

Wordpress

CMS

-Anthony PARIS- Formateur M2I-

Divi builder

Après avoir effectué votre paramétrage global (celui de votre choix) voyons comment modifier notre page d'accueil. Rendez vous sur pages/accueil ou directement sur l'url de la page d'accueil et cliquez sur «activer le visual builder»

The screenshot shows a WordPress dashboard with a Divi theme active. The top navigation bar includes icons for WYSIWYG editor, course management, creation, modification, and exit. The main content area features a large blue header with the word "divi". Below the header is a toolbar with various icons. The main content area contains two text blocks. The first block says: "Ceci est une page d'exemple. C'est différent d'un article de blog parce qu'elle restera au même endroit et apparaîtra dans la navigation de votre site (dans la plupart des thèmes). La plupart des gens commencent par une page « À propos » qui les présente aux personnes visitant le site. Cela pourrait ressembler à quelque chose comme cela :". The second block contains a quote: "Bonjour ! Je suis un mécanicien qui aspire à devenir acteur, et voici mon site. J'habite à Bordeaux, j'ai un super chien baptisé Russell, et j'aime la vodka (ainsi qu'être surpris par la pluie soudaine lors de longues balades sur la plage au coucher du soleil)." Below these blocks is a section titled "...ou quelque chose comme cela :" containing another quote: "La société 123 Machin Truc a été créée en 1971, et n'a cessé de proposer au public des machins-trucs de qualité depuis lors. Située à Saint-Remy-en-Bouzemont-Saint-Genest-et-Isson, 123 Machin Truc emploie 2 000 personnes, et fabrique toutes sortes de bidules supers pour la communauté bouzemontoise.". The bottom of the screenshot has a footer with "Accueil", "Contact", and a search icon.

Vous vous retrouverez en temps réel en mode modification de votre page.

Vous ne pouvez pas modifier directement le header/footer. Pour cela soit passer par le paramétrage du personnaliseur de thème ou le thème builder du back office section divi

Divi builder

Une fois en mode édition vous remarquerez des blocs colorés :

- Bloc bleu : élément parent : Il prend toute la largeur de l'écran c'est un conteneur. La modification est utile pour insérer une image de fond ou une couleur.

- Bloc vert : A l'intérieur d'un bloc bleu, il permet de créer les colonnes que l'on peut définir en cliquant sur le bouton rectangle

- Bloc gris : A l'intérieur d'un bloc vert, il permet d'ajouter un élément (texte, vidéo, image,...)

Il peut y avoir d'autres blocs (orange : structure personnalisée, violet : Plein écran, vert : Dynamique).

Les blocs divi



Ceci est une page d'exemple. C'est différent d'un article de blog parce qu'elle restera au même endroit et apparaîtra dans la navigation de votre site (dans la plupart des thèmes). La plupart des gens commencent par une page « À propos » qui les présente aux personnes visitant le site. Cel



quelque chose comme cela :

Bonjour ! Je suis un mécanicien qui aspire à devenir acteur, et voici mon site. J'habite à Bordeaux, j'ai un super chien baptisé Russell, et j'aime la vodka (ainsi qu'être surpris par la pluie soudaine lors de longues balades sur la plage au coucher du soleil).

..ou quelque chose comme cela :

La société 123 Machin Truc a été créée en 1971, et n'a cessé de proposer au public des machins-trucs de qualité depuis lors. Située à Saint-Remy-en-Bouzemont-Saint-Genest-et-Isson, 123 Machin Truc emploie 2 000 personnes, et fabrique toutes sortes de bidules supers pour la communauté bouzemontoise.

En tant que nouvel utilisateur ou utilisatrice de WordPress, vous devriez vous rendre sur [votre tableau bord](#) pour supprimer cette page et créer de nouvelles pages pour votre contenu. Amusez-vous bien !



Bloc bleu : équivalent à un ROW de bootstrap (une ligne)

Bloc vert : équivalent à un COL de bootstrap (une cellule)

Bloc gris : Element : Texte, image, formulaire, composant bootstrap diverses (blog, réseaux sociaux, pagination, etc..)

Les options des blocs

Les blocs possèdent un système d'options en haut à gauche : (Chaque bloc a ce système d'options)



Par ordre :

- La flèche permet de déplacer un élément en drag and drop (glisser et déposer)
- La roue dentée permet de paramétriser plus en détails le bloc (fond, espacement, taille, police, scroll effect, transition au démarrage des pages etc)

A screenshot of the Divi WordPress plugin settings panel. At the top, it says "Section Paramètres" and "Preset: Défaut". Below that is a navigation bar with "Contenu" (selected), "Style", and "Avancé". Under "Contenu", there are dropdown menus for "Lien", "Fond", and "Étiquette Admin". At the bottom, there's a "Aidez-moi" link and a footer with navigation icons.

- Contenu : fond, lien, informations (peut varier en fonction du bloc)
- Style : Partie CSS gérée par DIVI : Taille, espacement, ombre, bordure, police etc...
- Avancé : Système de surcharge CSS en CSS et options spécifiques de positionnement / transition

Les options des blocs

- Le double rectangle permet de dupliquer un bloc avec ses enfants et sa configuration. Un clone parfait!



Copier la rangée

+ ⚙️ ⏪ ⏴ ⏵ ⏷ ⏸ :

Ceci est une page d'exemple. C'est différent d'un article de blog parce qu'elle restera au même endroit et apparaîtra dans la navigation de votre site (dans la plupart des thèmes). La plupart des gens commencent par une page « À propos » qui les présente aux personnes visitant le site. Cela pourrait ressembler à quelque chose comme cela :

Bonjour ! Je suis un mécanicien qui aspire à devenir acteur, et voici mon site. J'habite à Bordeaux, j'ai un super chien baptisé Russell, et j'aime la vodka (ainsi qu'être surpris par la pluie soudaine lors de longues balades sur la plage au coucher du soleil).

..ou quelque chose comme cela :

La société 123 Machin Truc a été créée en 1971, et n'a cessé de proposer au public des machins-trucs de qualité depuis lors. Située à Saint-Remy-en-Bouzemont-Saint-Genest-et-Isson, 123 Machin Truc emploie 2 000 personnes, et fabrique toutes sortes de bidules supers pour la communauté bouzemontoise.

En tant que nouvel utilisateur ou utilisatrice de WordPress, vous devriez vous rendre sur [votre tableau de bord](#) pour supprimer cette page et créer de nouvelles pages pour votre contenu. Amusez-vous bien !

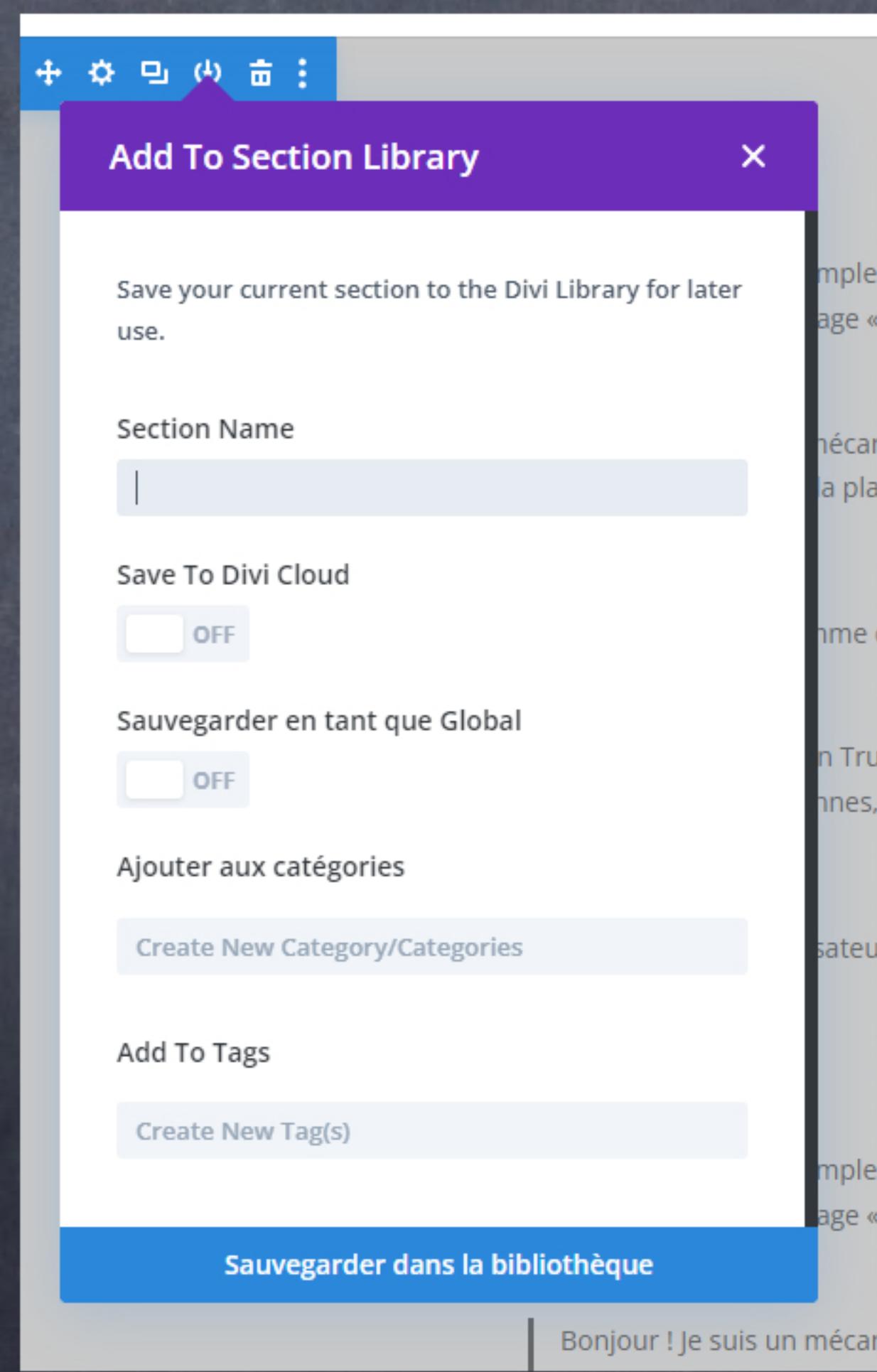
+

Ceci est une page d'exemple. C'est différent d'un article de blog parce qu'elle restera au même endroit et apparaîtra dans la navigation de votre site (dans la plupart des thèmes). La plupart des gens commencent par une page « À propos » qui les présente aux personnes visitant le site. Cela pourrait ressembler à quelque chose comme cela :

Bonjour ! Je suis un mécanicien qui aspire à devenir acteur, et voici mon site. J'habite à Bordeaux, j'ai un super chien baptisé Russell, et j'aime la vodka (ainsi qu'être surpris par la pluie soudaine lors de

