Aprender PHP para el desarrollo de WordPress – Guía completa

Supongamos que en estos momentos tienes un sitio web de WordPress y has configurado tu plantilla, lees un poco acerca de las etiquetas de plantilla, y tal vez incluso has modificado el archivo functions.php en el editor de temas del panel de administración.

Pero ahora, quieres que tus habilidades pasen al siguiente nivel y profundizar más en el código.

Por suerte, WordPress es un buen lugar para empezar. Hay un buen tocho de documentación disponible, el código es fácil de leer en su mayor parte, se explica por sí mismo y no demasiado es difícil de recordar.

En este artículo, voy a mostrar una breve introducción al mundo de la programación de WordPress. Mientras que este post está dirigido a principiantes, voy a suponer que sabes cómo editar archivos de WordPress y te has mirado un archivo de tu tema de WordPress, incluso si no entiendes lo que está pasando dentro del archivo.

Los lenguajes de programación de WordPress

WordPress utiliza un número de diferentes lenguajes de programación. Si un lenguaje tiene que ser señalado como el "principal", este sería PHP, pue es el lenguaje que está del lado del servidor y alimenta cerca del 82 por ciento de la web.

WordPress también utiliza HTML, CSS y Javascript. HTML se utiliza para dar la estructura del sitio y es empleado por todos los sitios web. CSS proporciona el estilo a un documento HTML. Por ejemplo, CSS hace que su fondo

sea blanco, el texto de color gris oscuro y coloca la barra lateral a la derecha. Javascript, por otro lado, añade características avanzadas como sliders y otras características interactivas.

Finalmente, WordPress también utiliza MySQL como base de datos. MySQL se utiliza por ejemplo para recuperar las últimas 10 entradas, o todas las entradas de una categoría en particular de la base de datos.

Pues si, la mala noticia es que se trata de una cantidad considerable de conocimientos. La buena noticia es que no necesitas saber de todo para empezar, de hecho, se puede empezar a funcionar con muy poco. Se puede aprender programación a través de WordPress con sólo copiar y pegar ejemplos de la documentación.

Algunos consejos para el que empieza a programar

La mejor forma de aprender es a través de tutoriales, documentación y el trabajo de los demás. La dificultad con la programación no proviene de la complejidad de los lenguajes implicados. Si se desglosa en componentes, todo lo que se aprende es fácil.

Si la programación puede ser difícil es por dos razones. Necesitas saber un montón de cosas simples y con el fin de crear un producto de éxito, tienes que ser capaz de pensar en términos de sistemas, que toma un poco de práctica.

Lo que debes tener claro es que mientras se aprende a codificar para WordPress, tendrás un montón de momentos de autentica desesperación. Vas a estar frustrado por la falta de comprensión desde el principio, pensando que un código está perfectamente formado y no funciona, pasarás horas luchando con él sólo para descubrir que has olvidado un punto y coma. Todo esto es perfectamente normal. Cada programador ha sentido esto, no sólo tu.

Lo que WordPress no es

Es importante darse cuenta de que técnicamente, no es lo mismo “codificar en WordPress” y el “código de WordPress”. WordPress es un montón de código escrito en PHP. Por ejemplo, Joomla y Drupal (otros dos sistemas de gestión de contenidos) también están escritos en PHP.

Vamos a explicar lo anterior un poco mejor. Decir “código de WordPress” es como decir un “coche BMW”. BMW, Mercedes y Nissan son todos coches. Todos se construyen con tuercas, pernos y soldaduras. La diferencia entre ellos es la forma en que se ponen juntas, las filosofías del diseño y las prácticas de encaje.

WordPress, Joomla, Drupal y todos los otros sistemas y frameworks que existen, están todos construidos con los mismos componentes. La diferencia entre ellos es la filosofía de codificación y metodologías que emplean.

Cómo trabajar con PHP

Como mencioné anteriormente, PHP es un lenguaje de script del lado del servidor. En contraste, el HTML es un lenguaje de marcado de documentos del lado del cliente. Analicemos HTML primero para entender lo que esto significa.

La forma en que tu navegador interpreta el código HTML es el siguiente: Cuando visitas una página HTML se envía el código HTML para tu navegador. El navegador procesa la información y devuelve algo que reconoces como una página web.

Cuando el navegador visita una página que utiliza PHP, se emplea un paso intermedio. En primer lugar el código PHP es procesado por el servidor. El resultado de este procesamiento es una página HTML, que se envía al navegador y se muestra para que puedas verlo.

