CMS : Création d'un thème Wordpress

Création d'un thème Wordpress

Structure d'un thème Wordpress

Les thèmes sont stockés dans le répertoire wp-content/themes.
Pour être reconnu par Wordpress, un thème doit contenir au minimum 2 fichiers : index.php et style.css.
Un thème Wordpress est généralement composé des fichiers :

- Modèle principal : **index.php**
- Feuille de style : **style.css**
- Modèle de barre latérale : **sidebar.php**
- Modèle d'en-tête : **header.php**
- Modèle de pied de page : **footer.php**
- Modèle de page : **page.php**
- Modèle d'article : **single.php**
- Modèle d'archive : **archive.php**
- Page d'Index des archives : **archives.php**
- Modèle des commentaires : **comments.php**
- Modèle de fenêtre des commentaires : **comments-popup.php**
- Liens : **links.php**
- Modèle de page de résultats de recherche : **search.php**
- Formulaire de recherche : **searchform.php**
- Modèle 404 : **404.php**
- Page d'accueil personnalisée (facultatif) : **front-page.php**

Étape 1 : création du dossier du thème et des fichiers de base

Nous allons monter le thème **Mindgeek** sous **Wordpress**.
Après avoir installé Wordpress, créez un dossier **mindgeek** sous **wp-content/themes**

Création des fichiers de base

Un thème est reconnu par Wordpress dès lors qu'il contient au minimum ces 2 fichiers de base :

- index.php
- style.css

Allez dans Apparence => Thèmes et vous devriez voir apparaître le nouveau thème créé. Activez-le !

Nous allons tout d'abord créer 5 fichiers :

- index.php
- header.php
- sidebar.php
- footer.php
- style.css (la feuille de style)

Fonctions du Codex :

https://codex.wordpress.org/Function_Reference/language_attributes
https://developer.wordpress.org/reference/functions/bloginfo/
https://codex.wordpress.org/Function_Reference/get_header
https://codex.wordpress.org/Function_Reference/get_footer
https://codex.wordpress.org/Function_Reference/wp_footer
https://developer.wordpress.org/reference/functions/get_sidebar/

Création du fichier style.css

On commence par créer un fichier css très simple. On le complètera par la suite.

```
/*
Theme Name: Mindgeek
Author: 3W Academy
Author URI: https://3wa.fr/
Description: Ce thème est mis à disposition pour l'usage personnel des étudiants.
Version: 1.1
License: 3W Academy
License URI: https://3wa.fr/propriete-materiel-pedagogique
*/

body {
  font-family: Helvetica, Arial, sans-serif;
  font-size:1em;
  line-height:2em;
  margin:0;
}
.container {
  max-width:80%;
  margin-left:auto;
  margin-right:auto;
}
```

Création du fichier header.php

Ce fichier affiche l'entête de page, de la balise <doctype> jusqu'à la balise </header>.
La fonction php **language_attributes()** est une fonction du codex de Wordpress ; elle permet de récupérer les attributs lang et ltr de la balise <html>.
Elle récupère la langue de l'installation de Wordpress.
La fonction **php bloginfo()** est une fonction du codex de Wordpress et récupère les informations des réglages du site (onglet réglages). Avec le paramètres charset, elle récupère l'encodage des pages.

```
<?php
/**
 * The template for displaying the header
 * @package WordPress
 * @subpackage Mindgeek
 * @since Mindgeek 1.0
 */
?>
<!DOCTYPE html>
<html <?php language_attributes(); ?> class="no-js">
<head>
    <meta charset="<?php bloginfo( 'charset' ); ?>">
    <meta name="viewport" content="width=device-width">
    <title>Mon 1er thème Wordpress</title>
</head>
<body>
```

Création du fichier footer.php

Ce fichier affiche le pied de page, de la balise <footer> à la balise </html>, ainsi que la toolbar de Wordpress en mode front.

```
<?php
/**
```

```
 * The template for displaying the footer
 * @package WordPress
 * @subpackage Mindgeek
 * @since Mindgeek 1.0
 */
?>
<footer id="footer">
  <!-- Affichage de la licence -->
  <p id="licence" class="container">
      <small><a rel="license" href="https://3wa.fr/propriete-materiel-pedagogique/"><img
alt="Propriété de la 3W Academy" style="border-width:0" width="50"
src="https://3wa.fr/wp-content/themes/3wa2015/img/logos/big.png" /></a><br /><span>Cet
exercice</span> de <a href="https://3wa.fr">3W Academy</a> est mis à disposition <a
rel="license" href="https://3wa.fr/propriete-materiel-pedagogique/">pour l'usage
personnel des étudiants, Pas de Rediffusion - Attribution - Pas d'Utilisation
Commerciale - Pas de Modification - International</a>.<br />Les autorisations au-delà du
champ de cette licence peuvent être obtenues auprès de <a href="mailto:contact@3wa.fr"
rel="cc:morePermissions">contact@3wa.fr</a>. Les maquettes ont été réalisées par <a
href="http://www.justine-muller.fr">Justine Muller</a>.</small>
  </p>
</footer>
<!-- La fonction wp_footer doit être appelée ici pour afficher la toolbar de Wordpress
au-dessus -->
<?php wp_footer(); ?>
</body>
</html>
```

Création du fichier index.php

On commence par un code très simple pour afficher le traditionnel « Hello World » en titre 1.
Et on affiche bien entendu le header et le footer à l'aide de 2 fonctions du codex de Wordpress : **get_header** et **get_footer**.

```
<!-- Affichage de l'entête -->
<?php get_header(); ?>
<main>
<?php print ('<h1>Hello World</h1>'); ?>
</main>
<!-- Affichage du footer -->
<?php get_footer(); ?>
```

Allez dans l'administration de Wordpress, onglet Apparence => Themes. Le nouveau thème créé devrait apparaître dans la liste des thèmes.
Il nous reste à le prévisualiser et puis à l'activer.