Les options des blocs

- L'option suivante (la flèche dans le cercle) permet de sauvegarder un élément afin de pouvoir le ré-utiliser dans une autre page. L'option Global permet lors d'une modification de l'élément d'affecter l'ensemble des clones sur les autres pages. Vous retrouverez également l'élément dans la bibliothèque divi du back office



Les options des blocs

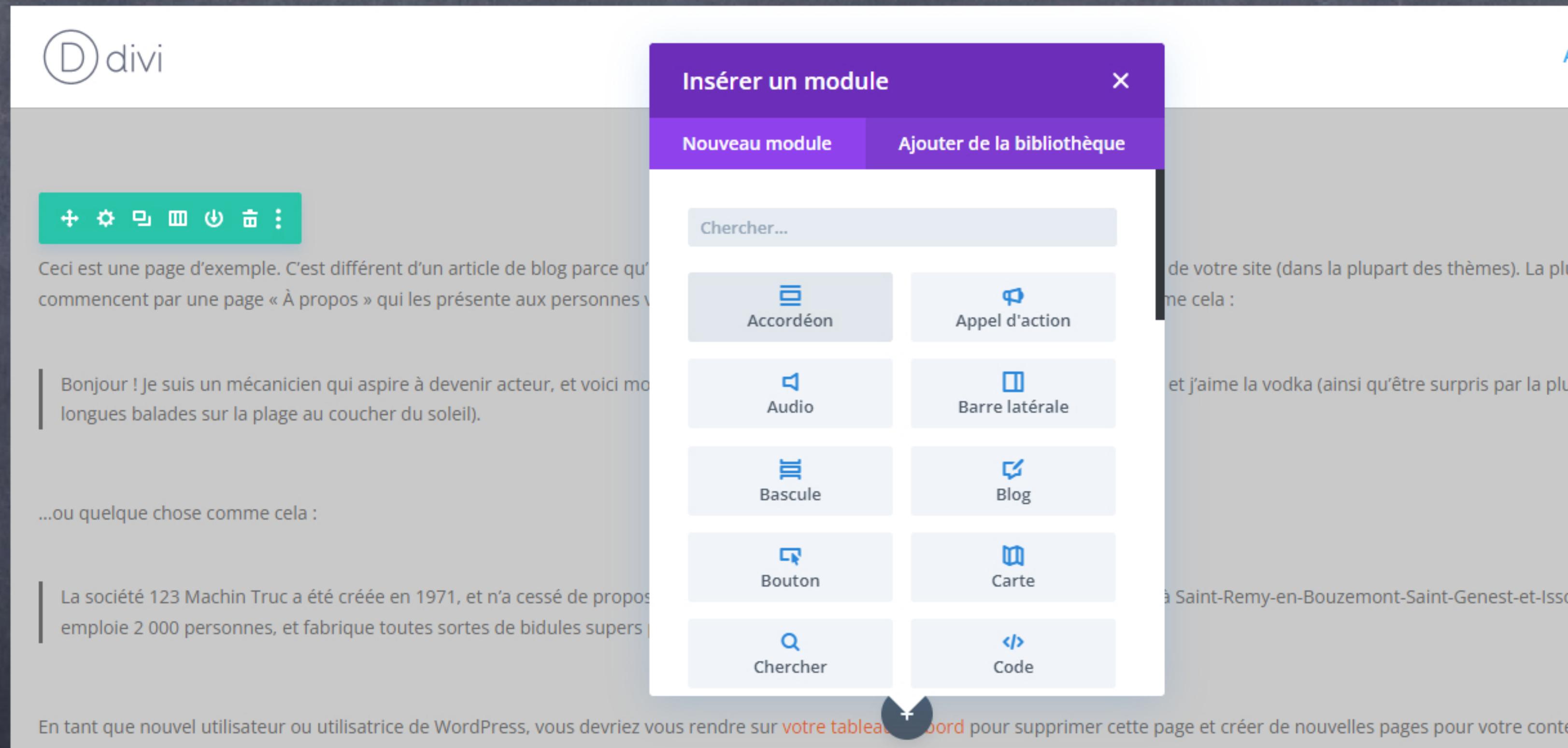
Sur le bloc vert vous trouverez une option (tableau) permettant de définir le colonnage d'une ligne.



- La corbeille permet de supprimer un élément et un CRTL + Z permet de revenir en arrière.
- Les ... permettent un accès rapide vers des options spécifiques

Les options des blocs

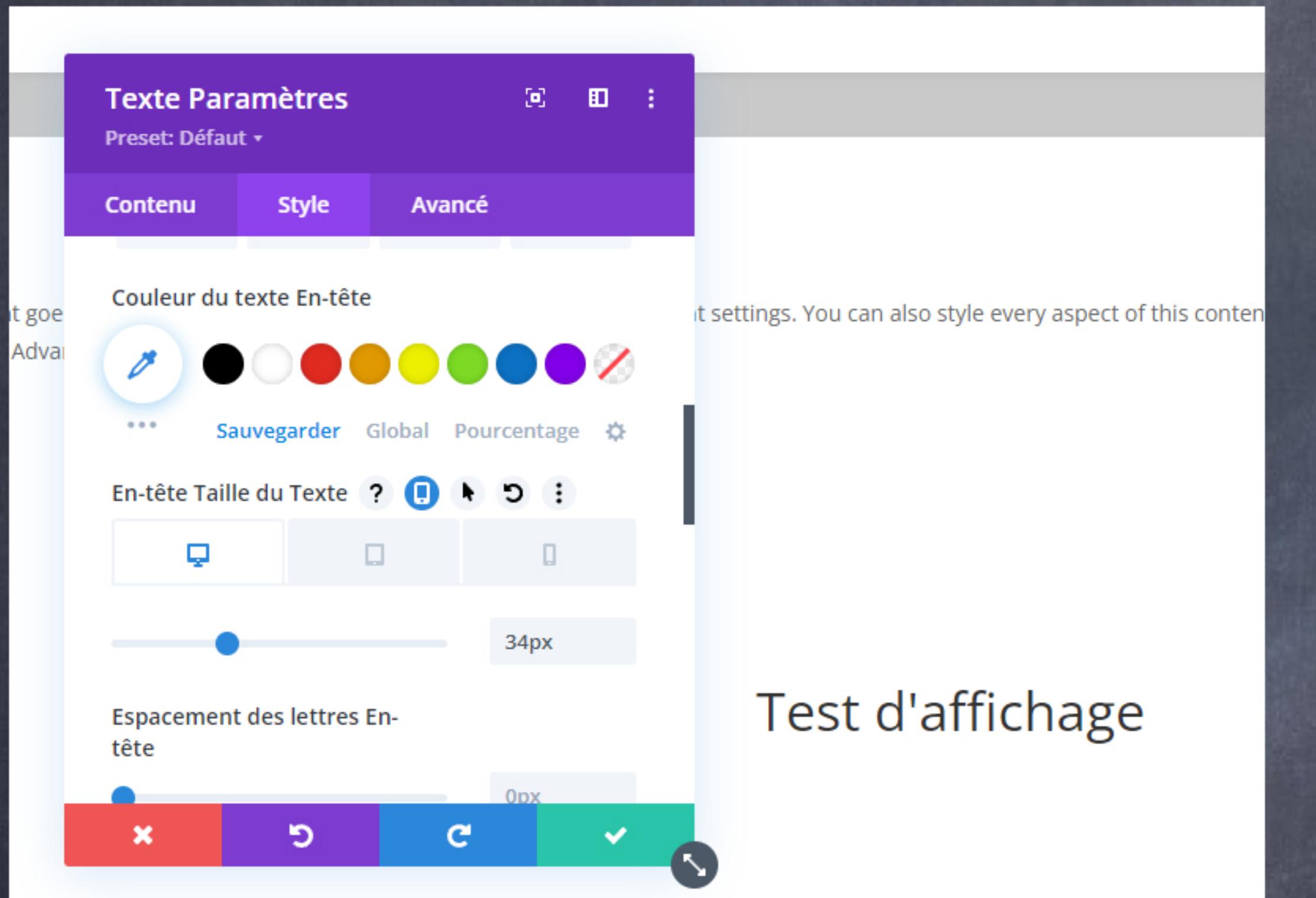
- le bouton + bleu permet de rajouter une bloc entier vide.
- le bouton + vert permet de rajouter une ligne entière vide.
- le bouton + gris permet de choisir un composant divi



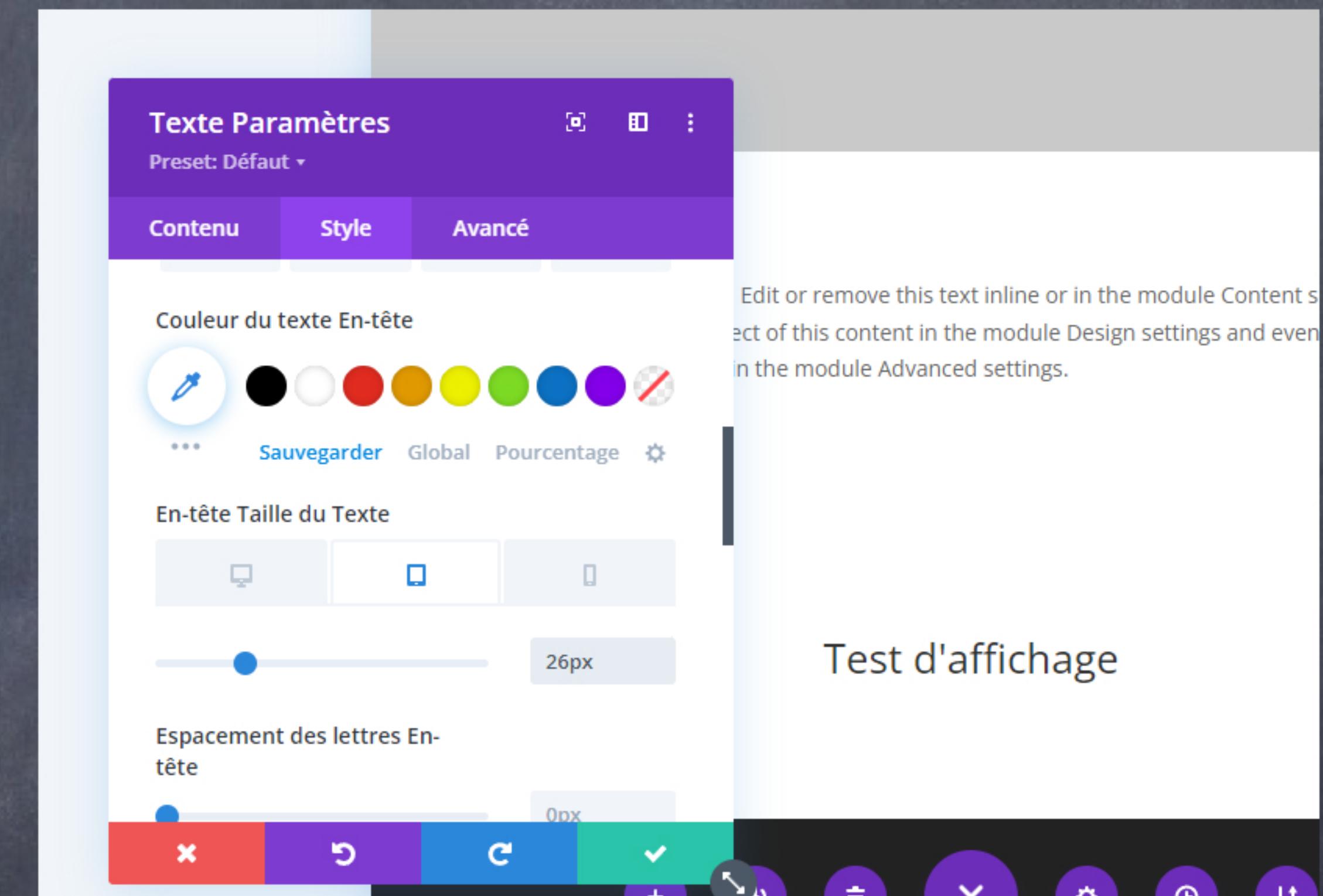
Essayez d'ajouter un formulaire, du texte ou une image. Vous pouvez même ajouter du code directement

