

Les fenêtres modales et le système de connexion

Elementor, JS, PHP

Web Compétences : 5

Rien de mieux que de créer un système de connexion-déconnexion avec des fenêtres modales. Elementor a intégré des fenêtres modales dans sa collection qui est le compagnon idéale pour créer les interactions nécessaires avec le visiteur. Alors pourquoi s'en priver ? Suivez le guide

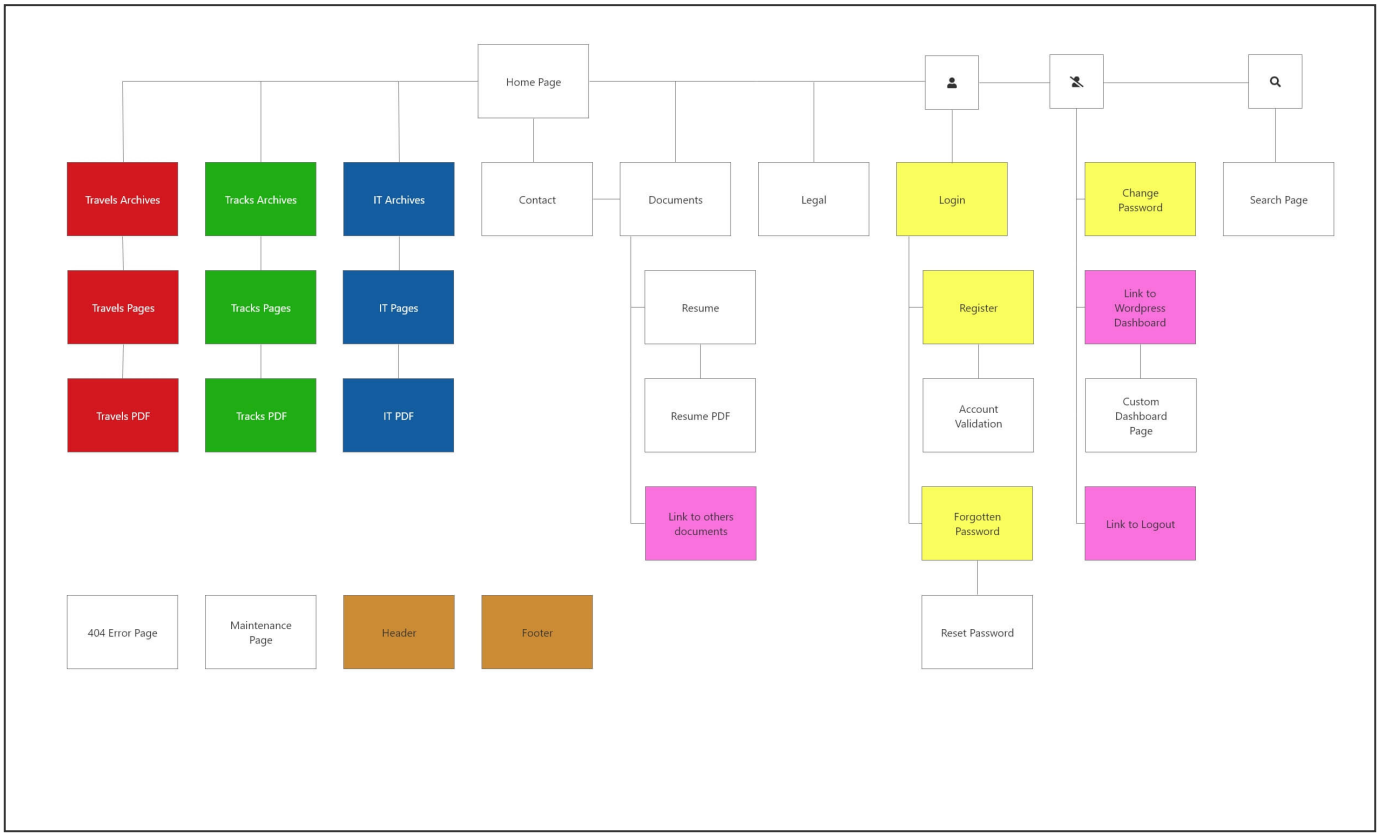
Publié jeudi 25 juin 2020, 18h27
Modifié lundi 26 août 2024, 10h05



By Olivier Paudex

Introduction

Selon le plan des pages du site, il était prévu de créer un système de connexion (login) des utilisateurs. Il n'a pas grande utilité, mais le défi à relever était très intéressant au point de vue académique. Elementor a prévu un type de modèle appelé les fenêtres modales (popup windows), qui facilite grandement la mise en place d'un tel système.



Le plan des pages du site

Bien démarrer

Si l'on se réfère au plan des pages ci-dessus, il va falloir créer quatre fenêtres modales (en jaune), deux nouvelles pages (en blanc), deux liens (en violet), ainsi que de personnaliser la console d'administration de WordPress.

Pour bien expliquer le raisonnement, il est temps de prendre un peu de recul et de réfléchir à l'interaction d'un administrateur ou d'un contributeur et des raisons de vouloir créer une suite logique de fenêtres modales ainsi qu'à la manière la plus simple et la plus logique de leur donner corps.

- La première raison bien évidente est de vouloir ouvrir le site à d'autres personnes que soi-même pour contribuer à la création de contenu.
- Une deuxième raison, est purement esthétique. Il s'agit de remplacer la vilaine fenêtre de connexion de WordPress par une autre, entièrement personnalisée.
- Une troisième raison serait de donner aux contributeurs, un moyen simple, de s'enregistrer et de modifier leur mot de passe, sans recours à l'administrateur du site.

Ceci étant dit, il faut penser à l'enchaînement des fenêtres modales pour créer le système de connexion. Voici la réflexion proposée :

- La première fenêtre qui doit s'afficher doit être celle de connexion, disponible en cliquant sur un bouton de connexion.
- Si le contributeur ne possède pas de compte utilisateur, il doit y avoir un lien pour le rediriger vers une fenêtre modale d'enregistrement de profil. Le site doit être en mesure d'envoyer un courriel à l'utilisateur pour confirmer que celui-ci est bien une personne physique avec une adresse valable.
- Si le contributeur a égaré son mot de passe, il doit y avoir un lien pour le rediriger vers une fenêtre modale permettant de le récupérer. Le site doit également être en mesure d'envoyer un courriel à l'utilisateur pour confirmer que celui-ci a bien demandé la mise à zéro de son mot de passe.
- Une fois le contributeur connecté, il faut le lui faire savoir en indiquant clairement son nom dans un menu personnalisé.
- Le contributeur doit pouvoir modifier son mot de passe, action qui n'a rien à voir avec l'oubli de son mot de passe. Ce qui nous conduit vers la création d'une quatrième fenêtre modale.
- Le contributeur connecté doit pouvoir accéder à une console d'administration de WordPress simplifié et personnalisé.
- Le contributeur doit pouvoir se déconnecter.

