

Domine **WordPress** Manual práctico



Francisco Javier Carazo Gil



Ra-Ma®

Domine WordPress.

Manual práctico

Francisco Javier Carazo Gil





Domine WordPress. Manual práctico. (E-Book)
© Francisco Javier Carazo Gil

© De la Edición Original en papel publicada por Editorial RA-MA.

ISBN: 978-84-9964-063-1

Todos los derechos reservados © RA-MA, S.A. Editorial y Publicaciones, Madrid, España.

MARCAS COMERCIALES. Las designaciones utilizadas por las empresas para distinguir sus productos (hardware, software, sistemas operativos, etc.) suelen ser marcas registradas. RA-MA ha intentado a lo largo de este libro distinguir las marcas comerciales de los términos descriptivos, siguiendo el estilo que utiliza el fabricante, sin intención de infringir la marca y sólo en beneficio del propietario de la misma. Los datos de los ejemplos y pantallas son ficticios a no ser que se especifique lo contrario.

RA-MA es marca comercial registrada.

Se ha puesto el máximo empeño en ofrecer al lector una información completa y precisa. Sin embargo, RA-MA Editorial no asume ninguna responsabilidad derivada de su uso ni tampoco de cualquier violación de patentes ni otros derechos de terceras partes que pudieran ocurrir. Esta publicación tiene por objeto proporcionar unos conocimientos precisos y acreditados sobre el tema tratado. Su venta no supone para el editor ninguna forma de asistencia legal, administrativa o de ningún otro tipo. En caso de precisarse asesoría legal u otra forma de ayuda experta, deben buscarse los servicios de un profesional competente.

Reservados todos los derechos de publicación en cualquier formato e idioma.

Según lo dispuesto en el Código Penal vigente ninguna parte de este libro puede ser reproducida, grabada en sistema de almacenamiento o transmitida en forma alguna ni por cualquier procedimiento, ya sea electrónico, mecánico, reprográfico, magnético o cualquier otro sin autorización previa y por escrito de RA-MA; su contenido está protegido por la Ley vigente que establece penas de prisión y/o multas a quienes, intencionadamente, reprodujeran o plagiaran, en todo o en parte, una obra literaria, artística o científica.

Editado por:

RA-MA Editorial

Calle Jarama, 3A. Polígono Industrial Igarsa

28860 PARACUELLOS DE JARAMA, Madrid

Teléfono: 91 658 42 80

Fax: 91 662 81 39

Correo electrónico: editorial@ra-ma.com

Internet: www.ra-ma.es y www.ra-ma.com

ISBN: 978-84-940090-3-7

Autoedición: Autores

Maquetación: Gustavo San Román Borrueco

Diseño Portada: Antonio García Tomé

E-Book desarrollado en España en Junio de 2012

*A María, mis padres, mis hermanos, mi familia, mis amigos, a los que ya no
están y, en definitiva, a todas y cada una de las personas que me han hecho y me hacen
tan agradables todos los días de mi vida.*

ÍNDICE

INTRODUCCIÓN	15
CAPÍTULO 1. ¿QUÉ ES WORDPRESS?.....	17
1.1 TÉRMINOS RELACIONADOS.....	17
1.1.1 Blog	17
1.1.2 Blogosfera	18
1.1.3 Blogging.....	18
1.1.4 Microblogging.....	18
1.2 ORÍGENES DEL BLOGGING.....	19
1.3 WORDPRESS	21
1.3.1 Breve historia	22
1.3.2 ¿Qué es un sistema gestor de contenidos?	24
1.4 ¿POR QUÉ WORDPRESS?.....	24
1.4.1 Instalación, actualización y manejo simple.....	25
1.4.2 Código fuente libre y entendible	25
1.4.3 Popularidad.....	27
1.4.4 Integración en motores de búsqueda.....	28
1.4.5 Seguridad.....	28
1.4.6 Protección contra el spam	29
1.4.7 WordPress.com	29
1.4.8 Otras ventajas.....	30
1.5 ALTERNATIVAS.....	30
1.5.1 Movable Type	30
1.5.2 Joomla!.....	31

1.5.3 Drupal	32
CAPÍTULO 2. WORDPRESS.COM	33
2.1 NUESTRO PRIMER BLOG	33
2.1.1 Alta de usuario y creación del blog	34
2.1.2 Explorando el escritorio	36
2.1.3 Nuestra primera entrada	37
2.1.4 Etiquetas y categorías	37
2.1.5 Gestión de las entradas	37
2.1.6 Publicación	39
2.1.7 Contenido estático	40
2.2 PERSONALIZACIÓN	41
2.2.1 Temas	41
2.2.2 Widgets	42
2.2.3 Menús	44
2.2.4 Otros	46
2.3 COMENTARIOS	46
2.4 PUNTUACIONES	47
2.5 SERVICIOS DE PAGO	47
2.5.1 Dominio	48
2.5.2 VideoPress	48
2.5.3 CSS personalizado	48
2.5.4 Espacio adicional	48
2.5.5 Sin anuncios	49
2.5.6 Usuarios privados ilimitados	49
2.6 OTRO SERVICIO DE BLOGS EN LÍNEA: BLOGGER	49
2.6.1 Blogger	49
2.6.2 A favor	50
2.6.2.1 Integración con Google	50
2.6.2.2 Ausencia de publicidad	50
2.6.2.3 Posibilidad de integrar publicidad	51
2.6.3 En contra	51
2.6.3.1 Interfaz gráfica menos atractiva e intuitiva	51
2.6.3.2 Menos opciones	52
2.6.3.3 No existe un gestor de contenidos asociado	52
2.6.3.4 Sin servicios de pago	52
2.6.3.5 Dificultad para encontrar un nombre	52

2.6.3.6 No es software libre	52
CAPÍTULO 3. DESCARGA, INSTALACIÓN Y CONFIGURACIÓN	53
3.1 INSTALACIÓN EN NUESTRO EQUIPO	54
3.1.1 En Microsoft Windows	54
3.1.1.1 Descarga	55
3.1.1.2 Instalación	55
3.1.1.3 Ejecución	56
3.1.2 En Ubuntu GNU/Linux	57
3.1.3 Pasos comunes	58
3.1.3.1 Descarga	58
3.1.3.2 Instalación	58
3.2 INSTALACIÓN EN UN SERVIDOR EXTERNO	63
3.2.1 Elección de un servicio de alojamiento	63
3.2.1.1 Características	63
3.2.1.2 Tipos	66
3.2.2 Proceso de instalación	67
3.2.2.1 Preparación previa	67
3.2.2.2 Presentación del cPanel	69
3.2.2.3 Creación de la base de datos	70
3.2.2.4 Subida del sistema	71
3.2.2.5 Archivo de configuración	72
3.3 MIGRACIÓN	73
3.3.1 Migración de los ficheros	73
3.3.2 Migración de los datos	73
3.3.3 Otras consideraciones	77
CAPÍTULO 4. PERSONALIZACIÓN Y ACTUALIZACIÓN	79
4.1 INSTALACIÓN DE TEMAS	79
4.1.1 ¿Dónde encontrarlos?	80
4.1.2 Descarga e instalación manual	81
4.1.3 Descarga e instalación automática	83
4.1.4 Permisos apropiados	84
4.1.4.1 Linux	86
4.1.4.2 Servidor remoto	87
4.1.5 Modificación básica	87
4.2 INSTALACIÓN DE PLUGINS	88
4.2.1 ¿Qué es un plugin?	88

5.6.2 Quick AdSense	127
5.7 COPIAS DE SEGURIDAD	129
5.8 DISPOSITIVOS MÓVILES	130
5.8.1 WPTouch	131
5.9 MEJORA DEL DIÁLOGO	132
5.9.1 WP-RecentComments	133
5.9.2 Subscribe To Comments Reloaded	134
5.10 SEGURIDAD	135
5.10.1 WP Security Scan	135
5.11 MEJORA DEL RENDIMIENTO	136
5.11.1 WP-Cache	137
5.11.2 Buenas prácticas	138
5.11.2.1 Desactiva complementos que no uses	138
5.11.2.2 Reduce peticiones y su tamaño	139
5.11.3 Modificaciones	139
5.11.3.1 Evitar que se generen miniaturas	139
5.11.3.2 Elige qué servicios de notificación usar	140
5.11.3.3 Define llamadas a funciones	140
CAPÍTULO 6. DESARROLLANDO CON WORDPRESS	141
6.1 ¿QUÉ ES UNA API?	141
6.2 INTRODUCCIÓN AL DESARROLLO	142
6.3 LA BASE DE DATOS DE WORDPRESS	145
6.3.1 wp_options	146
6.3.2 wp_usermeta	147
6.3.3 wp_users	148
6.3.4 wp_postmeta	149
6.3.5 wp_posts	149
6.3.6 wp_commentsmeta	151
6.3.7 wp_comments	152
6.3.8 wp_term_relationships	153
6.3.9 wp_term_taxonomy	153
6.3.10 wp_terms	154
6.3.11 wp_links	154
6.4 COOKIES	156
6.4.1 Cookie de usuario	157
6.4.2 Cookie de comentarista	158

INTRODUCCIÓN

La capacidad de difusión en la red ha pasado de ser algo prácticamente anecdótico a sobrepasar a los medios de comunicación tradicionales y, aun así, tener unas perspectivas de crecimiento constantes de cara al futuro. Cada día son más las personas que usan Internet y que lo hacen para actividades que nada tienen que ver con las puramente laborales.

En línea con este espectacular crecimiento de la red de redes, también ha habido un fenómeno que podríamos llamar “la democratización de los medios de comunicación”. De existir un número reducido de personas con capacidad para alcanzar todos los confines de la sociedad, gracias a las tecnologías de la información, cualquiera puede crear un medio de comunicación propio en la red y difundir la información que crea conveniente, de una forma libre, sencilla y gratuita.

La forma más común de hacerlo hoy en día es mediante los *blogs* o cuadernos de bitácora, cuya puesta en marcha, personalización y manejo son el objetivo central de este libro. Las herramientas existentes, como la que aquí se trata, WordPress, son un claro ejemplo de hasta dónde han evolucionado las llamadas “nuevas tecnologías”. Plataformas realmente sencillas de utilizar que nos proporcionan un medio para escribir sobre cualquier tema, y nos permiten ser leídos por cientos de millones de internautas en toda la esfera.

Sin embargo, y a pesar de lo que pueda parecer a través de estas líneas, no todos los blogs son medios que podrían llamarse periodísticos. Los blogs son una herramienta que puede ser utilizada para multitud de propósitos. Existen blogs de tipo político, muchos de los personajes públicos mantienen uno donde expresan sus ideas sobre temas concretos; corporativos, donde las empresas exponen nuevos

productos o servicios o noticias sobre la empresa; personales o educativos, entre otros muchos. Todos tienen algo en común, además de permitir escribir a sus autores, permiten comentar a los lectores, de forma que se establece un diálogo entre ambos. Los cuadernos de bitácora son uno de los máximos representantes de lo que se denomina la “Web 2.0” y ésta es su característica más relevante.

Antes de terminar esta introducción, me gustaría destacar que gracias a la política modular y de complementos de plataformas de creación de blogs, como WordPress, además de blogs propiamente dichos, es posible desarrollar multitud de soluciones, pasando por portales web tradicionales a revistas electrónicas sin dejar de lado pequeñas redes sociales.

En este libro se darán las instrucciones necesarias para poder crear nuestro propio blog, tanto en un servicio externo como en un equipo propio, para luego optimizarlo y personalizarlo a nuestro gusto. También comentaré soluciones más avanzadas como la creación de temas y complementos propios, sin dejar de lado la puesta en marcha de la monetización del blog o las mejoras pertinentes para conseguir la mayor difusión posible en la red.

Finalmente, como recopilación de todo lo tratado, se explicarán con detalle tres ejemplos prácticos. El primero, la creación de un blog personal; el segundo, la creación de una revista digital, y el tercero, el blog de una empresa u organización.



Figura 1.1. Logo de WordPress

El origen del término inglés proviene de la unión de las palabras *web* y *log*. La primera de ellas es ampliamente conocida por todos y la segunda probablemente la hayáis escuchado en más de una ocasión. Cuando en informática se habla de un *log* nos referimos a ficheros en los que hay presente información de un proceso ordenada de forma secuencial. Es muy común ver los *logs* de un servidor para comprobar posibles errores o, por ejemplo, intentos de ataque. La unión de ambas, *weblog*, se transformó posteriormente en la palabra que hoy todos conocemos, *blog*.

1.1.2 Blogosfera

Con este término nos referimos a todos los que de una forma u otra participan en un blog. Deriva del anglicismo *blogosphere* y en muchas ocasiones se emplea para designar a un grupo concreto de la blogosfera global, como puede ser la blogosfera hispana o la blogosfera “linuxera”.

1.1.3 Blogging

Blogging es un anglicismo que engloba a todas las actividades que se desarrollan alrededor del mundo de los blogs. No existe una traducción directa al español del término, pero es muy utilizado por toda la blogosfera hispana.

1.1.4 Microblogging

También puede encontrarse el término *microblogging*, subgénero de los blogs popularizado por el servicio Twitter en el que, a diferencia de los blogs tradicionales, los mensajes pueden tener como mucho un número determinado de caracteres, normalmente 140 ó 160 como los mensajes cortos de texto de los móviles.

Los *microblogs*, debido a su naturaleza, están más próximos a un diario de la vida de cada uno y de lo que está sucediendo a su alrededor que los blogs originales. Asimismo, son un excelente medio de comunicación en catástrofes o eventos de gran importancia porque los usuarios actúan como reporteros de forma prácticamente inmediata.

Se han hecho muy famosos por el uso de personajes públicos de los mismos, o por su uso extensivo en el sismo que sufrió Chile el 27 de febrero de 2010 o durante eventos de gran relevancia como la Copa del Mundial de Fútbol de ese mismo año. Además de Twitter, existen otras alternativas similares que han tenido menos éxito, como un proyecto libre creado por españoles llamado Jisko.



Figura 1.2. Logo de Twitter

1.2 ORÍGENES DEL BLOGGING

El 17 de diciembre de 1997, Jorn Barger, un estadounidense del estado de Ohio, comenzó a escribir entradas en un sitio web al que llamó “Robot Wisdom Weblog”. Ésa fue la primera aparición del término *weblog* en la red. Dos años más tarde, Peter Merholz escribió en la barra lateral de su cuaderno de bitácora, Peterme.com, la frase “we blog”, en la que separó el término acuñado por Barger en dos. Puesto que el vocablo *we* en inglés significa “nosotros”, *blog* se utilizó en un comienzo como verbo para designar la acción de escribir en un *Weblog*. Sin embargo, con el tiempo el término se popularizó y en la actualidad en castellano sólo se usa como sustantivo. En inglés sí se sigue empleando de cualquiera de las dos formas.

Aunque el término es más reciente, la forma de escribir información en un sitio web secuencialmente y establecer conversaciones con los lectores es anterior en el tiempo.

Podríamos decir que los antecesores de los blogs fueron en un comienzo las listas de correo, para posteriormente en la década de los noventa popularizarse los foros. Los foros incorporaban conceptos como los hilos de conversación de un tema que los acercaban algo más a lo que es un blog.

Aunque no hay bibliografía de referencia al respecto, se acepta popularmente en la red como primer blog de la historia el sitio web “Justin’s Links from the Underground”, que salió a la luz en 1994. Su creador, Justin Hall, era por aquel entonces un estudiante de periodismo en Swarthmore College, Pensilvania. A pesar de que en un inicio lo enfocó a compartir enlaces de interés, con el tiempo, el sitio derivó en un diario personal en línea. A partir de ahí, se popularizaron los diarios en línea en forma de página web.

Tras unos años de leve crecimiento, el nacimiento de plataformas como Open Diary, en octubre de 1998, o la archiconocida Blogger, creada por Pyro Labs

en agosto de 1999, que posteriormente, en 2003, sería adquirida por Google, democratizaron la creación de blogs para el gran público. El número de blogs en la red aumentó de forma exponencial.



The Official **Google** Blog

Insights from Googlers into our products, technology, and the Google culture.

Search This Blog

Buscar

con la tecnología de Google™

Site Feed

Google

com readers

Archives

Archives

More Blogs from Google

Visit our [directory](#) for more information about Google blogs.

Follow Google on Twitter

Visit our [Twitter directory](#) for more information.

Sign up to get our posts via email. No more than one message per day.

Subscribe

Delivered by [FeedBurner](#)

Received: 7/18/2010
From: Trichelle
This isn't really a question but a great story I thought Google would be interested in hearing. Today I called my daughter in St. Louis and found out her friends that were suppose to be coming to see her in St. Louis were stuck in Chicago because the driver's keys and wallet were lost. They searched everywhere cancelled credit cards and tried to have a new car key made....but

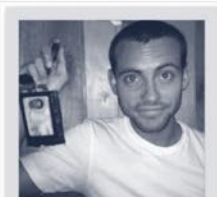


Figura 1.3. Aspecto del blog oficial de Google, basado en la plataforma Blogger

Como con todo tema candente de la red, la comunidad de usuarios de Internet comenzó a desarrollar manuales y tutoriales de cómo crear blogs en plataformas de este tipo. Otros factores, como la monetización de los mismos, que permitía a sus autores conseguir unos ciertos beneficios por las visitas que recibían a través de sistemas como Google AdSense, ayudaron a que el hecho de tener un blog, en la actualidad, sea común en personas que no son expertos, ni siquiera usuarios avanzados de ordenador.

Finalmente, llegó la maduración del mundo del blog en el que ahora nos encontramos. Tecnologías como RSS, del inglés *RDF Site Summary* “sumario del sitio en RDF” (aunque posteriormente también se ha identificado con *Really Simple Syndication* “sindicación realmente simple”), han permitido que podamos estar al tanto de las actualizaciones de cientos de blogs sin tener que entrar a diario

en cada uno de ellos. Otro factor importante para la maduración del mundo blog es la posibilidad de utilizar sistemas gestores de contenido libre, como es el caso de WordPress, que es el tema central de este libro, o Movable Type. Además del hecho de que existan empresas en todo el mundo, como la española Weblogs S. L., cuyo modelo de negocio es la creación y explotación de blogs de forma profesional.

Es imposible dar cifras reales del número de blogs en activo en la red, pero si sois usuarios frecuentes de Internet ya sabréis la importancia que tienen los blogs. La aparición de otros servicios, como el *microblogging* o las redes sociales, no ha supuesto una merma en la importancia de éstos, sino que forman una simbiosis entre todos retroalimentándose los unos a los otros.



Figura 1.4. Logo estándar de "facto" de la tecnología RSS

1.3 WORDPRESS

Ya estamos puestos en situación y podemos comenzar a hablar del tema concreto que ocupa este libro, WordPress. WordPress puede utilizarse como un servicio gratuito de blogs en línea o como un sistema gestor de contenidos propios. En este libro nos centraremos en el segundo aspecto, que es el que más juego y libertad nos da. La creación de blogs en el servicio gratuito en línea la trataremos en el siguiente capítulo.

Antes de continuar, comentaré brevemente cómo saber si un blog o sitio web está implementado haciendo WordPress. De esta forma, vosotros mismos podréis ver la cantidad de sitios que hacen uso de él. En ocasiones, el administrador puede borrar esa línea de código, por lo que nos daría un falso negativo, es decir, si estaría implementado con WordPress, pero no aparecería el código que os muestro

a continuación. Estos casos son verdaderamente escasos y el autor estaría infringiendo la licencia de WordPress, la licencia GPL (*General Public License*, “Licencia Pública General”).

El procedimiento que hay que seguir es el siguiente. Una vez estéis en el sitio en si tenéis que ver el código fuente de la página. El atajo de teclado para esta labor es pulsar simultáneamente la tecla “Control” y la tecla “U”. Si no, en Mozilla Firefox accedéis al menú “Ver” y ahí a la opción “Código fuente de la página”. En Google Chrome, a la derecha de la barra de menús, en el botón que tiene un folio con una esquina doblada, nos dirigimos a “Opciones para desarrolladores” y ahí “Ver código fuente”. Por último, en Microsoft Internet Explorer, en el menú “Página” aparece “Ver código fuente”.

Dentro del código que os aparece, deberéis buscar el siguiente código:

```
<meta name="generator" content="WordPress" />
```

Para buscar podéis usar el atajo de teclado “Control+F” o “Control+B”, el primero para Firefox y Chrome y el segundo para Internet Explorer; o en edición “Buscar”. Si aparece estaréis seguros de que el sitio utiliza WordPress. El contenido del atributo *content* puede cambiar. Por ejemplo:

- WordPress 3.0: Indica que utiliza WordPress y en este caso la versión 3.0.
- WordPress.com: Indica que utiliza los servicios gratuitos de edición de blogs de WordPress.

1.3.1 Breve historia

Los inicios de WordPress se remontan a mayo de 2003. El día 27 de ese mes se anunciaba el lanzamiento de la primera versión para uso del público, la 0.7. Ese lanzamiento fue el colofón a los esfuerzos de Matt Mullenweg y Mike Little para crear una variante mejorada b2/Cafelog, un sistema gestor de contenidos para blogs que ellos conocían, pero que tenía ciertos defectos que decidieron arreglar.

De los dos creadores, el que luego ha continuado con el proyecto más intensamente, y actualmente es su director, es Matt Mullenweg (podéis leer el blog de Matt en <http://ma.tt/>). Su primera experiencia con b2/Cafelog la tuvo el año anterior, 2002, cuando lo comenzó a utilizar para publicar sus fotos y experiencias de un viaje que había realizado a Washington. En ese momento comenzó a modificar el código de la plataforma. Fueron tales las mejoras que hizo que con el

tiempo decidió crear un *fork* (una variante independiente del proyecto primitivo), al que posteriormente se unieron más desarrolladores.



Figura 1.5. Matt Mullenweg participando en el Evento Blog 2006 en España

Como nota curiosa hay que comentar que el proyecto se llama WordPress por la sugerencia de una amiga de Matt, Christine Selleck. Asimismo, de forma parecida con los nombres de distribuciones de GNU/Linux como Ubuntu, los nombres de las nuevas versiones de WordPress guardan una relación entre ellos, todos pertenecen a músicos de jazz.

Las versiones se han ido sucediendo con el tiempo gracias al esfuerzo desinteresado de multitud de desarrolladores, además de Matt, entre los que me gustaría destacar nombres como el de Ryan Boren y Donncha O Caoimh.

En el momento en que se escribe este libro la versión más reciente es la 3.0.1 y el número total de descargas de la rama 3.0 desde su lanzamiento, el día 17 de junio, a agosto de ese mismo año, supera los 12 millones.

Aprovechando el éxito de su software y del propio mundo del *blogging*, Matt Mullenweg también creó la compañía Automattic para ofrecer un servicio gratuito de blogs al estilo de Blogger, pero basándose en su sistema gestor de

contenidos. Hoy en día, tanto Blogger como WordPress.com son los líderes mundiales de su mercado.

1.3.2 ¿Qué es un sistema gestor de contenidos?

Es una palabra que ha aparecido en repetidas ocasiones en este texto y veo conveniente aclarar. Los sistemas gestores de contenidos, o de gestión de contenidos, también conocidos por sus siglas en inglés CMS (*Content Management System*), son programas que permiten la creación y gestión de sitios web de forma fácil e intuitiva, sin necesitar conocer interioridades técnicas o lenguajes de programación.

Aunque en este texto siempre nos referimos a sistemas gestores orientados al mundo de los blogs, hay software de este tipo para todo tipo de necesidades. A continuación voy a enumerar, por temas, los más populares:

- Generales: Joomla!, Drupal y PHP-Fusion.
- Foros: phpBB, MyBB y SMF.
- Blogs: El propio WordPress y Movable Type.
- Comercio electrónico: OsCommerce, Magento, ZenCart y PrestaShop.
- *E-learning* (educativos): Moodle y ATutor.
- Wikis: MediaWiki (el que usa Wikipedia), DokuWiki y PmWiki.
- Imágenes: Coppermine y 4images.

Los hay más específicos aún, pero sólo con esta muestra espero que satisfagáis vuestras necesidades si queréis hacer en un futuro algo diferente a un blog.

1.4 ¿POR QUÉ WORDPRESS?

Como habéis visto, WordPress no es el único sistema gestor de contenidos que existe, ni siquiera en su rama, la de los blogs. De hecho, WordPress es la mejora de un sistema anterior, b2/Cafelog. ¿Por qué he decidido hablar de WordPress en lugar de hacerlo de cualquier otro? Los sistemas gestores de tipo

general, como Drupal o Joomla!, también entran dentro de esta discusión porque se pueden adaptar a la filosofía blog fácilmente.

Mi experiencia con distintos sistemas a lo largo de muchos años siendo un participante activo (autor en ocasiones y lector en otras) de blogs de todo tipo, me ha hecho confirmar las ventajas que muchos atribuyen a WordPress y hoy en día lo hacen líder dentro de su campo.

1.4.1 Instalación, actualización y manejo simple

A diferencia de otros CMS (en adelante, para no repetir tanto “sistema gestor de contenidos”, utilizaré la abreviatura en inglés), la instalación se lleva a cabo a través de una interfaz gráfica, donde deberemos indicar los datos de conexión y autenticación al sistema gestor de base de datos que tratará la información de nuestro sitio, además de indicar el título del blog. Una vez terminada la instalación, podemos seguir configurando el sistema a través de un escritorio simple e intuitivo.

Éste es uno de los puntos a favor de WordPress. El escritorio es mucho más intuitivo que el de Drupal y mucho más simple que el de Joomla!, sin perder un ápice de funcionalidad.

Funcionalidades avanzadas como la instalación de *plugins*, complementos o añadidos en español, podemos hacerla directamente desde la interfaz gráfica a base de clic.

Temas engorrosos en otros CMS, a la vez que vitales para preservar la seguridad de nuestro sitio, como son las actualizaciones tanto del propio WordPress como de los temas gráficos y complementos, son prácticamente automáticos. Además, el sistema nos avisa cuando hay una versión nueva, por lo que nos despreocuparemos en buena medida de este tema.

Los complementos y temas gráficos también suelen tener interfaces gráficas propias para hacerlos fácilmente configurables para todos.

1.4.2 Código fuente libre y entendible

El hecho de que el código de WordPress sea libre gracias a la licencia bajo la que se distribuye, la GPL, deriva en una serie de ventajas para todos los que lo usamos.

La primera de ellas es la gratuidad. Aunque el hecho de que sea libre el software no implica automáticamente que sea gratuito, como, por ejemplo, ocurre en el caso de Red Hat Enterprise Linux, en el caso de WordPress y en muchos otros sí es cierto. En un mercado cada día más vigilado, el software libre es una excelente oportunidad para usar software de calidad de forma gratuita y legal.



Figura 1.6. Logo del proyecto GNU, pionero en el software libre

Otro punto importante es la independencia tecnológica, que a su vez implica soporte y compatibilidad a largo plazo. No dependemos de una sola empresa o corporación que mañana puede decidir que cierra el proyecto. El hecho de que el código sea de libre acceso para todos garantiza que tengamos nuevas versiones, gracias al apoyo conjunto de la comunidad y de todos los interesados en que un proyecto tan grande y popular como éste avance: sus usuarios. Entre estos usuarios podemos enmarcar grandes empresas que invierten dinero en mejorarlo y adaptarlo.

También es importante destacar que el software libre usa formatos abiertos y estándares internacionales. En el caso de WordPress, podemos señalar el uso de XML para los ficheros de exportación de contenido. Aparte de la seguridad añadida que proporciona el saber que cualquier agujero de seguridad será solucionado por la comunidad de desarrolladores en un tiempo récord, en comparación con un proyecto de código cerrado mantenido por una organización.

Hay que tener en cuenta que la propia comunidad testea reiteradamente la seguridad del sistema.

Todo esto es posible gracias a que el proyecto nació siendo software libre y seguirá siéndolo para siempre, gracias a las cláusulas de su licencia. Aunque pueda parecer algo propio de idealistas, la idea de liberar el código, además de favorecernos a nosotros, también es una fuente de ingresos para su propio creador. Gracias al desarrollo tan importante que ha tenido el proyecto desde sus comienzos, ha llegado a tener las cotas de popularidad de las que goza hoy en día y su creador lo explota comercialmente gracias al servicio de blogs gratuitos en línea WordPress.com. Es otra forma de explotar el software, en lugar de venderlo como si se tratara de un producto con su licencia, vender un servicio adherido a un proyecto de carácter libre como éste.

Aparte, también añadido, que el código del proyecto es entendible. Cuando un proyecto llega a tener un tamaño tan considerable como en este caso, es complicado que los menos expertos en este tipo de labores puedan entender y leer el código con pocos problemas. De mi experiencia con proyectos similares, como Drupal o Joomla!, tanto en sus versiones 1.5.x como en sus versiones primeras 1.0.x, os puedo asegurar que, por lo general, el código está mejor estructurado y es más legible.

Antes de seguir, hay que comentar que una de las mayores críticas que se le hacían a este respecto a WordPress, es que no utilizaba orientación a objetos, cuando ya estaba disponible en la tecnología en que se implementa, PHP. Con la llegada de la rama de versiones inaugurada por la 3.0, el código se reescribió convirtiendo en clases y objetos la estructura interna, pero asegurando la compatibilidad con *plugins* y temas de versiones anteriores.

De los CMS libres más populares, pocos se implementan usando orientación a objetos (el único de los famosos, aparte de WordPress, es la nueva rama de Joomla!: la 1.5.x).

1.4.3 Popularidad

Creo que todos coincidiréis conmigo en que tener la mayor popularidad en su segmento es un factor a favor y no en contra. El aspecto negativo puede ser la pérdida de distinción respecto de la mayoría, pero gracias al elevado grado de personalización, esto no es un problema.

Por el otro lado, la gran popularidad se traduce en una mayor cantidad de complementos, temas gráficos y documentación disponible; además de en un desarrollo y resolución de agujeros de seguridad más veloz.

1.4.4 Integración en motores de búsqueda

Las visitas de un blog (y de cualquier sitio web en general) podemos dividir las en las siguientes categorías. Las que son directas, es decir, usuarios que entran directamente a tu sitio tras introducir la dirección en la barra de su navegador, o mediante favoritos. Las que son a través de un enlace, es decir, una tercera web, que no es un motor de búsqueda, donde se encuentra el usuario, tiene un enlace que apunta a tu sitio, y a partir de ahí accede. Y, finalmente, las visitas que vienen de enlaces que aparecen en los resultados de búsquedas.

Aunque puntualmente los dos primeros factores pueden ser relevantes, a largo plazo las visitas de un blog dependen directamente de la cantidad de visitas que reciben de los motores de búsqueda, principalmente Google, Yahoo! y Bing.

Para facilitar la vida a los buscadores, existen las llamadas técnicas SEO *Search Engine Optimization* “optimización de motores de búsqueda”, que posibilitan que a igualdad de contenido tengamos muchas más apariciones en los resultados de las búsquedas y, por lo tanto, más visitas desde los buscadores. WordPress integra por defecto multitud de técnicas SEO para conseguir desde cero una óptima integración en nuestras más relevantes fuentes de visitantes.



Figura 1.7. Logo de Google, líder en el mercado de buscadores

1.4.5 Seguridad

La seguridad es fundamental en un sitio web. Si tras muchas horas de trabajo, miles de visitas acumuladas con campañas publicitarias que marchan correctamente dejando ingresos económicos en el blog, no tenemos una seguridad adecuada, podemos sufrir un ataque que cuando menos deje el blog fuera de servicio y, en los peores casos, nos haga perder toda la información que hay en el mismo.

WordPress se ha caracterizado desde sus inicios por su seguridad y la falta de agujeros importantes en su estructura. En cuanto se han detectado, en cuestión de días el equipo de desarrollo ha sacado una nueva actualización que ha permitido solucionarlos. Esto unido a lo que hemos comentado antes, la facilidad de actualización, permite tener un blog fortificado sin necesidad de conocer profundamente la tecnología y sin tener que preocuparnos por este aspecto.

1.4.6 Protección contra el spam

Otro mecanismo de prevención en el que WordPress tiene una mejor valoración que sus rivales es en el de protección contra el spam. El spam, por si no conocéis el término, son los mensajes ni solicitados ni deseados, normalmente publicitarios, que se envían en grandes cantidades, en el caso de los blogs en forma de comentarios, y que nos perjudican porque llenan nuestro sitio de publicidad y, por otro lado, ocultan los comentarios reales. El gran responsable del éxito contra el spam en WordPress es un complemento que trae instalado por defecto, Akismet, creado por Matt Mullenweg.

1.4.7 WordPress.com

No son pocos los usuarios que comienzan sus pasos en WordPress.com para luego salirse de la plataforma en línea y crear su propio blog sin limitaciones en su propio alojamiento. Al tratarse del mismo entorno, también es el sitio perfecto para que los principiantes se suelten.



Figura 1.8. Aspecto de WordPress.com

1.4.8 Otras ventajas

Para terminar, agrupamos en un mismo punto otras más breves, pero no menos importantes:

- Multisitio: Puedes ofrecer la creación de otros sitios a tus lectores o amigos en tu misma instalación.
- Red social: El complemento BuddyPress nos permite crear una red social en nuestro WordPress. Es decir, los usuarios podrán crear grupos de amigos, comunicarse entre ellos... una pequeña red social en nuestro sitio.
- Integración con móviles: WordPress se visualiza y se gestiona a la perfección desde las principales plataformas móviles. Dado que este tipo de plataformas están en pleno crecimiento, es de vital importancia este apartado.
- No sólo blogs: Ya lo comentamos desde un comienzo. Aunque WordPress no está orientado a los sitios de cualquier temática, mediante temas gráficos, complementos y adaptaciones, podemos convertir nuestro blog en un sitio de cualquier temática: foros, redes sociales, portales, etc.

1.5 ALTERNATIVAS

Para terminar este capítulo, trataremos brevemente qué alternativas hay a WordPress en el panorama del software libre. Para poder juzgar con objetividad este tipo de plataformas, hay que conocer a sus competidoras y hacer un análisis de ventajas y desventajas como hemos hecho en el punto anterior.

Hablaremos de un sistema gestor orientado a *blogging*, Movable Type, y luego comentaremos los dos grandes competidores de WordPress a nivel global: Joomla! y Drupal. Aunque no sean competidores directos, con WordPress se pueden hacer sitios web de propósito general y tanto con Drupal como con Joomla! se pueden hacer blogs, por lo que tienen cierto grado de relación.

1.5.1 Movable Type

Tras WordPress, Movable Type (<http://movabletype.org/>) es el CMS orientado a blogs más popular en la red. Actualmente es mantenido por la compañía californiana Six Apart y la ayuda de los desarrolladores libres, ya que se trata de un proyecto de software libre que fue iniciado por Ben Trott y que vio la luz de forma pública por primera vez el 8 de octubre de 2001.

Las últimas versiones de este software tienen poco que envidiarle a WordPress y, en los últimos tiempos, las novedades de éste han sido posteriormente implementadas en Movable Type. Incluye muchas funcionalidades avanzadas, entre las que podemos destacar la posibilidad de gestionar varios sitios desde la misma instalación e integración con herramientas LDAP, propia de sistemas con servicios de directorio activo en marcha.

La principal diferencia con WordPress es el público objetivo. Mientras que MovableType está mucho más orientado al administrador de sistemas y al usuario avanzado en general, como demuestra la integración con LDAP o la posibilidad de utilizar Perl para gestionar ciertos aspectos, WordPress es más sencillo de instalar y poner en marcha. Ésta, probablemente, sea la razón por la que en la actualidad exista una diferencia tan grande entre los dos.



Figura 1.9. Logo Joomla!

1.5.2 Joomla!

Joomla! es el CMS de código libre de propósito general más famoso hoy en día en la red. Nació como una escisión de Mambo, un CMS privado anterior, cuando los propietarios de éste decidieron orientar su software hacia perspectivas más cerradas y comerciales. La escisión se produjo el 17 de agosto de 2005 y tras estos años el proyecto ha conseguido brillar por sí mismo.

El hito más importante en este tiempo fue abandonar la vieja estructura heredada de Mambo, que era la base de las versiones 1.0.x, para estrenar una moderna estructura interna orientada a objetos en las nuevas versiones 1.5 en adelante, aprovechándose de las últimas mejoras en la tecnología PHP, que todos los CMS que estamos viendo comparten.

Las dos principales características de Joomla! son la versatilidad y la facilidad de uso. Joomla! es verdaderamente versátil y tras instalarlo podemos

orientarlo de forma inmediata a diferentes necesidades. Por otro lado, tanto la instalación como la administración del sistema son verdaderamente simples e intuitivas y, por estas razones, se ha convertido en la estrella de los desarrollos web para empresas por parte de muchos profesionales.

Como nota contraria, podemos apuntar la pesadez del sistema en comparación con otros CMS similares, como pueden ser Drupal, del que hablamos a continuación, o el propio WordPress.



Figura 1.10. Logo Drupal

1.5.3 Drupal

Finalmente, comentaremos de forma breve Drupal, otro sistema gestor de contenido libre orientado a temática general. Creado por Dries Buytaert en enero de 2001. Lo primero que diría de Drupal es que es el CMS libre más serio que conozco.

A diferencia de Joomla! o WordPress, que tiene un trato más amigable con el usuario novel, Drupal está mucho más orientado al usuario avanzado y al profesional. De hecho, en ocasiones, en vez de ser considerado CMS es considerado *framework* por sus usuarios. ¿Qué quiere decir esto? Que ven Drupal más que como un sistema gestor sobre el que actúan de forma directa, como un sistema sobre el que ellos desarrollan posteriormente sus aplicaciones. Está todo tan estructurado y modulado que incluso los complementos siguen una política de dependencias y jerarquías propias de un sistema de paquetería de una distribución GNU/Linux.

Como nota curiosa, apuntar que es el sistema que está detrás del actual portal de la Casa Blanca (<http://www.whitehouse.gov/>) o de Data.gov.uk, el portal del gobierno británico para compartir datos de interés para la población del Reino Unido. También es la base de muchos blogs en la actualidad.

WORDPRESS.COM

Como he comentado anteriormente, Matt Mullenweg, además de creador del proyecto de software libre, es el actual presidente de Automattic, una empresa creada por él mismo dedicada a la explotación comercial como servicio de su sistema de blogs. Mediante esta empresa ofrece un servicio gratuito de blogs en línea, llamado WordPress.com, que utiliza WordPress como base.

En este capítulo, veremos cómo crear nuestro primer blog en línea, comentaremos los principios básicos de todos los blogs y terminaremos haciendo un repaso de los principales competidores en este sector.

2.1 NUESTRO PRIMER BLOG

¿Por qué elijo WordPress.com para crear nuestro primer blog en línea? Aunque hay factores que inclinan la balanza de este tipo de servicios hacia Blogger, la plataforma de Google de la que ya hemos hablado, hay un factor básico que hace que haya elegido WordPress.com para este capítulo: la similitud entre el sistema que nos presenta y el sistema que luego utilizaremos cuando manejemos una instalación de WordPress en nuestro propio equipo realizada desde cero.

Además, con el tiempo, las ventajas o desventajas de los servicios de blogs en línea quedan atrás porque, al estar alojados en servidores externos y tenernos que ceñir a unas cuadrículas definidas por la empresa que nos da el servicio, todos los grandes *bloggers* terminan migrando su blog hacia un sistema alojado en un servidor propio. WordPress.com en este punto también tiene otra ventaja: permite una migración directa a un blog creado con el CMS en un servidor propio.

De las ventajas y desventajas de WordPress.com en comparación con Blogger, su máximo competidor, hablaremos más profundamente en el último punto de este capítulo.

2.1.1 Alta de usuario y creación del blog

Lo primero que debemos hacer es acceder con nuestro navegador favorito a la dirección de WordPress.com: <http://www.wordpress.com>. En caso de que no aparezca por defecto en español, lo elegimos en un desplegable que tiene por título “Language”. Hacemos clic sobre “Regístrate ahora”, rellenamos los datos que nos pide y ya tendremos una cuenta de usuario.

El siguiente paso será crear el blog, de los datos que nos pide los más relevantes son el título y el dominio del blog, es decir, la dirección de acceso que tendrá. Las direcciones de los blogs gratuitos son siempre de la siguiente forma: http://dirección_elegida.wordpress.com. Elegid un buen nombre en este punto porque será uno de los factores clave para tener una buena impresión entre el público. Las otras opciones que hay son el lenguaje y la privacidad, lo lógico, a no ser que queramos tener un blog interno empresarial, es dejarlo en público.

Dominio del Blog

aprendiendoblog.wordpress.com

You may later choose to use your own domain name, such as myblog.com, through our domain mapping upgrade.

Título del blog

Aprendiendo WordPress

El nombre del blog puede cambiarse en cualquier momento.

Idioma

¿Qué idioma será el principal cuando escribas en tu blog?

es - Español

Privacidad

☒ **Public:** My blog should appear in search engines like Google and Bing, and in public listings around WordPress.com.

☐ **Block search engines:** Only allow normal visitors.

☐ **Private:** My blog should be visible only to users I choose.

[Crear Blog →](#)

Figura 2.1. Formulario de creación del blog

Para el ejemplo que desarrollaré a lo largo de este capítulo he elegido los siguientes datos, como podéis ver en la ilustración anterior, el resto lo he dejado por defecto:

- Dominio del blog: *http://aprendiendoblog.wordpress.com*.
- Título del blog: Aprendiendo WordPress.

Una vez hayáis rellenado los campos, hacéis clic sobre “Crear blog” y ya tendréis el blog en vuestro poder, preparado para ser visualizado por todos los usuarios de la red. Es posible que el dominio de vuestro blog esté ocupado, por lo que deberéis probar con nombres que os gusten y que sean fáciles de recordar hasta dar con el vuestro. Por esto mismo, os pido que si abris un blog y posteriormente lo abandonáis vacío, pidáis la baja del mismo para posibilitar que un tercero pueda usar ese nombre y no se quede bloqueado de forma indefinida.

El aspecto inicial del blog es el que podéis ver en la ilustración siguiente. Me gustaría destacar la presencia de la barra superior, que sólo veremos si hemos iniciado sesión en WordPress.com y desde donde tendremos acceso directo a los puntos más interesantes de gestión de nuestra cuenta y de nuestros blogs: *escritorio, escribir una nueva entrada, moderar comentarios, ver las estadísticas del blog y el acceso directo al blog en sí.*



Figura 2.2. Aspecto inicial del blog

2.1.2 Explorando el escritorio

Acabamos de poner en marcha el blog y tenemos ya visible para todo el público, a través del dominio que hemos elegido, un sitio web en el que comenzar a escribir, recibir comentarios y, sobre todo, con el que aprender el funcionamiento básico de un blog.

El aspecto estético está basado en la plantilla Twenty Ten, también llamada “2010”. Profundizando en este tema hay que comentar que ha supuesto un gran cambio respecto a la anterior plantilla por defecto, llamada “K2”. Según un comunicado del propio equipo de desarrollo: “El objetivo del tema es ser agradable, personalizable, sencillo y, sobre todo, legible”. Aunque la mayoría de los usuarios eligen posteriormente modificar o cambiar de tema, el tema por defecto supone la primera impresión que recibe el usuario de la plataforma y es de mucha importancia. Podéis leer más sobre él en un blog dedicado *ex profeso* al tema gráfico en: <http://2010dev.wordpress.com/>.

