action="" method="post">
        <p>
            <label for="zero_newsletter_email">Votre email :</
                label>
            <input id="zero_newsletter_email" name="
                zero_newsletter_email" type="email"/>
```

```
13            </p>
14            <input type="submit"/>
15        </form>
16        <?php
17        echo $args['after_widget'];
18    }
```

Le widget est maintenant prêt et doit s'afficher correctement avec un champ d'enregistrement d'emails (voir la figure 13.3).

FIGURE 13.3 – Le widget est installé

En résumé

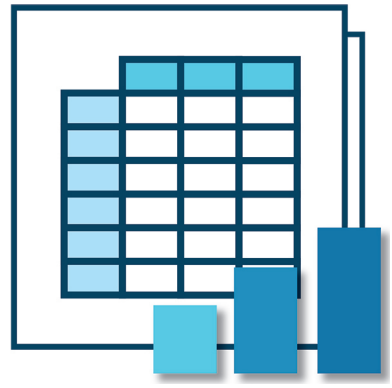
- Chaque nouveau widget doit être enregistré lors de l'événement widgets_init.
- Les widgets doivent hériter de la classe WP_Widget et surcharger la méthode widget().
- Le rendu du widget doit utiliser les paramètres fournis par le thème pour générer un code HTML cohérent avec celui-ci.

Chapitre 14

Modifier la base de données

Difficulté :

Pour aller plus loin, nous allons avoir besoin d'une table spécifique pour stocker les adresses email des abonnées à la newsletter. Cette table contiendra deux colonnes : un champ `id` pour la clé primaire et un champ `email` pour stocker l'adresse des inscrits.



Exécuter des requêtes SQL

Pour aller plus loin, nous allons avoir besoin d'une table spécifique pour stocker les adresses email des abonnées à la newsletter. Cette table contiendra deux colonnes :

— un champ id pour la clé primaire ;
— un champ email pour stocker l'adresse des inscrits.

Créer une nouvelle table

Pour créer la table contenant les adresses email, nous devons exécuter une requête SQL spécifique. Dans WordPress, l'accès à la base de données se fait à l'aide de la classe wpdb, qui contient de nombreuses méthodes pour interagir avec elle. Notamment, une méthode query() est destinée à exécuter les requêtes qui lui sont passées en paramètre.

Une instance de la classe wpdb est créée au chargement de l'application et stockée dans une variable globale. Sa récupération se fait donc depuis n'importe quel endroit du code de façon très simple :

```
1 <?php
2 global $wpdb ;
```

Dans la classe Zero_Newsletter, créons une fonction statique install() qui aura pour charge d'effectuer toutes les actions nécessaires lors de l'activation du plugin. Pour l'instant, seul un appel à la méthode wpdb : :query() est nécessaire.

```
1 <?php
2 public static function install()
3 {
4     global $wpdb;
5
6     $wpdb->query("CREATE TABLE IF NOT EXISTS {$wpdb->prefix}
          zero_newsletter_email (id INT PRIMARY KEY, email VARCHAR
          (255) NOT NULL);");
7 }
```

À quoi correspond la variable $wpdb->prefix dans la requête ?

Vous vous rappelez probablement que lors de l'installation, WordPress nous avait demandé de choisir un préfixe pour les tables de la base de données. L'attribut prefix de la classe wpdb contient la valeur de ce préfixe, qui peut être différent d'une installation de WordPress à l'autre. Par conséquent, nous devons l'utiliser pour déterminer le nom de la table que nous allons créer.

Il nous reste maintenant à appeler la méthode install() pour lancer la création de la table des adresses.

Tracer l'activation du plugin

Évidemment, la création de la table ne doit être effectuée qu'une seule fois, lorsque le plugin sera installé. Il nous faut donc un moyen de savoir que le plugin vient d'être installé, pour effectuer les modifications dans la base de donnée à cet instant uniquement.

Heureusement pour nous, WordPress déclenche des événements particuliers lorsqu'un plugin est activé, désactivé ou bien encore supprimé de l'application. C'est ici le premier cas qui nous intéresse mais les autres fonctionnent de façon similaire.

La fonction register_activation_hook(), définie dans wp-includes/plugin.php, permet d'ajouter une fonction à appeler lors de l'activation d'un plugin particulier.

Cette fonction doit être appelée dans le fichier principal du plugin (celui qui définit la valeur « Plugin Name » dans l'en-tête) et préciser le chemin du plugin ainsi que la fonction à exécuter à l'activation.

Pour passer le chemin du plugin, il suffit d'utiliser la constante __FILE__, indiquant le nom du fichier courant.

En interne, la fonction register_activation_hook() va transformer __FILE__ en une chaîne de la forme dossier_du_plugin/nom_du_plugin qui permet d'identifier le plugin de façon unique, dans notre cas zero/zero.

L'appel peut être placé directement dans le constructeur de la classe Zero_Plugin, il est finalement très similaire à ce que l'on avait pour l'ajout d'actions ou de filtres :

```
1 <?php
2 register_activation_hook(__FILE__, array('Zero_Newsletter', '
    install'));
```

La méthode install() étant statique, c'est le nom de la classe, et non une instance, que nous passons dans les paramètres.

Si votre plugin est déjà activé, désactivez-le puis réactivez-le à nouveau afin de déclencher la création de la table dans la base de données. Pour vérifier sa bonne exécution, nous pouvons aller dans phpmyadmin et voir sur la côté gauche la liste des tables de la base de données (voir la figure 14.1).

La désactivation et la désinstallation du plugin

En plus de l'activation, nous pouvons tracer la désactivation d'un plugin, ainsi que sa suppression pure et simple. Pour cela, il existe la fonction register_deactivation_hook(), dont l'appel est strictement identique à register_activation_hook(), mais vous n'en aurez pas l'utilité ici.

En revanche, nous devons penser à la désinstallation du plugin. En effet, si un utilisateur décide de supprimer le plugin de son installation WordPress, il faut aussi que la table que nous avons créée soit effacée de la base de données pour que celle-ci retrouve son

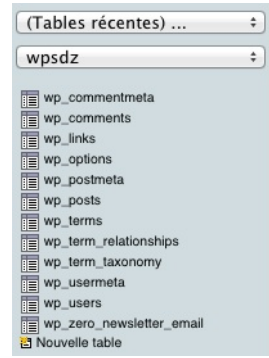


FIGURE 14.1 – Résultat de la table des adresses email

état original.

De même que pour l'installation, créons une fonction qui va cette fois supprimer la table des adresses.

```
1 <?php
2 public static function uninstall()
3 {
4     global $wpdb;
5
6     $wpdb->query("DROP TABLE IF EXISTS {$wpdb->prefix}
          zero_newsletter_email;");
7 }
```

Dans le cas de la désinstallation, la fonction appelée **doit** être statique (ou bien appartenir à l'espace global), ce qui n'était pas obligatoire pour l'activation.

Il ne reste qu'à appeler la fonction register_uninstall_hook() à la fin du constructeur de Zero_Plugin.

```
1 <?php
2 register_uninstall_hook(__FILE__, array('Zero_Newsletter', '
      uninstall'));
```

Maintenant, si vous supprimez le plugin à l'aide du lien « Supprimer » dans la page de gestion des plugins, la table des adresses sera elle aussi effacée.

L'insertion et la sélection

Il ne reste plus qu'une étape pour lier le widget et la base de données ; nous avons besoin d'enregistrer les adresses entrées par les visiteurs dans la table spécifiquement créée.

Pour cela, il nous faut avant tout une méthode save_email() se chargeant de l'ajout des adresses à la base de données. Cette méthode sera appelée sur toutes les pages du site et vérifiera la présence d'une variable zero_newsletter_email dans la superglobale $_POST. Si elle existe, la valeur contenue sera insérée dans la table dédiée.

Comme pour les requêtes précédentes, nous utilisons l'objet wpdb pour interagir avec la base de données. Avant d'effectuer l'insertion, nous devons vérifier si l'adresse proposée n'existe pas déjà dans la table, auquel cas nous arrêterions le processus. Nous utilisons pour cela la méthode wpdb : :get_row(), prenant en paramètre la requête SQL à exécuter.

