Le guide ultime des codes courts ou shortcodes WordPress (avec des exemples pour créer les vôtres)

KINSTA

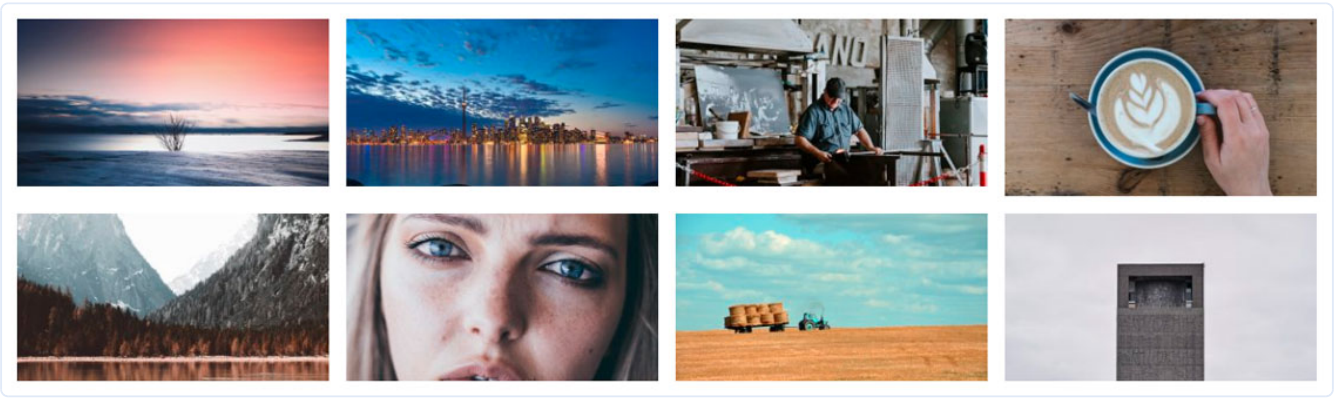
Les codes courts de WordPress sont une fonction puissante qui permet de faire des trucs sympas avec peu d'efforts. On peut faire à peu près n'importe quoi avec eux. Avec les codes courts, l'intégration d'éléments interactifs ou la création de mises en page complexes est aussi simple que l'insertion d'une seule ligne de code.

Si vous voulez ajouter une galerie, il suffit de saisir le code suivant :

```
[gallery ids="47 ,86, 92, 64, 48, 75, 89, 80" columns="4" size="medium"]
```

Cela produira une galerie avec les images des IDs mentionnés. Elle aura 4 colonnes et leur taille maximale sera « moyenne » (selon la définition de WordPress).

Il n'y a pas besoin d'un code HTML moche.



— Exemple de code court de galerie

Les codes courts éliminent le besoin de scripts compliqués. Même si vous avez peu ou pas de compétences en programmation, vous pouvez ajouter du contenu dynamique sans effort avec leur aide.

Ils sont très populaires au sein des développeurs de WordPress, car ils aident énormément à automatiser la création de contenu et de design. Les codes courts sont aux développeurs de WordPress ce que les macros sont aux analystes de données, ou les raccourcis clavier aux graphistes professionnels.

Dans ce guide, vous apprendrez tout ce qu'il y a à savoir sur les codes courts. Vous découvrirez comment travailler avec l'API des codes courts en créant vos propres codes courts. Enfin, nous discuterons de l'avenir des codes courts et de la place qu'ils occupent dans le nouvel éditeur de bloc de WordPress.

Ça vous excite ? Commençons !

Qu'est-ce qu'un code court ?

En bref, **Code court = Raccourci + Code**.

En général, les codes courts utilisent des balises entre crochets [] pour définir leur utilisation. Chaque code court remplit une fonction particulière dans un site. Elle peut être aussi simple que la mise en forme du contenu ou aussi complexe que la définition de la structure du site web dans son ensemble.

Par exemple, vous pouvez utiliser des codes courts pour intégrer des diaporamas, des formulaires, ou les tableaux de tarifs. Vous pouvez même les utiliser pour créer des modèles de design de pages réutilisables.

Une brève histoire des codes courts

Les codes courts ont d'abord été rendus populaires par un logiciel de forum en ligne appelé Ultimate Bulletin Board (UBB). En 1998, ils ont introduit le BBCode (Bulletin Board Code), une collection de balises faciles à utiliser pour permettre aux utilisateurs de formater facilement leurs articles.

BBCode Examples

Desired Formatting	Usage	Output
Bold/Strong Text	Be [b]bold[/b], but not too [b]bold[/b].	Be **bold**, but not too **bold**.
Strikethrough Text	Lightning [s]never[/s] strikes twice.	Lightning ~~never~~ strikes twice.
Change Text Color	Go [color=green]green[/color], or we scream!	Go green, or we scream!
Underline Text	Happiness [u]underlines[/u] success.	Happiness underlines success.

— Formatage facile avec des BBCodes simples

En tant que langage de balisage léger, le BBCode fonctionne sur les mêmes principes que le HTML, sauf qu'il est beaucoup plus simple.

L'utilisation de balises prédéfinies est également beaucoup plus sûre, car les utilisateurs ne peuvent pas insérer de code HTML et introduire des vulnérabilités en matière de sécurité. Par exemple, un utilisateur mal intentionné pourrait utiliser la balise <script> pour exécuter du code JavaScript et casser la fonctionnalité du site.

Peu après, d'autres logiciels de forum en ligne tels que phpBB, XMB Forum, et vBulletin a ajouté la fonctionnalité BBCode dans ses tableaux de message.

Les codes courts ont permis aux administrateurs d'avoir un plus grand contrôle sur ce que leurs utilisateurs peuvent et ne peuvent pas faire. De plus, ils ont permis aux utilisateurs de formater leur contenu par le biais de simples balises.

Pour les mêmes raisons de sécurité, WordPress empêche le code PHP de s'exécuter à l'intérieur du contenu du site. Pour surmonter cette limitation, WordPress 2.5 a introduit la fonctionnalité des codes courts en 2008 avec la publication d'API de code court. Il s'est avéré être l'une des fonctionnalités les plus utilisées par de nombreux développeurs d'extensions et de thèmes WordPress.

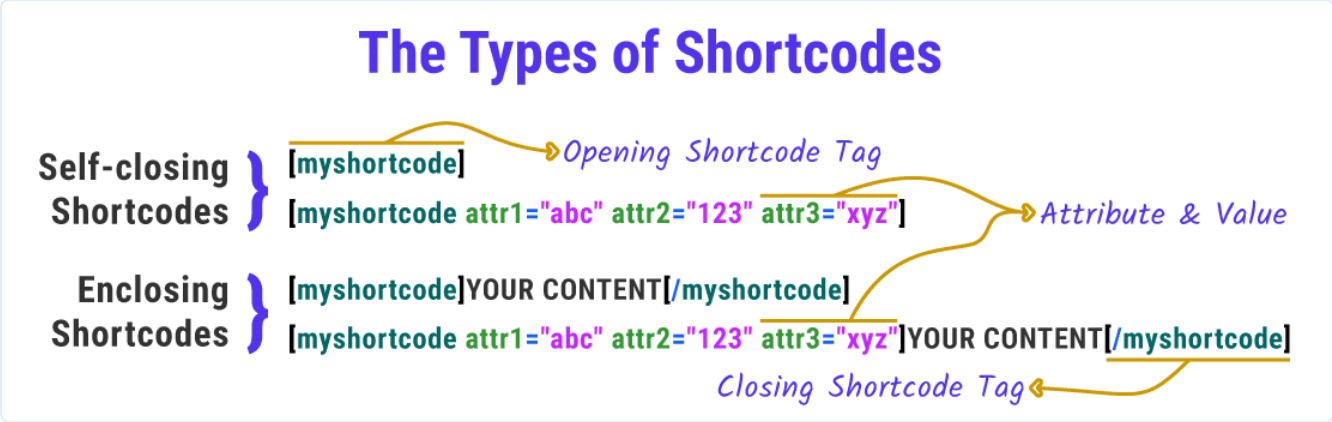
Que sont les codes courts de WordPress ?