Affichage

Divi possède un système de paramétrage en fonction des supports. Par exemple si j'ajoute un texte et que je souhaite changer la taille de la police en fonction des «devices» :



Test d'affichage



Test d'affichage

Sauvegarde et vérification

Il est toujours intéressant de visualiser son site sur l'ensemble des supports. Pour cela cliquez sur la boule violette en bas de l'écran

Device spécifique

Vue de la tablette 768px Make Default Tablet View

Test de support

Sauvegarde

Bibliothèque de thème et enregistrement

Vous pouvez également enregistrer votre template pour l'importer sur d'autres pages ou partir d'un thème existant

Charger à partir de la bibliothèque

Mises en pages prédefinies Télécharger le modèle Vos pages existantes

Charger une nouvelle mise en page

Modèles (Mise en page générale)

New To Old Sign In To Divi Cloud

Chercher + Filtre

Catégories

- Art & Design
- Affaires
- Community & Non-Profit
- Saturation
- Éléments
- Fashion & Beauty
- Food & Drink
- Health & Fitness
- Lifestyle
- Quitter L'Éditeur
- En Ligne
- Prestations De Service
- Simple

Divi Toy Store DESIGN YOUR SPACES Let's Find The Best Property for You

Toy Store Modèle Interior Designer Modèle Real Estate Agent Modèle

Handmade Leather Goods DISCOVER SAN FRANCISCO MAKE YOURSELF AT HOME

Leather Goods Modèle Hostel Modèle Home Remodeling Modèle

Bibliothèque de thèmes

Utilisez ce tableau de bord pour supprimer cette page et sauvegarder des nouvelles pages pour votre contenu.

+ ⌂ X Pressez les touches de votre clavier pour sauvegarder sur votre tableau de bord.

Sauvegarder son template

Exemple de template de base

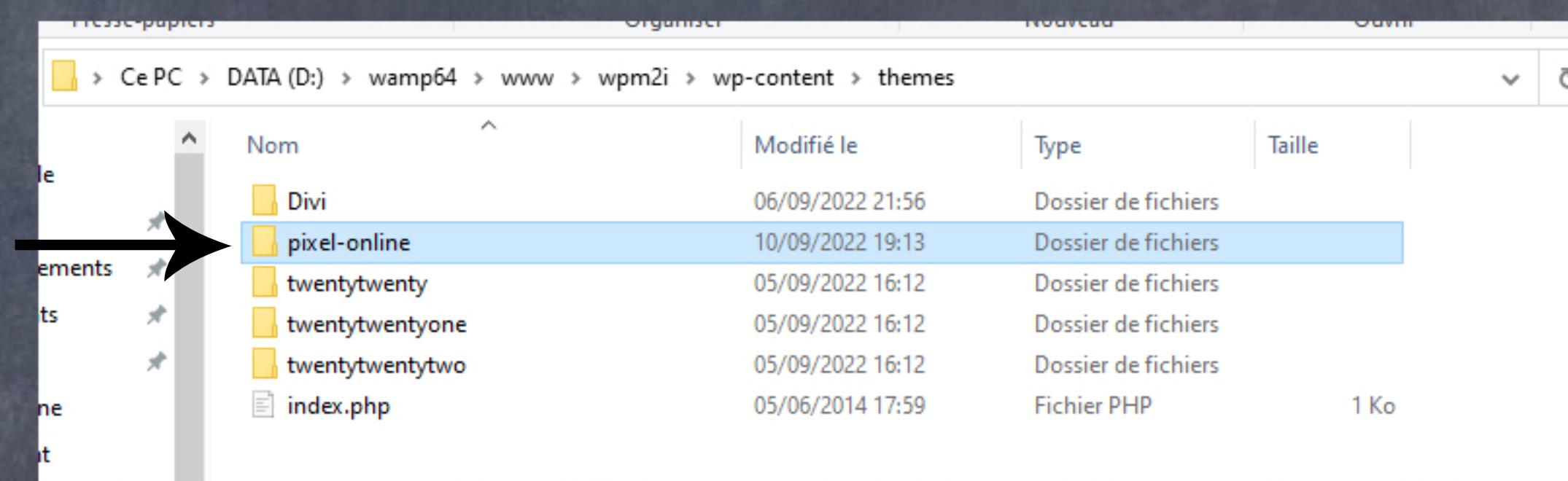
En partant d'un thème par défaut vous pouvez utiliser les blocs pour customiser votre template choisi.

The screenshot shows the Divi Visual Builder interface with a template for a "Divi Poke!" website. The header features a "divi" logo and navigation links for "Accueil" and "Contact". The main content area has a title "Divi Poke!" and a short text blurb. A central image of a poke bowl is surrounded by other images and a call-to-action button labeled "ONLINE MENU". The interface includes various editing tools and a preview mode at the bottom.

Création thème enfant

La création d'un thème enfant permet de ne pas perdre des modifications du code du thème lors des mises à jour. Par exemple je souhaiterai ajouter le code pour avoir un fil d'ariane sur contact et éviter sa suppression lors de la mise à jour de divi.

Dans le dossier thème, créez un nouveau dossier du nom de votre choix



Créez ensuite 3 fichiers :

- Functions.php pour informer WP qu'il s'agit d'un thème enfant
- Style.css votre surcharge css
- screenshot.jpg l'image de votre thème

	functions.php	10/09/2022 19:14	Fichier PHP
	screenshot.jpg	20/08/2020 19:01	Fichier JPG
	style.css	10/09/2022 19:14	Fichier CSS

Création thème enfant

style.css :

```
wp-content > themes > pixel-online > # style.css > ...
1  /*
2   Theme Name:      Pixel Online Création
3   Description:    Thème enfant de DIVI
4   Author:         Anthony PARIS
5   Author URI:    https://www.pixel-online.fr
6   Template:       Divi
7   Version:        1.0.0
8 */
9
10 /* ANIMATION */
11 @keyframes pulse {
12   0% {
13     transform: scale(0.95);
14     box-shadow: 0 0 0 0 rgba(255, 255, 255, 0.7);
15   }
16
17   70% {
18     transform: scale(1);
19     box-shadow: 0 0 0 10px rgba(0, 0, 0, 0);
20   }
21
22   100% {
23     transform: scale(0.95);
24     box-shadow: 0 0 0 0 rgba(0, 0, 0, 0);
25   }
26 }
27 .pulse {
28   animation: pulse 2s infinite;
29 }
30
31
32 |
```

Les commentaires en haut sont obligatoires, ils permettent à

WP de connaître le thème parent et votre thème enfant.

Theme Name : Le nom que je veux donner à mon thème enfant

Description : La description de mon thème enfant celle qui ap-

paraîtra dans mon gestionnaire de thème WordPress

Author : L'auteur du thème enfant, en l'occurrence c'est vous

Author URI : L'url du site de l'auteur parce qu'un peu de pub

ne fait pas de mal

Template : Le nom du thème parent en l'occurrence le nom du

répertoire tel qu'il est écrit sur le FTP

Version : La version de votre thème enfant à titre indicatif

info : <https://wpformation.com/theme-enfant-wordpress/>

Création thème enfant

functions.php

```
wp-content > themes > pixel-online > functions.php > PHP Intelephense > theme_enqueue_styles
1  <?php
2  /**
3   * activation theme
4   */
5  add_action( 'wp_enqueue_scripts', 'theme_enqueue_styles' );
6  function theme_enqueue_styles() {
7      wp_enqueue_style( 'parent-style', get_template_directory_uri() . '/style.css' );
8      wp_enqueue_style( 'load-fa', 'https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css' );
9  }
10 }
```

Ce fichier permet de modifier ou d'ajouter des éléments par rapport au thème enfant.