```
1 <?php
2 if (isset($_POST['zero_newsletter_email']) && !empty($_POST['
    zero_newsletter_email'])) {
3         global $wpdb;
4         $email = $_POST['zero_newsletter_email'];
5
6         $row = $wpdb->get_row("SELECT * FROM {$wpdb->prefix}
              zero_newsletter_email WHERE email = '$email'");
```

La méthode get_row() retourne un tableau des colonnes de la première ligne de résultats de la requête, ou bien la valeur null si aucun résultat n'est trouvé.

WordPress se charge de protéger toutes les requêtes contre les failles de sécurité lorsqu'elles sont soumises via la méthode query(), elle-même appelée par get_row(). Il n'est donc pas nécessaire de vous charger de nettoyer les variables présentes dans vos requêtes.

Si la requête ne retourne aucun résultat, nous pouvons insérer la nouvelle adresse en passant par la méthode insert(). Le premier paramètre de celle-ci est le nom de la table dans laquelle on souhaite insérer une ligne, le second est un tableau associatif contenant les valeurs de la ligne pour chaque champ de la table.

```
1 <?php
2 if (is_null($row)) {
3     $wpdb->insert("{$wpdb->prefix}zero_newsletter_email", array
          ('email' => $email));
4 }
```

C'est tout ! La méthode save_email() est prête à enregistrer toutes les adresses de vos visiteurs ! Voici un récapitulatif de son contenu :

```
1 <?php
2 public function save_email()
3 {
4     if (isset($_POST['zero_newsletter_email']) && !empty($_POST
          ['zero_newsletter_email'])) {
5         global $wpdb;
6         $email = $_POST['zero_newsletter_email'];
7
8         $row = $wpdb->get_row("SELECT * FROM {$wpdb->prefix}
              zero_newsletter_email WHERE email = '$email'");
```

```
9           if (is_null($row)) {
10              $wpdb->insert("{$wpdb->prefix}zero_newsletter_email
                    ", array('email' => $email));
11          }
12      }
13  }
```

Nous n'avons pas géré la vérification du format de l'adresse email car le but reste ici de montrer comment traiter la valeur reçue et l'insérer en base. Dans une application réelle, il faudrait bien entendu vérifier que la valeur envoyée correspond bien à une adresse email.

Il faut maintenant utiliser une action pour connecter la méthode save_email() à l'affichage des pages du site. Pour cela, nous pouvons utiliser l'identifiant wp_loaded qui correspond à l'instant où l'application est chargée et où elle s'apprête à effectuer le rendu du thème pour la page demandée.

```
1  <?php
2  add_action('wp_loaded', array($this, 'save_email'));
```

En entrant une adresse email puis en validant le formulaire, nous restons sur la même page mais la table des adresses doit contenir l'adresse choisie (voir la figure 14.2).

FIGURE 14.2 – Enregistrement des adresses dans la table

En résumé

— Les requêtes SQL sont traitées par l'objet wpdb.
— L'API (Interface de programmation) fournit des méthodes pour créer facilement des requêtes de sélection, d'insertion et de mise à jour des entrées d'une table.

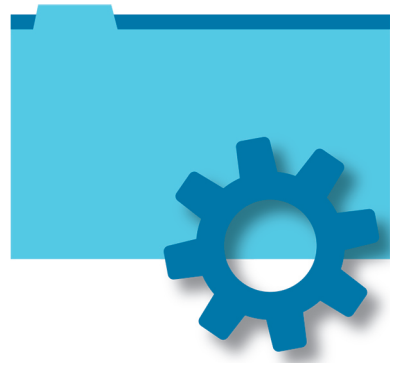
Chapitre 15

L'administration

Difficulté :

Les fonctionnalités de notre plugin sont presque complètes ! Il nous reste cependant un point important à finaliser : le plugin doit pouvoir être configuré dans l'administration, notamment pour lui définir des options mais aussi pour exécuter l'envoi de la newsletter lorsque l'administrateur le décide.

Il nous faut donc ajouter un menu spécifique pour la gestion du module, dans lequel nous disposerons des paramètres éditables ainsi que d'un bouton d'envoi des emails aux adresses enregistrées.



Ajouter des menus

Menu principal

Commençons par créer un élément de premier niveau qui apparaîtra dans le menu principal de l'administration, c'est-à-dire directement dans la colonne de gauche. Cet ajout se fait lors du chargement des menus de WordPress, un événement identifié par le déclenchement de l'action admin_menu. Dans le constructeur de la classe Zero_Plugin, il nous faut donc brancher une fonction add_admin_menu() que l'on définira plus bas.

```
1 <?php
2 add_action('admin_menu', array($this, 'add_admin_menu'));
```

La création d'un menu s'effectue avec la fonction add_menu_page(), qui peut prendre jusqu'à sept paramètres :

- le titre de la page sur laquelle nous serons redirigés ;
- le libellé du menu ;
- l'intitulé des droits que doit posséder l'utilisateur pour pouvoir accéder au menu. Si les droits sont insuffisants, le menu sera masqué ;
- la clé d'identifiant du menu qui doit être unique (mettre le nom du plugin est une bonne option) ;
- la fonction à appeler pour le rendu de la page pointée par le menu ;
- l'icône à utiliser pour le lien (vous pouvez laisser les valeurs par défaut) ;
- la position dans le menu (vous pouvez laisser les valeurs par défaut).

```
1 <?php
2 public function add_admin_menu()
3 {
4     add_menu_page('Notre premier plugin', 'Zero plugin', '
          manage_options', 'zero', array($this, 'menu_html'));
5 }
```

Une fois que cette fonction est définie, le menu doit apparaître dans l'interface d'administration (voir la figure 15.1).

FIGURE 15.1 – Le lien du plugin dans le menu

Il nous reste à définir la méthode menu_html() pour l'affichage de la page. Ce sera la page d'accueil du plugin, placez donc un contenu très simple simplement pour vérifier son bon fonctionnement (voir la figure 15.2).

```
1 <?php
2 public function menu_html()
3 {
4     echo '<h1>'.get_admin_page_title().'</h1>';
5     echo '<p>Bienvenue sur la page d\'accueil du plugin</p>';
6 }
```

La fonction get_admin_page_title() renvoie la valeur du premier argument donné à la fonction add_menu_page().

FIGURE 15.2 – La page principale du plugin

Les sous-menus

Ajoutons immédiatement un sous-menu « Newsletter » à notre élément du menu principal, afin de pouvoir gérer plusieurs sections si besoin.

Pour cela, nous devons passer par la fonction add_submenu_page(), qui comprend elle aussi de nombreux arguments. Le premier d'entre eux est l'identifiant du menu parent dans lequel devra apparaître le sous-menu, c'est donc la valeur que nous avions choisie précédemment. Notez que si vous voulez ajouter un sous-menu à l'un des menus natifs de WordPress, vous devrez ici mettre le nom du fichier PHP utilisé pour le rendu du menu en question, par exemple users.php pour le menu « Utilisateurs ».

Les détails sont disponibles sur la page de référence de la fonction indiquée sur le code web suivant :

▷ Page de référence
Code web : 317210

Sur cette même page de la documentation, vous pouvez voir qu'au lieu de spécifier le fichier PHP d'un menu natif, vous pouvez utiliser des fonctions dédiées qui appellent en interne add_submenu_page() avec les arguments adéquats. Par exemple, pour ajouter un sous-menu dans « Utilisateurs », il existe la fonction add_users_page().

Les arguments suivants de la fonction sont respectivement :

— le titre de la nouvelle page ;
— le libellé du menu ;
— les droits requis pour y accéder ;
— l'identifiant du sous-menu ;
— la fonction d'affichage.

Tout est donc très similaire à l'ajout d'un menu de premier niveau.

Nous allons déléguer la création du sous-menu à la classe Zero_Newsletter, il faut donc connecter une action à l'événement admin_menu et écrire la méthode correspondante.