Volviendo a lo que nos interesa en este punto, comencemos accediendo al escritorio o *dashboard*, como es denominado en las versiones inglesas. La traducción directa de esta palabra no es exactamente escritorio, sino tablero de mandos y, en efecto, así es. Para acceder nos dirigimos a la barra superior, en “Mis blogs”, escogemos “Aprendiendo WordPress” y, en el menú desplegable que aparece, “Escritorio”.

Nos encontramos ante una interfaz web en la que podemos ver los siguientes grandes bloques:

- Cabecera superior: Con el título del blog, accesos directos a ciertas opciones y enlaces para ir a nuestra cuenta y para cerrar sesión.
- Barra lateral: Acceso a todas las funciones de administración del blog. Se encuentran separadas en tres partes, que agrupan cada una varias opciones que a su vez pueden expandirse o no para acceder a acciones concretas. Está repartido de forma óptima para que el uso no sea complejo, a pesar de la cantidad de opciones presentes.
- Parte central: Variará en función de en qué opción nos encontremos y en ella estarán los formularios para interactuar con la plataforma. Al entrar presenta accesos directos y resúmenes de los aspectos más destacados del blog. Está compuesta por unos cuadros flotantes que podremos mover y minimizar a nuestro antojo.

En lugar de explicar cada una de las opciones de forma secuencial, creo que lo mejor es pasar a hacer una demostración de las acciones básicas que se llevan a cabo en un blog, además de proceder a las configuraciones más relevantes para hacer el desarrollo del texto más ameno.

2.1.3 Nuestra primera entrada

Ya estamos en condiciones de escribir nuestra primera entrada o *post*, palabra con la que los angloparlantes las denominan. Por defecto, WordPress.com ya tiene una entrada introducida en nuestro blog de título “¡Hola Mundo!”, al más puro estilo de los tutoriales de lenguajes de programación.

Si os fijáis, a la derecha, en la parte central de la pantalla que aparece nada más entrar en el escritorio, hay un cuadro flotante de título “Publicación rápida” con el que podremos introducir entradas sin pararnos en los detalles, sólo con indicar el título, el contenido y las etiquetas. ¿Qué es una etiqueta?

2.1.4 Etiquetas y categorías

Las etiquetas, junto con las categorías que descubriremos de inmediato, son los dos principales métodos que tiene WordPress para organizar la información. Los *tags*, como son conocidos en inglés, son palabras clave que definen temas sobre los que trata la entrada, mientras que las categorías ofrecen una organización jerárquica del contenido.

Por ejemplo, en un blog que trate sobre software libre podríamos definir las siguientes categorías: “Noticias”, “Manuales” y “Opinión”. Y en una entrada en concreto que hablara sobre el uso de Ubuntu en la empresa, podríamos elegir las siguientes etiquetas: “ubuntu”, “empresa”, “sistemas operativos”.

Las etiquetas y categorías además de organizar y facilitar el acceso a la información (pensad en blogs que pueden llegar a tener miles de entradas) le facilitan mucho la vida a los buscadores, por lo que su buen uso es una fuente de visitas.

2.1.5 Gestión de las entradas

En lugar de elegir la interfaz de escritura rápida de entradas, vamos a escribir nuestro primer *post* con el método tradicional, que nos da más posibilidades. Antes de eso, aprovecharemos para borrar la entrada de ejemplo y así conocer mejor la gestión de entradas.

En la segunda parte de la barra lateral, desplegamos la opción “Entradas”, elegimos de nuevo “Entradas” y nos aparece una tabla con todas las entradas del blog. Nos situamos encima de la única que hay y en los enlaces que aparecen abajo, al pasar el ratón por encima, elegimos “Papelera”. La entrada no aparecerá, pero estará en una especie de “Papelera de reciclaje” al estilo Windows lista para ser restaurada. Para eliminarla de forma definitiva, accedemos a la papelera (hay un enlace al lado del título “Posts”) y ahí volvemos a repetir la operación. Esta vez la opción que aparece es “Borrar permanentemente”. Al lado de ese mismo título, “Posts”, está el botón “Añadir” para crear nuestra primera entrada.

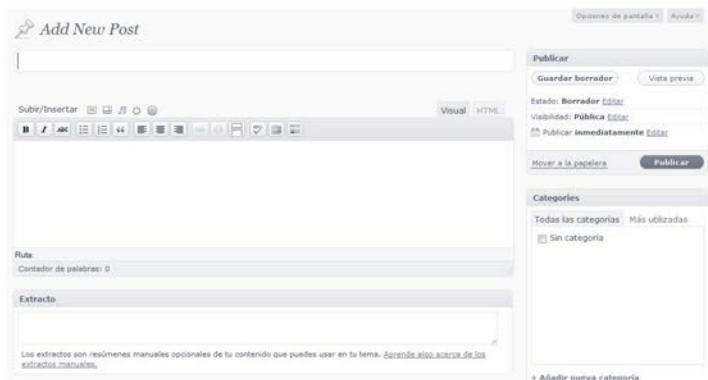


Figura 2.3. Interfaz de creación de entradas

La interfaz de creación de entradas que veis en la figura superior sigue el estilo gráfico de todo el menú de administración, como no podía ser de otra forma. Veamos uno a uno los elementos.

Primero, nos encontramos con el título, el contenido y el extracto. Lo más destacable es que tanto el contenido como el extracto se editan mediante un editor WYSIWYG (*What You See Is What You Get* “lo que ves es lo que obtienes”). No hace falta saber nada sobre HTML ni CSS para manejarlo, aunque si os fijáis en la pestaña superior podéis acceder al código fuente de lo que escribís para cosas puntuales. Incluso incluye un revisor ortográfico.

También tenéis acceso directo a los gestores de contenido multimedia y otros objetos. De izquierda a derecha tenemos: imágenes, videos, audio, objetos y encuestas. El extracto es opcional y yo sólo lo recomiendo en grandes entradas en

las que molesta ver un contenido tan largo en la página principal del blog. Abajo tenéis opciones sobre los comentarios y los llamados *trackbacks*.

Los *trackback*, concepto que ahora mismo no tiene traducción directa, son notificaciones que se realizan a la inversa para comunicar que en nuestra entrada se hace referencia a entradas de terceros. El hecho de referenciar entradas de terceros es muy importante en la filosofía del mundo de los blogs. La especificación de los *trackback* fue creada por los desarrolladores de Movable Type, y su plena integración en WordPress hoy en día es una de sus grandes ventajas respecto a sus rivales. Muy similar a este concepto son los *pingback*, que también los conoceréis con WordPress y que, a diferencia de los primeros, están más orientados a integrarse como si fueran comentarios.

Siguiendo con la descripción de la interfaz de creación de entradas llegamos a las opciones de publicación. Podemos guardar la entrada como borrador o como entrada pendiente de revisión, cambiar la visibilidad (pública, privada o protegida por contraseña) y publicar la entrada en el mismo momento o en otro diferente (podemos programar la aparición de la entrada para una hora y una fecha concretas).

Finalmente, llegamos a los bloques de gestión de categorías y etiquetas que hemos explicado en este mismo capítulo. Podemos añadir nuevas categorías directamente desde aquí (recordad que pueden tener una jerarquía propia y que una misma entrada puede tener más de una categoría). Para terminar, podemos elegir la imagen destacada de la entrada, que aparecerá de forma destacada a la hora de publicar la entrada (esto en realidad depende del tema gráfico que elijamos, hay temas que no admiten esta opción, en breve llegaremos a ese punto).

2.1.6 Publicación

Llegó el momento de publicar nuestra primera entrada. Como ejemplo he preparado un texto que podréis ver en una ilustración a continuación. Lo que queráis escribir o las imágenes que queráis subir es una decisión vuestra, seguro que os interesa mucho un tema en especial y tenéis mucha información que compartir. Escribir sobre un tema en el que estás interesado es algo realmente gratificante, y si con el tiempo llegas a tener visitas, y por lo tanto ingresos, puede incluso ayudar a darte un capricho cada cierto tiempo. Ésta es vuestra primera entrada y os recomiendo que probéis todas las opciones que os da el editor visual.

Una vez estéis preparados para publicar, en el panel de publicación, hacéis clic sobre “Publicar” y la entrada estará visible para todos. Si no queréis escribirla de una vez, está disponible la opción de guardarla como un borrador, y si os

interesa que la entrada no sea visible en un tiempo, programad su salida (por ejemplo, si os vais de vacaciones, dejáis unas pocas entradas programadas y así tiene el blog refresco aunque estéis ausentes).

Nuestra primera entrada en WordPress

Posted on 20 Agosto 2012 by carazo

Esta es la primera entrada que escribo en WordPress, aunque sea corta, vamos a comprobar qué fácil es integrar contenido como imágenes y estilos de todo tipo.



Acerca de carazo

Webmaster de LinuxHispano.net

[View all posts by carazo](#)

This entry was posted in [Aprendiendo](#). Bookmark the [permalink](#). [Edit](#)

Archivos

Categorías

- No hay categorías

Meta

- Administración
- Log out
- RSS de Entradas
- RSS de los comentarios
- WordPress.com

Figura 2.4. Aspecto público de nuestra primera entrada

2.1.7 Contenido estático

Existe contenido en un blog que no ha de ser dinámico, es decir, que no debe ser parte del eje central del sitio que sucede de forma secuencial, sino que debe permanecer accesible separado del resto de forma indefinida. Ejemplos de contenido estático serían:

- Acerca de: Explicación general de los inicios del blog, quién lo dirige y con qué sentido se creó.
- Contacto: Datos de contacto para que los visitantes puedan comunicarse directamente con el autor.

Para editar contenido estático en WordPress deberemos acceder a “Páginas” en el escritorio y actuar de igual forma con las entradas, pero sabiendo que ese contenido está enfocado de esa forma. Para acceder a ellas en nuestro tema

gráfico por defecto, aparecen enlaces debajo de la barra del título que ponen “Inicio”, “Acerca de...”, etc. Gracias a esta posibilidad tenemos un gestor de contenido más generalista de lo que podría parecer en un comienzo si sólo permitiera redactar entradas.

2.2 PERSONALIZACIÓN

Ya que tenemos las nociones suficientes para poder producir y gestionar contenido en el blog, ha llegado el momento de comenzar a personalizarlo a nuestro gusto. Las posibilidades en WordPress.com están mucho más limitadas que en una instalación desde cero, como veremos en los capítulos posteriores.

2.2.1 Temas

Desde hace mucho, en este tipo de desarrollos informáticos, se generalizó el concepto de separar el aspecto del contenido. Modelos formales de desarrollo de software, como el llamado Modelo-Vista-Controlador, así lo exponen, y desde la aparición de los estilos en las tecnologías web, todo fue mucho más simple y podemos mostrar diferentes formas gráficas para la información del blog, incluso dependiendo del tipo de visitante (para ello deberíamos tener una instalación hecha desde cero). A continuación, veremos cómo podemos cambiar por completo el aspecto general de nuestro blog con sólo unos cuantos clics. También podemos personalizar aspectos más detallados, pero eso lo veremos en los próximos puntos.



Figura 2.5. Nuevo aspecto del blog con el tema Motion

Para cambiar el tema gráfico debemos acceder a la opción “Apariencia”, “Temas”. Aparece una pantalla con quince miniaturas de temas listos para ser

elegidos. Podemos buscar más temas dándole al botón “Actualizar” o eligiendo alguna de las opciones que aparecen: “Añadido recientemente” o la que os recomiendo, “Más populares”. El número de temas disponibles es amplio, pero está restringido, cuando lo instalemos en un servidor propio vosotros mismo podréis crear el vuestro desde cero o modificar alguno de los miles que hay en la red listos para descargar libremente.

Cuando hayáis decidido cuál activar, debajo de la miniatura de presentación, elegís la opción “Activar” y os aparecerá un nuevo mensaje en el escritorio que indicará que habéis activado un nuevo tema. Ya podemos ver cómo ha quedado el sitio. Para el ejemplo, he utilizado el tema “Motion” como indico en la ilustración anterior.

Si conocéis CSS (*Cascading Style Sheets* “hojas de estilo en cascada”) podéis modificar el CSS de vuestro tema actual para personalizarlo en la opción que aparece en el menú “Temas” bajo la miniatura del tema que tenemos actualmente elegido: “Editar CSS”. Además, algunos temas incluyen opciones, a las que podréis acceder mediante el enlace situado justo al lado, “Opciones del tema”.

2.2.2 Widgets

Los *widgets* (la traducción al español sería algo así como aparato o artilugio) son unos elementos que podemos insertar en nuestro blog para dotar de mayor funcionalidad al mismo. La funcionalidades van desde mostrar los artículos más vistos a permitir que los usuarios inicien sesión en el sistema, pasando por un gestor de enlaces de relevancia.

Mucho mejor que definirlos es trabajar con ellos. En el menú “Apariencia” la segunda opción es “Widgets”. En el centro de la pantalla aparece un cuadro con todos los *widgets* que tenemos disponibles, debajo de ellos están los inactivos y finalmente, a la derecha, tenemos una barra vacía dividida en los siguientes apartados:

- *Sidebar*: Barra lateral.
- *Footer Left*: Pie de página a la izquierda.
- *Footer Middle*: Pie de página en el medio.
- *Footer Right*: Pie de página a la derecha.
- *Header*: Cabecera.

Estos apartados pueden variar de un tema a otro y muestran los distintos sitios donde posicionar los *widgets*. Vamos a posicionar el *widget* de últimos comentarios en la barra lateral, siempre muy útil para averiguar la última actividad de los lectores en el blog.

Para ello buscamos “Últimos comentarios” en los *widgets* disponibles, hacemos clic sobre él y sin soltar el botón del ratón lo depositamos sobre el espacio en la barra lateral donde viene *sidebar*. El típico movimiento de *drag&drop*, arrastrar y soltar.

Cuando ya está el elemento situado en su destino, hacemos clic sobre la flecha que aparece en la parte derecha del mismo. Se abrirán las opciones propias del *widget*, cada uno tendrá las suyas y nos permitirá definir una serie de parámetros relacionados con la funcionalidad del mismo, para personalizarlo en cierta medida. Veamos en este caso qué opciones nos da:

- Título: Podemos poner “Últimos comentarios” o elegir otro que nos guste más, por ejemplo, “Conversación”.
- Número de comentarios a mostrar: El máximo que nos permite es 15, por defecto vienen 5, dependiendo del número de comentarios ocupará más o menos en la barra lateral, por lo que deberemos ser cuidadosos para que no tape otros elementos de importancia.
- Tamaño del avatar: Todos los usuarios tienen un avatar. Si nunca has utilizado WordPress será uno con la silueta de la cabeza en negro y el fondo gris, en caso contrario, será el que hayas puesto de imagen en tu perfil de WordPress.com. Al lado del comentario aparecerá el avatar del usuario, podemos definir diferentes medidas en píxeles para adaptarlo a nuestro gusto y espacio disponible.
- Color de fondo del avatar.
- Color de fondo del texto.

Cuando hayamos terminado de configurarlo, le damos a guardar y ya podremos verlo en nuestra barra lateral. Para poder seguir trabajando con mayor comodidad, volvemos a hacer clic en la flecha situada a la derecha del *widget* y se replegará para volver a dejar más sitio.

Vamos a insertar otro *widget*, “Nube de etiquetas”. Las etiquetas más utilizadas en nuestras entradas aparecerán con letra de mayor tamaño rodeada de

las demás etiquetas. Formarán una nube de palabras con las etiquetas más usadas cuyo tamaño, como hemos dicho, será proporcional al número de ocurrencias en nuestro blog. Es uno de los *widget* más populares en WordPress.

El procedimiento es el mismo. Arrastramos y soltamos el *widget* que está en el contenedor de “Widgets disponibles” en la barra lateral. Podemos elegir si queremos que aparezca antes o después de “Conversación”, sólo con situar el ratón encima o debajo del mismo. Este mecanismo es muy intuitivo y es otro detalle más que hace a WordPress diferente respecto a otros sistemas gestores de contenido. Sólo tendremos que indicar el título, volvemos a guardar y volvemos a nuestra página principal para ver el aspecto que ha quedado.

En la ilustración de abajo aparece vacío porque no hay etiquetas suficientes, con el tiempo, se irá llenando y además de ser un *widget* muy dinámico también es muy representativo del contenido del blog.

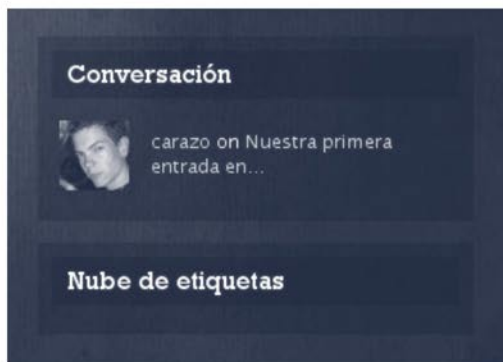


Figura 2.6. Aspecto de la barra lateral con los dos *widgets*

2.2.3 Menús

Los menús son otra forma de personalizar el aspecto. Dependiendo del tema gráfico puede estar activa o no esta sección. En el caso que he elegido no lo está, pero vamos a explicar cómo se utilizaría si el tema nos lo permitiese.

El término “menú” es ampliamente conocido por todos los que son usuarios de ordenador. Se trata de agrupaciones de elementos que se muestran unos junto a otros, de forma que podemos plegarlos o desplegarlos para acceder a estos.

En WordPress los menús suelen utilizarse para mostrar accesos directos a las páginas estáticas, a las categorías, a los autores o a vínculos personalizados.

Aunque el tema no lo permita nativamente, siempre podremos utilizar un *widget* que incluya un menú dentro, y es lo que vamos a hacer en este caso. Si el tema los permitiera, podríamos colocarlos, por ejemplo, en la cabecera de la página.

Lo primero será acceder a la opción “Menús” dentro del apartado “Apariencia” en el escritorio. Creamos un nuevo menú, en el ejemplo le he dado de nombre “Páginas”, que albergará enlaces a las páginas estáticas. Una vez creado, en la parte izquierda, tenemos las distintas opciones para rellenarlo. Elegimos “Pages”, “Ver todo” y marcamos las dos que salen: el inicio y la página de “Acerca de” que hemos creado antes.



Figura 2.7. Aspecto del **configurador** de widgets de la barra lateral

Hacemos clic en “Add to menu” y ya estarán incluidas. Salvamos el menú con el botón que pone “Save menu” y nos dirigimos a la gestión de *widgets*. Elegimos el *widget* de nombre “Custom menu” y lo colocamos en la barra lateral. Le damos un título, elegimos qué menú es y guardamos. Si volvemos a la página principal veremos el resultado.

Hago un inciso para comentar que, como habréis podido apreciar, la traducción a español en WordPress.com no es completa. En la instalación desde cero sí suele estar al cien por cien, aunque es posible que siempre queden botones, frases, etc., que no lo estén.

2.2.4 Otros

Podemos manejar de forma gráfica otros elementos de la apariencia gráfica de nuestro sitio con las opciones presentes en el menú “Apariencia”. Dependerá del tema que estemos utilizando el número de opciones que aparezcan, pero una bastante interesante es la posibilidad de modificar la imagen de la cabecera (de la que viene por defecto a una propia), presente en prácticamente todos los temas.

2.3 COMENTARIOS

Aunque podemos deshabilitar los comentarios en todas o en ciertas entradas, lo normal es que los dejemos activados. A partir de ahí pueden venir problemas derivados de un mal uso del sistema de comentarios (insultos, contenidos inapropiados, enlaces a sitios ilegales...) o el spam.

Como ya hemos repetido en multitud de ocasiones, WordPress tiene un poderoso sistema de protección contra el spam, pero, por desgracia, alguna vez puede haber un falso negativo y saltarse la barrera que interpone Akismet. También es posible que se produzcan insultos o peleas verbales entre los usuarios del blog, por lo que en cualquiera de estas circunstancias es necesario saber eliminar los comentarios. Otra situación típica es la contraria, los falsos positivos. Un comentario correcto es detectado como pernicioso por Akismet. Deberemos aprobarlo para que se vea en el sitio.

Nos dirigimos a comentarios y veremos todos los comentarios del sitio agrupados en las siguientes categorías:

- Pendientes: Comentarios que no son calificados como spam, pero que están esperando a ser aprobados o eliminados por el autor del blog.
- Aprobados: Comentarios que están ahora mismo visibles para todo el mundo.
- Spam: Comentarios que Akismet ha considerado que son spam. No son visibles para el público, deberemos aprobarlos primero para que pasen a pendientes y luego aprobarlos de nuevo para que pasen a visibles. Estos dos pasos son necesarios, porque un comentario que no es spam y Akismet lo ha visto como tal quizás no sea apropiado para estar aprobado por otras razones.
- Papelera: Similar a la papelera de las entradas.

La adecuada gestión de los comentarios en un blog es vital para el éxito del mismo.

2.4 PUNTUACIONES

Esta opción ha sido incluida de forma reciente en WordPress.com y permite que los visitantes del sitio puedan valorar tanto las entradas como los comentarios y las páginas estáticas.

Para activar esta posibilidad deberemos dirigirnos al menú “Calificaciones”, donde podremos configurarlas o ver los informes. Las valoraciones son de 1 a 5 y los usuarios sólo tienen que hacer clic sobre unas estrellas que se iluminan. En la configuración podemos elegir dónde aparecen (encima o debajo, básicamente) y para qué elementos se muestran (comentarios, entradas o páginas).

En opciones avanzadas podemos modificar el sistema de calificación sustituyendo el de las estrellas que van de 1 a 5 por un sistema binario en el que los usuarios comunican si les ha gustado o no la información.

2.5 SERVICIOS DE PAGO

Puesto que se trata de un servicio en línea, estamos hablando de una plataforma con unos gastos derivados de mantener las máquinas corriendo, pagar la línea de conexión y, sobre todo, el personal especializado encargado de todo ello. Como casi todos los servicios en línea, WordPress.com tiene dos vías de financiación. Primero, la que generan todos los usuarios del servicio vía visitas, que luego se traducen en ingresos gracias a la publicidad. Segundo, gracias a servicios de pago como éstos que os vamos a presentar a continuación.

¿Realmente es rentable pagar por ellos? Cada uno sabrá de sus necesidades y sus gustos. Mi experiencia personal me ha llevado siempre a montar blogs en alojamientos propios, por lo que no le pagaba nada a Automattic pero sí que tenía que pagar a mi compañía de alojamiento web. Si prefieres dejarte de configuraciones, no tener las limitaciones que impone la cuenta gratuita, y te gusta lo suficiente el *blogging* como para invertir unos euros al mes en él, este tipo de servicios son tu solución.

Puesto que los precios, y los servicios en sí, cambian con el tiempo, no os aseguro que cuando leáis este libro sean los mismos, pero aquí haré un breve análisis de todos ellos en el momento en que se escribe este texto.

2.5.1 Dominio

Si quieres tener un dominio propio en lugar del típico “.wordpress.com”, tendrás que pagar. Si el dominio es tuyo, el precio asciende a 9,97 \$ (los precios son siempre en dólares estadounidenses). Si el dominio no es tuyo, tendrás que pagar por el llamado “mapeo”, la redirección del mismo hacia tu blog, que son los 9,97 \$ que hemos comentado y aparte 5,00 \$ más del registro en sí. Estos precios son por año.

2.5.2 VideoPress

VideoPress es el servicio de vídeo en línea de Automattic. Poder utilizar el servicio en tu blog (subir tus vídeos y poder reproducirlos en línea) cuesta 59,97 \$ al año. Este servicio lo veo verdaderamente caro, porque si nos quedamos sin espacio por los vídeos, deberemos pagar por otro lado el espacio necesario; además, existen alternativas gratuitas para este mismo servicio de mucha calidad: el famoso YouTube o Vimeo.

2.5.3 CSS personalizado

Por 14,97 \$ al mes, Automattic comercializa 80 temas extra a los que ya ofrece desde el comienzo y la posibilidad de tocar más opciones y personalizar el aspecto gráfico del blog.

2.5.4 Espacio adicional

Todas las fotos, vídeos o sonidos que subamos a nuestro blog ocuparán un espacio que se almacenará en los servidores de Automattic. Con los 3 GB que tenemos de forma gratuita probablemente no tengamos problemas de este tipo, pero si explotamos mucho el blog necesitaremos aumentar la cuota. Los precios anuales son:

- 5 GB - 19,97 \$
- 15 GB - 49,97 \$
- 25 GB - 89,97 \$
- 50 GB - 159,97 \$
- 100 GB - 289,97 \$

2.5.5 Sin anuncios

Aunque en un comienzo WordPress.com se ofrecía sin publicidad para la cuenta gratuita, con el tiempo Automattic vio aquí un filón que está explotando en la actualidad. Sin embargo, es algo que no le gusta a prácticamente nadie y conozco a más de un usuario que ha abandonado la plataforma para migrar a Blogger (que en el próximo punto os presentaremos con algo más de detalle). Si queremos un blog sin publicidad deberemos pagar 29,97 \$ al año.

2.5.6 Usuarios privados ilimitados

Con la cuenta gratuita, el número de usuarios registrados en nuestro blog está limitado a 35. Si queremos poder mantener una comunidad más numerosa de usuarios asociados, deberemos pagar una cuota de 29,97 \$ al año.

Aprovechando que terminamos este punto y a modo de resumen, por un precio menor podemos tener WordPress alojado en un servidor propio sin limitación alguna. Este tipo de servicios están orientados a las personas que les gusta lo suficiente este mundo, pero que no quieren tener que estar pendientes de las configuraciones, actualizaciones y atenciones necesarias que necesita una instalación en un equipo propio.

2.6 OTRO SERVICIO DE BLOGS EN LÍNEA: BLOGGER

En la actualidad sólo existe un rival en el mercado en que se mueve WordPress.com: Blogger, la plataforma de Google de la que hemos hablado anteriormente.

También existen otros servicios que por posibilidades y tecnología quedan bastante atrás de los dos líderes. En este apartado podríamos nombrar a los españoles Bloguay.com, Obolog.com o Blogspot.es. Algunos de éstos, de hecho, son servicios que se basan en WordPress.org para dar un servicio al público al igual que lo hace Automattic con WordPress.com. Recordad que la licencia GPL permite este tipo de servicios siempre que se cumplan algunas condiciones como también liberar las novedades que se añadan a la plataforma.

2.6.1 Blogger

Blogger, como ya comentamos antes, es una plataforma creada por Pyra Labs que posteriormente fue adquirida por el gigante Google. De su lanzamiento en 1999 hace ya más de 11 años, por lo que es una plataforma experimentada,

pionera y muy rejuvenecida gracias al nuevo *look* que le aplico Google en 2010. La compra de la plataforma por parte del gigante en 2003 supuso un revulsivo para la misma y, gracias a ella, se mantuvo como líder indiscutible de los servicios de blogs en línea hasta la aparición de WordPress. En la actualidad, hay fuentes que dicen que WordPress.com tiene más usuarios y otras que dicen que Blogger es la vencedora en este aspecto. Lo que sí demuestran estos datos es que existe una bicefalia clara que beneficia al usuario al crear un espacio de competencia en un servicio gratuito.

Nada mejor que probéis vosotros mismos el por qué de esta igualdad y os creéis un blog en Blogger. Para hacerlo deberéis acceder al portal en cuestión: <https://www.blogger.com/> y con la cuenta de Google os bastará para autenticaros.

Vamos a ver los principales factores a favor y en contra de Blogger respecto de WordPress.com. He de deciros que esta comparativa sería muy distinta si la hubiera escrito hace un par de años. Las mejoras introducidas en Blogger en los últimos tiempos han limado aspectos muy importantes que antes fallaban.

2.6.2 A favor

2.6.2.1 INTEGRACIÓN CON GOOGLE

Posiblemente, uno de los aspectos más destacables de Blogger es su integración con los demás servicios de Google. El gigante de la red ofrece una variedad de “servicios en la nube” (a través de Internet) inigualable por ningún competidor. La posibilidad de tener integrado Blogger con Google Sites, por ejemplo, le añade posibilidades a esta plataforma, mientras que el hecho de poder iniciar sesión con la cuenta de Gmail le añade comodidad. Sólo tendremos que aceptar las condiciones de uso de Blogger y con la misma cuenta nos autenticaremos.

2.6.2.2 AUSENCIA DE PUBLICIDAD

A diferencia de WordPress.com, Blogger aún no añade publicidad al blog de forma gratuita para el anunciante, más bien al contrario, como explico luego. Digo aún porque, hasta hace bien poco, WordPress.com tampoco lo hacía y en su momento nadie pensaba que ese momento podía llegar. La maniobra es beneficiosa económicamente para la plataforma, si tienes publicidad ganan de ella y si quieres quitarla tienes que pagar.

También hay que pensar que Google es el propietario de la mayor plataforma de anuncios en línea en el mundo, Google AdSense y compañía, y por

lo tanto si pones publicidad en tu blog, también ganan ellos, dándote a ti parte del beneficio.

2.6.2.3 POSIBILIDAD DE INTEGRAR PUBLICIDAD

La posibilidad de monetizar el blog es algo siempre interesante para todos. Aunque sea testimonial la suma económica que nos puedan reportar los clics sobre los anuncios de nuestro blog, a algunos le vendrá bien para comprarse un capricho y para otros será un objetivo al que batir para seguir escribiendo entradas de interés.

Como ya he dicho, el hecho de que Google AdSense sea propiedad de Google, allana mucho el camino en este tipo de integraciones. La publicidad generada por los anunciantes va a parar en buena parte a Google. Aunque el propietario del blog se quede con una parte, Google sale ganando de la suya y, por lo tanto, genera una rentabilidad en un servicio que además es gratuito.

2.6.3 En contra

2.6.3.1 INTERFAZ GRÁFICA MENOS ATRACTIVA E INTUITIVA

WordPress es todo un referente en cuanto a las interfaces gráficas de administración vía web. La interfaz de Blogger tiene más años y ha sufrido menos actualizaciones. Además, no sólo ofrece menos opciones, sino que la forma de llegar a ellas es bastante más lenta.

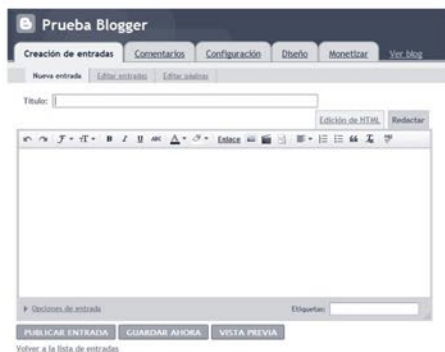


Figura 2.8. Interfaz de administración de Blogger

2.6.3.2 MENOS OPCIONES

Como he dicho en el punto anterior, ofrece menos opciones. Con sólo echar un vistazo a la administración de Blogger en la ilustración que se muestra, comprenderéis de qué os hablo. Las opciones son las básicas para un blog, pero parecen pocas con la cantidad de opciones que ofrece su competidor.

2.6.3.3 NO EXISTE UN GESTOR DE CONTENIDOS ASOCIADO

La imposibilidad de migrar a un gestor de contenidos asociados en un futuro para proseguir nuestro blog de forma libre y sin limitaciones es un gran lastre para Blogger. Aunque existen formas de importar el contenido desde Blogger a, por ejemplo, una instalación manual de WordPress, nunca será tan inmediato ni tan similar como en el caso de la transición de WordPress.com a WordPress. Una transición que han realizado miles de *blogueros* en el mundo.

2.6.3.4 SIN SERVICIOS DE PAGO

Seguro que a más de uno le ha extrañado ver este punto en inconvenientes, en lugar de en ventajas. El hecho de que se ofrezcan servicios de pago no es perjudicial, más bien todo lo contrario, el problema vendría si se hacen de pago servicios que hasta el momento se prestan de forma gratuita. Blogger no ofrece servicios de pago, por lo que el auténtico fan de este tipo de servicios verá imposibilitado ir más allá de lo que le ofrece la plataforma en un comienzo. Los servicios gratuitos de Blogger nada tienen que envidiar a los gratuitos de WordPress.com.

2.6.3.5 DIFICULTAD PARA ENCONTRAR UN NOMBRE

Muchos son los años que lleva Blogger en uso y, por desgracia, muchos usuarios del mismo tras abandonar al completo su blog no lo dan de baja, lo que provoca que cada día sea más complicado encontrar un nombre adecuado para nuestro blog. En WordPress.com también existe ese problema, pero dado que el número de años que lleva en línea es menor, está menos acentuado y aún pueden encontrarse nombres cortos y recordables que estén disponibles.

2.6.3.6 NO ES SOFTWARE LIBRE

Este punto está muy relacionado con el hecho de que no exista un gestor de contenidos asociado. Existe una retroalimentación constante entre WordPress como proyecto de software y WordPress.com como servicio de blogs en línea. La comunidad al completo se beneficia de las mejoras y el soporte que Automattic brinda al proyecto; a la vez que se aprovecha de las mejoras que se implementan en el mismo para ofrecer un servicio de más calidad.

DESCARGA, INSTALACIÓN Y CONFIGURACIÓN

Desde un comienzo hemos apuntado las limitaciones que tiene la plataforma en línea que ofrece Automattic. Aunque parte de éstas pueden verse resueltas de forma directa pagando, lo mejor, desde mi punto de vista, es saltar al alojamiento propio. Tendremos las dificultades típicas de cualquier persona que mantiene su propio servicio: instalarlo, configurarlo, actualizarlo y resolver los problemas que vayan surgiendo. Sin embargo, a la larga compensa y mucho tener nuestro blog en un alojamiento propio.

A lo largo de este capítulo comentaremos cómo instalar WordPress en un equipo propio y, posteriormente, cómo hacerlo en un servidor externo. Lo ideal, desde mi punto de vista, es instalarlo primero en nuestro equipo, aprender los cuatro detalles que debemos conocer de la instalación, para posteriormente dar el salto a un servidor externo. De esta forma, nos formamos en nuestro propio equipo sin hacer gasto alguno y de forma privada, para luego dar el salto a un servicio de alojamiento (gratuito o de pago) en el que hacer público el blog y desentendernos de la parte de mantenimiento del servidor propiamente dicha.

3.1 INSTALACIÓN EN NUESTRO EQUIPO

En este punto detallaremos cómo llevar a cabo la instalación de WordPress en nuestro propio equipo, para que nos sirva de plataforma de pruebas con la que aprender el manejo del proyecto sin limitaciones.

Puesto que para instalar WordPress necesitamos un servidor web, con capacidad de procesamiento de PHP y un servidor de base de datos MySQL, explicaremos también cómo instalar todo este software. Gracias a la popularización de estas tecnologías ya comprobaremos que es prácticamente inmediato hacerlo. Llevaremos a cabo dos explicaciones, una para el sistema operativo Microsoft Windows y la segunda para la distribución más popular de GNU/Linux, Ubuntu.

A partir de este punto, las explicaciones serán para el primero, dado que es más popular que el segundo, pero en la mayor parte de los casos valdrán para ambos.

3.1.1 En Microsoft Windows

Hasta hace no mucho, este tipo de procesos de instalación de software para servidor web con PHP y servidor MySQL eran por lo menos tediosos, en sistemas de Microsoft. Sin embargo, la popularización de estas tecnologías, nos ha proporcionado una serie de herramientas que automatizan el trabajo.

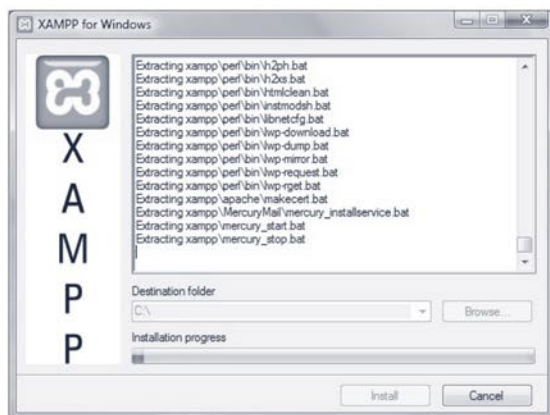


Figura 3.1. Proceso de instalación de XAMPP

3.1.1.1 DESCARGA

De las opciones disponibles, que cada vez son más, he elegido el proyecto XAMPP (<http://www.apachefriends.org/es/xampp.html>). Tras acceder a la URL que acabo de indicar, buscamos “XAMPP para Windows” y accedemos a un sitio donde se describe qué software incluye este preparado. Además de los que necesitamos: Apache, PHP, MySQL y phpMyAdmin; vienen otros programas que pueden resultar de interés pero que en este caso no usaremos: Mercury, servidor FTP FileZilla y el intérprete de Perl.

Bajamos a lo largo de la página y llegamos a la sección de descargas. Se nos informa de que podemos descargar dos versiones: una versión .EXE y otra comprimida ZIP, elegimos la primera. Hay más opciones de descarga: el paquete de actualización de versiones antiguas a las más nuevas o una versión reducida de XAMPP, pero las ignoramos. Al elegir la versión .EXE se nos redirige a la forja más importante del software libre Sourceforge (<http://sourceforge.net/>). La descarga comenzará automáticamente.

3.1.1.2 INSTALACIÓN

Nada más ejecutar nos aparecerá el cuadro de instalación, hacemos clic sobre “Install” y el proceso comenzará. Tras unos minutos en los que se descomprimirá todo el contenido, aparecerá una interfaz en modo texto en la que se nos preguntarán una serie de opciones de la configuración, responderemos sí o no con las iniciales *y* o *n*, de sus traducciones en inglés respectivamente:

- *Should I add shortcuts to the startmenu/desktop?* Nos pregunta si queremos añadir accesos directos al menú inicio y al escritorio. Dejo la respuesta a vuestra elección.
- *Should I locate the XAMPP path correctly?* Respondemos que sí queremos que establezca la ruta de XAMPP de forma correcta.
- *Should I make a portable XAMPP without drive letters?* Es mejor decir que no, preferimos que la instalación no sea portátil, no es nuestro objetivo y sí que podamos establecer los servidores como servicios.

En este momento el instalador nos comunicará que el proceso de instalación ha terminado. Ahora, se nos preguntará por la zona horaria para terminar la configuración:

- *Timezone:* Zona horaria, por defecto será la del sistema, que será correcta. Presionamos “ENTER” para continuar.

En este momento nos aparece un menú, en el que elegir qué hacer tras la instalación. Pulsamos “X” y salimos. Procederemos a abrir el panel de administración sobre el menú inicio.

3.1.1.3 EJECUCIÓN

Tenemos instalada y configurada la herramienta, estamos ya listos para probar su funcionamiento. Nos dirigimos al menú inicio y en la carpeta “XAMPP for Windows” ejecutamos “XAMPP Control Panel”. A través de este programa podremos controlar el estado de ejecución de los distintos servicios que hemos instalado. Podéis ver el aspecto en la figura siguiente.

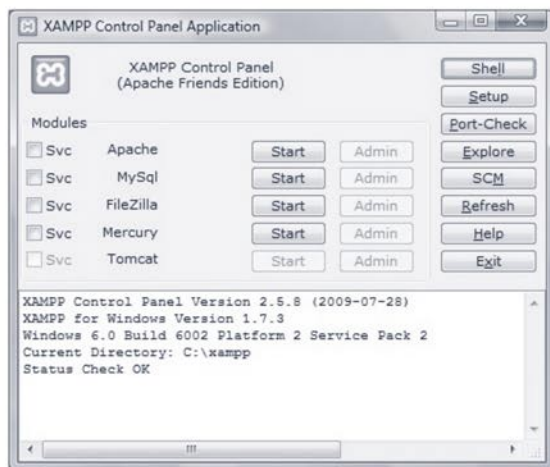


Figura 3.2. Panel de control de XAMPP

Deberemos iniciar sólo los servicios de Apache y de MySQL. Para ello, pulsamos el botón “Start”, situado a la derecha de los mismos. En el cuadro de texto inferior aparecerá que los servicios han iniciado y, entre el botón y la etiqueta, aparecerá el texto “Running” sobre fondo verde.

¿Cómo probar que todo está en marcha? Accedemos con nuestro navegador web favorito a la dirección de nuestro servidor local, es decir, a <http://localhost>. Deberá aparecer una página con un logo de XAMPP y la

posibilidad de elegir idioma. Elegimos español y llegamos a una página resumen del estado del servidor. Si estamos viendo esto, todo ha ido sobre la marcha.

¿Cómo posibilitar que todo el mundo vea este servidor? Deshabilitando el cortafuegos de Windows para el puerto 80 (el del tráfico web), podremos hacer que toda nuestra red local pueda ver nuestro contenido a través de nuestra IP. Configurando la parte NAT (*Network Address Translation* “traducción de direcciones de red”) de nuestro enrutador, podríamos posibilitar que todo el mundo en la red llegara a nuestro servidor. Sin embargo, como he comentado antes, no usaremos el servidor que acabamos de instalar para poner nuestro blog en producción, simplemente será nuestra estación de pruebas.

Si diéramos el paso de poner en producción el servidor, deberíamos pararnos a explicar toda la problemática básica que se requiere para mantener mínimamente un servidor. Probablemente, la primera elección sería pasarnos directamente a Linux, porque, como digo, esta plataforma es ideal para hacer pruebas y, cuando tengamos todo listo, pasar a un servidor en la red. En el último punto de este capítulo comentaremos cómo llevar a cabo la migración y qué posibilidades tenemos en el mercado en la actualidad.

Según la instalación que hemos seguido, tanto Apache como MySQL no se levantarán automáticamente cada vez que arranquemos Windows. Por lo tanto, deberéis acceder al panel de control y lanzarlos manualmente cada vez que lo necesitemos.

3.1.2 En Ubuntu GNU/Linux

Desde sus comienzos, GNU/Linux fue un sistema operativo orientado a los sectores más profesionales y expertos de la comunidad informática. Ubuntu es la distribución más popular hoy en día, y está dentro del grupo de las más amigables con el usuario. Sin embargo, no pierde esa cercanía con sus orígenes y, por eso, es realmente simple llevar a cabo la instalación de los servicios necesarios en este sistema operativo.

En el menú “Sistema”, entramos en “Administración” y ejecutamos el “Gestor de paquetes Synaptic”. Si preferís usar la consola deberéis introducir *gksudo synaptic*. Tras introducir nuestra contraseña, para tener permisos de superusuario, se abrirá el gestor. Nos dirigimos al menú “Editar”, “Marcar paquetes por tarea...” y en la lista que nos aparece seleccionamos “LAMP Server”. Las iniciales LAMP significan: Linux, Apache, MySQL y PHP; justo lo que necesitamos. Marcamos, aceptamos y aplicamos los cambios. Los paquetes necesarios se descargarán y se instalarán en nuestro equipo. La única interrupción

se producirá a la hora introducir la contraseña de administrador del servidor MySQL.

Finalmente, para comprobar que todo ha ido bien, al igual que antes, nos dirigimos a cualquier navegador web y abrimos la dirección: *http://localhost*. Esta vez debería aparecer una pantalla, que ponga “It Works!”, en la que Apache nos confirma que la instalación ha sido correcta.

Ubuntu sí lanza estos servicios de forma automática, según el método de instalación que hemos seguido.

3.1.3 Pasos comunes

3.1.3.1 DESCARGA

Ya tenemos listas todas las herramientas, en ambos sistemas, para poner en marcha el sistema gestor de contenidos. Lo primero ahora es descargar el código de WordPress. Accedemos a la web del proyecto libre: *http://www.wordpress.org* (no confundir con el servicio que hemos descrito antes: *wordpress.com*). Elegimos el idioma español, haciendo clic sobre un mensaje que aparece en mitad de la página: “WordPress is also available in Español”.