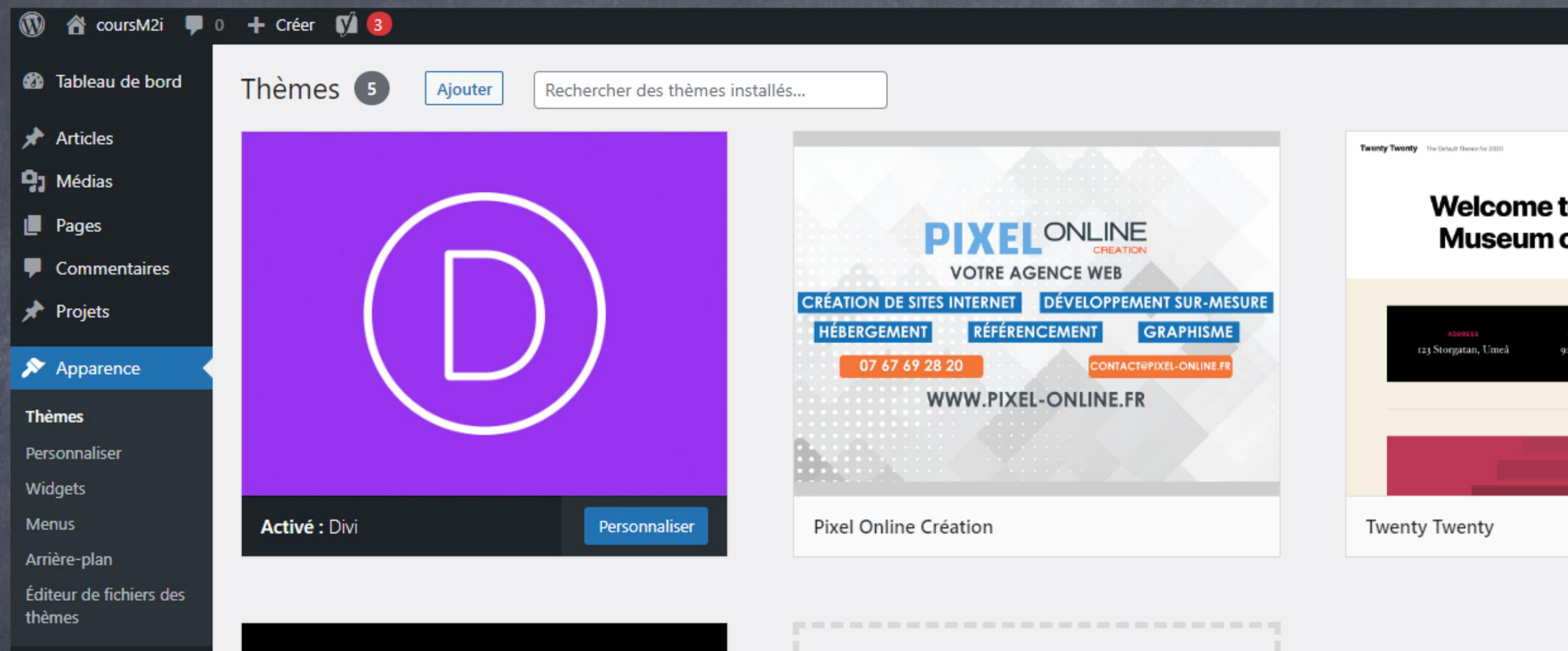
Ici nous souhaitons reprendre le CSS de divi en plus de notre fichier css de notre thème enfant et ajoutez

la prise en charge de font-awesome.

Nous reviendrons sur les hooks (add_actions / add_filters)

Création thème enfant

Revenez sur le back office dans apparence/thèmes et activez votre thème enfant

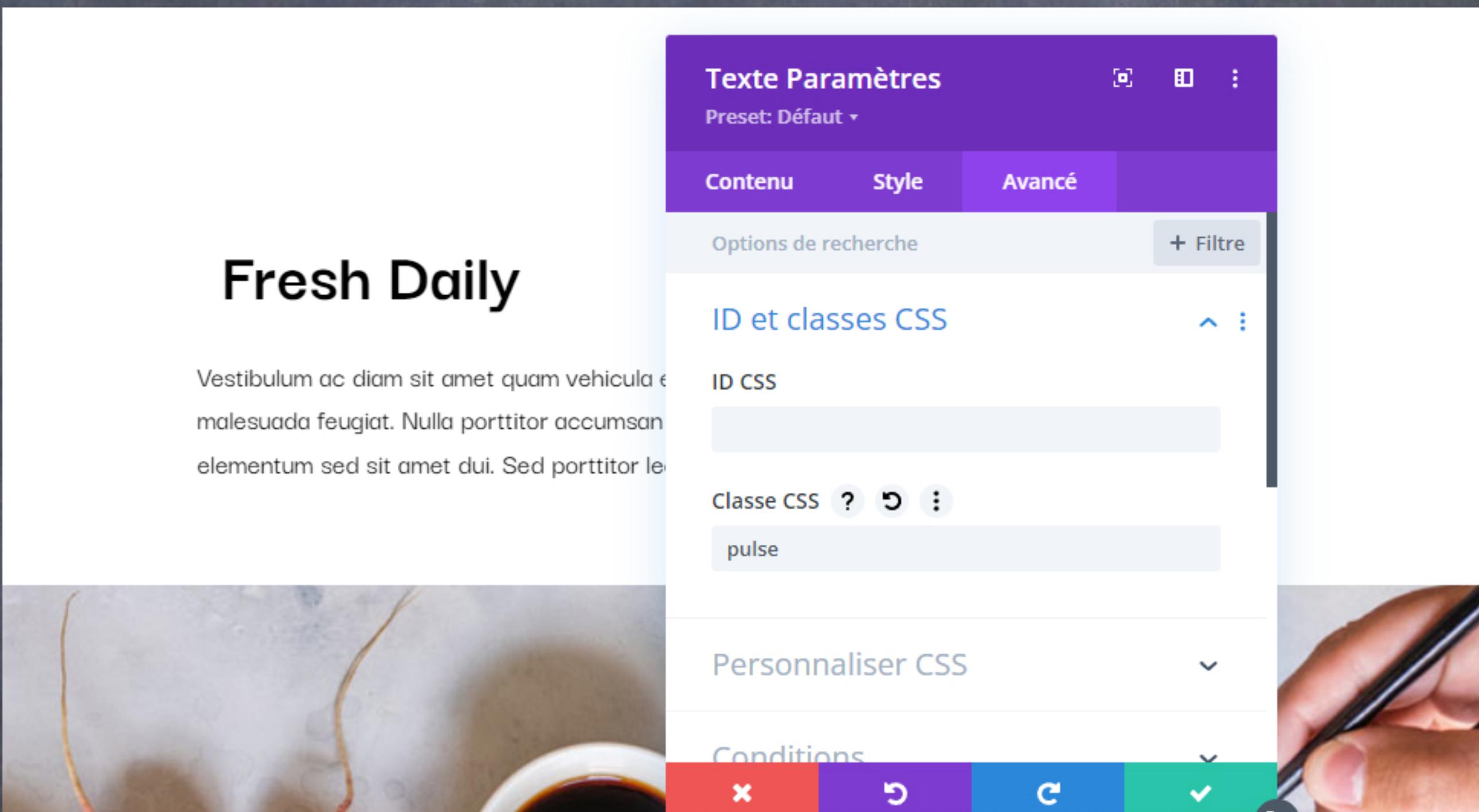


Visualisez la source de votre homepage, nous avons divi et en plus notre propre CSS!

```
.et_pb_text{word-wrap:break-word}.et_pb_text ol,.et_pb_text ul{padding-bottom:1em}.et_pb_text>:last-child{padding-bottom:0}.et_pb_text_inner{position:relative}
.et_pb_bg_layout_light.et_pb_module.et_pb_button{color:#2ea3f2}.et_pb_module.et_pb_button{display:inline-block;color:inherit}.et_pb_button_module_wrapper.et_pb_button
.et_pb_button[data-icon]:not([data-icon=""]):after{content:attr(data-icon)}@media (max-width:980px){.et_pb_button[data-icon-tablet]:not([data-icon-tablet=""]):after{c
</style>
<link rel='preload' id='divi-dynamic-css' href='http://localhost/wpm2i/wp-content/et-cache/2/et-divi-dynamic-2.css?ver=1662831929' as='style' media='all' onload="this.rel='stylesheet'" type='text/css' />
<link rel='stylesheet' id='yoast-seo-adminbar-css' href='http://localhost/wpm2i/wp-content/plugins/wordpress-seo/css/dist/adminbar-1961.css' type='text/css' media='all' />
<link rel='stylesheet' id='divi-style-css' href='http://localhost/wpm2i/wp-content/themes/pixel-online/style.css?ver=4.18.0' type='text/css' media='all' />
<link rel='https://api.w.org/' href='http://localhost/wpm2i/wp-json/' /><link rel="alternate" type="application/json" href='http://localhost/wpm2i/wp-json/wp/v2/pages' />
<link rel='wlwmanifest' type="application/wlwmanifest+xml" href='http://localhost/wpm2i/wp-includes/wlwmanifest.xml' />
<meta name="generator" content="WordPress 6.0.2" />
<link rel='shortlink' href='http://localhost/wpm2i/' />
<link rel="alternate" type="application/json+oembed" href='http://localhost/wpm2i/wp-json/oembed/1.0/embed?url=http%3A%2F%2Flocalhost%2Fwpm2i%2F' />
<link rel="alternate" type="text/xml+oembed" href='http://localhost/wpm2i/wp-json/oembed/1.0/embed?url=http%3A%2F%2Flocalhost%2Fwpm2i%2F&format=xml' />
```

Création thème enfant

Testons le CSS ajouté dans notre thème enfant.



Essayons d'ajouter le fil d'ariane sur contact :

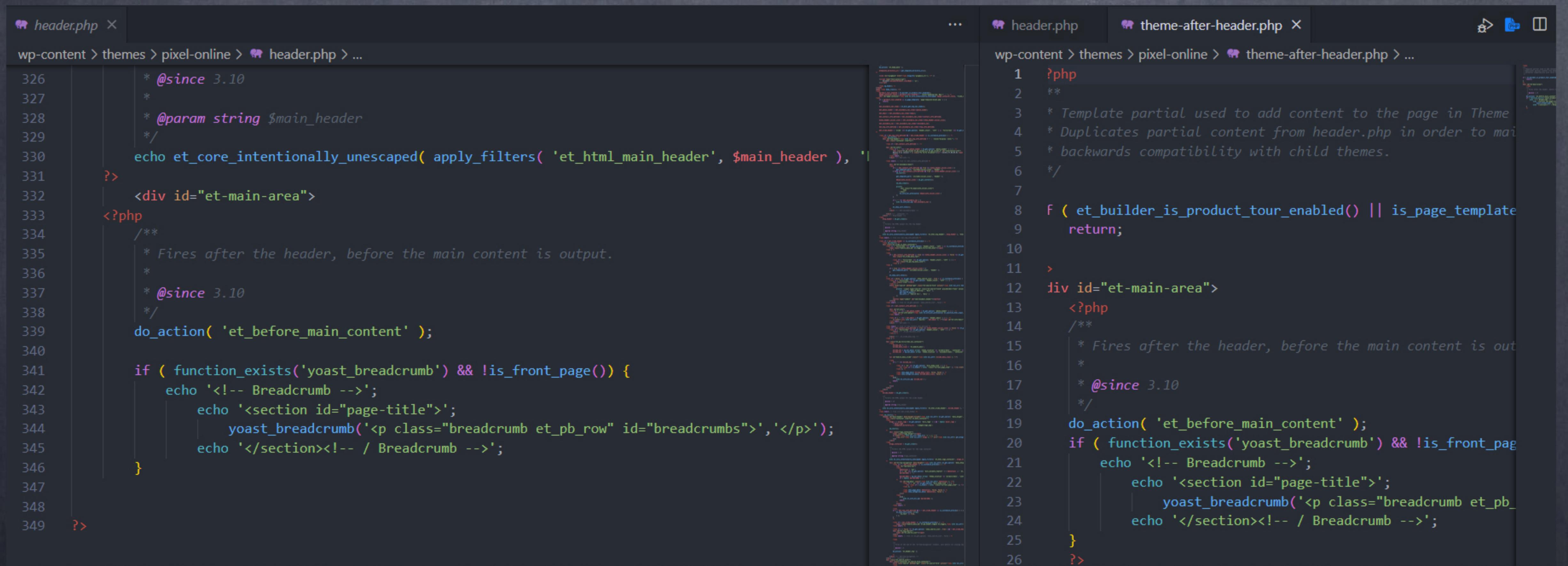
Pour cela nous devons ajoutez un code dans la page header si on utilise le menu simple ou thème-af-
ter-header si on utilise le thème builder du back office.

