

votre propre thème *part 2*

# **menus, widgets et autres paramètres**

**a.k.a. functions.php**



# **kahoot!**

à vous de jouer

A photograph of a red car parked in front of a building. The building has a green door and a brick wall. A tire is lying on the ground near the door. The scene is outdoors with some snow on the ground.

01

# **comment déboguer wordpress?**



Il était une fois, un étudiant de la Tim qui avait peur du grand méchant PHP. Mère-grand était derrière lui et l'encourageait sans arrêt. Il finit par prendre confiance en lui et tout alla bien. Et soudain, le thème apparut, les pages prenaient forme quand tout à coup..





est apparu..

**un problème..**

**vraiment épouvantable..**



# une erreur critique!!!

Une erreur critique est survenue sur votre site.

[En apprendre plus sur le débogage de WordPress.](#)



# **la page blanche de la mort**

(pour les anciennes versions de WordPress)

# on fait quoi?

- d'abord, on pleure un petit peu 🥹
- ensuite, on se rappelle que WordPress laisse des traces

toutes les erreurs sont stockées  
dans un fichier nommé **debug.log**  
à la racine de l'installation.

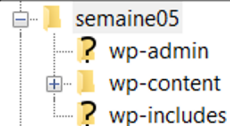
Filename	Filesize	Filetype
..		
wp-admin		File folder
wp-content		File folder
wp-includes		File folder
.htaccess	510	HTACCESS File
<b>error_log</b>	<b>1,612</b>	<b>File</b>
index.php	420	PHP File
license.txt	19,935	Text Document
readme.html	9,103	HTML File
wp-activate.php	6,939	PHP File
wp-blog-header.php	369	PHP File

# debug.log?

- c'est chiant à consulter
- ça peut contenir des milliers de lignes qui sont là depuis longtemps

WordPress peut nous afficher *live*, dans le navigateur, les erreurs. Il ne suffit que d'activer le mode debug en définissant les propriétés `WP_DEBUG` ET `WP_DEBUG_LOG` à `true` dans le fichier `wp-config.php` à la racine de votre installation.

Remote site: /public\_html/cours/cw4/semaine05



Filename	Filesize	Filetype
license.txt	19,550	Text Document
readme.html	9,103	HTML File
wp-activate.php	6,939	PHP File
wp-blog-header.php	369	PHP File
wp-comments-post.php	2,283	PHP File
wp-config-sample.php	3,600	PHP File
wp-config.php	3,930	PHP File
wp-cron.php	3,955	PHP File
wp-links-opml.php	2,504	PHP File

```
// define('WP_DEBUG', false);
```

```
define('WP_DEBUG', true);
```

```
define('WP_DEBUG_LOG', true);
```

## pendant qu'on développe le site

on ajoute

```
// define('WP_DEBUG', false);  
  
define('WP_DEBUG', true);
```

## quand le site va en ligne

on remet

```
define('WP_DEBUG', false);  
  
// define('WP_DEBUG', true);
```

# ce n'est pas parfait, mais ça aide!

## → on voit les erreurs

**Parse error:** syntax error, unexpected end of file, expecting elseif (T\_ELSEIF) or else (T\_ELSE) or endif (T\_ENDIF) in `/home/jfleblanc/public_html/cours/cw4/semaine05/wp-content/themes/mtl_lumiere/index.php` on line 34

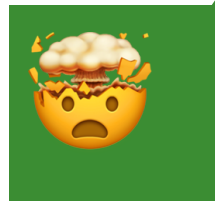
## → on peut corriger

```
<div id="principal">
  <?php if( have_posts() ) : ?>
    <?php while (have_posts()) : the_post(); ?>
      <article <?php post_class(); ?> id="article<?php the_ID(); ?>">
        ...
      </article>
    <?php endwhile; ?>
  </div>
```