Afficher le contenu

Modification du fichier header.php

Afficher le titre et la description du blog

Ajoutez les lignes suivantes <u>après</u> la balise <body> :

```
<header class="container">
    <!-- Affichage du titre du blog -->
    <h2><a href="<?php bloginfo('url'); ?>"><?php bloginfo('name');?></a></h2>
    <!-- Affichage de la description du blog -->
```

```
    <p><?php bloginfo('description'); ?></p>
    <?php endif; ?>
</header>
```

On retrouve la fonction bloginfo utilisée précédemment.

Modification du fichier index.php

Afficher les articles

Çà se complique ! On découvre d'autres fonctions du codex de Wordpress et aussi la boucle d'affichage du contenu.
Supprimez le code qui se trouve entre <main> et </main> et le remplacer par les lignes suivantes :

```
<main class="container">
<!-- Boucle sur les articles -->
<section class="content">
<!-- Début de la boucle. -->
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
<article class="post">
<!-- Affichage du titre en tant que lien vers le Permalien de l'article. -->
<h2><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a></h2>
<?php the_content(); ?>
<!-- Affiche la date de publication, l'auteur et les catégories de l'article. -->
<footer class="postmetadata">
  <p><small>Publié le <?php the_time('l d M Y'); ?>  par <?php the_author() ?> dans la
catégorie <?php the_category(', '); ?></small></p>
</footer>
</article> <!-- Fin de l'article -->
<!-- Fin de La Boucle -->
<?php endwhile; ?>
<?php else: ?>
<!-- Le premier "if" teste l'existence d'articles à afficher.
La partie "else" indique que faire si ce n'est pas le cas. -->
<p>Aucun contenu à afficher.</p>
<!-- Fin -->
<?php endif; ?>
</section>
</main>
```

Si tout va bien, l'article créé par défaut de de l'installation de WP « Bonjour tout le monde » s'affiche, ainsi que la date de publication, l'auteur du contenu et les catégories liées à l'article.Dans le navigateur, vous devriez obtenir ce rendu :

Test de Wordpress

Un site utilisant WordPress

Bonjour tout le monde !

Bienvenue dans WordPress.

Ceci est votre premier article.

Modifiez-le ou supprimez-le, puis lancez-vous !

Publié le lundi 15 Mai 2017 par admin dans la catégorie Non classé

Affichage des postmetadata

Après avoir inséré le contenu des articles, on va afficher les postmetadata. Ce sont des informations que l'on retrouve souvent sous le titre de l'article.

Affichage du lien avec le nombre de commentaires

Ajouter ces lignes avant la fin de la bouche (endwhile) :

```
<!-- Afficher le nombre de commentaires -->
<p><?php comments_popup_link('Pas de commentaires', '1 Commentaire', '% Commentaires',
'comments-link', 'Commentaires désactivés'); ?></p>
```

Ajout d'un lien pour éditer le contenu depuis le front-office

A jouter après l'affichage du nombre de commentaires

```
<!-- Permettre l'édition de l'article dans la page web -->
<?php edit_post_link('Éditer', '<p>', '</p>'); ?>
```

Votre fichier index.php devrait ressembler à ceci :

```
<!-- Affichage de l'entête -->
<?php get_header(); ?>
<main>
<!-- Boucle sur les articles -->
<section class="content">
<!-- Début de la boucle. -->
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
<article class="post">
<!--Affichage du titre en tant que lien vers le Permalien de l'article. -->
<h2><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a></h2>
<?php the_content(); ?>
<!-- Affiche la date de publication, l'auteur et les catégories de l'article. -->
<footer class="postmetadata">
  <p><small>Publié le <?php the_time('l d M Y'); ?>  par <?php the_author() ?> dans la
catégorie <?php the_category(', '); ?></small></p>
</footer>
</article> <!-- Fin de l'article -->
<!-- Fin de La Boucle -->
<?php comments_popup_link('Pas de commentaires', '1 Commentaire', '% Commentaires'); ?>
<!-- Permettre l'édition de l'article dans la page web -->
<?php edit_post_link('Editer', '<p>', '</p>'); ?>
<?php endwhile; ?>
<?php else: ?>
<!-- Le premier "if" teste l'existence d'articles à afficher.
La partie "else" indique que faire si ce n'est pas le cas. -->
<p>Aucun contenu à afficher.</p>
<!-- Fin -->
<?php endif; ?>
</section>
</main>
<!-- Affichage du footer -->
<?php get_footer(); ?>
```

Affichage de la barre latérale

La barre latérale sera utilisée plus tard dans les pages de type blog.
On y ajoute généralement les informations suivantes :

- Le formulaire de recherche
- Le calendrier par défaut de Wordpress
- La liste des catégories

- Les derniers articles de blog
- La liste des pages
- Les archives

Création du fichier sidebar.php

Pour le moment, nous allons juste afficher le calendrier de Wordpress.

```
<aside>
    <ul>
        <li id="calendar">
            <h2>Calendrier</h2>
            <?php get_calendar(); ?>
        </li>
    </ul>
</aside>
```

On pourrait également insérer d'autres informations, mais nous n'allons pas nous y attarder.
Nous verrons plus loin comment widgetiser la sidebar afin de pouvoir afficher par un simple glisser les informations que nous voulons y insérer.