Les codes courts WordPress sont des chaînes de crochets (**[ ]**) qui se transforment comme par magie en quelque chose de fascinant sur l'interface publique. Ils permettent aux utilisateurs de créer et de modifier facilement des contenus complexes sans avoir à se soucier de la complexité du HTML ou des codes d'intégration.

— Les codes courts de WordPress sont simples et faciles à utiliser

Les 2 types de codes courts

Il existe principalement deux types de codes courts dans WordPress.



— Les codes courts d'auto-fermeture et de fermeture peuvent être valides avec ou sans attributs.

- **Codes courts à fermeture automatique :** Ils n'ont pas besoin d'une balise de fermeture.

Exemple : Le code court **caption** n'a pas besoin de balise de fermeture. Nous ajoutons tout ce dont il a besoin avec différents attributs.

- **Codes courts de fermeture :** Ces derniers ont besoin d'une balise de fermeture. Les codes courts de fermeture manipulent généralement le contenu entre les balises d'ouverture et de fermeture.

Exemple *:* Le code court de **caption** est utilisé pour entourer une légende autour de n'importe quel contenu. Il est principalement utilisé pour ajouter une légende aux images, mais il fonctionne avec n'importe quel élément HTML.

Certains codes courts fonctionnent avec ou sans attributs. Cela dépend de la façon dont ils sont définis.

Les codes courts par défaut de WordPress

WordPress est livré avec 6 codes courts par défaut :

- audio : Intégrez des fichiers audio sur votre site web. Il comprend des commandes de lecture simples comme Play & Pause.

- caption **:** Enveloppez votre contenu avec des légendes. Il est surtout utilisé pour ajouter des légendes d'images, mais vous pouvez l'utiliser pour n'importe quel élément HTML.
- embed : Développe la fonctionnalité oEmbed par défaut. Ce raccourci vous permet de définir différents attributs à vos incorporations, comme la définition de leurs dimensions maximales.
- gallery : Insérez une simple galerie d'images sur votre site. Vous pouvez utiliser des attributs pour définir quelles images sont utilisées et personnaliser l'apparence de la galerie.
- playlist : Affichez une collection de fichiers audio ou vidéo avec ce code court à fermeture automatique. Vous pouvez lui donner un mode « dark » avec son attribut de style.
- video : Intégrer un fichier vidéo et le lire à l'aide d'un simple lecteur vidéo. Ce code court permet d'intégrer des vidéos dans ces formats : mp4, webm, m4v, webm, ogv, wmv, flv.

Pour plus de détails sur la manière d'utiliser les codes courts par défaut et sur les attributs qu'ils prennent en charge, vous pouvez vous référer aux documents du Codex liés.

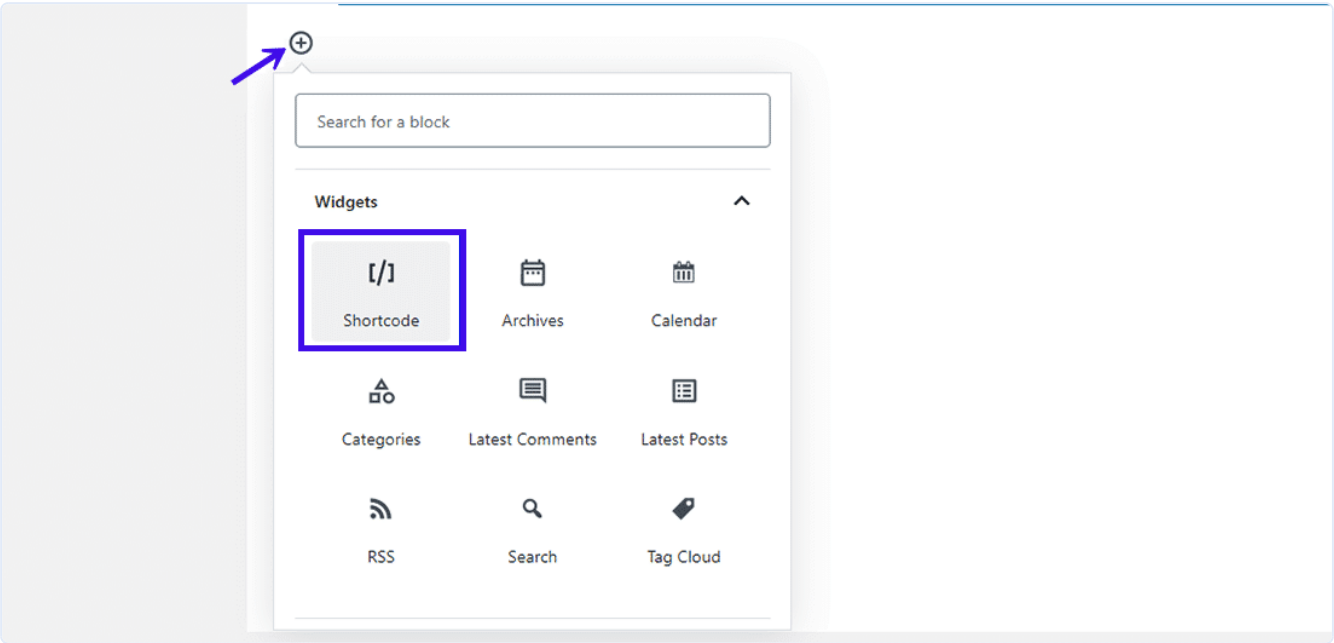
Comment utiliser les codes courts de WordPress

L'utilisation des codes courts dans WordPress est un processus simple. Mais cela dépend de l'endroit où vous voulez les ajouter sur votre site. Assurez-vous de lire la documentation sur les codes courts pour comprendre comment cela fonctionne. Apprenez les attributs qui sont supportés, afin d'obtenir exactement ce que vous voulez.

Utilisation des codes courts de WordPress dans les pages et les articles

Tout d'abord, allez dans l'éditeur de page ou d'article où vous voulez insérer le code court.

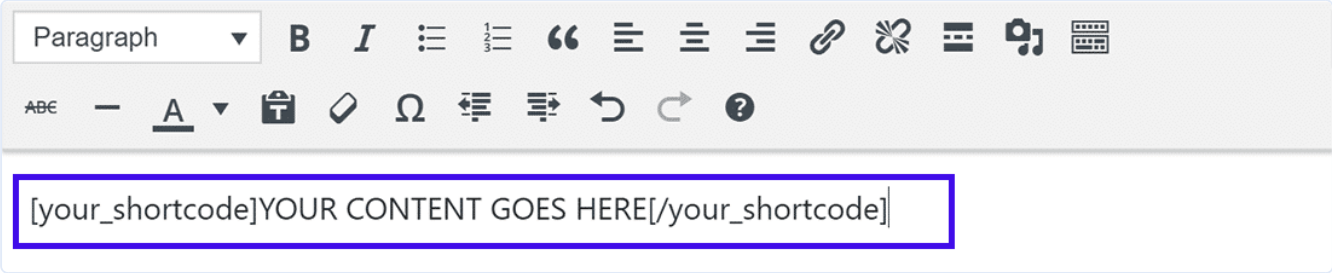
Si vous utilisez l'éditeur Gutenberg, vous pouvez ajouter la balise de code court dans le bloc autonome *Codes* courts. On peut la trouver dans la section *Widgets.*