endif manquant



# parfois, on doit chercher un peu



← → ↻ 🔒 jfleblanc.dectim.ca/cours/cw4/semaine05/ ☆ ⚙️ bw 🤖 🌈 | J ⋮

Elements Console Sources Network Performance >> ⋮ ✕

```
<!doctype html>
<html lang="fr">
  <head>
    <meta charset="UTF-8">
    <title>
      "Laboratoire semaine 05<br />
      <b>Fatal error</b>: Uncaught Error: Call to undefined function
      wp_tile() in /home/jfleblanc/public_html/cours/cw4/semaine05/wp-
      content/themes/mtl_lumiere/header.php:7
      Stack trace:
      #0 /home/jfleblanc/public_html/cours/cw4/semaine05/wp-
      includes/template.php(722): require_once()
      #1 /home/jfleblanc/public_html/cours/cw4/semaine05/wp-
      includes/template.php(671): load_template('/home/jfleblanc...', true)
      #2 /home/jfleblanc/public_html/cours/cw4/semaine05/wp-
      includes/general-template.php(41): locate_template(Array, true)
      #3 /home/jfleblanc/public_html/cours/cw4/semaine05/wp-
      content/themes/mtl_lumiere/index.php(1): get_header()
      #4 /home/jfleblanc/public_html/cours/cw4/semaine05/wp-
      includes/template-loader.php(98): include('/home/jfleblanc...')
      #5 /home/jfleblanc/public_html/cours/cw4/semaine05/wp-blog-
      header.php(19): require_once('/home/jfleblanc...')
      #6 /home/jfleblanc/public_html/cours/cw4/semaine05/index.php(17):
      require('/home/ifleblanc...')
```

## remarque #1

- une seule erreur est affichée à la fois
- dès que le PHP frappe une erreur, il arrête tout



PHP est un préprocesseur, c'est à dire qu'il va interpréter le code avant de le rendre, comme Sass que vous avez utilisé. Dès qu'il frappe une erreur, il arrête son interprétation et donc, n'affiche plus rien par la suite.

## remarque #2

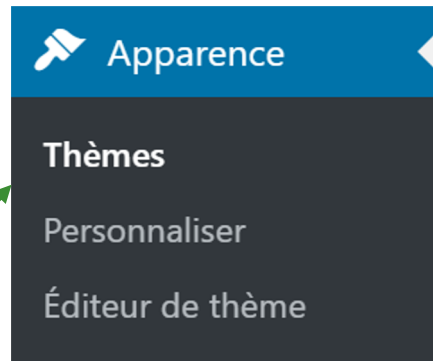
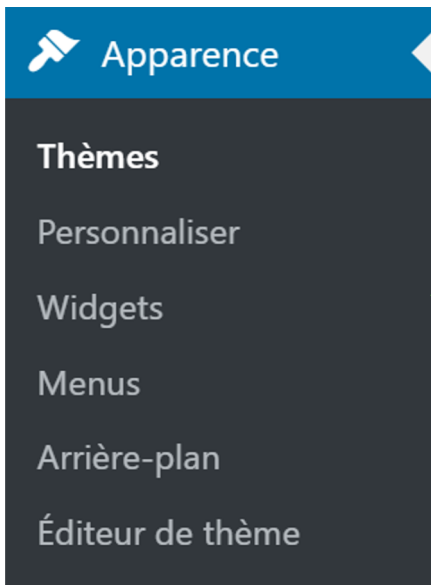
- quand vous avez une page blanche, vérifiez si l'admin est toujours accessible
- si oui, changez de thème
- si vous avez encore un pépin avec le nouveau thème, désactivez les extensions une à une

02

# les fonctions WordPress

elles sont rendues où, mes  
**images à la une?**

**certaines options ne sont  
affichées que lorsque les  
thèmes les supportent**



Pour que les menus et les widgets soient disponibles, le thème doit déclarer qu'il les supporte.

C'est possible en utilisant le fichier `functions.php` dont nous parlerons bientôt.