Étendre l'entête avec un menu de connexion

La première étape consiste à créer un bouton de connexion dans l'entête. Si l'on considère les points de réflexion ci-dessus, il va très rapidement ressortir qu'il va falloir créer un deuxième bouton pour la déconnexion. De plus, le bouton de déconnexion doit appeler un menu personnalisé pour que le contributeur puisse entre autres modifier son mot de passe et accéder à la console de WordPress. La solution sera donc meilleur d'utiliser un menu en lieu et place d'un bouton et de permuter celui-ci en fonction de l'état de la connexion ou de la déconnexion.

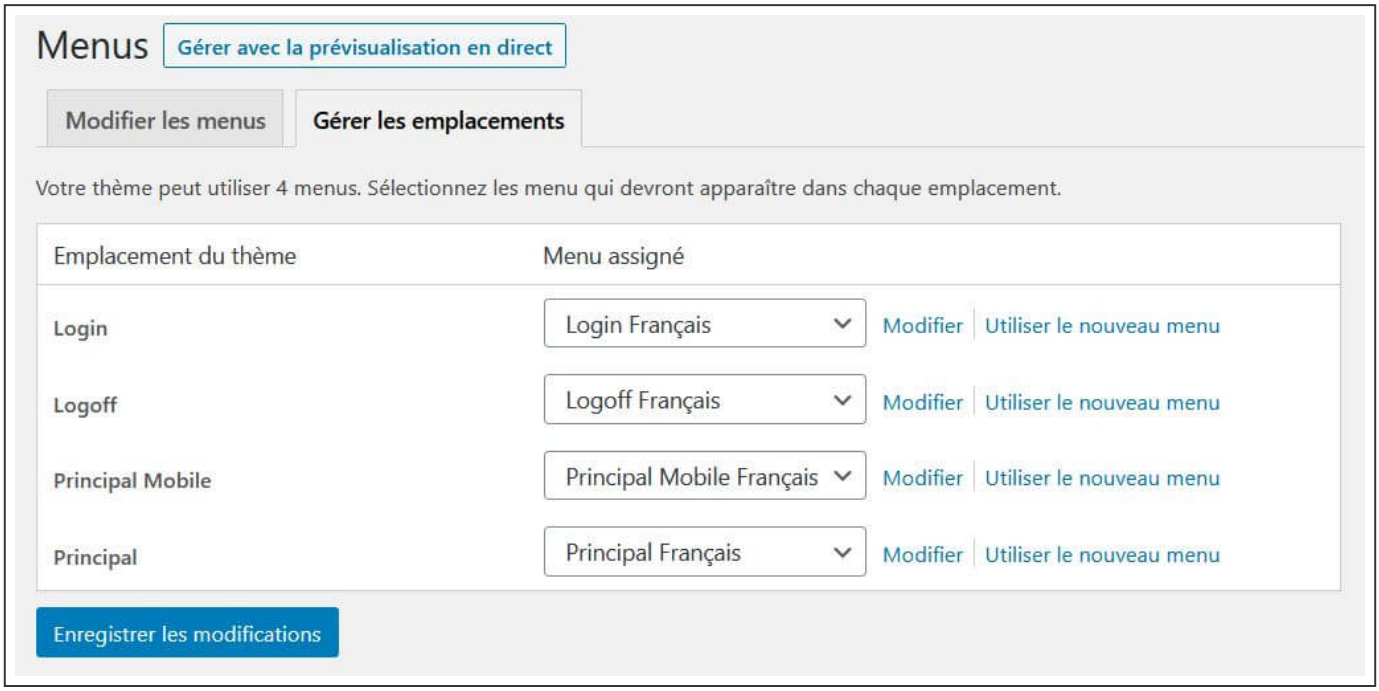
Création du menu de connexion

Les menus sont créés directement dans WordPress sous l'onglet **"Apparence"**. Le menu de connexion n'ayant rien à voir avec le menu principale du site, il faut créer un deuxième menu totalement indépendant. De même pour le menu de déconnexion. Pour créer des menus, il faut leur attribuer un emplacement. Cet emplacement est dépendant du thème utilisé. Avec le thème **"Hello"**, Elementor n'a prévu qu'un seul emplacement. Heureusement, ces emplacements peuvent être ajoutés par code PHP.

Commencez donc par créer un nouveau fichier PHP, et nommé le **"menu.php"**. Dans celui-ci, créez une fonction pour ajouter de nouveaux emplacements de menus au thème. Ce code permet de placer trois menus, l'emplacement du menu principale étant créé par défaut.

```
// Register the menus
function register_menus() {
  register_nav_menus (
    array (
      'login'                => ('Login'),
      'logoff'               => ('Logoff')
    )
  );
}add_action ('after_setup_theme', 'register_menus');
```

