WordPress : internationaliser son thème

Internalisation : introduction

Pour **internationaliser** votre thème WordPress, vous devez le rendre compatible avec la traduction dans différentes langues.

Info : l'internationalisation est également appelée **i18n** (il existe 18 caractères entre le premier « i » et le dernier « n »)

Cela permettra à vos utilisateurs d'afficher le contenu de votre site web dans leur langue préférée.

WordPress propose des fonctions spécifiques pour la traduction de texte.

Assurez-vous d'utiliser les fonctions de traduction au lieu de coder directement le texte « en dur » dans votre thème.

Exemple de mauvaise pratique :

```
<p>Publié le : <?php the_date(); ?></p>
```

Ici, le texte « **Publié le :** » est écrit en dur et ne pourra être traduit.

Les fonctions d'internationalisation

Vous devrez vous assurer que toutes les chaînes de texte statiques présentes dans le thème sont encapsulées dans des fonctions de traduction spéciales, généralement **`__()`** ou **`_e()`**. Ainsi que **`_n()`** pour gérer la traduction singulier / pluriel.

Ces fonctions permettent à WordPress de localiser le texte pour la traduction.

Fonction _e()

Lorsque vous avez besoin d'afficher un texte dans votre thème ou votre plugin, et que vous souhaitez le rendre traduisible, vous devez encapsuler cette chaîne de texte dans la fonction **`_e()`**.

La syntaxe est la suivante :

```
_e( $text, $domain );
```

- **`$text`** **:** c'est la chaîne de texte que vous souhaitez traduire.
- **`$domain`** **:** c'est le domaine de traduction pour le thème ou le plugin.

Par exemple : `_e( 'Hello World', 'mon-theme' );`

Remarque : la fonction **`_e()`** fait référence à la fonction **`echo()`** de **PHP** permettant d'afficher des informations à l'écran.

Conseil : une bonne pratique est d'écrire le texte original en anglais.

Fonction __()

La fonction **__()** est une autre fonction de traduction intégrée dans WordPress.

Tout comme la fonction **_e()**, elle est utilisée pour rendre les chaînes de texte traduisibles dans un thème ou un plugin WordPress.

Cependant, contrairement à _e(), la fonction __() ne renvoie pas directement le texte traduit, mais le renvoie sous forme de valeur de retour.

Cela signifie que vous pouvez stocker le texte traduit dans une variable pour une utilisation ultérieure.

Pour rendre une chaîne de texte traduisible, vous devez encapsuler cette chaîne de texte dans la fonction __() de la manière suivante :

```
__( $text, $domain );
```

- **$text :** c'est la chaîne de texte que vous souhaitez traduire.
- **$domain :** comme expliqué précédemment, c'est le domaine de traduction pour le thème ou le plugin.

Par exemple : `__( 'Hello World', 'mon-theme' );`

Exemple, pour les labels d'un CPT (Custom Post Type) déclaré dans le fichier functions.php du thème :

```
<?php

// type de publication
$labels = array(
    'name' => __( 'Portfolio', 'mon-theme' ),
    'all_items' => __( 'Toutes les réalisations', 'mon-theme' ),
    'singular_name' => __( 'Réalisations', 'mon-theme' ),
    'add_new_item' => __( 'Ajouter une réalisation', 'mon-theme' ),
    'edit_item' => __( 'Modifier la réalisation', 'mon-theme' ),
    'menu_name' => __( 'Portfolio', 'mon-theme' )
);
```

Fonction _n()

La fonction **_n()** est une autre fonction de traduction intégrée dans WordPress. Contrairement aux fonctions **_e()** et **__()**, qui sont utilisées pour traduire des chaînes de texte, **la fonction _n() est spécifiquement conçue pour gérer les règles de pluriel lors de la traduction**.

La fonction **_n()** est utilisée pour gérer les formes plurielles en fonction d'un nombre donné. Elle nécessite généralement deux chaînes de texte, une pour le singulier et une pour le pluriel.

Voici la syntaxe :

```
_n( $singular, $plural, $number, $domain );
```

- **`$singular` :** c’est la chaîne de texte à afficher lorsque le nombre est égal à 1 (cas singulier).
- **`$plural` :** c’est la chaîne de texte à afficher lorsque le nombre est différent de 1 (cas pluriel).
- **`$number` :** c’est le nombre qui détermine si la forme singulière ou plurielle doit être affichée.
- **`$domain` :** le domaine de traduction pour le thème ou le plugin.

Exemple :

```
<?php
    $comment_count = get_comments_number(); // Obtenir le nombre de
commentaires
?>

<h2>
    <?php echo sprintf( _n( '%s commentaire', '%s commentaires',
$comment_count ), $comment_count; ?>
</h2>
```

`%s` indique où il faudra afficher le nombre dans la chaine de caractères.

La fonction **`sprintf()`** va remplacer **`%s`** par le nombre de commentaires.

Générer les fichiers de traduction

Créez ensuite des fichiers de traduction qui vont contenir les équivalents de texte dans différentes langues.

Vous devrez créer un fichier **.pot (Portable Object Template)** pour votre thème qui contient la liste de toutes les chaînes pouvant être traduites.

Vous pouvez le faire à l’aide d’un outil comme **Poedit**.

Poedit facilite la gestion des traductions en fournissant une interface conviviale pour parcourir les chaînes de texte à traduire, saisir les traductions, et marquer les traductions comme complètes ou incomplètes.

Alternative : l’extension **Loco translate**

Une fois que vous avez votre fichier **.pot**, vous pouvez commencer à traduire les chaînes de texte dans différentes langues en créant des fichiers **.po (Portable Object)** pour chaque langue cible.

Ex. fr_FR.po

Ces fichiers contiendront les traductions spécifiques.

Générez les fichiers **.mo (Machine Object)** qui sont des fichiers binaires générés à partir des fichiers **.po** (version compilée).

Ils permettent à WordPress de charger les traductions sur le site.

Charger les traductions

Vous devez charger les fichiers **.mo** correspondants dans votre thème.

Cela peut être fait à l'aide de la fonction **`load_theme_textdomain()`** dans le fichier **functions.php** de votre thème.

Voici comment utiliser cette fonction :

```
function mon_theme_load_textdomain() {
    load_theme_textdomain( 'mon-theme', get_template_directory() .
'/languages' );
}

add_action( 'after_setup_theme', 'mon_theme_load_textdomain' );
```

Dans cet exemple, nous supposons que le texte du thème est encapsulé dans les fonctions de traduction avec le **textdomain `'mon-theme'`**. Vous devez remplacer **`'mon-theme'`** par le **textdomain** réel de votre thème utilisé dans les fonctions de traduction.

Le **textdomain** permet de faire la distinction entre les différentes traductions du cœur de WordPress, des thèmes et extensions.

Gérer le multilingue

Si vous souhaitez offrir une solution conviviale pour que les utilisateurs puissent traduire votre thème, vous pouvez recommander l'utilisation de plugins de traduction, tels que **WPML (WordPress Multilingual)**, **Weglot** ou **Polylang**.

Ces plugins facilitent la gestion de contenu multilingue.