Dans le doute, nous allons l'ajouter dans les deux fichiers

Création thème enfant

Dupliquez header.php et theme-after-header.php de divi vers votre thème enfant pour surcharger ses fichiers.

L'avantage du thème enfant montre que quand le template divi sera mis à jour nos fichiers de thème enfant ne le seront pas, donc aucun écrasement de code!



The image shows a code editor with two tabs open: 'header.php' and 'theme-after-header.php'. Both files are located in the 'wp-content > themes > pixel-online' directory.

header.php:

```
wp-content > themes > pixel-online > header.php > ...
326     * @since 3.10
327     *
328     * @param string $main_header
329     */
330     echo et_core_intentionally_unescaped( apply_filters( 'et_html_main_header', $main_header ), 'HTML' );
331 ?>
332     <div id="et-main-area">
333 <?php
334     /**
335      * Fires after the header, before the main content is output.
336      *
337      * @since 3.10
338      */
339     do_action( 'et_before_main_content' );
340
341     if ( function_exists('yoast_breadcrumb') && !is_front_page() ) {
342         echo '<!-- Breadcrumb -->';
343         echo '<section id="page-title">';
344         yoast_breadcrumb('<p class="breadcrumb et_pb_row" id="breadcrumbs">', '</p>');
345         echo '</section><!-- / Breadcrumb -->';
346     }
347
348 ?>
```

theme-after-header.php:

```
wp-content > themes > pixel-online > theme-after-header.php > ...
1 ?php
2 /**
3  * Template partial used to add content to the page in Theme
4  * Duplicates partial content from header.php in order to maintain
5  * backwards compatibility with child themes.
6 */
7
8 F( et_builder_is_product_tour_enabled() || is_page_template()
9     return;
10
11 >
12 <div id="et-main-area">
13 <?php
14 /**
15  * Fires after the header, before the main content is output.
16  *
17  * @since 3.10
18  */
19 do_action( 'et_before_main_content' );
20 if ( function_exists('yoast_breadcrumb') && !is_front_page() ) {
21     echo '<!-- Breadcrumb -->';
22     echo '<section id="page-title">';
23     yoast_breadcrumb('<p class="breadcrumb et_pb_row" id="breadcrumbs">', '</p>');
24     echo '</section><!-- / Breadcrumb -->';
25 }
26 ?>
```

Création thème enfant

Maintenant sur la page contact :

The screenshot shows a web browser window with a dark grey header bar. The address bar displays 'localhost/wpm2i/contact/'. Below the address bar is a horizontal menu bar with various icons and text labels: SYMFONY, JAVASCRIPT, JQUERY, NODEJS, HTML, TWIG, CSS, POSTGRESQL, MYSQL, SOLR, CMS, JOOMLA, WORDPRESS, PRESTASHOP, VPS, GIT, SEO, GOOGLE, RESEAUX SOCIAUX, GRAPHISME, and Autres favoris. On the far right of the menu bar, there are several small icons, including a gear, a star, and a search icon. Below the menu bar, the browser's toolbar includes icons for back, forward, home, and search, along with a refresh button. The main content area of the browser shows a website for 'divi'. The header of the website has a logo 'D divi' on the left and navigation links for 'Accueil' and 'Contact' on the right. A magnifying glass icon is also present in the header. The breadcrumb navigation at the top of the page reads 'Accueil » Contact'. The main content of the page is a large, handwritten-style text message.

Essayez de rajouter du css dans style.css pour modifier l'affichage, vous ne pouvez pas le modifier avec le builder divi car il s'agit d'un code non géré par divi.

Hooks

Sur le fichier functions.php nous avons une fonction add_action(). Il s'agit d'un Hook.

Les hooks WP vous permettent de vous «accrocher» au processus de construction de page et vous donnent plus de contrôle sur ce que vous créez.

Il existe deux types de hooks : Actions et Filtres.

```
function add_action( $tag, $function_to_add, $priority = 10, $accepted_args = 1 ) {  
    return add_filter( $tag, $function_to_add, $priority, $accepted_args );  
}
```

La fonction add_action() appelle simplement la fonction add_filter() et retourne sa valeur. Pourquoi ?

Parce qu'elles fonctionnent toutes deux fondamentalement de la même manière, à une différence près.

La fonction apply_filters() retourne une valeur qui peut modifier les types de données existants, tandis que la fonction do_action() ne renvoie rien (valeur NULL en PHP).

Hooks

Hooks

- Actions

```
do_action('action_name')
```

- Filters

```
apply_filters('filter_name', $to_be_filtered_value)
```

→ Hook Functions

- Add Action Function

```
add_action('action_name', 'some_callback_function')
```

- Add Filter Function

```
add_filter('filter_name', 'another_callback_function')
```

→ Callback Functions

- Action Callback Function

```
function some_callback_function() {
    // just do something here, nothing to be returned
}
```

- Filter Callback Function

```
function another_callback_function($to_be_filtered_value) {
    // filter the value here, and then return it
    return $filtered_value
}
```

Actions	Filters
Les actions sont utilisées pour exécuter des fonctions personnalisées à un moment précis de l'exécution du cœur de WordPress.	Les filtres sont utilisés pour modifier ou personnaliser les données utilisées par d'autres fonctions.
Les actions sont définies / créées par la fonction <code>do_action('action_name')</code> dans le code de WordPress.	Les filtres sont définis / créés par la fonction <code>apply_filters('filter_name', 'value_to_be_filtered')</code> dans le code de WordPress.
Les actions sont également appelées Action hooks .	Les filtres sont également appelés Filter hooks .
Les actions ne peuvent être connectées qu'à des fonctions d'action. Par exemple, <code>add_action()</code> , <code>remove_action()</code> .	Les filtres ne peuvent être accrochés qu'avec des fonctions de filtre. Par exemple, <code>add_filter()</code> , <code>remove_filter()</code> .
Les fonctions d'action n'ont pas besoin de faire passer d'arguments à leurs fonctions de rappel.	Les fonctions de filtre doivent transmettre au moins un argument à leurs fonctions de rappel.
Les fonctions d'action peuvent effectuer tout type de tâche, y compris modifier le comportement du fonctionnement de WordPress.	Les fonctions de filtre n'existent que pour modifier les données qui leur sont transmises par les filtres.
Les fonctions d'action ne doivent rien <code>return</code> . Cependant, elles peuvent faire <code>echo</code> à la sortie ou interagir avec la base de données.	Les fonctions de filtrage doivent <code>return</code> leurs modifications en sortie. Même si une fonction de filtre ne change rien, elle doit toujours <code>return</code> l'entrée non modifiée.
Les actions peuvent exécuter presque tout, tant que le code est valide.	Les filtres doivent fonctionner de manière isolée, afin de ne pas avoir d'effets secondaires involontaires.
Résumé : une action interrompt le processus normal d'exécution du code pour faire quelque chose avec les informations qu'elle reçoit, mais ne retourne rien en retour, puis sort.	Résumé : un filtre modifie l'information qu'il reçoit, la renvoie à la fonction d'appel, et d'autres fonctions peuvent utiliser la valeur qu'il retourne.

Hooks

Exemple de hook action et filtre

```
// define the callback function to modify the text
function change_text_another_callback( $content ) {
    // add the code to change text here and then return it
    return $filtered_content;
}

// hook in to 'the_content' filter with the add_filter() function
add_filter( 'the_content', 'change_text_another_callback');
```

```
// define the callback function to change the text
function change_text_callback() {
    // add the code to change text here
}

// hook in to the 'publish_post' action with the add_action() function
add_action( 'publish_post', 'change_text_callback' );
```

Hooks

Vous pouvez enregistrer une fonction de rappel avec une action en suivant ces étapes :

Définissez une fonction de rappel avec votre code personnalisé à l'intérieur. Cette fonction de rappel sera exécutée lorsque toute action à laquelle elle est enregistrée est déclenchée pendant l'exécution du code de WordPress.

Accrochez votre fonction de rappel à l'action que vous voulez avec la fonction `add_action()`. Comme le prévoit le Codex WordPress, la fonction `add_action()` doit passer au moins deux paramètres :

Nom de l'action à laquelle s'accrocher.

Nom de la fonction de rappel qui sera exécutée lorsque l'action est déclenchée.

La fonction `add_action()` accepte également deux paramètres optionnels pour fixer la priority et le number of arguments. Nous discuterons de ces deux paramètres plus tard.

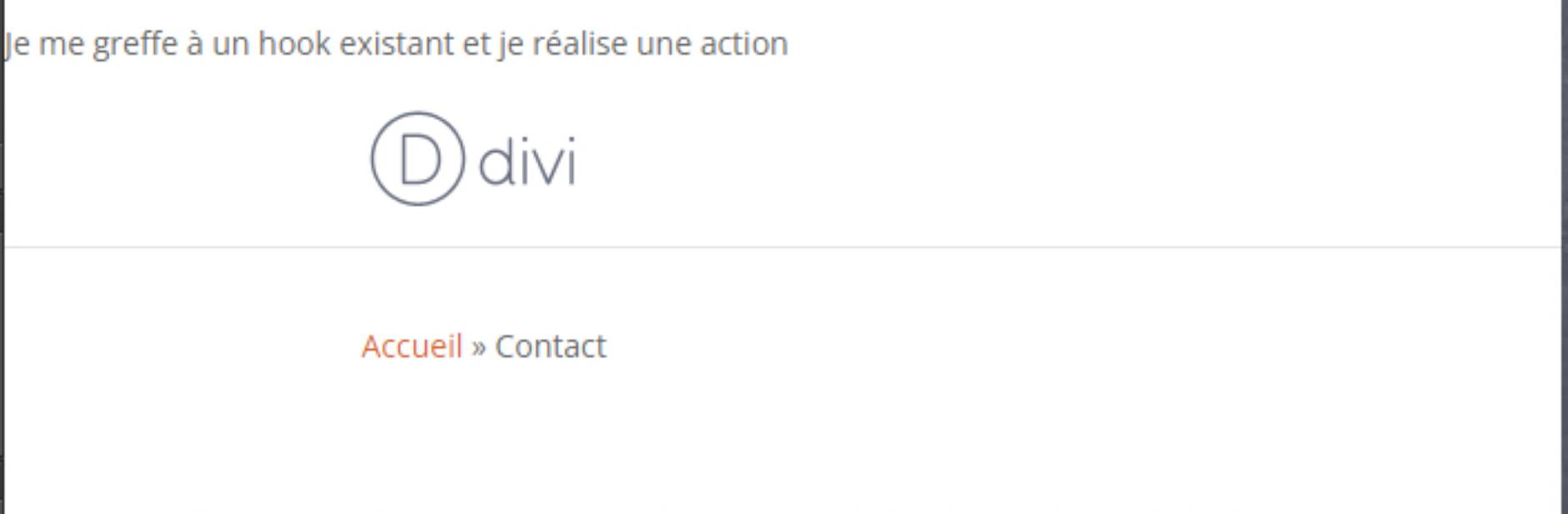
Il est bon de nommer vos paramètres de fonction de rappel aussi près que possible des paramètres passés par la fonction du hook.

