

Códigos para mejorar el backoffice de WordPress



Todo aquel que haya trabajado alguna vez con el CMS WordPress conocerá o habrá utilizado su backoffice o zona de administración. Es la parte que permite gestionar todo el contenido que aparece en la página web, así como instalar nuevos plugins y temas que utilizar. Por defecto, WordPress ofrece una serie de funcionalidades en su administración, pero éstas se pueden modificar mediante código para añadir nuestras características. A lo largo de nuestro WhitePaper veremos algunos fragmentos de código que nos permitirán añadir ciertas funcionalidades a nuestro **WordPress**.

Antes de empezar con el listado de nuevas funcionalidades hay que decir que todo el código que veamos a lo largo de este libro blanco habrá que colocarlo dentro del archivo “**functions.php**” que está dentro de la carpeta del tema que estemos utilizando. Por norma general, éste se encontrará en la siguiente ruta:

`https://midominio.com/wp-content/themes/mitheme/functions.php`

Este fichero es vital para el buen funcionamiento de WordPress, por lo que recomendamos que antes de realizar algún cambio en él, se realice una **copia de seguridad** a la que poder acudir en caso de cualquier problema.

También hay que decir que muchas de estas acciones se pueden realizar mediante el uso de plugins, pero un uso excesivo de estos añadidos provocará un aumento de la carga del servidor y depender de terceros.

Estar logueado durante más tiempo

Por defecto WordPress establece un tiempo máximo de logueo en su administración. Pasado ese tiempo sin realizar ninguna actividad, la sesión se cierra y hay que volver a loguearse con nuestras credenciales. Si quieres aumentar este tiempo, lo podemos hacer con el siguiente código:

```
add_filter( 'auth_cookie_expiration', 'tiemp_estar_logueado_admin' );  
  
function tiemp_estar_logueado_admin( $expire ) {  
    return 15552000; // 6 meses en segundos  
}
```

Lo que devuelve esta función es el tiempo, en segundos, que queremos que permanezca abierta la sesión en la administración. En nuestro ejemplo hemos calculado para 6 meses.

Eliminar opciones del menú



A la hora de diseñar una **página web** para un cliente que no tiene mucho conocimiento del uso de WordPress, puede ser buena práctica eliminar ciertas opciones del menú del backoffice que trae por defecto el CMS. De esta forma podemos evitar que toque cosas que puede hacer que deje de funcionar el sitio.

```
function remove_menus () {
```

```
global $menu;
```

```
    $restricted = array(__('Appearance'), __('Tools'), __('Users'), __('Settings'), __('Plugins'));
```

```
    end ($menu);
```

```
    while (prev($menu)){
```

```
        $value = explode(' ', $menu[key($menu)][0]);
```

```
        if(in_array($value[0] != NULL?$value[0]:"", $restricted)){unset($menu[key($menu)]);}
```

```
    }
```

```
}
```

```
add_action('admin_menu', 'remove_menus');
```

El código anterior se encarga de eliminar del menú las opciones que se indiquen dentro del array “\$restricted”. En nuestro ejemplo hemos eliminado las opciones de Apariencia, Herramientas, Usuarios, Configuración y Plugins. El menú quedaría tal y como se ve en la imagen anterior.