— Ajout d'un bloc de code court à Gutenberg

— Le bloc de codes courts de Gutenberg

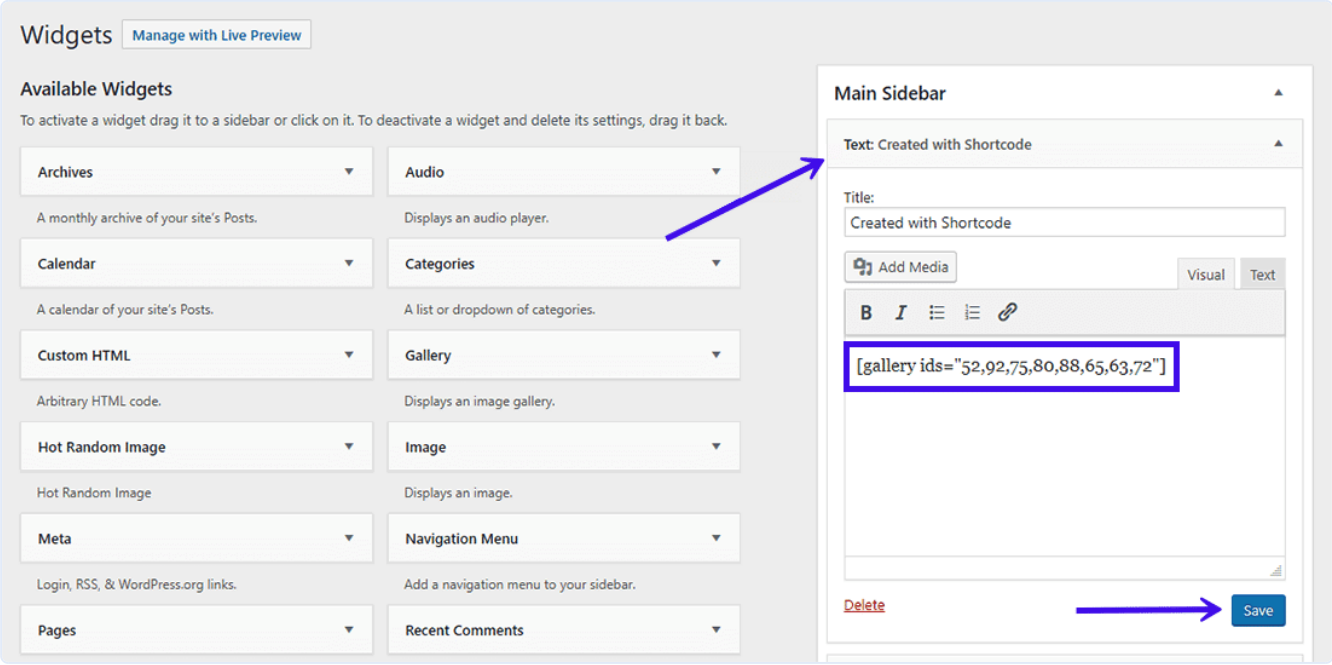
Vous utilisez toujours l'éditeur classique (ou l'extension Classic Editor) ? Vous pouvez saisir vos codes courts de la manière classique. Quelques codes courts peuvent même avoir un bouton dans l'écran de l'éditeur pour les insérer facilement.



— Ajout d'un code court dans l'éditeur classique

Utilisation des codes courts de WordPress dans les widgets de la barre latérale

Des codes courts peuvent également être insérés dans les widgets de la barre latérale. Pour les ajouter, allez dans **Apparence » Widgets** et ajoutez un widget **Texte** dans la section où vous voulez ajouter le code court.



— Ajoutez un code court dans votre barre latérale avec le widget Texte

Collez le code court dans le widget *Texte* et enregistrez-le. Vous pouvez visiter l'interface publique de votre site et voir le résultat du code court dans votre barre latérale.

— L'affichage du code court (galerie) qui peut être vu dans la barre latérale

Note : WordPress 4.8 et les versions inférieures ne prennent pas en charge les code court dans les widgets de la barre latérale. Lisez Améliorations des widgets dans la mise à jour de WordPress 4.9 pour en savoir plus.

Utilisation des codes courts de WordPress dans l'en-tête et le pied de page

Les codes courts de WordPress sont en général destinés aux pages, aux articles et aux widgets. Mais vous avez un moyen facile d'insérer des codes courts n'importe où dans votre site.

Disons que vous voulez ajouter un bouton d'appel à l'action dans votre pied de page, ou dans tous vos articles avant la section des commentaires. La fonction de callback do_shortcode() s'avère utile ici.

Vous devez ajouter le code suivant au fichier *header.php* ou *footer.php de* votre thème, ou à l'un de ses fichiers de modèle :

```
<?php echo do_shortcode("[name_of_your_shortcode]"); ?>
```

Le code court sera alors affiché à l'endroit où vous l'avez inséré.

Vous devez inclure les crochets entre les guillemets pour faire écho au code court. Le simple fait d'inclure son nom ne fonctionnera pas.

De même, vous pouvez utiliser la fonction de callback *do_shortcode()* pour activer les codes courts partout dans WordPress, comme dans la section des commentaires.

Une introduction rapide à l'API des codes courts

L'API des codes courts de WordPress définit la manière dont vous pouvez utiliser les codes courts pour personnaliser et étendre les fonctionnalités de votre site. Il permet aux développeurs de créer un contenu unique (par exemple des formulaires, des carrousels, des diaporamas, etc.) que les utilisateurs peuvent ajouter sur leurs sites en collant le code court correspondant.

Vous pouvez ajouter presque toutes les fonctionnalités que vous pouvez imaginer à votre site web à l'aide de codes courts.

L'API prend en charge les codes courts à fermeture automatique et les codes courts de fermeture. Cela gère toutes les analyses délicates et comprend des fonctions d'aide pour définir et récupérer les attributs par défaut.

Grâce à l'API, vous pouvez vous plonger directement dans le développement et la personnalisation des codes courts, plutôt que de perdre un temps précieux à définir des expressions régulières pour chaque code court que vous créez.

Comprendre les bases de l'API de code court

Chaque fois que vous ouvrez une page ou un article dans WordPress, il recherche les codes courts enregistrés tout en traitant le contenu du site.

Si un code court enregistré est trouvé, l'API des codes courts prend le relais et renvoie l'affichage du ou des codes courts. La chaîne renvoyée remplace la balise de code court à l'endroit où elle a été ajoutée.

Vous enregistrez un code court dans WordPress avec la fonction add_shortcode(). Voici comment on procède :

```
add_shortcode( 'shortcode_name', 'shortcode_handler_function' );
```

- **shortcode_name :** La balise WordPress sera recherchée lors de l'analyse du contenu des articles. L'API des codes courts vous recommande de n'utiliser que des lettres minuscules, des chiffres et des underscores pour définir sa valeur (évitez les tirets).
- **shortcode_handler_function :** La fonction de callback qui sera exécutée après que WordPress aura confirmé la présence d'un code court enregistré.
- La fonction handler shortcode est définie comme ceci :
- `function shortcode_handler_function( $atts, $content, $tag ){ }`
- **$atts :** Un tableau associatif d'attributs (c'est-à-dire un tableau de paires clé-valeur). Si vous ne définissez aucun attribut, il s'agira par défaut d'une chaîne vide.
- **$content :** Le contenu de fermeture, si vous définissez un code court de fermeture. C'est la responsabilité de la fonction handler de s'assurer que la valeur de $content est retournée à la sortie.
- **$tag :** La valeur de la balise du code court (*shortcode_name* dans l'exemple ci-dessus). Si deux ou plusieurs codes courts partagent la même fonction (valide) de callback, la valeur $tag vous aidera à déterminer quel code court a déclenché la fonction handler.

L'API analyse la balise du code court, ses attributs et le contenu inclus (le cas échéant) en contournant les valeurs vers la fonction handler, qui les traite et renvoie une chaîne de sortie.