Hooks

Dans votre fichier functions.php essayez :

```
<?php
/**
** activation theme
*/
add_action( 'wp_enqueue_scripts', 'theme_enqueue_styles' );
function theme_enqueue_styles() {
wp_enqueue_style( 'parent-style', get_template_directory_uri() . '/style.css' );
wp_enqueue_style( 'load-fa', 'https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/
}

function custom_callback_function(){
    // add your custom code here to do something
    echo 'Je me greffe à un hook existant et je réalise une action';
}
add_action( 'init', 'custom_callback_function' );
```



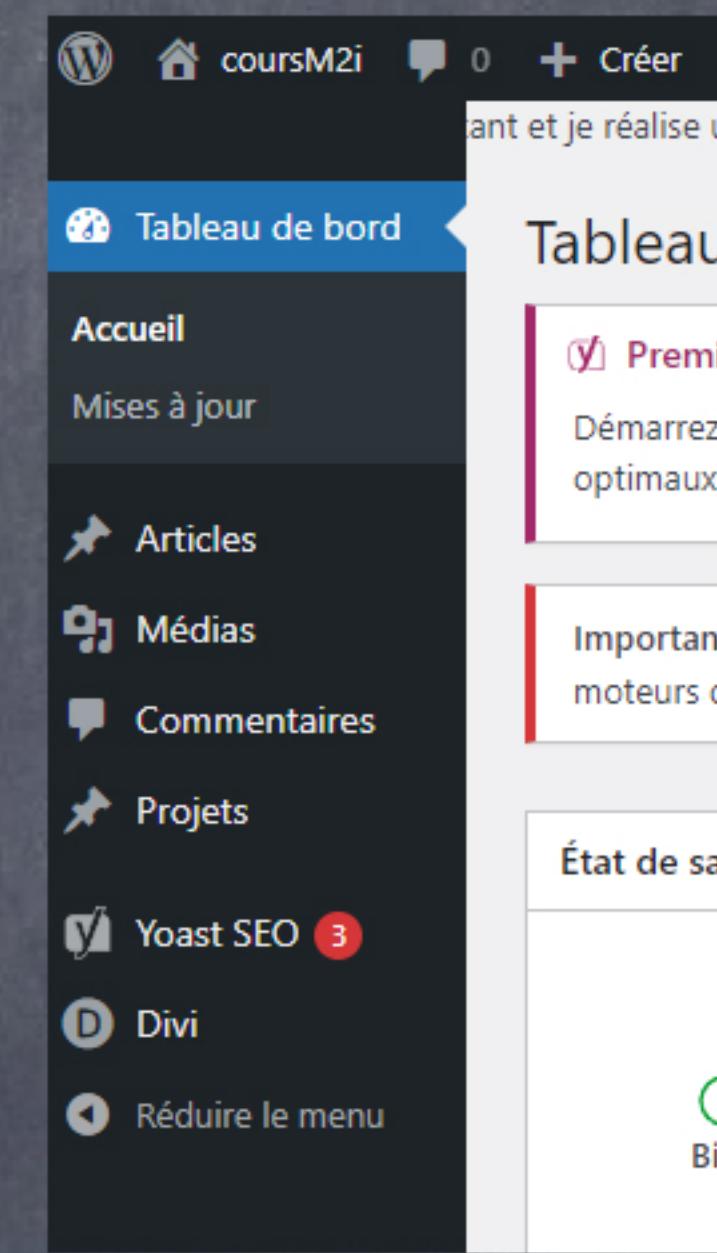
Voici la liste des hook natifs disponible :

https://codex.wordpress.org/Plugin_API/Action_Reference

Hooks

Autre exemple, masquer des éléments du menu :

```
add_action( 'admin_menu', 'hide_admin_menus' );
function hide_admin_menus() {
    // if (current_user_can( 'create_users' ) ) return;
    // if (wp_get_current_user()->display_name == "Anthony") return;
    remove_menu_page( 'plugins.php' );
    remove_menu_page( 'themes.php' );
    remove_menu_page( 'tools.php' );
    remove_menu_page( 'users.php' );
    remove_menu_page( 'edit.php?post_type=page' );
    remove_menu_page( 'options-general.php' );
}
```



`admin_menu` est une action qui se déclenche avant le chargement du menu d'administration dans la zone du tableau de bord de WordPress.

Accrochez-vous à l'action `admin_menu` à l'aide de la fonction `add_action()` en utilisant la fonction de rappel `hide_admin_menus()`.

La fonction de rappel `hide_admin_menus()` définit la logique du code. Elle est exécutée à chaque fois que se déclenche l'action `admin_menu`.

Dans la fonction de rappel, la fonction `current_user_can('create_users')` vérifie si l'utilisateur connecté est un administrateur. Étant donné que seuls les administrateurs de site ont la capacité `create_user`, la fonction se termine par une déclaration `return` si l'utilisateur est un administrateur.

La fonction WordPress `wp_get_current_user()` récupère l'objet `current user`. Avec cette fonction, nous pouvons vérifier si l'utilisateur connecté a un `display_name` particulier. Cette ligne est facultative, au cas où vous souhaiteriez éviter que certains utilisateurs non administrateurs soient bloqués en raison de cette fonction de rappel.

La fonction `remove_menu_page()` de WordPress supprime les menus d'administration de haut niveau. Dans l'exemple de code ci-dessus, je supprime les menus d'administration suivants : Extensions, Thèmes, Outils, Utilisateurs, Pages et Options.

Hooks

Les filtres :

Vous pouvez accrocher une fonction de rappel à un filtre en suivant les étapes suivantes :

Définissez une fonction de rappel qui sera exécutée lorsque WordPress lancera le filtre. Les fonctions de rappel pour les filtres doivent avoir au moins un argument spécifié, car tous les filtres transmettent au moins une valeur à leurs fonctions de rappel.

Enregistrez la fonction de rappel à un filtre avec la fonction `add_filter()`. Le filtre se chargera d'appeler la fonction de rappel. Selon le Codex WordPress, la fonction `add_filter()` doit passer au moins deux paramètres :

Le nom du filtre auquel s'accrocher.

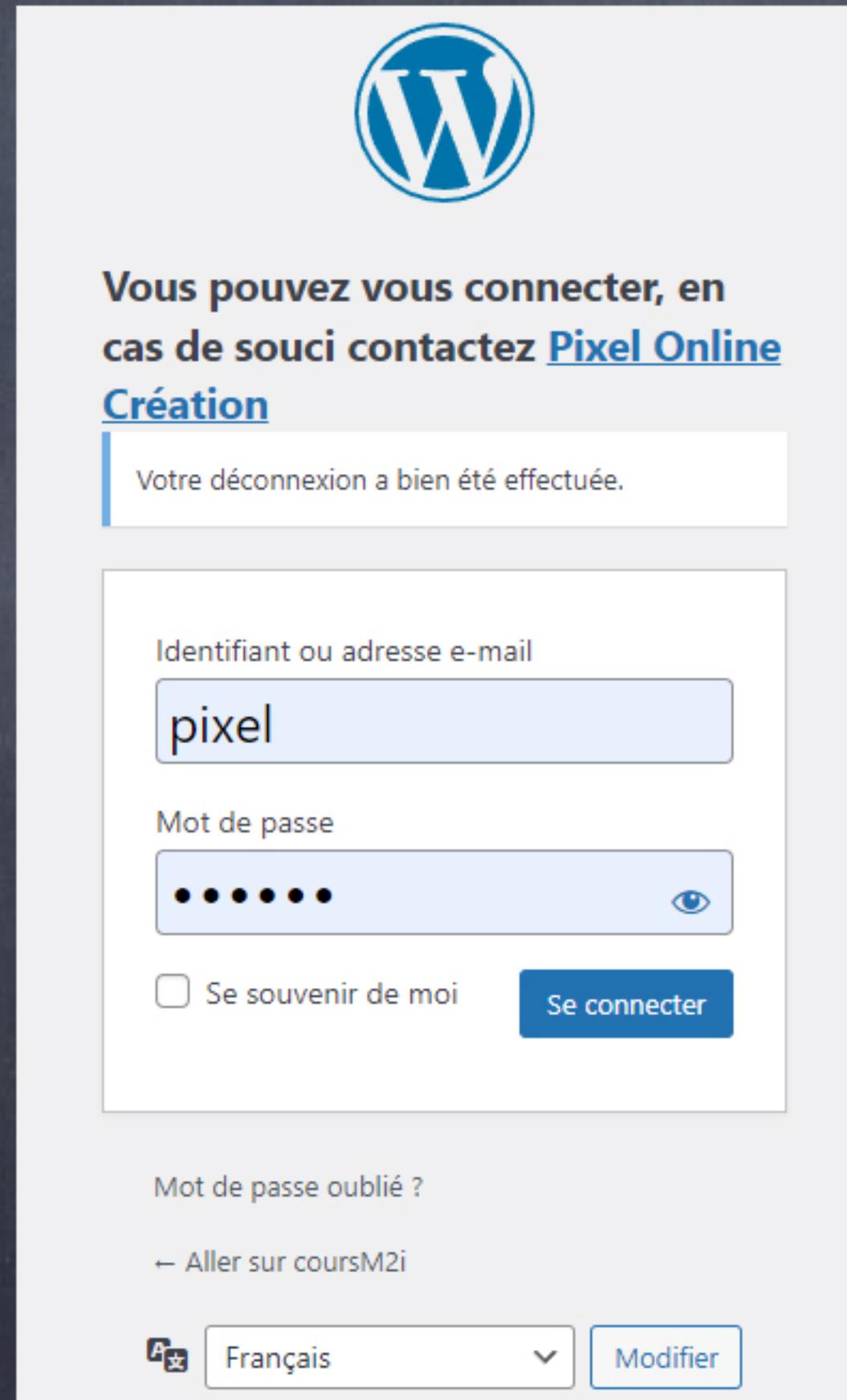
Le nom de la fonction de rappel qui sera exécutée lorsque le filtre se déclenchera.

La fonction `add_filter()` accepte également deux paramètres optionnels supplémentaires pour fixer la priority et le number of arguments. Ces paramètres fonctionnent de la même manière que la fonction `add_action()`.

