Make Wordpress Great Again !

Avril 2021

Rappel

Thème gratuit

Avantages

- Gratuit !
- Certains sont très bien faits

Inconvénients

- Manque de fonctionnalités
- Manque de support
- Peu ou pas de mise à jour
- Risque important de spam/malware/etc
- Mal codé

Thème premium

Avantages

- Qualité du thème
- Nombreuses fonctionnalités
- Support
- Mise à jour régulière
- Documentation
- Extensions premium
- Prix (60$)

Inconvénients

- Trop de fonctionnalités !
- Utilisation complexe (page builder, shortcodes, widgets)
- Options à rallonge
- Code obscur
- Démos trop vendeuses

Thème « from scratch »

Avantages

- Design unique
- Qualité du développement
- Respect des attentes du client
- Très évolutif

Inconvénients

- Complexité
- Délais de réalisation
- Tarif

Conclusion

Sauf cas très particulier, on évite :

- les thèmes gratuits
- les thèmes premium

Est-ce vraiment complexe de développer un thème ?

Démarrage

Installation de Wordpress

3 méthodes:

1. En dézippant l'archive officielle disponible sur:
   https://fr.wordpress.org/download/

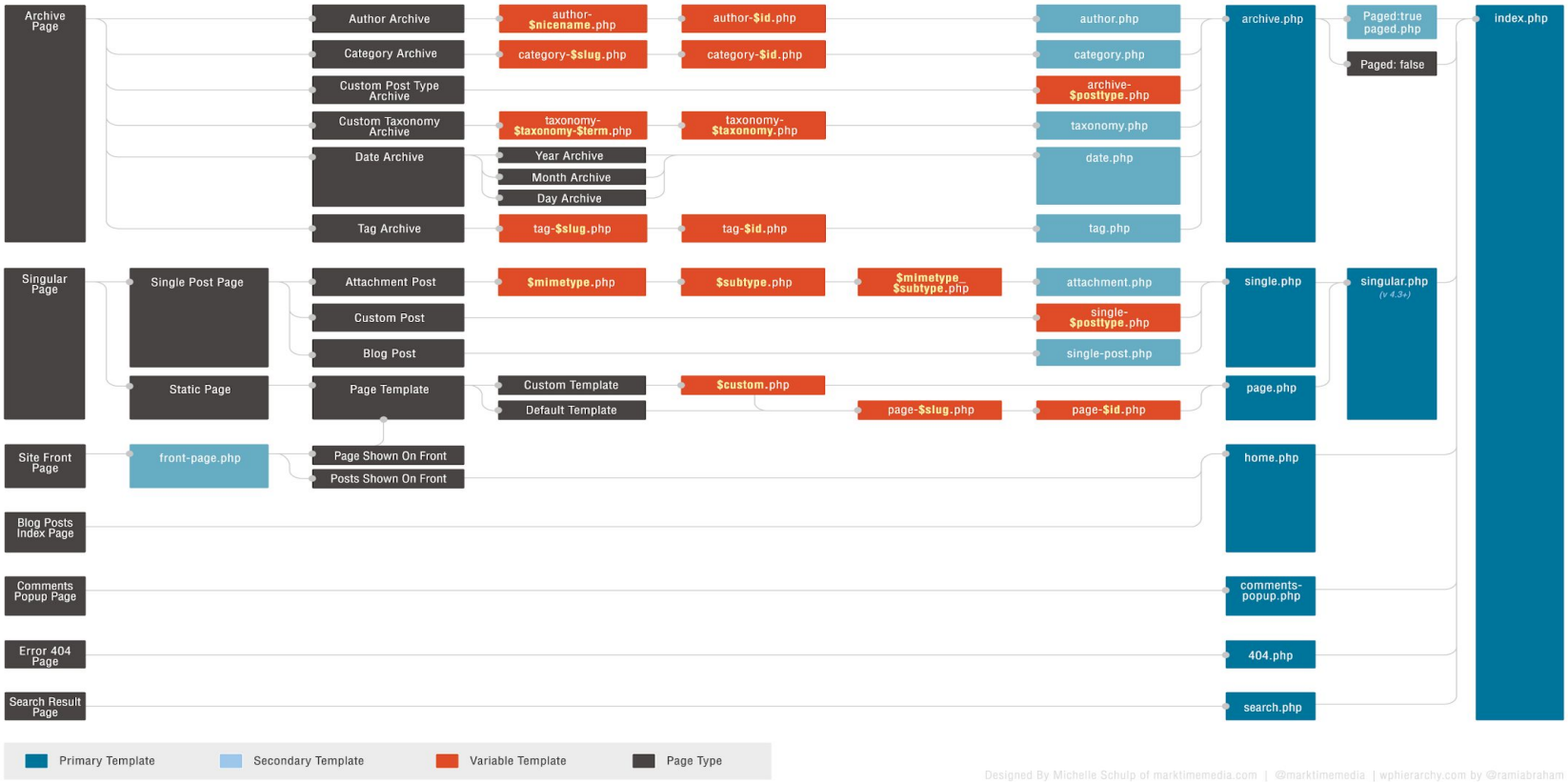
2. En ligne de commande via WP-CLI:
   https://wp-cli.org/fr/