El procesamiento adicional por el servidor parece como un paso innecesario, pero ni mucho menos. Veamos un ejemplo práctico con código PHP real:

```
<?php if( date( 'G' ) > 18 ) : ?>

    <h2>Buenas noches</h2>

<?php else : ?>

    <h2> Buenos días </h2>

<?php endif ?>
```

Sin ningún conocimiento de código PHP, ya podemos obtener alguna información sobre este. Sólo con verlo, se puede interpretar que en un conjunto particular de circunstancias se va mostrar “Buenas noches”, y en otras se mostrará “Buenos días”.

Cuando nos fijamos en la fuente de la página web resultante no habrá rastro de este código. Todo lo que se ve es “Buenos días” o “Buenas noches” según que circunstancias. Esto se debe a que el servidor realiza el procesamiento y sólo envía el resultado.

En el ejemplo anterior se ha utilizado la función de fecha para determinar qué hora es. date (‘G’) devuelve un número del 0 al 23, donde 0 representa la medianoche y el 23 representa las 23:00. Si el valor de esta función es más de 18 (que es posterior a las 18:00) mostramos las buenas noches. De lo contrario nos mostrará los buenos días.

Ahora sabemos dos cosas acerca de PHP. Se nos permite utilizar sentencias if para mostrar contenido en función de nuestros propios criterios. También sabemos que tiene funciones, que nos ayudan a llevar nuestro

objetivo a cabo. La función date() devuelve la fecha actual en un formato determinado. La función strtolower () convierte cualquier texto a minúsculas.

PHP en WordPress

Con ese último párrafo en mente, puedes reconocer PHP en todas las partes de WordPress. Abre el content.php del tema por defecto Twenty Fourteen y échale un vistazo. Este archivo se encarga de mostrar el contenido de las entradas del blog en el tema.

Vamos a comparar la primera línea de este archivo (desechando el comentario en la parte superior)...

```
<article id="post-<?php the_ID(); ?>" <?php post_class(); ?>>
```

... Con la salida que genera cuando se visita la página:

```
<article id="post-344" class="post-344 post type-post status-publish format-standard has-post-thumbnail sticky hentry category-photos tag-example tag-tag">
```

Podemos deducir de la comparación que la función the_ID() se sustituye por el ID del artículo en cuestión. La función post_class() añade una gran cantidad de clases para el elemento HTML. Estos nos ayuda a estilizar nuestras entradas más adelante. No es importante en esta etapa saber por qué se añaden estas clases específicas, sólo estamos familiarizándonos con funciones.

Más adelante, mirando de las líneas desde la 24 a la 28 también podemos ver una sentencia if:

```
if ( is_single() ) :
the_title( '<h1 class="entry-title">', '</h1>' );
else :
the_title( '<h1 class="entry-title"><a href="' . esc_url( get_permalink() ) . '" rel="bookmark">', '</a></h1>' );
endif;
```

La sentencia if contiene is_single(). Ésta es una función que devuelve 'verdadero' si estamos ante una única página de entrada "single post page", de lo contrario devolverá 'falso'. Cuando es verdadero, y estamos en una sola página, utilizamos la función the_title() para emitir el título.

Si es falsa, seguimos usando la función the_title(), pero nos aseguramos de que se trata de un enlace a la única página de entrada.

Ten en cuenta que algunas funciones son vacías "empty", mientras que algunas tienen partes y piezas dentro de ellas. Por ejemplo, is_single() es una función de vacío mientras the_title() tiene algo dentro del paréntesis.

Los elementos dentro del paréntesis se llaman argumentos. Cada función tiene diferentes argumentos separados por comas, que se puede aprender a través de la documentación. El artículo del Codex sobre the_title () nos muestra que esta función tiene tres argumentos:

1. El primer argumento nos permite agregar HTML antes del título.
2. La segunda nos permite agregar HTML después del título.
3. El tercer parámetro determina el tiempo que el título se muestra o simplemente se almacena para su uso posterior.

Con base a esto, ahora entendemos lo que está pasando en la línea 25 del archivo content.php:

```
1the_title( '<h1 class="entry-title">', '</h1>' );
```

La función muestra el título, pero se antepone un tag de apertura H1 en este y anexa la etiqueta final.

El resultado de este código es el siguiente en el navegador:

```
1<h1 class="entry-title">El título de mi artículo</h1>
```

¿Cómo subir el nivel en programación de WordPress?