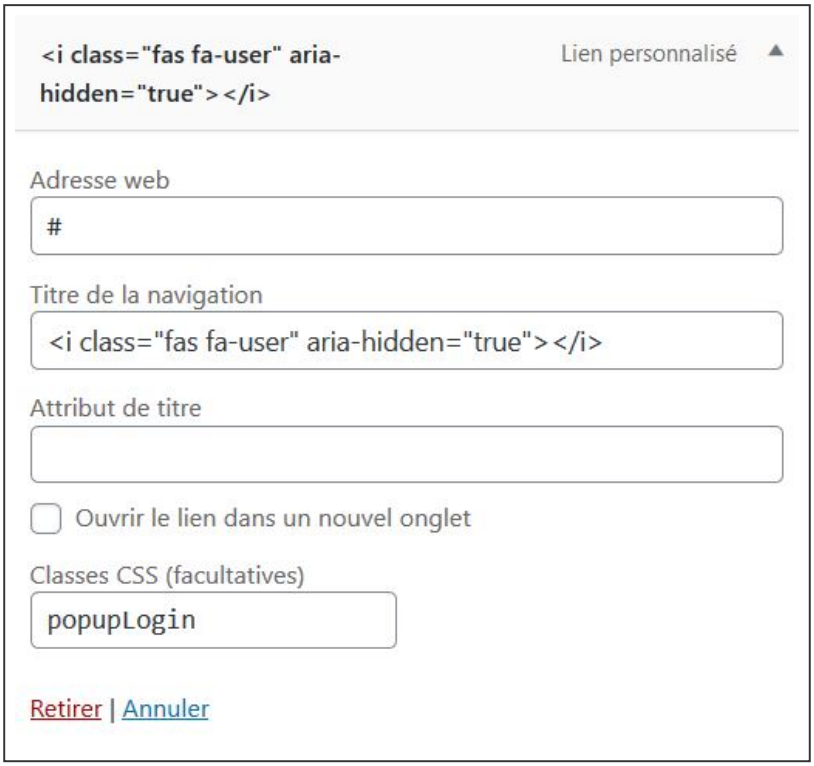
Les emplacements ajoutés par code s'affichent dans la fenêtre de configuration du menu. Dans l'exemple ci-dessous. On peut voir le mot **"Français"**, à côté du menu. Celui-ci est ajouté automatiquement par le plugin **"Polylang"** quand le site est traduit en plusieurs langues. Le sujet est traité dans le chapitre concernant le multilinguisme. Le menu **"Principal Mobile"** est un menu spécialement créé pour les écrans de téléphones portables. Quant au menu **"Principal"**, celui-ci est créé par défaut.



La configuration du menu

Une fois ceci réalisé, rendez-vous dans l'onglet **"Apparence"** de WordPress pour créer le menu de connexion. Insérez le code HTML pour afficher l'icône et un dièse (#) pour l'adresse URL. Affichez les classes CSS depuis les options de l'écran si celles-ci ne s'affichent pas et entrez la classe**"popupLogin"**. C'est cette classe qui va permettre d'ouvrir la fenêtre modale.

Enfin, n'oubliez pas d'affecter le menu à l'emplacement login.



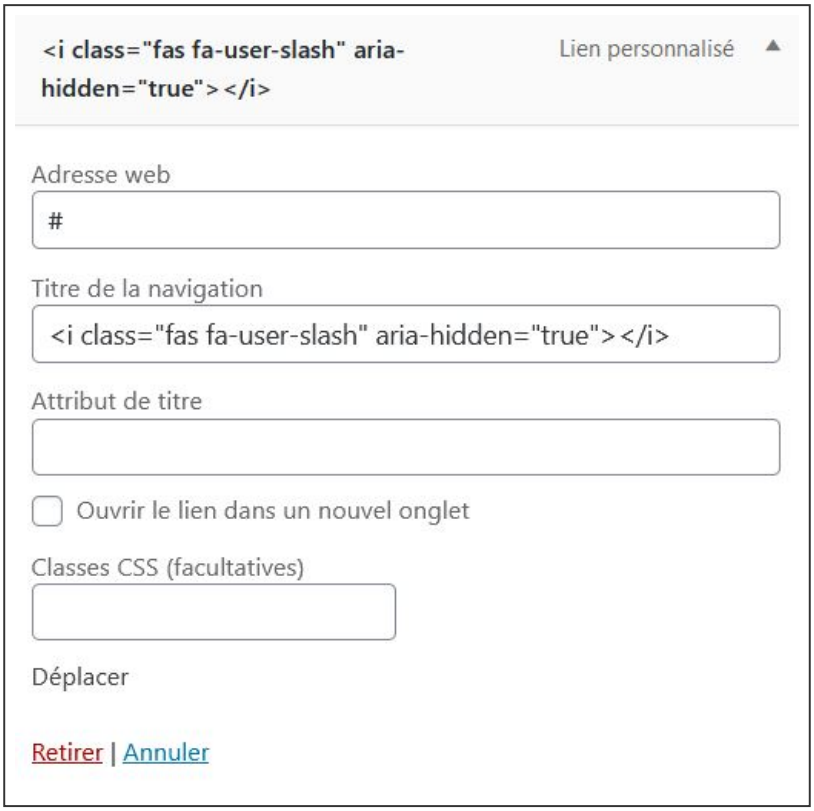
Le menu login

Création du menu de déconnexion

Le menu de déconnexion est un peu plus complexe, puisqu'il est constitué de l'icône et d'un sous-menu comprenant les options suivantes :

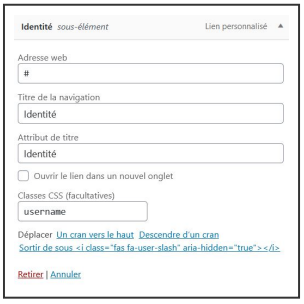
- Identité de l'utilisateur
- Mon tableau de bord
- Changer votre mot de passe
- Déconnexion

Le menu de déconnexion est semblable au menu de connexion, sauf qu'il ne possède pas de classe CSS, parce qu'il n'appelle aucune fenêtre modale.



Le menu logoff

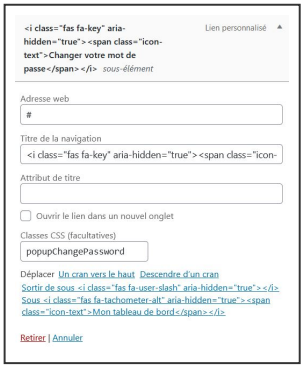
- Le menu **"Identité"** possède un attribut de titre HTML du même nom et une classe CSS appelé **"username"**. Il va afficher le nom du contributeur actuellement connecté.
- Le menu **"Mon tableau de bord"** n'est rien d'autre qu'un lien vers la console **"Wordpress"**, appelé **"wp-admin"**. Le paramètre **"?lang=fr"** est lié au multilinguisme.
- Le menu **"Changer votre mot de passe"** possède une classe CSS appelé **"popupChangePassword"**. Il va appeler une fenêtre modale.
- Le menu **"Déconnexion"** possède un attribut de titre HTML du même nom. Il va permettre à l'utilisateur de se déconnecter sans passer par l'écran de déconnexion standard de WordPress.



