Liaison de pièces jointes

Elementor, PHP, Wordpress

Web Compétences : 3

Quand on publie une publication, il arrive régulièrement qu'une pièce jointe y soit liée. Wordpress propose un type bien particulier pour gérer celles-ci, sorte de lien dans la publication. Un petit tour d'horizon depuis l'autorisation de joindre un certain type de fichiers, jusqu'à son téléchargement. Suivez le guide

Publié lundi 1 juin 2020, 21h01
Modifié lundi 26 août 2024, 10h05

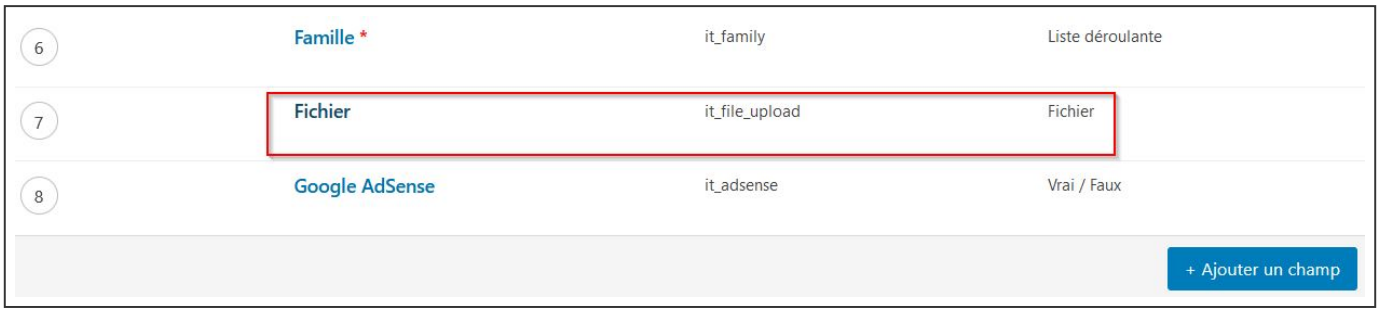
By Olivier Paudex

Introduction

Ce site utilise les pièces jointes dans deux des trois types personnalisés de publications, les parcours à vélo et le blog informatique. Pour la section parcours, les liens proposés sont présents sur toutes les publications puisqu'ils sont les **GPX**, à savoir les fichiers qui tracent le parcours et peuvent être lu sur n'importe quel appareil **GPS**. Dans la section blog informatique, les fichiers peuvent être de n'importe quel nature, souvent un fichier de configuration ou système qui accompagne le sujet. L'idée de base était de pouvoir téléverser une pièce jointe directement à l'écriture de la publication via l'interface WordPress, sans que celle-ci devienne indisponible si la publication devait changer d'emplacement. C'est là qu'intervient la pièce jointe, aussi appelée dans le jargon WordPress, **l'attachement**.

Le champ ACF de type fichier

ACF permet de créer des champs de type fichier et c'est exactement ce dont il est question pour que le rédacteur de la publication puisse téléverser un fichier depuis l'interface Gutenberg de WordPress.



Le champ ACF de type fichier

Les paramètres sont par défaut. Une fois le champ créé dans ACF, on peut apercevoir une nouvelle possibilité d'ajouter un fichier dans l'éditeur.

La sélection de fichiers depuis l'editeur

L'autorisation de type de fichier spéciaux

WordPress autorise le téléversement de certains types de fichiers seulement, comme les images **JPG** ou **PNG**, les documents **DOC** ou **XLS**, mais pas des fichiers ayant comme extensions, par exemple **SQL**. On encourt toujours un certain risque de vouloir autoriser d'autres types de fichiers, mais il est possible de le faire.

les fichiers de type SQL sont interdits

Le fichier wp-config.php

Ce fichier contient les paramètres globaux de l'installation WordPress, comme le nom de la base de données, le login de l'administrateur, et même son mot de passe. **Attention, toutes ces informations ne sont pas cryptées, alors ne le divulguer pas.**

Pour autoriser un certain type de fichiers, il va falloir ajouter une ligne de configuration au fichier **wp-config.php**.

```
/** Authorized files */define('ALLOW_UNFILTERED_UPLOADS', true);
```

Mais ce n'est pas tout. Il va encore falloir ajouter le type de fichiers**MIME** à la configuration.