Es probable que no quieras pasar semanas trasteando a través de la documentación de PHP y aprendiendo todo desde el principio. Pero debes hacerlo, y también te recomiendo que experimentes tanto como sea posible.

¿Quieres pasar la lista de etiquetas de la parte inferior del artículo a la parte de arriba? La función the_tags() en la parte inferior del archivo content.php parece ser la clave.

En primer lugar vamos a eliminarla. Luego, al guardar y actualizar la página, la lista de etiquetas desaparece. Esto es bueno, ya que significa que es la función que da salida a las etiquetas. Ahora sólo tienes que copiarlo y pegarlo en diversas partes del archivo para ver dónde termina.

Es probable que cuanto más alto incluyas el código, más alto estará en el contenido. Con un poco de experiencia, serás capaz de identificar las cosas como the_excerpt() y the_content() siendo responsable de mostrar el contenido, por lo que poner en cualquier lugar por encima de estos, los colocará por encima de la página principal.

Aprender cómo codificar para WordPress de esta manera es divertido y te anima a leer la documentación, que siempre es algo muy bueno. No te preocupes si no lo entiendes todo, se llega a un punto en que lo haces lo suficientemente pronto.

Aprender las malas prácticas

Una desventaja de este método es que empleas malas prácticas. Mientras que mi recomendación de copiar y pegar la función the_tags() para la parte superior del archivo funciona en alguna parte, el HTML para el pie de página, que utiliza la etiqueta de pie de página, necesitará alguna modificación para que sea un código correcto.

Una vez más, olvídate de esto por ahora. No estás construyendo a nivel profesional un código listo para mostrarlo a Google. Estás tratando de aprender lo básico y averiguar cómo funciona todo. Esto no es una tarea fácil y los errores son parte del proceso.

Una vez que tengas un buen conocimiento del código detrás de WordPress, puedes empezar un aprendizaje de sus malas prácticas, y puedes comenzar a estudiar los patrones de codificación y averiguar por qué hacemos las cosas de la manera que lo hacemos.

Una visión general importante sobre el código de WordPress

WordPress tiene una serie de “subsistemas”, como el “loop” que controla las entradas que se muestran. Los ganchos “hooks” que permiten modificar la funcionalidad por defecto, varias APIs y el curso de temas y plugins. Vamos a ver una breve introducción a algunos de los sistemas más importantes que pueden aparecer.

Habilitación de depuración

Por defecto, WordPress ocultará cualquier error de código. Esto es recomendable, pero puede dar lugar a dos problemas durante el desarrollo. Si cometes un error no fatal, no recibirás mensajes de error y tu código, o bien no va a hacer nada o no producirá el resultado esperado.

La otra cuestión es una pantalla blanca de la muerte. No hay mensajes de error, sólo una pantalla en blanco sin acceso a la parte delantera o backend. Para asegurarte de que esto no ocurra debes habilitar la depuración, el cual te proporcionará los mensajes de error.

Esto se puede hacer mediante la edición del archivo wp-config.php en el directorio raíz de la instalación de WordPress. Busca la línea que contiene: define(‘WP_DEBUG’, false); y cambiar falso a verdadero “true”. Eso es todo lo que hay que hacer.

Temas Hijo “child theme”

Los temas hijo son temas separados, que se basan en un tema principal. Estos heredan todo desde el tema padre, a menos que se especifique lo contrario. Esta es la única manera segura para modificar un tema. Como mencioné anteriormente, la forma más fácil de aprender es modificar un tema existente. Me gustaría añadir que con eso y el uso de un tema hijo.

Si creas un tema hijo basándote en Twenty Fourteen, aún puedes personalizarlo a tu gusto, pero también puedes actualizar el tema sin perder todos los cambios. Esto es algo que también debes tener en cuenta cuando se trabaja con los clientes. Siempre – Siempre utiliza un tema hijo.

La creación de un tema hijo es un juego de niños. Solo es necesario crear una nueva carpeta en el directorio themes y el nombre que quieras darle. Para nuestro ejemplo, vamos a crear una carpeta llamada “tema-hijo”. Dentro de esta carpeta creamos dos archivos; style.css y functions.php. Abre la hoja de estilo y utiliza el siguiente para crear el tema hijo:

```
/*
Theme Name: Twenty Fourteen Child
Theme URI: http://mitema.com
Description: Mi tema hijo
Author: Pedro Mendez
Author URI: https://reinspirit.com
Template: twentyfourteen
Version: 1.0.0
Tags: light, dark, two-columns, right-sidebar, responsive-layout, accessibility-ready
Text Domain: mi-tema-hijo
*/

/* =La personalización del tema empieza aquí
-------------------------------------------------------------- */
```

En realidad se puede usar lo que quieras en el ejemplo anterior, la única restricción es la línea que comienza con “Template”. Esto debe contener el nombre del directorio del tema principal.