# comprendre le fichier functions.php

s'il est présent à la racine du thème, ce fichier PHP sera automatiquement chargé et exécuté à chaque requête

Il permet :

- de spécifier les fonctionnalités que supportera notre thème (images à la une, widgets, menus, etc.)
- d'ajouter des **filtres** et des **actions** modifiant dynamiquement le rendu des pages



**mise en garde**



# mise en garde

afin d'éviter beaucoup de douleur potentielle, il est fortement recommandé de ne pas mettre la balise de fermeture dans le fichier functions.php

```
<?php
```

```
// insérez le code PHP ici
```

```
// Ne mettez pas la balise de fermeture
```

en fait, on ne devrait jamais mettre la balise de fermeture **?>** dans les fichiers qui ne contiennent que du PHP



# fonctionnalités supportées

la fonction `add_theme_support()` permet de spécifier quelles fonctionnalités sont supportées par votre thème

→ pour autoriser l'utilisation des images à la une :

```
add_image_size('vignette', 220, 180);  
add_image_size('xxl', 2200, 1800);
```

→ pour que WordPress crée des copies des images à une dimension donnée :

```
add_theme_support('post-thumbnails');  
set_post_thumbnail_size(800, 480);
```

**<extension-regenerate-thumbnails>**

attention, ce n'est pas du vrai code



## Regenerate Thumbnails

[Installer](#)

Recrée les miniatures pour une ou plusieurs de vos images téléversées. Utile si vous changez de tailles d'images ou de thème.

[Plus de détails](#)

Par [Alex Mills \(Viper007Bond\)](#)



1 million et + installations  
actives

Dernière mise à jour : il y a 2 semaines

✓ Compatible avec votre version de WordPress

Lorsque vous modifiez les grandeurs de vos images. Cette petite extension est très intéressante pour vous assurer que WordPress régénère les images au bon format.

# regenerate thumbnails

→ toutes d'un coup

 Outils

Outils disponibles

Importer

Exporter

Santé du site

Exporter les données

Effacer les données

**Régénérer les miniatures d'images**

Régénérer les miniatures pour toutes les 9 pièces jointes.

Régénérer les miniatures pour les 9 images mises en avant uniquement.

### Tailles de miniatures

Voici toutes les tailles de miniatures qui sont actuellement enregistrées :

**thumbnail** : 150×150 pixels (Recadré pour correspondre.)

**medium** : 300×300 pixels (Redimensionné proportionnellement pour respecter les tailles)

**medium\_large** : 768×0 pixels (Redimensionné proportionnellement pour respecter les tailles)

**large** : 1024×1024 pixels (Redimensionné proportionnellement pour respecter les tailles)

**1536x1536** : 1536×1536 pixels (Redimensionné proportionnellement pour respecter les tailles)

**2048x2048** : 2048×2048 pixels (Redimensionné proportionnellement pour respecter les tailles)

→ une à la fois

☐



Shenzhen

Shenzhen.jpg

[Modifier](#) | [Supprimer définitivement](#) | [Afficher](#) | [Régénérer les miniatures](#)

## comme toute bonne chose a une fin

Une fois terminé, vous pouvez désinstaller l'extension Regenerate Thumbnails.

À l'avenir, toutes les images que vous téléversez seront dans les bonnes dimensions que vous aurez spécifiées.



### Bonnes pratiques

Toujours effacer les extensions et les images qui ne servent plus. Vos sauvegardes en seront plus légères et vous aurez plus d'espace libre sur votre serveur.