Comment trouver le type MIME du fichier ?

Chaque fichier appartient à un certain type appelé **MIME**. Même si un fichier porte une extension comme **TXT**, il n'est pas nécessairement de type texte. Cette astuce est souvent utilisé par les virus informatiques pour tromper l'utilisateur et/ou le système. Pour vérifier que le type de fichiers est bien ce qu'il prétend

être, il faut vérifier son type **MIME**. Il existe plusieurs logiciels libres en ligne pour effectuer ces tests dont celui-ci :

https://htmlstrip.com/mime-file-type-checker

Par exemple, pour autoriser le fichier de type **SQL** ci-dessus, le logiciel me retourne le type **"text/x-Algol68"**

Autoriser un type MIME

Il va falloir créer un nouveau fichier PHP et lui ajouter ces quelques lignes. La syntaxe est toujours la même. Il faut juste connaître l'extension du fichier et son type **MIME**. Il est tout à fait possible d'ajouter plusieurs type **MIME** pour une seule extension, mais toujours une extension par ligne de code.

```
// Authorize SQL files to be uploaded into the media library
function autorize_tracks_mimes_type ($mime_types) {
  /* Use a mime type check application like 'https://htmlstrip.com/mime-file-type-checker' to r
  // Adding gpx extension
  $mime_types['sql'] = 'text/x-Algol68';
  return $mime_types;
}add_filter ('upload_mimes', 'autorize_tracks_mimes_type');
```

Une fois l'interface réalisée avec ACF et le type de fichier autorisé, le rédacteur de la publication peut téléverser une pièce jointe qui va se lier automatiquement à celle-ci. On peut apercevoir l'état du fichier dans l'onglet **Médias** de WordPress qui note que le fichier est attaché. Autrement dit, il est devenu **un attachement** de la publication.

Modifier la destination des fichiers téléversés

Par défaut, WordPress va téléverser les fichiers dans le dossiers **"uploads"**, qui se trouve être un sous-dossier de **"wp-content"**. Les fichiers sont ensuite classés par années, puis par mois. Si les fichiers sont de types images, WordPress va effectuer automatiquement un redimensionnement des images en plusieurs formats avant de les afficher dans **la bibliothèque de Médias**.

Dans le cas de téléversements de fichiers liés aux publications, surtout si ces derniers ne sont pas des images, il est peut-être judicieux de vouloir les sauvegarder dans d'autres dossiers. Pour réaliser ceci-ajoutez le code ci-dessous dans un fichier PHP.

```
// Change upload path for IT attachments files
function upload_it_attachment_prefilter($errors) {
  function field_name_upload_dir($uploads) {

    // Upload path
    $uploads['path'] = $uploads['basedir'].'/it';
    $uploads['url'] = $uploads['baseurl'].'/it';
    $uploads['subdir'] = '';
    return $uploads;
  }
  add_filter('upload_dir','field_name_upload_dir');
  return $errors;
}add_filter('acf/upload_prefilter/name=it_file_upload', 'upload_it_attachment_prefilter');
```