Modification du fichier index.php

Pour afficher la sidebar, il faut utiliser une fonction du codex de Wordpress :
Avant le </main>, ajoutez ce code :

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

Modification du fichier style.css

La sidebar s'affiche en dessous du contenu. Pour l'afficher à droite, il va falloir aller modifier notre fichier css.

- Faites flotter le main content à gauche et la sidebar à droite.
- Largeur du contenu : 74%
- Largeur de la sidebar : 24%
- Laissez 2% de marge entre les 2 éléments.
- Supprimez les puces des ul de l'aside.

Modification du fichier header.php

Jusqu'à présent, la feuille de style du thème n'est pas chargée.
Il faut tout simplement ajouter cette ligne avant la balise </head> :

```
<link rel="stylesheet" href="<?php echo get_stylesheet_uri(); ?>">
```

Création du fichier functions.php

On va créer ce nouveau fichier dans le dossier de notre thème.
Il va nous permettre d'implémenter de nouvelles fonctionnalités à notre thème :

- Gérer la balise title (pour le moment, elle est encodée en dur) et plus généralement l'affichage des balises meta
- Supprimer la balise meta generator
- Création de widgets
- Permettre d'ajouter à l'aide d'un simple clic une image à la une dans un article ou une page
- Ajouter le logo à l'aide d'un simple clic
- Modifier la couleur de fond à l'aide d'un simple clic
- Ajout d'emplacements pour les menus du site
- Widgetiser la sidebar
- Charger la typo FontAwesome

Title tag

Pour le moment, la balise <title> est hardcodée.
Nous allons récupérer les informations des réglages de Wordpress.

Modification du fichier functions.php

```
/*
* Add support for title tag.
*/
add_theme_support('title-tag');
```

Modification du fichier header.php

On supprime la ligne insérée au début avec la balise <title>.
Et on va appeler la fonction **wp_head**, qui permet d'ajouter au thème les différents metatags.
!!! Cette fonction doit toujours être appelée juste avant la fermeture de la balise </head>.

```
<head>
  <meta charset="<?php bloginfo( 'charset' ); ?>">
  <meta name="viewport" content="width=device-width">
  <link rel="stylesheet" href="<?php echo get_stylesheet_uri(); ?>">
<?php wp_head(); ?>
</head>
```

Nous allons ensuite implémenter d'autres fonctionnalités dans ce fichier et modifier au fur et à mesure index.php, header.php, sidebar.php, footer.php

Supprimer l'affichage de la balise meta generator

Autant ne pas donner trop d'informations aux robots qui tentent de pirater votre site Wordpress !

1ère méthode

Cette méthode est généralement utilisée mais incomplète, car la version Wordpress reste affichée dans les flux RSS.
Ajoutez ce code dans le fichier **fonctions.php** créé :

```
<?php
/*
* Remove meta generator tag.
*/
remove_action("wp_head", "wp_generator");
?>
```

2ème méthode

Cette méthode est meilleure car elle supprime le meta generator partout.

Ajoutez ce code :

```php
<?php
function wp_remove_version() {
  return '';
}
add_filter('the_generator', 'wp_remove_version');
?>
```

Création d'un widget pour la sidebar

Depuis la version 2.2 de Wordpress, les Sidebar Widgets permettent de modifier la sidebar sans avoir à toucher au code dans les templates. Mais il faut le prévoir dans le thème.

Modification du fichier functions.php

Si vous n'avez qu'un seul widget, vous pouvez utiliser cette syntaxe :

```php
<?php if ( function_exists('register_sidebar') ) register_sidebar(); ?>
```

Si vous avez plus d'une colonne dans votre sidebar, entrez le chiffre correspondant au nombre de colonnes dans la dernière ligne de php. Par exemple, si vous avez deux colonnes, vous devez avoir le code suivant :

```php
<?php if ( function_exists('register_sidebar') ) register_sidebar(2); ?>
```

Allez dans l'administration de Wordpress, onglet Apparence => Widgets
Vous pouvez maintenant bouger chaque module de votre sidebar comme vous le souhaitez !
WordPress passera désormais par les Sidebar Widgets pour l'affichage de la sidebar.
Pour Mindgeek, nous allons plutôt utiliser cette syntaxe plus sophistiquée :

```php
/**
 * Add a sidebar.
 */
function mindgeek_widgets_init() {
  register_sidebar( array(
    'name'          => __( 'Colonne latérale du blog', 'mindgeek' ),
    'id'            => 'sidebar-1',
    'description'   => __( 'Ajoutez ici des widgets pour les faire apparaître dans votre colonne latérale d'articles de blog ou de pages d'archives. ', 'mindgeek' ),
    'before_widget' => '<li id="%1$s" class="widget %2$s">',
    'after_widget'  => '</li>',
    'before_title'  => '<h2 class="widgettitle">',
    'after_title'   => '</h2>',
  ) );
}
add_action( 'widgets_init', 'mindgeek_widgets_init' );
```

Modification du fichier sidebar.php

Pour ajouter la sidebar sur une colonne, on va ajouter cette ligne entre le 1er <ul>... du <aside> et supprimer le code inséré auparavant :

```php
<aside>
    <ul>
<?php if ( !function_exists('dynamic_sidebar') || !dynamic_sidebar() ) : ?>
<?php endif; ?>
    </ul>
</aside>
```

Ajout d'un custom logo dans le thème

Afin de pouvoir modifier le logo sans devoir toucher au code du template, nous allons permettre de le charger via l'interface de Wordpress.