Para el uso de temas hijos, la regla es la siguiente: Cada vez que un archivo se carga en WordPress lo busca en el tema hijo primero. Si no existe, el mismo archivo desde el tema padre será el que carge. La única excepción a esto es el functions.php . Los archivos de función de ambos temas serán cargados, primero el tema hijo y luego el tema principal.

En este punto puedes cambiar a tu tema hijo, pero tu sitio vas a estar desprovisto de cualquier estilo. Basándonos en nuestra regla anterior es fácil ver por qué. La hoja de estilo se carga desde el tema hijo, ya que style.css existe en el tema hijo, pero esta no contiene ninguna información de estilo.

El siguiente paso es cargar los estilos del tema principal. Esto se puede hacer con “enqueue” para la hoja de estilo del tema padre. No te preocupes demasiado por esto. Puedes copiar y pegar el siguiente código en tu tema hijo, en el archivo functions.php. Ten en cuenta que esto carga los estilos del tema padre.

```
1add_action( 'wp_enqueue_scripts', 'my_parent_styles' );
2function my_parent_styles() {
3       wp_enqueue_style( 'parent-style', get_template_directory_uri().'/style.css' );
4}
```

En este momento tu tema hijo es exactamente el mismo que tu tema padre. Ahora puedes empezar a modificar lo que quieras. Puedes utilizar la hoja de estilos para anular estilos o añadir tus reglas adicionales. Si deseas modificar el archivo “index”, por ejemplo, todo lo que necesitas hacer es crearlo.

Si creas un archivo index vacío entonces cualquier página que utilice ese archivo estará en blanco. Todas las demás páginas continuarán trabajando bien desde que usan el tema principal. Puedes comenzar, ya sea escribiendo tu propio código en el archivo index o puedes copiar y pegar el código del tema padre y modificar este.

El resultado de esto debería ser el siguiente: Puedes modificar el contenido de un tema existente desde su núcleo, pero aún así ser capaz de actualizar el tema padre o cambiar de nuevo el tema padre en cualquier momento.

El Query y el Loop

El Query “la consulta” es el sistema que “sabe” que entradas mostrar en una página y el Loop “bucle” es la parte que realmente pasa por cada entrada y los muestra. Por ejemplo, en tu página principal el query busca las 10 entradas más recientes. En una página archivo de categorías, la consulta busca las 10 entradas más recientes de la categoría dada. La consulta se utiliza incluso en las páginas individuales donde se ve un solo post en la base de datos.

La consulta es algo que se puede modificar y utilizar para tus propias necesidades, pero por ahora nos concentraremos en el uso por defecto o estándar. Sólo usaremos el resultado a través del bucle.

El bucle toma todos las entradas que la consulta ha devuelto y pasa a través de cada uno de ellos uno por uno. En algunas páginas, como las páginas individuales, sólo hay una entrada. Esto cuenta como una “colección” de entradas, en este caso la colección se compone de una sola entrada.

Veamos el código básico para un bucle y como pasa línea por línea:

```
<?php if( have_posts() ) ?>
<?php while( have_posts() ) : the_post() ?>
<div <?php post_class() ?>>
<h2><?php the_title() ?></h2>
<div class='content'>
<?php the_content() ?>
</div>
</div>
<?php endwhile; ?>
<?php else: ?>
Aquí no hay entradas
<?php endif ?>
```

La primera línea utiliza una sentencia if junto con la función have_posts() para averiguar si hay alguna entrada para la consulta. Si no hay ninguna entrada, se ejecuta el código después de la sección “else”, que notifica al usuario que no hay entradas.

Si hay entradas utilizamos un bucle PHP. Hay unos cuantos tipos de bucles en PHP. Para repasar la sintaxis y algunos ejemplos más, échale un vistazo a este tutorial sobre tipos de bucles en PHP.

En el código anterior usamos un bucle “while”, que contiene la función have_posts() de nuevo. Esta función devuelve “false” cuando no hay posts en el bucle, o no hay más posts en el bucle porque todos estos se han mostrado.