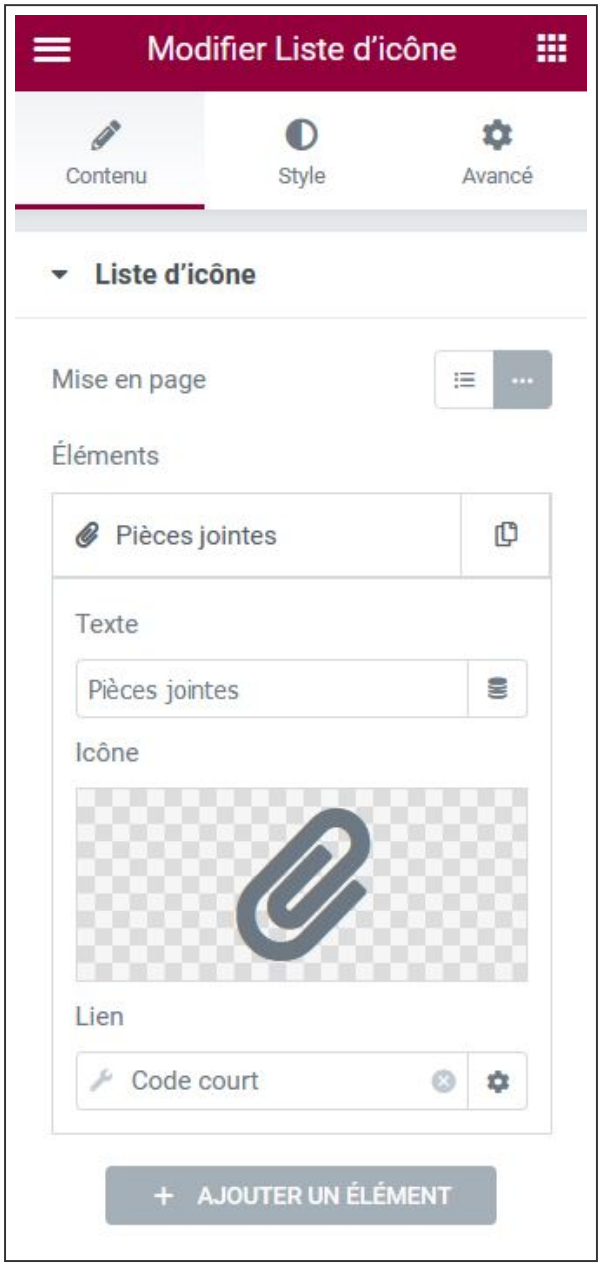
Elementor n'est pas le seul à proposer des fonctions étendues pour ajouter des fonctionnalités à un site web. ACF le fait également. Ci-dessus le filtre appelé porte le nom de**"upload_prefilter"** et va s'exécuter à chaque fois que le champ **"it_file_upload"** est rencontré.

Le deuxième filtre appelé **"upload_dir"** est quant à lui, une fonction WordPress qui va rediriger le téléversement vers un dossier personnel.

Dans l'exemple, le dossier de destination devient **"/wp-content/uploads/it"**. Tous les fichiers téléversés depuis l'éditeur et du champ ACF, finiront dans ce nouveau dossier.

Le lien de la pièce jointe

Maintenant que l'interface pour téléverser un fichier est prête, que celui-ci est autorisé et que sa destination a été modifiée, il reste à effectuer le chemin inverse pour que le visiteur puisse télécharger la pièce jointe. Le principe reste le même que pour celui du lien**"Imprimer"**. Le widget Elementor utilisé ici est la liste d'icônes. Celui-ci permet de créer des liens URL, avec texte et image. Mais surtout, ce lien se doit d'être dynamique, car il est différent à chaque publication. La solution retenue ici, est le**code court**.



Les paramètres du widget

Le code court

Le code court appelé ici par le lien **"Pièces jointes"** porte le nom de **"IT_Attachment_Page_Link"**, qu'il faut placer entre crochets.

Une fois le widget configuré avec le nom du code court, il faut ajouter celui-ci au thème enfant. Pour réaliser ceci, il faut créer un nouveau fichier **PHP** et y insérer le code ci-dessous.

```
// Get the permalink of the IT attachment page
function it_attachment_page_link () {

  global $post;

  // Retrieve all attachments for the current post
  $attachments = get_posts(array(
    'post_type'      => 'attachment',
    'posts_per_page' => -1,
    'post_parent'    => $post->ID,
    'post_status'    => 'inherit'
  ));
  // If there is an attached file corresponding to the post, return permalink of the attachment page
  foreach ($attachments as $attachment) {
    if ($attachment->guid == get_field('it_file_upload', $post->ID)['url']) {
      return get_permalink($attachment->ID);
    }
  }
}add_shortcode ('IT_Attachment_Page_Link','it_attachment_page_link');
```

Le code ci-dessus est relativement simple.

- Premièrement, on utilise une variable globale **"$post"**, qui représente la publication.
- On va ensuite créer une requête **WP_Query** pour récupérer tous les attachements de la publication en fonction de l'ID de la publication.
- Enfin, on compare l'URL de l'attachement avec celui du champ ACF **"it_file_upload"**. Si les deux URL correspondent, c'est que la publication a bien une pièce jointe.