**</extension-regenerate-thumbnails>**

on revient aux choses sérieuses



# retour à functions.php

il est aussi possible de spécifier le nombre maximum de mots pour l'affichage du texte résumé (`the_excerpt`)

dans functions.php :

```
function new_excerpt_length($length) {  
    return 20; // Nombre de mots limite  
}  
  
add_filter('excerpt_length', 'new_excerpt_length');
```

03

# **ajouter un menu**

**c'est quoi le plat du jour?**

# actions

la fonction `add_action()` permet d'ajouter une opération personnalisée à une certaine étape du processus de génération de la page

```
function ma_fonction() {  
    // insérez le code PHP ici  
}  
  
// la plupart des actions liées au thème  
// et extension pourront utiliser l'action init  
add_action('init', 'ma_fonction');
```

codex: référence `add_action()`

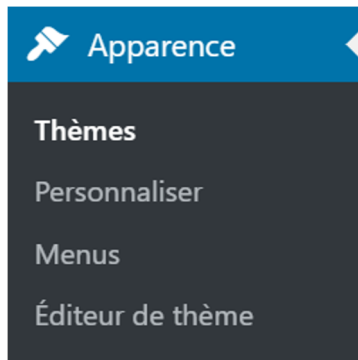
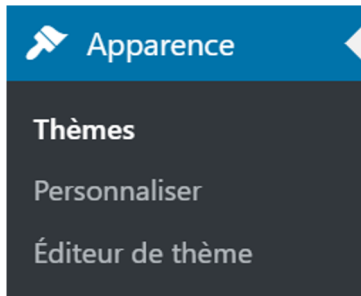


## enregistrer des emplacements pour les menus

- lorsque vous utilisez l'interface WordPress pour ajouter un **menu** ou un **widget**, vous indiquez au logiciel l'endroit où le menu/widget devrait s'afficher (ex.: sidebar, footer 1, menu\_principal, etc.)
- pour pouvoir permettre cela dans votre thème, vous devez **enregistrer des emplacements** prévus à cet effet et ajouter le code dans votre thème aux endroits désirés afin d'afficher le contenu de ces nouveaux emplacements

# enregistrer des emplacements pour les menus

Par défaut, WordPress n'active pas de menu dans un thème personnalisé. La première étape consiste donc à activer les options de menu de WordPress dans `functions.php`



# 1. l'indiquer à WordPress

→ enregistrer un emplacement de menu

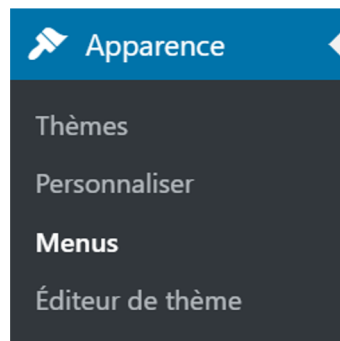
```
function creer_menu() {  
    register_nav_menu('menu_principal', 'Menu principal');  
}  
add_action('init', 'creer_menu');
```

→ enregistrer plusieurs emplacements de menus

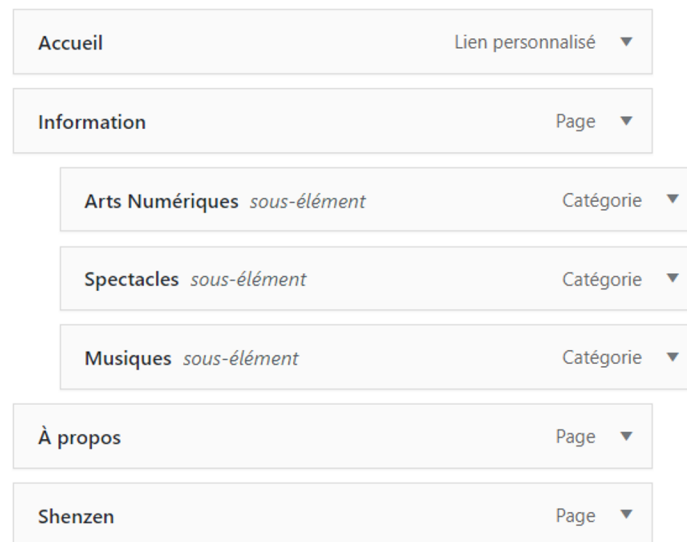
```
register_nav_menus(array(  
    'menu_principal' => 'Menu principal',  
    'menu_footer' => 'Menu du pied de page',  
    'menu_sociaux' => 'Menu réseaux sociaux',  
));
```