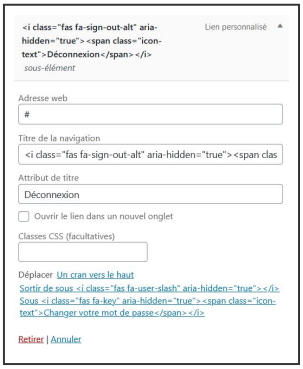
Le menu Identité



Le menu "Mon tableau de bord"



Le menu "Changer votre mot de passe"



Le menu "Déconnexion"

Basculement entre les menus de connexion et de déconnexion

Pour basculer entre les deux modes, il va falloir entrer un bout de code en PHP. Celui-ci s'appuie sur le filtre **"wp_nav_menu_args"**. Il va afficher le menu de connexion ou de déconnexion en fonction de l'état de l'utilisateur, grâce à la fonction **"is_user_logged_in"** et au nom du menu, grâce à la fonction **"wp_get_nav_menu_object"**.

```
// Switch between login and logoff menu
function menu_loginout($args = '') {
  // Get object menu slug
  $menu_obj = wp_get_nav_menu_object($args['menu']);
  $menu_slug = $menu_obj->slug;
  // If user is logged in and menu is not primary
  if (is_user_logged_in() && $menu_slug != 'principal' && $menu_slug != 'principal-mobile') {
    $args['menu'] = 'logoff';
  // If user is logged off and menu is not primary
  } elseif (!is_user_logged_in() && $menu_slug != 'principal' && $menu_slug != 'principal-mobi
    $args['menu'] = 'login';
  }
  return $args;
}add_filter ('wp_nav_menu_args', 'menu_loginout');
```

La fonction "Déconnexion" et "Identité"

Les deux fonctions se basent sur le filtre **"wp_nav_menu_objects"**. Celui-ci permet d'agir sur les objets d'un menu spécifique. Dans le cas du menu **"Logoff"**, on différencie les deux objets grâce à leurs attributs de titre, **"Déconnexion"** et **"Identité"**.

- La fonction **"wp_logout_url"**, permet de rediriger l'utilisateur sur la page d'accueil après la déconnexion.
- La fonction **"get_current_user_name"** est une fonction propriétaire décrite ci-dessous.

```
// Update Logoff menu
function update_logoff_items_menu ($items,$args) {

  // If menu is 'logoff'
  if (!is_admin() && $args->menu == 'logoff') {
    // Get current URL
    global $wp;
    $current_url = home_url($wp->request);
    foreach ($items as $item) {
      // Update nonce of the logout link
      if ($item->attr_title == 'Déconnexion') {
        $item->url = esc_url(wp_logout_url($current_url));
      }
      // Update username
      if ($item->attr_title == 'Identité') {
        $item->title = get_current_user_name();
      }
    }
  }
  return $items;
}add_filter ('wp_nav_menu_objects','update_logoff_items_menu',10,2);
```

La fonction propriétaire "get_current_user_name"

Voici une des nombreuses fonctions écrites pour la réalisation de ce site. Cette fonction permet d'afficher les 30 premiers caractères du prénom et du nom de l'utilisateur.

```
// Get the current user name and display the first 30 characters
function get_current_user_name() {
   $current_user = wp_get_current_user();
   if (!$current_user->exists()) {
     return;
   }
   return substr($current_user->user_firstname . ' ' . $current_user->user_lastname,0,30);}
```

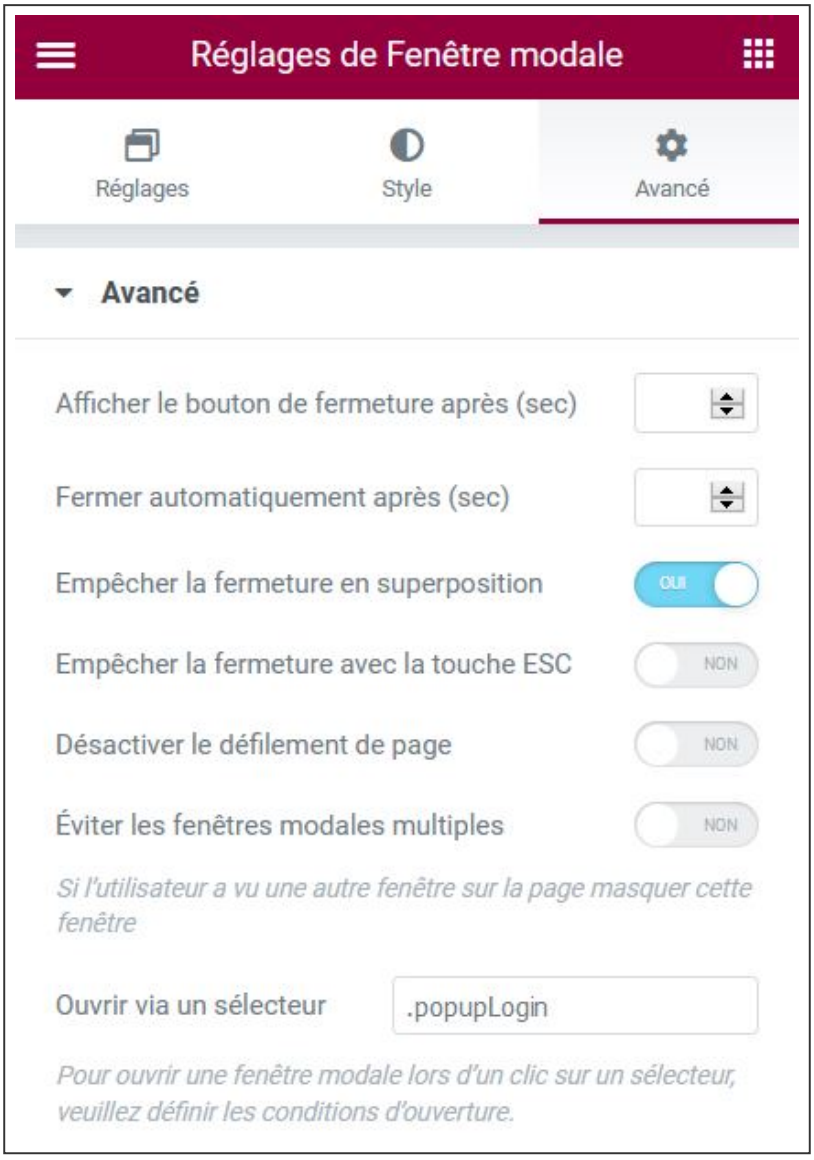
Les fenêtres modales

Maintenant que le système de menu est en place, il est temps de créer les fenêtres modales, qui sont au nombre de quatre.