> Pourquoi comparer les deux URL ? Ce contrôle a pour but de bien récupérer le bon URL de la pièce jointe. En effet, il est tout à fait possible que la publication possède plusieurs attachements, dont des images mis en avant, par exemple.

Le code court va retourner l'URL de la pièce jointe au widget **"Liste d'icône"** et ainsi créer le lien de téléchargement.

Afficher le lien de téléchargement

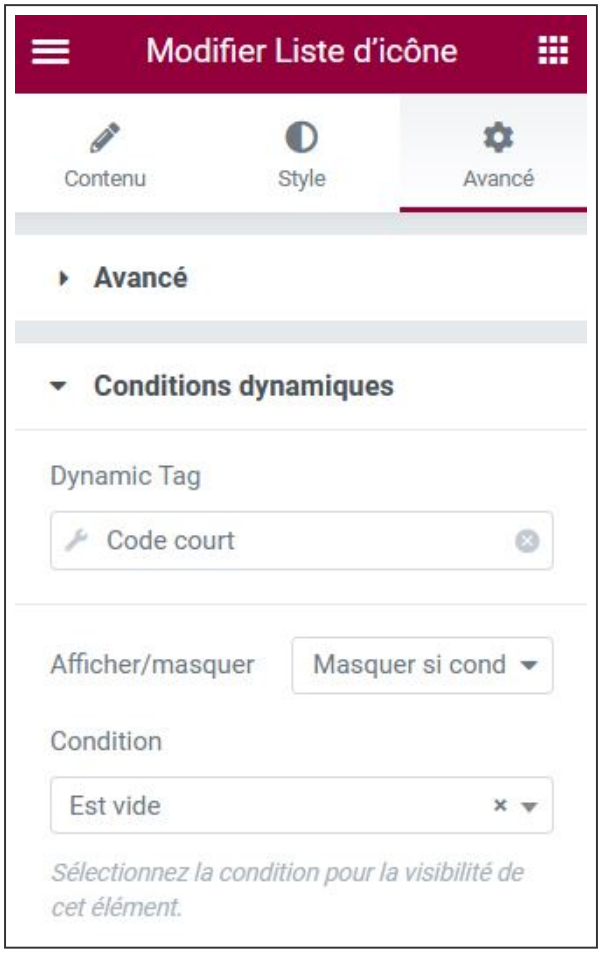
Le lien s'affiche maintenant dans la publication et son URL est automatiquement ajouté en fonction de l'emplacement de la pièce jointe. Mais que se passe-t-il si la publication ne contient pas de pièce jointe ? La réponse est évidente, le lien va s'afficher quand même, mais il ne pourra pas être actionné.

Malheureusement, Elementor n'a pas prévu, et c'est bien dommage, de pouvoir afficher ou cacher un widget en fonction de son état. Mais qu'à cela ne tienne, d'autres l'ont fait. Le plugin**"Dynamic Conditions"** réalise ce tour de magie. Il est certainement aussi possible de le réaliser en Javascript.

Le plugin Dynamic Conditions

Celui-ci s'intègre avec Elementor en créant un onglet **"Conditions dynamiques"**. Le principe en est relativement simple. On compare un champ avec un état ou une chaîne de texte et celui-ci est affiché ou masqué en fonction du résultat.

Dans l'exemple ci-dessous, le code court est réutilisé (c'est le même qui a été configuré pour le lien URL). Son contenu est comparé à **"l'état vide"**. Si le cas est positif, alors le lien **"pièce jointe"** est masqué.



Les paramètres du plugin "Dynamic conditions"

La limite des possibilités ne s'arrête pas ici. Il est tout à fait possible de comparer n'importe quel champ dynamique avec un autre. Ce plugin est bien pratique et c'est vraiment étonnant qu'Elementor n'a pas intégré cette fonctionnalité nativement dans son constructeur de pages.

Téléchargement de la pièce jointe

La parenthèse sur le plugin **"Dynamic Conditions"** refermée, il ne reste plus qu'à effectuer le téléchargement. Pour ceci, il faut maîtriser quelques notions de WordPress et de HTML.