Cette chaîne de sortie remplace la macro de raccourci sur l'interface publique de votre site. Ce que vous voyez finalement dans le navigateur, c'est cette sortie.

Où ajouter vos scripts personnalisés de codes courts ?

Vous pouvez ajouter vos scripts personnalisés de codes courts au fichier *functions.php du* thème ou les inclure dans une extension.

Si vous l'ajoutez à un fichier de thème, vous pouvez exécuter la fonction add_shortcode() telle qu'elle.

Mais si vous l'ajoutez à une extension, je vous recommande de ne l'initialiser qu'après le chargement complet de WordPress. Vous pouvez vous en assurer en enveloppant la fonction *add_shortcode()* dans une autre fonction. C'est ce qu'on appelle un enveloppement de fonction :

```
function shortcodes_init(){
 add_shortcode( 'shortcode_name', 'shortcode_handler_function' );
}
add_action('init', 'shortcodes_init');
```

La fonction add_action() accroche la fonction ***shortcodes_init*** pour ne se déclencher qu'une fois le chargement de WordPress terminé (elle est appelée 'init' hook).

Comment créer un code court personnalisé dans WordPress (niveau débutant)

Maintenant que nous avons couvert les bases, il est temps de créer un code court personnalisé.

Pour suivre les étapes indiquées ci-dessous, vous devez être familiarisé avec le code PHP et modifier vos fichiers de thème WordPress. Lorsque vous aurez terminé le tutoriel, vous aurez votre premier code court WordPress personnalisé prêt à être lancé !

Nous commencerons par le code court le plus simple possible, puis nous passerons à des codes courts plus complexes. Profitez des courtes étapes sur le chemin de la maîtrise des codes courts !

Exemple 1 : code court utilisant [current_year]

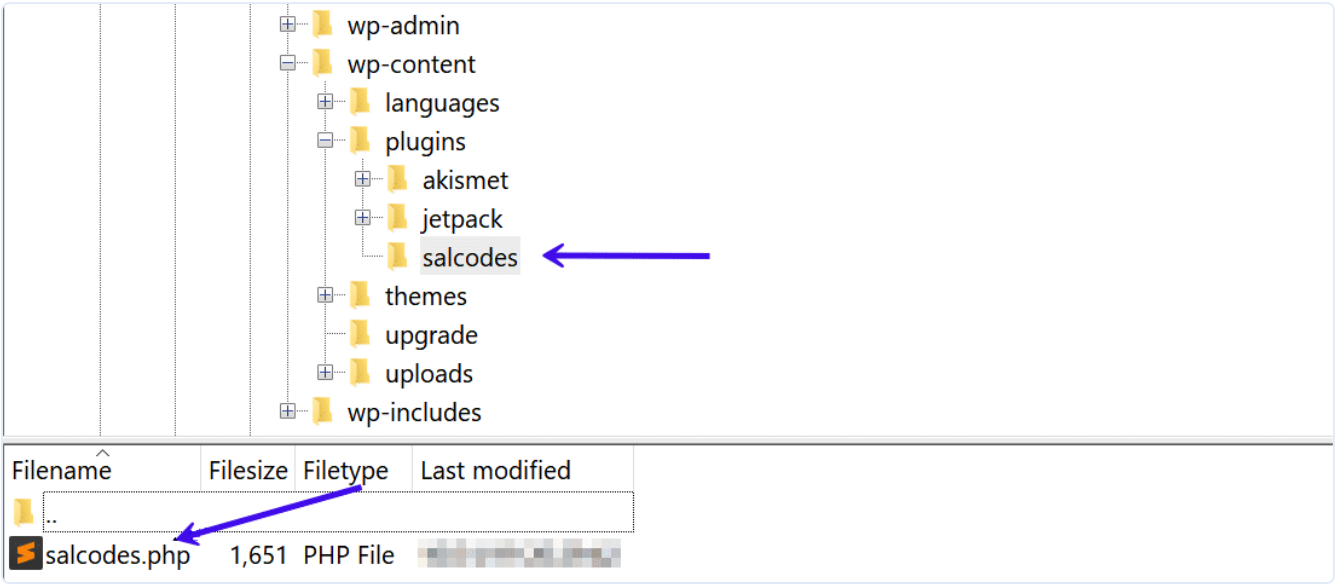
Créons un code court appelé [current_year] qui affiche l'année en cours sur votre site web.

Ce code court est utile si vous ajoutez à votre site web du contenu qui doit être mis à jour chaque année. Par exemple, en ajoutant une mention de droit d'auteur en bas de page de votre site.

Je vais utiliser une extension barebones pour ajouter mes fonctions de code court. Vous pouvez l'ajouter au fichier *functions.php de* votre thème et obtenir les mêmes résultats, mais je ne le recommande pas. Mais c'est bon pour tester et apprendre !

> **Info**
>
> Faites une sauvegarde avant d'apporter des modifications à votre site. Kinsta fournit des sauvegardes automatiques à tous ses clients

Commençons par créer une extension. Créez un nouveau dossier dans votre répertoire ***/wp-content/plugins/.***



— Notez l'emplacement du répertoire des extensions

Je nomme mon plugin « salcodes » mais vous pouvez lui donner le nom que vous voulez.

Dans le répertoire de l'extension **salcodes,** créez un fichier PHP du même nom (***salcodes.php***). Une fois fait, ajoutez l'en-tête suivant au fichier de votre extension :

```
<?php

/*
Plugin Name:  Salcodes
Version: 1.0
Description: Output the current year in your WordPress site.
Author: Salman Ravoof
Author URI: https://www.salmanravoof.com/
License: GPLv2 or later
License URI: https://www.gnu.org/licenses/gpl-2.0.html
Text Domain: salcodes
*/
```

Ce simple en-tête d'extension est suffisant pour nos besoins. Vous pouvez en savoir plus sur les exigences en matière d'en-tête d'extension dans le Codex WordPress. Enregistrez ce fichier, puis allez sur votre Tableau de bord WordPress pour activer l'extension.

Maintenant, enregistrons le code court et sa fonction handler. Pour ce faire, ajoutez le code suivant à votre fichier d'extension :

```
/**
 * [current_year] returns the Current Year as a 4-digit string.
 * @return string Current Year
*/

add_shortcode( 'current_year', 'salcodes_year' );
function salcodes_init(){
 function salcodes_year() {
 return getdate()['year'];
 }
}
add_action('init', 'salcodes_init');

/** Always end your PHP files with this closing tag */
?>
```