- La fenêtre de connexion
- La fenêtre d'enregistrement
- La fenêtre d'oubli de mot de passe
- La fenêtre de changement de mot de passe

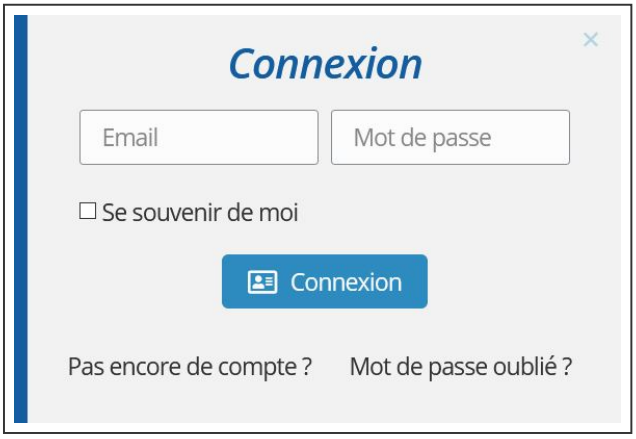
Création de la fenêtre modale de connexion

Créer une fenêtre modale avec Elementor se fait comme tous les autres modèles. Les réglages généraux se font à l'aide de la petite roue dentée se trouvant en bas à gauche de la fenêtre. L'option intéressante se trouve dans l'onglet avancé et se nomme **"Ouvrir via un sélecteur"**. C'est lui qui va lier le menu et la fenêtre modale. Pour la fenêtre de connexion, celui-ci porte le nom de**".popupLogin"**. Comme il s'agit d'une classe CSS, on y ajoute un point devant le nom de la classe.

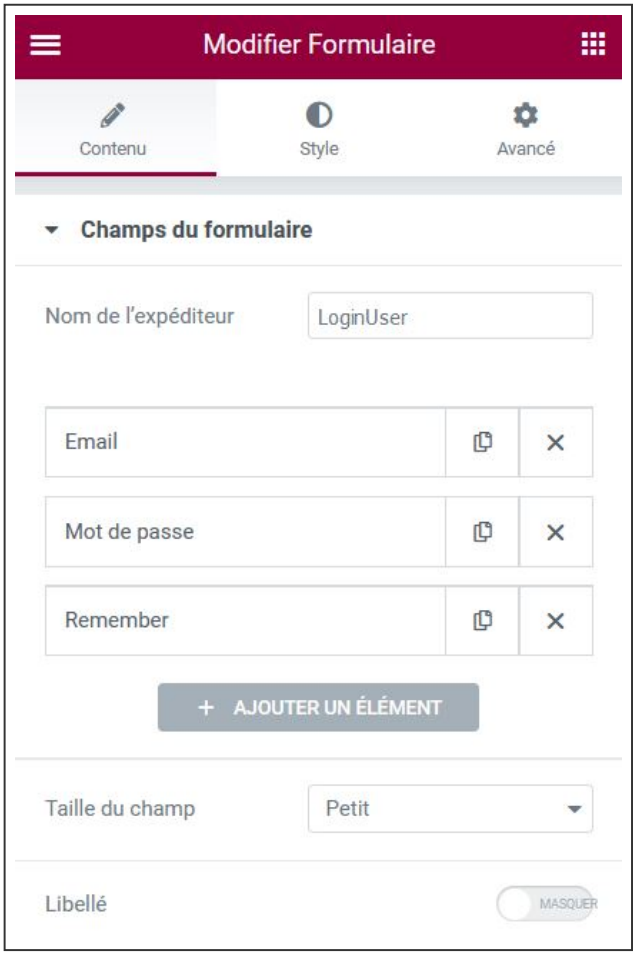


Les réglages de la fenêtre modale de connexion

La fenêtre est construite sur la base d'un widget de formulaire et de deux liens qui renvoie chacun vers une autre fenêtre modale. Le formulaire comporte trois champs, email (qui fait office de nom de connexion), mot de passe, une case à cocher pour que le site web se souvienne de l'utilisateur, ainsi que d'un bouton de validation. Le formulaire porte un nom **"LoginUser"**, qui va servir à lier le formulaire et son action de connexion.



La fenêtre modale de connexion



Le formulaire de la fenêtre modale de connexion

Validation du formulaire de connexion

La validation du formulaire se fait par code. Elementor a créé plusieurs actions pour travailler avec les formulaires. L'une d'entre-elles porte le nom de **"new_record"**. Elle permet d'ajouter un gestionnaire de formulaire personnalisé, comme un message d'erreur ou un état.

Le code ci-dessous va vérifier que l'appelle se fait bien depuis le formulaire**"LoginUser"**, puis récupère les différents champs, et enfin, connecte l'utilisateur. Si l'utilisateur et/ou le mot de passe n'existe pas, un message d'erreur ainsi que l'état du formulaire de connexion sont ajoutés. L'action s'arrête alors ici et le message d'erreur est affiché. Dans le cas contraire, un cookie est conservé pour identifier l'utilisateur et celui-ci est connecté.

Ce n'est certainement pas la seule manière de connecter un utilisateur, mais cette action à l'avantage de pouvoir ajouter un message d'erreur personnalisé.