Cuando estemos dentro de la página “WordPress español”, en la parte derecha veremos un cuadro que indica: “Descargar WordPress” y el número de versión. Por razones de compatibilidad elegimos la versión ZIP en lugar de la TAR.GZ. Una vez descargado, extraemos el contenido del fichero.

Si estamos en Windows deberemos copiar y pegar el contenido a la dirección raíz del servidor Apache, con XAMPP será *c:\xampp\htdocs*. En el caso de Ubuntu, el directorio será */var/www*, y para copiar y pegar deberemos ser superusuarios, la instrucción sería algo así: *\$ sudo cp /carpeta/wordpress /var/www/*.

3.1.3.2 INSTALACIÓN

3.1.3.2.1 Creación de usuario y base de datos MySQL

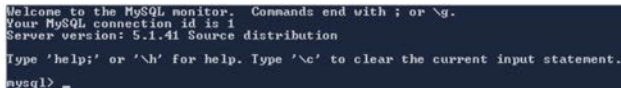
WordPress necesita una base de datos en la que almacenar toda la información que maneja: entradas, usuarios, categorías, etiquetas, comentarios... Esta base de datos estará alojada en MySQL y debemos crearla previamente para que WordPress pueda comenzar a manejarla. Asimismo, crearemos un usuario y le asignaremos permisos totales para que maneje esa base de datos.

Aunque no estemos en un sistema en producción, siempre es conveniente crear usuarios diferentes al administrador de la base de datos para poder interactuar con aplicaciones particulares, como es el caso. En el caso de que la seguridad se vea comprometida, de cara a la base de datos sabemos que no afectará al resto de datos almacenados en ésta.

Podríamos hacer estos pasos con una interfaz gráfica como phpMyAdmin, que instala XAMPP por defecto, pero lo haremos mejor desde la terminal de MySQL para ver uno a uno los pasos que hay que dar.

¿Cómo acceder a la terminal de MySQL? En Windows deberemos dirigirnos, desde la consola, al directorio donde se encuentra. Para ejecutar la consola, en el diálogo ejecutar de nuestro sistema operativo (ya sea XP, Vista, 7 o un Server) escribimos *cmd.exe*. En la consola que se nos abre nos dirigimos al directorio siguiente con la orden *cd* (cambiar directorio en inglés): *c:\xampp\mysql\bin*, es decir: *cd c:\xampp\mysql\bin*. En Linux no es necesario este paso porque el cliente de MySQL está en una ruta del llamado “PATH del sistema”. Abrimos la terminal presionando de forma simultánea “Alt” y “F2”, en el cuadro de ejecución introducimos “gnome-terminal” y ya tendremos la consola de Linux en pantalla.

La orden que hay que ejecutar ahora es la siguiente: *mysql -u root -p*, a continuación nos pedirá la contraseña del administrador del sistema. En el caso de Windows será el vacío (es la que trae por defecto XAMPP y sólo tendremos que pulsar “ENTER” para continuar), en el caso de Ubuntu, será la que hemos elegido en la instalación. Nos aparecerá una pantalla como la que se muestra en la ilustración siguiente.



```
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.1.41 Source distribution

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> _
```

Figura 3.3. Consola de MySQL

La consola de MySQL es un intérprete de comandos propios de este sistema gestor de bases de datos. Podremos hacer operaciones básicamente de dos tipos: de administración del servidor y de manejo de los datos mediante sentencias SQL. Las sentencias que hay que escribir son las siguientes:

1. Creamos la base de datos: *CREATE DATABASE wordpress;*
2. Creamos el usuario “usuario_wp”, que será quien interactúe con la base de datos. Le asignamos la contraseña “password_mysql” y le damos todos los permisos sobre la base de datos que acabamos de crear, limitando su acceso desde el servidor local. Todo esto se puede hacer de una sola vez: *GRANT ALL PRIVILEGES ON wordpress.* TO 'usuario_wp'@'localhost' IDENTIFIED BY 'password_mysql' WITH GRANT OPTION;*
3. Refrescamos usuarios y privilegios en la base de datos: *FLUSH PRIVILEGES;*
4. Salimos de MySQL: *QUIT;*

3.1.3.2.2 Instalación del CMS

El instalador de WordPress prepara el fichero de configuración del sistema de forma automática. Como en ocasiones pueden surgir problemas, al final de este punto, también comentaremos la creación del fichero de forma manual. Como ya tenemos el contenido descomprimido de WordPress en el directorio pertinente del servidor web, accedemos al navegador y escribimos <http://localhost/wordpress/>.

Nos aparecerá una pantalla de WordPress indicándonos que no tenemos archivo de configuración. Pasemos a crearlo mediante la interfaz que nos proporciona el instalador. Podéis ver el aspecto de la misma en la siguiente ilustración.

Como podéis ver en la figura, nos pide los siguientes datos:

- Nombre de la base de datos: Para nuestro caso “wordpress”.
- Nombre de usuario: “usuario_wp”.
- Contraseña: “password_mysql”.
- Host de la base de datos: Dejamos el servidor local, “localhost”.
- Prefijo de tabla: Dejamos el que viene por defecto “wp_”.

Hacemos clic sobre “Continuar” y nos aparecerá una pantalla en la que se nos comunica que los datos son correctos.



The screenshot shows the WordPress installation screen. At the top is the WordPress logo. Below it is a paragraph: "A continuación deberás introducir los detalles de conexión con tu base de datos. Si no estás seguro de cuáles son contacta con tu proveedor de alojamiento." Below this is a form with five rows, each with a label, a text input field, and a description:

Nombre de la base de datos	wordpress	El nombre de la base de datos en la que quieres que se ejecute WP.
Nombre de usuario	usuario_wp	Tu nombre de usuario de MySQL.
Contraseña	password_mysql	...y la contraseña de MySQL.
Host de la base de datos	localhost	Si no funciona localhost tendrás que contactar con tu proveedor de alojamiento para que te diga cuál es.
Prefijo de tabla	wp_	Si quieres ejecutar varias instalaciones de WordPress en una sola base de datos cambia esto.

At the bottom left of the form is a button labeled "Enviar".

Figura 3.4. Instalación de WordPress. Datos de conexión con MySQL

Si al continuar nos aparece “Error de conexión con la base de datos”, el proceso ha fallado. Podríamos averiguar las causas y volver a repetirlo, pero es más práctico modificar manualmente el código del fichero de configuración.

Para ello nos dirigimos al directorio donde está WordPress y abrimos el fichero “wp-config.php” con cualquier editor de texto. En Windows nos bastará el que trae el sistema por defecto, “notepad.exe”. En Ubuntu, deberemos abrir Gedit con el rol de superusuario: `gksudo gedit /var/www/wordpress/wp-config.php`.

Si el editor de notas de Windows te parece demasiado simple y con pocas posibilidades, te recomiendo que te descargues e instales Notepad++, un editor de texto y código fuente libre, que aún lo mejor de los editores de texto: resaltado de código, posibilidad de usar varios documentos simultáneamente (en varias pestañas), comparación, diferenciación, búsqueda y reemplazo (incluida la opción de usar patrones y reglas), etc. Podéis encontrarlo en la web principal del proyecto <http://notepad-plus-plus.org/> o buscándolo en Google. Para los que usáis Ubuntu, aunque Gedit no es un mal editor en absoluto, recomiendo que busquéis Geany, en Synaptic o el Centro de Software.

Tenemos ante nosotros el fichero en el que WordPress define las variables necesarias para acceder a la base de datos. A nosotros nos interesan las primeras líneas que aparecen. Los datos los rellenaré con los valores que hemos introducido anteriormente.

Una vez modificados los datos, sólo tendréis que guardar el fichero, volver a la interfaz de instalación y refrescar con vuestro navegador.

```
/** El nombre de tu base de datos de WordPress */  
  
define('DB_NAME', 'wordpress');  
  
/** Tu nombre de usuario de MySQL */  
  
define('DB_USER', 'usuario_wp');  
  
/** Tu contraseña de MySQL */  
  
define('DB_PASSWORD', 'password_mysql');  
  
/** Host de MySQL (es muy probable que no necesites cambiarlo) */  
  
define('DB_HOST', 'localhost');
```

Al volver al navegador, como ya he comentado, refrescamos y nos aparecerá ya el diálogo de las opciones específicas de WordPress. Estamos llegando al final de la instalación. Los datos que nos piden son:

- Título del sitio: He elegido “Blog de pruebas para Ra-Ma”.
- Nombre de usuario: Por defecto “admin”, podéis cambiárselo si os parece mejor uno propio, por ejemplo, “jcarazo”.
- Password: Será la contraseña del usuario administrador. Lo introducimos dos veces para evitar equivocaciones.
- Correo electrónico: El tuyo o uno de los que tengas datos de acceso.
- Permitir que mi sitio aparezca en motores de búsqueda: Como es un blog de pruebas, lo desmarcamos. Cuando vayamos a ponerlo en producción, ya activaremos esa opción.

Hacemos clic sobre “Instalar WordPress” y tras unos instantes, en los que el CMS crea y rellena la estructura en la base de datos, ya podremos acceder a

nuestro blog. Hacemos clic en “Acceder” y ya tendremos delante de nosotros la pantalla de petición de credenciales de WordPress. Sólo queda identificarnos con nuestro usuario y contraseña y empezar a trabajar.

3.2 INSTALACIÓN EN UN SERVIDOR EXTERNO

Ya tenemos la plataforma totalmente operativa en nuestro equipo. Podemos hacer todas las pruebas que queramos y podremos aprender a desenvolvernó a la perfección en el entorno. Sin embargo, tendremos una limitación, a no ser que tengamos nociones de mantenimiento de servidores y dispongamos de la infraestructura necesaria: un equipo conectado a la red y sin apagarse las veinticuatro horas del día, no podremos tener un blog en línea. Al fin y al cabo, lo que desea todo el que monta un blog es poder tener el blog en línea y que los visitantes lean las entradas e interactúen con éste.

¿Cómo conseguirlo? Pues necesitaremos tener un servicio de alojamiento en la web que permita PHP y nos proporcione acceso a una base de datos MySQL. En la actualidad, incluso los más básicos, satisfacen estos dos requisitos.

Puesto que el mercado del alojamiento web está realmente vivo y podríamos decir que cambia incluso a diario, el objetivo de este punto no es recomendaros ningún proveedor de *hosting*, así es como se le llama en inglés al alojamiento web, sino daros las nociones básicas para que sepáis qué debéis buscar cuando estéis estudiando a qué proveedor migrar vuestro WordPress. También explicaremos cómo llevar a cabo este paso, la migración del servidor local al remoto, para que cuando deis el paso no tengáis problemas.

3.2.1 Elección de un servicio de alojamiento

A continuación, describimos una serie de características que hay que valorar a la hora de elegir un servicio de *hosting*. La característica más limitante siempre será nuestro dinero. La intención es encontrar un servicio que nos ofrezca lo máximo a un precio contenido, e incluso gratuito.

3.2.1.1 CARACTERÍSTICAS

3.2.1.1.1 Espacio de almacenamiento

El primer factor que salta a la vista cuando buscamos alojamiento web es el espacio en disco que nos ofrecen. Éste es uno de los problemas que nos quitamos de encima cuando tenemos un servidor casero, pero que puede ser un factor limitante en un servidor externo. La evolución ha sido muy rápida en este aspecto.

Hace años ofrecían 50 MB por un precio considerable y hoy en día incluso llegan a ofrecer espacio ilimitado en paquetes básicos. Con el tiempo esta limitación desaparecerá pero mientras tanto es una cifra a tener en cuenta.

Nos limitará sobre todo a la hora de subir contenido multimedia o ficheros de datos. Para nuestro propósito deberemos buscar un servicio que ofrezca al menos 100 MB de espacio, tanto para el CMS en sí como para los datos de la base de datos y el contenido multimedia (imágenes principalmente) que añadamos en las entradas.



Figura 3.5. Granja de servidores

3.2.1.1.2 Volumen de transferencia

Es la cantidad de datos que nos permiten transmitir cada cierto tiempo, suele ser mensualmente. Por ejemplo, si tenemos 10 GB/mes de datos, tendremos la posibilidad de que los clientes (los visitantes) se descarguen a lo largo del mes 10 GB de datos de nuestro blog.

Por lo general, para un blog la cantidad necesitada no es demasiado alta. Los sistemas como WordPress, en realidad, lo que transmiten es texto plano, en HTML; con los correspondientes ficheros binarios (imágenes, vídeos, sonidos) que lo complementan.

Será proporcional al número de visitas que tengamos, por lo que de primeras no tendremos problemas si tenemos un servicio que ofrece poco volumen de transferencia. Con el tiempo podremos ampliarlo para adecuarlo a nuestras necesidades.

3.2.1.1.3 Sistema operativo

Los proveedores de servicio web pueden funcionar bajo distintos sistemas operativos. La opción mayoritaria es GNU/Linux, la segunda más popular Microsoft Windows Server y en raras ocasiones otros (como puedan ser cualquier rama de Unix como Solaris; Mac OS es verdaderamente raro encontrarlo). A priori, no hay duda, es mucho mejor un servidor Linux que uno Windows.

La primera razón es por todas las ventajas que tiene Linux en el mundo del servidor profesional en comparación con Windows, y que son el resultado de liderar esta parte del mercado desde sus inicios. La segunda es el precio. Un servidor Windows será casi siempre más caro que uno con Linux, la razón de este hecho tenéis que buscarla en el coste de las licencias del primero.

¿Por qué nos puede interesar un servicio con Windows? Si utilizamos tecnología .NET para el desarrollo de nuestras aplicaciones web o tenemos necesidad de albergar nuestros datos sobre una plataforma SQL Server, esta opción será casi limitante. Aunque hay alternativas, como el Proyecto Mono, cuando hablamos de .NET en un entorno de este tipo, necesitamos Windows Server.

3.2.1.1.4 Disponibilidad de tecnologías

Muy relacionado con el punto anterior. Existen tecnologías que no están disponibles en todos los servicios de alojamiento. PHP, hoy en día, es casi un estándar, pero otros casos como Ruby (para Rails) o Python (para Django, Turbo Gears...) son menos frecuentes.

Respecto a la base de datos, deberemos buscar también una base de datos MySQL. Al igual que PHP, al ser la más popular en el mundo de los desarrollos web, no deberemos tener problemas.

3.2.1.1.5 Capacidad de cálculo

Si tenemos una cuenta en un *host* compartido con otros usuarios (ahora se explican los diferentes tipos de cuenta que existen), para evitar que el rendimiento decaiga, las empresas vendedoras de *hosting*, en ocasiones, limitan el número de procesos que podemos ejecutar simultáneamente. El problema puede venir con ciertas tecnologías que crean procesos nuevos con nuestro usuario y pueden llegar al tope permitido. En el caso de WordPress no tendremos problemas en ese sentido.

Sin embargo, uno de los grandes defectos de WordPress es la carga de CPU que conlleva. Este problema aparece cuando el blog tiene gran cantidad de visitas el mismo día. Las continuas peticiones pueden hacer que nuestro servicio saturé al proveedor y tengamos problemas en este sentido.

Posteriormente, explicaré cómo paliar este problema para evitar sobrecargas y tener mecanismos que disminuyan y rentabilicen el uso de procesador.

3.2.1.1.6 Situación y lenguaje del soporte

La situación del servidor también es importante. A algunos les puede interesar que el servicio esté en España o fuera de España por temas fiscales. Otro aspecto muy importante es el idioma del soporte. Si preferís comunicarnos en español buscad una empresa hispanohablante, no tienen por qué ser españolas, hay muchas empresas sudamericanas que dan un servicio excelente. En caso de problemas, siempre querremos una comunicación directa con nuestro proveedor.

3.2.1.1.7 Otras

Para terminar, apuntamos otras características que pueden ser de interés. Desde tener acceso por SSH, a poder usar FTP sobre canal seguro, pasando por tener buena valoración en la red, toda característica positiva será siempre bien recibida.

3.2.1.2 TIPOS

Tras comentar las características más importantes que hay que estudiar en un servicio de alojamiento web, ahora pasamos a ver qué tipo de paquetes podemos contratar. Cualquiera de los siguientes será válido para nuestro propósito, pero cada uno tiene sus particularidades.

3.2.1.2.1 Cuenta en servidor compartido

Sin lugar a dudas la más popular. Ofrece un usuario con el que conectarnos a una máquina en la que hay más usuarios con sus datos y aplicaciones web. Tiene la ventaja de ser la más barata, pero es la más limitada de todas. Para nuestro caso es suficiente y la mayoría de blogs es la que usan. Si tenemos un gran tamaño deberemos migrar a un servicio superior.

3.2.1.2.2 Cuenta de revendedor

Similar a la anterior, pero con una particularidad importante: permite crear nuevos usuarios a los que asignarles cuotas de espacio y volumen de transferencia. Como su nombre indica su uso principal es ser revendedores de *hosting*.

3.2.1.2.3 Servidores privados virtuales

Servidor virtual que funciona sobre una máquina física en la que hay más servidores virtuales. Se nos asignan unos recursos de cálculo, memoria y espacio en disco y actuamos remotamente como si tratáramos con un servidor propio. Popularmente llamados VPS (*Virtual Private Server*).

3.2.1.2.4 Servidor dedicado

Servidor físico que tenemos alquilado en la empresa de alojamiento. Sus potencialidades son muy similares al anterior, pero a diferencia de éste, tenemos alquilado físicamente el equipo entero. Es la solución más cara.

3.2.2 Proceso de instalación

Ahora nos toca llevar a cabo un proceso de instalación análogo al anterior, pero en lugar de en un servidor establecido en nuestro equipo, en uno remoto que hemos contratado en la red. Puesto que hay muchas partes en común, nos centraremos en las diferencias.

Cuando contratas una cuenta en un servidor compartido en la red, el proveedor de *hosting* te ofrece una interfaz para que controles los recursos que pone a tu disposición. Puesto que para la mayoría es mucho más incómodo, incluido para el administrador del sistema, tratar con la consola del sistema vía conexión SSH, lo más frecuente es que tengamos a nuestra disposición un panel gráfico en formato web con el que interactuar.

De entre todas las soluciones posibles, la que se usa en la mayor parte de los casos es el llamado cPanel. Se trata de un proyecto de código cerrado, que ha tenido gran éxito entre los proveedores de servicios de alojamiento. Ofrece al cliente una interfaz intuitiva para administrar los recursos y servicios que contratamos. Ésta será la herramienta que usemos para hacer un ejemplo lo más parecido posible a la situación que os encontréis.

3.2.2.1 PREPARACIÓN PREVIA

Antes de seguir adelante, necesitamos disponer de una herramienta base para trabajar con un servidor remoto: un programa que nos permita subir y bajar ficheros a éste. La herramienta que usaremos es software libre y está disponible para los grandes sistemas operativos de la actualidad: Microsoft Windows, las distintas distribuciones de GNU/Linux y Mac OS X. Su nombre es FileZilla y es un cliente de FTP (*File Transfer Protocol* “protocolo de transferencia de ficheros”).

Si utilizamos Windows, para instalarlo deberemos acceder a su sitio web: <http://filezilla-project.org/>, y una vez allí, hacer clic sobre “Download FileZilla Client” (descargar cliente FileZilla). Dentro de la web que aparece, elegimos la versión “setup” (la recomendada) y comenzará la descarga. El fichero que se nos descargará es un instalador, con extensión “.exe”. Lo ejecutamos y tras unas cuantas pantallas del proceso de instalación, ya tendremos el cliente instalado.

Si usamos Ubuntu, podemos buscar e instalar el programa desde el gestor de paquetes. Abrimos el “Centro de software”, buscamos “filezilla” y sólo tendremos que pulsar sobre instalar.

Tras la instalación, debemos ejecutarlo y configurar los datos de acceso a nuestro servidor. Estos datos debe habérmolos proporcionado nuestro proveedor de *hosting* al darnos de alta en el servicio. Los datos en cuestión son:

- Servidor: Dirección del servidor FTP.
- Puerto: Si no se dice nada, es el 21.
- Tipo de servidor: Si no se dice nada, es servidor FTP, en caso de querer opciones más seguras como sFTP u otra alternativa debemos consultarlo con nuestro proveedor.
- Tipo de inicio de sesión: Elegimos normal.
- Usuario: Debe proporcionárnoslo nuestro proveedor.
- Contraseña: Ídem que el anterior.

En la parte superior izquierda de la pantalla tenemos el icono del “Gestor de sitios”. Hacemos clic sobre él y, en el formulario que aparece, indicamos “Nuevo sitio”. Le damos nombre, uno que sirva para que lo identifiquemos nosotros, y rellenamos los datos tal como se acaba de explicar.

Conectamos y ya estamos viendo en la parte derecha de la pantalla los directorios del servidor. ¿Cómo subimos y bajamos ficheros? La pantalla principal se divide en dos partes: la izquierda, en la que tenemos nuestro ordenador, y la derecha, en la que vemos los directorios del servidor.

Para subir un fichero, directorio o grupo de ficheros, deberemos elegirlos en la parte izquierda de la pantalla, hacer clic derecho sobre ellos y elegir “Subir”. Para bajar ficheros haremos lo mismo, pero esta vez en la parte derecha de la pantalla. También tenéis otros métodos, como “arrastrar y soltar”, para intercambiar la información. Como podéis comprobar el funcionamiento es muy intuitivo.

Recordad que la mayoría disponemos de una línea ADSL doméstica (*Asymmetric Digital Subscriber Line* “línea de abonado digital asimétrica”). Como habréis comprobado en más de una ocasión, la velocidad de bajada es mucho mayor que la de subida (de ahí lo de asimétrica). Esto nos afectará negativamente cuando subamos ficheros al servidor, incluso para subir unos pocos megas, tendremos que esperar algunos minutos. Os lo comento porque la mayor parte de operaciones que llevaremos a cabo serán de subida.

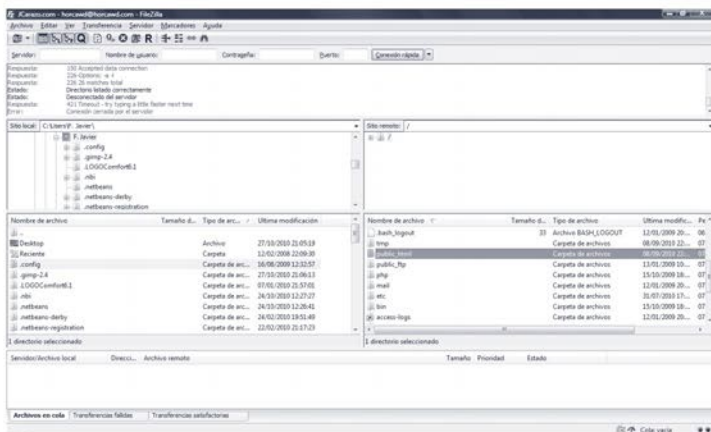


Figura 3.6. Interfaz gráfica de FileZilla

3.2.2.2 PRESENTACIÓN DEL CPANEL

El cPanel será nuestro aliado en el manejo de los servicios del servidor. Accederemos a él mediante una dirección web (que se nos proporcionará) y que será del estilo *dominio/cpanel*. Nos autenticaremos con un usuario y una contraseña y se nos presentará ante nosotros una interfaz web dividida en dos columnas:

- Columna izquierda: De menor tamaño. Incluye búsqueda, accesos directos frecuentes y datos muy importantes, las estadísticas y el estado de los servicios.
- Columna derecha: Acceso a todas las configuraciones. Se encuentran separadas por grupos.

Dependiendo de la versión, el aspecto gráfico variará levemente. Las líneas generales podéis verlas en la figura siguiente.

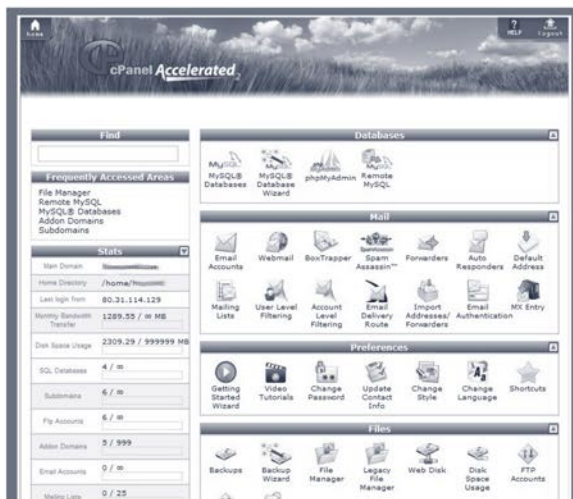


Figura 3.7. Interfaz web de un cPanel en un alojamiento web

Como podéis apreciar, las posibilidades son amplias. Para nuestros propósitos sólo tendremos que hacer uso de la gestión de bases de datos MySQL. En el punto anterior comenzamos creando la base de datos y un usuario que la maneje. Haremos lo mismo en este caso.

3.2.2.3 CREACIÓN DE LA BASE DE DATOS

Nos dirigimos a la opción “MySQL® Databases” dentro de la zona dedicada a las bases de datos en la barra derecha. Debemos hacer lo mismo que hicimos antes: crear la base de datos, crear el usuario correspondiente y otorgarle permisos, pero con la interfaz gráfica. Como veréis es más simple.

Lo primero será crear la base de datos. Al comienzo de la sección, donde indica “Create new database”, indicamos en el cuadro de texto el nombre que queremos ponerle, “wordpress”, y hacemos clic sobre “Create database”. Un mensaje nos comunicará que la base de datos se ha añadido correctamente.

Volvemos y nos dirigimos a crear nuestro usuario. Para ello, descendiendo un poco más en la página, llegamos a un título que indica: “MySQL Users”. Ahí es donde debemos indicar el nombre de usuario (o *username*) y su contraseña (*password*). Debemos repetirla dos veces y asegurarnos una fortaleza adecuada (el propio cPanel nos indicará el grado de fortaleza de la misma). Hacemos clic sobre “Create user” y se nos indicará, de forma similar a como se nos ha indicado antes, que el usuario ha sido creado.

Finalmente, tenemos que darle permisos al usuario para que pueda manejar la base de datos. En el punto anterior, hacíamos estas dos acciones de un sólo paso: crear el usuario y otorgarle permisos. Para ello nos dirigimos a la sección “Add user to database” y en los dos menús desplegables elegimos el usuario y la base de datos en cuestión. Hacemos clic sobre “Add” y en la pantalla siguiente seleccionamos todos los privilegios. Confirmamos los cambios y ya tendremos toda la infraestructura de datos preparada para el sistema.

3.2.2.4 SUBIDA DEL SISTEMA

De igual forma que antes hemos copiado el contenido del fichero de WordPress descomprimido en el directorio del servidor web (podéis repasar cómo lo hemos hecho exactamente en 3.1.3.1), ahora deberemos hacer algo análogo: subir el contenido de WordPress a la carpeta principal de nuestra cuenta en el servidor. Para ello, lo primero que hacemos es ejecutar FileZilla y conectarnos como lo hemos hecho antes.

En la parte izquierda tenemos nuestro equipo. Debemos dirigirnos al directorio donde hemos descomprimido todo el contenido de WordPress. En la parte derecha, deberemos acceder al directorio principal de nuestra cuenta en el servidor. Suele ser: “public_html” o directamente al que llegamos cuando iniciamos sesión.

Una vez tenemos estas dos localizaciones abiertas, seleccionamos todos los ficheros de la parte izquierda (los de WordPress que están en nuestro equipo), hacemos clic derecho y procedemos a subirlos. Los ficheros se pondrán en cola y subirán de forma sucesiva. Podemos ver el estado de la transferencia en la parte inferior del cliente de FTP, donde veremos tres pestañas:

- Archivos en cola: Los que quedan por subir.
- Transferencias fallidas: En caso de haber alguna deberemos hacer clic sobre ellas y pedirle que vuelvan a intentar subirse.
- Transferencias satisfactorias: Que deben ser todas al final del proceso.

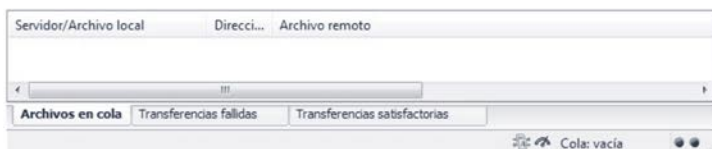


Figura 3.8. Estado de la transferencia en FileZilla

3.2.2.5 ARCHIVO DE CONFIGURACIÓN

Antes hemos probado primero con la interfaz gráfica para generar el fichero de configuración, para posteriormente explicar cómo hacerlo manualmente si tenemos problemas. Podéis volver a intentar este primer paso. Esta vez para acceder a él deberéis intentar acceder a la dirección URL del sitio web. Yo estoy preparando el sistema en <http://jcarazo.com/blog/>, es decir, en la carpeta “blog” del dominio que tengo direccionado a mi servidor, “jcarazo.com”. Os aparecerá un mensaje de “WordPress>Error” y os sugerirá crear el fichero de configuración.

El proceso será el mismo que en el punto *Instalación del CMS*. Asimismo, si falla deberemos hacer lo siguiente: modificar el fichero en nuestro equipo (con los editores que hemos comentado antes) y subirlo de nuevo al servidor con FileZilla (sólo ese fichero, el resto no hará falta). Si nos pregunta qué queremos hacer con el fichero (al existir el mismo fichero en el servidor por seguridad siempre pregunta), le decimos que queremos sobrescribirlo.

Los datos que hay que rellenar serán los siguientes:

- DB_NAME: Tal como hemos comentado a la hora de crear la base de datos en el servidor, “wordpress”.
- DB_USER: El que hayamos introducido.
- DB_PASS: Ídem.
- DB_HOST: Suele ser en la mayoría de los casos “localhost”, en caso de que no lo sea, nuestro proveedor de *hosting* nos lo habrá comunicado.

Una vez modificado el fichero de configuración y subido de nuevo al servidor, estaremos listos para acceder a la URL de nuestro blog y completar el proceso de instalación (selección del título de blog y demás), de una manera idéntica a como lo hemos hecho cuando trabajábamos sobre nuestro propio equipo.

3.3 MIGRACIÓN

Hasta ahora hemos explicado de forma independiente cómo poner en marcha WordPress en un servidor casero montado en nuestro propio equipo y cómo hacerlo en un servicio de alojamiento en la red. La idea de este libro es que hagáis exactamente eso, primero ponerlo todo a punto en un equipo sin coste para luego dar el salto a la red. Hemos definido todos los pasos que hay que dar, pero queda el que hace de puente entre ambos: migrar el sistema de un servidor a otro.

Si no tenemos nada hecho, directamente la solución será hacer uno de los dos pasos de forma independiente, pero si tenemos preparado ya algo de contenido y, sobre todo, mejoras y adaptaciones que explicaremos a partir del próximo capítulo, deberemos asegurarnos de que la migración es completa y no hay ningún problema de por medio.

La migración se separa en dos bloques bien diferenciados: migración de los ficheros y migración de los datos.

3.3.1 Migración de los ficheros

En esta primera parte, nos encargaremos de transportar, de un emplazamiento a otro, todos los ficheros que componen el sistema junto con todos los añadidos y modificaciones que hayamos hecho.

Por suerte, todo el contenido en cuestión estará presente en la carpeta de WordPress. Todo tema gráfico, complemento o modificación que hagamos se almacenará ahí. Esto facilita enormemente el proceso de migración. ¿Qué debemos hacer? Transferir todos los datos, mediante el cliente FTP, de un servidor a otro.

3.3.2 Migración de los datos

Si la primera parte era más directa, esta segunda también lo es, pero requiere de cierta intervención mayor. Para entender el porqué de esta migración en dos pasos, debemos entender también cómo funciona WordPress. Aunque ya se ha comentado en capítulos anteriores, a grandes rasgos podemos decir que unos ficheros PHP (los que hemos ya traspasado en el punto anterior) tratan con unos datos presentes en bases de datos MySQL, para mostrar un resultado al usuario.

Tenemos ya los ficheros PHP que tratan con los datos, pero faltan éstos. Como no tenemos acceso a los ficheros donde MySQL almacena esa información, deberemos hacer un volcado de los datos a formato SQL (en texto), que luego cargaremos de forma directa contra el otro servidor.

Hay varios métodos y podríamos usar alternativas como clientes gráficos de escritorio para MySQL o usar la consola como hemos hecho antes, pero creo conveniente usar phpMyAdmin. ¿Por qué? phpMyAdmin es un cliente de MySQL que se instala en forma de programa web, por lo que nos olvidamos de restricciones de seguridad para las comunicaciones con equipos exteriores.

Su interfaz es muy intuitiva y ha tenido un éxito paralelo al de PHP y MySQL en el mundo de la web. Podríamos decir que es el estándar de facto para gestionar MySQL en estos casos. La práctica totalidad de servicios de alojamiento lo incorpora, al igual que la solución que hemos instalado en Windows (XAMPP). En Ubuntu no lo hemos instalado, pero podemos hacerlo de forma directa a través de un paquete llamado “phpmyadmin” en Synaptic.

Suponemos una migración de datos de nuestro equipo al alojamiento web. En caso contrario, deberíamos seguir los mismos pasos, pero alternando el origen y el destino.

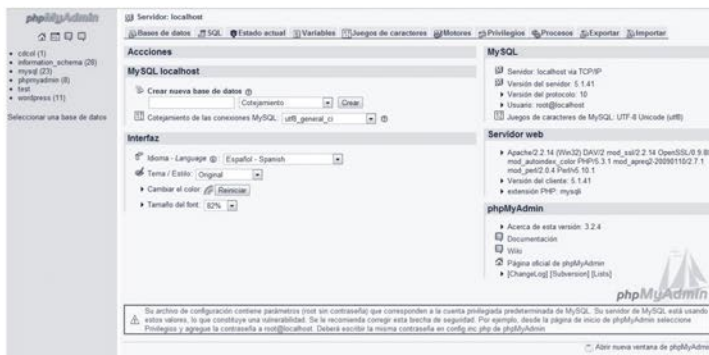


Figura 3.9. Aspecto del menú inicial de phpMyAdmin

Para ejecutar phpMyAdmin en nuestro equipo, abrimos el navegador y nos dirigimos a la siguiente dirección: <http://localhost/phpmyadmin/>. Recuerdo que debemos tener lanzados los servidores web y MySQL (en Ubuntu no es problema, pero en Windows hay que recordarlo y proceder al arranque desde panel de control de XAMPP).

Tenemos ante nosotros la interfaz gráfica de usuario de phpMyAdmin. En la parte inferior, es muy probable que se os avise de un fallo de seguridad. Dado

que estamos ante un servicio de pruebas y nuestro servidor no tiene acceso desde el exterior (suponemos que estamos trabajando en una red doméstica y el puerto 80 de nuestro equipo no lo tenemos redirigido al exterior), ignoramos el error.

En la parte izquierda, tenemos una barra lateral en la que se muestran las diferentes bases de datos, debemos comprobar que una de ellas será “wordpress”. Aparte, en la zona superior, una de las pestañas indica “Exportar”, ésta será la opción que utilizemos.

Como podemos observar en la figura de la página siguiente, ya tenemos delante de nosotros el diálogo de opciones para exportar bases de datos. ¿Qué debemos hacer?

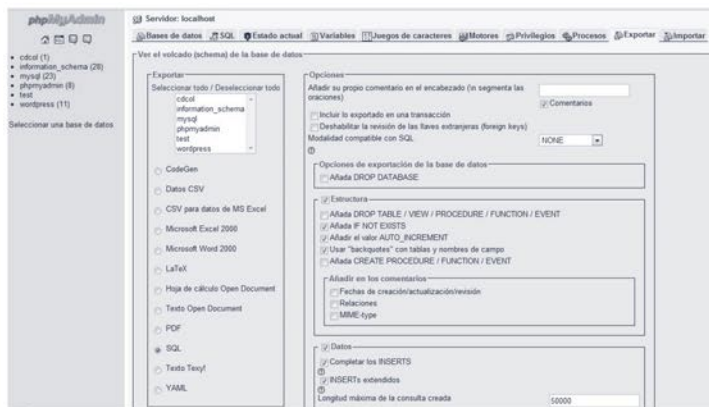


Figura 3.10. Diálogo de exportación de base de datos

- Dentro de exportar elegimos:
 - Base de datos: Wordpress.
 - Formato: SQL (viene por defecto).
- Opciones: Dejamos todo por defecto.
- Enviar (en la parte de abajo): En plantilla del nombre escribimos “wordpress_migracion”.

Hacemos clic en “Continuar” y se nos descargará un fichero SQL, con la estructura y todos los datos almacenados en la base de datos WordPress, de nombre “wordpress_migracion.sql”. Ya tenemos los datos, listos para ser subidos a la base de datos MySQL de nuestro alojamiento web.

Para subirlos, necesitaremos de la ayuda de phpMyAdmin. Accedemos a nuestro cPanel en nuestro *host* y en la parte de bases de datos, *databases*, elegimos phpMyAdmin. Si hay alguna diferencia entre ésta versión y la nuestra, será mínima y en todo caso para lo que lo queremos, no influye.

Nos dirigimos a la pestaña “Importar”, lo primero que debemos hacer es elegir el fichero a importar. Será justamente el que acabamos de importar. Dejamos el juego de caracteres tal cual. Debe ser UTF-8, el que viene por defecto que es también el que hemos elegido para generar el fichero. Siempre que nos sea posible deberemos usar este juego de caracteres, para no tener problemas con acentos, eñes y otros símbolos no incluidos en otros juegos de caracteres.

Procedemos a llevar a cabo la importación, si se interrumpe y nos aparece un mensaje de error del estilo a “Access denied”, es decir, acceso denegado, no os preocupéis. Lo más normal en las cuentas compartidas dentro de los grandes servicios de alojamiento web es que, para evitar problemas entre diferentes usuarios, las bases de datos siempre empiezan por un sufijo. En mi caso será “jcarazo_”, en el vuestro será “vuestro_nombre_de_usuario_”, recordad la finalización en barra baja.

Pero si tengo el fichero SQL ya creado, ¿cómo hago ahora para cambiar el nombre de la base de datos? Sólo hay que modificar dos sentencias en el fichero SQL: la de creación de la base de datos y la que especifica el manejo de la misma. Veámoslo sobre el código, deberemos modificar las siguientes dos líneas (situadas al comienzo del fichero SQL):

```
CREATE DATABASE `wordpress` DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
```

```
USE `wordpress`;
```

Por las siguientes dos líneas:

```
CREATE DATABASE `horcawd_wordpress` DEFAULT CHARACTER SET latin1 COLLATE latin1_swedish_ci;
```

```
USE `horcawd_wordpress`;
```


Para llevar a cabo la modificación, podemos usar Notepad++ o Geany. Tras hacer este cambio, ya sí que no deberemos tener problemas en la importación y, tras la subida de datos, aparecerá un mensaje en phpMyAdmin que nos confirmará: “La importación se ejecutó exitosamente, se ejecutaron 31 consultas”.

Estamos terminando de migrar los datos de nuestro equipo al alojamiento web, o viceversa, pero quedan un par de detalles por comentar. Lo vemos en el siguiente punto y, con éste, ya terminamos este capítulo.

3.3.3 Otras consideraciones

Como acabo de decir, hay dos detalles más que hay que considerar. El primero de ellos es modificar el fichero “wp-config.php” a las necesidades de configuración del alojamiento web. Deberemos modificarlo y volverlo a subir (con el cliente FTP sobrescribir el fichero).

El segundo es modificar las opciones básicas de la instalación. Este segundo paso podemos hacerlo desde el “Escritorio”. Accedemos a él (a través de su URL: http://direccion_blog/wp-admin) y tras autenticarnos, en “Ajustes”, “Generales”, modificamos lo necesario. Por lo menos, deberemos cambiar la dirección del sitio y la dirección del blog.

PERSONALIZACIÓN Y ACTUALIZACIÓN

Tras estos tres primeros capítulos de presentación de la plataforma web y de instalación del proyecto en servidores propios, llega el momento de tener libertad plena sobre el sistema para personalizarlo a nuestro gusto, sin restricción alguna. A lo largo de este capítulo veremos cómo instalar temas, complementos y cómo tratar con los menús y los llamados *widgets*. Trabajaremos siempre contra la plataforma que hemos montado en nuestro propio equipo. En capítulos posteriores, trataremos temas más técnicos que también tienen que ver con la personalización, la creación de temas y *plugins* a nuestro gusto, utilizando la API (*Application Programming Interface* “interfaz de programación de aplicaciones”) de WordPress.

4.1 INSTALACIÓN DE TEMAS

Cuando en el segundo capítulo hablábamos de personalizar nuestro blog en WordPress.com, uno de los aspectos que comentábamos era la posibilidad de modificar el tema gráfico que venía por defecto. Ya comentamos que los temas eran limitados y que las posibilidades de personalización se limitaban a poder modificar parcialmente el CSS.

Dado que tenemos el sistema instalado desde cero en nuestro equipo, ya no existen limitaciones. De hecho, podríamos hablar ahora de cómo hacer un tema desde el principio, pero dada la amplitud de dicho contenido, lo dejamos para un capítulo posterior. En este punto nos dedicaremos a comentar dónde podemos encontrar multitud de temas de WordPress gratuitos y listos para usar (además de algún sitio de temas *premium*, es decir, de pago), así como los pasos que hay que dar para instalarlos y desinstalarlos en nuestro sistema.

4.1.1 ¿Dónde encontrarlos?

Lo primero que debemos saber es dónde encontrarlos. Existe un repositorio oficial de temas para WordPress dentro de la web del propio proyecto. Podemos acceder a él a través del botón “Extend” presente en la parte superior de la web oficial del proyecto, <http://wordpress.org>. Dentro de esta sección de extensión, que es la traducción al español del vocablo inglés *extend*, además de temas, o *themes*, podemos encontrar un repositorio de *plugins*, que trataremos en el próximo punto; además de una sección para exponer vuestras ideas y quejas (las llamadas *kvetch*).

El repositorio oficial de temas (<http://wordpress.org/extend/themes/>) cuenta con una cifra cercana a los 1.500 temas diferentes para elegir. En la barra derecha del sitio tenemos tres clasificaciones de los temas en función de tres criterios: los más populares (en función del número de descargas), los más nuevos y los que han sido actualizados más recientemente.

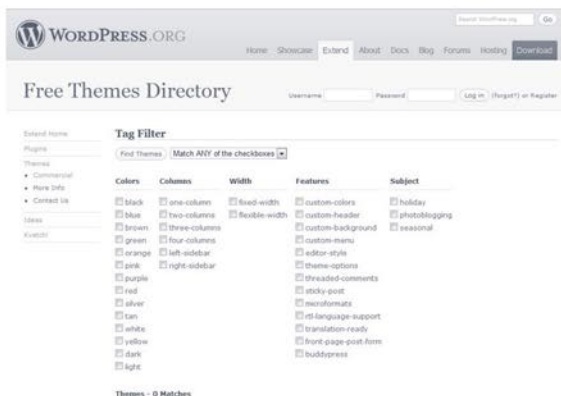


Figura 4.1. Directorio oficial de temas gratuitos

Para poder buscar entre tal cantidad de temas, tenemos a nuestra disposición el buscador típico (por palabras) y otro método más eficaz, un filtro por etiquetas. Para acceder a él, deberéis hacer clic en un enlace que dirá algo así como “check out our new filter and tag interface”, las palabras *filter* (filtro) y *tag* (etiqueta), seguro que las encontraréis aunque la frase haya cambiado.