Imagen destacada obligatoria para poder publicar

Hay ocasiones que en el diseño de un blog, es necesario que se muestre una imagen destacada en el listado de publicaciones. Si queremos evitar que se nos olvide esta imagen, podemos añadir el siguiente código que hace que sea obligatorio indicarla.

```
add_action('save_post', 'wpds_check_thumbnail');

add_action('admin_notices', 'wpds_thumbnail_error');

function wpds_check_thumbnail( $post_id ) {

    if( get_post_type($post_id) != 'post' )

        return;

    if ( ! has_post_thumbnail( $post_id ) ) {

        set_transient( "has_post_thumbnail", "no" );

        remove_action('save_post', 'wpds_check_thumbnail');

        wp_update_post(array('ID' => $post_id, 'post_status' => 'draft'));

        add_action('save_post', 'wpds_check_thumbnail');

    } else {

        delete_transient( "has_post_thumbnail" );

    }

}

function wpds_thumbnail_error() {

    if ( get_transient( "has_post_thumbnail" ) == "no" ) {

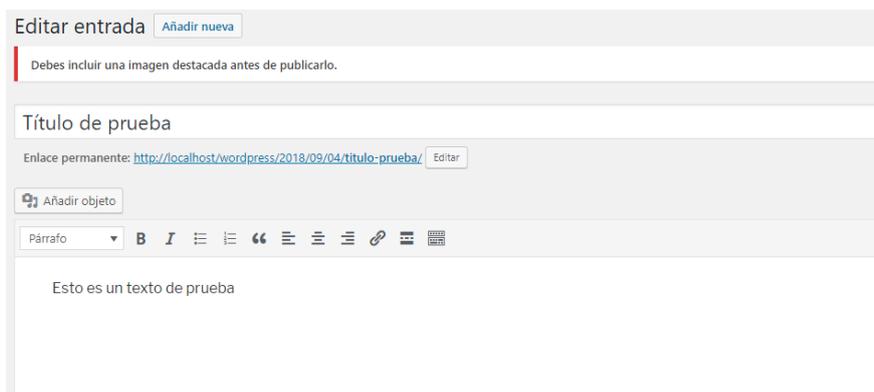
        echo "<div id='message' class='error'><p><strong>Debes incluir una imagen destacada antes de publicarlo.</strong></p></div>";

        delete_transient( "has_post_thumbnail" );

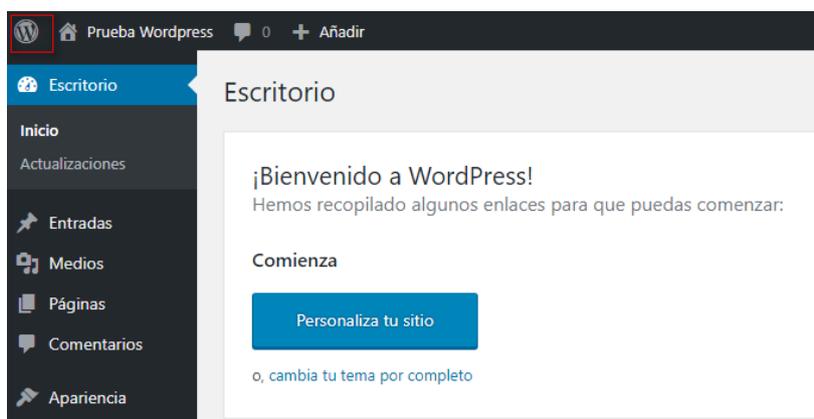
    }

}
```

Una vez que tengamos este código dentro de nuestro archivo **“functions.php”**, al pulsar sobre el botón de **“Publicar”** nos aparecerá un mensaje informando de que debemos indicar una imagen destacada quedando la publicación en borrador, tal y como podéis ver en la imagen siguiente.