3. En utilisant une image Docker:
   https://docs.docker.com/compose/wordpress/

Thème - hiérarchie

- Stockés dans le dossier wp-content/themes
- 3 thèmes pré-installés
- 3 fichiers importants:
  - style.css : contient les informations du thème
  - functions.php : contient les fonctions PHP utiles au thème
  - index.php : le template de base

Thème - hiérarchie



https://codex.wordpress.org/images/c/ca/Template_Hierarchy_2015.png

Thème - hiérarchie

- Accueil → front-page.php
- Article → single.php
- Page → page.php
- Catégorie → category.php
- Catégorie « News » → category-news.php
- 404 → 404.php

- Template de base → index.php

Dynamisation maquette

Utilisation d'un "starter theme": Underscore

https://underscores.me/

Avantages:

- Une base de démarrage avec le minimum vital
- CSS minimaliste avec un reset CSS
- Des exemples de fonctionnalités courantes
- Thème libre de droits et collaboratif (Github)

Création de notre thème

- Création d'un nouveau dossier dans wp-content/themes
- On copie à l'intérieur les fichiers présents sur le repository:

http://cpc.cx/ot8

Création de notre thème

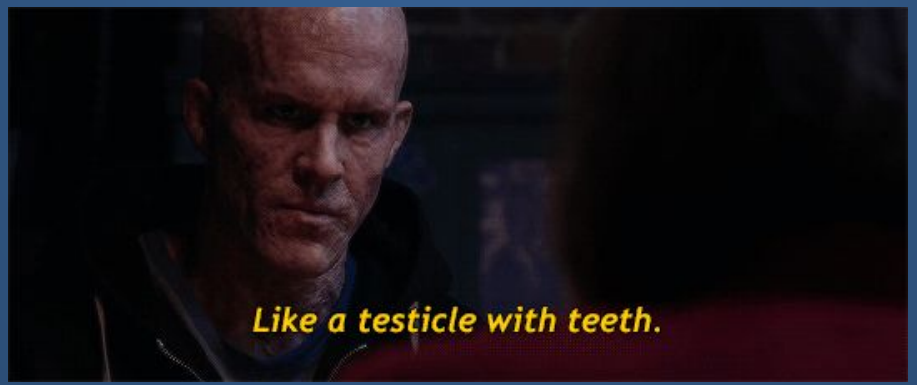
On ouvre style.css et on déclare notre thème:

```
/*
Theme Name: Bootstrap WordPress
Author: Cyril Bosson
Description: Thème pour apprendre à créer son propre thème
Version: 0.0.1
Tags: bootstrap
*/
```

On renomme le fichier index.html et index.php

On active le thème dans le gestionnaire de thème de l'administration

Cool ! … mais c’est super laid !?



Création de notre thème

On indique à Wordpress l'emplacement de notre fichier de style grâce à la fonction get_bloginfo():

```
<link href="blog.css" rel="stylesheet" />
```

Devient :

```
<link href="<?php echo get_bloginfo('template_directory'); ?>/blog.css" rel="stylesheet">
```

Divisons pour mieux coder !