¿Cómo utilizar este filtro? Es muy simple, tenemos una serie de condiciones englobadas en una serie de grupos: color (*color*), columnas (*column*),

anchura (*width*), características (*features*) y temática (*subject*), que podremos señalar en función de nuestras necesidades. Posteriormente, sólo tenemos que pedirle que busque temas en función de esos criterios (*find themes*). Podemos afinar nuestra búsqueda eligiendo si queremos que se cumplan todas nuestras elecciones (*match all of the checkboxes*) o sólo alguna de ellas (*match any of the checkboxes*).

Encontraréis temas de todo tipo. Desde algunos que imitan el aspecto de la red social Facebook a otros que estaréis muy acostumbrados a ver porque los usan blogs que visitáis a menudo. Para no ir más lejos, en Linux Hispano (<http://www.linuxhispano.net>), blog del que soy uno de sus fundadores y administrador, utilizamos el tema oficial de WordPress 3, “Twenty-Ten”; parcialmente modificado.

Aunque el tema no sea exactamente de vuestro gusto, podemos modificarlo. Algunos, como veremos a continuación, tienen una interfaz gráfica de configuración, algo verdaderamente rápido y sencillo. Si no dispone de esta opción el tema que queremos modificar, siempre podremos cambiar secciones del código. En un capítulo posterior veremos cómo crear un tema y cómo modificarlo, escribiendo directamente el código. WordPress facilita enormemente este aspecto en relación a otros sistemas gestores de contenido.

Dado el éxito que ha tenido la plataforma desde su creación, en la red podemos encontrar multitud de sitios dedicados a los temas de WordPress. Algunos de ellos, incluso llegan a tener una cantidad parecida de temas. Antes de elegir vuestro tema definitivo, recomiendo encarecidamente que os deis una vuelta por vuestro buscador favorito y consultéis todos los sitios que podáis.

Algunos de ellos tendrán un coste económico, pero serán los menos. Existen temas gratuitos de mucha calidad y gracias a que posteriormente podemos modificarlos, no creo que tengáis problemas en este sentido. Si de verdad os gusta un tema de pago, siempre podréis pagar por él (normalmente los precios no son en absoluto abusivos) y tendréis la posibilidad (en función de la licencia del mismo) de modificarlo para adaptarlo a vuestras necesidades. Existen incluso profesionales dedicados a crear y adaptar temas para WordPress. Podéis buscar y solicitar sus servicios en la red.

4.1.2 Descarga e instalación manual

Atahualpa es el nombre del que es considerado último emperador incaico y también es el nombre del tema que vamos a usar como ejemplo en este punto y el siguiente. ¿Por qué éste? Aparte de su popularidad, lo he probado personalmente y su interfaz de configuración es de las mejores que he visto. En capítulos posteriores

veremos cómo manejar a nivel de código el tema, pero existen temas que tienen interfaces que facilitan mucho todo y éste es uno de esos casos.

¿Dónde puedo encontrarlo? En el repositorio oficial de temas de WordPress que hemos presentado antes, buscándolo por su nombre o directamente entrando en su dirección web (<http://wordpress.org/extend/themes/atahualpa>). En la parte derecha de la pantalla tenemos dos botones, uno de previsualización (*preview*) y otro de descarga (*download*). Hacemos clic en este segundo botón y se descargará el tema comprimido en formato ZIP.

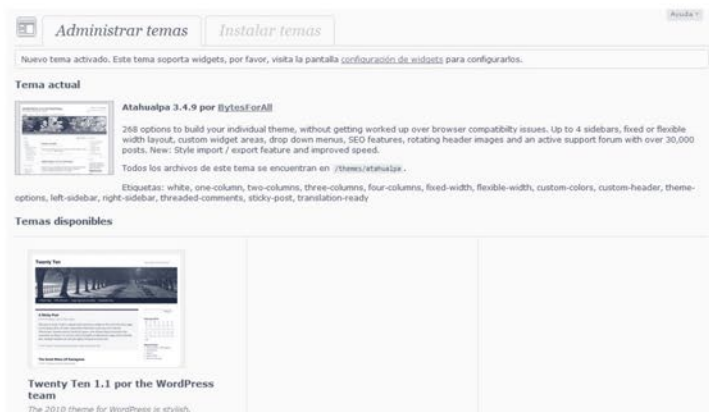


Figura 4.2. Interfaz de administración de temas

Los pasos que hay que dar son los siguientes:

1. Descomprimir el fichero ZIP, tanto en Windows como en Ubuntu podemos hacerlo directamente haciendo clic derecho sobre el fichero con el menú contextual.
2. En caso de que existiera un fichero “readme.txt”, debemos leerlo por si acaso existe alguna instrucción especial.
3. Colocamos el directorio que hemos obtenido como resultado de descomprimir el fichero, en este caso “Atahualpa”, en el directorio “/wp-content/themes/” de nuestro blog. Lo haremos copiando

directamente si trabajamos en local, y en caso de ser sobre el servidor remoto, a través del cliente FTP.

4. Finalmente, ya podemos activarlo a través del panel de administración. En la pestaña “Apariencia”, elegimos “Temas” y en la parte inferior de la pantalla nos aparecerá “Atahualpa” como tema disponible.
5. Hacemos clic sobre activar y ya tendremos el tema en nuestro portal de cara al público.

Éste es el método tradicional de instalación y podríamos decir que el más seguro. A continuación presentamos otro método, mucho más automatizado que éste. Dado que con éste nos aseguramos no tener problemas, suelo recomendar utilizar este primero.

4.1.3 Descarga e instalación automática

Dentro de la administración de temas de WordPress, además de la pestaña de título “Administrar temas”, tenemos otra que dice explícitamente “Instalar temas”. Se trata de una integración del repositorio oficial de temas del CMS dentro del panel.

¿Cómo funciona? Tenemos una interfaz de búsqueda con el sistema de etiquetas que hemos explicado antes. Buscamos el tema que nos gusta y directamente podemos instalarlo. Veamos un ejemplo.

Buscamos el tema de título “Pixel”. Aparece el resultado de la búsqueda y entre los temas que aparecen está el que queremos. Podemos hacer la vista previa o instalarlo. Si nos convence procedemos a instalarlo. Nos aparecerá una pantalla como la que muestro a continuación.

Automáticamente, el sistema se encargará de buscar el fichero, descargarlo, descomprimirlo e instalarlo. Mientras sucede el proceso, que es muy rápido, se nos informará del resultado de las distintas operaciones. Si todo ha concluido exitosamente, podemos pasar a activarlo.

Como veis el método es muy cómodo, pero tiene limitaciones. La más importante es que sólo podemos instalar temas de los que hay en el repositorio oficial y lo segundo es que es más susceptible a errores que el anterior. Por lo demás, son todo ventajas y para los usuarios más noveles será muy cómodo.



Figura 4.3. Instalación automática de temas

4.1.4 Permisos apropiados

WordPress es un sistema lleno de mecanismos que ayudan al administrador para que tenga que hacer lo mínimo fuera del sistema. Acabamos de ver cómo se instalan temas de forma automática, sin haber salido del panel de administración, pero no sólo tenemos esa opción. En el punto siguiente veremos cómo editar ficheros del tema desde el mismo panel y al final de este capítulo explicaremos una de las características más importantes de WordPress, la actualización automática. Este mecanismo de actualización automática afecta a todas las partes del sistema susceptibles de recibir mejoras tanto en funcionalidad como en seguridad: el sistema en sí mismo, además de los temas y *widgets*.

¿Qué problemas podemos tener a la hora de utilizar estos mecanismos automáticos? La mayoría de ellos estarán provocados por la existencia del sistema de permisos UNIX, que es el que usan los sistemas operativos Linux y la mayoría de los servidores web que podamos contratar (que también serán Linux).

El sistema de permisos UNIX es una de las características esenciales para dotar de seguridad a los equipos. Dado que este tipo de sistemas nació en entornos de muchos usuarios, este tipo de configuraciones de permisos y atributos de los archivos viene desde muy lejos. En sistemas operativos como Microsoft Windows, que progresaron del monousuario al multiusuario, los sistemas de ficheros son más primitivos en este sentido y no tendremos problemas para nuestro caso.

El hecho de tener problemas, aunque parezca paradójico, es bueno. Quiere decir que hay unas restricciones que aseguran que nadie va a tocar nuestros ficheros sin nuestro permiso. Podríamos explicar ampliamente el sistema de ficheros UNIX y los permisos en el mismo, pero como no es el sentido de este texto, nos limitaremos a aplicarlo a nuestro caso.

¿Por qué se explica ahora? Estamos entrando de lleno en la sección de instalación, configuración y actualización de distintos componentes dentro de WordPress. Hasta el momento, el problema de los permisos no había existido, pero, a partir de ahora, será probable que nos encontremos con él.

Aunque en el fondo es lo mismo, la forma de hacerlo es diferente. Explicaremos primero cómo resolver este tipo de problemas sobre Linux (para los que hacéis pruebas en local con este sistema operativo) y luego lo explicaremos para el caso del servidor remoto (haciendo uso del cliente de FTP).

Usaremos dos tipos de permisos, básicamente:

- 755: Permiso de lectura y ejecución para cualquiera, todos los permisos para el propietario del archivo.
- 644: Permiso de lectura y escritura para ti, pero sólo de lectura para el resto.

El significado de estos números y su posición depende de tres grupos de usuarios: el propietario, los usuarios del grupo del propietario y el resto; y de tres tipos de permiso: lectura, escritura y ejecución. En la red podéis encontrar más información sobre este tema, para nuestras necesidades con esto será suficiente.

En la siguiente tabla os presento los permisos que deben tener cada directorio y cada fichero.

Nombre	Ubicación	Permiso
Directorio raíz	./	755
Directorio de <i>includes</i>	./wp-includes/	755
.htaccess	./htaccess	644
Administración	./wp-admin/index.php	644
Scripts JavaScript administración	./wp-admin/js/	755
Temas	./wp-content/themes	755
<i>Plugins</i>	./wp-content/plugins	755
Directorio administración	./wp-admin	755
Directorio de contenido	./wp-content/	755

Tabla 1 Permisos apropiados para directorios y ficheros

Ya sabemos qué permisos deben tener los ficheros y directorios de nuestro sistema, veamos cómo cambiarlos en nuestras instalaciones. Recordad que este tema no afecta a Windows, por lo que sólo nos centraremos en equipos Linux y en cómo hacerlo con el cliente de FTP.

4.1.4.1 LINUX

Lo primero que debemos hacer es abrir la consola y dirigimos al directorio donde se encuentra WordPress, recordad que para cambiar de directorio usamos el comando “cd” seguido de la ruta a donde queremos ir. Una vez allí, ya podemos usar la orden para cambiar los permisos: “chmod”, del inglés *change mode*. Antes de eso, comprobamos con el comando “ls -l” los permisos reales de cada fichero y directorio.

Usaremos el comando como sigue a continuación, para cambiar los que sean incorrectos:

```
chmod permiso archivo/directorio
```

Por ejemplo, para nuestro caso (incluyo “sudo” porque el directorio en el que trabajamos, “/var/www”, es del superusuario):

```
chmod 644 ./wp-admin/index.php
```

4.1.4.2 SERVIDOR REMOTO

Haremos exactamente lo mismo que en el punto superior, pero esta vez manejaremos FileZilla para subir y bajar ficheros y también para cambiar permisos. Para ello, deberemos acceder al directorio o fichero en cuestión, pulsando el botón derecho veremos que en el menú contextual que aparece tenemos: “Permisos de archivo...”. Dentro de él, aparecerá un formulario donde podremos elegir el valor numérico del permiso en sí. Podemos ver en la parte superior cómo cambian los permisos para los grupos que he comentado antes. Al terminar, pulsamos sobre “Aceptar” y el cambio se habrá realizado. Para comprobar qué permisos están bien y mal, FileZilla los muestra por defecto en una columna donde aparece el valor numérico del mismo.

Podéis ver el aspecto del formulario de cambio de permisos en la siguiente imagen.



Figura 4.4. Formulario de gestión de permisos

4.1.5 Modificación básica

Como ya hemos comentado, algunos temas como “Atahualpa” incluyen una interfaz de configuración propia para acercar la personalización del tema a todos los usuarios. Aunque esta opción es rara, en los temas como éste que la incluyen, supone una ayuda enorme para el administrador.

Podréis encontrar las opciones propias del tema gráfico, en el menú de administración, en la sección “Apariencia” dentro de la barra lateral. Para este caso es “Atahualpa Theme Options”.

¿Qué nos encontramos dentro de este tipo de administradores? Pues depende mucho del tema, como os digo, no es nada estándar y este tipo de gestores específicos, aunque se integran dentro del administrador de WordPress, están diseñados sin seguir unas reglas básicas. La tendencia es que cada vez más temas los posean y probablemente, en una versión posterior del CMS, se dicten unas directrices.

¿Qué nos ofrece el gestor de opciones del tema “Atahualpa”? Como podéis ver si habéis accedido al gestor, tenemos desde una herramienta para configurar opciones propias del SEO (*Search Engine Optimization* “mecanismos de optimización para motores de búsqueda”, hablaremos ampliamente en un capítulo posterior), hasta cómo deben aparecer las entradas o las páginas, pasando por opciones más propias de cada tema como son el aspecto de la cabecera.

Dado que cada gestor es diferente, en caso de que lo tenga el tema que vayáis a utilizar, os dejo que probéis vosotros mismos las opciones que aparecen. Aprenderemos el método general para todos: la creación y edición de temas para WordPress directamente sobre el código en un capítulo posterior.

4.2 INSTALACIÓN DE PLUGINS

Antes de hablar de los *plugins* y de su instalación y configuración, en caso de que la tengan, vamos a explicar de forma muy breve qué son y qué posibilidades nos brindan.

4.2.1 ¿Qué es un plugin?

Los *plugins*, o complementos, son módulos software que añaden una característica o servicio al sistema, en este caso WordPress. Los complementos están presentes en gran parte del software moderno. De hecho, son un factor clave del éxito en algunas aplicaciones, por ejemplo, en el caso del navegador Mozilla Firefox. Es más frecuente encontrarlos dentro de programas de código libre que en los de código privado, ya que el propio carácter abierto del software anima a ampliarlo y a ofrecer algún mecanismo para facilitar la creación de éstos (normalmente a través de una API).

Además de *plugin* o complemento, también podéis encontrar programas que lo llamen extensiones o por el nombre en inglés, *add-on*. En términos generales, uno de los *plugins* más famosos y que prácticamente todos conoceréis es el de Flash. Gracias a él podemos visualizar contenido desarrollado en esta tecnología dentro de los sitios web. Se instala al navegador de forma casi automática cuando llegamos a un sitio que utiliza esta tecnología de Adobe, probablemente sea el *plugin* más extendido del mundo. En el mundo del desarrollo web están muy de moda los *plugins* en jQuery y, como no podía ser de otra forma, en WordPress.

Muchas de las funcionalidades que hoy en día están integradas dentro de WordPress fueron en su día *plugins* aparte, que el sistema ha ido adquiriendo como propios gracias a su funcionalidad y popularidad.

Entre los complementos más famosos para WordPress que podemos encontrar en la actualidad, destacaremos los siguientes.

- Akismet: Probablemente el mejor mecanismo anti-spam disponible para blogs.
- All in One SEO Pack: Multitud de mejoras para optimizar el posicionamiento en los motores de búsqueda.
- Google Analytics for WordPress: Complemento para permitir la interacción con el sistema de estadísticas en línea de Google.
- WP Super Cache: Posibilita la creación de una caché en nuestro sistema para descargar los recursos del servidor. Muy importante en alojamientos web.
- Contact Form 7: Facilita la creación de un formulario de contacto.
- Broken Link Checker: Comprueba enlaces que no funcionan en entradas, comentarios y páginas.

Esta lista la podríamos alargar durante páginas y páginas. En el siguiente capítulo, explicaremos cómo optimizar el blog, por lo que explicaremos más detenidamente los complementos que usaremos. Veamos ahora dónde encontrarlos, cómo instalarlos y, en caso de que tengan opciones de configuración, cómo acceder a ellas.

4.2.2 ¿Dónde encontrarlos?

Al igual que los temas, los *plugins* tienen un repositorio oficial dentro del proyecto WordPress. Podemos acceder a él a través del botón “Extend” presente en la parte superior de la web oficial del proyecto, eligiendo *plugins* en lugar de *themes*.

Es básicamente el mismo repositorio, pero aplicado a complementos en lugar de a temas, así que todo lo dicho anteriormente respecto a búsqueda, filtros y descargas de los mismos también es válido para este caso.

4.2.3 Descarga e instalación manual

A diferencia de los temas, en los que sólo podemos activar uno, podemos activar tantos *plugins* como necesitemos. Por lo tanto, tenemos un concepto nuevo, primero instalación y luego activación y desactivación en función de nuestras necesidades. Mi consejo es que sólo se activen los complementos que verdaderamente necesitamos, los que no, es mejor tenerlos desactivados e, incluso, desinstalados.

¿Por qué? Por espacio, por consumo de recursos del equipo y por posibles fallos de seguridad. Recordad que al añadir funcionalidad, también pueden añadirse agujeros, por lo que hay que ser cauto con los *plugins* y activar sólo los que necesitamos.

Al igual que el en caso de los temas, los pasos que hay que dar para instalar manualmente un *plugin* son: búsqueda, descarga, descompresión (vienen en ZIP) y colocación en el directorio oportuno. Aquí está el único cambio, antes era el directorio “/wp-content/themes”, ahora es “/wp-content/plugins”.

Lo siguiente será, dentro del panel de administración, activarlo. La activación suele ser instantánea a excepción de casos muy concretos. Uno de ellos es el de Akismet, que necesitas darte de alta en Automattic (gratuito a no ser que lo uses con fines comerciales, que sí tiene un coste) e introducir una clave que te envían para evitar abusos.

El tema de los permisos apropiados para los *plugins* podéis verlo directamente en la tabla que hemos mostrado cuando hablábamos de los temas, el directorio “/wp-content/plugins” debe tener el permiso 755.

4.2.4 Descarga e instalación automática

La opción más rápida es buscar directamente desde nuestro panel de administración el tema requerido y proceder a instalarlo. Para ello, en la barra lateral, tenemos una sección llamada “Plugins” en la que podemos encontrar la opción “Añadir nuevo”. Allí lo buscamos por término, autor o etiqueta y en la lista de resultados en forma de tabla que nos aparece, tenemos la opción directa de instalarlos: “Instalar ahora”. El sistema nos preguntará si estamos seguros y paso seguido tendremos el *plugin* instalado, sólo a falta de activarlo para tenerlo en marcha.

Buscar | Resultados de la búsqueda | Subir | Destacados | Populares | Recientes | Actualizados recientemente

Etiqueta: Buscar plugins

1 2 3 ... 28

Nombre	Versión	Valoración	Descripción
myEASyBackup Detalles Instalar ahora	1.0.3	☆☆☆☆☆	Backup, restore, migrate your WordPress installation, both code and MySQL tables, with a single click. Check out the myEASyBackup for WordPress video: When performing a backup, myEASyBackup creates a single file, called "data set", that includes your data in compressed format (.zip). Data sets are saved outside the WordPress installation directory to avoid someone else discover the links an... Por Ligo Grandolin aka "carnelio".
leank.me Detalles Instalar ahora	1.1.6	☆☆☆☆☆	leank.me automatically publishes a tweet to your Twitter account, a status update to your Facebook profile and/or Fan Page, and a Buzz to your Google Buzz profile whenever you publish a new post in your WordPress website. What can you do with leank.me? Publish automatically to your social networks when you publish a new post in WordPress. Works with Posts, Pages, and Custom Post Types. (Requires... Por Leo Avellan @leankme
Twitter Feed for WordPress Detalles Instalar ahora	0.1	☆☆☆☆☆	The WordPress Twitter Feed Plugin lets you simply output any users tweets into your WordPress page, template or sidebar! You can customize the username, number of tweets, and style of output. This very simple plugin adds a small JS file to your header to let this happen! Twitter Feed WordPress Plugin plugin homepage. More WordPress Plugins. Follow @alexmosaSEO on Twitter. Alex Moss. SEO Manches... Por Alex Moss.
Highlighter Detalles Instalar ahora	1.0.9	☆☆☆☆☆	Highlighter is a WordPress plugin that allows authors to actively engage their audience. They can direct readers attention to any part of the post by highlighting any word, sentence or image. Readers can now highlight, comment and share any word, sentence, paragraph or image on a blog post as well, giving the audience unprecedented ability to interact with content. Watch a video of what it can d... Por ANNGType, Inc...
myEASywebally Detalles Instalar ahora	1.0.3	☆☆☆☆☆	How much time do you spend to perform your WordPress blogs/sites maintenance? Are you sure the new plugins you are going to install will not break your WordPress installation? Are all your plugins and your template ready for the latest version of WordPress? When using WordPress to professionally produce sites for your customers, doing the proper maintenance can really take a lot of time. When... Por Ligo Grandolin aka "carnelio".
Book.31 Highlighter, AutoPublisher, and AutoTweeter	6.0	☆☆☆☆☆	Makes Your Web-Site/Blog VERY Viral! Put a horizontal/vertical highlighter pen; anyone can make highlights on any part of your web page and automatically post them to any Social Networking, Social Bookmarking site such as Facebook, Twitter, LinkedIn, MySpace, Digg, Delicious etc. (Auto includes a Short URL to your highlighted page.)

Figura 4.5. Resultados de la búsqueda de plugins para la etiqueta AJAX

Existe un método intermedio, en el que debemos descargar el fichero comprimido de la página del complemento y usar la opción “Subir”, que aparece en esta misma interfaz gráfica encima de la tabla donde se muestran los *plugins*. El sistema subirá el fichero y lo activará de forma automática. El proceso es prácticamente el mismo que ahora, pero nosotros tenemos que buscar el complemento en su fichero ZIP.

4.2.5 Activación y configuración

Volviendo al gestor de *plugins*, nos encontramos que ya tenemos el complemento en cuestión instalado y sólo falta activarlo. Para ello, deberemos hacer clic sobre el botón “Activar” y la activación se efectuará, en caso de que haga falta hacer algo más, como en el caso de Akismet, se nos informará debidamente. Podéis ver el mensaje para este caso particular en la siguiente figura:



Akismet casi está preparado. Debes introducir tu clave de API de Akismet para que funcione.

Figura 4.6. Información para la activación de Akismet

Aunque este tipo de procesos tras la activación no son frecuentes, sí que lo son los *plugins* que necesitan una configuración posterior. La configuración de cada *plugin* es particular para cada caso, por lo que, por ahora, sólo explicará cómo acceder a la misma.

En la tabla de gestión de *plugins*, tenemos dos partes: la izquierda, con el nombre del complemento y la opción de activación y desactivación; y la derecha, donde encontramos un resumen del funcionamiento del *plugin*, enlaces a la web del desarrollador y del *plugin* y, en caso de que lo haya, enlace a las opciones de éste. En ocasiones, este enlace a las opciones se encuentra incluso dentro del resumen, por lo que miradlo detenidamente en caso de que lo necesitéis.

¿Qué tipo de configuración posterior puede ser necesaria? Por daros sólo un par de ejemplos, cuando estamos tratando con el complemento que maneja las estadísticas de Google, debemos introducir los datos de nuestra cuenta de estadísticas. Otro ejemplo sería cuando manejamos un *plugin* para generar el fichero XML con el *sitemap* del blog (ya contaremos qué es exactamente el mapa del sitio), debemos indicar algunos parámetros.

4.3 MENÚS Y WIDGETS

La personalización que permite WordPress de forma inmediata no se refiere sólo a temas y a complementos. Además de ellos, tenemos otros dos elementos, menos autónomos, pero igualmente importantes para mejorar la estética y funcionalidad de nuestro cuaderno de bitácora.

Nos referimos a los menús y a los llamados *widgets*. Ya los hemos conocido de forma directa cuando tratábamos en el capítulo segundo la personalización de un blog dentro del servicio WordPress.com.

¿En qué se diferencian aquellos menús y *widgets* que explicábamos entonces de los que tenemos ahora? Pues básicamente en nada, pero hay un detalle importante entre ellos. Antes teníamos sólo el control básico que nos permitía el panel de administración y disponíamos de los que nos querían asignar. Ahora podemos modificar los menús a nivel de código y, sobre todo, tenemos a nuestra disposición miles de *plugins*, algunos de los cuales nos ofrecen *widgets* para presentar su funcionalidad. Buscando por la etiqueta “widget” dentro del repositorio oficial de complementos, podréis ver la cantidad de funcionalidades extras que podemos añadir a nuestro blog y presentar al usuario a través de un *widget*.

4.4 ACTUALIZACIÓN

Llegamos al último punto de este capítulo. Desde el comienzo hemos repetido que una de las mayores ventajas de WordPress es la posibilidad de actualizarse de forma automática, indicándonos, además, cuándo es necesario.

Si tratáis con otros CMS diferentes a éste, comprobaréis que, por un lado, en cada actualización hay un proceso que hay que seguir, que, más fácil o más complejo, requiere de una atención extra que hace que se dilate el proceso de actualización. Ningún software es perfecto y los fallos se detectan prácticamente a diario. Si no actualizamos cuanto antes, tendremos verdaderos agujeros de seguridad, además de imposibilitarnos a nosotros mismos disfrutar de las mejoras que se incluyan.

La otra gran ventaja de WordPress es que te avisa de forma directa de que hay actualizaciones disponibles, tanto del sistema en sí como de los diferentes temas y complementos (que también pueden tener mejoras y fallos de seguridad). En otros sistemas, hemos de estar pendientes nosotros de posibles actualizaciones. Ya sea a través de RSS, apuntados a una lista de correo o visitando regularmente los portales de todo este software (imaginad si instalamos diez o veinte complementos), debemos estar pendientes nosotros de qué versión tenemos y qué versión van a sacar. En WordPress nos preocupamos de todo esto. ¿Cómo? Él comprueba automáticamente posibles actualizaciones y nos informa cada vez que entramos al panel de administración.

Aunque sean muy parecidas en su desarrollo, detallaremos, por un lado, el proceso de actualización del sistema, y por otro, el de actualización de temas y *plugins*.

4.4.1 Actualización del sistema

La actualización del sistema es el punto más importante de las actualizaciones. Aunque la actualización sea básica para proveer del mayor nivel de seguridad, también es cierto que las actualizaciones pueden suponer problemas de incompatibilidad y que el proceso puede no ser todo lo correcto que desearíamos. Aunque no son en absoluto frecuentes, para evitar todo este tipo de problemas, recomiendo encarecidamente hacer una copia de seguridad de todos los datos del blog antes de proceder. En el capítulo siguiente detallaremos cómo hacerlo.

Cuando actualizamos el sistema, la versión que va a ser sustituida descarga la nueva versión, la descomprime, la sustituye por sí misma y activa la nueva. Dentro de todo este proceso se comprueban posibles incompatibilidades con temas o complementos, además de chequearse en cada momento si aparecen problemas de cualquier tipo. Antes de que apareciera el método de actualización automático actual, el proceso era manual al igual que en el resto de los CMS y, básicamente, se trataba de reinstalar la versión nueva sobre la antigua.

¿Cómo sabremos cuándo hay que actualizar? WordPress nos avisará nada más entrar al panel de administración. Aquí hay un punto en el que debemos ser precavidos. Si tenemos instalado WordPress en idioma castellano, la versión traducida se actualizará algo más tarde que la versión inglesa.

Esto puede dar lugar a malentendidos. Si vemos un mensaje de actualización de WordPress, pero que no es del lenguaje español, recomiendo ignorarlo. Es posible que estemos un día sin tener la última versión, pero también es cierto que actualizar con la versión inglesa puede traernos aún más quebraderos de cabeza. ¿Qué debemos hacer entonces? Esperar de forma paciente a que salga la versión española. Generalmente, suele salir entre uno y dos días después de la versión en inglés.

Cuando salga la versión en español, WordPress nos informará al entrar de que está disponible a través de un mensaje llamativo, además de quedarse situado un número justo al lado del título “Actualizaciones”, en la barra lateral dentro de la sección “Escritorio”.

¿Qué mensaje debemos esperar para estar seguros de que es la versión española? WordPress nos informará de que está disponible la versión “en_US” (inglés de Estados Unidos de América, la primera en salir) y luego, al lado, otro botón indicará “es_ES”, es decir, español de España.

Tras pulsar sobre éste, el sistema nos pedirá la contraseña de nuestro servicio FTP y él mismo se encargará de proceder a actualizarse. Por medio de mensajes, se nos comunicará paso a paso el estado de la actualización, para informarnos, finalmente, del resultado. El portal estará disponible durante todo el tiempo. El tiempo de actualización es verdaderamente rápido.

Finalmente, tras actualizar, seguiremos trabajando de igual forma que hasta entonces. Existen mejoras entre versiones, pero no suelen ser lo suficientemente grandes como para que notemos un cambio brusco. En todo caso, será de esperar un cambio de este tipo, sin perder la estructura, cuando se salte a la versión 4 y a la versión 5 de WordPress (si siguen con este sistema de numeración).

4.4.2 Actualización de temas y plugins

No menos importante que la actualización del sistema gestor de contenido, es la actualización de temas y complementos. Gracias a que existe un repositorio central de temas y de *plugins*, el sistema puede interactuar con él para ser consciente de estas actualizaciones.

¿Cómo sabemos que disponemos de una actualización de un tema o de *plugin*? Al entrar en el panel de administración, en la pestaña “Escritorio”, en la barra lateral, veremos un número (el número de actualizaciones disponibles en ese momento). Al hacer clic sobre él nos aparecerá explicado de qué actualización estamos hablando y nos aparecerá un botón para llevarla a cabo: “Actualizar plugins” o “Actualizar temas”. De la misma forma, también aparecerá un número en la sección que tiene una actualización disponible. En la imagen siguiente podéis ver el ejemplo para el caso concreto del *plugin*.



Figura 4.7. Actualización de plugin disponible

Al hacer clic sobre la misma llegamos a la misma información que por el camino anterior, pero presentada de otra forma. En el complemento correspondiente elegimos “Actualizar automáticamente”, y, siguiendo el proceso habitual, la actualización se llevará a cabo.



Figura 4.8. Plugins susceptibles de actualización

Con este punto llegamos al final de este capítulo. Ya sabemos cómo personalizar de forma básica y mantener al día los aspectos esenciales de nuestro blog. El siguiente paso es optimizarlo para sacarle el máximo provecho gracias a las posibilidades de expansión que nos ofrece WordPress.

OPTIMIZACIÓN

Conocemos ya cómo poner en marcha WordPress sin depender del servicio de Automattic y sabemos cómo personalizarlo de forma básica, aprovechando además opciones muy interesantes como la actualización automática. Sin embargo, todavía queda bastante por hacer para conseguir un sistema optimizado. La instalación básica de WordPress incluye muchas más optimizaciones que otros sistemas, pero hay muchas otras que aún no se incluyen por defecto. Existen optimizaciones que son ideales para blogs comerciales, otras que están más orientadas a quien tiene un blog personal y, la mayoría de ellas, a todos los usos que podamos darle a un blog.

Desde la adaptación a dispositivos móviles a la monetización del blog, pasando por su integración en las redes sociales o la mejora de los resultados en las búsquedas de los buscadores, cualquier sitio basado en WordPress ganará muchas visitas, referencias e incluso euros, si sigue los pasos que detallo a continuación.

5.1 MANEJO DEL SPAM

El origen del término spam es, cuando menos, curioso. Justo antes de que comenzara la Segunda Guerra Mundial, Hormel Foods, una empresa charcutera estadounidense, lanzó al mercado una carne en lata que fue consumida a lo largo de la guerra por buena parte de la tropa aliada. El nombre de la misma era Hormel's Spiced Ham, que abreviado se quedaba en "spam". Pasado este convulso período de la historia de la humanidad, en la década de los cincuenta, la empresa decide comercializar en el mercado esa misma carne.

Tras extenderse en buena parte del mercado estadounidense y británico, el grupo de humoristas Monty Python (el mismo que le da nombre al popular lenguaje de programación Python) comenzó a burlarse de la carne en lata y, a raíz de esta circunstancia, comenzó a denominarse de forma coloquial spam al correo no deseado. Con el tiempo, gracias a la ampliación de medios de comunicación que hay en la red, se ha exportado el término a otros conceptos: mensajes cortos de texto no deseado o comentarios no deseados (que es el caso que nos ocupa).



Figura 5.1. Imagen de los Monty Python a final de la década de los sesenta

El spam es un problema de gran envergadura y uno de los grandes lastres de la Web 2.0. En el momento en que se permite la interacción inmediata, sin filtros de por medio, del usuario en la web, surgen los problemas. Y, en la mayoría de los casos, no hablamos de problemas derivados de copias ilegales, de apología de ciertas actitudes o de comentarios que podrían ser delitos, sino de la publicidad. En el momento en que el anunciante ve el resquicio para conseguir de forma gratuita su producto, se abre el problema.

Tras las duras sanciones legislativas establecidas en todas las naciones desarrolladas, el problema no son los grandes anunciantes, sino los anuncios generados por ordenadores infectados que propagan sin control, y sin una responsabilidad legal clara, anuncios que suelen centrarse en una serie de temas muy concretos: medicamentos (Viagra y similares, dietéticos, contra la alopecia), pornografía y temas de actualidad.

Los ordenadores infectados generadores de spam se dedican a bombardear mediante correos, mensajes en foros o comentarios en blogs (la parte que nos ocupa), llegando a generar un volumen de tráfico tal, que llega a afectar a la estabilidad de ciertos sitios (sin contar la incomodidad para los lectores y para los administradores). Como veis, el problema es grave y afecta incluso a sitios recién creados, por lo que nadie está a salvo de él.

5.1.1 Mecanismos de defensa

Puesto que el problema existe, hemos de darle una respuesta lo más inmediata posible. Hemos de conseguir la mejor relación entre tres variables:

- Comodidad para el usuario.
- Tiempo invertido por el administrador.
- Cantidad de spam que pasa el filtro.

En WordPress tenemos diferentes alternativas. Explicaremos tres, de la menos eficaz a la más eficaz, que a su vez pueden estar combinadas.

5.1.1.1 COLA DE MODERACIÓN

Es el mecanismo más antiguo de defensa contra el spam. Se basa en el principio de que nada sale a portada si no ha sido antes aprobado por el administrador del blog. Tiene dos grandes inconvenientes: requiere de mucha dedicación por parte del propietario del blog y, a la vez, resta mucha interacción entre la comunidad. Los usuarios pierden interés en comentar y participar si ven que pueden pasar horas e incluso días hasta que es aprobado su comentario.

¿Cómo funciona en WordPress? Existen tres colas de comentarios:

- Spam: Han sido considerados como spam de forma preventiva por algún complemento que tengamos instalado o han sido marcados como spam por parte del administrador.
- Pendiente: El usuario debe decidir si el comentario es apropiado o es spam.
- Aprobado: Aquí están los comentarios que ya han pasado el filtro y son vistos por el blog.

Estas tres colas son visibles nada más iniciar sesión como administradores en WordPress. Además, si la cola de pendientes no está vacía, un indicador numérico similar al de las actualizaciones estará presente en el botón de los comentarios en la barra lateral.

¿Cómo gestionamos las tres colas? Accediendo a cada una de ellas podremos ver los comentarios y decidir en cada momento si aprobar, marcar como spam o borrar el comentario.

Para mejorar la gestión de estas colas, WordPress nos ofrece unos mecanismos básicos que pueden ahorrarnos mucho tiempo con pocos pasos. Os presentamos los más relevantes, aunque podéis acceder a ellos y configurarlos debidamente dentro de la opción “Debate” del menú “Ajustes” en la barra lateral.

- Para que un comentario aparezca directamente, ha de tener un comentario previamente aprobado por el administrador (muy útil, ya que los usuarios suelen repetir si comentan y sólo se produce la molestia la primera vez).
- Crear una lista negra. Si siempre recibimos comentarios del tipo: “viagra”, podemos incluir esa palabra en la lista negra y cuando un comentario contenga esa palabra, automáticamente serán marcados como spam.
- Obligar a los usuarios a registrarse para comentar. Nos ahorramos prácticamente todo el spam, pero será verdaderamente incómodo para los usuarios comentar.

Además de estas opciones, existen otras que vosotros mismos podéis modificar según vuestro gusto y necesidades.

¿Qué os recomiendo? Aunque todas estas características son muy útiles, la idea más correcta para atraer público a nuestra web es facilitar el diálogo entre los usuarios, por lo tanto la idea es evitar técnicas restrictivas. Debemos abrir al máximo las facilidades, por lo que deberemos desactivar opciones que vienen marcadas por defecto como “El autor del comentario debe tener un comentario previamente aprobado”, para que el comentario aparezca inmediatamente.

Sin embargo, para evitar que nuestro blog se convierta en una plataforma de anuncio de los robots emisores de spam, debemos zanjar el peligro con técnicas menos invasivas.

5.1.1.2 CAPTCHA

La primera de estas técnicas son los llamados *captcha*. Este vocablo es en realidad el acrónimo de *Completely Automated Public Turing test to tell Computers and Humans Apart* “prueba de Turing pública y automática para diferenciar máquinas y humanos”.

Es algo que se usa muy a menudo y que seguro que todos conocéis. A pesar de que su nombre formal no sea muy indicativo, explicado de una forma más sencilla podríamos definir los *captcha* como aquellas imágenes algo borrosas que contienen números y letras, y que aparecen en muchas web al registrarnos o dejar un comentario, en las que se nos pide que digamos cuál es el contenido del mismo. Imaginad una imagen con unos números que salen algo deformados y justo al lado un cuadro en el que debemos escribir estos números.

¿Por qué es tan eficaz este mecanismo contra el spam? Los *captcha* son imágenes que son relativamente fáciles de comprender para el humano, pero complejas de entender para el robot que genera spam. El robot, al intentar dejar el comentario, no podrá leer el contenido de la imagen y el humano sí. De esta forma podemos dejar nuestro sistema de comentarios totalmente abierto, de forma que no haya problemas de diálogo y tengamos total seguridad de que no va a pasar spam.



Figura 5.2. reCAPTCHA, uno de los captcha más famosos

¿Cuál es el problema de este mecanismo? El usuario quiere escribir lo más rápida y cómodamente posible, aunque sea algo bastante trivial, el hecho de tener que escribir el *captcha* puede llegar a incomodarlo y penaliza el diálogo en nuestro sitio. Podríamos decir que es mucho mejor que moderar todos los comentarios, pero no tan cómodo como el método que veremos a continuación.

Existe variedad de software que implementa esta funcionalidad. Uno de los más famosos es reCAPTCHA (<http://www.google.com/recaptcha>), además de ser

muy efectivo, es respetuoso con las personas con minusvalía visual, ya que les ofrece la posibilidad de leerles el texto. Hay que tener en cuenta que los demás *captcha* son muy inaccesibles para personas con deficiencias visuales y debemos tener una web lo más accesible posible.

¿Es posible instalar reCAPTCHA en WordPress e integrarlo para que actúe de protector con el spam en los comentarios? Sí, a través de un *plugin* llamado “WP-reCAPTCHA”. Veamos cómo ponerlo en marcha.

Lo primero será acceder al menú de administración de nuestro blog. Dentro de la barra izquierda, en la parte de “Plugins”, nos dirigimos a “Añadir nuevo” e instalamos el *plugin* de forma automática como hemos comentado en el capítulo anterior. Nada más instalarlo, nos aparecerá en la parte superior de la pantalla un mensaje avisándonos de que debemos adquirir una clave (una *key*) para activarlo.

Hacemos clic sobre el enlace que nos avisa y nos dirige a la página de opciones del complemento. En la primera línea, podemos acceder al sitio que nos provee de la clave de forma gratuita (el enlace indica *free reCAPTCHA keys*). Rellenamos nuestro nombre de dominio y pedimos que se nos cree la clave.

Con el par de claves que se nos proporcionan, pública y privada, rellenamos los campos que nos faltan en la configuración del complemento y luego pasamos a centrarnos en las opciones propias del mismo. Son muy básicas:

- Activar reCAPTCHA en los complementos (indicando si queremos que desaparezca a los usuarios registrados).
- Activar reCAPTCHA para el formulario de inscripción.
- Mensajes que aparecen si rellena mal la clave (es conveniente cambiarlos porque vienen por defecto en inglés).
- Opciones generales: Idioma (podemos elegir español) y si debe cumplir o no XHTML 1.0 (a priori no es necesario).

Tras estas opciones aparecen las opciones propias del servicio de *MailHide* (ocultación de correo). Para evitar el spam que afecta a los correos electrónicos, debemos tratar de publicar lo menos posible nuestra dirección de correo en sitios web. En ocasiones, es inevitable hacerlo y el servicio *MailHide* se ocupa de permitir que el correo se vea, pero lo ofusca de forma que los robots no puedan rastrearlo y almacenarlo.

Para poner en marcha este servicio (no es obligatorio hacerlo) en la página donde hemos obtenido las claves de reCAPTCHA, podemos obtener las claves de *MailHide*. En la parte izquierda debe aparecer la opción “Email protection” y dentro de allí nos dirigimos a “Get an API key”. Rellenamos los datos al igual que hemos hecho antes y ya tendremos el problema resuelto.

Para activar ambos servicios, una vez hemos terminado, tendremos que actualizar las opciones (al final de la pantalla) y ya podremos ver cómo aparece el *captcha* cuando vas a hacer un comentario.

Finalmente, deberíamos modificar la parte de moderación de las colas, ya que este complemento no es incompatible con las colas de comentarios y deberíamos permitir que los comentarios no tengan que ser aprobados para que aparezcan, ya que ya no existe el problema del spam.

A pesar de todo lo que hemos hablado hasta ahora, este método no es probablemente el mejor. Para el formulario de alta sí es muy útil (por eso mismo lo hemos instalado), pero para el tratamiento del spam en los comentarios la mejor opción es la siguiente.

5.1.1.3 AKISMET

Finalmente, llegamos a Akismet. Probablemente, el mejor sistema de protección anti-spam para blogs que existe. Es tal su utilidad, que viene instalado por defecto en cualquier instalación de WordPress.

The image shows the Akismet logo, which consists of the word "Akismet" in a stylized, lowercase, sans-serif font. The letters are dark grey or black, and the font has a slightly rounded, friendly appearance.

Figura 5.3. Logo de Akismet

Puesto que Matt Mullenweg, el creador de WordPress, conocía el problema del spam de primera mano, creó dentro de Automattic un grupo para desarrollar un *plugin* para WordPress que evitara el spam de forma transparente, tanto al usuario como al propietario del blog. Este complemento es Akismet.

Su modo de funcionamiento está basado en mecanismos bayesianos y otros mecanismos avanzados de reconocimiento de spam, al igual que el de los filtros de los correos electrónicos. Dado que para poder aplicar esos filtros es necesario tener recopilada una gran cantidad de información, el funcionamiento de Akismet está basado en una arquitectura cliente-servidor. Existen una serie de servidores en Automattic a los que el *plugin* envía los datos del comentario. Los servidores, tras

analizarlo, responden si es o no spam, de forma que el *plugin* de manera prácticamente instantánea aprueba o mete en la cola de spam los comentarios.

Posteriormente, el administrador del blog puede revisar de forma rápida dicha cola (y la de aprobados) por si acaso ha habido un falso positivo (comentario correcto que es considerado spam) o viceversa. Os puedo asegurar, desde mi experiencia personal y tras miles de comentarios, que la tasa de acierto es superior al 99%. El propio Akismet nos informa de lo que él llama el spam y el ham (comentarios que no son spam). Dentro de la web del sistema, hay gráficas donde podemos observar la magnitud del problema del spam.