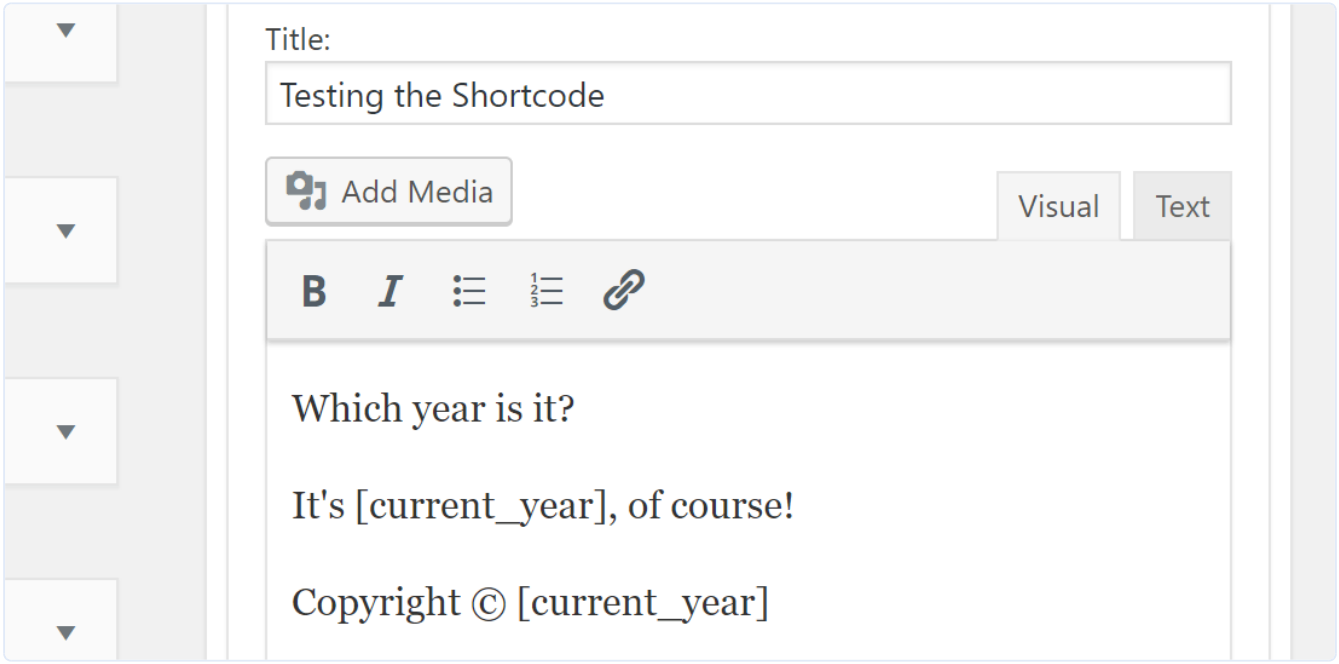
- La balise @return dans le commentaire PHP définit le type de sortie renvoyée. Il est suivi d'une brève description de la même chose.
- **current_year** est la balise ou le nom du code court. Elle définit la balise à fermeture automatique que vous devez ajouter à votre contenu, qui dans ce cas serait [current_year].
- **salcodes_year** est le nom de la *fonction de traitement des* codes *courts* qui renvoie la chaîne de sortie. Nous définirons cette fonction de rappel dans les prochaines lignes. Comme nous créons un simple code court à fermeture automatique, vous n'avez pas à lui transmettre de valeurs variables telles que $attributes, $content ou $tag.
- **salcodes_init** est la fonction d'encapsulation qui est reliée à « init » pour s'assurer que le code court est enregistré et exécuté seulement après que WordPress a fini de se

charger. La fonction ***add_action()*** intégrée à WordPress rend cela possible.

- **getdate()** est une fonction PHP qui renvoie un tableau d'informations sur la date de l'horodatage actuel. La clé de l'année contient la valeur de l'année en cours (sous la forme d'une chaîne de 4 chiffres). Ainsi, **getdate() ['year']** renvoie l'année en cours. Ce résultat est exactement ce que nous voulons.

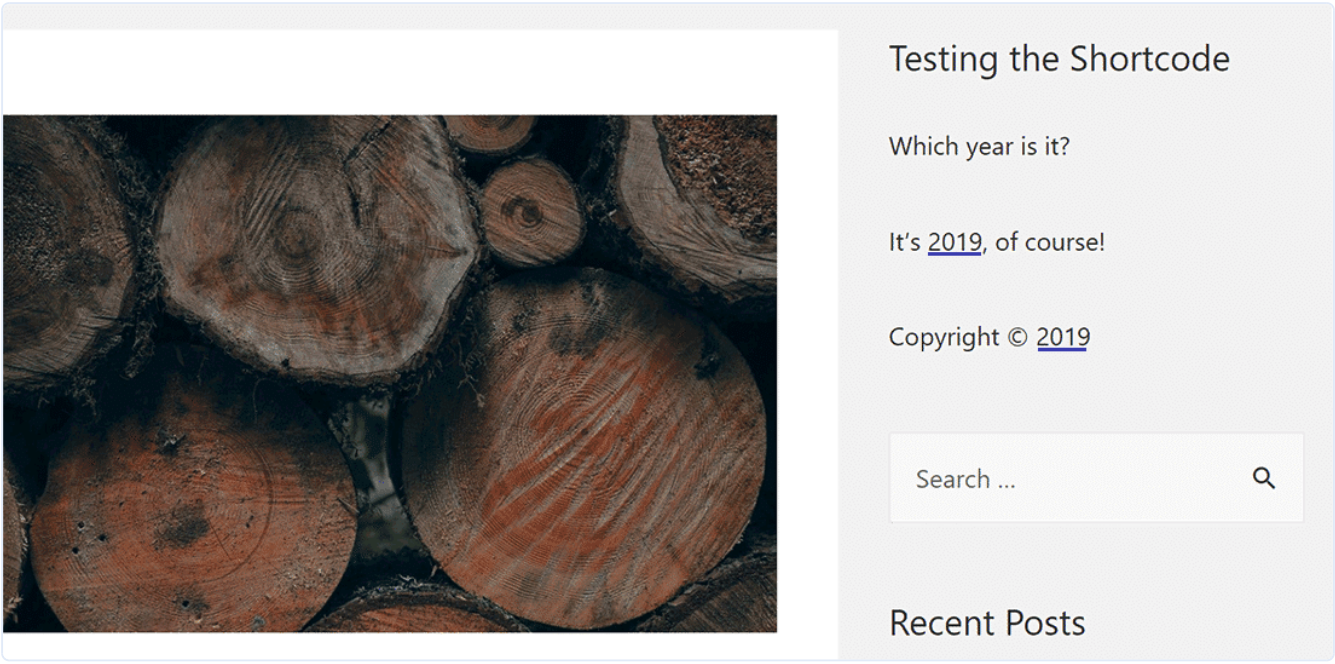
Enregistrez votre fichier d'extension. Il est maintenant temps de vérifier si le code court fonctionne comme prévu.

Ajoutez le code court n'importe où dans votre site (page, article, widget de barre latérale, etc.). Je l'ajoute au widget *Texte de la* barre latérale de mon site.



— Tester le code court personnalisé en l'ajoutant au site.

Et comme prévu, il fonctionne parfaitement.



— Affichage du code court de l'année

Félicitations pour avoir atteint votre première étape !

Le code court que vous venez de créer n'a pas de variables **$attributes** ou **$content** associées. Vous apprendrez à les utiliser dans les exemples suivants.

Exemple 2 : Code court pour un bouton d'appel à action

Créons un code court de **bouton d'appel à action** personnalisable. Ce sera aussi une fermeture automatique (désolé **$content**, vous devez attendre la prochaine).

Je veux que les utilisateurs puissent personnaliser la taille et la couleur du bouton d'appel à action avec les attributs du code court.

Comme le résultat final est un élément de bouton, ses attributs HTML tels que **href**, **id**, **class**, **target** & **label peuvent** être utilisés pour le personnaliser facilement.

Vous pouvez utiliser les attributs **id** et **class** pour styliser le bouton, car ce sont tous deux des sélecteurs CSS courants.