On crée les fichiers suivants dans notre dossier:

- header.php
- sidebar.php
- footer.php
- content.php

header.php

1. On extrait le début du fichier jusqu'au début du contenu principal

2. On ajoute simplement la fonction wp_head avant la balise </head>

```
    <!-- Custom styles for this template -->
    <link href="<?php echo get_bloginfo('template_directory'); ?>/blog.css" rel="stylesheet">

    <?php wp_head() ?>
  </head>
  <body>
```

footer.php

1. On extrait à partir de la fin du contenu principal et jusqu'à la fin du fichier

2. On ajoute simplement la fonction wp_footer avant la balise </body>

```
</footer>
<?php wp_footer() ?>
</body>
</html>
```

sidebar.php / content.php

1. On extrait le code de la sidebar et on l'enregistre dans sidebar.php

2. On extrait le code d'un article (on peut supprimer les autres) et on l'enregistre dans content.php

index.php

On ajoute à présent les références aux différentes parties:

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

<?php get_template_part( 'content' ); ?>

<?php get_sidebar(); ?>

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

Paramètres généraux

Administration > Réglages > Général

On renseigne le titre et le slogan

On les utilise dans le thème grâce à la fonction get_bloginfo()

```
<?php echo get_bloginfo( 'name' ); ?>

<?php echo get_bloginfo( 'description' ); ?>
```

On utilise la fonction home_url() pour récupérer l'URL de base

```
<a href="<?php echo home_url() ?>">
```

La boucle

Code de base de la boucle Wordpress

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

  <!-- contents of the loop -->

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

Dans notre cas, la boucle devient:

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

  <?php get_template_part( 'content' ); ?>

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

Contenu

Utilisation des fonctions suivantes dans notre fichier content.php

the_title() → Affiche le titre du contenu

the_date() → Affiche la date de création du contenu

the_author() → Affiche le nom du rédacteur

the_content() → Affiche le contenu

Article

1. On duplique le fichier index.php et on le renomme en single.php

2. On supprime le bloc du haut

3. Dans la fonction get_template_part(), on remplace content par content-single

4. On duplique le fichier content.php et on le renomme en content-single.php

Article

Dans le fichier content.php, on ajoute un lien dans le titre de l'article et on utilise la fonction the_permalink() comme valeur de l'attribut "href"

```
<a href="<?php the_permalink() ?>"><?php the_title() ?></a>
```

On remplace la fonction the_content() par the_excerpt()

Navigation

On utilise les fonctions suivantes pour générer les liens:

- next_posts_link()
- previous_posts_link()

```
<nav class="blog-pagination">
   <?php next_posts_link( 'Older' ) ?>
   <?php previous_posts_link( 'Newer' ) ?>
</nav>
```

Sidebar

- On dynamise la liste des archives:

```
<h4 class="font-italic">Archives</h4>
<ol class="list-unstyled mb-0">
   <?php wp_get_archives( 'type=monthly' ); ?>
</ol>
```

- Et nos infos personnelles:

```
<h4 class="font-italic">About</h4>
<p class="mb-0"><?php the_author_meta( 'description' ); ?></p>
```

Page statique

1. On duplique le fichier index.php et on le renomme en page.php

2. On supprime le bloc du haut, la sidebar et on affiche le contenu en pleine largeur

Actions et filtres

2 méthodes pour interagir avec les fonctions natives Wordpress:

Actions: des morceaux de codes qui s'exécute à la suite d'un événement de Wordpress (chargement de la page, affichage du header, etc)

Filters: des morceaux de codes qui vont transformer une valeur passée en entrée

Actions et filtres

Action … en action

```php
<?php

add_action("save_post", "publish_to_facebook");
function publish_to_facebook( $post_id ){
   // Ce code sera exécuté lorsqu'un utilisateur sauvegarde un article
}
```

Filtre … en action

```php
<?php

add_filter("the_title" , "capitalize_post_titles" );
function capitalize_post_titles( $post_title ){
    $title = ucwords(  $post_title  );
    return $title;
}
```

Actions et filtres

Les fonction add_action() et add_filter() acceptent les mêmes paramètres:

1. nom → nom de l'action ou du filtre existant
2. fonction → fonction à exécuter
3. priorité → ordre d'exécution dans l'ordre croissant
4. nb paramètres → nombre de paramètres acceptés

functions.php

Ajout des styles et scripts

```php
<?php

// Add scripts and stylesheets
add_action( 'wp_enqueue_scripts', 'custom_scripts' );
function custom_scripts()
{
     wp_enqueue_style( 'bootstrap', get_template_directory_uri() .
                          '/css/bootstrap.min.css', array(), '3.3.6' );

     wp_enqueue_style( 'blog', get_template_directory_uri() . '/css/blog.css' );

     wp_enqueue_script( 'bootstrap', get_template_directory_uri() .
                          '/js/bootstrap.min.js', array( 'jquery' ), '3.3.6', true );

}
```