Modification du fichier functions.php

```
/*
* Enable custom logo.
*/
add_theme_support( 'custom-logo', array(
  'height'      => 132,
  'width'       => 132,
  'flex-height' => true,
  'flex-width'  => true,
  'header-text' => array( 'site-title', 'site-description' ),
) );
function mindgeek_the_custom_logo() {
  if ( function_exists( 'the_custom_logo' ) ) {
    the_custom_logo();
  }
}
```

Allez dans le menu Apparence => Personnaliser puis « Identité du site ».
Vous devez voir apparaître la possibilité de sélectionner un logo. Il ne vous reste qu'à attacher le logo de Mindgeek.

Modification du fichier header.php

Pour afficher le logo sur la page, nous allons modifier le header. Si un logo a été défini, on l'affichera, sinon on affiche le titre et la description du blog comme précédemment.

```
    <!-- Affichage du logo ou de l'info du blog -->
    <?php
      $custom_logo_id = get_theme_mod( 'custom_logo' );
      $image = wp_get_attachment_image_src( $custom_logo_id , 'full');
    ?>
    <?php if ($custom_logo_id > 0) : ?>
     <p class="logo"><a href="<?php echo esc_url(home_url('/')); ?>" title="<?php echo
esc_attr(get_bloginfo('name', 'display')); ?>" rel='home'><img src='<?php echo
$image[0]; ?>' alt='<?php echo esc_attr(get_bloginfo('name', 'display')); ?>'></a></p>
    <?php else : ?>
    <!-- Affichage du titre du blog -->
    <h2><a href="<?php bloginfo('url'); ?>"><?php bloginfo('name');?></a></h2>
    <!-- Affichage de la description du blog -->
    <p><?php bloginfo('description'); ?></p>
    <?php endif; ?>
```

Personnalisation de la couleur de fond via l'interface

Comme pour le logo, nous allons permettre de pouvoir modifier la couleur de fond par un simple clic bouton.

Modification du fichier functions.php

```
// Set up the WordPress core custom background feature.
add_theme_support( 'custom-background', apply_filters(
'mindgeek_custom_background_args', array(
    'default-color' => 'FFFFFF',
    'default-image' => '',
) ) );
```

Allez dans le menu Apparence => Personnaliser puis « Couleurs ». Vous pouvez pouvoir modifier la couleur d'arrière-plan.

Modification du fichier header.php

Remplacer la balise <body> par cette ligne (le nom de la classe doit être le même que celui déclaré dans le fichier des fonctions) :

```
<body class="custom-background">
```

Ajout de la navigation

Wordpress permet de définir les emplacements pour la navigation.
On va ici en définir 2 :

- 1 dans le header
- 1 dans le footer

Modification du fichier functions.php

```
function register_nav() {
 register_nav_menus(
   array(
   'header-menu' => __( 'Header Menu' ),
   'footer-menu' => __( 'Footer Menu' )
   )
 );
}
add_action( 'init', 'register_nav' );
```

Allez dans l'onglet Apparence => Menus. Et créez les 2 menus suivants :

- Menu du haut
- Menu du pied de page

Pour chaque menu, ajoutez la page d'accueil et la page d'exemple. Enregistrez le menu.
Allez dans l'onglet « Gérer les emplacements » et assigner les 2 menus aux bons emplacements du thème.

Modification du fichier header.php

On ajoute aussi ces 2 lignes avant la fermeture avant le </header> :

```
<!-- Affichage de la navigation -->
<nav><?php wp_nav_menu( array( 'theme_location' => 'header-menu' ) ); ?></nav>
```

Modification du fichier footer.php

On ajoute ces 2 lignes juste avant le copyright :

```
<!-- Affichage de la navigation -->
<nav class="container"><?php wp_nav_menu( array( 'theme_location' => 'footer-menu' ) );
?></nav>
```

Ajouter Font Awesome

Modification du fichier functions.php

On prend celui du CDN de Bootstrap

```
/*
* Enqueue Font Awesome
*/
function mindgeek_enqueue_scripts() {
    wp_enqueue_style( 'font-awesome', '//maxcdn.bootstrapcdn.com/font-
awesome/4.3.0/css/font-awesome.min.css' );
}
add_action( 'wp_enqueue_scripts', 'mindgeek_enqueue_scripts' );
```

Ajout des typos Google Fonts

Modification du fichier header.php

Avant l'appel à la fonction get_stylesheet_url, ajoutez :

```
  <link href="https://fonts.googleapis.com/css?family=Candal" rel="stylesheet">
  <link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet">
```