Todo dentro de nuestro bucle while se ejecuta mientras que el valor de esta función es “true”. Esto es exactamente lo que necesitamos. Tan pronto como hemos mostrado el último post, el valor de have_posts() será false por lo que el bucle termina.

Dentro del bucle se ha creado una pantalla muy rudimentaria de un post utilizando las etiquetas de plantilla que hemos aprendido anteriormente.

El bucle se debe utilizar en cualquier archivo de plantilla del tema que enumera las entradas. Buscar páginas, páginas de entrada individuales, páginas de archivo, el archivo de index, en cualquier momento que están listas las entradas usa un loop.

Consultas personalizadas “Custom Queries”

Es poco común aprender primero sobre consultas personalizadas, pero lo visto, es uno de los temas más buscados después de las características de WordPress. En la sección anterior hemos aprendido cómo se puede enumerar entradas utilizando el bucle, pero estamos limitados por lo que se devuelve de forma predeterminada. ¿Qué pasa si deseas mostrar los siguientes posts relacionados en la misma categoría en una entrada individual? Esto es fácil con custom queries y el loop.

Puedes crear una consulta personalizada usando la clase WP_Query. No hemos hablado de las clases, pero su uso es bastante sencillo. He aquí un ejemplo que muestra las entradas programadas de una categoría específica. Puedes usar esto para mostrar una sección de “Siguientes artículos en esta categoría”.

```
<?php
$args = array(
'post_type' => 'post',
'post_status' => 'future',
'category_name' => 'app_reviews',
'posts_per_page' => 3
);
$coming_soon = new WP_Query( $args );
?>
```

Como puedes ver esto es bastante sencillo. Para modificar esto para tus necesidades, modificando el contenido de $args array. Hay un montón de parámetros que puedes utilizar para restringir las entradas en función de tu fecha de publicación, en base a sus autores, categorías, campos personalizados y mucho más. Echa un vistazo a la documentación WP_Query para obtener una lista completa.

Ahora que tenemos una consulta personalizada podemos utilizar un bucle personalizado para mostrar el contenido. Todo lo que necesitamos hacer es agregar el prefijo de have_posts() y the_post() que funciona con el nombre de la variable que contiene la consulta y una "flecha":

```
<?php if( $coming_soon->have_posts() ) ?>
<ul>
<?php while( $coming_soon->have_posts() ) : $coming_soon->the_post() ?>
<li <?php post_class() ?>>
<a href='<?php the_permalink() ?>'><?php the_title() ?></a>
</li>
<?php endwhile; ?>
</ul>
<?php endif ?>
```

Fijate que no se ha utilizado la parte else del bucle y se utiliza una lista HTML en lugar de divs. Desde este bucle se pretende enumerar una lista de posts debajo de un post individual completo, es mejor no mostrar nada si no hay entradas. Además, una simple lista con enlaces en ese lugar, debe ser suficiente para que los usuarios pulsen en estos.

Ganchos "Hooks"

WordPress utiliza un ingenioso sistema que te permite modificar las funciones del núcleo "core". Si no entiendes el núcleo de WordPress te aconsejo lo siguiente: bajo ninguna circunstancia se debe modificar los archivos centrales. Esto significa que no se puede editar cualquier archivo que viene con WordPress por defecto.

Sé que a veces parece que es la única manera, pero nunca se debe hacer. Todo lo que puedas necesitar, se puede hacer con 'hooks' u otros métodos. La modificación de los archivos centrales no sólo es peligroso, si no que además cualquier cosa que hagas será sobrescrita por una versión actualizada de WordPress.

Los hooks permiten modificar el funcionamiento por defecto de WordPress. Vienen en dos formatos: acciones y filtros "actions/filters". Las acciones te permiten ejecutar una función propia tuya en lugares específicos en el código de WordPress. Por ejemplo, mediante el uso de un hook que puede ejecutar una de tus propias funciones cuando WordPress publica un post. Esto permite notificar el autor, por ejemplo.

Los filtros te permiten modificar los datos antes de su uso. Por ejemplo, se puede usar un filtro para modificar el texto que se muestra al usuario cuando se guarda una entrada. En lugar de "Guardar borrador, Publicar, Actualizar," podrías modificar esto para decir "El borrador se ha guardado".