```
1  <?php
2  class Zero_Newsletter
3  {
4      public function __construct()
5      {
6          //...
7          add_action('admin_menu', array($this, 'add_admin_menu')
               , 20);
8      }
9
10     public function add_admin_menu()
11     {
12         add_submenu_page('zero', 'Newsletter', 'Newsletter', '
               manage_options', 'zero_newsletter', array($this, '
               menu_html'));
13     }
14 }
```

Pourquoi avoir mis une priorité à 20 dans la méthode add_action() ?

Ici, la priorité augmentée (la valeur par défaut est 10) permet de s'assurer que la fonction sera exécutée après celle qui ajoute le menu parent du plugin. Il faut en effet que celui-ci ait été créé avant de pouvoir créer les sous-menus correspondants.

Rafraîchissez votre interface d'administration, vous devriez avoir, comme sur la figure 15.3, le sous-menu affiché au survol du menu parent (si le menu parent était déjà sélectionné, alors le sous-menu sera visible par défaut).

Deux menus on été créés ! Pourtant il n'y a qu'un seul appel à add_submenu_page() !

En effet, lors de la création d'un sous-menu, WordPress ajoute par défaut un premier

FIGURE 15.3 – Le sous-menu contient le nouvel élément

lien identique au menu parent. Vous pouvez cependant en changer le libellé en créant un sous-menu donc l'identifiant est égal à l'identifiant parent (voir la figure 15.4). Par exemple, nous pouvons ajouter une ligne juste après la génération du menu parent :

```
1 <?php
2 public function add_admin_menu()
3 {
4     add_menu_page('Zero plugin', 'Zero plugin', 'manage_options
          ', 'zero', array($this, 'menu_html'));
5     add_submenu_page('zero', 'Apercu', 'Apercu', '
          manage_options', 'zero', array($this, 'menu_html'));
6 }
```

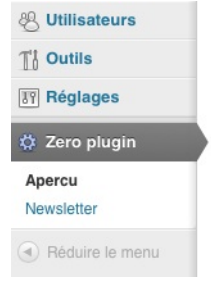


FIGURE 15.4 – Le menu mis à jour

Avant de continuer, créons la fonction d'affichage du sous-menu « Newsletter » qui se contentera pour l'instant d'afficher le titre de la page (voir la figure 15.5).

```
1 <?php
2 class Zero_Newsletter
3 {
4     //...
5     public function menu_html()
6     {
7         echo '<h1>'.get_admin_page_title().'</h1>';
8     }
9 }
```

FIGURE 15.5 – La page newsletter

Créer des options

Nos pages sont maintenant créées, il nous faut donc écrire le formulaire permettant de renseigner les différentes options de notre choix via la méthode menu_html(). Dans un premier temps, nous allons travailler avec un seul champ pour comprendre la démarche à effectuer, puis nous en ajouterons de nouveaux.

Le fonctionnement des options

Pour conserver un maximum de souplesse, WordPress inscrit un grand nombre de valeurs de configuration dans la base de données, dont un certain nombre sont éditables au travers du menu « Réglages » de l'administration. Ce fonctionnement permet d'avoir des attributs facilement modifiables par l'administrateur pour ne pas avoir à modifier le code PHP lorsqu'il désire modifier des paramètres comme le titre du site, le format des dates, la taille des médias...

Ces différents paramètres sont appelés des options et sont stockés dans la base de données dans la table wp_options, l'identifiant de l'option étant rangé dans la colonne option_name et sa valeur dans option_value. Ainsi, rien ne nous empêche de définir nos propres valeurs de configuration, notamment lorsque l'on veut pouvoir paramétrer un plugin. Il suffit pour cela d'ajouter une ligne à la table avec l'identifiant et la valeur désirés.

Pour récupérer la valeur d'une option, il faut utiliser la fonction get_option() en lui indiquant l'identifiant de l'option à récupérer. Si l'option n'existe pas, la fonction retourne false ou bien la valeur du second paramètre s'il a été fourni.

L'ajout d'une option dans la base de données se fait avec la méthode add_option(), à laquelle il faut envoyer l'identifiant ainsi que la valeur de l'option. Toutefois, cette fonction ne permet pas de mettre à jour une option déjà existante, il faut alors utiliser update_option(), dont les paramètres sont identiques. Notez qu'en utilisant update_option(), l'option sera créée si elle n'existe pas encore. Elle est donc utilisée dans la majorité des cas car elle convient aux deux usages.

Enfin, pour supprimer une option, on utilise la fonction delete_option() qui ne prend en paramètre que le nom de l'option. Elle renvoie la valeur true si l'option a été correctement supprimée, ou bien false si l'opération à échoué ou que l'option n'existait pas.

Le formulaire

Le code HTML

Lorsque l'on crée un formulaire destiné à enregistrer des données dans la table des options, celui-ci doit appeler le fichier wp-admin/options.php lors de la soumission des données. Commencez donc à compléter la méthode Zero_Newsletter : :menu_html() comme ceci :

```
1  <?php
2  public function menu_html()
3  {
4      echo '<h1>'.get_admin_page_title().'</h1>';
5      ?>
6      <form method="post" action="options.php">
7
8      </form>
9      <?php
10 }
```

Rajoutons tout de suite le champ de texte permettant l'enregistrement de l'adresse email de l'expéditeur de la newsletter, ainsi que le bouton de validation du formulaire grâce à la fonction submit_button(). Cette dernière permet de générer un bouton de validation avec les classes CSS nécessaires à un affichage homogène dans l'interface d'administration. Le libellé du bouton peut être modifié en passant une nouvelle valeur en paramètre.

```
1  <label>Expéditeur de la newsletter</label>
2  <input type="text" name="zero_newsletter_sender" value="<?php
       echo get_option('zero_newsletter_sender')?>"/>
3  <?php submit_button(); ?>
```

Le formulaire doit maintenant s'afficher correctement, mais si vous l'envoyez, vous verrez qu'aucune valeur n'est sauvegardée dans la base de données (le champ apparaît vide lorsque vous revenez sur la page). Ceci est du au fait que nous n'avons pas encore autorisé WordPress à enregistrer la valeur de l'option zero_newsleter_sender.

Enregistrer les options modifiables

La première chose à rajouter à l'intérieur du formulaire est un appel à la fonction settings_fields(), qui affiche un certain nombre de champs cachés permettant notamment à l'application de savoir à quel groupe d'option les champs que vous allez rajouter appartiennent.

Qu'est-ce qu'un groupe d'options ?

Un groupe d'options correspond à l'ensemble des options modifiables sur une page

donnée. WordPress a besoin de connaître le groupe auquel appartiennent les options afin de décider si vous avez le droit de modifier ces valeurs. Pour déterminer si les options peuvent être mises à jour, il sera nécessaire de déclarer l'ensemble des options modifiables dans un groupe d'options donné avec la fonction register_setting(). Nous appellerons notre groupe d'options zero_newsletter_settings, c'est donc l'argument à utiliser pour l'appel à settings_fields().