Desde sus comienzos hasta finales del año 2010, había reconocido en total más de 20 millones de comentarios basura, mientras que los comentarios correctos (los *ham*) no llegaban a 4 millones. Esto quiere decir que podemos decir que de cada cinco comentarios, cuatro son spam y uno es correcto. Un gran problema hasta para el medio ambiente. Hay que tener en cuenta que, aunque sea una cantidad ínfima de energía, cada vez que se realiza un envío, recepción y tratamiento de un comentario, es necesario computarlo y por lo tanto hay un coste energético asociado.

Al igual que en el caso de reCAPTCHA, es necesaria una clave para evitar abusos del sistema (tened en cuenta que nosotros enviamos la información del comentario y el servidor de Akismet tiene que procesarla). La clave es gratuita para particulares sin ánimo de lucro y tiene un coste de 5 dólares para blogs con ánimo de lucro o empresas. El procedimiento para activarlo y conseguir la clave es muy similar al del complemento anterior, sólo tendréis que introducir la clave que os proporcionan y el *plugin* estará en marcha.

Akismet casi está preparado. Debes [introducir tu clave de API de Akismet](#) para que funcione.

Figura 5.4. Enlace con instrucciones para conseguir la clave de Akismet

Tras activarlo, os recomiendo desactivar el complemento anterior para los comentarios (es interesante mantenerlo para el formulario de registro) y dejar las opciones de moderación lo más laxas posibles. De esta forma, posibilitaremos una conversación activa y sin problemas en nuestro blog. El único cuidado que debemos tener es revisar, cada cierto tiempo, la cola de comentarios marcados como spam y aprobar los comentarios que estén ahí de forma incorrecta (falsos positivos) y borrar de forma definitiva (hay un botón al final de la pantalla) los que sí lo son, para que no ocupen espacio en nuestra base de datos.

5.2 SEO

SEO es el acrónimo del inglés *Search Engine Optimization* “optimización para los motores de búsqueda”. La mayor parte de los navegantes en la red utilizan los motores de búsqueda de forma muy frecuente. De hecho, en muchas ocasiones, en lugar de acceder a los sitios a través de su dirección, van primero al buscador, introducen el nombre del sitio en el mismo y acceden a través de los resultados. Es tal la importancia de conseguir una buena interacción con los motores de búsqueda que, si no la conseguimos, podemos estar seguros de que nuestro sitio, sea del tema que sea, ha fracasado.

Los motores de búsqueda modernos, en los que incluyo no sólo a Google, sino a otros menos populares, pero que tienen su cuota de mercado como Bing o Yahoo!, han automatizado y mejorado enormemente sus mecanismos de búsqueda e indexación, de forma que incluso sin hacer nada, si nuestra web es pública, podemos llegar a aparecer.

Sin embargo, esto no es suficiente. No sirve sólo aparecer, sino que hay que procurar aparecer mucho y bien. Bien significa aparecer en lo más alto de las búsquedas (y no sólo cuando buscan tu nombre). Un ejemplo claro sería el siguiente. En nuestro blog hacemos una serie de entradas dedicadas a un curso de un lenguaje de programación. Nos dirigimos a Google y buscamos “curso de Java” (imaginad que es Java el lenguaje que estamos tratando), seguro que no salimos ni en la primera ni en la segunda página, y muy posiblemente nuestro curso sea mejor que la mayoría que aparecen ahí. ¿Por qué?

Ningún buscador sabe si el contenido es bueno (al menos de primeras). Ellos utilizan unos parámetros cuantificables (en la informática al final hay que cuantificar en todo) como pueden ser: número de enlaces que apuntan a ese curso o cantidad de movimiento en tu blog. Ambos parámetros llegan con el tiempo si el contenido es bueno, pero todo este proceso es en parte un círculo vicioso, si no tenemos una buena posición en los buscadores no conseguiremos muchos enlaces.

¿Cómo romperlo? Además de ese tipo de parámetros, que demuestran de forma objetiva que el contenido es bueno, hay que procurar facilitarle la vida al máximo a los buscadores. WordPress es un ejemplo de cómo hacerlo y, tras instalarlo, ya nos provee de multitud de mecanismos que hacen que cualquier cosa que escribamos llegue de manera casi inmediata a los buscadores. Sin embargo, hay mucho más que hacer. Por un lado, hay que seguir una serie de buenas costumbres a la hora de escribir entradas, y por otro, tener herramientas automáticas (*plugins*) que nos faciliten el trabajo, además de aprovechar las capacidades que WordPress incorpora por defecto.



Figura 5.5. Logo de Yahoo!, el segundo buscador más popular tras Google

5.2.1 Optimizando WordPress

5.2.1.1 ENLACES PERMANENTES

El primer punto es optimizar la configuración por defecto de WordPress para que sea aún más amable con los buscadores. Puesto que versión tras versión va incorporando más características de este tipo, cada vez será más complejo rellenar este punto porque ya hará prácticamente todo por defecto.

Sin embargo, hay un asunto verdaderamente importante que deja a elección del lector, los enlaces permanentes, también conocidos por su nombre en inglés, *permalink*.

Cuando accedemos a un sitio web, la URL que aparece en la barra de direcciones es muy importante, tanto para el humano como para el motor de búsqueda. Probablemente, en más de una ocasión hayáis entrado en sitios donde las URL eran verdaderamente ilegibles. Lo más simple es dejar la dirección con la llamada a fichero que procesa la petición (por ejemplo, `index.php`) y llena de parámetros GET (esos que se separan por el símbolo *ampersand*, que en español se le llama *et*, &). Como podéis imaginar, una dirección de este tipo no dice nada al usuario y mucho menos al buscador.

Los enlaces permanentes buscan justamente eso, hacer que las direcciones sean limpias y amigables, de forma que en lugar de leer una serie de símbolos sin sentido para los humanos, leamos algo así como: “miblog.com/noticias/telefonica-compra-tuenti”.

WordPress trae incorporada esta opción, pero debemos configurarla. Dentro del menú de administración, tenemos en la sección de “Ajustes” un botón que nos lleva a “Enlaces permanentes”. Dentro de esta parte ya podemos pasar directamente a explicitar los ajustes.

Por defecto vienen una serie de opciones predefinidas, lo ideal es utilizar una que traiga el nombre, por ejemplo: “Día y nombre” o “Mes y nombre”. Nosotros mismos podemos construirnos cualquier otra combinación en el espacio

destinado para ello: “Estructura personalizada”. Sólo deberemos jugar con las variables:

- `%year%`: Año de publicación de la entrada.
- `%monthnum%`: Mes de publicación de la entrada en formato numérico.
- `%day%`: Día de publicación.
- `%postname%`: Título de la entrada adaptado a las normas de compatibilidad para URL (por ejemplo, si el título incluye una almohadilla o un símbolo de ese tipo, lo elimina de la URL o lo cambia por otro similar).
- `%post_id%`: Identificador de la entrada.

De esta forma podremos indicar en ese cuadro cualquier combinación que queramos, incluyendo partes que sean constantes o variables, por ejemplo: `“/hemeroteca/%year%/post_name%”`. Lo importante es que aparezca el nombre de la entrada, también es conveniente que aparezca el mes por lo menos, para evitar repeticiones.



Figura 5.6. Bing, el buscador de Microsoft

Los enlaces permanentes también son importantes para designar categorías y etiquetas. En la parte inferior del cuadro de configuración de los *permalinks*, nos encontramos bajo el título opcional la etiqueta base para designar categorías base y etiquetas base. Podemos elegir dejarlo tal cual, o poner algo más adecuado a nuestro blog, por ejemplo, en lugar de etiqueta, tema; y en lugar de categoría, sección.

Como nota técnica, hay que comentar que los equipos Linux con Apache 2 necesitarán activar la opción “AllowOverride” a “all”, presente en el archivo de configuración de Apache (el fichero `httpd.conf`, situado en el directorio “/etc”) para permitir que esta técnica sea posible. Tras modificar este valor, reiniciáis el

servicio del servidor web y ya estará en marcha. Para reiniciarlo deberéis hacer lo siguiente desde la terminal: `sudo /etc/init.d httpd restart` o, dependiendo de la distribución, deberéis hacer `sudo /etc/init.d apache2 restart`. El servicio se reiniciará e incluirá esta nueva opción para posibilitar el funcionamiento de los enlaces permanentes.

5.2.1.2 PERMITIR BUSCADORES

Aunque parezca obvio, debemos permitir de manera explícita que el blog sea visible para los buscadores. Esta comprobación suele realizarse en el inicio, y si la tenemos desactivada nos aparecerá en el título del panel de administración algo así como: “Buscadores bloqueados”.

Esta opción existe, porque antes de ponernos al público, podemos tener el blog abierto, pero en fase de pruebas, antes de pasarlo a lo que se dice técnicamente “producción”. Para desactivarla o activarla, nos dirigimos a la opción “Privacidad”, dentro de “Ajustes”, en el menú lateral del panel de administración.

5.2.1.3 NOTIFICACIONES AUTOMÁTICAS

Nada más escribimos una entrada, si ya hemos desactivado la opción de privacidad del punto anterior, tenemos la opción de agregar una lista de sitios a los que avisar cuando escribimos algo. ¿Hay que avisar de forma automática? No, WordPress trae por defecto un servicio de notificación automático, que busca en una lista que nosotros mismos introducimos, avisando uno a uno de la nueva entrada. Este tipo de servicios, aunque no nos reportan visitas de forma automática, pueden ser la fuente de actualización de otros sitios, por lo que es importante tener una lista lo más amplia posible.

Podemos ver la lista dentro de “Ajustes”, al final de “Escritura”. Por defecto, sólo viene una dirección: <http://rpc.pingomatic.com/>. Sin embargo, debemos rellenar muchas más, en las que debemos incluir los servicios de búsqueda de blogs de Google. Veamos los sitios de mayor relevancia, aparte de Ping-O-Matic:

- <http://api.my.yahoo.com/rss/ping>
- <http://bitacoras.net/ping>
- <http://ping.blogalaxia.com>
- <http://ping.feedburner.com>
- <http://pingoat.com/goat/RPC2>
- <http://pingqueue.com/rpc/>

- <http://rpc.technorati.com/rpc/ping/>
- <http://rpc.weblogs.com/RPC2>
- <http://blogsearch.google.es/ping/RPC2>

5.2.2 Buenas costumbres al escribir

Aparte de los métodos que acabamos de comentar, y de los que explicaremos en el próximo punto, que automatizan buena parte del trabajo, nosotros a la hora de escribir entradas también debemos ser cuidadosos y seguir una serie de normas para conseguir nuestro objetivo del buen posicionamiento.

- **Título:** El título de la entrada es una parte muy importante de la misma. Ha de ser claro, conciso y tampoco demasiado largo.
- **Etiquetas y categorías:** Ya que WordPress nos provee de un potente sistema para aportar palabras clave debemos aprovecharlo. Las etiquetas deben definir bien el contenido, y debemos tener una estructura de categorías lo más simple y cercana a la realidad posible.
- **Descripción de las imágenes:** El texto alternativo de las imágenes es necesario indicarlo. Primero, para facilitar la navegación a los invidentes, y segundo, para permitir que los buscadores tengan información de dicha imagen.
- **Uso de las letras en negrita:** Además de las etiquetas, para hacer énfasis en las palabras con más carga semántica del contenido, debemos hacer uso de esta herramienta.
- **Referencias internas:** Si tenemos contenido relacionado debemos enlazarlo. Los buscadores dan mucha importancia a los enlaces, por lo que debemos ser nosotros mismos los primeros en dotar de relación a nuestro contenido.

Aparte de estas técnicas, podemos señalar puntos que no son puramente tecnológicos y que muchos ya supondréis, pero no viene mal volverlo a mencionar.

- **Refresco continuo:** No se trata de escribir todos los días, pero la sensación de ver un blog que tiene una entrada cada mes es bastante pobre. El refresco es otro parámetro que tienen en cuenta los buscadores.

- **Contenido de calidad:** La calidad de un contenido es algo un tanto subjetivo, pero hay algunas reglas que debemos seguir, como, por ejemplo, nunca copiar y pegar contenido de un tercero (aunque luego lo refirmamos) o escribir entradas demasiado cortas (menos de 200 caracteres). Para esa labor ya existen las plataformas de *microblogging* como Twitter.
- **Implicación del lector:** Que el blog esté repleto de comentarios favorece las visitas, que el lector se sienta protagonista y que el blog se enriquezca con el contenido de éstos. En ocasiones, los comentarios son incluso más valiosos que el propio contenido. Los motores de búsqueda también rastrean los comentarios.

Antiguamente, se añadían algunas como que el sitio fuera antiguo. Hace años ayudaba el tiempo que estaba en línea la web, pero, hoy en día, no es una ventaja, y puede convertirse en una contra si no usamos herramientas actuales como WordPress. Otro aspecto cada vez menos importante es el nombre de dominio. Aunque sigue siendo esencial para tener una marca en Internet, las visitas procedentes de buscadores representan un porcentaje tan grande que los nombres fáciles de recordar, y relacionados directamente con la temática del sitio al que representan, ya no son un requisito.

5.2.3 Plugins relacionados

Para terminar este punto de mejora del SEO, comentaremos algunos complementos que nos ayudan en todo este proceso. El posicionamiento en buscadores es algo muy actual que está en plena eclosión. A partir de este momento, todos los sitios intentarán mejorar su posicionamiento, debemos cuidarlo para no quedarnos atrás.

5.2.3.1 ALL IN ONE SEO PACK

De todos los complementos dedicados a mejorar el SEO de un blog, éste es el más popular. ¿Qué hace exactamente? Hace algo realmente simple, pero que si no lo automatizara sería muy tedioso de hacer manualmente. Inserta el título de la entrada, las palabras clave y la descripción en la cabecera de la entrada o en la página principal del blog. Con esto conseguimos que el buscador lea mucha más información de cada *post* sin que el lector se moleste.

Para instalarlo, repetiremos el mismo proceso que siempre. Lo buscamos y lo instalamos directamente desde la interfaz. Es tal la fama del complemento, que probablemente os aparezcan muchos *plugins* con nombre similares (herramientas y

utilidades para el mismo), sed cuidadosos e instalad el original (se escribe tal cual está en el título). Nada más activarlo, nos pedirá que configuremos las opciones básicas del mismo a través de un cuadro de fondo rojo, al estilo del que muestra Akismet.

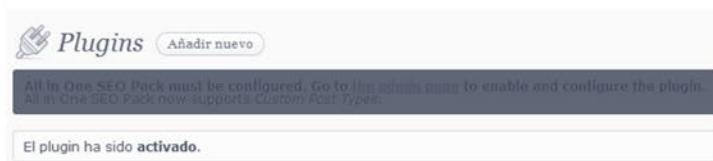


Figura 5.7. All In One SEO Pack recién activado

En la página de configuración del *plugin*, se nos presenta de primeras bastante publicidad. Este tipo de complementos tienen una fama tal, que sus autores recuperan parte del esfuerzo invertido con este tipo de *marketing*. En este caso, además, se sigue el modelo *freemium*. Esto quiere decir que hay una versión gratuita que incluye gran parte de los servicios y que engancha a la gente; y luego existe otra versión, en este caso llamada “Pro”, con un coste, que incluye algunas cosas más. Para un uso básico, no es necesario contratar esta versión de pago, con lo que trae el *plugin* gratuito tenemos suficiente para mejorar nuestros resultados.

Siempre que queramos acceder a esta página de configuración de nuevo, tras este primer acceso, tendremos un vínculo disponible en el panel lateral, en la sección “Ajustes”.

¿Qué opciones deberemos tocar en este administrador? Lo primero es activar el complemento (*plugin status* a *enabled*) y, posteriormente, rellenar las demás opciones. Es especialmente importante rellenar los *keywords* de nuestro sitio. Las palabras clave tienen un fuerte contenido semántico y ayudan enormemente al buscador a enmarcar el sitio en una temática concreta. Debemos separarlas por comas y deben ser concretas y suficientes para tratar todo lo que abarca el blog.

Cuando terminemos de rellenar los campos, al final hay un botón de actualización “Update options”, tras el cual los cambios habrán quedado reflejados.

Estos son los ajustes generales del complemento, pero cada vez que escribamos una entrada o una página, en la parte inferior de la misma, tendremos a nuestra disposición nuevos campos propios para cada entrada: título, descripción y palabras clave. Es muy interesante rellenarlos, aunque sea de forma breve, ya que,

por poco tiempo invertido, ayudamos enormemente a que los buscadores puedan indexarnos óptimamente.

5.2.3.2 GOOGLE XML SITEMAPS

Un *sitemap* es, como indica su nombre en inglés, un mapa del sitio donde se le indica a los motores de búsqueda qué información hay en la web, en este caso el blog. Pero, si los motores de búsqueda son capaces de leer la información al igual que lo hacemos las personas, ¿por qué es necesario un *sitemap* de este tipo?

Por inteligentes y desarrollados que estén los motores de búsqueda en la actualidad, el contenido semántico de los sitios (lo verdaderamente importante para ofrecer unas búsquedas óptimas) no es indexado de forma óptima, por la propia complejidad del lenguaje natural. A lo largo de este capítulo, estamos viendo diferentes métodos para ayudarlos en su labor y de esta forma conseguir un mejor posicionamiento. Una de las más importantes es el mapa del sitio.

Antes de continuar explicando el contenido del mapa del sitio, cómo hacérselo llegar a Google y cómo implementar todo esto en nuestro blog, veamos qué significa la otra parte del nombre del *plugin*, XML.

La tecnología XML, siglas en inglés de *eXtensible Markup Language* “lenguaje de marcado extensible”, es un formato estándar para transmisión de información a través de ficheros tipo texto. Éste será el lenguaje en que el complemento creará el mapa del sitio y estará correctamente adaptado a los buscadores.

¿Qué estructura aproximada tendrá nuestro mapa del sitio? Como todo fichero XML, tendrá forma de árbol, habrá una entrada en el fichero por cada *post*, página o elemento importante del blog. Tendrá lo siguientes campos:

- *Loc*: Dirección del elemento.
- *Lastmod*: Última modificación.
- *Changefreq*: Frecuencia de cambio (diariamente, mensualmente, semanalmente...).
- *Priority*: Un valor entre 0 y 1 que indicará muy poca prioridad o mucha prioridad, respectivamente.

Los valores típicos para los elementos en WordPress, con el complemento que vamos a instalar, son los siguientes:

- Raíz de nuestro blog:
 - *Changefreq*: Diaria.
 - *Priority*: 1 (máxima).
- Página del blog
 - *Changefreq*: Semanal.
 - *Priority*: 0,6 (media).
- Entrada de nuestro blog:
 - *Changefreq*: Mensual.
 - *Priority*: 0,2 (baja).

Con el complemento, esta información se actualizará de forma automática cada vez que creemos una entrada o página, de forma que el motor del buscador, al llegar a nuestro sitio, con sólo leer esta información que aquí os presentamos, ya tendrá datos suficientes para comenzar su indexación.

Para instalarlo seguiremos el proceso de siempre. Al igual que antes, deberemos tener cuidado con los complementos de nombre similar. Una vez instalado, lo activamos y pasamos a ver su configuración (hay un enlace de configuraciones directamente en el cuadro resumen de los *plugins*).

A pesar de la cantidad de opciones que aparecen, las que vienen por defecto son correctas. Podemos ampliar elementos, además de la raíz, las páginas y las entradas, podemos añadir categorías, etiquetas, páginas de autor..., sin embargo, tened en cuenta que cuanto mayor sea el mapa, mayor será el consumo de memoria y tiempo de CPU en su creación. La ganancia de marcar dichas entradas no es muy alta por lo que ésta, y otras opciones similares, sólo las marcaremos si estamos seguros de disponer de recursos suficientes.

```

1 <?xml version="1.0" encoding="UTF-8"?><?xml-stylesheet type="text/xsl" href="http://www.linuxhispano.net/generator/sitemap.xml"?><!-- generator="wordpress/3.0.1" -->
2 <!-- sitemap-generator-url="http://www.arsenbrachhold.de" sitemap-generator-version="3.2.4" -->
3 <!-- generated-on="24 November 2010 15:10 pm" -->
4 <urlset xmlns:xsl="http://www.w3.org/2001/XMLSchema-instance" xsl:schemaLocation="http://www.sitemaps.org/http://www.sitemaps.org/schemas/sitemap/0.9/sitemap.xsd" xmlns="http://www.sitemaps.org/schemas/sitemap/0
5
6         <loc>http://www.linuxhispano.net/</loc>
7         <lastmod>2010-11-24T15:09:41+00:00</lastmod>
8         <changefreq>daily</changefreq>
9         <priority>1.0</priority>
10
11     </url>
12
13     <url>
14         <loc>http://www.linuxhispano.net/2010/11/24/instalar-mod_rewrite-en-apache-los-pretty-per
15         <lastmod>2010-11-24T15:09:41+00:00</lastmod>
16         <changefreq>monthly</changefreq>
17         <priority>0.2</priority>
18
19     </url>
20
21     <url>
22         <loc>http://www.linuxhispano.net/2010/11/23/&c2&bfque-ea-la-neutralidad-de-la-red/</loc>
23         <lastmod>2010-11-23T10:51:57+00:00</lastmod>
24         <changefreq>monthly</changefreq>
25         <priority>0.2</priority>
26
27     </url>

```

Figura 5.8. Fragmento de sitemap con la sintaxis XML resaltada

Lo que sí es interesante que adaptéis a vuestras necesidades son las tasas de refresco de cada parte, así como la prioridad de las mismas. Ambas características podéis modificarlas al final de panel de configuración del *plugin*.

Cuando tengamos todo listo, guardamos los cambios y nos dirigimos al comienzo para indicar que queremos crear ya el mapa del sitio. El *plugin*, automáticamente, creará tanto el mapa del sitio como el mapa del sitio comprimido (para ahorrar ancho de banda de subida) y, finalmente, notifica a los diferentes buscadores.

Google, Bing y Ask.com no necesitan registro, por lo que lo hace automáticamente (podemos darnos de alta en las respectivas herramientas de *webmasters* para ver estadísticas, pero no es obligatorio), sin embargo, con Yahoo! sí es necesario, por lo que deberéis registraros y comunicarle al complemento el identificador que obtendréis.

Finalmente, hablamos de los problemas que pueden aparecer. Si os fijáis en la siguiente ilustración, donde vemos el diálogo que muestra el complemento al generar el fichero por primera vez, tenemos a nuestra disposición un enlace a funciones de depuración donde observar problemas. Lo más frecuente es la falta de permisos. Para solucionarlo, os recomiendo crear dos ficheros sin contenido y con los siguientes nombres: “sitemap.xml” y “sitemap.xml.gz” en el raíz de vuestro blog. Paso seguido le asignamos permiso de escritura para el grupo “otros” (un 646, por ejemplo) a ambos y ya tendremos el problema solucionado. Otra opción es cambiar el usuario de ambos ficheros al usuario del servidor web.

Resultados del último proceso de creación (fecha y hora de inicio: 24 November, 2010 20:38).

Su [sitemap](#) se creó por última vez el **24 November, 2010 20:38**.

Su [sitemap \(comprimido\)](#) se creó por última vez el **24 November, 2010 20:38**.

Google ha sido **correctamente notificado** sobre los cambios.

Bing ha sido **correctamente notificado** sobre los cambios.

Ask.com ha sido **correctamente notificado** sobre los cambios.

El proceso de creación tardó aproximadamente **3.68 segundos** en completarse y utilizó 21.25 MB de memoria.

Si ha cambiado algo en su servidor o blog debería [crear el sitemap](#) manualmente.

Si encuentra algún problema durante el proceso de creación, puede utilizar la [función de depuración](#) para obtener más información.

Figura 5.9. Mensaje tras la creación del mapa del sitio

5.3 INTEGRACIÓN EN REDES SOCIALES

Hoy en día, la sociedad al completo conoce el poder de redes sociales como Facebook o Tuenti, o de servicios de *microblogging* como Twitter. Pero, ¿qué provecho podemos sacarle a este tipo de servicios con nuestro blog? Si en el punto anterior tratábamos la mejora del posicionamiento en buscadores para conseguir el mayor número de visitas posible de Google y compañía, ahora buscamos lo mismo, pero aplicado a estos sitios.

El hecho de que alguien nos haga referencia en Twitter, o comente sobre nosotros en un tablón de Facebook, puede llegar a ser un escaparate mayor que aparecer en la primera página de resultados de una búsqueda.

Además de este tipo de servicios, este punto también tiene mucha conexión con los agregadores de noticias, estilo Menéame en español o Digg en inglés. ¿En qué consisten estos agregadores de noticias? Aunque cada uno tiene sus detalles, a grandes rasgos hablamos de plataformas en las que los usuarios suben enlaces de noticias, que posteriormente son valorados por la comunidad y, en caso de que gusten a la mayoría, salen a portada. ¿Qué supone salir a portada? Por experiencia propia puedo decir que supone una inyección cercana a las 10.000 visitas directas (en el caso de Menéame). Estas visitas suelen rebotar al momento (en el punto próximo hablaremos de qué es esto de rebotar), pero siempre habrá un porcentaje que nos agregue a su RSS o a su lista de favoritos.



Figura 5.10. Logo de Menéame

5.3.1 Medidas generales

Si queremos aparecer en Twitter o Facebook, la primera opción será crear una cuenta para nuestro blog. Cuesta algo de tiempo y los resultados varían, pero a la larga es más fácil meter la cabeza si ya estamos dentro y nos siguen. En el caso de portales como Menéame, donde se sanciona anunciarse a uno mismo, no deberemos enviar nuestras propias noticias, sino que debemos ayudar a que nos puedan “menear”, de forma que nuestra noticia salga al portal.

Otro punto importante es permitir los *pingback* y *trackback*, es decir, las notificaciones vía comentarios de enlaces de otros sitios. Podéis ver el estado de esta opción (aparece activada por defecto) en el panel de administración, dentro de “Ajustes” en la opción “Debates”.

5.3.2 Plugins relacionados

5.3.2.1 IC BESOCIAL

La forma más simple de aumentar la interacción es ofreciendo un botón al usuario con el que pueda enviar directamente el contenido que le gusta de nuestro blog a la red social en cuestión. Si además le añadimos que el *plugin* informa al usuario del grado de popularidad de dicha entrada en la red en cuestión, mucho mejor, ya que incitará más aún a participar.

Esto que acabo de comentar es justo lo que hace ic BeSocial. Su instalación es posible mediante la misma opción de siempre a través del repositorio y su configuración es básica. Debemos elegir dónde aparecen dentro de la entrada los

botones de las redes sociales (debajo, encima o centrado) y con qué plataformas. Tenemos disponibles:

- Menéame.
- Bitácoras: Red social de *bloggers* hispanicos.
- Reddit: Sitio web de marcadores sociales.
- Delicious: Conocido hasta hace poco por del.icio.us es un servicio similar al anterior.
- Buzz: Red social de Google.
- Facebook.
- Twitter.

Para alguno de ellos deberemos conseguir una clave (o *key* como se dice en inglés) para hacerlos funcionar, pero la mayoría son directos.



Figura 5.11. Aspecto gráfico de los botones de ic BeSocial

5.3.2.2 BUDDY PRESS

Por ahora hemos visto complementos que aportan funcionalidades o mejoran el blog en sí, pero Buddy Press es algo más que eso, llega a cambiar, aunque sea parcialmente, la concepción del sistema orientándolo más hacia una red social.

Podríamos decir que incrementa enormemente las posibilidades de los usuarios de nuestro sitio, de forma que sin relacionarse directamente con redes sociales de fuera, como hemos visto hasta el momento, se crea una minired social dentro de nuestro blog. De hecho, las funcionalidades que tiene son bastante embrionarias, el proyecto es mucho más ambicioso pero aún se encuentra en una versión muy temprana.

¿Qué posibilidades ofrece? Primero, permite enriquecer más los perfiles de los usuarios, de esta forma se añaden más campos y se consigue conocer más de los usuarios. Relacionado con esto, los usuarios pueden crear grupos por afinidad y dentro de los grupos mantener discusiones y crear hilos de conversación. Los grupos también soportan información propia, entre otras cosas, un logo. Si los usuarios quieren escribirse directamente, sin necesidad de publicar comentarios, pueden hacerlo a través de mensajes internos entre ellos.

También hay otras opciones que personalizan la web al lector, por ejemplo, cada usuario puede elegir con qué tema gráfico quiere visualizar la información. Finalmente, a través de WordPress MU (un proyecto independiente que en la versión 3.0 se ha incluido dentro de la rama principal), los usuarios pueden crear sus propios blogs dentro de nuestra plataforma.

Como veis, la potencialidad del sistema crece enormemente, aunque a la vez también crece la carga que tiene que soportar la máquina sobre la que ejecuta y la cantidad de problemas posibles. Nunca he puesto en producción un entorno de este tipo, porque en los campos que he trabajado, nunca me he visto en la necesidad de ofrecer este tipo de alternativas a mis lectores. Sin embargo, es una opción más a tener en cuenta y a partir de este punto vamos a ver cómo se instala y configura.

Es tal la cantidad de complementos asociados a BuddyPress que si lo buscáis en el instalador del panel de administración, aparecerán tantos resultados que será mejor que procedáis a instalarlo manualmente. Os recuerdo el proceso:

1. Descarga de la web del *plugin*:
(<http://wordpress.org/extend/plugins/buddypress/>).
2. Descarga, descompresión y subida (o copia) al directorio “/wp-content/plugins”.
3. Finalmente, activación desde el panel de administración.

Tras terminar esta secuencia, el *plugin* estará instalado y en marcha. Sólo hay una opción que necesita para funcionar: deben estar activados los enlaces permanentes personalizados. En este libro ya hemos explicado cómo hacerlo y deberíais tenerlos activados llegados a este punto. Dada la variedad y cantidad de funciones que ofrece BuddyPress, nos aparecerá una sección nueva en la barra lateral, en la posición siguiente al “Escritorio”.

Es tal la cantidad de opciones que aporta BuddyPress, unido a la cantidad de complementos que han nacido alrededor del mismo, que podríamos dedicarle un capítulo entero, pero como éste no es el objetivo de este texto y probablemente en un porcentaje muy alto de casos BuddyPress no interese a los fines que se persiguen, explicaremos lo más básico.

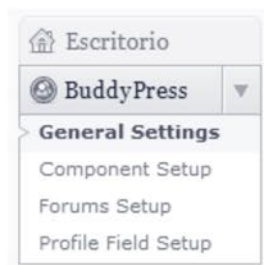


Figura 5.12. Opciones básicas de BuddyPress

La configuración de BuddyPress se centra en los cuatro puntos que veis en la ilustración:

- Opciones generales.
- Configuración de componentes: Activamos o desactivamos los distintos componentes que forman BuddyPress. Son los siguientes: flujo de actividad, foros, amigos, grupos, mensajería privada y perfiles extendidos.
- Instalador de un foro: BuddyPress se relaciona directamente con el CMS para crear foros phpBB.
- Configuración de los perfiles: Añadimos, modificamos y eliminamos los campos que pueden rellenar los usuarios de sus perfiles, a la vez que concretamos si son de relleno obligatorio, el tipo de campo que es y la descripción de cada uno.

Finalmente, si queréis disponer de todas las funcionalidades de BuddyPress, debemos tener un tema de WordPress que lo permita o modificar el nuestro para que lo soporte. En la parte superior de la pantalla lo comprobaremos, en caso de no ser compatible, el sistema nos lo avisará con el clásico mensaje de aviso de fondo amarillo o rojo en la parte superior del administrador.



Figura 5.13. Ejemplo de grupo en BuddyPress

5.4 ESTADÍSTICAS Y SEGUIMIENTO

Cuando comenzamos un blog, lo más importante siempre, sea del tipo que sea, es poder tener una difusión muy alta. Hemos estado estudiando métodos para conseguir una mejor difusión, pero si no conocemos el número de visitantes, su procedencia o a través de qué sitio entran, nunca llegaremos a estar seguros de la difusión que hemos alcanzado y, por lo tanto, no nos motivaremos.

Mi experiencia en el mundo del *blogging* me dice que más aún que conseguir sumas de dinero con los blogs, lo más importante para motivarte a seguir trabajando es tener muchas visitas y muchas referencias.

Existen métodos propios para instalar en WordPress que hacen todo este seguimiento, pero sin lugar a dudas, nunca llegan al nivel del líder de esta materia en la red: Google Analytics. Como no podía ser de otra forma, Google, que podríamos afirmar que es la compañía que mejor conoce Internet, nos proporciona una herramienta que además de ser la más potente, gráfica e interactiva, es gratuita.

WordPress se adapta a la misma, proporcionándonos complementos para que nuestro sitio pueda interactuar con dicha plataforma. Sin lugar a dudas, la mezcla perfecta. Además de estadísticas de visitas, también nos interesará saber qué cantidad de personas nos siguen vía RSS, pero como este apartado es más propio de la sindicación, lo vamos a dejar para el punto siguiente, que se dedicará en exclusiva a la sindicación.

Antes de continuar, el hecho de tener activados los *trackback* y los *pingback* en los comentarios, como podemos ver en las opciones de “Debate”, nos

dará una idea de si una entrada nuestra ha sido referenciada en otro sitio. El problema de este sistema es que el otro sitio tiene que tener también esta tecnología activada para que nos quede constancia. Cada día son más los sitios que la tienen activada, pero no deja de ser una información que nunca estará completa.

5.4.1 Google Analytics

Como ya hemos dicho, el líder de las estadísticas para sitios en red es un producto de Google. Su liderazgo proviene de su facilidad de uso, de su fiabilidad y de la cantidad de informes que proporciona a sus usuarios. El producto nació gracias a la compra de Urchin por parte de Google, hasta aquel entonces empresa líder en estas cuestiones.

¿Cómo funciona Google Analytics? El mecanismo de la plataforma es el siguiente. El cliente, nosotros, introducimos directamente, o a través de un *plugin* (como es nuestro caso), una referencia a un *script* JavaScript alojado en los servidores de Google, a la vez que también enviamos un parámetro (nuestro identificador). Ese fragmento de código es el llamado GATC (*Google Analytics Tracking Code* “código de seguimiento de Google Analytics”). Este código se encarga de descargar más código de dichos servidores, que será el encargado de llevar a cabo la monitorización y envío de datos al servicio.

Todo este proceso se realiza de forma transparente tanto al administrador del sitio como al visitante. Pasado un tiempo (la búsqueda de datos no es instantánea porque Google la procesa), tenemos acceso a toda la información de nuestras visitas a través de una interfaz web muy gráfica e intuitiva desarrollada con tecnología Adobe Flash, dentro de Google Analytics.

La interfaz de Analytics es altamente personalizable y nos permite personalizar criterios de seguimiento. Entre los datos más importantes que se nos presentan podemos destacar: el seguimiento de usuarios exclusivos, el rendimiento de un determinado segmento de usuarios (muy importante para el comercio en línea), las fuentes de entrada desde los motores de búsqueda, el rendimiento del contenido y el análisis de navegación de cada usuario (qué páginas visitó, cuánto tiempo estuvo en ellas...). También podemos marcar objetivos y, desde hace poco, una de las últimas mejoras que se incluyeron fue la integración dentro de la plataforma de Google AdSense (de la que hablaremos en el punto de monetización).

5.5 SINDICACIÓN

La sindicación es un conjunto de tecnologías que permiten a los lectores tener constancia de las nuevas publicaciones de los sitios a los que se agregan. Como todo gestor de contenidos para blogs, WordPress nos ofrece de forma automática la sindicación integrada. La tecnología RSS, de la que hemos hablado en varias ocasiones a lo largo del libro, es una tecnología de sindicación. Con el canal de RSS que nos ofrece WordPress, tenemos suficiente para dar un servicio de este tipo, pero si queremos poder tener un seguimiento (¿cuántos lectores nos leen a través de este canal?) y una mejor difusión debemos usar servicios externos.

5.5.1 FeedBurner

Sin lugar a dudas, el líder en este segmento del mercado web es FeedBurner (<http://feedburner.google.com/>). La compañía nació en 2004 de la mano de Dick Costolo, Eric Lunt, Steve Olechowski y Matt Shobe. Al igual que ha ocurrido en muchas ocasiones en la historia de la red (acabamos de ver el caso de Urchin), Google decide comprarla el 3 de junio de 2007. Un año y unos meses más tarde, FeedBurner se encuentra totalmente integrada en los servicios Google.

Este último punto significa que con la misma cuenta Google de Gmail podéis manejar FeedBurner sin problemas, además de, por supuesto, Google Analytics o la plataforma de publicidad en línea de la que hablaremos a continuación: Google AdSense.

Lo primero que debemos hacer es crearnos una cuenta en la plataforma, una vez la tengamos creada, nos creamos un canal RSS en la misma (tendrá la forma <http://feedburner.com/micanal>) en el que indicaremos la dirección de nuestro blog. El siguiente paso será integrarlo en nuestra bitácora.

5.5.2 FD FeedBurner Plugin

Instalamos el *plugin* por la vía de siempre, lo activamos y lo configuramos en sólo un paso. Aparecerá una opción nueva en la sección “Plugins” de la barra lateral. Sólo tendremos que incluir la dirección del canal que acabamos de crear a “Redirects my feeds here”. Si queremos crear un canal de comentarios, repetimos la operación, creamos un canal en FeedBurner y posteriormente desde el *plugin* redirigimos los comentarios (el segundo cuadro de texto).

Finalmente, para darle difusión al canal, podemos descargar de la web un logo de RSS y crear un *widget* en un lateral con una imagen que nos redirija al canal. De esta forma, todo el que quiera agregarnos sólo deberá hacer clic sobre la imagen.

Creamos el *widget* de tipo texto, lo asignamos a la zona donde queremos que aparezca y le introducimos el siguiente código en lenguaje HTML: `<a href="direccion_canal_feedburner"></a>`. Lo guardamos y ya tendremos en la pantalla los pertinentes canales de sindicación a sólo un clic de distancia para nuestros visitantes.



Figura 5.15. Widget con imágenes y enlaces a varios canales RSS

5.6 MONETARIZACIÓN

En el punto anterior hemos dicho que lo más motivante para seguir trabajando para un blog es poder comprobar cómo aumenta el número de visitas, o el número de suscriptores al canal RSS, que están íntimamente ligados. Sin embargo, no es lo único. Cuando llegamos a tener un número de visitantes considerable, podemos empezar a sacarle un pequeño rendimiento económico al blog. Podemos tomárnoslo como una actividad económica más (hay empresas que viven del *blogging*) o como un ingreso extra que nos ayude a pagar lo que nos cuesta el dominio y el alojamiento web que tenemos contratado.

A mayor difusión, mayores ingresos, aunque no sólo cuenta el número de visitantes. La temática del blog es verdaderamente importante en este sentido. Podemos tener un blog con 10.000 visitas diarias, que gane menos que otro con 1.000 visitas en el mismo período de tiempo (suponiendo que usamos la misma plataforma de publicidad y hay el mismo número de anuncios por página).

¿Por qué esta diferencia? Al igual que en todos los mercados, el precio de los anuncios en la web se ajusta a la ley de la oferta y la demanda. Por un lado, si el tema que escogemos es muy dado al mundo del anuncio, tendremos mucha

demanda. Si hablamos de política o tenemos un blog personal probablemente el precio de los mismos será menor, porque no hay un interés económico de cara a vender en la red un producto. Si hablamos de un blog de viajes, de hoteles o de ropa, las cosas cambian. El producto se vende, hay mucha gente que quiere vender y, probablemente, encontremos mejores precios para nuestros anuncios.

Por el otro lado tenemos la oferta. El caso claro es la tecnología, es un tema ideal para la demanda, ya que es muy fácil vender productos tecnológicos en la red, pero a su vez ocurre otra cosa: hay muchísimos blogs de tecnología. Por lo tanto, los anunciantes, que son muchos, se pueden anunciar en muchos blogs y el precio por anuncio baja. Como veis, la temática del blog influye enormemente en la ganancia.

Como podéis ver, los anuncios son un verdadero mundo y, si no dispusiéramos de plataformas como la que os voy a presentar, sería mucho más complicado sacarle partido. Tened en cuenta que deberíamos buscar anunciantes, ofrecerles datos fiables de nuestra difusión y de los clics que han tenido sus anuncios, gestionar los cobros con cada uno..., es decir, una gran cantidad de trabajo de gestión y promoción que no podríamos asumir, a no ser que llegáramos a ser un “grande”.

Si no queremos quebrarnos tanto la cabeza y disponer de los anuncios adecuados en cada momento, aunque perdiendo margen de las ganancias, podemos contratar un intermediario. El intermediario más grande de toda la red y la mayor fuente de ingresos de su compañía matriz es Google AdSense. Como veis, llevamos ya unos pocos apartados de este capítulo hablando de la compañía de Mountain View, pero sus productos hacen que merezca la pena hacerlo.

5.6.1 Google AdSense

Al igual que en el caso de Analytics y Urchin, y de FeedBurner, AdSense nació dentro de una empresa externa a Google en 1998. La empresa se llamaba Oingo y fue fundada por Gilad Elbaz y Adam Weissman en Los Angeles, California. La idea de negocio era crear un algoritmo que fuera capaz de comprender parcialmente la semántica de una página, para aplicar la información de la misma a dar servicios o vender productos relacionados. La empresa tuvo éxito y en 2001 decidieron cambiar el nombre de la misma para tener una marca más relacionada con su actividad. Pasó a llamarse compañía de semántica aplicada, Applied Semantics.

En 2003 Google la compró por 102 millones de dólares y aplicó toda su tecnología a uno de sus productos estrella: el *marketing* en línea a través de la

plataforma de AdSense. Es tal su peso dentro del total de la compañía que aproximadamente un tercio de los ingresos de ésta provienen de esta fuente.

¿En qué se distingue AdSense del resto? La base sobre la que se asienta la plataforma es ofrecer al lector anuncios que le puedan resultar interesantes. Para ello, se basa en la tecnología semántica de la que hemos hablado, para relacionar anuncios y contenidos. Este mecanismo interesa tanto al usuario, que ve una publicidad, que aunque no le interese, está relacionada con lo que está leyendo, como al administrador del sitio, ya que de esta manera es más fácil que con las mismas visitas los visitantes se interesen más y por lo tanto gane más dinero. Esta gran ventaja tiene un problema, los márgenes de Google probablemente sean mucho mayores que los de su competencia.

Otra gran ventaja del sistema es que AdSense se encarga de contratar la publicidad por un lado y, posteriormente, asigna campañas a sitios con el mecanismo anterior. Podremos ver en nuestro sitio cómo, incluso en el mismo día, hay decenas de diferentes anuncios. Nos despreocupamos totalmente de la contratación, muestra y verificación de los mismos.



Figura 5.16. Logo de Google AdSense

¿Qué criterio se sigue para pagar al administrador del sitio? Básicamente hay dos, el más popular es el criterio del ingreso por clic. Cuando un usuario hace clic en un anuncio y es redirigido al sitio del anunciante, el *blogger* recibe una cantidad. Esta cantidad varía enormemente, de 0,01 € a 2,00 € por clic. Si queréis saber un valor medio para un sitio tecnológico podría deciros que se encuentra alrededor de los 0,15 €. La otra forma de cobrar no es por clic, sino por impresiones, es mucho menos normal y difícilmente llegaremos a tenerla en un sitio pequeño o mediano.