functions.php

Gestion du meta <title> par Wordpress

```php
<?php

add_action( 'after_setup_theme', 'custom_theme_setup' );
function custom_theme_setup()
{
   // WordPress Titles
   add_theme_support( 'title-tag' );
}
```

functions.php

Activation des images à la une

```
function custom_theme_setup()
{
   ...
   add_theme_support( 'post-thumbnails' );
}
```

```
if ( has_post_thumbnail() ) {
      the_post_thumbnail();
}
```

```
if ( has_post_thumbnail() ) {
      the_post_thumbnail( 'thumbnail' );
}
```

functions.php

Déclaration d'un menu

```
function custom_theme_setup()
{
   ...
   register_nav_menus( array(
      'primary' => 'Menu principal',
    ) );
}
```

Menu principal

On remplace le menu par le code:

```php
<?php
  wp_nav_menu([
    'theme_location' => 'primary',
    'container' => false,
    'menu_class' => 'nav d-flex justify-content-between',
  ]);
```

Pour rendre compatible notre menu avec Bootstrap, on peut utiliser un Walker spécifique:

https://github.com/wp-bootstrap/wp-bootstrap-navwalker

Menu Bootstrap

```php
<?php

  require_once get_template_directory() . '/class-wp-bootstrap-navwalker.php';

  ...

  wp_nav_menu([
    'theme_location' => 'primary',
    'container' => false,
    'menu_class' => 'nav d-flex justify-content-between',
    'fallback_cb' => 'WP_Bootstrap_Navwalker::fallback',
    'walker' => new WP_Bootstrap_Navwalker(),

  ]);
```

Custom content type

On déclare notre contenu personnalisé dans functions.php

```
add_action( 'init', 'register_movie_post_type' );

function register_movie_post_type() {
   register_post_type( 'movie', array(
      'labels' => array(
         'name' => __( 'Movies' ),
         'singular_name' => __( 'Movie' ),
      ),
      'public' => true,
      'has_archive' => true,
      'supports' => array(
         'title',
         'editor',
         'thumbnail',
      )
   ) );
}
```

Boucle personnalisées

Pour parcourir nos contenus personnalisées, on utilise une boucle personnalisée via l'objet WP_Query()

```php
<?php

$custom_query = new WP_Query( array(
      'post_type' => 'movie',
      'orderby' => 'menu_order',
      'order' => 'ASC'
) );

while ( $custom_query->have_posts() ) : $custom_query->the_post();
      // Contents of the custom Loop
endwhile;

wp_reset_postdata();
```

Données personnalisées (custom meta)

Possibilité 1: from scratch

1. On crée un nouvelle boite sur l'administration avec l'action add_meta_boxes
2. On affiche nos champs
3. On contrôle les données et on les enregistre lors de la sauvegarde du post avec la fonction update_post_meta()
4. On affiche nos données personnalisées sur le front avec la fonction get_post_meta()

Données personnalisées (custom meta)

Possibilité 2: plugin Advanced Custom Fields

- Pas de code à écrire pour gérer le back
- Possibilité d'ajouter des données à tous les types de contenus de Wordpress
- Tous les types de base (text, checkbox, radio, select)
- De nombreux types avancées (upload, galerie, colorpicker, datepicker, Google Maps, onglets, …)
- Possibilité de développer ces propres types de champ

Advanced Custom Fields

1. On télécharge et on installe le plugin
2. On crée un nouveau groupe de champ dans ACF
3. On ajoute les champs suivants pour tous les articles:
    a. Temps de lecture
    b. Vidéo
    c. Fichier complémentaire
4. On ouvre notre template d'article content-single.php
5. On affiche nos données:

```
<?php the_field('my_custom_awesome_field'); ?>
```

Advanced Custom Fields

Afficher une donnée

```php
<?php the_field('my_custom_awesome_field'); ?>
```

Récupérer une donnée

```php
<?php echo get_field('my_custom_awesome_field'); ?>
```

Récupérer une donnée d'un article précis

```php
<?php echo get_field('my_custom_awesome_field', 123); ?>
```

Advanced Custom Fields

Récupérer différents types de donnée

```php
<?php

   // Donnée de l'utilisateur 2
   $value = get_field( 'my_field', 'user_2' );

   // Donnée de la catégorie 3
   $value = get_field( 'my_field', 'category_3' );

   // Donnée du contenu personnalisé "event" 4
   $value = get_field( 'my_field', 'event_4' );
```

Récupérer une donnée brute

```php
   <?php echo get_field('my_image_field', 123, true); ?>
```