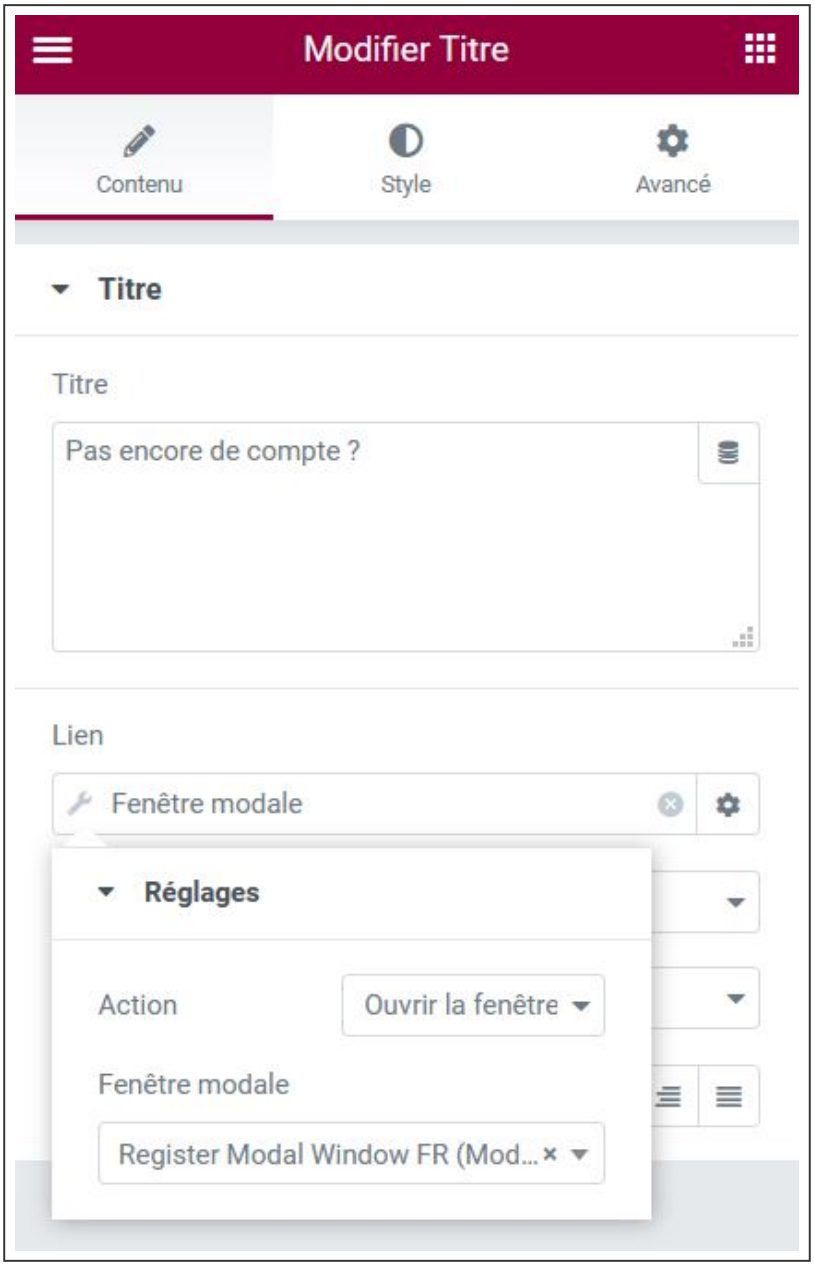
```
<?php
  /**
  * Plugin Name: Login_Form
  * Description: User login for the Elementor form widget
  * Author: Olivier Paudex
  * Author Web Site: https://www.fuyens.ch
  */
  function login_user_form($record,$ajax_handler)
  {
    // Check the Elementor form name
   $form_name = $record->get_form_settings('form_name');
   if ('LoginUser' !== $form_name) {
      return;
    }
    // Get the form datas
    $form_data = $record->get('fields');
    $creds = array(
      'user_login'        => $form_data['email']['value'],
      'user_password'     => $form_data['password']['value'],
      'remember'          => $form_data['remember']['value']
    );

    // Signon the user
    $user = wp_signon ($creds, false);
    // If error
    if (is_wp_error($user)) {
      // Default error message
      $error_message = pl__("L'utilisateur n'existe pas ou le mot de passe est invalide");

      // Display error message
      $ajax_handler->add_error_message($error_message);
      $ajax_handler->is_success = false;
      return;
    }
    // Set the cookie for the user session
    wp_set_auth_cookie ($user->ID);
  }
  add_action('elementor_pro/forms/new_record','login_user_form',10,2);?>
```

Liaison avec la fenêtre modale d'enregistrement

Le formulaire d'enregistrement s'actionne avec le lien **"Pas encore de compte ?"**, sur la fenêtre de connexion. On peut ajouter une action sur un objet de type **"Titre"**. Ici, le lien est de type **"fenêtre modale"**, l'action est d'ouvrir une fenêtre modale et celle-ci doit être la fenêtre d'enregistrement.



Liaison d'un widget Titre avec une fenêtre modale

Création de la fenêtre modale d'enregistrement

La fenêtre modale d'enregistrement ne diffère pas beaucoup de celle de connexion, à la grande différence que celle-ci doit envoyer un message au nouvel utilisateur dans sa boîte de messagerie pour inviter celui-ci à donner son accord de création de compte. Ceci protège également des systèmes frauduleux qui voudraient créer de faux comptes.

A la création d'un nouvel utilisateur dans la base de données de WordPress, il va falloir l'accompagner de deux meta-données qui porte les noms de **"email_verification_key"** et **"email_not_verified"**. La procédure d'enregistrement doit donc créer un lien contenant l'URL du site, accompagné de ces clés de vérification ainsi que d'autres données concernant l'utilisateur. Ce lien est ensuite envoyé à l'utilisateur sous forme de message électronique.

```
function set_email_verification_status($userID) {
    // Add metadata 'email_not_verified', when user register
    update_user_meta($userID, 'email_not_verified', true);
    update_user_meta($userID, 'email_verification_key', wp_generate_password(12,false,false));
  }  add_action('user_register', 'set_email_verification_status');
```

A l'authentification de l'utilisateur, si la meta-donnée **"email_not_verified"** est vrai, alors le compte est bloqué.

```
function get_email_verification_status($user) {
  // Add error if account email is not verified
  if (get_user_meta($user->ID, 'email_not_verified', true) == true) {
    $user = new WP_Error ('inactive_account', pl__('Inactive account'));
  }
  return $user;
}add_filter('wp_authenticate_user', 'get_email_verification_status',10,2);
```

L'extrait de code ci-dessous envoie un message électronique à l'utilisateur comprenant le lien à cliquer pour valider la création du compte.