```
<form method="post" action="options.php">
<?php settings_fields('zero_newsletter_settings') ?>
```

Il nous faut ensuite enregistrer le champ zero_newsletter_sender dans le groupe d'options. Ceci doit impérativement se faire lorsque le système d'administration est initialisé, c'est-à-dire au déclenchement de l'événement admin_init. Ajoutons donc une fonction register_settings dans la classe Zero_Newsletter, ainsi que l'enregistrement de l'action dans son constructeur.

```
<?php
add_action('admin_init', array($this, 'register_settings'));
```

```
<?php
public function register_settings()
{
    register_setting('zero_newsletter_settings', '
        zero_newsletter_sender');
}
```

Le formulaire est en place, l'option que nous souhaitons enregistrer est déclarée comme autorisée, nous pouvons donc vérifier que l'enregistrement fonctionne. Pour cela, entrez une adresse et validez le formulaire. La page doit se réafficher et le champ être prérempli avec la valeur envoyée (voir la figure 15.6).

FIGURE 15.6 – Sauvegarde de la valeur du champ

Génération automatique des champs

Il reste deux options à rajouter dans notre formulaire : l'objet du mail et son contenu. Pour ces deux nouveaux champs, nous allons utiliser un autre moyen de création du code HTML, en nous reposant un peu plus sur l'API de WordPress. L'idée consiste à déclarer des champs avec leurs propriétés, puis à demander l'affichage de tous les

champs associés à un formulaire donné. Chaque champ doit être ajouté à une section, c'est à dire un groupe d'option, elle-même étant appelée depuis le formulaire final.

L'intérêt de ce fonctionnement est de pouvoir rajouter de nouveaux champs dans le formulaire sans pour autant devoir modifier la méthode de rendu de celui-ci, simplement en enregistrant une nouvelle section ou un nouveau champ. Il est donc tout à fait possible, grâce à ce mécanisme, de rajouter de nouvelles options dans un formulaire natif de WordPress ou créé par un autre plugin.

La création d'une section se fait par la fonction add_settings_section(), que l'on peut appeler directement dans la méthode Zero_Newsletter : :register_settings().

```
1 <?php
2 add_settings_section('zero_newsletter_section', 'Paramètres d\'
      envoi', array($this, 'section_html'), '
      zero_newsletter_settings');
```

Les paramètres de cette fonction sont l'identifiant de la section, son titre, une fonction appelée au début du rendu de la section et enfin la page sur laquelle devra s'afficher la section.

Nous allons créer immédiatement la méthode section_html() afin d'afficher une description de la section au début de celle-ci.

```
1 <?php
2 public function section_html()
3 {
4     echo 'Renseignez les paramètres d\'envoi de la newsletter.'
          ;
5 }
```

Même si la section ne contient pour l'instant aucun champ, nous pouvons en activer son affichage. Pour cela, il suffit d'appeler la fonction do_settings_sections() avec comme argument l'identifiant de la page pour laquelle la section a été créée. Si plusieurs sections on été déclarées pour une page donnée, elle seront toutes affichées consécutivement (voir la figure 15.7).

FIGURE 15.7 – Affichage de la section avec sa description

Modifiez donc le formulaire en supprimant le champ « Expéditeur » pour le remplacer par l'appel de cette nouvelle fonction.

```
1 <form method="post" action="options.php">
2 <?php settings_fields('zero_newsletter_settings') ?>
```

```
<?php do_settings_sections('zero_newsletter_settings') ?>
<?php submit_button(); ?>
</form>
```

Pour déclarer le champ « Expéditeur » ainsi que les deux champs supplémentaires, il faut passer par la fonction add_settings_field() qui se charge de l'ajout d'un champ à une section donnée.

```
<?php
add_settings_field('zero_newsletter_sender', 'Expéditeur',
    array($this, 'sender_html'), 'zero_newsletter_settings', '
    zero_newsletter_section');
```

Les paramètres de cette fonction sont l'identifiant du champ, le libellé à afficher, la fonction de rendu du champ, pour terminer par la page concernée et la section à laquelle il appartient. La méthode de rendu est très simple et ne doit se charger d'afficher que le champ en lui même comme nous l'avions fait précédemment (voir la figure 15.8).

```
<?php
public function sender_html()
{?>
    <input type="text" name="zero_newsletter_sender" value="<?
        php echo get_option('zero_newsletter_sender')?>"/>
    <?php
}
```

FIGURE 15.8 – Retour du champ expéditeur dans une section

Le formulaire est maintenant revenu à l'état précédent, il ne reste plus qu'à déclarer les deux nouveaux champs en passant par add_settings_field() et register_setting() et le formulaire sera mis à jour automatiquement, sans modifier la méthode menu_html(). Voici donc un récapitulatif des méthodes mises à jour avec les deux nouveaux champs « Objet » et « Contenu ».

```
<?php
public function register_settings()
{
    register_setting('zero_newsletter_settings', '
        zero_newsletter_sender');
    register_setting('zero_newsletter_settings', '
        zero_newsletter_object');
```

```
6       register_setting('zero_newsletter_settings', '
            zero_newsletter_content');
7
8       add_settings_section('zero_newsletter_section', 'Newsletter
             parameters', array($this, 'section_html'), '
            zero_newsletter_settings');
9       add_settings_field('zero_newsletter_sender', 'Expéditeur',
            array($this, 'sender_html'), 'zero_newsletter_settings',
             'zero_newsletter_section');
10      add_settings_field('zero_newsletter_object', 'Objet', array
            ($this, 'object_html'), 'zero_newsletter_settings', '
            zero_newsletter_section');
11      add_settings_field('zero_newsletter_content', 'Contenu',
            array($this, 'content_html'), 'zero_newsletter_settings'
            , 'zero_newsletter_section');
12  }
13
14  public function object_html()
15  {?>
16      <input type="text" name="zero_newsletter_object" value="<?
            php echo get_option('zero_newsletter_object')?>"/>
17      <?php
18  }
19
20  public function content_html()
21  {?>
22      <textarea name="zero_newsletter_content"><?php echo
            get_option('zero_newsletter_content')?></textarea>
23      <?php
24  }
```

Vous pouvez maintenant rafraîchir la page pour renseigner les valeurs de ces nouvelles options (voir la figure 15.9).

FIGURE 15.9 – Le formulaire final

Traiter des actions

Lorsque l'ensemble des options sont configurées, il ne manque plus qu'un bouton afin d'envoyer la newsletter aux adresses inscrites. Ce bouton doit donc déclencher l'appel d'une fonction spécifique qui envoie l'email à tous les destinataires souhaités.

Nous avons deux solutions pour arriver à nos fins. Nous pouvons utiliser un lien vers un fichier spécifique de notre plugin qui traitera directement l'envoi des emails avant de nous rediriger vers l'interface d'administration de notre plugin. C'est une solution qui paraît assez simple mais qui implique de charger les fonctionnalités de WordPress manuellement dans le fichier cible, c'est pourquoi elle n'a pas ma préférence. Au lieu de cela, nous allons simplement rafraîchir la page en cours et brancher une nouvelle action se chargeant de vérifier si l'envoi a été demandé et de l'exécuter le cas échéant.

Nous allons pour cela insérer un second formulaire dans la page, à la fin de la méthode menu_html().