Je n’emballe pas ma fonction de gestionnaire ici pour que les choses restent simples à expliquer.

```
/**
 * [cta_button] returns the HTML code for a CTA Button.
 * @return string Button HTML Code
*/

add_shortcode( 'cta_button', 'salcodes_cta' );

function salcodes_cta( $atts ) {
 $a = shortcode_atts( array(
 'link' => '#',
 'id' => 'salcodes',
 'color' => 'blue',
 'size' => '',
 'label' => 'Button',
 'target' => '_self'
 ), $atts );
 $output = '<p><a href="' . esc_url( $a['link'] ) . '" id="' . esc_attr( $
 return $output;
}
```

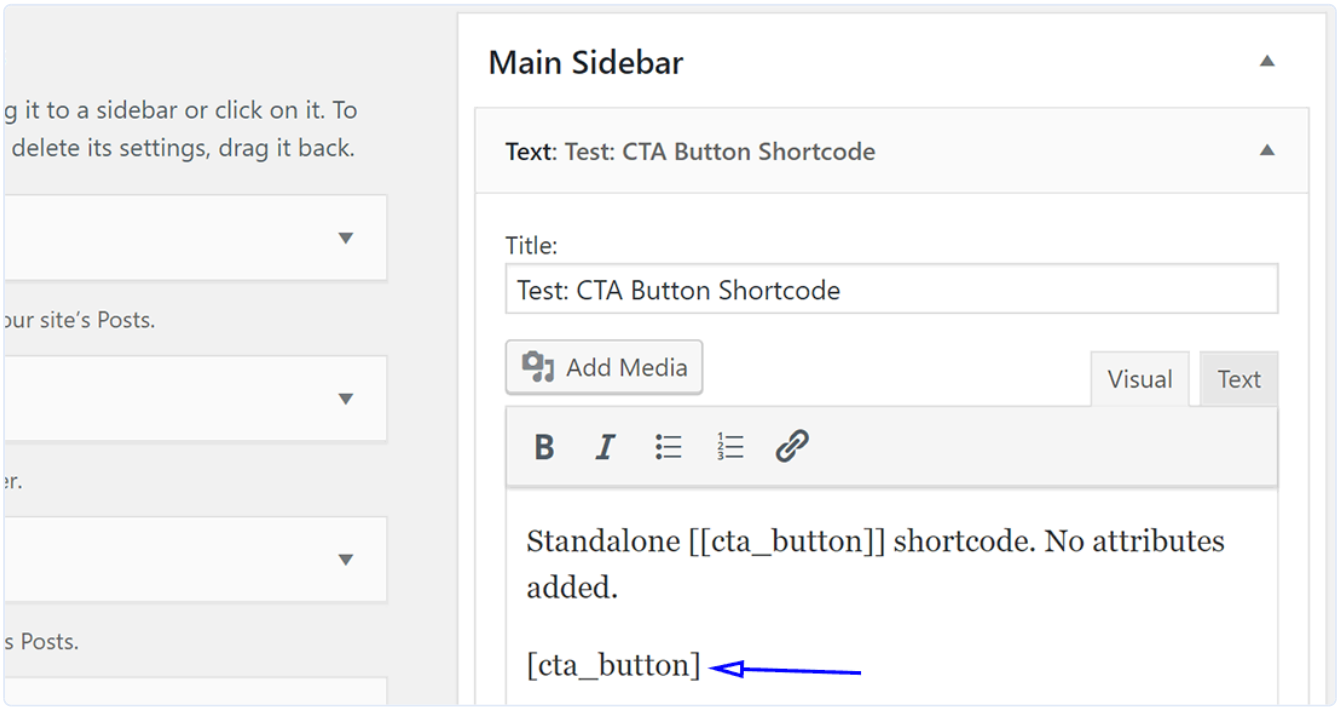
Wouah, il y a beaucoup de choses à déballer ici. Je vais vous l’expliquer ligne par ligne, pour que vous puissiez comprendre comment cela fonctionne.

- Nous avons abordé la fonction ***add_shortcode()*** et son fonctionnement dans la section précédente.
- shortcode_atts() est une fonction WordPress qui combine les attributs de code court de l’utilisateur avec des attributs connus. Il remplit les valeurs par défaut si nécessaire (vous pouvez également les définir vous-même). Le résultat sera un tableau contenant toutes les clés des attributs connus, fusionnées avec les valeurs des attributs de codes courts définis par l’utilisateur.

- Dans la fonction de gestion des codes courts, nous définissons une variable (**$a**) et l'affectons au tableau renvoyé par *shortcode_atts()*. Nous attribuons aux attributs leurs valeurs par défaut avec la syntaxe : **'attribute' => 'default-value'.** Par exemple, dans le code ci-dessus, nous définissons la valeur par défaut de l'attribut label au bouton avec la syntaxe 'label' => 'Button'**.**
- Nous pouvons extraire les valeurs de chaque clé d'attribut avec la syntaxe PHP pour les tableaux : **$a['attribut']**.
- La variable **$output** stocke le code HTML de l'élément bouton (balise <a> avec la classe 'button'). C'est la chaîne qui est finalement renvoyée par la fonction.

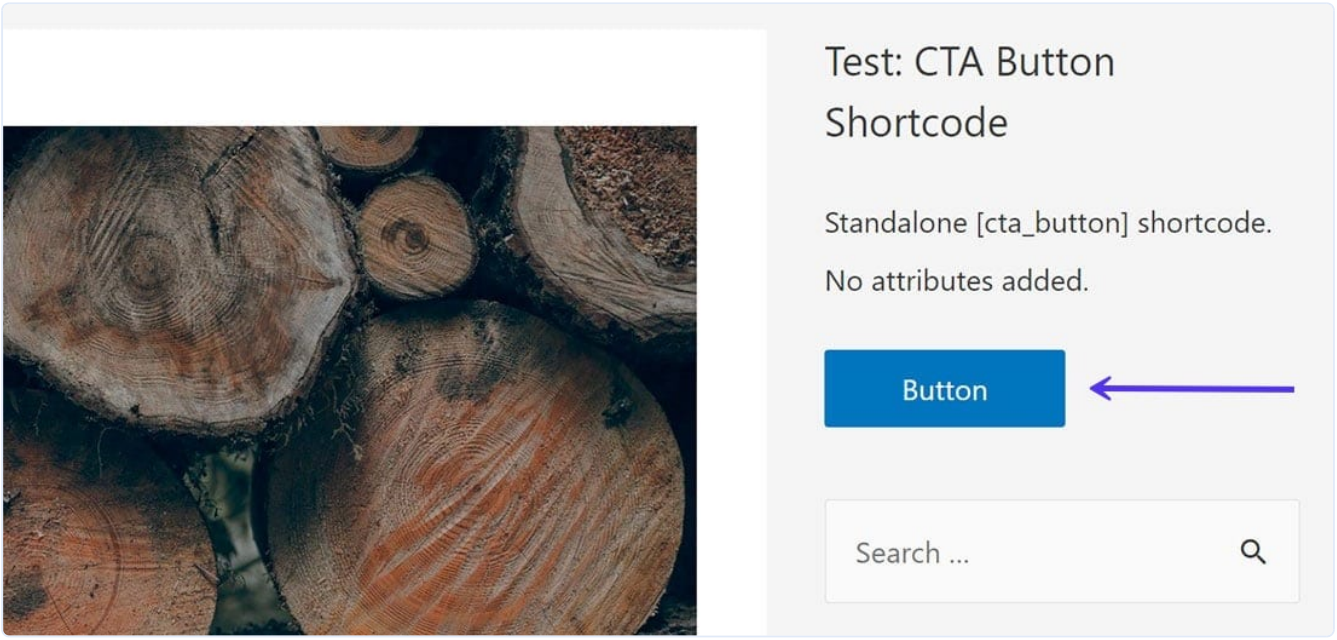
Si vous souhaitez que le lien par défaut soit l'URL de la page d'accueil du site, vous pouvez utiliser la fonction WordPress home_url().

Essayons d'utiliser le code court tel quel, sans aucun attribut défini, et voyons ce qui en ressort.