## 2. création et assignation du menu

il faut ensuite aller créer notre premier menu dans l'interface d'administration



n'oubliez pas  
d'assigner votre menu  
à un emplacement



### Réglages du menu

Ajoutez automatiquement des pages

☐ Ajouter automatiquement les pages de premier niveau à ce menu

Afficher l'emplacement

- ☒ Menu principal
- ☐ Menu du pied de page
- ☐ Menu réseaux sociaux



### 3. sélection d'un emplacement

Le menu est maintenant disponible, mais rien ne s'affiche dans notre page.

Ne vous inquiétez pas, c'est normal! Nous devons maintenant lui dire où nous voulons que le menu s'affiche. Nous appelons ça une zone d'emplacement.

Pour déterminer une zone dans votre thème, vous devrez mettre le code suivant à l'endroit désiré dans votre gabarit :

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

# Houston! on a un menu

la structure par défaut du menu affiché sera :

```
<div class="menu-nom-menu-container">
  <ul id="menu-nom-menu" class="menu">
    <li id="menu-item-id-item" class="menu-item">
      <a href="lien">Texte</a>
    </li>
  </ul>
</div>
```

codex: référence  
wp\_nav\_menu()



# voici les différentes classes (css) qui sont générées par WordPress

- **.menu-nom-menu-container**: sur le conteneur du menu (<div> ou <nav>)
- **.menu**: sur le <ul> contenant le menu
- **.sub-menu**: sur le <ul> contenant les pages enfants
- **.menu-item**: sur chaque <li> du menu
- **.current-menu-item**: sur le <li> de la page courante
- **.current-menu-parent**: sur le <li> parent de la page courante (lorsque sous-nav)
- **.current-menu-ancestor**: sur les <li> ancêtres de la page courante (lorsque sous-sous-nav)
- **.menu-item-home**: sur les <li> de la page d'accueil



Lors de l'intégration html/css, prenez tout de suite en considération le nom des classes générées par WordPress.

Vous sauverez énormément de temps.

De toute façon, c'est une belle nomenclature de classes de menu, pourquoi ne pas toujours l'utiliser.

04

# **ajouter un widget**

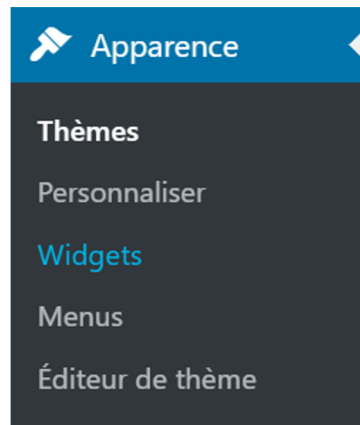
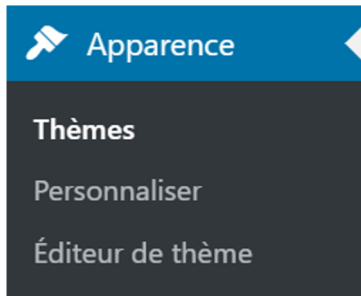
**sinon, à quoi sert ma barre latérale?**

## enregistrer des emplacements pour les widgets

- lorsque vous utilisez l'interface WordPress pour ajouter un **menu** ou un **widget**, vous indiquez au logiciel l'endroit où le menu/widget devrait s'afficher (ex.: sidebar, footer 1, menu\_principal, etc.)
- pour pouvoir permettre cela dans votre thème, vous devez **enregistrer des emplacements** prévus à cet effet et ajouter le code dans votre thème aux endroits désirés afin d'afficher le contenu de ces nouveaux emplacements
- **en gros, c'est pareil que pour les menus**

# enregistrer des emplacements pour les widgets

Par défaut, WordPress n'active pas de widget dans un thème personnalisé. La première étape consiste donc à activer les options de widget de WordPress dans `functions.php`