```
<form method="post" action="">
    <input type="hidden" name="send_newsletter" value="1"/>
    <?php submit_button('Envoyer la newsletter') ?>
</form>
```

Le formulaire contient simplement un champ caché qui sera envoyé en POST afin de demander l'envoi des emails. Au clic sur le bouton, la page actuelle sera rafraîchie avec le paramètre send_newsletter présent dans la requête.

Nous devons donc intercepter la présence de ce paramètre au chargement de la page. Pour cela, il existe dans WordPress un événement déclenché au chargement de toutes les pages d'administration, de la forme load-[identifiant de la page]. L'identifiant de la page est récupérable lors de la création du menu qui permet d'accéder à la page : c'est le retour de la fonction add_submenu_page() (cela fonctionne aussi avec add_menu_page()).

Nous modifions donc la méthode Zero_Newsletter : :add_admin_menu() :

```
<?php
public function add_admin_menu()
{
    $hook = add_submenu_page('zero', 'Newsletter', 'Newsletter'
        , 'manage_options', 'zero_newsletter', array($this, '
        menu_html'));
    add_action('load-'.$hook, array($this, 'process_action'));
}
```

Dans cette méthode process_action(), nous vérifions la présence du paramètre send_newsletter avant d'appeler la méthode d'envoi.

```
<?php
public function process_action()
{
    if (isset($_POST['send_newsletter'])) {
        $this->send_newsletter();
    }
```

```
}
```

Il ne reste plus maintenant qu'à écrire la méthode send_newsletter() pour envoyer les emails à nos visiteurs inscrits. Nous allons donc devoir récupérer les paramètres de configuration choisis ainsi que la liste des emails, puis appeler la fonction wp_mail() qui permet de construire un email.

```php
<?php
public function send_newsletter()
{
    global $wpdb;
    $recipients = $wpdb->get_results("SELECT email FROM {$wpdb
        ->prefix}zero_newsletter_email");
    $object = get_option('zero_newsletter_object', 'Newsletter'
        );
    $content = get_option('zero_newsletter_content', 'Mon
        contenu');
    $sender = get_option('zero_newsletter_sender', 'no-
        reply@example.com');
    $header = array('From: '.$sender);

    foreach ($recipients as $_recipient) {
        $result = wp_mail($_recipient->email, $object, $content
            , $header);
    }
}
```

Et voilà ! Le site est prêt à envoyer des newsletters à ses inscrits en un seul clic sur le bouton d'envoi !

En résumé

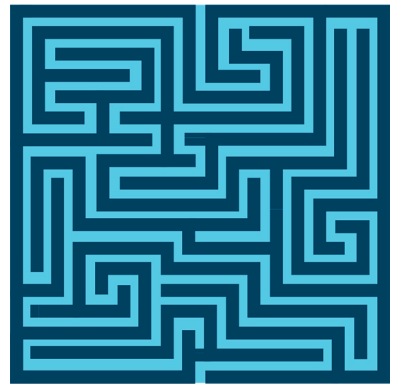
- De nouvelles pages peuvent être ajoutées dans l'interface d'administration, accessibles avec des menus spécifiques.
- Les options permettent de sauvegarder les valeurs de vos paramètres de configuration.
- Vous pouvez créer des options spécifiques à votre plugin et les éditer dans l'administration.
- L'enregistrement des sections et des champs pour les formulaires permet plus de souplesse que la construction manuelle.

Chapitre 16

Les shortcodes

Difficulté :

Un shortcode est un morceau de code PHP que l'on peut placer directement dans un article ou une page de votre blog. Cette balise code est automatiquement interprétée par WordPress et permet d'ajouter des fonctionnalités (une galerie image, une insertion vidéo, etc.) sans programmation de votre part ! Grâce aux shortcodes, votre blog devient dynamique puisque son contenu peut être changé à tout moment, en temps réel et sans compétence informatique.



Utiliser un shortcode

Format

Pour utiliser un shortcode, il suffit de mettre son nom dans le contenu d'un article, encadré de crochets.

```
[nom_du_shortcode]
```

Par exemple, WordPress connaît un shortcode gallery qui renvoie l'ensemble des médias de type image associés à un article, dont l'affichage sur le site changera en fonction du nombre d'images dans la bibliothèque sans qu'il y ait besoin de modifier l'article en question.

Choisissez donc un article ou une page possédant des images, et introduisez le shortcode au sein de son contenu (voir la figure 16.1).

FIGURE 16.1 – L'ajout du shortcode dans le contenu d'un article

Lorsqu'un visiteur parcourra l'article choisi, il verra, comme sur la figure 16.2, l'ensemble des images associées à celui-ci.

Paramètres

Pour ajouter de la souplesse aux shortcodes, il est tout à fait possible de leur attribuer des paramètres utilisés par la fonction PHP lors du rendu, en rajoutant des paires clés/valeur à la suite du nom du shortcode.

```
[nom_du_shortcode attribut1="valeur1" attribut2="valeur2"]
```

On peut aussi ajouter une grosse portion de texte dans un shortcode en l'écrivant sous une forme différente, avec une balise ouvrante et une balise fermante. Ce texte sera alors considéré comme un nouveau paramètre, généralement utilisé comme contenu du shortcode.

```
[nom_du_shortcode]le contenu vient ici[/nom_du_shortcode]
```

De plus, le contenu peut être du texte brut ou bien du HTML ; il n'y a pas de restriction. Ce sera ensuite le rôle du shortcode de traiter le texte fourni en cas de besoin.

FIGURE 16.2 – Affichage du shortcode « Gallery »

Créer un shortcode

Que diriez-vous de créer un shortcode simple, dont le rôle sera par exemple de lister les derniers articles publiés sur votre blog ? Placé par exemple dans une page statique, il fournira sous forme de liste les titres des derniers articles avec un lien vers chacun d'eux.

Pour traiter le shortcode, il faudra utiliser une nouvelle classe de notre plugin qui sera dédiée au rendu de la liste des derniers articles. Ajoutez donc un nouveau fichier recent.php dans lequel vous définirez une classe Zero_Recent, elle-même étant instanciée dans le constructeur de la classe Zero_Plugin, comme cela a été fait pour les classe précédentes.

```
<?php
class Zero_Plugin
{
    public function __construct()
    {
        //...
        include_once plugin_dir_path( __FILE__ ).'recent.php';
        new Zero_Recent();
```

Il faut ensuite déclarer le shortcode afin que WordPress sache le reconnaître dans le contenu d'une publication. On utilise pour cela la méthode add_shortcode(), qui attend en paramètres le nom du shortcode ainsi que la fonction à appeler lors de son rendu. Placez donc cet appel dans le constructeur de la classe Zero_Recent.

```
<?php
class Zero_Recent
{
    public function __construct()
    {
        add_shortcode('zero_recent_articles', array($this, '
            recent_html'));
    }
}
```

WordPress a dorénavant connaissance du nouvel élément, il ne lui manque que la définition de la fonction à appeler pour l'afficher. Cette dernière recevra deux arguments lors de son appel : le premier est un tableau des paramètres permettant de configurer le shortcode, le second est la valeur du contenu si le shortcode a été déclaré avec les deux balises ouvrante et fermante.

```
<?php
public function recent_html($atts, $content)
{
}
```

La première chose à faire est d'initialiser des valeurs par défaut pour les paramètres du shortcode, si jamais ceux-ci n'ont pas été fournis. En effet, un shortcode doit au maximum pouvoir être utilisé sans paramètres, il faut donc prévoir un comportement

fonctionnel dans ce cas. Pour fusionner simplement le tableau de paramètres fournis et les valeurs par défaut, WordPress propose la fonction shortcode_atts().

```
<?php
$atts = shortcode_atts(array('numberposts' => 5), $atts);
```

Seules les clés du tableau des valeurs par défaut seront présentes en sortie de cette fonction, tous les autres paramètres fournis au shortcode seront ignorés et supprimés !

Une fois les arguments filtrés, vous pouvez récupérer la liste des derniers articles avec la fonction get_posts(), pour laquelle les paramètres doivent être envoyés dans un tableau comme premier argument. La fonction renverra alors une liste d'articles, qui sont par défaut triés par date décroissante, ce qui correspond à nos besoins.