¿Qué podemos mostrar en los anuncios? Hay unos formatos predefinidos que a través de una interfaz web muy intuitiva se generan. Desde pequeños anuncios a paneles que ocupan barras al completo. Nosotros elegimos tipo, colores y con el código generado lo incluimos en nuestro sitio. Veremos cómo hacerlo más adelante, cuando veamos cómo crear y modificar temas directamente en el código, mientras tanto, nos valdremos de complementos que facilitan la labor pero no dan tanta libertad.

Además de mostrar anuncios en nuestra web, tenemos otros productos que vender. Desde anuncios en el RSS, muy fácilmente configurable si usamos FeedBurner, a anuncios para aplicaciones móviles dentro de, por ejemplo, la plataforma Android. Tendremos acceso a informes muy detallados y, gracias a que este servicio también es de Google, tendremos acceso a los datos de Analytics.

Lo primero para comenzar a hacer uso del servicio es darnos de alta en el sitio web oficial del mismo: <http://www.google.com/adsense>. El proceso de alta es muy similar al de otros servicios de Google, por lo que no tendréis problema. Al final del mismo, obtendréis un código de cliente de AdSense que será el que utilicemos en el siguiente paso.

5.6.2 Quick AdSense

Dentro de la plataforma podemos crear diferentes anuncios y, a su vez, debemos saber dónde colocar cada uno de ellos. Veremos en un capítulo posterior cómo modificar el código del tema de forma que podamos hacerlo sobre el código, sin embargo, por ahora vamos a instalar y a ver cómo funciona un *plugin* que se encarga de todo esto.

Su nombre es Quick AdSense y podéis encontrarlo en el repositorio oficial de complementos, por lo que la instalación es directa. Existen otros muchos complementos para manejar AdSense, pero éste es el más popular. Nada más activarlo nos dirigimos a “Ajustes” en el panel de administración y allí tendremos el acceso a las opciones del *plugin*.



Figura 5.17. Espacio destinado al código del anuncio

Dentro de la plataforma de anuncios de Google, habréis visto que, al generar los anuncios, se genera el correspondiente código JavaScript. En este panel de configuración del complemento, disponemos de una serie de cuadros de texto destinados justo a esto: encuadrar en cada uno de ellos tantos anuncios como generemos. Por defecto tenemos 10 espacios para anuncios dentro del texto y otros 10 espacios para anuncios en la barra lateral, a través de un *widget* que el complemento se encarga de generar.

Lo primero que debemos hacer es copiar y pegar el código de los anuncios en las áreas de texto. Posteriormente elegiremos qué anuncio mostrar. Podemos obligar a que se muestren por defecto en ciertas áreas o, por el contrario, podemos elegir en cada caso qué anuncio mostrar. Veamos cómo hacerlo para ambos métodos.

En el primer caso, tenéis dentro de la configuración del *plugin* una parte del panel de nombre *Position (Default)*, es decir, posición por defecto, en donde asignaremos un espacio a un anuncio en particular o un anuncio aleatorio a una posición dentro del texto:

- *Begginig of post*: Comienzo de la entrada.
- *Middle of post*: Mitad de la entrada.
- *End of post*: Final de la entrada.
- *After paragraph... to end of post*: Desde el párrafo... hasta el final de la entrada.

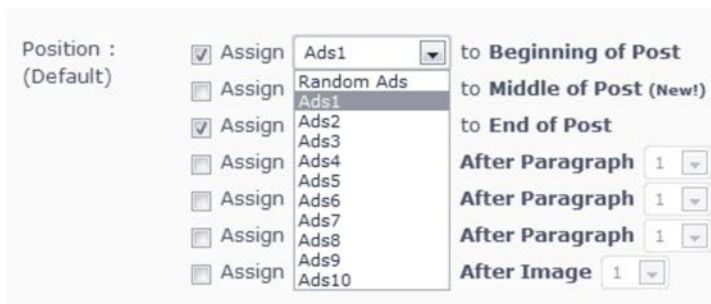


Figura 5.18. Asignando anuncios a espacios

Como podéis ver en la ilustración, la interfaz para asignar anuncios a posiciones es muy intuitiva. Igualmente intuitivo es el método para asignar anuncios de forma explícita al escribir entradas. Lo primero que debemos hacer es deshabilitar todos los anuncios por defecto. Una vez estén todos deshabilitados, nos dirigimos al editor WYSIWYG y en la parte superior elegimos HTML. Veremos unos nuevos botones de nombre: Ads1, Ads2, Ads3... Cuando hagamos clic sobre ellos se insertará el anuncio en el espacio en el que estemos escribiendo en dicho

momento. Para volver al editor visual, hacemos clic otra vez sobre la pestaña visual y seguiremos escribiendo como siempre.

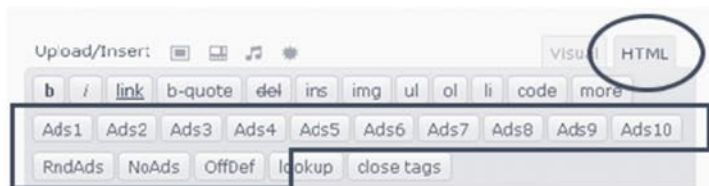


Figura 5.19. Botones dentro del editor HTML para introducir anuncios

5.7 COPIAS DE SEGURIDAD

Tenemos ya listo el blog, comenzamos a introducirle contenido y nuestras noticias tienen éxito. Llegamos a tener miles de visitantes al día, cientos de euros de ingresos por el programa AdSense y, de repente, un día sufrimos una catástrofe. Nuestro servicio de alojamiento web nos informa de que el equipo en el que se alojaba nuestro blog ha salido ardiendo o, si lo alojábamos en un servidor propio, el disco duro se rompe, sin posibilidad de recuperación. Nos hemos puesto en un caso muy negativo pero no por ello imposible.

Ante este tipo de desgracias, podemos articular mecanismos preventivos de todo tipo, desde duplicar la fuente de alimentación de nuestro servidor casero a tener un servidor virtualizado en un *cluster* que, en caso de que se caiga una hoja, podamos levantar otra. Sin embargo, ningún método es infalible, ni los más avanzados, y debemos tener un mecanismo de respaldo, de forma que llegado el momento en menos de una hora podamos volver a tener todo en marcha.

Lo ideal es contratar un alojamiento web que ofrezca este servicio de copia de seguridad. Sin lugar a dudas, es lo más cómodo, es muy efectivo y tiene la ventaja de que normalmente la recuperación se efectúa en sólo unos instantes tras hacer clic en un botón del panel de administración del alojamiento.

Si no disponemos de esta posibilidad, existen mecanismos alternativos, que son los que vamos a explicar. Nos centraremos sólo en la recuperación que podríamos decir manual, porque la experiencia me dice que es la más efectiva y no dependemos de la interacción de complementos o programas dedicados expresamente a esta labor.

Por fortuna, en estas mismas líneas ya hemos explicado cómo hacer este proceso al hablar de la migración del blog de un servidor propio a uno en la nube. La idea es justo la contraria, descargarnos los datos, tanto ficheros como MySQL, a nuestro equipo de forma periódica. Puesto que se trata de una descarga y no de una subida, el tiempo de descarga será relativamente breve y, hasta que el blog no empiece a pesar muchos megabytes, en menos de un minuto tendremos la copia hecha. Hablo de copias completas, podríamos complicarnos con programas de copia incremental o que usen algoritmos avanzados, como rsync, pero como hablamos de, en el peor caso, cientos de megas, las líneas actuales de banda ancha nos darán suficiente rapidez como para no tener que complicarnos.

En el momento en que suframos un problema, sólo tendremos que subir tanto el fichero SQL con los datos del gestor como los ficheros que le dan vida y todo volverá a estar en marcha. De hecho, si queréis tardar aún menos, de los ficheros, sólo tendréis que descargar los que cuelgan de la carpeta “wp-content”. Tened en cuenta que el resto es propio de WordPress (podéis descargarlo de nuevo) y ahí es donde están el tema gráfico, los complementos y los ficheros que habéis ido subiendo.

5.8 DISPOSITIVOS MÓVILES

Conforme pasa el tiempo, los llamados móviles inteligentes ganan cuota de mercado, cada día son más las líneas móviles que nos permiten tener Internet a través de una conexión de datos desde cualquier lugar en el que haya cobertura. El precio es cada vez más asumible y las prestaciones mejores, por lo tanto, esta tendencia no parará a corto plazo. De hecho, hay estudios que le dan más relevancia al dispositivo móvil en un futuro que al ordenador de escritorio, que nos ha acompañado en las últimas décadas.

Está claro que para ciertas labores, como todo lo relacionado con trabajo de desarrollo u oficina, el ordenador de escritorio seguirá siendo el referente, pero los móviles y tabletas tendrán, y tienen ya de hecho, un lugar privilegiado en lo relacionado con el ocio. Puesto que los blogs se encuadran en este sector, debemos cuidar nuestra integración en los dispositivos móviles para llegar al máximo número posible de lectores.

El método para conseguirlo es adaptar nuestro sistema para que muestre dos temas gráficos diferentes: uno para los visitantes que usan equipo de escritorio y otro para usuarios de dispositivo móvil. Como supondréis, el tema para el móvil debe tener una anchura más reducida, menor presencia de imágenes y complementos pesados y darle más relevancia al texto. Hacer esto a mano sería además de complicado bastante laborioso, pero por fortuna, como tantas veces en

WordPress, disponemos de los complementos ideales para llevar a cabo esta labor de forma automática.

5.8.1 WPtouch

Al igual que en otros temas, existe variedad de complementos, y me gustaría presentaros todos, pero, al fin y al cabo, debemos elegir, así que me quedo con WPtouch, que es el más popular complemento para la labor que buscamos.

El *plugin* nació para adaptar la estética de los blogs WordPress a la típica visualización de los programas en el iPhone. De hecho, el tema que aplica por defecto el complemento para visualizar la web en el móvil está basado estéticamente en la librería gráfica de las aplicaciones de iPhone.

Cuando alguien entra desde un dispositivo móvil: iPhone o iPod Touch, Google Android, Blackberry, Palm Pre o cualquier otro dispositivo móvil moderno verá, en lugar de nuestro tema por defecto, un tema gráfico especialmente adaptado a estos dispositivos que será totalmente independiente del tema que tengamos puesto.

Respecto a los *plugins*, hay algunos soportados y otros que no lo están. Una vez instaléis el complemento, en el menú de configuración del mismo, podréis comprobar al final de la pantalla el listado de complementos compatibles y no compatibles. Tened en cuenta que algunos complementos dejan de tener sentido al visualizarse en un móvil, algunos por pura incompatibilidad técnica y otros porque prácticamente no se visualizarían correctamente en una pantalla de esas dimensiones.

Además de estas comprobaciones de compatibilidad podremos modificar:

- Ajustes generales del blog: Aplicados expresamente a la versión móvil.
- Opciones avanzadas: Activar comentarios, mostrar los *gravatars* (las imágenes identificativas de cada usuario) o mostrar en la cabecera el cuadro de búsqueda y las categorías.
- Opciones de estilo y color: Para variar los aspectos básicos de la interfaz web para móvil.
- Catálogo de iconos para el complemento.
- Elementos del menú.
- Datos del AdSense para que también aparezca en esta interfaz.

La configuración por defecto es bastante correcta para casi todos los casos. En caso de que queramos optimizar más, de cara al tiempo de carga, os recomendaría no mostrar los *gravatar* (por el consumo de ancho de banda). El resto me parece correcto, aunque si tenéis un dispositivo móvil podéis jugar con las opciones hasta dejarlo a vuestro gusto.

Como veis, con sólo instalar el complemento le estáis dando vida a un pequeño WordPress para móviles dentro del mismo sistema, adaptado totalmente a sistemas móviles y sin tocar nada de nuestra configuración previa.



Figura 5.20. Aspecto de un blog WordPress utilizando WPTouch en un iPhone

5.9 MEJORA DEL DIÁLOGO

El hecho de que la gente nos encuentre fácilmente en Google o tener presencia en redes sociales y agregadores de noticias, no nos asegura que el visitante que haya entrado una vez en el sitio lo haga en más ocasiones. La persona que entra en un blog, si comenta y ve como su comentario es contestado, se sentirá atraído de nuevo por el mismo blog porque de esta forma empezará una especie de “conversación”.

Por defecto, en WordPress pueden establecerse conversaciones, a través del sistema de comentarios tan típico de todos los blogs. Sin embargo, hay que facilitar este punto. ¿Cómo? Informando de que ha habido comentarios nuevos, ya sea mostrando los últimos comentarios o enviando por correo las contestaciones a los autores de los comentarios originales.

BuddyPress crea un sistema mucho más complejo de interacción con el usuario, una pequeña red social, como ya hemos comentado. Pero en este punto no nos referimos a eso, nos referimos a usando los métodos estándares de *blogging* facilitar la interacción de los usuarios con nosotros mismos y entre ellos. Veamos dos complementos que facilitan enormemente esta labor.

5.9.1 WP-RecentComments

Son muchos los complementos que se dedican a esta tarea: ofrecernos un listado de los últimos comentarios. Sin lugar a dudas, éste es el primer paso para facilitar el diálogo en la web. Por un lado, tenemos a la persona que ha comentado y que de esta forma puede ver cómo le responden; y por otro lado, tenemos al visitante que llega y, además de fijarse en las entradas, quiere ver cuáles han sido los últimos comentarios. De esta forma, puede responder o directamente ver información que en ocasiones es más valiosa que la de las entradas.

He elegido éste en lugar de sus alternativas por dos razones. La primera es la popularidad, que aunque siempre la anoto como razón pocas veces argumento el porqué. Siempre que un complemento es popular, tenemos la seguridad de que funciona medianamente bien y de que, probablemente, haya detrás del mismo un equipo que vaya a seguir actualizándolo. Si un complemento es genial, pero no tiene descargas, difícilmente seguirá evolucionando.

Las otras ventajas de este complemento, respecto a su competencia particular, se basan en que sólo se dedica a esto (soy de los que piensan que los *plugins* deben realizar una sola función y no mezclar temas diferentes) y, además, ofrece características interesantes, como separar los *trackback* de los comentarios propiamente dichos y mostrar los *gravatar* de los usuarios en el listado.



Figura 5.21. Arrastrando y soltando el widget a su zona de destino

Los pasos que hay que dar son prácticamente los mismos que en todos los complementos: instalación, activación y breve configuración (en el menú lateral del escritorio en la sección de ajustes). Como siempre, tened cuidado, que el nombre es muy popular y hay *plugins* con nombre muy similar. Finalmente, elegimos dónde mostrarlo. Para ello, haremos uso de un *widget* que aparecerá

dentro de la interfaz de control de estos elementos. Utilizamos la técnica de arrastrar y soltar encima del área de la web en donde queramos que salga (dependiendo del tema tendrá diferentes nombres: *left sidebar*, zona principal de *widgets*, *top*...). Una vez esté allí situado, elegimos el nombre y guardamos. Tened en cuenta que para elegir el orden de aparición utilizamos esta misma técnica, arrastrar y soltar, también conocida por su traducción inglesa, *drag&drop*.



Figura 5.22. Aspecto del plugin en funcionamiento

En este momento, podremos volver a la interfaz pública de nuestro blog y comprobar cómo se muestran los últimos comentarios, tal como hemos definido en los ajustes del complemento.

5.9.2 Subscribe To Comments Reloaded

Al entrar en la web ya podemos ver los últimos comentarios. Sin embargo, el visitante está obligado a entrar en la web para verlos. ¿Qué soluciones hay? WordPress nos ofrece varias opciones:

- Usar el canal de comentarios.
- Suscribirte a un canal que se crea con cada entrada. La dirección del mismo es `http://URL_entrada/feed`, sinceramente, para una persona no tiene mucha utilidad, ya que los comentarios de una sola entrada no son una fuente muy apropiada para un canal de sindicación.
- Ofrecerle al usuario la posibilidad de que se suscriba por correo, de forma que, cuando reciba respuestas a sus comentarios, le lleguen avisos.

Para implementar esta funcionalidad, disponemos de este *plugin*. Al escribir el comentario, al usuario se le da la posibilidad de marcar una casilla de verificación, de forma que cuando lleguen nuevos comentarios a la entrada se le envíen al correo. Si le interesa responder, responderá, y de esta forma enriquecerá la conversación en el sitio.

Usaremos el método que ya hemos visto en multitud de ocasiones en este capítulo para instalar y configurar el complemento. Hay detalles muy curiosos en la configuración del *plugin*, como la gestión de las diferentes suscripciones. También es importante traducir el texto que viene por defecto para generar los correos de *There is a new comment on the post* a “Hay un nuevo comentario en la entrada” o algo parecido.

5.10 SEGURIDAD

Siendo sinceros, ya hemos comentado el punto más importante de cara a mantener un nivel de seguridad elevado en WordPress: tener el sistema, los temas y los complementos siempre actualizados. Por lo tanto, no nos pararemos demasiado en este punto, que es prácticamente el final del capítulo. Sólo quería comentar la existencia de un complemento que nos ofrece una pequeña auditoría de seguridad, de forma que en un simple vistazo podemos mejorar algunos detalles para fortificar aún más nuestra instalación.

5.10.1 WP Security Scan

Antes de comenzar a hablar más sobre el complemento, lo presento con un solo dato: su creador es el mismo que ha desarrollado All In One SEO Pack, sin lugar dudas, una muy buena carta de presentación.



Figura 5.23. Opciones de seguridad ofrecidas por WP Security Scan

¿Qué nos ofrece el *plugin* una vez está instalado? Una serie de comprobaciones básicas sobre los puntos más importantes de nuestra seguridad. Las alertas principales son:

- Versión de WordPress: Nos informará de si tenemos o no la última versión instalada.
- Prefijo de la base de datos: Si es “wp_” nos dará un mensaje de aviso. Si alguien ejecuta una sentencia SQL vía inyección en el código y tenemos ese prefijo, el que viene por defecto, no tendrá problemas en saber el nombre de las tablas. El *plugin* permite cambiarlo.
- Versión de WordPress escondida (las versiones modernas del CMS están así por defecto).
- Errores de la base de datos: Si está a *off* será bueno para no ofrecer información indebida en caso de problemas.
- Las llamadas etiquetas “WP ID META” ya no aparecen en ningún núcleo de las versiones modernas de WordPress.
- Si el usuario administrador se llama “admin.”, le damos más facilidad al atacante. Lo mejor es tener usuarios administradores con nombres diferentes a éste.
- El hecho de tener un fichero .htaccess que restrinja el acceso al directorio “wp-admin/” también es una ventaja, pero supone demasiadas molestias para los beneficios que conlleva (si tenemos IP fija puede ser útil, pero no es el caso generalmente).

En la opción escáner nos aparecerá un listado de permisos de cada directorio y fichero de interés (en un capítulo anterior los hemos comentado) por lo que deberíais tenerlos todos correctos. Sin embargo, si estáis usando una instalación en un equipo propio con Windows, os aparecerán todos mal porque la política de permisos de Windows no llega a los niveles de restricción necesarios.

En la opción “Password Tool” tenemos un generador de contraseñas fuertes, en “Database” la herramienta para renombrar los prefijos de las tablas, que acabamos de ver hace un momento y, finalmente, en “Support” disponemos de enlaces de interés para sitios relacionados con la seguridad y la arquitectura del sistema de gestión de contenido.

5.11 MEJORA DEL RENDIMIENTO

Finalmente, llegamos al punto de optimización del rendimiento. Ya hemos dicho que WordPress es un sistema relativamente simple, pero dada su implementación en PHP (lenguaje interpretado, no compilado), su propia arquitectura y el uso de clases, cuando las visitas se cuentan por miles podemos

llegar a tener problemas derivados de sobrecarga. Esta situación será aún más peligrosa si estamos alojados en un servidor web en la red, ya que nuestro proveedor puede verse obligado a cancelar temporalmente nuestra cuenta, para evitar la caída no sólo de nuestro sitio, sino de nuestros “vecinos”.

En este punto daremos una serie de recetas que podemos dividir en tres grupos: instalación de un complemento para hacer caché, buenas prácticas y modificaciones parciales.

Tened en cuenta que, cuando hablamos de rendimiento, hablamos de carga del sistema que ha de procesar y generar nuestro sitio. Éste es uno de los problemas que sufren las aplicaciones web que se generan de forma dinámica y que no están almacenadas en ficheros HTML que simplemente se han de enviar en cada petición. Otra medida importante relacionada con este tema es el tamaño del envío. No es lo mismo tener un blog que ocupa 40 KB al enviarse por la red que otro que cada vez que refresque requiera la recarga de 1 MB.

5.11.1 WP-Cache

No es el único complemento que realiza esta labor, pero lo presento en lugar de otros también populares como WP Super Cache, porque lo he usado personalmente durante mucho tiempo y estoy muy contento con su rendimiento.

Cuando alguien accede a nuestro blog, el servidor ha de componer toda la información a mostrar a través del sistema de plantillas, complementos y la propia arquitectura del CMS. Tened en cuenta que este proceso es bastante pesado porque se realiza sobre un lenguaje, como PHP, que en lugar de ser compilado es interpretado. Los lenguajes interpretados por norma general son mucho más lentos porque requieren de algunos pasos extra para llegar a convertirse en lenguaje máquina. Aparte, hay que tener en cuenta que nos conectamos a una base de datos, a la que debemos hacer la petición, esperar a que la procese y, finalmente, recibirla para seguir con nuestro procesamiento. La conexión a la base de datos se realiza a través de una conexión de tipo cliente-servidor, aunque esté alojada en la misma máquina, por lo que aún se carga más el procesamiento.

¿Qué hace un *plugin* de este tipo? Los complementos de este tipo, al igual que todos los mecanismos de caché existentes en el mundo de la informática, basan su funcionamiento en duplicar algunos datos, a costa de tenerlos mucho más accesibles que en su lugar original. En nuestro caso, el lugar original tiene que componerse por los mecanismos antes descritos y la duplicación es un fichero de texto plano en HTML, que ya ha sido procesado previamente. Como entenderéis, este segundo mecanismo es mucho más inmediato.

Cada vez que alguien hace una petición nueva, el complemento guarda una copia de la misma en formato HTML. Cuando otro visitante llega y pide lo mismo, por ejemplo, ver el sitio principal, en lugar de pedirle a WordPress que cree el contenido, el complemento se encarga de la petición, devolviendo la página HTML que tiene almacenada en memoria.

WP-Cache es un *plugin* creado por Ricardo Gallir, creador de Menéame, y que tenéis disponible en el repositorio oficial de complementos. Su instalación es directa y tendréis acceso a sus opciones a través de la sección “Ajustes”, dentro del panel de administración de WordPress. Las opciones que aparecen por defecto son correctas para la mayoría de los casos, pero me gustaría advertiros de un problema que podéis sufrir. Cuando tenemos activado un complemento de caché, si estamos trabajando con nuestro tema o modificando *plugins* (como veremos en próximos capítulos), los cambios no se verán reflejados en ciertas circunstancias porque la web a la que estaremos accediendo estará guardada en la caché.

Este problema es de fácil solución, pero si no somos conscientes del mismo podremos llegar a creer que estamos haciendo mal las modificaciones. Tenemos dos opciones. La primera es desactivar el *plugin* mientras hacemos los cambios o, si lo preferimos, podemos ir a la administración del mismo y elegir la opción “Delete caché”, que borrará los datos almacenados y obligará a recargar la página.

5.11.2 Buenas prácticas

Además de tener el complemento oportuno instalado, también deberemos ser conscientes de que con nuestras decisiones podemos empeorar o mejorar radicalmente el rendimiento.

5.11.2.1 DESACTIVA COMPLEMENTOS QUE NO USES

En un capítulo posterior veremos que los complementos al instalarse, y activarse, se asocian a ciertas acciones para poder llevar a cabo su labor. En ocasiones, su ejecución puede ser muy rápida, pero, a veces, el rendimiento puede caer enormemente por culpa de un complemento que, además, no lleva a cabo una labor vital. Por esta razón, es conveniente tener activados (y diría que hasta instalados) sólo los complementos absolutamente necesarios y que nos sean útiles.

Lo mismo podríamos extrapolar a los temas. Aunque en este caso sólo podemos tener uno activado, tener instalados decenas de ellos no sólo es innecesario, sino que además alargará enormemente los procesos de copia de seguridad y mantendrá en el servidor datos innecesarios.

5.11.2.2 REDUCE PETICIONES Y SU TAMAÑO

Cada vez que cargas una imagen, ya sea de la plantilla o de las entradas, debe realizarse una petición HTTP. Da igual el tamaño de la misma, las cabeceras y todo lo que conlleva una petición ha de enviarse. La idea no es dejar de usar imágenes, mucho menos en las entradas, pero sí eliminar en los temas las más superfluas.

Aparte, también es muy importante el tamaño de las mismas. Si en nuestra cabecera tenemos un logo que mide 100 píxeles de ancho y de largo, pero en realidad la referencia es a una imagen de 1.000 x 1.000, el tiempo de descarga será enorme para una ganancia nula (más bien una pérdida, porque el navegador tendrá que redimensionarla). La idea es tener imágenes siempre del tamaño que se vayan a mostrar. De esta forma ahorraremos la reducción o aumento de tamaño al navegador y evitaremos transmitir muchos bytes que entorpecerán la navegación.

5.11.3 Modificaciones

Dentro de este punto, incluimos tanto modificaciones que podemos hacer directamente desde el panel de administración de WordPress como otras que deberemos llevar a cabo sobre el código de ciertos ficheros.

5.11.3.1 EVITAR QUE SE GENEREN MINIATURAS

Cuando subimos imágenes a nuestras entradas o páginas, WordPress automáticamente genera una serie de miniaturas de las mismas. Si nuestro tema no las usa, ni nosotros vamos a usarlas porque seguimos el consejo que acabo de comentar de usar siempre imágenes del tamaño necesario, podemos desactivar la creación de estas pequeñas imágenes. Tened en cuenta que el proceso de creación de las mismas supone un incremento del consumo de CPU y a su vez un espacio almacenado de forma innecesaria. Los procesos de redimensionado de imágenes son muy costosos desde el punto de vista de gasto de cómputo.

La solución más rápida e inmediata es establecer a cero tanto la anchura como la altura máxima de las miniaturas en sus diferentes tamaños, de forma que WordPress no volverá a crearlas. Para ello, accedemos dentro del panel de administración a “Ajustes” y, dentro de “Multimedia”, rellenamos los datos: anchura y altura, y confirmamos.

5.11.3.2 ELIGE QUÉ SERVICIOS DE NOTIFICACIÓN USAR

Antes hemos comentado la utilidad de los servicios de notificación. Gracias a ellos podemos avisar de cambios en nuestro sitio antes de que el robot del sitio, ya sea Google o cualquier otro, pase a visitarnos. Sin embargo, como toda medida de este tipo, también tiene sus desventajas. Cada vez que avisamos por un *ping* de este tipo a un servicio de estas características, cargamos nuestro servidor con un envío. Ya que el beneficio es muy grande, lo que debemos hacer es elegir bien a quién realizamos dicha notificación. Si en nuestra lista hay presentes sitios repetidos, por ejemplo, el mismo motor de búsqueda pero en distintos países, deberemos dejar sólo el referente a nuestro país (por ejemplo, el referente a google.es) y el general (el referente a google.com). Sed cuidadosos y eliminad sólo los necesarios, porque, como ya hemos dicho, el beneficio de estos sistemas de notificación es muy grande.

5.11.3.3 DEFINE LLAMADAS A FUNCIONES

Cuando WordPress hace una llamada a función, normalmente ha de buscarla dentro de la base de datos (luego veremos cómo). Existen funciones a las que se hacen llamadas de forma muy continua y que nosotros mismos podemos saber su resultado, ya que son datos muy generales del blog. Para evitar tanta llamada sin necesidad, podemos definir las directamente en “wp-config.php”. Al final del mismo, deberemos incluir las siguientes líneas de código (siempre antes de la etiqueta que marca el fin de código PHP “>”):

```
define('WP_HOME', 'http://tublog');
```

```
define('WP_SITEURL', 'http://tublog');
```

```
define('TEMPLATEPATH', '/ruta_absoluta_a/wp-content/themes/mitema');
```

```
define('STYLESHEETPATH', '/ruta_absoluta_a/wp-content/themes/mitema');
```

DESARROLLANDO CON WORDPRESS

6.1 ¿QUÉ ES UNA API?

Como ya hemos comentado en más de una ocasión a lo largo de este libro, API son las siglas de *Application Programming Interface* “interfaz de programación de aplicaciones”.

Hablando de forma más concreta podemos decir que una API es un conjunto de clases y procedimientos que ofrecen una capa de abstracción superior al desarrollador, de forma que vea como una caja negra el funcionamiento interno de ciertas funcionalidades, para facilitar el desarrollo de tareas más complejas.

Es decir, una API nos ofrece una serie de métodos para crear aplicaciones complejas sin tener que ir al fondo de la cuestión. Imaginad que el desarrollador de aplicaciones Windows tuviera que encargarse de redibujar una ventana al desplazarse por la ventana. Sería realmente difícil que él dibujara todas las formas que componen el típico formulario de Windows basados en las *WinForms*, conforme el ratón o el teclado le indicaran con sus movimientos. Nada de esto preocupa al desarrollador que trabaja directamente con la API de Windows para este tema, ya que él instancia formularios y agrega funcionalidad y lógica al mismo, despreocupándose de este tipo de cuestiones de más bajo nivel, que las proporciona la API.

Como veis, si no fuera por este concepto de desarrollo en niveles de abstracción, que está presente no sólo en el mundo del desarrollo software, sino también en otros campos como las telecomunicaciones y la propia arquitectura hardware, sería imposible llegar a tener las aplicaciones y sistemas que disfrutamos

hoy en día. De hecho, no sólo por facilidad y ahorro de tiempo, sino por pura capacidad de compatibilidad, homogeneidad y escalabilidad, estamos obligados a hacer uso de diferentes API conforme desarrollamos en distintos campos.

6.2 INTRODUCCIÓN AL DESARROLLO

En el caso de sistemas de tan altísimo nivel como WordPress (recordad que no estamos hablando de una librería ni de una interfaz gráfica, sino de un sistema gestor de contenido), el hecho de homogeneizar el desarrollo es aún más importante. El desarrollo en WordPress podemos estructurarlo en tres pilares básicos:

- Desarrollo de temas: El próximo capítulo.
- Desarrollo de *plugins*: El capítulo siguiente al próximo.
- Desarrollo de WordPress.

Como veis, el último pilar no será tratado en este libro. Es posible que cuando os habituéis al desarrollo de temas y complementos, tengáis las nociones suficientes para ayudar de forma puntual al desarrollo de todo el sistema de gestión de contenido, pero, si empezamos desde cero, es una meta muy lejana. Conforme vayamos describiendo cómo desarrollar temas y complementos, os daréis cuenta de cómo está implementado el corazón de este sistema. Sin lugar a dudas, es uno de los sistemas más potentes y mejor diseñados en este mundo y es un ejemplo a seguir para futuros desarrollos similares.

Cualquier tema o complementos que desarrollemos, podremos subirlo al repositorio oficial de WordPress de forma que quede disponible para todos los públicos, y, la comunidad al completo, lo mejore y lo evalúe. Hay también otras formas de colaborar con el proyecto WordPress que no requieren elevados conocimientos técnicos. Por ejemplo, podemos ayudar a traducir cualquier componente del sistema gestor de contenido, para que esté disponible en nuestra lengua. De esta forma, ayudamos a universalizar el sistema.

¿Qué debo saber para poder entender sin problemas las complicaciones que a partir de ahora se van a realizar? Veamos los pilares tecnológicos sobre los que se sustenta el CMS:

- Diseño y maquetación: HTML como lenguaje de representación de la información y CSS como estándar para definir estilos.
- Funcionalidad: PHP, el lenguaje más popular del lado del servidor.

- Manejo de datos: MySQL, el sistema gestor de base de datos de WordPress.



Figura 6.1. Logo de PHP

¿A qué nivel debemos tener conocimientos? Debemos desenvolvemos bien con la creación de sitios web directamente desde código HTML usando CSS, asimismo, debemos tener práctica en el desarrollo con lenguajes de programación procedimental, no necesariamente PHP, si conocéis C, C++, Java, C#... o similares, no tendréis problema en entender el código. Por otro lado, dado que se trata de un lenguaje del lado del servidor, si el lenguaje con el que habéis trabajado es para programas que ejecutan directamente en cliente, debéis pensar siempre que la acción en un sitio de este tipo se desarrolla de la siguiente forma:

- El usuario accede a la web desde el navegador.
- El servidor recibe la petición que puede incluir ciertos datos extra: qué categoría ver, qué página quiere visualizar, el texto de un comentario, etc.
- El servidor procesa (con PHP) y hace las llamadas oportunas a la base de datos (MySQL), conforme a la petición del usuario.
- El resultado final del procesamiento es un fichero HTML que será el que se le devuelva al usuario.
- El navegador del usuario recibe el resultado del procesamiento y lo muestra por pantalla.

Este proceso se repite de forma reiterada cada vez que un usuario maneja una aplicación. Se le pueden añadir más factores, pero a efectos básicos éste es el funcionamiento.



Figura 6.2. Logo de MySQL

La tecnología AJAX (*Asynchronous JavaScript And XML* “JavaScript y XML asíncrono”), que la hemos mencionado una vez en este texto, hace este proceso mucho más transparente al usuario, ya que realiza todos los pasos anteriores sin que el usuario cambie de página. En WordPress se usa la tecnología AJAX tanto en temas y complementos como en el mismo panel de administración. Por ejemplo, cuando movemos en la barra lateral un *widget* a otra posición, nosotros no vemos refresco alguno de la página, ya que el motor AJAX se encarga de realizar todas las acciones dejando la interfaz de usuario sin el típico refresco.

Finalmente, respecto a bases de datos, no debemos tener conocimientos profundos de MySQL, ya que toda la API de WordPress recubre muy bien las operaciones de inserción, borrado, lectura y actualización de registros. Lo que sí debemos conocer es la estructura interna de la base de datos que maneja todos los datos del sistema. Como todo sistema de bases de datos relacional, la base de datos de WordPress está organizada en forma de tablas, en la que cada registro representa una fila y las columnas definen los datos que pueden almacenarse. Es el mismo concepto que usamos en programas como Microsoft Access o incluso con Microsoft Excel cuando tratamos datos.

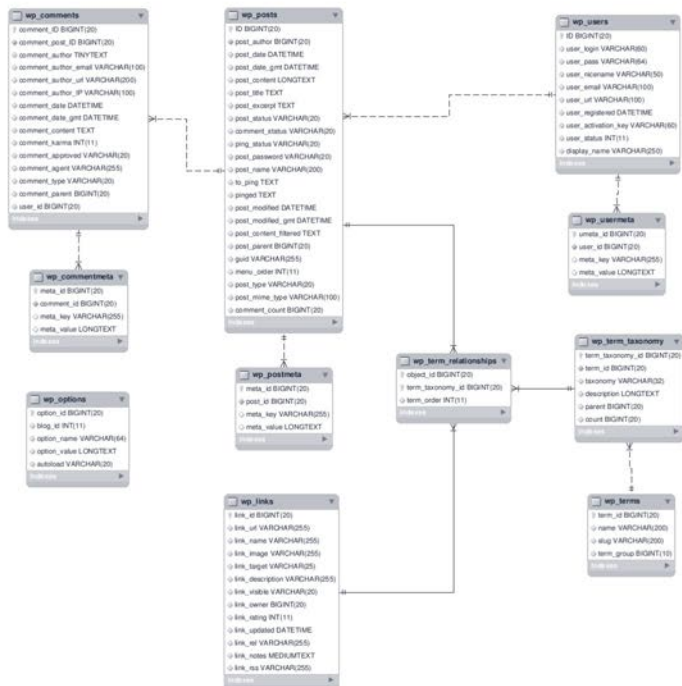


Figura 6.3. Diagrama entidad-interrelación de la base de datos de WordPress 3.0

6.3 LA BASE DE DATOS DE WORDPRESS

Una instalación básica de WordPress crea y maneja en la base de datos un total de 11 tablas, un número sensiblemente inferior al que puedan manejar otros sistemas gestores libres similares. Alrededor de estas tablas se organiza el funcionamiento básico de WordPress, desde usuarios a entradas pasando por comentarios y opciones propias del sistema, sin olvidar las relaciones entre ellos. Los *plugins* pueden crear tablas nuevas para su propia funcionalidad (tras instalar muchos complementos podemos llegar a tener más tablas de estas que del sistema principal).

Antes de pasar a hablar más detenidamente de cada una de ellas, en la página anterior podéis ver el diagrama de entidad-interrelación de las tablas de WordPress en su versión 3.0. Los cambios entre versiones existen, pero son mínimos.

Aunque no llegaréis a ver con detalle la imagen, con sólo un vistazo, podréis llegar a comprender la potencia de algo tan relativamente simple. Además, podéis observar que la mayoría de relaciones son del tipo uno a muchos “(1,n)”, lo que facilita enormemente la transformación a la práctica y la comprensión por parte del desarrollador. Otra nota curiosa es que sólo hay una tabla sin relaciones, la que se encarga de las opciones: *wp_options*.



Figura 6.4. Representación común de una base de datos en la web

6.3.1 wp_options

Comenzamos por la única tabla que almacena información que no se relaciona con el resto. Se trata de *wp_options* y, como podéis imaginar por el nombre, en ella se almacenan las opciones básicas del sistema, excepto la conexión a la base de datos, que recordad que se almacena en un fichero PHP, si no fuera así, sería imposible realizar una conexión a la base de datos.

Las opciones a las que me refiero son las que están presentes en la interfaz de administración del blog, en la pestaña “Ajustes”. En la referencia de opciones (http://codex.wordpress.org/Option_Reference), podéis ver todas las opciones posibles junto a lo que significan los valores que pueden tomar.

Veamos el ejemplo de un registro:

- *option_id*: 3, identificador de opción.
- *blog_id*: 0, en caso de que alberguemos más de un blog en un mismo sistema (ejemplo de cómo el antiguo WordPress MU se ha integrado en el núcleo).
- *option_name*: Nombre de la opción, por ejemplo “blogname”, nombre del blog.
- *option_value*: Valor de la opción, “Blog de ejemplo para Ra-Ma”.
- *autoload*: Autocarga, sí o no.

Otras opciones que aquí se almacenan son las que encontramos en el menú de administración en las siguientes pestañas:

- Ajustes: general, escritura, lectura, debates, privacidad y enlaces permanentes.
- Apariencia: *widgets*.

6.3.2 wp_usermeta

Seguimos nuestra descripción, comenzando con la primera tabla que sí tiene relación con otras. Se trata de la tabla destinada a almacenar metainformación de los usuarios del sistema. El prefijo “meta”, como en este caso metainformación, es ampliamente utilizado en las ciencias de la computación. Para nuestro caso significa “información que describe otra información”.

Es una tabla muy sencilla y se relaciona con la tabla “usuarios” tal como sigue: un usuario puede tener una o más descripciones de metainformación de usuario, es decir, un registro de usuario hace referencia a uno o más registros de metainformación de usuario.

¿Qué datos alberga?

- *umeta_id*: Identificador del registro.
- *user_id*: Usuario al que hace referencia.

- *meta_key*: Clave del registro, por ejemplo, *nickname*.
- *meta_value*: Valor de la clave, el *nick* para este ejemplo es *admin*.

6.3.3 wp_users

Remontamos en el diagrama y llegamos a otra de las tablas más importantes del sistema, la que alberga la información de los usuarios. Se relaciona de forma directa con la tabla anterior y con la tabla *wp_posts*, que describiremos a continuación. Los campos que alberga no son un misterio si hablamos de usuarios, de hecho coinciden con la mayoría de otros CMS:

- *ID*: Clave primaria del usuario.
- *user_login*: Nombre de usuario con el que iniciamos sesión en el sistema.
- *user_pass*: Contraseña del usuario, codificada en MD5 (un algoritmo *hash* de 128 muy utilizado).
- *user_nicename*: Nombre amigable del usuario, suele coincidir con el *login*.
- *user_email*: Correo electrónico del usuario.
- *user_url*: Dirección web personal del usuario.
- *user_registered*: Fecha y hora de registro del usuario.
- *user_activation_key*: Clave de activación, utilizada cuando hay pérdida de la contraseña.
- *user_status*: Valor ya no utilizado. Se mantiene en la base de datos por compatibilidad.
- *display_name*: Nombre que mostrar a los usuarios. Por ejemplo, en mi caso el *login* es carazo, pero mi nombre para mostrar podría ser Francisco Javier Carazo.

- *meta_key*: Clave del registro, por ejemplo, *nickname*.
- *meta_value*: Valor de la clave, el *nick* para este ejemplo es *admin*.

6.3.3 wp_users

Remontamos en el diagrama y llegamos a otra de las tablas más importantes del sistema, la que alberga la información de los usuarios. Se relaciona de forma directa con la tabla anterior y con la tabla *wp_posts*, que describiremos a continuación. Los campos que alberga no son un misterio si hablamos de usuarios, de hecho coinciden con la mayoría de otros CMS:

- *ID*: Clave primaria del usuario.
- *user_login*: Nombre de usuario con el que iniciamos sesión en el sistema.
- *user_pass*: Contraseña del usuario, codificada en MD5 (un algoritmo *hash* de 128 muy utilizado).
- *user_nicename*: Nombre amigable del usuario, suele coincidir con el *login*.
- *user_email*: Correo electrónico del usuario.
- *user_url*: Dirección web personal del usuario.
- *user_registered*: Fecha y hora de registro del usuario.
- *user_activation_key*: Clave de activación, utilizada cuando hay pérdida de la contraseña.
- *user_status*: Valor ya no utilizado. Se mantiene en la base de datos por compatibilidad.
- *display_name*: Nombre que mostrar a los usuarios. Por ejemplo, en mi caso el *login* es carazo, pero mi nombre para mostrar podría ser Francisco Javier Carazo.

6.3.4 wp_postmeta

Si *wp_usermeta* ofrece metainformación de la tabla de los usuarios, *wp_postmeta* es el mismo concepto aplicado a las entradas que veremos a continuación.

¿Qué datos alberga? Los mismos que en el caso anterior, pero cambiando el nombre de los identificadores:

- *meta_id*: Identificador del registro.
- *post_id*: Entrada a la que hace referencia.
- *meta_key*: Clave del registro, por ejemplo, *btfa_body_title*.
- *meta_value*: Valor de la clave, el nick para este ejemplo es *Bienvenido al nuevo portal de LinuxHispano*.

6.3.5 wp_posts

Como dice la referencia del CMS, esta tabla es el corazón de todo el sistema. Por su nombre podréis deducir que almacena las entradas, pero además de ellas, también encontramos la información referente a páginas y a ítems del menú de navegación. Su estructura interna es más extensa que en el caso de la tabla anterior, pero en absoluto es difícil de entender.

Es la tabla con más referencias directas, cuatro en total:

- *wp_users*: Un usuario puede tener cero o más entradas.
- *wp_comments*: Una entrada puede tener cero o más comentarios.
- *wp_postmeta*: Relación análoga a la de los usuarios con su metainformación, una entrada tiene cero o más registros metainformación de entrada asociada.
- *wp_term_relationship*: Las entradas tienen categorías y etiquetas gracias a la existencia de esta relación. Una entrada puede aparecer en una o más relaciones de términos.