Hooks

Dans votre functions.php

```
function custom_login_message( $message ) {
    if ( empty( $message ) ) {
        return "<h2>Vous pouvez vous connecter, en cas de souci contactez <a href='https://www.pixel-online.fr'>Pixel Online Création</a></h2>";
    }
    else {
        return $message;
    }
}
add_filter( 'login_message', 'custom_login_message' );
```



Hooks

La déclaration if-else de la fonction de rappel vérifie si le message de connexion est déjà défini, le plus souvent par une autre extension ou un autre thème. Dans ce cas, la fonction de rappel renvoie la valeur originale sans la modifier. C'est une façon d'éviter les conflits avec d'autres extensions ou thèmes.

Vous pouvez trouver une liste exhaustive de tous les filtres supportés par WordPress dans la page Plugin API/Filter Reference.

https://codex.wordpress.org/Plugin_API/Filter_Reference

Hooks

Il est possible de créer des customs Hooks . Des hooks non présents dans WP nativement

```
/**  
 * AJOUTE UN LIEN PIXEL ONLINE  
 */  
if( ! function_exists( 'render_pixel' ) ) {  
    function render_pixel() {  
        //if ( is_front_page() ) {  
        //    echo html_entity_decode(sprintf('%s', ' - Conception Web & référencement : <a href="https://www.pixel-online.fr" >Pixel Online Cr ation</a>' ));  
        // }  
    }  
}  
  
add_action( 'pixel', 'render_pixel' );
```

Il faut placer ce hook dans notre code, faites une copie de footer.php de divi vers votre thème enfant et modifiez le fichier pour lancer le hook et supprimer les crédit WP

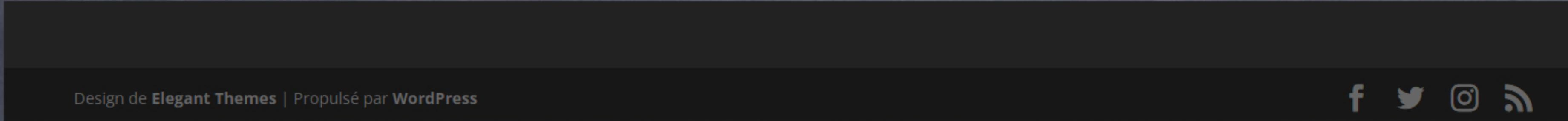
add_XXX : Ajoute la fonction

do_XXX : Exécute les fonctions gr ff es

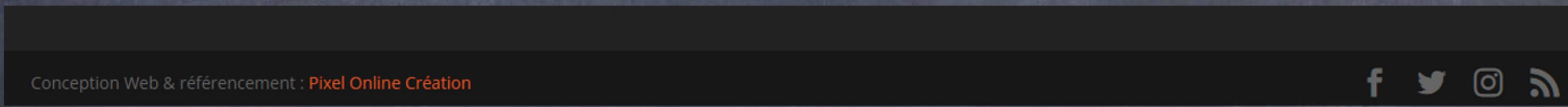
```
<div class= container cleartix >  
<?php  
    if ( false !== et_get_option( 'show_footer_social_icons', true ) ) {  
        get_template_part( 'includes/social_icons', 'footer' );  
    }  
  
    // phpcs:disable WordPress.Security.EscapeOutput.OutputNotEscaped  
    // echo et_core_fix_unclosed_html_tags( et_core_esc_previously( et_get_footer_credits() ) );  
    // phpcs:enable  
    do_action('pixel');  
?>  
    </div> <!-- .container -->  
</div>
```

Hooks

Avant :



Après :



Vous avez un contrôle total sur le code ne l'oubliez pas! Si un hook n'est pas natif, vous pouvez le créer avec `add_action` ou `add_filter` et le lancer à l'endroit voulu avec `do_action` ou `do_filter`!

Création d'un plugin

Nous avons vu qu'il était possible de télécharger des plugins pour ajouter ou modifier des fonctionnalités.

Il en existe énormément mais parfois vous avez besoin d'ajouter une fonctionnalité spécifique :

- système de devis
- création de hook
- connexion api
- etc.

Vous pouvez effectivement ajouter la fonctionnalité dans le thème enfant mais pour plus de modularité et un découplage thème / fonctionnalité il est préférable de créer un plugin.

Création d'un plugin

Nous allons créer un petit plugin :

Dans un premier temps créez un dossier plugin-m2i avec un fichier plugin-m2i.php dans le dossier wp-content/plugins

Les commentaires permettent de donner à WP la connaissance de votre plugin.

Le code en dessous initialise la session sur non créé et quelques constantes pour les chemins

```
<?php

/**
 * Plugin Name: Pixel Online plugin cours
 * Plugin URI: https://www.pixel-online.fr
 * Description: Mon plugin m2i
 * Version: 1.0.0
 * Author: PARIS Anthony
 * Author URI: https://www.pixel-online.fr
 * License: GPL2
 */

if ( ! defined( 'ABSPATH' ) ) exit; // Exit if accessed directly

if (session_status() == PHP_SESSION_NONE) {
    session_start();
}

define('BASE_PATH_PLUGIN', plugin_dir_path(__FILE__));
define('BASE_URL_PLUGIN', plugin_dir_url(__FILE__));|
```

Création d'un plugin

Rendez-vous dans extension / plugin pour activer ce dernier :

The screenshot shows the 'Plugins' section of the WordPress admin dashboard. There are two active plugins listed:

- Pixel Online plugin cours** (Version 1.0.0) by PARIS Anthony. It has a status of "Mon plugin m2i".
- Yoast SEO** (Version 19.6.1) by L'équipe Yoast. It is described as "La première solution SEO tout-en-un pour WordPress, y compris l'analyse des pages de contenu, les plans de site XML et bien plus encore."

A third item, "Extension", is listed but not activated.

Vous retrouverez les informations définies dans votre fichier.

Nous allons créer un plugin de manière professionnelle avec composer pour les dépendances

Ouvrez le terminal et placez vous dans le dossier du plugin

Création d'un plugin

```
PS D:\wamp64\www\wpm2i\wp-content\plugins\plugin-m2i> composer init
```

```
Welcome to the Composer config generator
```

This command will guide you through creating your composer.json config.

```
Package name (<vendor>/<name>) [hayu-/plugin-m2i]: wp/plugin-m2i
Description []:
Author [Anthony PARIS <contact@pixel-online.fr>, n to skip]:
Minimum Stability []:
Package Type (e.g. library, project, metapackage, composer-plugin) []: composer-plugin-wp
License []:
```

Define your dependencies.

```
Would you like to define your dependencies (require) interactively [yes]?
Search for a package: symfony/var-dumper
Enter the version constraint to require (or leave blank to use the latest version):
Using version ^6.1 for symfony/var-dumper
Search for a package: google/recaptcha
Enter the version constraint to require (or leave blank to use the latest version):
Using version ^1.2 for google/recaptcha
Search for a package:
Would you like to define your dev dependencies (require-dev) interactively [yes]?
Search for a package:
Add PSR-4 autoload mapping? Maps namespace "Wp\PluginM2i" to the entered relative path. [src/, n to skip]:
```

Ici nous ajoutons des dépendances pour l'exemple afin de démontrer que nous pouvons utiliser ce que l'on souhaite!

Création d'un plugin

Ouvrez le fichier `composer.json` et modifiez la partie `autoload > psr4` en prévision d'ajout de nos classes

Nous utiliserons l'autoloader de composer.

```
{  
    "name": "wp/plugin-m2i",  
    "description": "plugin WP",  
    "type": "composer-plugin-wp",  
    "require": {  
        "symfony/var-dumper": "^6.1",  
        "google/recaptcha": "^1.2"  
    },  
    "autoload": {  
        "psr-4": {  
            "Pixel\\Plugin\\": "inc/classes"  
        }  
    },  
    "authors": [  
        {  
            "name": "Anthony PARIS",  
            "email": "contact@pixel-online.fr"  
        }  
    ]  
}
```

Créez ensuite un dossier `inc/classes` dans le dossier du plugin

Création d'un plugin

Créons un classe controller.php dans inc/classes nous permettant d'executer notre plugin le plus tôt possible dans wordpress en utilisant un hook natif

```
<?php
namespace Pixel\Plugin;

final class Controller
{

    public function __construct($method="") {
        add_action( 'template_redirect', [$this,'plugin_is_page'] );
    }
    public function plugin_is_page() {
        echo "bonjour à tous";
    }
}
```

hook : https://developer.wordpress.org/reference/hooks/template_redirect/

Ici nous créons un classe controller permettant de lancer la méthode plugin_is_page() en front juste avant le chargement du template grace au hook template_redirect

Création d'un plugin

Dans plugin-m2i.php : Ce fichier se charge directement dans wordpress c'est notre «index» de plugin

```
if ( ! defined( 'ABSPATH' ) ) exit; // Exit if accessed directly

if (session_status() == PHP_SESSION_NONE) {
    session_start();
}

define('BASE_PATH_PLUGIN', plugin_dir_path(__FILE__));
define('BASE_URL_PLUGIN', plugin_dir_url(__FILE__));

//COMPOSER
require BASE_PATH_PLUGIN . 'vendor/autoload.php';
// include the classes

use Pixel\Plugin\Controller;

$controller = new Controller();
```

Nous ajoutons notre autoloder de composer, et lançons notre controller. Il faut régénérer le fichier de loading de composer

```
PS D:\wamp64\www\wpm2i\wp-content\plugins\plugin-m2i> composer dump-autoload -o
Generating optimized autoload files
Generated optimized autoload files containing 75 classes
PS D:\wamp64\www\wpm2i\wp-content\plugins\plugin-m2i> []
```

Création d'un plugin

Visualisez le code source de votre homepage. Votre écho s'affiche avant le Doctype!

```
nvoi à la ligne automatique □
1 bonjour à tous<!DOCTYPE html>
2 <html lang="fr-FR">
3 <head>
4     <meta charset="UTF-8" />
5 <meta http-equiv="X-UA-Compatible" content="IE=edge">
6     <link rel="pingback" href="http://localhost/wpm2i/xmlrpc.php" />
7
8     <script type="text/javascript">
9         document.documentElement.className = 'js';
10    </script>
11
12    <meta name='robots' content='noindex,nofollow' />
13    <!-- Le contenu de la page -->
```

Nous allons créer un shortcode, il s'agit d'un code court que l'on place sur notre site qui en étant interprété par wordpress peut lancer du code personnalisé!