Cambiar el logo del backoffice por el nuestro



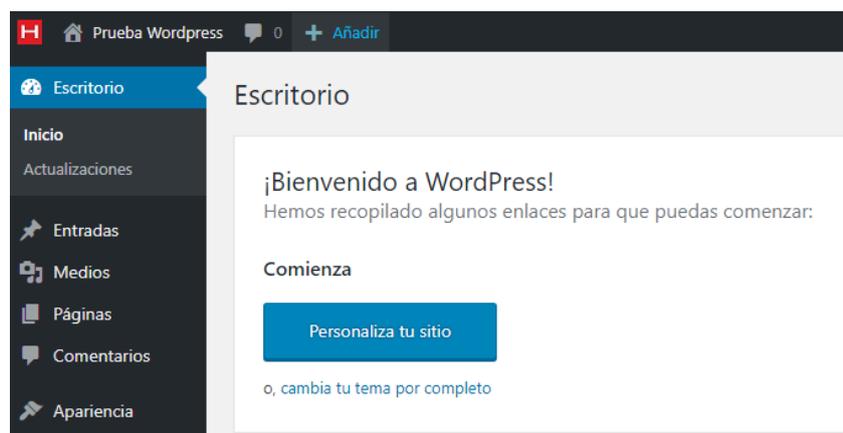
Por defecto, dentro de la administración de WordPress en la esquina superior izquierda, aparece el logotipo de este famoso CMS. Cambiarlo resulta muy fácil mediante el siguiente código que os dejamos y que habrá que colocar dentro de nuestro **“functions.php”**.

```
function wpb_custom_logo() {
    echo '
<style type="text/css">
    #wpadminbar #wp-admin-bar-wp-logo > .ab-item .ab-icon:before {
        background-image: url(' . get_bloginfo('stylesheet_directory') . '/images/logo-hostalia.jpg)
!important;
        background-position: 0 0;
        color:rgba(0, 0, 0, 0);
    }

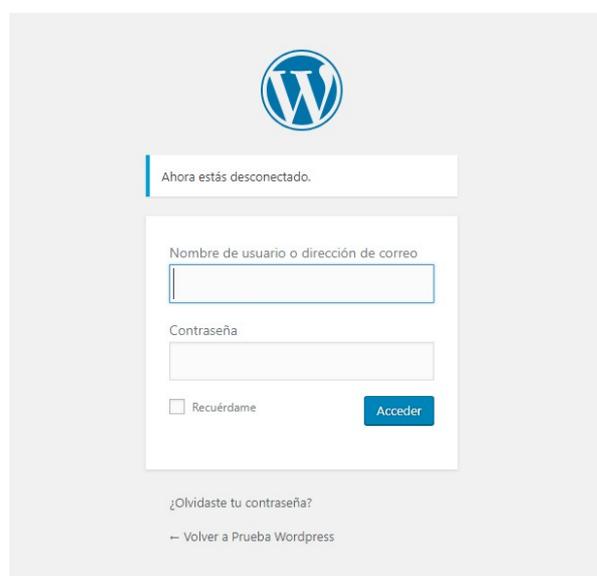
```

```
#wpadminbar #wp-admin-bar-wp-logo.hover > .ab-item .ab-icon {  
  
    background-position: 0 0;  
  
}  
  
</style>  
  
';  
  
}  
  
add_action('wp_before_admin_bar_render', 'wpb_custom_logo');
```

Lo que hay que hacer es subir la imagen que queremos utilizar como logo en nuestra administración en nuestra instalación de WordPress. Nosotros la hemos colocado dentro de una carpeta de nombre “**images**” en el directorio de la plantilla que utilizamos. El resultado lo podéis ver en la imagen siguiente.



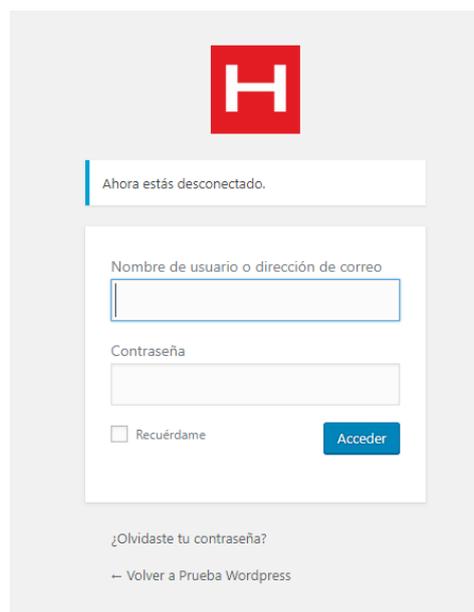
Cambiar el logo del login por el nuestro



En el formulario de acceso a la administración de WordPress, aparece su logotipo en la parte superior. Al igual que en el caso anterior, esta imagen puede ser cambiada para colocar nuestro propio logotipo y conseguir de esta forma una mayor personalización. Para llevar a cabo esta acción, deberemos utilizar el siguiente código.

```
function my_custom_login_logo() {  
  
    echo '<style type="text/css">  
  
        h1 a { background-image:url('.get_bloginfo('template_directory').'/images/logo-hostalia-admin.jpg)  
!important; }  
  
    </style>';  
  
}  
  
add_action('login_head', 'my_custom_login_logo');
```

Al igual que hemos comentado anteriormente, deberemos colocar la imagen que queremos utilizar en la instalación de nuestro WordPress para después hacer referencia a ella. El resultado lo podéis ver en la siguiente imagen.