# 1. l'indiquer à WordPress

→ enregistrer un emplacement de widget

```
function creer_widgets() {  
    register_sidebar(array(  
        'name' => 'Ma nouvelle zone de widget',  
        'id' => 'zone-widget',  
        'before_widget' => '<div class="zone-widget">',  
        'after_widget' => '</div>',  
        'before_title' => '<h2 class="zone-widget-title">',  
        'after_title' => '</h2>',  
    ));  
}  
  
add_action('widgets_init', 'creer_widgets');
```



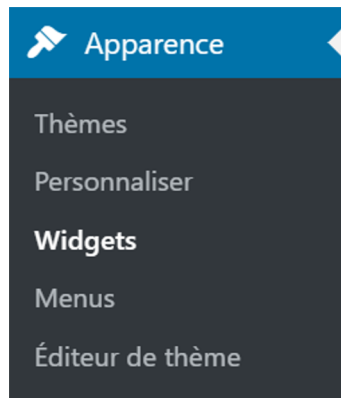
Vous pouvez utiliser plus d'une fois `register_sidebar()` afin d'avoir plusieurs emplacements de widget



```
register_sidebar(array(  
    // nom de l'emplacement qui apparaîtra dans l'interface administrateur  
    'name' => 'Ma nouvelle zone de widget',  
  
    // identifiant unique de l'emplacement widget  
    'id' => 'zone-widget',  
  
    // balise HTML qui ouvrira votre emplacement, ajoutez-y une classe qui vous sera utile pour le styler  
    'before_widget' => '<div class="zone-widget">',  
  
    // fermer la balise  
    'after_widget' => '</div>',  
  
    // balise pour le titre du widget (<h2>, <h3>, <h4>), ajoutez-y une classe qui vous sera utile pour le styler  
    'before_title' => '<h2 class="zone-widget-title">',  
  
    // fermer la balise du titre  
    'after_title' => '</h2>',  
));  
}
```

## 2. création et assignation d'un widget

il faut ensuite aller créer notre premier widget dans l'interface d'administration

Two screenshots of the WordPress widget configuration interface. The top screenshot shows the 'Articles récents' (Recent Articles) widget configuration. It includes a title field with 'Articles récents', a 'Nombre d'articles à afficher' (Number of articles to display) field set to '3', a checked checkbox for 'Afficher la date de la publication ?' (Show the publication date?), a 'Supprimer' (Delete) link, and an 'Enregistrer' (Save) button. The bottom screenshot shows the 'Catégories' (Categories) widget configuration. It includes a title field with 'Mots clés' (Keywords), three checkboxes: 'Afficher comme liste déroulante' (Show as dropdown menu) which is unchecked, 'Afficher le nombre d'articles' (Show the number of articles) which is checked, and 'Afficher la hiérarchie' (Show hierarchy) which is unchecked. It also features a 'Supprimer' (Delete) link and an 'Enregistrer' (Save) button.

### 3. sélection d'un emplacement

Le menu widget est maintenant disponible, mais rien ne s'affiche dans notre page.

Ne vous inquiétez pas, c'est normal! Nous devons maintenant lui indiquer l'endroit où les widgets devraient s'afficher. Nous appelons ça une **zone d'emplacement**.

### 3. sélection d'un emplacement (bis)

Pour déterminer une zone dans votre thème, vous devrez mettre le code suivant à l'endroit désiré dans votre gabarit :

```
<?php if (is_active_sidebar('zone-widget')) { ?>
    <?php dynamic_sidebar('zone-widget'); ?>
<?php } ?>
```

Petit conseil pour plus facilement ajuster le CSS : ajoutez un div pour englober la zone de widget.

```
<?php if (is_active_sidebar('zone-widget')) { ?>
    <div class="widget_header">
        <?php dynamic_sidebar('zone-widget'); ?>
    </div>
<?php } ?>
```



**la fin**

exercice s05 (suite)

# **créer son thème personnalisé**

➡ instructions sur le site du cours dans semaine06 ⬅