Autres manières

Faire un enqueue au niveau du fichier fonctions.php

```
function mindgeek_enqueue_google_fonts() {
    wp_enqueue_style( candal, 'https://fonts.googleapis.com/css?family=Candal );
}
add_action( 'wp_enqueue_scripts', 'mindgeek_enqueue_google_fonts' );
```

Importer via la feuille de style => modification du fichier style.css

Placer ces 2 lignes en tête de votre fichier

```
@import url(http://fonts.googleapis.com/css?family=Candal);
@import url(http://fonts.googleapis.com/css?family=Open+Sans);
```

Les includes de code dans Wordpress

Alors qu'en php pur et dur, nous utilisons les instructions include(_once) ou require(_once), nous allons découvrir une autre fonction de Wordpress : il s'agit de la fonction get_template_part.
Nous allons d'abord créer dans notre thème un nouveau répertoire qui s'appelera **template-parts**.
Et tout d'abord créer un include pour le moteur de recherche que nous allons nommer **searchform.php** :

```
<?php
/**
 * Template part for displaying posts.
 * @link https://codex.wordpress.org/Template_Hierarchy
 * @package mindgeek
 */
?>
<section class="search">
     <form method="get" id="searchform" action="<?php bloginfo('home'); ?>/">
          <p>
               <input type="text" value="<?php the_search_query(); ?>" name="s" id="s"
placeholder="Que recherchez-vous ?"/>
          </p>
     </form>
</section>
```

Ensuite nous allons l'inclure dans notre page index.php entre le get_header() et le <main> :

```
<?php get_template_part( 'template-parts/searchform' ); ?>
```

Vous remarquerez que contrairement au php, on ne précise pas l'extension « .php ».
Ensuite procéder de même avec la boucle que vous pouvez inclure dans un fichier **content.php** et remplacer **the_content()** par **the_excerpt()**.
La fonction **the_excerpt()** affiche un résumé du contenu plutôt que le contenu entier.

```
<?php
/**
 * Template part for displaying posts.
 *
 * @link https://codex.wordpress.org/Template_Hierarchy
 * @package mindgeek
 */
?>
<!-- Début de la boucle. -->
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
<article class="post">
<!-- Affiche le Titre en tant que lien vers le Permalien de l'Article. -->
<h2><a href="<?php the_permalink(); ?>"><?php the_title(); ?></a></h2>
<?php the_excerpt(); ?>
<!-- Affiche la date de publication, l'auteur et les catégories de l'article. -->
<footer class="postmetadata">
  <p><small>Publié le <?php the_time('l d M Y'); ?>  par <?php the_author() ?> dans la
catégorie <?php the_category(', '); ?></small></p>
</footer>
</article> <!-- Fin de l'article -->
<!-- Afficher le nombre de commentaires -->
<p><?php comments_popup_link('Pas de commentaires', '1 Commentaire', '% Commentaires');
?></p>
<!-- Permettre l'édition de l'article dans la page web -->
<?php edit_post_link('Editer', '<p>', '</p>'); ?>
```

```
<!-- Fin de La Boucle -->
<?php endwhile; ?>
<?php else: ?>
<!-- Le premier "if" teste l'existence d'Articles à afficher. Cette -->
<!-- partie "else" indique que faire si ce n'est pas le cas. -->
<p>Aucun contenu à afficher.</p>
<!-- Fin -->
<?php endif; ?>
```

Et dans le fichier index.php remplacer le code qui est à présent dans l'include par :