Un gran ejemplo de un hook de acción es wp_footer. Esta acción se lleva a cabo justo antes de la etiqueta del cuerpo de cierre de un tema. Te permite añadir tus propios contenidos hasta la parte de final de un tema sin necesidad de modificar el archivo de pie de página del tema en sí mismo. En el functions.php de tu tema podrías utilizar lo siguiente para agregar un código de seguimiento a tu sitio:

```
1
2add_action( 'wp_footer', 'my_tracking_code' );
3function my_tracking_code() {
4?>
5// Pega el código de Google Analytics aquí
6<?php
}
```

La primera línea le dice a WordPress que nos gustaría añadir nuestra función my_tracking_code() para el hook wp_footer. Cuando WordPress carga una página y ve el hook wp_footer, este mira todas las funciones vinculadas a la misma y los ejecuta.

Nuestra función a continuación, agrega el código de seguimiento de Google Analytics para el pie de página.

Esta es la base del funcionamiento de los plugins. Si quieres crear un plugin y pegar el mismo código allí no tendrías que modificar tu tema en absoluto. Lo que esto significa es que incluso si cambias de tema, tu código de Google Analytics continuará trabajando sin problemas.

Para mostrar cómo funcionan los filtros, vamos a modificar el contenido de un post con uno. El filtro the_content() se ejecuta antes de que muestre el contenido de un post. Utilizamos un hook para atar una función para que podamos modificarlo.

El código siguiente agrega el texto “Comprobado por” automáticamente después de cada entrada individual (o más exactamente, en cualquier momento que el contenido del post completo es mostrado).

```
1add_filter( 'the_content', 'my_content_filter' );
2function my_content_filter( $content ) {
3	$content .= ' (Comprobado por Pedro Mendez)';
4}
```

Ten en cuenta que la función en este tiempo ha recibido un parámetro. Cada filtro y acción pueden tener uno o más parámetros. Tendrás que comprobar la documentación para ver lo que puede hacer el hook en concreto que se estamos usando. Para obtener una lista de las acciones y filtros recomiendo la acciones de referencia y el filtro de referencia.

Lectura adicional

Hay mucho que se puede aprender acerca de WordPress y una gran cantidad de los conocimientos están disponibles de forma gratuita. He reunido algunos recursos para ti y por categorías. Espero que encuentres estos recursos útiles.

Documentación WordPress

WordPress Codex – La página principal de toda la documentación de WordPress. El Codex pronto será reemplazado por recursos para desarrolladores
Etiquetas de Plantilla – Funciones que puedes utilizar en los bucles
Etiquetas Condicionales – Funciones que devuelven verdadero o falso en escenarios específicos
Referencia al WP_Query – El lugar para ir si necesitas una consulta personalizada
Referencia a Función – Una enorme lista de funciones que puedes utilizar
Referencia a Acción – Si necesitas para ejecutar tus propias funciones
Referencia a Filtro – Si necesitas modificar cadenas o matrices utilizadas por WordPress
Referencia al Hook – Completa referencia a acción y filtro
Desarrolla un Plugin – La guía de inicio para escribir plugins
Desarrolla un tema – La guía de introducción a la creación de un tema
Temas Hijo – Una guía para crear un tema hijo
API de WordPress – Una lista de los usos API de WordPress
Referencia a Clases – Una lista de usos clases para el uso en WordPress

Cursos completos

Codecademy – Codecademy tienen clases interactivas para varios idiomas
Treehouse – vídeos increíbles en una variedad de codificación de temas relacionados
Tuts + – Grandes cursos sobre una serie de temas diferentes

Aprendiendo sobre PHP

Manual de PHP – La documentación oficial de PHP
Codecademy – Tutorial interactivo PHP completo
W3Schools – Gran tutorial completo de PHP
Tizag – Otra guía completa PHP

HTML, CSS y Javascript

W3Schools – W3Schools tiene tutoriales completos a todos los idiomas y más mucho más
HTML 5 Doctor – Un gran lugar para aprender sobre nuevas etiquetas y las sutilezas de HTML5

Obtención de ayuda

WordPress Foros de soporte – Foros Oficiales de WordPress

Temas avanzados

Sass – CSS con superpoderes
LESS– CSS con soporte para las variables y funciones
OOP PHP – Orientado a objetos PHP
Tutorial SQL – Aprende cómo consultar la base de datos tu mismo
Laracasts – PHP Moderno y tutoriales Laravel
Koala – Gratuito, multiplataforma de código del compilador
Prepros – Premium, multiplataforma de código del compilador
CodeKit – Compilador código OSX
Grunt – Gratis, terminal basado en código compilador