En savoir plus sur les shortcodes :

<https://codex.wordpress.org/fr:Shortcode>

Création d'un plugin

Créons une classe Shortcode.php à la racine de notre plugin

```
<?php
namespace Pixel\Plugin;
/**
 * This is the class for this plugin.
 */
class Shortcode
{
    private $shortcodes;
    public function __construct()
    {
        $this->shortcodes = array();
    }
    public function addshortcode($shortcode)
    {
        array_push($this->shortcodes, $shortcode);
    }
    private function registerShortcodes()
    {
        if (count($this->shortcodes)) {
            foreach ($this->shortcodes as $shortcode) {
                $shortcode->init();
            }
        }
    }
    public function init()
    {
        // Register all shortcodes
        $this->registerShortcodes();
    }
}
```

Cette classe va enregistrer l'ensemble de nos shortcodes en utilisant la méthode addshortcode()

Dans notre controller nous ajouterons les shortcodes et on lancerà la méthode init() permettant d'enregistrer les shortcodes.

Dans ce tableau que l'on itérera nous lancerons chaque shortcode objet en utilisant sa propre méthode init.

Ajoutons maintenant un shortcode!

Création d'un plugin

Créez un dossier «Shortcode» dans inc/classes et un fichier DisplayTodaysDateShortCode.php avec le code suivant :

```
<?php
namespace Pixel\Plugin\Shortcodes;

class DisplayTodaysDateShortCode
{
    public function register($atts, $content = null)
    {
        return 'Aujourd\'hui nous sommes le : ' . date("d/m/Y");
    }
    public function init()
    {
        add_shortcode('display_todays', array($this, 'register'));
    }
}
```

Quand on lancera la méthode «init» wordpress prendra en charge le shortcode [display_todays] et lancera la méthode register qui affichera à l'écran la date!

Création d'un plugin

Revenons sur notre Controller.php pour mettre en place notre shortcode()

```
<?php  
namespace Pixel\Plugin;  
  
use Pixel\Plugin\Shortcode;  
use Pixel\Plugin\Shortcodes\DisplayTodaysDateshortcode;  
  
final class Controller  
{  
  
    public function __construct($method="") {  
        add_action( 'template_redirect', [$this,'plugin_is_page'] );  
    }  
    public function plugin_is_page() {  
        $shortcodeDate = new DisplayTodaysDateshortcode();  
        $shortcodes = new Shortcode();  
        // register all shortcodes  
        $shortcodes->addshortcode($shortcodeDate);  
        $shortcodes->init();  
    }  
}
```

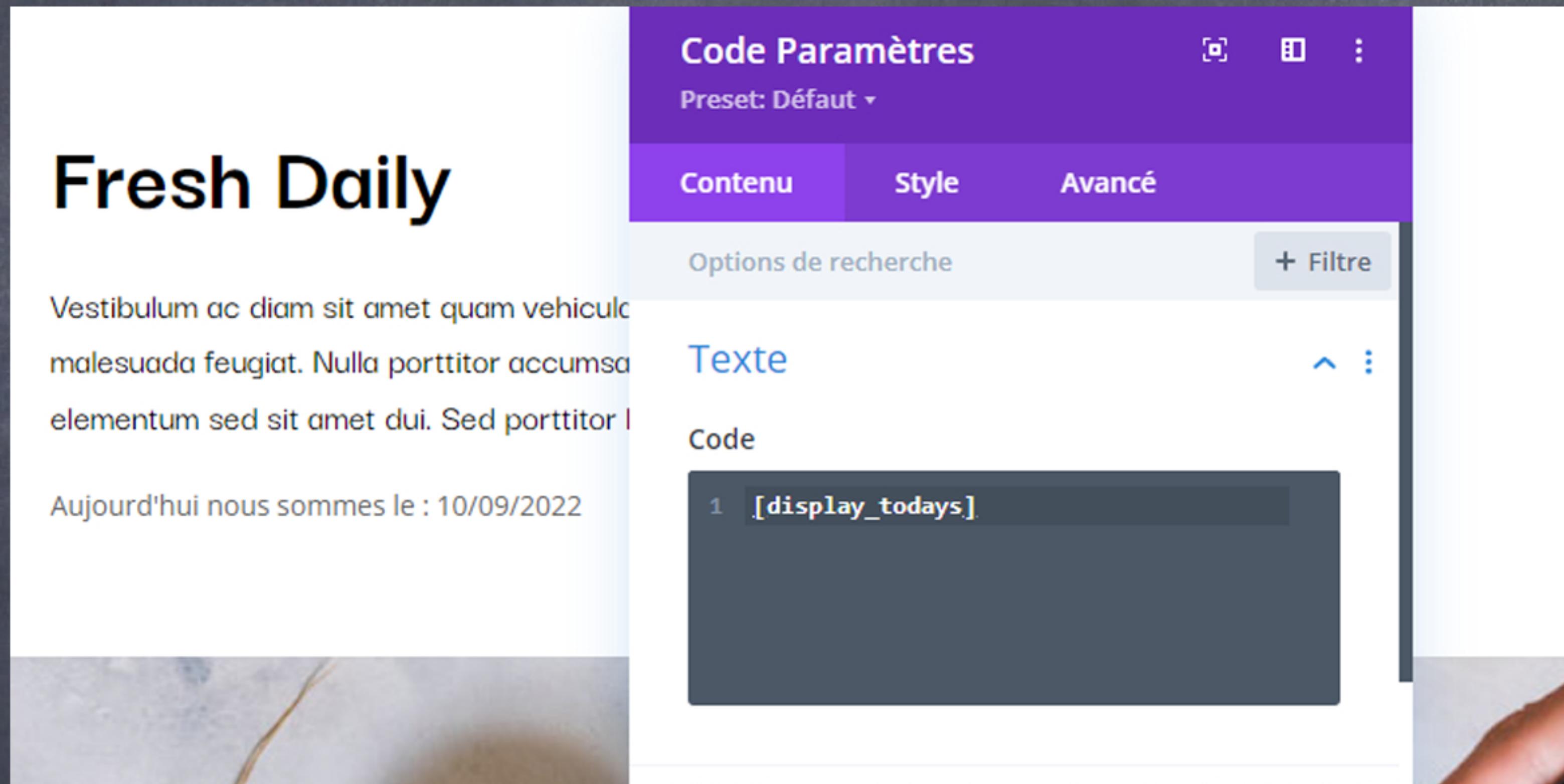
Relancez la commande pour dump l'autoload de composer

```
PS D:\wamp64\www\wpm2i\wp-content\plugins\plugin-m2i> composer dumpautoload -o  
Generating optimized autoload files  
Generated optimized autoload files containing 77 classes  
PS D:\wamp64\www\wpm2i\wp-content\plugins\plugin-m2i> █
```

Création d'un plugin

Ajoutons notre shortcode en front avec Divi! Et oui c'est possible!

Activez le visual builder et ajoutez un bloc gris de type «code»



Vous avez maintenant un système professionnel avec composer pour ajouter du code à travers des shortcodes() pour lancer du code personnalisé dans wordpress.

Création d'un plugin

Rajoutons un shortCode dans le dossier Shortcodes ajoutez un fichier Requetewp.php

```
<?php  
namespace Pixel\Plugin\Shortcodes;  
  
class Requetewp  
{  
    public function register($atts, $content = null)  
    {  
        ob_start();  
        $args = array(  
            'post_type' => 'post',  
        );  
        // 2. On exécute la WP Query  
        $my_query = new \WP_Query( $args );  
        // 3. On Lance la boucle !  
        if( $my_query->have_posts() ) {  
            while( $my_query->have_posts() ) {  
                $post = $my_query->the_post();  
                echo "<div class='custom-article-plugin'>";  
                echo "<h2>";  
                echo "<a href='".get_permalink($post)."'>";  
                the_title();  
                echo "</a>";  
                echo "</h2>";  
                echo "<div>";  
                the_content();  
                echo "</div>";  
                echo "<div>";  
                the_post_thumbnail();  
                echo "</div>";  
                echo "</div>";  
            }  
        }  
        // 4. On réinitialise à la requête principale (important)  
        wp_reset_postdata();  
        $infos = ob_get_clean();  
        return $infos;  
    }  
    public function init()  
    {  
        add_shortcode('display_requete_wp', array($this, 'register'));  
    }  
}
```

Sur le même principe que précédemment nous ajoutons un nouveau shortcode «display_requete_wp» nous permettant d'afficher la liste des articles de blog via un requête avec wordpress pour lancer une requête nous utilisons l'objet WP_Query en lui fournissant un tableau de paramètre.

Ensuite nous bouclons et affichons les éléments.

On utilise ob_start() et ob_get_clean() pour stocker le résultat et le renvoyer pour qu'il ne soit pas affiché directement mais seulement quand nous utiliserons le shortcode!

Création d'un plugin

Dans notre controller.php

```
<?php  
namespace Pixel\Plugin;  
  
use Pixel\Plugin\Shortcode;  
use Pixel\Plugin\Shortcodes\Requetewp;  
use Pixel\Plugin\Shortcodes\DisplayTodaysDateShortCode;  
  
final class Controller  
{  
  
    public function __construct($method="") {  
        add_action( 'template_redirect', [$this,'plugin_is_page'] );  
    }  
    public function plugin_is_page() {  
        $shortcodeDate = new DisplayTodaysDateShortCode();  
        $shortcodeRequete = new Requetewp();  
        $shortcodes     = new Shortcode();  
        // register all shortcodes  
        $shortcodes->addShortcode($shortcodeDate);  
        $shortcodes->addShortcode($shortcodeRequete);  
        $shortcodes->init();  
    }  
}
```

Pensez à lancer un «composer dump-autoload -o»

```
PS D:\wamp64\www\wpm2i\wp-content\plugins\plugin-m2i> composer dump-autoload -o  
Generating optimized autoload files  
Generated optimized autoload files containing 77 classes  
PS D:\wamp64\www\wpm2i\wp-content\plugins\plugin-m2i>
```

Création d'un plugin

Voici le résultat quand vous ajoutez le shortcode en front :

metus placerat luctus. Nulla eu tincidunt lectus.

Bonjour tout le monde !

Bienvenue sur WordPress. Ceci est votre premier article. Modifiez-le ou supprimez-le, puis commencez à écrire !



Pas forcément esthétique, rajoutons du css dans le plugin pour plus de modularité, qui d'ailleurs ne se chargera que quand on utilisera le shortcode!

Création d'un plugin

Créons le dossier assets/css et le fichier requete.css avec ce contenu :

```
.custom-article-plugin {  
    background: #f4f4f4;  
    padding: 30px;  
    border-radius: 5px;  
}  
  
.custom-article-plugin h2 {  
    font-weight: bold;  
    text-align: center;  
    text-transform: uppercase;  
}
```

Création d'un plugin

Modifions requetewp.php pour charger notre feuille CSS

```
class Requetewp
{
    public function __construct()
    {
        add_action( 'wp_enqueue_scripts', array($this, "css") );
    }

    public function css() {
        wp_register_style('requete_css', BASE_URL_PLUGIN.'assets/css/requete.css');
        wp_enqueue_style('requete_css');
    }
}
```

Résultat :