> La fonction **"pll_translate_string"** et son raccourci **"pl__"** permet d'écrire les messages en plusieurs langues. Ceci est réalisé avec le plugin **Polylang**. Les messages écrit ici en anglais sont automatiquement traduit en français si l'utilisateur parle le français ou restent inchangés si l'utilisateur parle l'anglais.

```
function register_user_form($record,$ajax_handler)
{
  // Get the verification email key from user metadata and create the link
  $email_verification_key = get_user_meta($user_id, 'email_verification_key', true);
  $link = wp_login_url() . esc_url_raw('?action=validation&key=' . $email_verification_key . '&
  $rplink = '<a href="' . $link . '">' . $link . '</a>';
  // Email validation
  if (function_exists('pll_translate_string')) {
    $message = pll_translate_string('Hello', $language) . ' ' . $first_name . ' ' . $last_name
    $message .= pll_translate_string('A request for registration on', $language) . ' ' . get_bl
    $message .= pll_translate_string('To write a contribution to this site, select My Dashboard
    $message .= pll_translate_string('All new contribution will be subject to proofreading befo
    $message .= pll_translate_string('If you are not the author of this request, please disrega
    $message .= pll_translate_string('Please click on the link below to validate your registrat
    $message .= $rplink . '<br><br>';
    $message .= pll_translate_string('The administrator of', $language) . ' ' . get_bloginfo('n
  }
  // Add a subjet to the mail
  $subject = pll_translate_string('Inscription on', $language) . ' ' . get_bloginfo('name');
  // Add Text/HTML filter
  add_filter('wp_mail_content_type', function($content_type) {return 'text/html';});
  // Send mail
  wp_mail($email, $subject, $message);
  // Reset content-type filter to avoid conflicts
  remove_filter('wp_mail_content_type', 'set_html_content_type');
}add_action ('elementor_pro/forms/new_record','register_user_form',10,2);
```

Redirection vers la page de validation du compte

Le lien créé ci-dessus pour inviter l'utilisateur à valider son compte doit bien atterrir quelque part. En l'occurrence, sur la page de validation. WordPress est fourni avec de nombreuses fonctions dont **"wp_login_url"** qui redirige automatiquement sur la page de connexion de celui-ci, nommée **"wp-login.php"**.

Pour des raisons esthétiques, ce site fourni sa propre page de validation, appelée **"account-validation"**. Le lien va donc devoir automatiquement rediriger le navigateur vers cette page personnalisée.

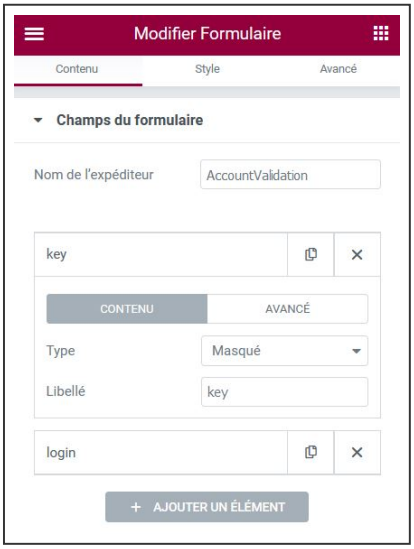
Le code ci-dessous va récupérer l'URL et si la destination est bien **"wp-login.php"**, suivi du paramètre **"action=validation"**, alors le navigateur va être rediriger vers la page personnalisée traduite en deux langues, auquel on va ajouter le login du compte et la clé de validation.

```
// Redirect to account validation page
function redirect_to_account_validation_page() {

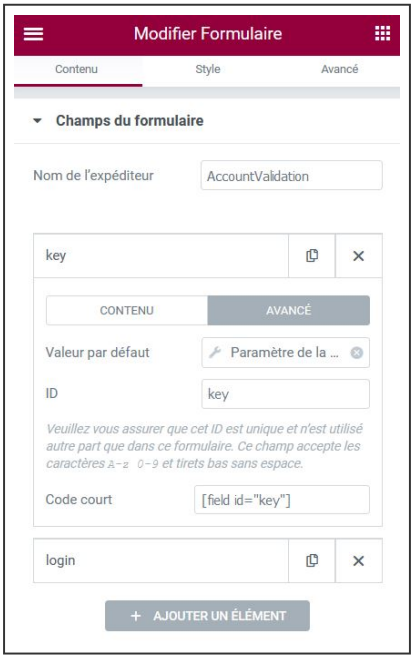
  // Access to the global variable '$pagenow'
  global $pagenow;
  // Check and store the arguments
  $action = (isset($_GET['action'])) ? $_GET['action'] : '';
  $key = (isset($_GET['key'])) ? $_GET['key'] : '';
  $login = (isset($_GET['login'])) ? $_GET['login'] : '';
  $lang = (isset($_GET['lang'])) ? $_GET['lang'] : '';
  // Check if we're on the WP default login page, and ensure the action is 'validation'
  if ($pagenow == 'wp-login.php' && ($action == 'validation')) {
      // Redirect to french page
      if ($lang == "fr_FR") {
        $page = site_url() . '/validation-accompte/';
      }
      // Redirect to english page
      if ($lang == "en_US") {
        $page = site_url() . '/account-validation/';
      }
      // Redirect to the account validation page
      wp_redirect(esc_url_raw(add_query_arg(array('key' => $key, 'login' => $login),$page)));
      // Stop execution to prevent the page loading for any reason
      exit();
  }
}add_action('after_setup_theme', 'redirect_to_account_validation_page');
```

La page de validation du compte

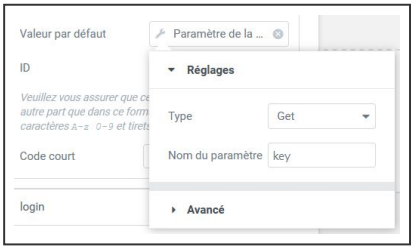
La page de validation du compte comporte un formulaire nommé **"AccountValidation"**, qui contient lui-même deux champs cachés, **"key"** et **"login"**. Dans l'onglet avancé des champs du formulaire, il est possible de donner comme valeur par défaut, un paramètre de requête de type**"Get"**. Ainsi les champs cachés seront automatiquement remplis avec les données de l'URL.