Como veis, conforme avanzamos aparecen nuevas tablas que gracias a sus relaciones proveen la funcionalidad que todos disfrutamos en WordPress. Veamos los campos que almacena esta tabla como tal y qué campos sirven de clave foránea para mantener las relaciones con las demás tablas.

- *ID*: Clave primaria del *post*.
- *post_author*: Autor de la entrada. Clave foránea con *wp_users*.
- *post_date*: Fecha de creación de la entrada.
- *post_date_GMT*: Fecha de creación de la entrada en horario GMT (*Greenwich Meridian Time* “tiempo del meridiano de Greenwich”, unidad estándar de tiempo cuando podemos manejar diferentes zonas geográficas).
- *post_content*: Contenido de la entrada.
- *post_title*: Título de la entrada.
- *post_excerpt*: Extracto de la entrada.
- *post_status*: Estado de la entrada, puede ser:
 - *publish*: Publicada.
 - *inherit*: Utilizada.
 - *pending*: Pendiente.
 - *private*: Privado.
 - *draft*: Borrador.
 - *trash*: En la basura.
- *comment_status*: Estado de los comentarios, abierto o cerrado.
- *ping_status*: Estado del *ping* automático, abierto o cerrado.
- *post_password*: Contraseña de la entrada, en caso de que la tenga.

- *post_name*: Nombre de la entrada. Si está publicada será el del enlace permanente. Si no, es un nombre interno.
- *to_ping*: Debe hacerse *ping* automático o no.
- *pinged*: El *ping* automático se ha realizado o no.
- *post_modified*: Fecha y hora de modificación.
- *post_modified_gmt*: Ídem pero en tiempo GMT.
- *post_content_filtered*: Contenido de la entrada que ha sido filtrado.
- *post_parent*: Entrada padre. Las revisiones tendrán como entrada padre a la publicada. Es una referencia dentro de la propia tabla.
- *guid*: URL de la entrada sin enlace permanente.
- *menu_order*: Si está dentro de un menú, aquí se almacena el orden. En las entradas no tiene sentido, sí en las páginas.
- *post_type*: Tipo de registro (entrada, página, adjunto).
- *post_mime_type*: En caso de ser un adjunto, se indica qué tipo de fichero es (imagen con su formato, PDF, audio con su formato...).
- *comments_count*: Contador de comentarios.

6.3.6 wp_commentsmeta

Como no podía ser de otra forma, los comentarios también tienen metainformación relacionada con ellos. Por lo general, esta tabla es más utilizada por complementos que interactúan con los comentarios que por el propio CMS (en una instalación sin *plugins* que la usen esta tabla normalmente estará vacía).

La estructura es idéntica a las otras tablas de metainformación, pero adaptando los nombres de las filas a este caso particular:

- *meta_id*: Identificador del registro.
- *comment_id*: Comentario al que hace referencia.

- *meta_key*: Clave del registro.
- *meta_value*: Valor de la clave.

6.3.7 wp_comments

Llegamos a la tabla encargada de almacenar la información de los comentarios, además de los *trackback*, *pingback* y otros enlaces entrantes. Como ya supondréis, tiene dos relaciones: con la tabla de entradas (una entrada, varios comentarios) y con su metainformación (un comentario, varios registros de metainformación de comentarios). Sin llegar a almacenar tanta información como la tabla destinada a las entradas, sí se almacenan bastantes datos de cada registro de comentario.

- *comment_ID*: Clave primaria.
- *comment_post_ID*: Entrada a la que hace referencia el comentario.
- *comment_author*: Nombre del autor del comentario.
- *comment_author_email*: Correo electrónico del autor del comentario.
- *comment_author_url*: Dirección web del autor del comentario.
- *comment_author_IP*: Dirección IP desde la que se ha hecho el comentario (en caso de comentarios ofensivos o que incluyan contenido ilegal, será necesario revisar este dato).
- *comment_date*: Fecha y hora de realización del comentario.
- *comment_date_gmt*: Ídem pero en horario GMT.
- *comment_content*: Contenido del comentario.
- *comment_karma*: Puntuación del comentario. Si no tenemos puntuaciones en los comentarios, será cero.
- *comment_approved*: Comentario aprobado o no.
- *comment_agent*: Agente desde el que se realiza el comentario. Se trata de *trackbacks* u otros enlaces entrantes, almacena qué tecnología se ha

usado, y, en caso de ser un comentario estándar, almacena navegador y sistema operativo.

- *comment_type*: Tipo de comentario, vacío si es un comentario, en caso contrario indicará de qué se trata: *pingback* o *trackback*, básicamente.
- *comment_parent*: Si el comentario tiene padre, su referencia, en caso contrario, cero.
- *user_id*: Si el usuario está registrado, su identificador en el sistema, en caso contrario, cero.

6.3.8 wp_term_relationships

Llegamos ya al extremo del diagrama donde se da forma a la información referente a las categorías, las etiquetas y los enlaces, a la vez que se establecen las relaciones necesarias con el resto de la información.

Comenzamos por la tabla dedicada a representar las relaciones. Es la única que tiene relación directa con la tabla de las entradas. Además de con la tabla *wp_posts* también se relaciona con la tabla de los enlaces (*wp_links*) y con la que almacena la taxonomía de los términos (*wp_term_taxonomy*). Para diferenciar entre un tipo de categoría y otra en las consultas, hay que mirar en *wp_term_taxonomy* si la categoría es de tipo entrada o enlace. Al tratarse una tabla que soporta relaciones entre terceras, su estructura es muy simple: dos claves foráneas que hacen a la vez el papel de primarias para la tabla en sí, y una columna más donde se almacena el orden, en caso de que lo haya. Veamos:

- *object_id*: Clave foránea a la tabla de las entradas o los enlaces, según corresponda.
- *term_taxonomy_id*: Clave foránea a la tabla de la taxonomía de los términos, que veremos a continuación.
- *term_order*: Orden del término, en caso de que sea necesario.

6.3.9 wp_term_taxonomy

La información relativa a las categorías y etiquetas se almacena en tres tablas: la anterior para las relaciones con las otras entidades, ésta para la taxonomía de las mismas y la siguiente para la información propia de cada término. Tiene una relación uno a muchos con la tabla *wp_terms* (un término, varias taxonomías) y

otra del mismo tipo con la tabla de las relaciones (una taxonomía, varias apariciones en las relaciones).

La información que se maneja desde aquí es:

- *term_taxonomy_id*: Clave primaria del conjunto.
- *term_id*: Clave foránea a la tabla de los términos.
- *taxonomy*: Tipo de término (categoría, categoría de enlace o etiqueta).
- *description*: Descripción.
- *parent*: Taxonomía de término padre (sirve para realizar relaciones de jerarquía).
- *count*: Número de ocurrencias (en complementos como la nube de etiquetas es muy importante).

6.3.10 wp_terms

Finalmente, llegamos a la tabla que almacena los datos esenciales de cada categoría y etiqueta. Su nombre es *wp_terms* y se relaciona únicamente la tabla de la taxonomía. Por lo tanto, la interrelación entre las categorías y etiquetas con las entradas se produce a través de dos tablas intermedias (las dos que acabamos de ver). La relación es como hemos visto con la tabla de las taxonomías, y los datos que se almacenan en cada registro de esta tabla son los siguientes:

- *term_id*: Clave primaria de la tabla.
- *name*: Nombre de la etiqueta o categoría.
- *slug*: Nombre corto de la etiqueta o categoría (es el utilizado en enlaces permanentes).
- *term_group*: Si pertenece a algún grupo de términos, se almacena aquí.

6.3.11 wp_links

Llegamos a la última tabla que vamos a describir de WordPress. Por su nombre ya supondréis que es donde se almacenan los enlaces. Aunque son una característica que en la versión actual de WordPress no tiene demasiado uso, es

ideal para crear directorios web y en versiones futuras es posible que empiece a usarse para dar forma a las referencias internas.



Figura 6.5. Enlaces web

La única tabla con la que se relaciona de forma directa es con *wp_term_relationships*, de forma que a raíz de ahí se puede ver a qué categoría de enlace corresponde nuestro registro. Los datos que se almacenan propios de cada registro son los siguientes:

- *link_id*: Identificador de la tabla.
- *link_url*: Dirección del enlace.
- *link_name*: Nombre del enlace.
- *link_image*: Ruta de la imagen que identifique el enlace, en caso de que la tenga.
- *link_target*: *none*, *blank* o *top*, como los *target* de los enlaces HTML.
- *link_description*: Descripción.
- *link_visible*: Si es visible o no.
- *link_owner*: 1 si somos propietarios del enlace, 0 en caso contrario.
- *link_rating*: Valoración en caso de que la tenga.
- *link_updated*: Última actualización.

- *link_rel*: Relación con el enlace (si se trata de otra dirección nuestra, de la web de un amigo o de un contacto o un conocido...).
- *link_notes*: Notas que queramos añadir, aparte de la descripción.
- *link_rss*: Canal de *feed* asociado en caso de que lo tenga.

6.4 COOKIES

A pesar de la mala fama que ha perseguido a las *cookies*, o “galletitas”, desde su aparición en el mundo de Internet allá por el año 1995 de la mano de Lou Montulli, son una herramienta indispensable para ofrecer ciertas funcionalidades que, de otra forma, sería imposible proveer, debido a que el protocolo HTTP es incapaz de almacenar información por sí mismo.

Aunque es muy probable que no tratemos con ellas de forma directa, siempre viene bien saber qué información se gestiona con ellas en el cliente, para ver hasta qué punto la tecnología en sí nos deja ofrecer más o menos posibilidades a la hora de desarrollar.

Antes de seguir explicando más detalles sobre las *cookies* de WordPress, voy a explicar brevemente qué son y para qué sirven. Una *cookie* es una porción de código que contiene datos que se genera en el lado del cliente (es decir, en el del usuario, a través de su navegador).

¿Qué contienen esos datos? Siempre serán datos en formato texto, es decir, nada de contenido multimedia (imágenes, videos, audio, etc.), que son almacenados en la caché del navegador y estarán también alojados en la memoria rígida del usuario, por lo que seguirán estando vigentes de forma permanente, hasta que lo indique el desarrollador o hasta que el navegador lo considere oportuno. Puesto que el protocolo HTTP no almacena información de la conexión por sí mismo, si no dispusiéramos de las *cookies*, cada petición sería nueva para el navegador. De esta forma, sería imposible gestionar conceptos como el de sesión, el hecho de ya estar *logueado* o mantener los credenciales para futuras comprobaciones de identidad.

Sin embargo, al ser una parte de información que se almacena en el disco duro del cliente y que el desarrollador recolecta en función de sus necesidades, puede constituir una fuente de información de la navegación del usuario. Por esta razón, durante mucho tiempo las *cookies* han tenido esa marca de elemento negativo y casi de *spyware* (software de espionaje). Actualmente, dada las limitaciones que imponen los navegadores y el nivel de desarrollo de la tecnología, no constituye un problema de privacidad y como veis es una herramienta básica

incluso para sistemas que, al fin y al cabo, no mantienen mucha información del visitante, como es el caso de WordPress.



Figura 6.6. *Típica cookie americana*

Distinguiremos dos tipos de *cookies* en este CMS: las que se generan cuando eres usuario del sitio y las generadas cuando comentan los visitantes.

6.4.1 Cookie de usuario

Los usuarios son las personas que tienen una cuenta en tu blog de WordPress. Cuando inicias sesión en el sistema, a través del fichero *wp-login.php* o el directorio *wp-admin* (que te redirige al anterior), el CMS genera y almacena una *cookie* con dos datos:

- Tu nombre de usuario.
- Tu contraseña tras aplicarle dos veces una función *hash*.

¿Qué es una función *hash*? Se trata de una función o algoritmo que convierte una serie de datos en una cadena totalmente diferente. Sus características más importantes son las siguientes:

- El resultado es siempre el mismo para una misma entrada.
- Es imposible conseguir la entrada con la salida.

Puesto que las *cookies* deben tener un tiempo de validez (que puede ser finito o infinito), en WordPress la *cookie* tiene una validez de dos semanas. Cuando salimos del sistema la *cookie* se borra. Si no existiera esta *cookie* deberíamos autenticarnos continuamente.

6.4.2 Cookie de comentarista

Las personas que hacen comentarios en el blog también son objetivo de almacenamiento de parte de su información. No se hace por fines de autenticación como en el caso anterior, sino por mera comodidad del usuario, de forma que no tenga que repetir los mismos datos cuando vuelve a comentar. Las *cookies* almacenan los siguientes datos:

- Nombre.
- Correo electrónico.
- Dirección web.

6.4.3 Funciones de manejo de cookies

Éste es el momento en el que vamos a comenzar a comprobar las funcionalidades que provee directamente la API del CMS. Más concretamente, las tres primeras funciones que vamos a conocer de la misma. Una función en programación es una porción de código que recibe una serie de parámetros, realiza unas acciones y, en ocasiones, devuelve un valor. Las funciones articulan tareas básicas, de forma que se facilita y se estructura el desarrollo.

Las funciones WordPress están codificadas en PHP, por lo que siguen la sintaxis propia de dicha tecnología. Quizás no sea la funcionalidad más importante, la gestión de *cookies*, pero es buen momento para introducir este concepto, ya que en el capítulo siguiente lo vamos a tratar ampliamente y éstas que os presento aquí tienen una funcionalidad bien acotada. Las funciones que vamos a ver están definidas en el directorio `/wp-includes/pluggable-functions.php` (siempre podréis ver el código de las mismas, una de las bondades del software libre). Veamos los prototipos de las mismas:

- `wp_setcookie($username, $password, $already_md5 = false, $home = "", $siteurl = "")`: Establece una *cookie* para un usuario, con su contraseña, indicando si está codificada con MD5 y añadiendo, de forma opcional, lugar de inicio y URL.

- `wp_clearcookie()`: Borra la *cookie* del navegador. Se ejecuta cuando cerramos sesión.
- `auth_redirect()`: Comprueba si estás autenticado a través de la *cookie*, en caso de no estarlo, te redirige a *wp-login.php*.

Acabamos de ver las primeras funciones propias de la API de WordPress. Probablemente no sean las más funcionales e incluso es posible que nunca las uséis, pero el objetivo era presentaroslas para que veáis lo simple que es realizar funciones de muy alto nivel apoyándonos en el código que nos proporciona el sistema gestor de contenido.

Comenzaremos aplicándolas primero a la creación y modificación de temas WordPress, para, posteriormente, ver cómo funcionan internamente los *plugins* y desarrollar nosotros uno. Antes de terminar el libro, veremos dos ejemplos prácticos de aplicación de WordPress a situaciones reales.

CREACIÓN DE TEMAS

Hemos visto cómo instalar temas para WordPress tanto desde el repositorio oficial del proyecto como desde otros sitios, subiendo el fichero a nuestro servidor. Sin embargo, posiblemente, siempre tendremos detalles de los temas que instalemos que queramos cambiar. Modificar frases, formas de mostrar las cosas, colores o imágenes para adaptarlo más a nuestro gusto. Si la interfaz de administración del tema no permite esto o directamente es que no existe como tal, debemos manejar el código.

En este tema aprenderemos a manejar el código de los temas (también llamadas plantillas o, por su nombre en inglés, *themes*) WordPress, desde su parte de estructuración en HTML a la disposición de los estilos gráficos con CSS, para terminar viendo cómo funciona la parte lógica. Haremos pequeñas modificaciones, para, posteriormente, hacer cada vez modificaciones más grandes y terminar creando un tema de WordPress desde cero, basándonos en una plantilla con HTML y CSS.

Antes de seguir, quería insistir mucho en que en esta parte no hace falta tener grandes conocimientos de programación, ni siquiera de PHP, ya que como veréis la API es muy clara y usaremos siempre una serie de funciones muy entendibles. Lo que sí es necesario es entender el código HTML y las hojas de estilo CSS. Si sabéis manejar algún programa como Photoshop o Gimp (programa de manipulación de imágenes de GNU), para editar imágenes, mejor.

Aprovecho este punto para recomendaros Gimp (<http://www.gimp.org/>), el mejor programa libre (y por supuesto gratuito) para manipulación de imágenes. Un gran aliado en este tipo de tareas.



Figura 7.1. Logo de Gimp

7.1 ESTRUCTURA BÁSICA

El primer punto que hay que tratar es la estructura básica de un tema WordPress. Vamos a basar la explicación en el tema que incluye por defecto WordPress 3.0, “Twenty Ten”, toda una referencia en cuanto a plantillas. Accedemos al repositorio oficial y descargamos el fichero del tema.

Como veréis es un .ZIP que incluye los siguientes directorios y ficheros.

7.1.1 Directorio

Suelen ser estos dos, aunque puede haber temas con más o menos directorios, en función de las necesidades del autor.

- *Images*: Imágenes que utiliza el tema de forma interna.
- *Languages*: Donde estarán los ficheros de traducción del tema a otros idiomas.

WordPress utiliza los ficheros .pot, .po y .mo para las traducciones. Voy a mencionar qué hace cada uno de ellos por si en algún momento los usáis en un tema o complemento que hagáis en varios idiomas (nosotros por defecto los haremos en español).

- *.pot*: Ficheros de plantilla. Incluirá todas las cadenas que hay que traducir. Se manejan dos conceptos: *mugid*, identificador de mensaje y *msgstr*, el texto traducido. Al ser una plantilla, estos segundos estarán vacíos.
- *.po*: Son específicos de cada idioma (*es-ES.po*, *fr.po*...) y están basados en un *.pot*, de forma que incluyen las traducciones para el idioma en concreto en el lugar adecuado (*msgstr*).
- *.mo*: Es el fichero que finalmente usan las aplicaciones. Es un compilado del *.po*, en formato binario, de forma que es mucho más rápido en operaciones de acceso y lectura.

Existen utilidades de código abierto como Poedit (<http://www.poedit.net/>) que facilitan enormemente el tratamiento de estos ficheros.

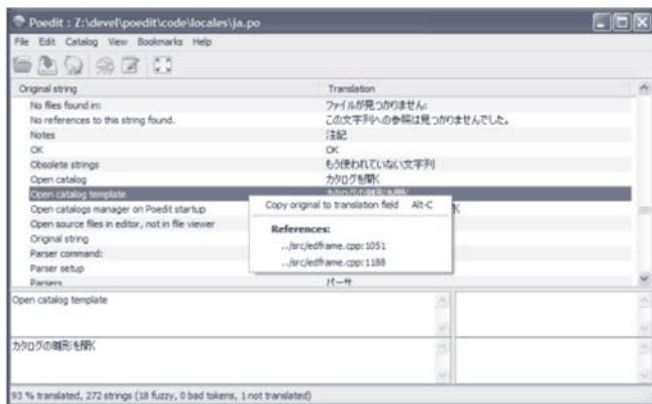


Figura 7.2. Aspecto Poedit

7.1.2 Ficheros

Aquí es donde está la verdadera funcionalidad y estructura del tema. Twenty Ten, aunque de aspecto simple, es un tema que reúne gran variedad de ficheros, no sólo los indispensables, sino otros que aportan más posibilidades. Veamos uno a uno.

Comenzamos por los ficheros que no son PHP ni hojas de estilo. Tenemos:

- *screenshot.png*: Es el pantallazo del tema. Sirve para mostrar en el repositorio y en la administración de temas el aspecto del mismo.
- *license.txt*: Fichero texto con la licencia del tema. En este caso GPL. Recordad que si nos basamos en este tema para hacer uno propio, deberemos hacer el tema con la misma licencia y referenciar que nos hemos basado en éste.

Seguimos con las hojas de estilo (CSS):

- *style.css*: Hoja de estilos principal.
- *rtl.css*: Hoja de estilo para escritura de derecha a izquierda (*right to left*).
- *editor-style.css*: Hoja de estilo del editor WYSIWYG.
- *editor-style-rtl.css*: Ídem pero para escritura de derecha a izquierda, como en el segundo caso.

Y, finalmente, el grueso del tema, los ficheros PHP:

- *404.php*: Cuando un usuario busca algo que no está en el blog (error 404), se ejecutará este fichero.
- *archive.php*: Define cómo se muestran los archivos de noticias.
- *attachment.php*: Muestra los adjuntos.
- *author.php*: Página donde se muestran un archivo de entradas de cada autor.
- *category.php*: Ídem que la anterior pero muestra un archivo de categorías.
- *comments.php*: Plantilla para mostrar tanto comentarios, como formulario para comentar.
- *footer.php*: Pie de página. Será llamado desde prácticamente todos los demás ficheros, al igual que la cabecera y la barra lateral.

- *functions.php*: Fichero donde se definen funciones propias del tema, que además de permitir la funcionalidad en otros ficheros, modifican parte del funcionamiento del núcleo general de WordPress.
- *header.php*: Igual que el pie de página, pero en esta ocasión es la cabecera.
- *index.php*: Aunque parezca el fichero más importante, es todo lo contrario. Sólo llama a las funciones respectivas y se ejecuta cuando se accede al portal.
- *loop.php*: Aquí se define cómo se muestran las entradas (con todo lo que conlleva). Es el mismo *loop* para múltiples situaciones, por lo que el código es algo complejo. Sin lugar a dudas, uno de los pilares de WordPress. También puede existir un fichero muy relacionado con éste: *loop-template.php*.
- *onecolumn-page.php*: Muestra la página sin barras laterales.
- *page.php*: Plantilla para mostrar páginas.
- *search.php*: Muestra los resultado de una búsqueda.
- *sidebar.php*: Barra lateral. Permite inclusión de *widgets*.
- *sidebar-footer.php*: Zona de *widgets* del pie de página.
- *single.php*: Plantilla para mostrar una sola entrada.
- *tag.php*: Ídem que en el caso de los archivos para los autores, pero esta vez para las etiquetas.

Acabamos de ver de una forma muy genérica cuál es el objetivo principal de cada fichero que compone el tema. Nos toca empezar a modificarlos, pero antes de empezar con esta labor debemos comprender el concepto de tema heredado. Si hacemos modificaciones sobre un tema de forma directa, tendremos problemas y muy graves. El mayor de ellos es que al actualizar el tema se destruyan nuestras modificaciones, pero hay otros que, sin ser tan graves, son molestos, como que WordPress nunca sabrá que hemos modificado el tema y hasta para elegirlo en el administrador tendremos problemas.

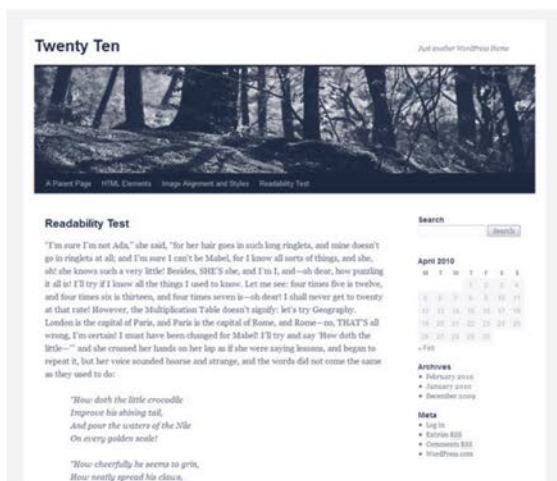


Figura 7.3. Test de legibilidad del tema Twenty Ten

7.2 TEMAS HIJOS

Los *Child Themes*, temas hijos traducido al español, son la forma recomendada por WordPress para modificar temas. ¿Qué hace un tema hijo? Básicamente, hereda la funcionalidad de un tema, al que llamamos padre, al que, además, añade o modifica parte de la funcionalidad.

¿Qué ventajas tiene sobre hacer un tema desde cero? Cuando trabajas sobre un tema hijo, sigues llamando a funciones del tema padre, de forma que te aprovechas de, por un lado, poder personalizar lo que creas conveniente y, por otro, de todas las actualizaciones que sufra el tema padre. Por esta razón, el equipo de WordPress recomienda siempre su uso.

7.2.1 Creación

El tema hijo más básico que podemos crear es el siguiente. Nos dirigimos a la ruta donde tenemos WordPress instalado, accedemos a la carpeta donde están los temas (*wp-content/themes*) y allí veremos una carpeta con el tema Twenty Ten (*twentyten* de nombre). Creamos una nueva carpeta con nombre: “twentyten-hijo”.

Ahora copiamos y pegamos el fichero CSS del directorio de Twenty Ten a este nuevo directorio. Este fichero CSS es requerido por los temas hijos y como mínimo ha de estar presente en el nuevo tema. Aparte de este fichero hijo podemos añadir los ficheros que creamos necesarios, pero éste es obligatorio, los demás son opcionales.

El siguiente paso es modificar dicho fichero CSS. Si os fijáis, en la cabecera del mismo, comentado, vienen una serie de metadatos que deberemos modificar. Son los siguientes:

- *Theme name*: Nombre del tema, pondremos Twenty Ten hijo. Es obligatorio.
- *Theme URI*: Referencia web del tema. Es opcional, por lo que la quitamos.
- *Description*: Descripción, es opcional. La rellenamos “Tema hijo de...”.
- *Author*: Autor del tema, es opcional.
- *Author URI*: Referencia web del autor del tema. También opcional.
- *Version*: Número de versión. Opcional.
- *Tags*: Etiquetas que describen el tema. Son básicas para permitir una búsqueda en el repositorio de WordPress.
- *Template*: Obligatorio, indica el directorio donde se encuentra el tema padre.

Tenemos ya la metainformación del tema lista, el siguiente paso es eliminar todo el contenido del CSS, a excepción de la cabecera que acabamos de modificar, e importar el contenido de la hoja de estilos original. ¿Cómo? Con la siguiente directiva (colocada justo tras los comentarios):

```
@import url("../twentyten/style.css");
```

Tras hacer esto, ya podemos acceder a nuestro gestor de temas en el escritorio de WordPress y podremos ver cómo aparece el tema hijo. Fijaos que no tiene toma de pantalla y que si lo activáis el efecto es el mismo que si tuviéramos el tema padre activado, pero éste es el primer paso.

La sentencia *@import* debe ser la primera en el fichero tras los comentarios, porque si no, queda invalidada.

A partir de ahora podemos comenzar a modificar el CSS nuevo, de forma que todas las reglas que añadamos tendrán más prioridad que las del CSS anterior, por lo que podremos escribir ahí directamente lo que queramos.



Figura 7.4. Tema hijo en el gestor de temas de nuestro blog

7.2.2 La hoja de estilos

Puesto que éste no es un libro de hojas de estilo, como ya hemos comentado antes, no vamos a explicar todas y cada una de las reglas que modificamos pero sí las más importantes.

La situación actual es la de una hoja de estilos en el nuevo tema, el tema hijo, en la que sólo aparecen dos elementos:

- La cabecera con la metainformación.
- La sentencia de importación.

Nuestra labor ahora es seguir rellenando la hoja de estilos. Para ello, deberemos tener siempre abierta la hoja de estilos antigua, de forma que podamos tomar referencias de los elementos que componen el sitio sin tener que ir a verlo directamente al código PHP.

Primer elemento que vamos a modificar: el color de las letras de cabecera. ¿Cómo? Lo primero que debemos hacer es acceder al código fuente de la página. Para ello, con el atajo de teclado “Control+U” desde Firefox o Chrome, o en Internet Explorer accediendo al menú “Página” en la opción “Ver código fuente”. Una vez tengamos el código fuente delante, buscamos el texto que tenemos como título del blog: “Blog de pruebas para Ra-Ma”. Encontremos algo así:

```
<h1 id="site-title">
    <span>
        <a href="http://localhost/wordpress/" title="Blog de
pruebas para Ra-Ma" rel="home">Blog de pruebas para Ra-Ma</a>
    </span>
</h1>
```

Como veis, la etiqueta “h1”, de cabecera de primer nivel (*header*), tiene un identificador de nombre “site-title”. Para cambiar el color de fuente de un elemento de tipo enlace, en función de su identificador, tenemos la siguiente regla CSS que deberemos incluir en el fichero de estilos del tema hijo, al final del mismo:

```
#site-title a { color: blue; }
```

Guardamos el fichero y recargamos el blog (teniendo activado el tema hijo), veremos como las letras de cabecera han cambiado de color. De igual forma que hemos modificado las letras de la cabecera, podemos aplicar cualquier regla CSS que queramos a cualquier otro elemento del blog.



Figura 7.5. Logo de validación de CSS del W3C

No profundizaremos demasiado en el tema, porque en realidad sólo deberemos aplicar reglas CSS de igual forma que haríamos con otro diseño web. Por ejemplo, para cambiar la fuente del sitio sólo deberíamos agregar la siguiente regla:

```
body{  
  
    font-family: times, serif;  
  
    font-size: 14pt;  
  
    font-style: italic  
  
}
```

Si queremos ir modificando aspectos más concretos, deberemos ver en el código fuente a qué elemento nos estamos refiriendo. Además de por identificadores, siempre podréis hacer reglas específicas para clases. Por ejemplo, en Twenty Ten las entradas pertenecen todas a la clase “post” y en el contenido de las mismas a la clase “entry-content”. De esta forma, podríamos cambiar el color de la fuente del contenido. Veamos cómo cambiarlo a amarillo:

```
.entry-content{ color: yellow;}
```

Siguiendo estas reglas generales, podremos aprovechar todo el potencial de las hojas de estilo para modificar los temas de WordPress a nuestro antojo.

7.2.3 El fichero de funciones

Acabamos de ver cómo modificar el estilo de nuestro tema hijo mediante la adición de un CSS, en el que importamos el del tema padre y luego añadimos contenido propio. Pasamos a otro de los ficheros más importantes de un tema WordPress, el que define las funciones propias del mismo. Está implementado en PHP y, a diferencia del anterior, cuando lo creamos en el tema hijo éste no sobrescribe al del padre (de ahí que tuviéramos que importarlo), sino que se cargan ambos, más concretamente, el fichero hijo se carga justo antes que el fichero del padre.

El primer paso será crear un fichero vacío de nombre *functions.php* en la carpeta principal del tema hijo. Internamente se trata de un fichero PHP, encerrado por los tokens “<?php” y “?>”, para el comienzo y el final respectivamente, en el que se definen una serie de funciones y se hacen llamadas a funciones propias de la API de WordPress. De esta forma, podemos crear acciones a nuestra medida, implementando nuestras propias funciones, a las que luego unimos a acciones propias del tema, gracias a la API de WordPress.

Veamos un ejemplo sencillo. Queremos añadir un icono para nuestra web, los llamados *favicon* (del inglés, *favorites icon*, icono de favoritos). Para ello, debemos crear una función que devuelva la sentencia HTML necesaria (vamos a usar el *favicon* de WordPress):

```
<link rel="shortcut icon" type="image/x-icon" href="
http://codex.wordpress.org/favicon.ico" />
```

La función se limitará a devolver dicha cadena en forma de eco:

```
function enlace_favicon() {
    echo '<link rel="shortcut icon" type="image/x-icon" href="
http://codex.wordpress.org/favicon.ico" />'. "\n";
}
```

Ahora debemos indicarle a WordPress que ejecute esa función en el momento en que esté generando la cabecera del sitio (ahí es donde debe ir esa etiqueta). Por lo tanto, llamaremos a la función *add_action* (añadir acción), una de las funciones que más usaremos en este capítulo y el siguiente. ¿Cómo se hace una llamada a esa función? Aunque soporta más argumentos, concretamente para definir prioridades y para definir cuántos argumentos pueden aceptar la función a la que se hace la llamada, nos limitaremos a los dos básicos:

- Acción a la que unir (también llamado *hook*, gancho).
- Función a la que llamar.

Ambos argumentos son de tipo cadena. Para ver una referencia completa de posibles acciones a las que unir funciones, podéis ver la referencia oficial: http://codex.wordpress.org/Plugin_API/Action_Reference. En nuestro caso nos interesa la acción “wp_head”, que tal como podéis ver en la referencia, se ejecuta cuando el tema llama a la función que crea la cabecera, también llamada “wp_head”.

Por lo tanto, el código completo para nuestro caso será:

```
<?php
function enlace_favicon () {
    echo '<link rel="shortcut icon" type="image/x-icon" href="
http://codex.wordpress.org/favicon.ico" />'. "\n";
```

```
}  
  
add_action('wp_head', 'enlace_favicon');  
  
?>
```

Tras salvar el fichero y refrescar la página, comprobaréis que el icono del sitio ha cambiado y es ahora el de WordPress.

Tened en cuenta que, al cargar ambos ficheros, no podréis usar funciones que compartan nombre a no ser que la función esté definida por el padre como *pluggable* (que se puede enchufar, traducido al español), y por lo tanto, podréis modificar el funcionamiento de la misma. Para saber si una función puede ser sobrescrita siguiendo este método, sólo deberemos ver si en su declaración hace la comprobación oportuna: *if(!function_exists('nombre_funcion'))*. La secuencia de ejecución sería la siguiente:

1. Se carga la función del hijo.
2. La función del padre comprueba si existe la función ya, en caso afirmativo no se declara, en caso negativo, se declara.

De esta forma se consigue mejorar la compatibilidad, sin sacrificar la funcionalidad y sin emborronar demasiado el código.

7.2.4 Otros ficheros

Los demás ficheros del tema también pueden ser modificados desde el tema hijo sin modificar el tema padre y preservándonos de los problemas relacionados con las actualizaciones.

El criterio seguido por los creadores de WordPress es que si creamos un fichero en el tema hijo que ya existe en el tema padre, se invocará al del tema hijo ignorando al del tema padre (se sobrescribirá de igual forma que ocurre con el fichero de la hoja de estilos). De esta forma, nuestras modificaciones siempre tendrán más relevancia que las posibles actualizaciones que sufra el tema padre.

Así mismo, también podremos incluir cualquier tipo de fichero que creamos oportuno, desde imágenes o sonidos a librerías de JavaScript, siguiendo los criterios de cualquier otro diseño web.

7.3 DESCRIPCIÓN DE UN TEMA

Tras aprender a desarrollar un tema hijo, es en el momento de estudiar la estructura interna de un tema con algo más de detalle. La referencia será el tema que estamos usando desde el comienzo de este capítulo, Twenty Ten. Veremos uno a uno los ficheros del mismo y resaltaremos los detalles de más importancia.

Sólo hablaremos de los ficheros con código PHP y podréis ver las llamadas a funciones de la API de WordPress más prácticas y usuales, de esta forma podréis llegar a basaros en este tema o directamente en una plantilla HTML para desarrollar el vuestro a medida.

Despreciaremos los comentarios y, tras mostrar el código fuente de cada fichero, explicaremos una a una las llamadas a funciones que aparezcan por primera vez. En el caso de ficheros muy extensos, nos referiremos sólo a las partes más importantes del mismo. Explicaremos primero el fichero índice y, posteriormente, seguiremos un orden alfabético, ignorando los ficheros menos relevantes.

7.3.1 index.php

Como en cualquier diseño web, el índice constituye el equivalente (salvando diferencias) a lo que podría ser la función *main* en un programa en C. A raíz de él comienzan las llamadas al resto de ficheros y funciones, es el primer fichero que se llama cuando se ejecuta el tema.

7.3.1.1 CÓDIGO FUENTE

```
<?php get_header(); ?>

<div id="container">

    <div id="content" role="main">

        <?php get_template_part( 'loop', 'index' );?>

    </div><!-- #content -->

</div><!-- #container -->

<?php get_sidebar(); ?>

<?php get_footer(); ?>
```

7.3.1.1.1 `get_header()`

Incluye el contenido del fichero *header.php* de nuestro tema. También puede ser llamada pasándole una cadena por parámetro: *get_header(\$nombre)*, en el caso de que tengamos que distinguir entre varias cabeceras. En tal caso, llamará a los ficheros *header-nombre.php*.

7.3.1.1.2 `get_template_part($slug, $nombre)`

Incluye un fichero diferente a la cabecera, barra lateral o pie de página (estos ficheros tienen funciones específicas para incluirse). Las inclusiones se hacen usando la función *require()* de PHP.

En este caso concreto incluirá las siguientes funciones en el siguiente orden:

- *wp-content/themes/twentyten/loop-index.php* (no existe)
- *wp-content/themes/twentyten/loop.php*

Esta función también está pensada para trabajar con temas hijos, por lo que, en caso de que así fuera, la secuencia de carga de las llamadas sería la siguiente:

- *wp-content/themes/twentyten-hijo/loop-index.php*
- *wp-content/themes/twentyten-hijo/loop.php*
- *wp-content/themes/twentyten/loop-index.php*
- *wp-content/themes/twentyten/loop.php*

7.3.1.1.3 `get_sidebar()`

Comportamiento análogo a *get_header()*, pero en este caso para la carga de la barra lateral.

7.3.1.1.4 `get_footer()`

Ídem que el anterior, pero para el pie de página.

7.3.2 404.php

El fichero para mostrar errores de búsquedas no encontradas se limitará a hacer la carga básica del tema (cabecera y pie de página), a la vez que incluir la estructura de una entrada, donde mostrar el error y proporcionar al usuario un formulario de búsqueda.

7.3.2.1 CÓDIGO FUENTE

```
<?php get_header(); ?>

<div id="container">

    <div id="content" role="main">

        <div id="post-0" class="post error404 not-found">

            <h1 class="entry-title"><?php _e( 'Not Found',
'twentyten' ); ?></h1>

            <div class="entry-content">

                <p><?php _e( 'Apologies, but the page
you requested could not be found. Perhaps searching will help.', 'twentyten' );
?></p>

                <?php get_search_form(); ?>

            </div><!-- .entry-content -->

        </div><!-- #post-0 -->

    </div><!-- #content -->

</div><!-- #container -->

<script type="text/javascript">

    document.getElementById('s')&&
document.getElementById('s').focus();

</script>

<?php get_footer(); ?>
```

7.3.2.1.1 `php_e($texto, $dominio)`

Devuelve la traducción para el lenguaje actual dentro del entorno definido por el dominio. En nuestro caso, estaríamos hablando del texto “Not found”, que traducido al español sería “No encontrado” si el español está disponible dentro de las traducciones del tema, de las que hemos hablado antes. La misma función es llamada en un punto posterior.

7.3.2.1.2 `get_search_form()`

En caso de que tengamos un fichero *searchform.php* en nuestro tema, se cargará ese fichero, en caso contrario, como nos ocurre con Twenty Ten, WordPress generará un formulario de búsqueda que incluye por defecto. Se hace foco sobre el mismo directamente con JavaScript tras la carga del formulario.

7.3.3 `archive.php`

El fichero que muestra el archivo de entradas. Su funcionamiento se basa en el famoso bucle de WordPress, que muestra en este caso las entradas para los días, meses o años en función de lo que haya pedido el usuario. Sólo mostraremos el código fuente de más importancia en el siguiente punto.

7.3.3.1 CÓDIGO FUENTE

```
<?php

if ( have_posts() )

    the_post();

?>

<h1 class="page-title">

<?php if ( is_day() ) : ?>

    <?php printf( __( 'Daily Archives: <span>%s</span>', 'twentyten'
), get_the_date() ); ?>

<?php elseif ( is_month() ) : ?>

    <?php printf( __( 'Monthly Archives: <span>%s</span>',
'twentyten' ), get_the_date('F Y') ); ?>
```

```
<?php elseif ( is_year() ) : ?>

    <?php printf( __( 'Yearly Archives: <span>%s</span>',
'twentyten' ), get_the_date('Y') ); ?>

<?php else : ?>

    <?php _e( 'Blog Archives', 'twentyten' ); ?>

<?php endif; ?>

</h1>

<?php

rewind_posts();

get_template_part( 'loop', 'archive' );

?>
```

7.3.3.1.1 have_posts()

Comprueba el siguiente ítem en la colección de entradas: si hay algún otro ítem, retorna verdadero; en caso contrario, retorna falso. Es uno de los mecanismos básicos para el funcionamiento del bucle que muestra entradas en el blog.

7.3.3.1.2 the_post()

La función toma el elemento actual en la colección de entradas y lo hace disponible para el uso dentro de esta iteración del bucle. Una vez que los datos de la entrada están disponibles, la plantilla puede comenzar a mostrar los datos de dicha entrada al usuario. Con la función anterior, son dos de los elementos más importantes del bucle de WordPress, base del funcionamiento del mismo para mostrar una serie de entradas en pantalla en función a los criterios establecidos (en este caso el de un archivo, en otros pueden ser el caso por defecto, por categorías, etiquetas...).

7.3.3.1.3 is_day()

Devuelve verdadero si se trata de un archivo de tipo diario, en caso contrario, devuelve falso. Muestra el título oportuno con la función de PHP *printf*.

7.3.3.1.4 is_month()

Ídem para archivo mensual.

7.3.3.1.5 is_year()

Ídem para un archivo anual.

7.3.3.1.6 get_the_date()

Muestra la fecha en la que la entrada fue escrita. Puede pasarsele un parámetro con el formato de la fecha.

7.3.3.1.7 rewind_posts()

Rebobina el bucle y permite hacer uno nuevo.

7.3.4 author.php

Este fichero muestra primero la información básica de un autor de blog y, posteriormente, hace un archivo (idéntico al anterior), pero de entradas del autor en cuestión, en lugar de ser por criterios de fecha. La estructura de la segunda parte es la misma, sólo cambia la llamada a `get_template_part( 'loop', 'author' )`; que, como veis, hace referencia a un archivo del autor en cuestión.

Nos limitaremos a comentar el código fuente y las funciones que muestran los datos del autor.

7.3.4.1 CÓDIGO FUENTE

```

<?php

if ( get_the_author_meta( 'description' ) ) : ?>

<div id="entry-author-info">

<div id="author-avatar">

<?php echo get_avatar( get_the_author_meta(
'user_email' ), apply_filters( 'twentyten_author_bio_avatar_size', 60 ) ); ?>

</div><!-- #author-avatar -->

<div id="author-description">

```

```

        <h2><?php printf( __( 'About %s', 'twentyten' ),
get_the_author() ); ?></h2>

        <?php the_author_meta( 'description' ); ?>

    </div><!-- #author-description -->

</div><!-- #entry-author-info -->

<?php endif; ?>

```

7.3.4.1.1 get_the_author_meta(\$campo)

Muestra metainformación del usuario. Si tiene sólo un parámetro, muestra el del usuario cargado, en caso de tener dos parámetros, podemos pasarle el usuario en cuestión a cargar, a través de su identificador.

La lista completa de campos que podemos consultar con esta función está disponible en el sitio web de la referencia de la misma dentro del Codex: http://codex.wordpress.org/Function_Reference/the_author_meta.

7.3.4.1.2 get_avatar(\$identificador)

Obtiene la imagen identificativa de un usuario o autor, el llamado *avatar* o *gravatar*. La referencia puede ser tanto el identificador de usuario dentro del sistema como el correo electrónico del mismo. Además del parámetro obligatorio, también podemos pasarle parámetros opcionales:

- Tamaño.
- URL de la imagen por defecto.
- Texto alternativo por si no carga la imagen (atributo “alt” de la etiqueta “img” de HTML).

7.3.4.1.3 apply_filters(\$etiqueta, \$valor)

Ésta es una de las funciones más importantes de toda la API de WordPress. Para comprender su funcionamiento, debemos conocer previamente cómo funcionan los filtros. A través de la función *add_filter()*, podemos ir agregando una lista de funciones a cada etiqueta, de forma que, al aplicar un filtro para dicha etiqueta, se ejecuten todas las funciones contra ese valor y finalmente se devuelva.

Veamos un ejemplo. Por defecto, en este CMS, para la etiqueta “the_content”, se aplican las siguientes funciones:

- *wptexturize()*
- *convert_smilies()*
- *convert_chars()*
- *wpautop()*
- *shortcode_unautop()*
- *prepend_attachment()*
- *do_shortcode()*

El resultado de una es la entrada de la otra y al hacer la llamada *apply_filters* para la etiqueta en cuestión, en este caso “the_content”, se ejecutarán todas ellas. De esta forma, podemos crear una serie de filtros que nos permitan automatizar secuencias necesarias para ciertos valores.