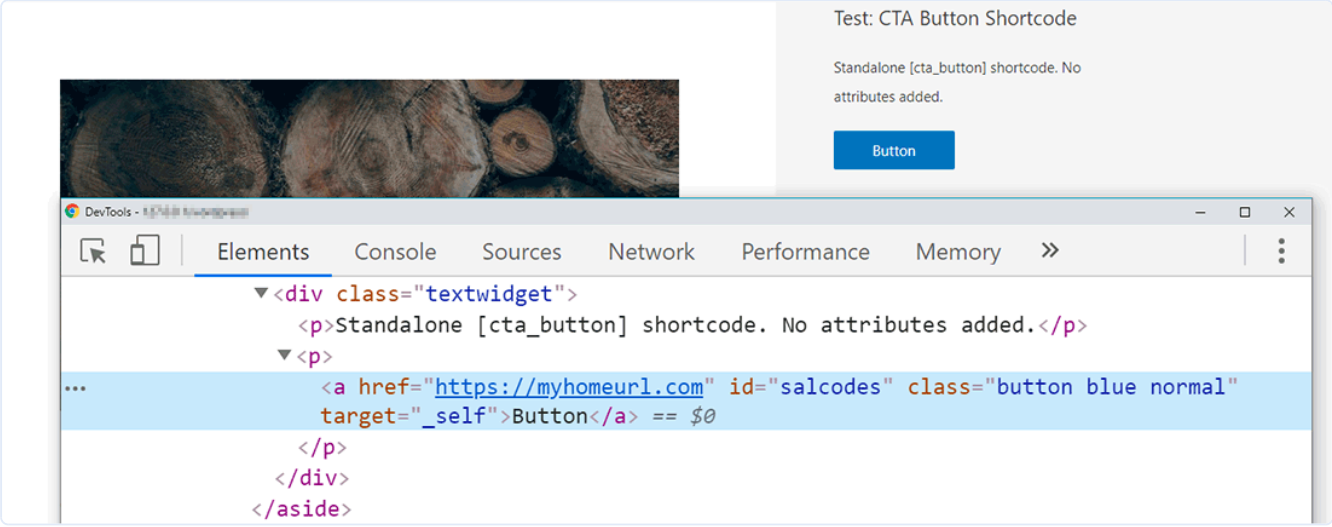


— J'ajoute le code court à un widget de texte de la barre latérale pour le tester.

Si vous vous demandez à quoi servent les doubles crochets (**[[**cta_button**]]**), ils s’appellent codes courts d’escape. Ils vous aident à afficher tout code court enregistré sur votre site sous forme de texte normal, comme dans l’image ci-dessous.



— Affichage du code court du bouton d’appel à action montrant qu’il fonctionne parfaitement comme prévu



— La sortie HTML du bouton d'appel à action sans attributs.

Les utilisateurs peuvent personnaliser la taille et la couleur du bouton à l'aide du code court. Nous avons déjà défini leurs valeurs par défaut dans la fonction de traitement, mais nous devons inscrire et faire figurer la feuille de style dans la liste des ressources disponibles. Cette feuille de style doit comporter toutes les classes définies dans le code court.

Vous pouvez également définir ces classes dans la feuille de style globale de votre thème, mais il est recommandé de les charger séparément. Cela garantit que même si vous changez ou mettez à jour votre thème WordPress, ces classes se chargeront toujours avec le code court.

```
/** Enqueuing the Stylesheet for the CTA Button */

function salcodes_enqueue_scripts() {
 global $post;
 if( is_a( $post, 'WP_Post' ) && has_shortcode( $post->post_content, 'cta
 wp_register_style( 'salcodes-stylesheet',  plugin_dir_url( __FILE__ ) .
     wp_enqueue_style( 'salcodes-stylesheet' );
```

```
 }
}
add_action( 'wp_enqueue_scripts', 'salcodes_enqueue_scripts');
```

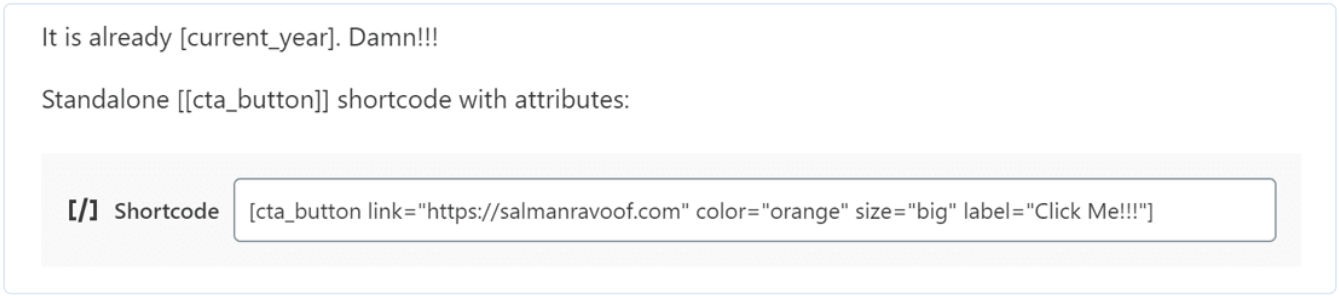
La fonction ***salcodes_enqueue_scripts()*** définit la variable globale **$post**, puis confirme deux conditions via :

- ***is_a()*** : vérifie si **$post** est une instance de l'objet Il s'agit de tous les types d'articles dans WordPress.
- ***has_shortcode()*** : vérifie si le contenu de l'article contient le code court **[cta_button].**

Si les deux conditions sont remplies, la fonction enregistre et met en file d'attente la feuille de style **style.css** incluse dans le dossier **CSS.** La fonction plugin_dir_url($file) permet d'obtenir facilement l'URL du répertoire de l'extension.

Je ne vous montrerai pas le code CSS ici, mais vous pouvez le trouver dans le code source à la fin de cette section.

Enfin, testons le code court **[cta_button]** en l'ajoutant au contenu de l'article :



— Remarquez le lien personnalisé, la couleur, la taille et les attributs du libellé.

L'image ci-dessous montre l'aspect du bouton d'appel à action sur l'interface publique :

— Le bouton d'appel à action a maintenant une nouvelle URL, une nouvelle couleur, une nouvelle taille et un nouveau libellé.

Maintenant que vous avez appris à définir des attributs personnalisés et à inclure des styles, vous pouvez ajouter diverses autres fonctionnalités à votre code court de bouton d'appel à action. Par exemple, vous pouvez donner à vos utilisateurs la possibilité d'ajouter des animations, des effets de survol et divers autres styles de boutons.

Exemple 3 : Code court utilisant $content.

Pour notre dernier exemple, construisons un code court appelé **[boxed]** qui affiche tout contenu entre ses balises dans une boîte avec des titres colorés.

Commençons par enregistrer le code court et définir sa fonction de gestionnaire.

```
/**
 * [boxed] returns the HTML code for a content box with colored titles.
 * @return string HTML code for boxed content
*/

add_shortcode( 'boxed', 'salcodes_boxed' );

function salcodes_boxed( $atts, $content = null, $tag = '' ) {
 $a = shortcode_atts( array(
```

```
 'title' => 'Title',
 'title_color' => 'white',
 'color' => 'blue',
 ), $atts );

 $output = '<div class="salcodes-boxed" style="border:2px solid ' . esc_a

 return $output;
}
```

- **$content = null** : ceci enregistre le code court comme un type d'encapsulation. Vous pouvez utiliser la variable **$content** dans votre fonction de gestion pour modifier votre sortie comme vous le souhaitez.
- **$tag = «** : ceci définit la variable **$tag** du code court. Ce n'est pas nécessaire dans cet exemple, mais c'est une bonne pratique de l'inclure.

Dans cet exemple, nous modifions le contenu en utilisant des styles CSS en ligne.

Les styles de toutes les classes utilisées à l'intérieur du code court sont enregistrés et mis en file d'attente comme nous l'avons fait dans l'exemple de code court précédent.