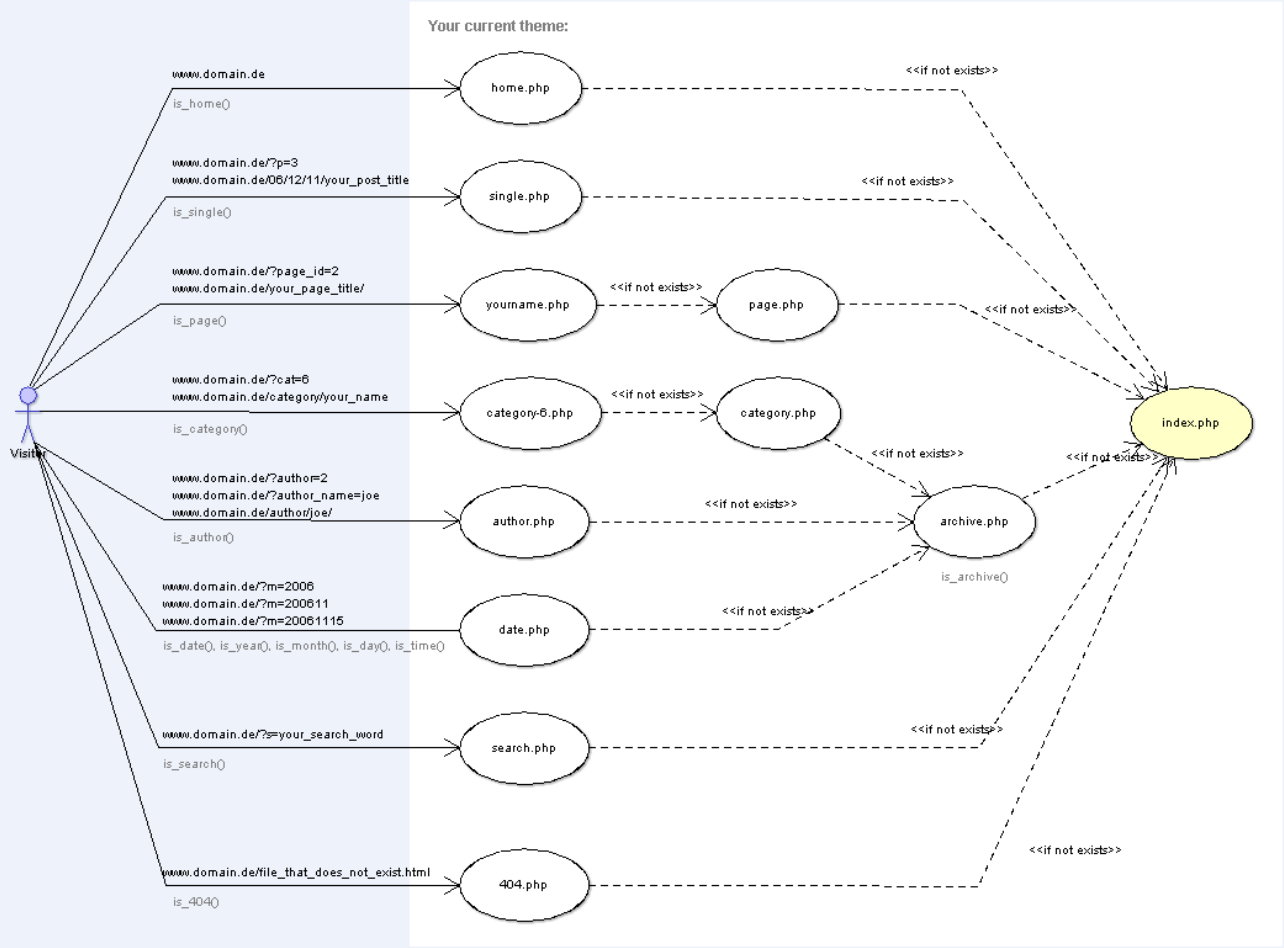
```
<?php get_template_part( 'template-parts/content', get_post_format() ); ?>
```

Vérifiez que l'affichage de votre page est correct.

Création des sous-templates

Ces sous-templates vont nous permettre d'afficher correctement les archives, les résultats des recherches, les pages d'articles ou les pages du blog. Ils vont nous permettre de « styliser » différemment certaines pages comme la page d'accueil. Ces sous-templates ne sont pas obligatoires pour faire tourner le site/blog mais sont fortement recommandés.

Hiérarchie des templates Wordpress



Le template page.php

Ce template permet d'afficher le contenu des pages du site sans les metadata.

Création du fichier template-parts/content-page.php

On copie d'abord le contenu du fichier « template-parts/**content.php** » sous « template-parts/**content-page.php** ».
Ensuite, on supprime la partie postmetadata et l'affichage des commentaires.
Sur une page, on va afficher le contenu complet plutôt que juste une introduction.
Il faut donc remplacer **the_excerpt()** par **the_content().**
On remplace cette ligne :

```
<!-- Permettre l'édition de l'article dans la page web -->
<?php edit_post_link('Éditer', '<p>', '</p>'); ?>
```

Par celle-ci

```
<!-- Permettre l'édition de l'article dans la page web -->
<?php edit_post_link('Modifier cette page', '<p>', '</p>'); ?>
```

Création du fichier page.php (à la racine du thème)

On copie le fichier **index.php** en **page.php** et on inclut le nouvel include créé :

On remplace la section du main par celle-ci :

```
  <section class="page">
<?php get_template_part( 'template-parts/content', 'page' ); ?>
  </section>
```

Le template single.php

Ce fichier est à créer à la racine du thème.

Ce template permet d'afficher un article de blog. On copie le fichier index.php sous single.php.
On va reprendre le contenu de l'index et y faire quelques petites modifications pour afficher ou supprimer quelques informations.
On va modifier le code du fichier template-parts/content.php pour afficher le post complet ou l'extrait suivant qu'on est sur l'aperçu des articles ou l'affichage de l'article seul.

```
<?php if (is_single()) :?>
<!-- Affiche le post complet -->
<?php the_content(); ?>
<?php else: ?>
<!-- Affiche une introduction de l'article -->
<?php the_excerpt(); ?>
<?php endif; ?>
```

Idem pour l'affichage du nombre de commentaires ou la possibilité d'en poster un :

```
<?php if (is_single()) :?>
<!-- Permettre la publication de commentaires -->
<section class="comments-template"> <?php comments_template(); ?> </section>
<?php else: ?>
<!-- Afficher le nombre de commentaires -->
<p><?php comments_popup_link('Pas de commentaires', '1 Commentaire', '% Commentaires');
?></p>
<?php endif; ?>
```

Voir aussi le tutoriel d'Open Class Room : https://openclassrooms.com/courses/propulsez-votre-site-avec-wordpress/les-themes-7

Le template archive.php

Ce fichier est à créer à la racine du thème (cf index.php).

Ce fichier va nous permettre d'afficher uniquement les premières lignes des articles.
Afficher les premières lignes des articles donne déjà un aperçu au visiteur de ce qui est publié sur le blog.
On va tout simplement copier le contenu de l'index (index.php) dans un nouveau fichier que l'on va nommer archive.php. Le contenu affiché par ce template sera le même que l'index. Mais on va utiliser plutôt ce code pour n'afficher que les premières lignes des articles.