```
<?php
$posts = get_posts($atts);
```

Vous pouvez bien entendu rajouter de nouveaux paramètres compatibles avec la fonction dans le tableau $atts, il vous suffit pour cela de regarder la définition de la fonction pour connaître la liste des clés autorisées.

Il ne reste plus qu'à parcourir la liste des articles et à les insérer dans une liste au format HTML, avec un lien permettant d'afficher chacun d'entre eux. Si la variable $content a été définie par l'utilisateur du shortcode, vous pouvez l'insérer par exemple au dessus de la liste, ce qui permet de rajouter un texte de présentation.

```
<?php
$html = array();
$html[] = $content;
$html[] = '<ul>';
foreach ($posts as $post) {
    $html[] = '<li><a href="'.get_permalink($post).'">'.$post->
        post_title.'</a></li>';
}
$html[] = '</ul>';

echo implode('', $html);
```

Le shortcode est terminé et prêt à être utilisé ! Rendez-vous pour cela dans l'éditeur de contenu d'une de vos publications, et placez le code suivant à l'intérieur :

```
[zero_recent_articles numberposts=3]Voici les derniers articles
    publiés sur le blog : [/zero_recent_articles]
```

À l'affichage de la page, comme sur la figure 16.3, vous obtenez la liste à jour des trois derniers articles publiés.

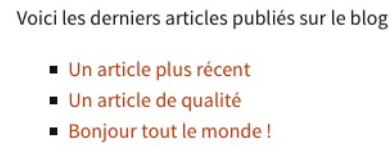


FIGURE 16.3 – Affichage des trois derniers articles

En résumé

— Un shortcode est un morceau de code PHP que l'on insère directement au sein d'une publication. Il permet l'affichage de contenu sans savoir même coder.
— Les shortcodes peuvent récupérer des paramètres fournis par le contributeur d'un article pour modifier leur affichage.
— Il existe plusieurs formats pour les shortcodes, avec une seule ou deux balises pour rajouter du contenu.

Quatrième partie

Exploiter votre site

Chapitre 17

Mettre en production

Difficulté :

Voilà, vous avez maintenant une bonne maîtrise de WordPress. Vous avez trouvé ou programmé les thèmes et plugins dont vous aviez besoin pour que votre site puisse être mis en ligne ? Il est temps de nous attaquer aux derniers réglages pour la mise en production de WordPress !

Suivant la solution que vous aurez choisie pour votre hébergement, la procédure est quelque peu différente, nous allons donc faire un tour d'horizon des solutions qui s'offrent à vous.

Sur un hébergement mutualisé

L'hébergement mutualisé est la solution la moins onéreuse et permet d'avoir un site en place rapidement sans avoir à passer beaucoup de temps en configuration. C'est celle qui est conseillée si vous n'attendez pas (pour l'instant !) un fort trafic sur votre site. De plus, les offres d'hébergement s'accompagnent souvent d'un nom de domaine pour votre site Internet (c'est le cas chez OVH notamment), vous n'avez donc pas à payer autre chose que le serveur.

Lorsque vous prenez une offre chez un hébergeur, vous recevrez des identifiants permettant de vous connecter à la base de données ainsi qu'au serveur à l'aide d'un logiciel FTP pour envoyer vos fichiers sur le site. Vous pouvez pour cela utiliser le logiciel FileZilla qui est plutôt simple d'utilisation.

Après avoir lancé FileZilla, vous devez indiquer vos identifiants ainsi que l'adresse du serveur FTP dans la barre de connexion en haut. Toutes les informations doivent avoir été fournies lors de l'inscription par l'hébergeur (voir la figure 17.1).

FIGURE 17.1 – Connexion au serveur avec vos identifiants

Une fois connecté, vous devrez, comme sur la figure 17.2, envoyer la totalité des fichiers de votre site dans le répertoire racine du serveur, le plus souvent intitulé « www ».

Dès que la copie est terminée, vous pouvez vous rendre à l'adresse de votre site Internet qui est maintenant en ligne. Il faut alors lancer la procédure d'installation de WordPress comme nous l'avons fait au début de ce cours.

Sur un serveur dédié

Si vous choisissez comme solution d'hébergement un serveur dédié, il y a un peu plus de travail que précédemment car il faut aussi s'occuper de l'installation et de la configuration du serveur. Une fois votre serveur en place, vous aurez besoin de vous connecter sur celui-ci afin de débuter l'installation des différents fichiers. Les identifiants de connexion au serveur vous ont normalement été fournis par l'hébergeur et permettent de s'y connecter avec le protocole SSH. Pour cela, vous pouvez utiliser le logiciel Putty sur Windows ou bien un terminal sur Linux et Mac.

Installation et copie des fichiers

La configuration du serveur est sensiblement identique à l'installation d'un serveur local sur Linux. Les logiciels à installer sont les mêmes (mis à part phpMyAdmin qui est inutile ici) et suivront la même procédure.

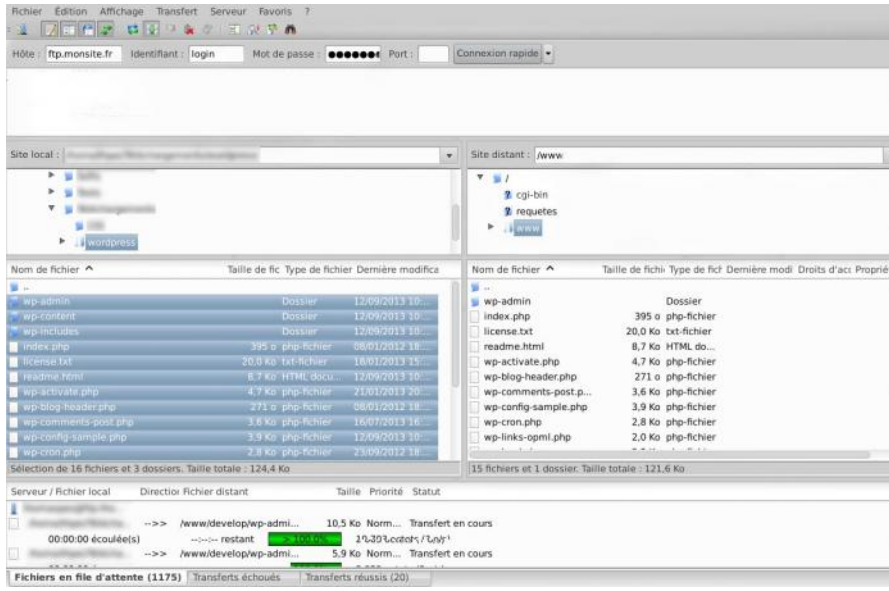


FIGURE 17.2 – Transfert des fichiers de WordPress dans le répertoire

```
sudo apt-get install apache2 php5 mysql-server libapache2-mod-
    php5 php5-mysql
```

Vous pouvez maintenant copier les fichiers de WordPress sur votre serveur.

```
cd /var/www
curl -O http://fr.wordpress.org/wordpress-3.6.1-fr\_FR.tar.gz
tar -xzf wordpress-3.6.1-fr\_FR.tar.gz
```

Le nom du fichier peut varier suivant la version de WordPress. Pensez à le vérifier sur le site officiel afin de récupérer la dernière version.

Vous obtenez ainsi un dossier intitulé « wordpress » contenant les sources de la dernière version du CMS prêt à être lu par Apache.

Définition du Virtual Host

Il nous reste à configurer un Virtual Host pour que votre serveur sache que l'adresse url de votre site, par exemple http ://monsite.com correspond au répertoire de WordPress. Sans cela, vous ne pourrez y accéder que par une URL de la forme suivante, http ://monsite.com/wordpress.

Voici un exemple de Virtual Host basique pour WordPress, qui reprend entre autre le contenu du fichier .htaccess fourni, vous devrez bien sûr adapter les valeurs à votre installation. Vous devez placer cette configuration dans le répertoire /etc/apache2/sites-available, dans un fichier intitulé wordpress.conf. Le nom peut varier, mais l'extension doit être .conf.