- La première est une question d'infrastructure liée à WordPress. Dans la base de données de WordPress, toutes les publications sont enregistrées dans une table qui se nomme **"wp_posts"**. Les publications ont un type qui leurs est associé. Par défaut ce type est justement **"posts"** ou le nom du type personnalisé. Dans l'exemple de ce site, toutes les publications du blog informatique porte

le type **"it"**. Mais ce qui est vraiment intéressant, c'est que les pièces jointes ou attachements porte aussi un type. Et celui-ci est de type **"attachment"** (sans e, c'est en anglais). En résumé, tous les attachements sont des publications à part entière, porte un ID, et sont enregistrés dans la même table **"posts"** que les publications normales.

- La deuxième découle automatiquement de la première. Quand on clique sur le lien d'une pièce jointe, son contenu s'affiche dans une nouvelle page, comme si celle-ci était une autre publication.

Une fois ces notions connues, on peut facilement en déduire qu'il va falloir trouver une astuce pour rediriger le lien URL de la pièce jointe, non pas, vers une nouvelle page, comme c'est le cas par défaut, mais vers le navigateur, pour que celui-ci propose de le télécharger.

La redirection d'un modèle

WordPress propose un crochet qui porte le nom de **"template_redirect"**. Celui-ci s'exécute à chaque fois qu'un nouveau modèle est affiché. Plus simplement, à chaque fois qu'une publication, une page, un attachement, un média est affiché, WordPress exécute le crochet **"template_redirect"**.

On peut alors prendre la main et modifier le cours des choses en téléchargeant la pièce jointe, plutôt que de l'afficher. En voici le code :

```
// Check if the page is an attachment and force download it
function download_it_attachment_page() {
  global $post;
  // Check if attachment page
  if (is_attachment()) {
    // Get custom post type of attachment
    $post_type = get_post($post->post_parent)->post_type;
    // Check if attachment is of type "it"
    if ($post_type == 'it') {

      // Get attachment filepath
      $filename = get_attached_file($post->ID);
      // Get the mime type of the attachment
      $mime_type = $post->post_mime_type;
      // Discard all buffers before reading file
      while (ob_get_level()) {
        ob_end_clean();
      }

      header("Content-Type: " . $mime_type, true, 200);
      header("Content-Transfer-Encoding: Binary");
      header("Cache-Control: must-revalidate, post-check=0, pre-check=0");
      header("Pragma: no-cache");
      header("Expires: 0");
      header("Content-Length: " . filesize($filename));
      header("Content-Disposition: attachment; filename=" . basename($filename));
      readfile($filename);
      exit();
    }
  }
}add_action('template_redirect','download_it_attachment_page');
```

Le code ci-dessus peut s'expliquer ainsi :

- Une variable globale **"$post"** est utilisée, pour représenter l'attachement. En effet, WordPress est déjà passé sur la page de la pièce jointe, sans pour autant l'avoir encore affichée.
- Une condition vérifie que le traitement est bien à l'attention d'un attachement.
- Une autre condition vérifie que l'attachement est bien de type **"it"**. Pour ceci, on récupère le type de la publication.
- Le nom de l'attachement est ensuite récupéré, ainsi que son type **MIME**.
- Enfin, une nouvelle entête (**header en anglais**) est créé, afin de donner l'ordre au navigateur de télécharger la pièce jointe.

> Une petite note sur les deux fonctions **“ob_get_level”** et **“ob_end_clean”**. La première indique le nombre de tampons mémoires actuellement actifs. La deuxième permet de supprimer un tampon mémoire (**buffer en anglais**) et de ne pas l’envoyer au navigateur. Il est indispensable de supprimer tous les buffers ouvert avant d’en créer de nouveaux, ceci afin que le fichier téléchargé corresponde à l’original. Sans cette précaution, le fichier pourrait être modifié et devenir inutilisable.

Le mot de la fin

Ici, encore, **Elementor et WordPress** se suffisent à eux-même pour créer de toutes pièces un système de téléchargement de pièces jointes liées aux publications. A l’exception de l’affichage du lien en fonction de son état géré par un plugin tiers (qui pourrait très bien être remplacé par quelques lignes de Javascript), les fonctions de WordPress sont des merveilles de simplicité autant pour rediriger vers un modèle (**grâce à l’action template_redirect**) ou d’afficher un URL sur un lien Elementor (**grâce aux codes courts**). En cherchant un peu, nul besoin d’installer multitudes de plugins tiers.