Le template search.php

Ce fichier est à créer à la racine du thème.

On copie tout simplement le fichier **archive.php** sous **search.php**.
Tester le fonctionnement en lançant une recherche.

Le template comments.php

Ce fichier est à créer à la racine du thème.

Insérer le code de ce fichier : http://www.fran6art.com/documents/comments.txt

Gestion des images à la une

Les images à la une sont un moyen pour l'utilisateur d'attacher une image à un post.
Ici, on choisit d'inclure les images à la une uniquement pour les contenus de type article.

Modification du fichier functions.php

```
/*
* Enable support for Post Thumbnails on posts.
*/
add_theme_support('post-thumbnails', array( 'post' ));
```

Allez dans un article. Dans la sidebar de droite, vous devez voir apparaître une boîte avec « Image mise en avant ».
Ajouter en une.

Modification du fichier template-parts/content.php

Insérer ce code entre l'affichage du h2 et l'appel de la fonction the_excerpt :

```
<!-- Affiche la miniature de l'image -->
<?php
  $image_attributes = wp_get_attachment_image_src( get_post_thumbnail_id( $post->ID ));
  //var_dump($image_attributes);
  if ( $image_attributes ) :?>
    <img class="left" src="<?php echo $image_attributes[0]; ?>" width="<?php echo
$image_attributes[1]; ?>" height="<?php echo $image_attributes[2]; ?>" />
<?php endif; ?>
```

Vous devriez voir apparaître une miniature de l'image définie pour l'article.

Ajouter des champs personnalisés

On souhaite n'afficher certains contenus qu'entre une plage de date.
Pour le réaliser, nous allons créer 2 champs personnalisés **date_debut_publication** et **date_fin_publication**.
Voir aussi : https://codex.wordpress.org/fr:Utiliser_les_champs_personnalis%C3%A9s
https://www.pushaune.com/blog/tutoriels/bien-utiliser-les-champs-personnalises-avec-wordpress/

La première chose à faire est d'ouvrir le module Champs personnalisés dans la page d'ajout d'article du Back Office.
Créez les 2 champs en leur donnant un nom, qui sera le même pour tous les articles, et une valeur qui sera différente pour chaque contenu.
Cliquez sur le bouton « Ajouter un champ personnalisé ».
Ensuite, il va falloir filtrer les posts à l'aide de ce code :
On adapte le code comme suit :

```
<!-- Boucle sur les posts -->
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
<?php
  $today = date(Ymd);
  $date1 = get_post_meta($post->ID, date_debut_publication);
  $date2 = get_post_meta($post->ID, date_fin_publication);
?>
<?php if ((count($date1) == 0 || $date1[0] <= $today) && (count($date2) == 0 ||
$date2[0] >= $today)) : ?>
<article class="post">
// Votre code
…
```

Le template front-page.php

Ce fichier est à créer à la racine du thème.
Il sera utilisé pour personnaliser la page d'accueil.
Copiez le fichier **page.php** sous **front-page.php**
On va widgetiser l'image d'entête de la page d'accueil.

Modification du fichier functions.php

En dessous du 1er register_sidebar, ajoutez ce code :

```
  register_sidebar( array(
    'name'          => __( 'Bloc page accueil', 'mindgeek' ),
    'id'            => 'slider',
    'description'   => __( 'Ajoutez ici le texte pour l\'image de la home page.',
'mindgeek' ),
    'before_widget' => '<section id="%1$s" class="widget %2$s">',
    'after_widget'  => '</section>',
    'before_title'  => '<h1 class="widget-title">',
    'after_title'   => '</h1>',
  ) );
```

La nouvelle zone de widget doit apparaître dans l'interface d'administration de Wordpress.

Modification du fichier front-page.php

Après le get_header et avant l'inclusion du moteur de recherche, insérez ce code :

```
<?php
if ( is_active_sidebar( 'slider' ) ) :?>
  <section class="slider">
  <?php dynamic_sidebar( 'slider' ); ?>
  </section><!-- .widget-area -->
<?php endif; ?>
```

Créer un modèle de page

Vous pouvez créer vos propres types de page sous Wordpress.
Voici la marche à suivre :
1ère étape : création du fichier PHP
Dans le dossier de votre thème, créer un fichier qui portera le nom template-nom.php
Par exemple pour une page de contact : template-contact.php
Ce fichier est à créer à la racine du thème.
Le fichier devra contenir ces lignes :

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

Ensuite, copier le contenu du fichier **front-page.php** dans ce fichier et adaptez la structure.
Allez dans Wordpress pour créer une nouvelle page : dans les attributs de page (sidebar de droite), le nouveau modèle devrait apparaître dans la dropdown. Il ne vous reste qu'à le sélectionner.
On va widgetiser le plan Google Map de la même manière que l'on a fait pour l'image de fond dans la home page.