```
<VirtualHost *:80>
    ServerName monsite.com

    DocumentRoot /var/www/wordpress
    <Directory /var/www/wordpress>
        Deny from none
        Allow from all
        Order allow,deny
        AllowOverride None

        RewriteEngine On
        RewriteCond %{REQUEST_FILENAME} !-f
        RewriteCond %{REQUEST_FILENAME} !-d
        RewriteRule .* /index.php [L]
    </Directory>
</VirtualHost>
```

Il faut ensuite activer ce VirtualHost avec la commande a2ensite, et désactiver la configuration par défaut avec a2dissite. Enfin, demandez à Apache de recharger sa configuration pour prendre en compte les dernières modifications.

```
a2ensite wordpress
a2dissite default
service apache2 reload
```

En résumé

— Un site peut être hébergé soit sur un serveur mutualisé partagé par plusieurs utilisateurs, soit sur un serveur dédié.
— Le logiciel FileZilla permet d'envoyer vos fichiers sur un serveur mutualisé avec un accès FTP.
— Sur un serveur dédié, la configuration du serveur Apache doit se faire manuellement.

Chapitre 18

Améliorer le référencement

Difficulté :

Une fois votre site en ligne, la prochaine étape importante est d'avoir des visiteurs. L'objectif de ce chapitre est de comprendre comment optimiser votre site et vos contenus, afin d'apparaître dans les meilleurs résultats des moteurs de recherche (par exemple, Google). Pour cela, il faut configurer correctement WordPress et mettre en avant les sujets en rapport avec votre site.

Des URLs propres

Le contenu des URLs qui pointent vers votre site, et plus précisément vos publications, est essentiel pour avoir un bon référencement. Par défaut, les URLs des publications sont de la forme suivante http ://monsite.com ?p=3, ce qui ne donne aucune information quant au contenu de la page. Si votre page s'appelle « Bienvenue sur mon blog », il est nettement plus intéressant d'avoir l'URL de la forme suivante, http ://monsite.com/bienvenue-sur-mon-blog car les moteurs de recherche récupèrent des informations en plus, rien qu'en lisant l'adresse de la page. Pour faire cela, il suffit d'aller sur le panneau l'administration du site, dans « Réglages > Permaliens », puis de sélectionner le réglage « Nom de l'article » sur la page des options (voir la figure 18.1).

FIGURE 18.1 – Réglage des paramètres en fonction des URLs souhaitées

Une fois ce réglage effectué, vous aurez la possibilité, sur chaque publication, de choisir le lien qui mènera à la page correspondante.

Veillez à ne pas modifier ce lien une fois que l'article a été publié, car si un visiteur (ou un moteur de recherche) récupère le lien original avant qu'il soit modifié, alors une erreur 404 apparaîtra lors de l'accès à la page.

Si malgré tout vous souhaitez modifier une URL après coup, il est conseillé de créer une redirection de l'ancien lien vers le nouveau afin d'indiquer que le contenu a été déplacé pour éviter de perdre vos visiteurs. Certains plugins permettent de gérer cela automatiquement, comme celui indiqué dans le code web suivant :

▷ Redirection
Code web : 387572 .

Un contenu de qualité

L'un des critères essentiels pour un bon référencement est la mise à jour régulière du contenu de votre site. Les moteurs de recherche sont en effet conçus pour détecter les

modifications apportées à vos pages, aux nouvelles publications et plus généralement, au changement. Un site à l'abandon descendra rapidement dans les résultats de recherche, car on considère alors que les informations qu'il fournit sont obsolètes.

On ne parle pas ici de publier des articles tous les jours, il y a très peu de blogs qui font cela. L'important est d'avoir un rythme régulier qui convient aux sujets auxquels vous choisissez de vous consacrer, c'est ce qui détermine la fréquence de mise à jour. Si vous publiez sur un projet personnel, vous aurez envie de tenir vos visiteurs au courant de chaque avancée. Si c'est une entreprise que vous représentez, alors le site peut être mis à jour lors d'événements particuliers ou de nouvelles activités. À chaque site convient son rythme. Ce qu'il faut retenir, c'est qu'il ne faut pas laisser vos pages à l'abandon pour conserver un classement intéressant sur les moteurs de recherche.

Ceci étant dit, il ne faut pas faire valoir la quantité au détriment de la qualité. Publier des articles au contenu riche et pertinent est plus valorisant pour votre site que de publier régulièrement du contenu avec peu d'intérêt. Pour mettre en avant vos pages et vos articles, n'oubliez pas de placer les mots importants dans le contenu de ceux-ci pour faciliter la recherche de vos potentiels visiteurs. Pensez aussi à associer des catégories et des mots-clés pertinents à vos articles.

Faciliter l'indexation

Le sitemap

Le sitemap est un fichier contenant un plan de votre site, c'est-à-dire la liste de toutes les pages qui existent avec leurs URLs. Cela permet à un moteur de recherche de trouver plus facilement vos pages pour les indexer. Le sitemap doit dans l'idéal être mis à jour à chaque fois qu'une nouvelle page est créée, afin d'être parcourue rapidement par les robots et donc de contribuer à votre référencement. Là encore, il existe des plugins qui permettent de générer automatiquement le sitemap de votre blog, notamment celui proposé dans le code web suivant :

▷ Sitemap generator
Code web : 175538

Les robots

On désigne par le terme de *robots* les programmes qui parcourent les pages Internet à la recherche de contenu à indexer dans les moteurs de recherche. Il est possible, sur votre site, de demander aux robots de ne pas indexer une partie des pages dans les moteurs. Cela peut être utile par exemple pour la page de connexion à l'administration de WordPress, qui n'a pas de raison de se retrouver dans les résultats de recherche d'un internaute. Pour cela, on utilise des directives spécifiques destinées à ces robots afin de leur indiquer les pages pertinentes pour l'indexation.

Malgré ce système, les robots peuvent très bien choisir d'indexer vos pages, cela ne reste qu'une indication. Ils sont cependant généralement respectueux de vos demandes.

WordPress offre la possibilité de cacher l'ensemble de votre site aux robots, il faut donc bien vérifier que cette fonctionnalité est désactivée lors de la mise en ligne de votre site. Dans le cas contraire, vous pourriez ne pas apparaître du tout dans les moteurs de recherche ! Pour vérifier cela, allez dans « Réglages > Lecture », l'option « Demander aux moteurs de recherche de ne pas indexer ce site » doit être décochée.

En résumé

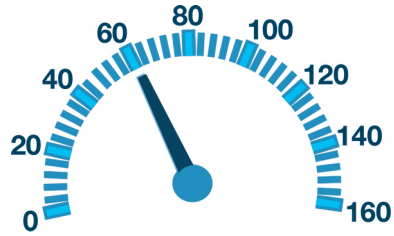
— Les URLs réécrites permettent de donner du sens à vos liens pour les moteurs de recherche.
— Plus les contenus seront mis à jour, plus ceux-ci auront de la valeur pour les moteurs de recherche.
— Les mots-clés rajoutent des informations pour mettre en avant les articles.
— Faciliter l'indexation du site (sitemap, directives adressées aux robots) permet d'améliorer le référencement.

Chapitre 19

Optimiser les performances

Difficulté :

L'un des critères les plus importants dans l'expérience de navigation d'un visiteur sur votre site est sans aucun doute le temps que celui-ci prend à s'afficher. Il est donc important de réduire ce temps au minimum. Pour cela, il faut chasser toutes les lenteurs qui peuvent amoindrir la qualité de la visite, au niveau du code PHP ou bien de l'affichage HTML sur le navigateur. Accorder de l'attention à ces deux points permet d'améliorer significativement les performances ressenties lorsque l'on parcourt votre site. Nous allons donc étudier quelques astuces qui lui donneront un coup de jeune.



Utiliser le cache WordPress

Sur les applications web, l'une des actions les plus coûteuses en termes de temps de calcul de la page est l'accès aux informations de la base de données. En effet, une vaste proportion du contenu d'une page donnée provient de la configuration et des contributions des administrateurs du site, tout ceci se trouvant dans différentes tables de la base de données. L'application doit donc, pour chaque page affichée, établir une connexion et effectuer des requêtes pour obtenir les données nécessaires à son fonctionnement.

Pour accélérer le temps de rendu de vos pages et donc diminuer la charge de votre serveur, il est important de faire attention aux ressources utilisées, notamment lorsque vous développez votre propre plugin, qui devra vraisemblablement écrire et lire dans la base de données. Si le premier conseil à connaître est de ne pas récupérer des données depuis la base plusieurs fois dans une même page, cela n'est pas toujours possible de façon simple et les développeurs de WordPress s'en sont bien rendu compte.

La classe WP_Object_Cache fournit une interface au travers de laquelle il est possible de stocker des informations (sous forme de paires clé/valeur) afin de les récupérer plus tard au cours du chargement de la page. Cette solution offre l'avantage d'éviter d'effectuer à plusieurs reprises des calculs ou des requêtes coûteuses en enregistrant le résultat de l'opération pour un usage ultérieur.

Pour utiliser l'objet gérant le cache, il faut passer par des fonctions d'accès définies dans le fichier wp_includes/cache.php. Ainsi, la fonction wp_cache_add() permet d'ajouter une entrée de cache, en fournissant en paramètres la clé à utiliser et la valeur à stoker. Pour récupérer la valeur en cache, on appellera wp_cache_get() avec la clé en paramètre, qui renverra la valeur stockée ou bien false si aucune entrée n'est trouvée.

En utilisant ces fonctions, l'idée est donc de faire un appel à wp_cache_get() lorsque l'on a besoin d'une valeur qui est potentiellement présente en cache. Si cette valeur n'est pas trouvée dans le cache, on la calcule comme on l'aurait fait sans le cache, puis on la stocke pour un accès ultérieur, éventuellement dans une autre partie du code.