Les champs sont de type masqués



La valeur par défaut prend les paramètres de la requête



Les paramètres sont de type "Get"

Une fois le formulaire validé par l'utilisateur, il suffit d'appeler une dernière action pour valider le compte et effacer la meta-donnée "**email_not_verified**", qui bloque son utilisation.

```
function account_validation_form($record,$ajax_handler) {

  // Check the Elementor form name
  $form_name = $record->get_form_settings('form_name');
  if ('AccountValidation' !== $form_name) {
      return;
  }
  // Get the form datas
  $form_data = $record->get('fields');
  // Get the hidden fields
  $key = $form_data['key']['value'];
  $login = $form_data['login']['value'];
  // Get user by email
  $user = get_user_by('email',$login);
  // If error
  if (is_wp_error($user)) {
    $ajax_handler->add_error_message(pl__('Unknown user'));
    $ajax_handler->is_success = false;
    return;
  }
  // Get email verification key
  $email_verification_key = get_user_meta($user->id, 'email_verification_key', true);

  // Compare both keys and delete user metadata
  if ($key == $email_verification_key) {
    delete_user_meta($user->id, 'email_not_verified');
    delete_user_meta($user->id, 'email_verification_key');
  } else {
    // Key error
    $ajax_handler->add_error_message(pl__('Key not valid'));
    $ajax_handler->is_success = false;
    return;
  }
}add_action ('elementor_pro/forms/new_record','account_validation_form',10,2);
```

Création de la fenêtre modale de réinitialisation du mot de passe

La troisième fenêtre modale va permettre à un utilisateur de réinitiliser son mot de passe en cas d'oubli. La procédure est un peu similaire à l'enregistrement, à savoir :

- Un lien qui déclenche l'ouverture de la fenêtre modale à partir de la fenêtre de connexion.
- L'envoi d'un message électronique invitant l'utilisateur à cliquer sur un lien pour réinitiliser son mot de passe.
- La redirection vers une page personnalisée, contenant des champs cachés, invitant l'utilisateur à entrer un nouveau mot de passe et à le valider.
- La mise à jour du mot de passe si tout se déroule correctement.

Création de la fenêtre modale de modification du mot de passe

Dernière fenêtre modale du site, celle-ci s'ouvre depuis le menu **"Déconnexion"** et réagit sur la classe nommée **"popupChangePassword"** (voir le chapitre sur la création du menu de déconnexion). Le

principe est encore plus simple que pour la réinitialisation du mot de passe, puisque l'utilisateur est déjà connecté, mais désire pour une raison qui lui est personnel, de modifier son propre mot de passe. Ici, pas besoin d'envoyer de message électronique à l'utilisateur pour vérifier son identité, ni de redirection vers une page personnalisée. La seule fonction qui doit être réalisée ici est la vérification que l'utilisateur connaisse bien son mot de passe actuel.

Voici le code :

```
function update_password_form($record,$ajax_handler) {

  // Check the Elementor form name
  $form_name = $record->get_form_settings('form_name');
  if ('PasswordChange' !== $form_name) {
    return;
  }
  // Get the form datas
  $fields = $record->get('fields');
  $old_password = $fields['oldpassword']['value'];
  $new_password = $fields['newpassword']['value'];
  $confirm_password = $fields['confirmpassword']['value'];
  // Get current user
  $current_user = wp_get_current_user();
  // Check old password
  if (!($current_user && wp_check_password($old_password, $current_user->data->user_pass, $current_user
    $ajax_handler->add_error_message(pl__('The user does not exist or the old password is invalid'));
    $ajax_handler->is_success = false;
    return;
  }
  // Check if $new password = $confirm_password
  if ($new_password !== $confirm_password) {
    $ajax_handler->add_error_message(pl__('The two new passwords do not match'));
    $ajax_handler->is_success = false;
    return;
  }
  // Reset Password
  wp_set_password($new_password, $current_user->ID);
}add_action ('elementor_pro/forms/new_record','update_password_form',10,2);
```

Rechargement des pages à la fermeture d'une fenêtre modale

Dernier point concernant les fenêtres modales. Celles-ci font l'office d'interfaces entre l'utilisateur et son navigateur, avec le serveur. Si un utilisateur veut par exemple se connecter au site, celui-ci va entrer son login et son mot de passe, puis valider la demande de connexion. La demande est alors traitée en PHP par le serveur qui va connecter l'utilisateur à WordPress. Mais l'utilisateur n'a aucun retour dans son navigateur concernant sa connection. La solution est bien entendu de recharger la page. Pour réaliser ceci, Elementor à créé un événement qui se déclenche à la fermeture d'une fenêtre modale et qui porte le nom **"submit_success"**.

Comme il s'agit d'un événement qui se déclenche du côté du navigateur, il va sans dire que celui-ci doit être écrit en Javascript. Et comme pour tout code JS, celui-ci doit-être préalablement mis en queue.

Voici le code :

```
/**
* Plugin Name: RefreshPage
* Description: Reload a page after closing an elementor popup
* Author: Olivier Paudex
* Author Web Site: https://www.fuyens.ch
*/
// Refresh page on Elementor popup submit
jQuery(document).ready(function($) {
  $(document).on('submit_success', function(event, response) {
    location.reload();
  });
});
```

Le mot de la fin

Voici un chapitre sur la création d'un système de connexion, inscription, réinitililsation du mot de passe et mise à jour relativement complet, entièrement réalisé avec Elementor et ses fonctions de gestion des formulaires. C'est l'un des points forts du constructeur de page que l'on peut facilement inscrire du côté de ses avantages, **il est capable de créer des fenêtres modales et des formulaires** et tout ceci de manière native, sans ajout de plugins supplémentaires.

Le vrai problème d'un système de connexion à WordPress vient de sa lenteur. En effet, pour optimiser un site web, le premier plugin à installer est un système de mise en cache des pages. Dès le moment où un utilisateur est connecté à WordPress, le cache ne tient pas compte de l'utilisateur connecté et le système de connexion ne fonctionne pas correctement. La plupart des plugins de cache ont une option de

désactivation du cache pour les utilisateurs connectés, ce qui revient à ne pas installer de plugin de cache du tout. Les pages étant rechargées à chaque ouverture, le temps d'attente augmente.

Ce site utilise le plugin **WP-Optimize**, qui propose bel et bien l'option de ne pas servir des pages mise en cache aux utilisateurs connectés. Ceci étant dit, vérifier bien le plugin de cache du site avant de créer un système de connexion.