Mais le fait que deux codes courts utilisent la même feuille de style signifie que vous devez la charger si l'un d'eux est utilisé. Mettons donc à jour la fonction ***salcodes_enqueue_scripts() :***

```
/** Enqueuing the Stylesheet for Salcodes */

function salcodes_enqueue_scripts() {
 global $post;
 $has_shortcode = has_shortcode( $post->post_content, 'cta_button' ) || ha
 if( is_a( $post, 'WP_Post' ) && $has_shortcode ) {
 wp_register_style( 'salcodes-stylesheet',  plugin_dir_url( __FILE__ ) .
```

```
        wp_enqueue_style( 'salcodes-stylesheet' );
    }
}
add_action( 'wp_enqueue_scripts', 'salcodes_enqueue_scripts');
```

- **$has_shortcode** : une variable définie par l'utilisateur qui vérifie si l'un des codes courts existe sur la page ou l'article. Le || (opérateur OR) rend cela possible.

Maintenant, prenons notre code court **[boxed]**.

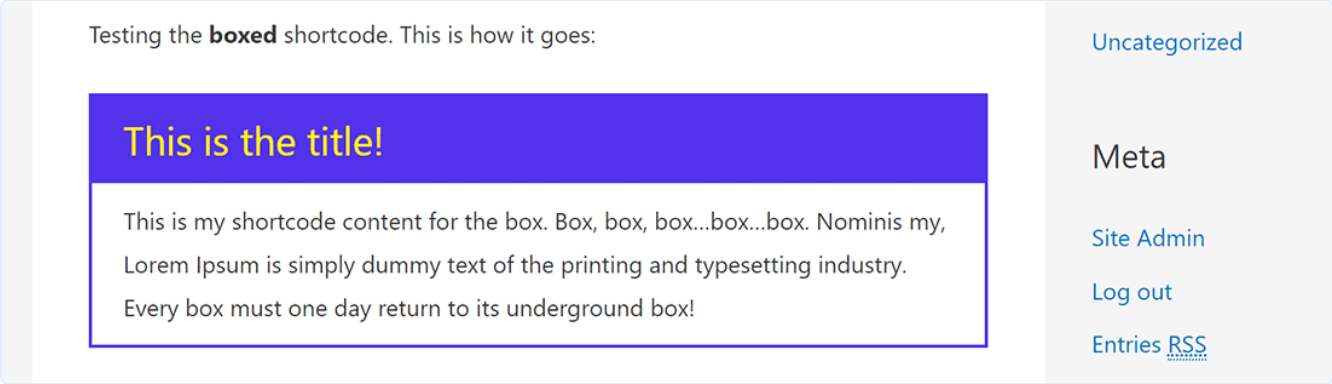
Testing the **boxed** shortcode. This is how it goes:

[/] Shortcode

```
[boxed title="This is the title!" title_color="yellow" color="#5333ed"]This is my shortcode content for the box. Box, box, box...box...box. Nominis my, Lorem Ipsum is simply dummy text of the printing and typesetting industry. Every box must one day return to its underground box![/boxed]
```

— Ajoutez le code court boxed ainsi que les attributs title, title_color et color.

La capture d'écran ci-dessous affiche le résultat que nous obtenons.



— Une jolie boîte n'est pas si difficile à obtenir après tout !

Maintenant que vous avez appris à faire vos propres codes courts, vous pouvez quitter **[boxed]** et y donner votre propre tournure. N'oubliez pas de partager vos créations avec nous !

Si vous le souhaitez, vous pouvez télécharger le code source de l'extension de code court ici.

Codes courts de WordPress : Avantages et inconvénients

Les avantages

- Les codes courts simplifient l'ajout de fonctionnalités complexes dans les sites WordPress. Vous pouvez ajouter presque tout en saisissant une seule ligne de code.
- Les codes courts automatisent le flux de développement. Ils éliminent la nécessité d'écrire des scripts complexes chaque fois que l'on veut insérer un certain élément.
- Les codes courts sont plus conviviaux que l'ajout de code HTML ou de scripts PHP.
- Les codes courts peuvent être regroupés dans des extensions. Même si vous mettez à jour WordPress ou si vous modifiez/mettez à jour votre thème, les codes courts resteront valables et continueront à fonctionner comme avant.
- Le regroupement des codes courts à l'intérieur des extensions les rend faciles à utiliser sur plusieurs sites WordPress. Si vous êtes un développeur qui gère de nombreux sites, le fait d'avoir tous vos codes courts personnalisés prêts à l'emploi est un atout majeur.

- Comme les codes courts acceptent également des attributs, les utilisateurs peuvent modifier le comportement du même code court en modifiant simplement les options de ses attributs.

Les inconvénients

- L'utilisation des codes courts n'est pas intuitive pour l'utilisateur final, surtout si un grand nombre d'entre eux sont utilisés sur une page. Dans ce cas, ils sont plus adaptés aux développeurs.
- Il est difficile de dire ce qu'un code court fait simplement en le regardant. L'équipe centrale de WordPress les a judicieusement nommés « mystery meat embed codes» pour cette raison précise.
- Les codes courts qui sont intégrés avec des thèmes cesseront de fonctionner si vous changez de thème.
- Les codes courts sont ambigus quant à leur syntaxe. Par exemple, certains d'entre eux sont favorables à la fermeture facultative, de sorte que vous pouvez les utiliser soit comme fermeture automatique, soit comme fermeture, voire les deux s'ils sont emboîtés. Vous pouvez deviner comment cela peut devenir très rapidement super confus.
- Les codes courts peuvent casser le HTML en raison de balises conflictuelles ou de problèmes d'interopérabilité. Il n'est jamais bon de les voir sur l'interface publique d'un site.
- Les codes courts ajoutent une charge supplémentaire sur votre serveur. Plus le nombre de codes courts sur votre page ou vos articles augmente, plus cette charge augmente. Trop de codes courts peuvent amener votre site web à ramer (par exemple, l'affichage de la plupart des constructeurs de pages).

Codes courts et blocs Gutenberg

L'introduction de Gutenberg a réduit l'intérêt des codes courts. Les utilisateurs peuvent désormais ajouter des blocs directement à partir de l'interface de l'éditeur, plutôt que de s'occuper des balises de codes courts, aussi simple soient-elles.

Et si vous souhaitez ajouter des codes courts, Gutenberg propose même un bloc dédié pour ajouter des codes courts. *Les blocs sont les nouveaux codes courts.*

> « Si vous pouvez le faire avec un code court, vous pouvez le faire avec un bloc. »
> — James Huff, ingénieur du bonheur chez Automattic

Cela explique pourquoi tous les codes courts populaires sont convertis en blocs. De nombreux développeurs de WordPress ont décidé de faire en sorte que leurs produits et services fonctionnent exclusivement avec l'éditeur de bloc (Gutenberg).

Mais cela ne signifie pas que c'est la fin du monde pour les codes courts. L'équipe centrale de WordPress a promis de nombreuses améliorations à l'éditeur de blocs, que vous pouvez voir dans Twenty Twenty, mais d'ici là, les codes courts sont là pour rester !

Résumé

Il est facile d'ajouter des fonctionnalités complexes n'importe où dans votre site WordPress, grâce à des codes courts. Ils donnent aux utilisateurs des balises faciles à taper qui peuvent être utilisées sans avoir à se soucier de l'utilisation de codes complexes.

Bien qu'il n'y ait pas de raccourcis dans la vie, il y a certainement de nombreux codes courts à utiliser dans WordPress. Mes préférés sont Shortcodes Ultimate et Shortcodes by par Angie Makes.

Et si vous ne trouvez pas celui que vous cherchez, vous pouvez en créer un vous-même.

Vous pouvez même prendre un raccourci pour créer votre code court personnalisé en utilisant l'extension Shortcoder. Elle s'occupe de l'essentiel pour vous. Et n'oubliez pas : la vie est courte, utilisez les codes courts !