```
1  <?php
2  function getMaValeur()
3  {
4      if (wp_cache_get('ma_valeur')) {
5          return wp_cache_get('ma_valeur');
6      } else {
7          // on calcule notre variable normalement, stockée dans
               $value
8          //...
9          wp_cache_add('ma_valeur', $value);
10         return $value;
11     }
12 }
```

En utilisant le cache de cette façon, vous êtes certain de ne traiter les données lourdes que lorsque vous en avez besoin, tout en garantissant de ne faire ce traitement qu'une seule fois, c'est-à-dire lors du premier accès.

Optimiser l'affichage des pages

Cacher les ressources

À chaque chargement d'une page, le navigateur doit récupérer à la fois le code HTML, les fichiers CSS et JavaScript ainsi que les images affichées. Même si ces différentes ressources (hormis le code HTML) ne changent que très rarement, le navigateur va donc par défaut les récupérer en effectuant pour cela une requête auprès du serveur. Pour chaque fichier, une requête différente est émise et doit être traitée pour renvoyer la ressource, ce qui constitue un coût en temps de rendu de la page et en bande passante.

Afin de limiter le nombre de requêtes envoyées à votre serveur, il faut explicitement indiquer au navigateur des visiteurs quels sont les fichiers qui ne changeront que rarement lors de leurs visites. Dans ce cas, ces ressources seront cachées sur l'ordinateur du visiteur et réutilisées à chaque fois qu'une page de votre site sera affichée, tant que le délai minimum d'expiration ne sera pas écoulé. Les feuilles de style CSS peuvent par exemple être stockées plusieurs semaines si vous ne mettez pas à jour le design de façon très régulière. Il n'y a pas de raison de forcer vos visiteurs à la télécharger à chaque affichage.

Pour cacher des ressources, il faut effectuer des modifications dans la configuration du serveur, c'est-à-dire dans le VirtualHost (entre les balises <VirtualHost> du fichier) si vous en avez défini un, ou bien directement dans le fichier .htaccess (situé à la racine de WordPress) dans le cas contraire.

```
1 ExpiresActive On
2 ExpiresByType text/css "access plus 7 days"
3 ExpiresByType text/javascript "access plus 7 days"
4 ExpiresByType image/jpeg "access plus 7 days"
5 ExpiresByType image/x-icon "access plus 7 days"
```

Pour chaque type de fichier que l'on souhaite cacher, on ajoute une ligne indiquant la durée après laquelle le fichier doit être redemandé par le navigateur. Ici, j'ai choisi une durée de sept jours, c'est-à-dire qu'un visiteur n'aura pas besoin de télécharger ces ressources lors de toutes ses visites dans la semaine qui suit sa première connexion.

Fusionner les fichiers JS et CSS

Principe

Si cacher les ressources permet de limiter le nombre de requêtes sur votre serveur à partir de la deuxième visite d'un internaute, cela n'a aucun impact pour les nouveaux arrivants qui devront quoi qu'il arrive télécharger toutes vos images ainsi que vos fichiers CSS et JavaScript. Si vous possédez plusieurs feuilles de style ou fichiers JavaScript sur votre site, il peut devenir intéressant de les fusionner afin de n'en faire qu'un, qui sera alors téléchargé en une seule fois par vos visiteurs. La page sera ainsi téléchargée plus rapidement et donc affichée tout aussi vite.

Ici encore, des plugins existent pour vous faciliter la tâche et fusionner vos fichiers à la volée lors du chargement de la page. Vous n'avez pas à vous occuper de la génération du fichier final. Tout est fait automatiquement pour vous, comme c'est le cas avec WP Minify, proposé dans le code web suivant :

▷ WP Minify
Code web : 675432

Configurer WP Minify

Une fois le plugin WP Minify installé, un nouveau menu est disponible dans l'administration en cliquant sur « Réglages > WP Minify ». La configuration par défaut est normalement correcte : la compression des fichiers JavaScript et CSS est activée et le nombre de requêtes adressées à votre serveur est immédiatement réduit. Vous pouvez à tout instant désactiver la fusion des fichiers en modifiant la configuration dans le cas où des problèmes apparaîtraient. Le plugin prend aussi en charge la compression du code HTML en supprimant des caractères inutiles comme les espaces, mais l'impact sur les performances est moindre par rapport à la fusion des fichiers de ressources (voir la figure 19.1).

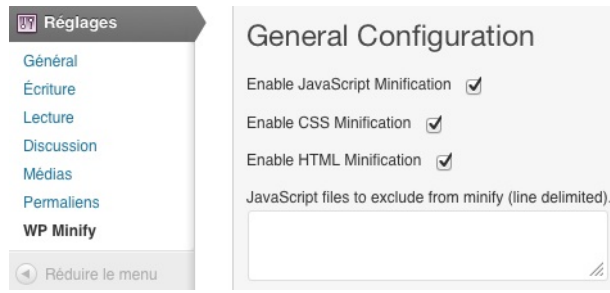


FIGURE 19.1 – Les paramètres du plugin WP Minify

En résumé

— Le cache WordPress permet de limiter les requêtes sur les base de données, en enregistrant les informations requises plusieurs fois par page.
— La configuration du serveur indique au visiteur les contenus qu'il peut cacher sur son ordinateur pour limiter les téléchargements.
— La compression des fichiers de ressources permet d'accélérer le chargement des pages en limitant les requêtes sur le serveur.

Index