Mostrar mensajes informativos personalizados para los usuarios

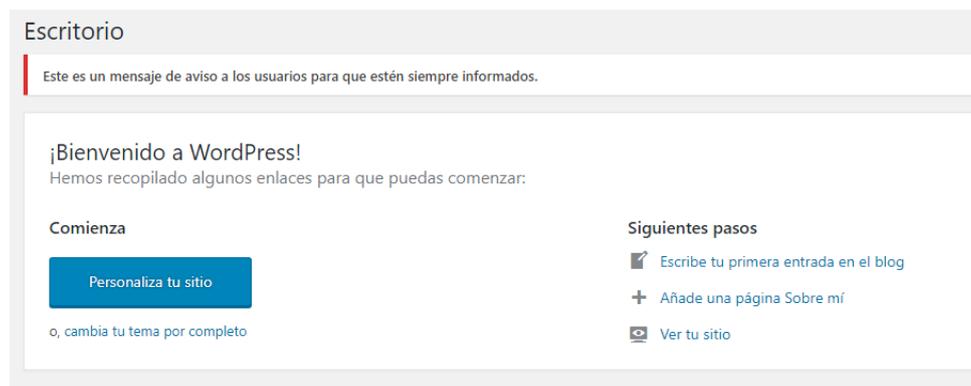
Hay ocasiones en las que al realizar un desarrollo para un cliente, puede ser de gran ayuda mostrarle ciertos mensajes informativos para que no tenga duda a la hora de utilizar esta herramienta. Para colocar estos mensajes personalizados, lo podemos hacer de la siguiente forma.

```
function showMessage($message, $errmsg = false)
{
    if ($errmsg) {
        echo '<div id="message" class="error">';
    }
    else {
        echo '<div id="message" class="updated fade">';
    }
    echo "<p><strong>$message</strong></p></div>";
}

function showAdminMessages()
{
    showMessage("Este es un mensaje de aviso a los usuarios para que estén siempre informados.", true);
}

add_action('admin_notices', 'showAdminMessages');
```

El resultado lo podéis ver en la siguiente imagen.



Deshabilitar sugerencias de inicio de sesión de WordPress

Cuando vamos a loguearnos para acceder a la administración de WordPress y ponemos los datos mal, la aplicación nos devuelve un mensaje de error. En ocasiones, estos mensajes pueden ofrecer información valiosa para los **atacantes**. Por suerte, deshabilitar estos mensajes y poner uno personalizado es fácil. Copia el siguiente código en tu archivo "**functions.php**" y personaliza el mensaje que quieres que aparezca.



Añadir campos extra en el perfil de autores

¿Quieres añadir campos extra en el perfil de autores de WordPress? Puedes llevar a cabo esto añadiendo el siguiente código en el archivo “functions.php” de tu theme.

```
function wp_campos_extra( $contactmethods ) {  
    // Añadimos Twitter  
    $contactmethods['twitter'] = 'Twitter';  
    // Añadimos Facebook  
    $contactmethods['facebook'] = 'Facebook';  
  
    return $contactmethods;  
}  
add_filter('user_contactmethods','wp_campos_extra',10,1);
```

De esta forma estaremos indicando que al editar los datos del usuario, éste pueda indicar su cuenta de **Facebook** y **Twitter**. Podemos meter tantos campos como queramos.

Información de contacto	
Correo electrónico (obligatorio)	<input type="text"/>
Web	<input type="text"/>
Twitter	<input type="text"/>
Facebook	<input type="text"/>

Como hemos podido ver a lo largo de este **White Paper**, el archivo “**functions.php**” ofrece mucho juego a la hora de realizar cambios en nuestra instalación de WordPress.