Modification du fichier fonctions.php

```
  register_sidebar( array(
    'name'          => __      ( 'Google Map', 'mindgeek' ),
    'id'            => 'map',
    'description'   => __( 'Ajoutez ici le widget Google Map pour la page contact.',
'mindgeek' ),
    'before_widget' => '<section id="%1$s" class="widget %2$s">',
    'after_widget'  => '</section>',
    'before_title'  => '<h1 class="widget-title">',
    'after_title'   => '</h1>',
  ) );
```

Contenu du fichier template-contact.php

```
<?php
/*
Template Name: Contact
*/
?>
<!-- Affichage de l'entête -->
<?php get_header(); ?>
<?php get_template_part( 'template-parts/searchform' ); ?>
<main class="clearfix">
  <section class="page container contact">
<?php get_template_part( 'template-parts/content', 'page' ); ?>
  </section>
<?php if ( is_active_sidebar( 'map' ) ) :?>
  <section class="map">
  <?php dynamic_sidebar( 'map' ); ?>
  </section><!-- .widget-area -->
<?php endif; ?>
</main>
<!-- Affichage du footer -->
```

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

Autres fonctionnalités de Wordpress

Afficher la liste des catégories (dans la sidebar)

```php
<li class="categories">
   <h2>Categories</h2>
   <ul>
     <?php wp_list_cats('sort_column=name&optioncount=1&hierarchical=0'); ?>
   </ul>
</li>
```

Afficher la liste des pages (dans la sidebar)

```php
<li class="pages">
<?php wp_list_pages('title_li=<h2>Pages</h2>'); ?>
</li>
```

Afficher les archives (dans la sidebar)

Les archives de Wordpress permettent de classer les articles. Le classement se fait généralement sur une base mensuelle.

```php
<li class="archives">
    <h2>Archives</h2>
<ul><?php wp_get_archives('type=monthly'); ?></ul>
</li>
```

Fonctions de référence

Voir https://codex.wordpress.org/Function_Reference

Fonctions d'includes

get_header() : affiche le contenu du fichier header.php
get_footer() : affiche le contenu du fichier footer.php
get_sidebar() : affiche le contenu du fichier sidebar.php
get_search_form() : affiche le formulaire de recherche
dynamic_sidebar() : affichage d'une sidebar précise : en paramètre le nom ou le numéro de la sidebar => par défaut c'est la 1ère qui s'affiche

Fonctions pour afficher le contenu des posts

have_post() : teste s'il existe des posts
the_title() : renvoie le titre d'une page
the_excerpt() : affiche le résumé d'un post
the_content() : affiche le contenu du post
the_author() : affiche le nom de l'auteur d'un post
the_permalink() : affiche l'URL du post dans la boucle
get_permalink() : renvoie l'URL complète de la page ou de l'article

La boucle

```
<?php if ( have_posts() ) : while ( have_posts() ) : the_post(); ?>
```

Elle doit être présente dans le fichier index.php ainsi que dans tous les fichiers du thème affichant des articles.

Tags conditionnels

is_front_page() : teste s'il s'agit de la page d'accueil, quel que soit son type (page de base ou page de type blog)
is_home() : c'est la page définie comme « post page » dans la configuration de Wordpress
is_single() : teste s'il s'agit d'un post unique
is_page() : teste s'il s'agit d'une page

Installer le corrigé sur votre ordinateur

1) Copiez le répertoire de wordpress-mindgeek dans le répertoire de base de votre serveur Apache
2) Créez une base de données, en choisissant le bon charset.
3) Éditez le dump de la base de données (fichier se terminant par .sql)
   - Remplacez le chemin de l'URL du site par le chemin sur votre machine (chercher quelque chose comme http://localhost).
   - Remplacez le **realpath** par celui de votre machine :
     - À la 3W Academy, on a quelque chose comme **/home/wali1**.
     - Sur Mac avec **MAMP** => **/Applications/MAMP/htdocs**.
     - Sur PC avec **WAMP** => **C:/wamp/www**
   - Vous devrez peut-être aussi modifier le charset de utf8mb4 vers utf8 si votre base de données a été créée avec ce charset.
4) Importer le fichier .sql dans la base de données que vous avez créée.
5) Modifiez le fichier de configuration de Wordpress **wp-config.php** et adaptez les paramètres de connexion à la base de données avec les données de votre serveur :
   a. DB_NAME
   b. DB_USER
   c. DB_PASSWORD
   d. DB_HOST
6) Vous devrez peut-être adapter aussi le fichier **.htaccess** qui se trouve à la racine du site.
   Il s'agit d'un fichier caché qui par défaut n'est pas affiché dans l'explorer.
   Pour l'afficher, il faut demander à explorer d'afficher les fichiers cachés ou aller dans terminal et lancer la commande **ls -al** dans le répertoire. Sous Terminal, vous pouvez éditer le fichier avec vi (mais il faut bien le connaître) ou avec nano.
   Vous devrez éventuellement modifier ces 2 lignes en adaptant le chemin du site, sinon la réécriture d'URL ne fonctionnera pas et vous ne pourrez afficher que la page d'accueil.
   RewriteBase /wordpress-mindgeek/
   RewriteRule . /wordpress-mindgeek/index.php [L]
7) Pour rentrer dans l'administration du site : login => admin, mdp => admin (surtout ne jamais faire çà en production !!!)

Ressources

Création d'un thème Wordpress : http://www.fran6art.com/tutoriels/creation-theme-wordpress-tutorial-6-le-header/
Création d'un modèle de page : https://wpchannel.com/creer-modele-page-wordpress/
Wordpress theme Handbook (en anglais) : https://developer.wordpress.org/themes/basics/theme-functions/