7.3.4.1.4 **get_the_author()**

Devuelve el autor de la entrada que estamos tratando en el bucle.

7.3.4.1.5 **the_author_meta(\$campo, \$identificador)**

Hace exactamente lo mismo que la función *get_the_author_meta()*, pero, a diferencia de ésta, no sólo retorna el valor, sino que permite mostrarlo directamente por pantalla (es decir, con sus etiquetas, clases, etc.).

7.3.5 **category.php**

Muestra un archivo para una categoría concreta. Su estructura es análoga a los ficheros que hacen archivo de otro tipo de información.

7.3.5.1 **CÓDIGO FUENTE**

```
<h1 class="page-title">
```

```
<?php printf( __( 'Category Archives: %s', 'twentyten' ), '<span>' .  
single_cat_title( '', false ) . '</span>' );?>
```

```
</h1>
```

```
<?php

$category_description = category_description();

if ( ! empty( $category_description ) )

    echo '<div class="archive-meta">' . $category_description .
'</div>';

get_template_part( 'loop', 'category' );

?>
```

7.3.5.1.1 single_cat_title()

Muestra el título de la categoría de la página actual. Podemos pasarle dos parámetros opcionales:

- Prefijo para mostrar delante del título.
- Elegir si se debe devolver el valor formateado para mostrarlo directamente (opción por defecto) o sólo la cadena del valor.

7.3.5.1.2 category_description()

Devuelve la descripción de la categoría actual. Podemos pasarle un parámetro indicándole explícitamente el identificador de la categoría.

En nuestro caso concreto, el valor devuelto se guarda en una variable y se comprueba posteriormente si está vacío, con la función *empty* de PHP.

7.3.6 comments.php

Muestra tanto los comentarios de una entrada como el formulario para realizar un nuevo comentario. El listado de comentarios se lleva a cabo gracias a la ejecución de una función presente en *functions.php* llamada *twentyten_comment()*.

7.3.6.1 CÓDIGO FUENTE

No mostramos el código fuente debido a la longitud del mismo y al parecido con ficheros anteriores. Pasamos directamente a comentar las funciones de interés que acaban de aparecer.

7.3.6.1.1 post_password_required()

Devuelve verdadero o falso en función de si la entrada requiere o no contraseña.

7.3.6.1.2 have_comments()

Devuelve verdadero o falso en función de si la entrada tiene o no comentarios.

7.3.6.1.3 have_comments()

Devuelve verdadero o falso en función de si la entrada tiene o no comentarios.

7.3.6.1.4 get_comments_number()

Devuelve el número de comentarios de la entrada.

7.3.6.1.5 get_the_title()

Devuelve el título de la entrada.

7.3.6.1.6 get_comment_pages_count()

Calcula el número total de páginas de comentarios.

7.3.6.1.7 get_option(\$opcion)

Hace una consulta en la tabla de opciones y devuelve la opción que se ha pasado por parámetro. Podemos pasarle un segundo parámetro para que, en caso de que no exista dentro de la tabla dicha opción, lo muestre.

7.3.6.1.8 previous_comments_link()

Devuelve el enlace al comentario anterior.

7.3.6.1.9 next_comments_link()

Devuelve el enlace al comentario siguiente.

7.3.6.1.10 wp_list_comments(\$opciones)

Muestra la lista de comentarios en función del vector de opciones que se le pasa por parámetros. En el caso de Twenty Ten, el vector se forma en función de una retrollamada a la función *twentyten_comment* de la que hemos hablado antes.

7.3.6.1.11 `comments_open()`

Devuelve verdadero o falso en función de si los comentarios están abiertos.

7.3.6.1.12 `comment_form()`

Muestra el formulario de comentarios.

7.3.7 footer.php

Además de mostrar el pie de página, cierra todas las etiquetas para hacer válido el documento HTML generado y permite la inclusión de una barra de *widgets* en el pie si es necesario.

7.3.7.1 CÓDIGO FUENTE

```
<?php get_sidebar( 'footer' );?>

<div id="site-info">

    <a href="<?php echo home_url( '/' ) ?>" title="<?php echo esc_attr(
get_bloginfo( 'name', 'display' ) ); ?>" rel="home">

        <?php bloginfo( 'name' ); ?>

    </a>

</div><!-- #site-info -->

<div id="site-generator">

    <?php do_action( 'twentyten_credits' ); ?>

    <a href="<?php echo esc_url( __( 'http://wordpress.org/', 'twentyten' ) );
?>"

        title="<?php esc_attr_e( 'Semantic Personal Publishing Platform',
'twentyten' ); ?>" rel="generator">

        <?php printf( __( 'Proudly powered by %s.', 'twentyten' ),
'WordPress' ); ?>

    </a>
```

```
</div><!-- #site-generator -->
<!-- código recortado [...] -->
<?php wp_footer(); ?>
```

7.3.7.1.1 home_url()

Devuelve la URL inicial del blog. Se le pueden pasar dos parámetros:

- Camino relativo al inicio.
- Esquema para dar la URL (básicamente, HTTP o HTTPS).

7.3.7.1.2 esc_attr(\$texto)

Sustituye los caracteres problemáticos de cara a la seguridad y a ser mostrados dentro de textos en HTML, por sus correspondientes equivalentes en HTML. Los símbolos que trata son más concretamente: menor que, mayor que, *ampersand*, comillas simples y comillas dobles.

7.3.7.1.3 get_bloginfo()

Devuelve información del blog. Se le pueden pasar dos parámetros:

- Información a mostrar. Si no se incluye, muestra el título por defecto.
- Filtro a aplicar.

7.3.7.1.4 do_action(\$etiqueta)

Ejecuta la acción predefinida para la etiqueta en cuestión. Estas acciones están predefinidas gracias a la función *add_action()*, que hemos explicado antes. También se le pueden pasar más parámetros si la acción lo requiere.

7.3.7.1.5 esc_url(\$url)

Elimina posibles problemas de seguridad de la URL.

7.3.7.1.6 esc_attr_e(\$texto)

Mismo comportamiento que *esc_attr()*, pero, en este caso, prepara el valor para ser mostrado directamente.

7.3.8 header.php

Como su nombre indica, es el fichero encargado de mostrar toda la cabecera del sitio para cada situación. No sólo muestra la parte de la cabecera HTML (encerrada por las etiquetas *head*), sino que muestra el comienzo del cuerpo del fichero justo hasta el comienzo del contenedor principal.

Sólo comentaremos la parte del código fuente de la cabecera HTML, de la parte correspondiente al cuerpo, y explicaremos las funciones de relevancia, ya que, en esta segunda parte, hay más código de maquetación del tema que de funcionamiento en sí del mismo.

7.3.8.1 CÓDIGO FUENTE CABECERA HTML

```
<html <?php language_attributes(); ?>>

<head>

<meta charset="<?php bloginfo( 'charset' ); ?>" />

<title><?php

    global $page, $paged;

    wp_title( '|', true, 'right' );

    bloginfo( 'name' );

    $site_description = get_bloginfo( 'description', 'display' );

    if ( $site_description && ( is_home() || is_front_page() ) )

        echo " | $site_description";

    if ( $paged >= 2 || $page >= 2 )

        echo ' | '. sprintf( __( 'Page %s', 'twentyten' ), max( $paged, $page

    ) );

?></title>

<link rel="profile" href="http://gmpg.org/xfn/11" />
```

```
<link rel="stylesheet" type="text/css" media="all" href="<?php bloginfo(
'style_sheet_url'); ?>" />
```

```
<link rel="pingback" href="<?php bloginfo( 'pingback_url' ); ?>" />
```

```
<?php wp_head(); ?>
```

```
</head>
```

7.3.8.1.1 language_attributes()

Devuelve los atributos de lenguaje del documento (codificación, idioma, dirección de escritura).

7.3.8.1.2 wp_title()

Devuelve el título para la página. Éste variará en función de qué sección, categoría, tipo de archivo, autor... estemos visualizando. Permite el uso opcional de tres atributos:

- Un texto a mostrar en alguna parte del título (para nuestro caso, el carácter '|').
- Si debe emitirse el texto (mediante un *echo*) o sólo devolverse.
- Localización del texto del primer parámetro dentro del título.

7.3.8.1.3 is_home()

Devuelve verdadero o falso en función de si está visualizando o no la página de inicio.

7.3.8.1.4 wp_head()

Ejecuta la acción "wp_head". Es importante ponerlo justo antes de finalizar la etiqueta *head*, porque muchos complementos incluyen acciones a realizar en ese punto y ligan inclusión de las mismas a dicha acción.

7.3.8.1.5 body_class()

Devuelve las clases para el cuerpo HTML, de cara a la maquetación con CSS, en función de lo que se esté visualizando.

7.3.8.1.6 `is_singular()`

Devuelve verdadero o falso en función de si estamos viendo o no una página singular. Una página es considerada singular en alguno de los siguientes casos: cuando se visualiza una sola entrada, cuando se está viendo una página o cuando se está viendo un adjunto.

7.3.8.1.7 `has_post_thumbnail()`

Devuelve verdadero o falso en función de si la entrada tiene o no una miniatura asociada. Podemos asignar imágenes a las entradas de forma que tengamos una imagen destacada para cada entrada.

7.3.8.1.8 `get_the_post_thumbnail()`

Devuelve la imagen destacada. Existen funciones similares, como `get_the_post_thumbnail_id`, que permiten buscar la imagen destacada para una entrada en función de su identificador.

7.3.8.1.9 `wp_nav_menu()`

Muestra un menú de navegación, de los que se crean directamente en el panel de administración de WordPress. Para personalizar la llamada, podemos pasar un vector de opciones a la función.

7.3.9 `loop.php`

El *loop* o bucle es la iteración que da vida al blog y que permite mostrar una serie de entradas en función de unos requisitos de búsqueda. Es una de las bases sobre las que se soporta WordPress. Dejando de lado el fichero de funciones propias del tema, que no nos interesa en este punto porque depende muy directamente del tema en sí, éste es el fichero más extenso de toda la plantilla.

Su funcionamiento se basa en una iteración, que va recorriendo una colección de entradas y mostrando para cada una de ellas los datos que dan vida al blog: título, fecha, autor, contenido, enlace a comentarios... La colección se rellena mediante una búsqueda que tiene en cuenta qué está buscando el usuario en cada caso.

7.3.9.1 CÓDIGO FUENTE

Debido a la extensión del código fuente de este fichero, nos limitaremos a hablar de las funciones más importantes que aparecen en él. Gran parte del fichero es maquetación HTML y mecanismos de control para tratar con la iteración.

7.3.9.1.1 `next_posts_link()`

Proporciona el enlace a las siguientes entradas.

7.3.9.1.2 `previous_posts_link()`

Proporciona el enlace a las entradas previas.

7.3.9.1.3 `in_category($categoría)`

Devuelve verdadero o falso en función de si la entrada se encuentra o no en la categoría o las categorías indicadas. La categoría se pasa como parámetro a través de un vector. También permite un segundo parámetro para explicitar a qué entrada en cuestión nos referimos.

7.3.9.1.4 `the_ID()`

Devuelve el identificador de la entrada actual.

7.3.9.1.5 `post_class()`

Devuelve las clases para la maquetación CSS para la entrada actual.

7.3.9.1.6 `the_permalink()`

Devuelve el enlace permanente para la entrada actual.

7.3.9.1.7 `the_excerpt()`

Devuelve el extracto de la entrada actual.

7.3.9.1.8 `the_content()`

Devuelve el contenido de la entrada actual. Podemos hacer uso de tres parámetros opcionales diferentes:

- El texto a mostrar cuando se muestre el enlace a “Leer más”.
- Mostrar u ocultar el texto de “Leer más”.
- URL a la que apunta el enlace de “Leer más”.

7.3.9.1.9 `is_search()`

Devuelve verdadero o falso en función de si estamos o no en los resultados de una búsqueda.

7.3.9.1.10 `is_archive()`

Devuelve verdadero o falso en función de si estamos o no en un archivo de entradas.

7.3.9.1.11 `get_the_category()`

Devuelve la categoría de la entrada actual.

7.3.9.1.12 `get_the_tag_list()`

Devuelve la lista de etiquetas de la entrada. Podemos pasarle hasta tres parámetros opcionales:

- Texto previo a las etiquetas.
- Cadena para separar las etiquetas.
- Texto tras las etiquetas.

7.3.9.1.13 `comments_template()`

Carga la plantilla por defecto para hacer comentarios. Permite dos argumentos opcionales:

- El fichero a cargar, por defecto, *comments.php* de nuestro tema.
- Si se han de separar o no los comentarios en función de su tipo.

7.3.9.1.14 `edit_post_link()`

Proporciona el enlace para editar la entrada, se muestra cuando el usuario es administrador del sitio o autor de la misma. Permite los siguientes parámetros opcionales:

- Texto del enlace.
- Texto a mostrar antes del enlace.
- Texto a mostrar tras el enlace.
- Identificador de la entrada.

7.3.9.1.15 wp_link_pages()

Muestra enlaces para entradas paginadas. Se le pasan las opciones a través de un parámetro en forma de vector.

7.3.10 onecolumn-page.php

Plantilla para mostrar una página sin barras laterales, en una sola columna. Mostraremos el código fuente, pero no comentaremos ninguna función, porque todas las que aparecen han sido comentadas en puntos anteriores de este mismo capítulo.

7.3.10.1 CÓDIGO FUENTE

```
<?php get_header(); ?>

<div id="container" class="one-column">

<div id="content" role="main">

<?php if ( have_posts() ) while ( have_posts() ) : the_post(); ?>

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

        <h1 class="entry-title"><?php the_title(); ?></h1>

        <div class="entry-content">

            <?php the_content(); ?>

            <?php wp_link_pages( array( 'before' => '<div class="page-
link">'. __( 'Pages:', 'twentyten' ), 'after' => '</div>' ) ); ?>

            <?php edit_post_link( __( 'Edit', 'twentyten' ), '<span class="edit-
link">'. '</span>' ); ?>

        </div><!-- .entry-content -->

    </div><!-- #post-## -->

    <?php comments_template( "", true ); ?>

<?php endwhile; ?>
```

```
</div><!-- #content -->

</div><!-- #container -->

<?php get_footer(); ?>
```

7.3.11 page.php

Plantilla para mostrar una página. No mostramos el código fuente porque es el mismo que el anterior a excepción de la inclusión de una llamada a incluir la barra lateral.

7.3.12 search.php

Plantilla para mostrar una página de resultados de una búsqueda. Su código es prácticamente el mismo que el del fichero *404.php*, pero incluyendo la llamada al bucle que muestra los resultados de la búsqueda. Por esta razón, tampoco mostramos el código fuente.

7.3.13 sidebar.php

Barra lateral que contiene, en este caso, la zona primaria y secundaria de *widgets*. El código fuente se divide en estas dos partes:

- La zona primaria, que incluye soporte por si falla la llamada a rellenar los *widgets* y muestra tres *widgets* por defecto: el formulario de búsqueda, el archivo y el llamado “Meta”, para registrarse o iniciar sesión.
- La zona secundaria, igual que la anterior, pero sin el soporte ante fallos.

7.3.13.1 CÓDIGO FUENTE

```
<div id="primary" class="widget-area" role="complementary">

<ul class="xoxo">

<?php

if ( ! dynamic_sidebar( 'primary-widget-area' ) ) : ?>
```

```

<?php endif; // end primary widget area ?>

<!-- zona preparada por si falla la llamada a los widgets -->

</div><!-- #primary .widget-area -->

<?php

if ( is_active_sidebar( 'secondary-widget-area' ) ) : ?>

<div id="secondary" class="widget-area" role="complementary">

    <ul class="xoxo">

        <?php dynamic_sidebar( 'secondary-widget-area' ); ?>

    </ul>

</div><!-- #secondary .widget-area -->

<?php endif; ?>

```

7.3.13.1.1 dynamic_sidebar()

Llama a todos los *widgets* activos para la zona de *widgets* identificada por el parámetro que se le pasa a la función. La zona por defecto si no se indica nada es 1 y en caso de fallo en la carga de los *widgets* devuelve falso.

7.3.13.1.2 is_active_sidebar()

Devuelve verdadero o falso en caso de que esté activada o no la barra lateral identificada por el parámetro que se le pasa a la función.

7.3.13.1.3 wp_register()

La llamada devuelve el enlace a registrarse, en caso de que el usuario no esté registrado, o un enlace a la administración en caso de que si lo esté.

7.3.13.1.4 wp_loginout()

La llamada devuelve un enlace a iniciar sesión, si el usuario no la ha iniciado, o a cerrar sesión, si ya ha iniciado sesión.

7.3.14 sidebar-footer.php

Fichero encargado de mostrar las zonas del pie de página que sirven para mostrar *widgets*. En este caso hay presentes cuatro posibles zonas para activar. Su código es similar al de *sidebar.php*, con las modificaciones pertinentes en cuanto maquetación.

7.3.15 single.php

Plantilla para mostrar una entrada en concreto. Se usa cuando estamos visualizando una entrada del sitio, en lugar de varias a través del bucle. Su código es justamente una de las partes de *loop.php*, la dedicada a mostrar la información de cada entrada, sin incluir aspectos relacionados con el control del bucle.

7.3.16 tag.php

Archivo para una etiqueta. Código análogo a *author.php* o *category.php*, pero haciendo el bucle de salida orientado una etiqueta en concreto, en lugar de una categoría o un autor. La llamada al *loop* en esta ocasión es la siguiente: `get_template_part('loop', 'tag')`.

7.4 CONCLUSIONES

Terminamos el primero de los dos capítulos más técnicos del libro. Hemos visto las bases sobre las que se asientan los temas en WordPress, cómo podemos basarnos en un tema para crear otro relacionado, para, finalmente, ver y estudiar el código fuente de la referencia en temas para WordPress, Twenty Ten.

A pesar de la dificultad intrínseca de este tipo de desarrollos, sabiendo las bases de cualquier maquetación web, HTML y CSS, podemos llegar a convertirnos en desarrolladores de temas para WordPress sin tener que dedicar horas al estudio de cómo funciona el sistema. La opción de los temas hijos es el primer paso para desarrollar un tema desde cero y, sin lugar a dudas, a diferencia de otros sistemas gestores de contenido, en este caso la curva de aprendizaje es bastante progresiva y cualquiera, incluso sin conocer el lenguaje PHP en profundidad, podrá abordar la problemática.

Antes de seguir con el próximo capítulo, os recuerdo que debéis estar atentos a las licencias de los temas. Como cualquier otro tipo de software, cada tema tendrá su propia licencia, que podréis consultar en el fichero *license.txt* que traen en su raíz. Muchos de ellos están liberados con licencia GPL en cualquiera de

sus versiones. La GPL (*General Public License* “Licencia Pública General”) es una licencia vírica, por lo tanto, si nos basamos en un tema de WordPress liberado bajo GPL, tenemos la obligación de liberarlo con la misma licencia. Estad atentos a este tema porque si usamos software libre, con todas sus ventajas, debemos también seguir sus reglas para que terceros puedan aprovechar y mejorar nuestro trabajo.

Para finalizar, hay que comentar que, aunque en el punto anterior se han descrito buena parte de las funciones más importantes de la API de WordPress, si queréis tener una descripción exacta de todas ellas, siempre deberíais consultar la referencia oficial del CMS: http://codex.wordpress.org/Main_Page.

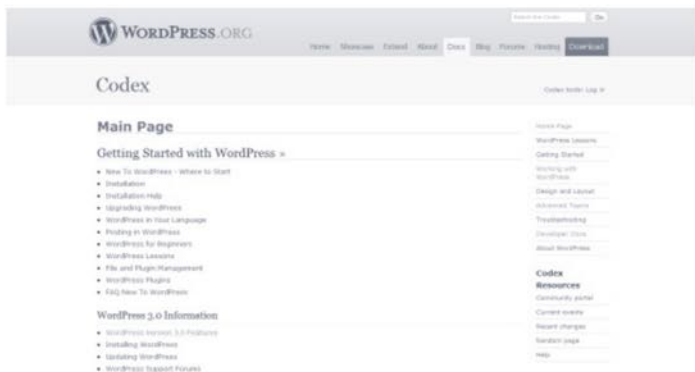


Figura 7.6. Aspecto del sitio web de la referencia oficial de WordPress

CREACIÓN DE PLUGINS

Saber modificar temas es básico para poder mejorar y adaptar el aspecto estético de nuestro blog a nuestras necesidades, sin embargo, a través de los temas, la funcionalidad no puede ser personalizada ni ampliada tanto como querríamos. Desde el primer capítulo hemos hablado de los *plugins* o complementos para solventar esta problemática. Ahora nos toca el turno de estudiarlos en profundidad y llegar a ser nosotros quienes aportemos nuevos *plugins*.

Pero, ¿qué es un *plugin*? Aunque en capítulos anteriores lo hemos descrito de una forma informal, ahora nos toca dar una descripción más exacta del mismo. Un *plugin* de WordPress es un programa o un conjunto de una o más funciones, escritas en lenguajes PHP, que añade un paquete específico de funcionalidades o servicios a un blog implementado con WordPress. Además, se integrará con el sistema raíz sin tocarlo, gracias a los mecanismos que nos proporciona la API del CMS.

8.1 NOMBRES, FICHEROS Y RUTAS

Empezamos el capítulo metiéndonos de lleno en lo que nos ocupa. Antes de comenzar a trabajar sobre el código, debemos tener en cuenta una serie de consideraciones.

8.1.1 Nombres y motivación

Lo primero que debemos hacer a la hora de crear un *plugin* es pensar en su nombre y en qué hará. Hay que tener en cuenta que, al igual que en todo el mundo de la informática, lo último que debemos hacer es reinventar la rueda. Sin embargo, dado el carácter introductorio y divulgativo de este texto, es muy probable que la funcionalidad que desarrollemos esté cubierta por otro complemento.

Si queremos subir el *plugin* al repositorio oficial, deberemos buscar un nombre que sea único, es decir, que no esté ya en uso en el repositorio. El nombre es una cadena, por lo que podéis formarla a base de una o tantas palabras como queráis. Lo más frecuente es que el nombre sea identificativo de la funcionalidad del *plugin*.

8.1.2 Ficheros

Los *plugins* están desarrollados en lenguaje PHP, por lo que necesitaremos tener uno o varios ficheros de este tipo. Si tenemos un solo fichero, podemos colocarlo directamente en el directorio oportuno “wp-content/plugins/fichero.php”. Ya que los *plugins* se instalan en ese directorio, es conveniente que busquemos también un nombre para el fichero que sea, a la vez que identificativo, lo suficientemente particular como para que no se repita. Cuando tenemos más de un fichero, deberemos crear un directorio ex profeso para él, el nombre del fichero debe seguir las mismas reglas que acabamos de explicar.

No es obligatorio que el nombre de fichero o directorio sea el mismo que el del complemento. De hecho, no es ni recomendable, porque todos sabréis los problemas de las URL que contienen símbolos especiales como espacios o letras acentuadas. Lo que sí deben es estar relacionados.

Así mismo, si os animáis a hacer público el *plugin* al repositorio oficial de WordPress, deberéis crear un fichero *readme.txt* siguiendo el formato que se muestra en el siguiente sitio: <http://wordpress.org/extend/plugins/about/readme.txt>, así como, si tenéis posibilidad, crear un sitio web donde presentéis la labor del complemento al público (aparte de la descripción que hagáis en el repositorio oficial).

8.1.3 Cabeceras

Al igual que ocurre con los temas, los *plugins* necesitan presentarse al sistema a través de una información que se encuentra en las cabeceras de los ficheros. En el caso de los temas, recordaréis que dichos datos se encontraban en la

hoja de estilos en cascada y que estaba encuadrada en forma de comentarios. Para los complementos, el método es el mismo, pero en lugar de ser una hoja de estilos, se trata de un fichero PHP.

El fichero será el principal de nuestro complemento y el formato será el siguiente:

```
<?php
```

```
/*
```

Plugin Name: Nombre del plugin

Plugin URI: Dirección web del plugin

Description: Breve descripción de lo que hace el plugin

Version: Número de version actual

Author: Autor del plugin

Author URI: Dirección web del autor del plugin

License: Licencia bajo la que está liberado el plugin

```
*/
```

```
?>
```

De todos los datos, el único absolutamente necesario es el primero, el nombre del *plugin*, el resto son opcionales, aunque es conveniente introducirlos. El orden en que se introducen estos datos opcionales no es relevante.

8.2 ANTES DE COMENZAR A PROGRAMAR

Tenemos las bases para saber con qué estamos tratando y estamos a punto de meternos con el código, pero antes de empezar a programar debemos conocer varios aspectos más de cómo interactúan y se implementan los complementos en WordPress.

8.2.1 Interacción

Antes de nada, debemos saber qué dos métodos básicos hay para hacer llamadas a la funcionalidad de los complementos.

8.2.1.1 GANCHOS

Cuando hablábamos del fichero de funciones en el tema anterior, se mencionó el término *hook* o gancho. ¿Qué es exactamente? Como dijimos en su momento, podemos asignar funciones a una serie de ganchos que trae incorporados WordPress, de forma que en ciertos momentos de la ejecución se hagan llamadas a dichas funciones. Hicimos un ejemplo, en el que incluíamos el *favicon* en la cabecera, para ello implementamos una función que hacía el eco correspondiente y lo asignamos al gancho de la cabecera, *wp_head*. De igual forma, podemos hacer que nuestro *plugin* se ejecute en alguno de estos *hooks*.

8.2.1.2 ETIQUETAS DE PLANTILLA

Las etiquetas de plantilla o *template tags* son la otra opción para hacer llamadas a nuestros complementos. La idea es la de crear una función PHP que pueda ser llamada desde el tema o *widget*, de forma que en dicho momento se ejecute el código del complemento.

Las etiquetas de plantilla las hemos usado en el capítulo anterior de forma intensiva, de hecho, cuando hacíamos llamadas del tipo *the_title()*, nos estábamos refiriendo justo a eso. En caso de elegir este método, debemos dejar muy claro en la documentación del *plugin* cómo se hace la llamada, con qué parámetros, qué significa cada uno de ellos, además de cuáles son opcionales y cuáles no.

8.2.2 Acceso a datos

Dependiendo de la funcionalidad de nuestro complemento, deberemos tener acceso a determinada información de la base de datos o almacenar información del mismo. Tenemos tres mecanismos que podremos usar en función de nuestras necesidades.

8.2.2.1 META INFORMACIÓN DE LAS ENTRADAS

La opción más simple de todas, y que sirve sólo para gestionar información con referencia a las entradas, es el uso de la tabla *wp_postmeta*. Dentro de ella, podremos almacenar nuevos datos relacionados con las entradas de una forma directa gracias a la existencia de las funciones:

- `add_post_meta()`
- `delete_post_meta()`
- `update_post_meta()`
- `get_post_meta()`

Sirven, respectivamente, para añadir, borrar, actualizar u obtener información de dicha tabla de forma directa y haciendo referencia a una entrada en concreto. Es una opción limitada, pero muy práctica si nuestro complemento está relacionado de forma directa con las entradas.



wp_postmeta	
meta_id	BIGINT(20)
post_id	BIGINT(20)
meta_key	VARCHAR(255)
meta_value	LONGTEXT
Indexes	

Figura 8.1. Tabla `wp_postmeta`

8.2.2.2 OPCIONES

Este segundo método es también muy directo y hace uso de la tabla `wp_options` de WordPress. Se basa en el uso de una serie de funciones que acceden a dicha tabla para guardar pares de datos del tipo: nombre, valor...

Las funciones en cuestión son:

- `add_option()`
- `get_option()`
- `update_option()`

De esta forma podremos añadir nuevos pares, leer cualquier opción del sistema o actualizarlos. Es muy útil para casos en los que tratamos con información muy básica y que no tiene relación con otras tablas.

8.2.2.3 ACCESO DIRECTO A BASE DE DATOS

La última y más completa de las opciones es actuar directamente contra la base de datos. En realidad, en las dos opciones anteriores hemos actuado de forma indirecta a través de la API y a tablas preparadas para ellos; pero esta vez somos nosotros quienes debemos crear la tabla, en caso de que sea necesario, y posteriormente hacer las consultas oportunas.



wp_options	
option_id	BIGINT(20)
blog_id	INT(11)
option_name	VARCHAR(64)
option_value	LONGTEXT
autoload	VARCHAR(20)
Indexes	

Figura 8.2. Tabla *wp_options*

Para todo ello, WordPress incluye una clase que facilita el trabajo y sobre todo cuida el aspecto de la seguridad, tan problemático cuando hay accesos a bases de datos. La clase se llama “wpdb” (*WordPress DataBase*) y ofrece métodos para:

- Hacer cualquier consulta.
- Consultar una variable.
- Consultar una fila.
- Consultar una columna.
- Consultar resultados genéricos.
- Introducir datos.
- Actualizar datos.
- Preparar las consultas para evitar ataques de inyección SQL.
- Mostrar u ocultar errores.
- Obtener información de una columna en concreto.
- Limpiar la caché.

Si queréis conocer más detalles sobre esta clase, la referencia de la misma está disponible en http://codex.wordpress.org/Function_Reference/wpdb_Class. Podríamos hablar durante horas de la misma, es otro de los pilares que soportan toda la estructura del sistema gestor de contenido. Además, lo hace a más bajo nivel que otros mecanismos como los ganchos o los filtros.

Sin lugar a dudas es el método más flexible y más potente, pero tiene el inconveniente de que debemos conocer cómo interactuar contra bases de datos MySQL (aunque wpdb ayude), y para muchas labores no será necesario llegar a este puto.

8.2.3 Otros aspectos

Para terminar este punto, me gustaría mencionar dos aspectos más de relevancia que contemplan los complementos, sobre todo los de mayor popularidad. Por un lado, la internacionalización. Al igual que en el caso de los temas, podemos hacer uso de los ficheros .po/.mo para tener nuestro complemento traducido a diferentes lenguajes.

Por otro lado, están los paneles de administración. Como habréis visto en el tema de optimización, los complementos más importantes siempre suelen tener un panel de administración dentro del principal de WordPress para permitirnos configurarlos gráficamente. Nosotros obviaremos este punto porque supone una complicación extra muy grande que desvirtúa el objetivo del texto, pero siempre es importante tenerlo en cuenta si estamos pensando en desarrollar un complemento de cierta relevancia.

Recordad que si desarrolláis un complemento de cierto tamaño, deberéis tratar de orientar vuestro código a un paradigma como el de la orientación a objetos, haciendo uso de las clases que están ya totalmente integradas en PHP versión 5. También es conveniente que tengáis una estructura de directorios, que os permita ordenar el contenido sin problemas: una carpeta para los *script* JavaScript, otra para los ficheros PHP que no son el principal, otra para imágenes... y así tantas como creáis convenientes.

8.3 ¡HOLA MUNDO!

Desde que en 1974 Brian Kernighan, uno de los dos padres de C, escribiera en uno de los libros más míticos para las Ciencias de la Computación, *The C Programming Language*, la frase “printf(‘hello, world’);” como ejemplo para mostrar un texto por pantalla, se ha repetido una y otra vez en prácticamente todos

los manuales de programación. Hoy en día, podríamos decir que es algo más que una tradición y que es extraño ver un texto sobre el tema en que este ejemplo no ocupe la primera posición. Como no podía ser de otra forma, nosotros también vamos a hacer nuestro propio “¡Hola Mundo!” aplicado a los *plugins* de WordPress.

¿Qué hará nuestro *plugin*? Simplemente escribirá la frase “¡Hola Mundo!” en el pie de la página, justo tras el contenido. Aunque el funcionamiento no es práctico, como todo ejemplo de este tipo, nos ayudará a romper el hielo y a conocer la estructura de un complemento.

8.3.1 Código fuente

Pasemos directamente a ver el código fuente. Deberemos crear un directorio de nombre “hola-mundo” en el sitio oportuno (/wp-content/plugins) y, en dicho directorio, crear un fichero vacío “hola-mundo.php”. Para evitar problemas con el juego de caracteres de los ficheros, deberéis usar codificación ANSI (la que viene por defecto), no UTF-8, que puede hacer que WordPress dé algún mensaje de error (depende de la configuración, pero la codificación ANSI en este caso es la opción más segura).

Dentro de este fichero definimos las siguientes zonas diferenciadas:

- Cabecera con datos del complemento.
- Descripción y licencia.
- Función holaMundo().
- Configuración de la acción con su respectivo gancho.

Veamos el código fuente del complemento:

```
<?php
```

```
/* Plugin Name: HolaMundo
```

```
Plugin URI: http://tu_url.com/plugin
```

```
Description: Hola Mundo en WordPress
```

```
Version: 1.0
```

```
Author: Francisco Javier Carazo Gil
```


Author URI: <http://www.jcarazo.com>

License: GPLv3

Plugin de ejemplo con el típico ¡Hola Mundo!

Este programa es software libre; puedes modificarlo/redistribuirlo

bajo los términos de la licencia GNU GPLv2 (o posteriores)

publicada por la Free Software Foundation

**/*

```
function holaMundo(){
```

```
    echo "<h1>&iexcl;Hola Mundo!</h1>";
```

```
}
```

```
add_action("wp_footer", "holaMundo");
```

```
?>
```

Como veis, el código es bien simple, sólo hay una función que emite un eco en HTML mostrando “¡Hola Mundo!” en letras de titular principal y, aparte, dos funciones, para activación y desactivación del complemento, que en este caso están vacías. La única acción que asignamos es la que ejecuta la función que emite el eco.

Guardáis el fichero, accedéis al panel de administración de vuestro blog y en la parte dedicada a los *plugins* veréis lo siguiente.



Figura 8.3. Plugin HolaMundo dentro del gestor de plugins

Ya estamos en condiciones de activarlo, tras hacerlo, podemos volver a nuestro blog y ver en la parte inferior que se ha imprimido el texto “¡Hola Mundo!” en el pie de página, tal como hemos indicado. Éste es nuestro primer *plugin*, y, como véis, sabiendo un poco de PHP, las opciones que se nos plantean son

verdaderamente grandes. Tened en cuenta que podemos incluir librerías de parte del cliente como jQuery o API de herramientas en línea como Twitter, Facebook o cualquier otra que tenga soporte para PHP.

8.4 FAVICON

Tenemos las nociones suficientes para crear un complemento, y en el capítulo anterior se mostró cómo incluir un *favicon* en nuestra web. El problema es que lo hicimos de una forma poco elegante, dentro del fichero *functions.php*, y sin dar posibilidad a cambiar la ruta. La idea es crear un complemento sencillo, que haga justo lo mismo, asignar un *favicon* a nuestra web, pero que dé la opción de cambiar la ruta del mismo. Esto supone que debemos guardar ese dato. De las tres opciones que hemos planteado antes, la más razonable, sin lugar a dudas, es usar la tabla de opciones.

Vamos a ver qué parámetros aceptan las funciones de manejo de opciones.

8.4.1 Funciones de manejo de opciones

La primera función que nos interesa es la de añadir una nueva opción. En nuestro caso será la URL del *favicon*, podríamos llamarla “*favicon_url*”, que contendrá un valor, en este caso una cadena con la referencia completa al mismo.

La función para crear opciones es *add_option(\$nombre, \$valor)*, y además de estos dos parámetros obligatorios acepta otros dos que no necesitaremos, pero que creo conveniente comentar:

- Parámetro en desuso: El tercer parámetro de la función ha dejado de ser utilizado desde WordPress 2.3, se mantiene por compatibilidad (antes era la descripción de la opción), pero ya no se usa.
- Autocarga: Sí o no (‘yes’ o ‘no’ para la función). Si es “sí”, esta opción se incluirá al llamar a la función *get_alloptions()*.

La función para leer la opción, *get_option(\$nombre)*, devuelve el valor de la opción definida por el nombre. Acepta un segundo parámetro opcional que permite establecer el valor devuelto en caso de que no exista la opción.

Otra función necesaria en muchas ocasiones, pero que no nos hará falta en este caso, es la de actualización del valor de la opción. La llamada a la función es `update_option($nombre, $nuevo_valor)`, y devuelve verdadero si el valor se ha cambiado, falso en caso contrario.

Finalmente, la función que usaremos cuando el *plugin* se desactive para que elimine la opción de la base de datos: `delete_option($nombre)`. Devuelve verdadero si la opción se ha eliminado, falso en caso contrario.

8.4.2 Activación y desactivación del plugin

Cuando el usuario activa el complemento, en muchas ocasiones hace falta ejecutar una serie de acciones para que luego pueda funcionar correctamente. En nuestro caso necesitaremos crear la opción y asignarle un valor inicial. Igualmente, cuando el complemento se desactiva, hará falta deshacer parte de las acciones creadas en la activación, así como otras que puedan haberse ejecutado en el funcionamiento del *plugin*.

Para manejar las acciones que se efectúan al activar y desactivar el complemento, deberemos crear dos ganchos y asignarles dos funciones que sean las responsables de ejecutar el código en cada caso. Veamos cuáles son:

- Activación: `register_activation_hook($fichero, $funcion)`
- Desactivación: `register_deactivation_hook($fichero, $funcion)`

“Fichero” será el archivo principal del complemento, para nuestro caso será el mismo fichero donde está dicho código, por lo que podremos hacer referencia a él directamente con la constante predefinida de PHP: “`__FILE__`”, que devuelve la ruta del fichero. La función será referenciada por una cadena con el nombre de la misma, al igual que en el caso de las acciones.

8.4.3 Código fuente

Ya estamos en condiciones de ver el código fuente del ejemplo. La función que devuelve el eco en HTML para mostrar el *favicon* en la cabecera es la misma que presentamos en el fichero anterior, pero haciendo una llamada a la opción.

Otra función necesaria en muchas ocasiones, pero que no nos hará falta en este caso, es la de actualización del valor de la opción. La llamada a la función es `update_option($nombre, $nuevo_valor)`, y devuelve verdadero si el valor se ha cambiado, falso en caso contrario.

Finalmente, la función que usaremos cuando el *plugin* se desactive para que elimine la opción de la base de datos: `delete_option($nombre)`. Devuelve verdadero si la opción se ha eliminado, falso en caso contrario.

8.4.2 Activación y desactivación del plugin

Cuando el usuario activa el complemento, en muchas ocasiones hace falta ejecutar una serie de acciones para que luego pueda funcionar correctamente. En nuestro caso necesitaremos crear la opción y asignarle un valor inicial. Igualmente, cuando el complemento se desactiva, hará falta deshacer parte de las acciones creadas en la activación, así como otras que puedan haberse ejecutado en el funcionamiento del *plugin*.

Para manejar las acciones que se efectúan al activar y desactivar el complemento, deberemos crear dos ganchos y asignarles dos funciones que sean las responsables de ejecutar el código en cada caso. Veamos cuáles son:

- Activación: `register_activation_hook($fichero, $funcion)`
- Desactivación: `register_deactivation_hook($fichero, $funcion)`

“Fichero” será el archivo principal del complemento, para nuestro caso será el mismo fichero donde está dicho código, por lo que podremos hacer referencia a él directamente con la constante predefinida de PHP: “`__FILE__`”, que devuelve la ruta del fichero. La función será referenciada por una cadena con el nombre de la misma, al igual que en el caso de las acciones.

8.4.3 Código fuente

Ya estamos en condiciones de ver el código fuente del ejemplo. La función que devuelve el eco en HTML para mostrar el *favicon* en la cabecera es la misma que presentamos en el fichero anterior, pero haciendo una llamada a la opción.

```
<?php

/*      Plugin Name: ElegirFavicon

      Plugin URI: http://tu_url.com/plugin

      Description: Un plugin para elegir un favicon de la tabla de opciones

      Version: 1.0

      Author: Francisco Javier Carazo Gil

      Author URI: http://www.jcarazo.com

      License: GPLv3 */

function elegirFavicon(){

    echo '<link rel="shortcut icon" type="image/x-icon" href="' .
    get_option("favicon_url") . "' />'; "\n";

}

function elegirFavicon_activa(){

    add_option("favicon_url", "http://codex.wordpress.org/favicon.ico");

}

function elegirFavicon_desactiva(){

    delete_option("favicon_url");

}

register_activation_hook(__FILE__, 'elegirFavicon_activa');

register_deactivation_hook(__FILE__, 'elegirFavicon_desactiva');

add_action('wp_head', 'elegirFavicon');

?>
```


Como podéis ver en el código anterior, el funcionamiento es sencillo. Tras la activación se crea la opción con el valor por defecto, el icono de la web de Codex WordPress. Se crea un gancho para que ejecute la función que emite el eco tras la creación de la cabecera. Finalmente, en caso de que el complemento se desactive, eliminamos la opción.

Para cambiar la dirección del *favicon* sólo deberemos cambiar el valor dentro de la base de datos, directamente con SQL o con la función que acabamos de mostrar de actualización. La sentencia SQL en este caso sería:

```
UPDATE wp_options SET option_value = 'nuevo_valor' WHERE option_name = 'favicon_url';
```

Si hemos cambiado el prefijo de las tablas de WordPress, deberemos poner nuestro propio prefijo: “nuestroprefijo_options”. Podemos ver en la base de datos el resultado de una consulta para ver el valor del campo:

```
SELECT * FROM wp_options WHERE option_name = 'favicon_url';
```

option_id	blog_id	option_name	option_value	autoload
☐	840	0 favicon_url	http://codex.wordpress.org/favicon.ico	38B yes

Figura 8.4. Resultado de la consulta

Si desactivamos el *plugin* y volvemos a hacer la consulta, comprobaréis que la opción se ha eliminado correctamente de la base de datos y no aparece ninguna fila en el resultado.

8.5 BLOGROLL

El *blogroll* es un término muy difundido en el mundo de los blogs y hace referencia a una lista de blogs amigos o de interés, que presentamos en nuestro propio blog. Dependiendo del sitio pueden llamarlo *blogroll*, blogs amigos, comunidad... pero todos son, en el fondo, lo mismo: una lista con enlaces con la que tratamos de dar a conocer blogs interesantes y a la vez mostrar nuestra aprobación por el trabajo realizado por otros compañeros *bloggers*.

Si nuestro tema soporta *widgets*, una de las formas ideales de presentar la lista de blogs amigos es dentro de un *widget* en la barra lateral, que se alimente de información de una tabla de la base de datos creada para este objetivo. La manera de gestionar todo esto es mediante un *plugin* que desarrollaremos en este punto.

Como veréis, introduciremos dos conceptos nuevos e importantes con los que cerramos este capítulo dedicado al desarrollo de complementos: la creación de *widgets* desde un complemento y el acceso a la base de datos de forma directa.

8.5.1 Widgets

Para crear un *plugin* que nos proporcione un *widget*, debemos seguir una serie de reglas para poder integrarlo dentro de WordPress. Básicamente, deberemos crear dos funciones y luego añadir una acción a un gancho determinado:

- Iniciador del *widget* que lo registre en la zona de *widgets*.
- Función del *widget* en sí.

La acción a añadir será la siguiente:

- `add_action("plugins_loaded", $iniciador);`

Veamos ahora el contenido de las funciones. El iniciador simplemente hará una llamada a la función `wp_register_sidebar_widget()` (antes era la función `register_sidebar_widget()`, pero ya no se debe usar esta función), que se encargará de registrar el *widget* para que pueda ser utilizado en la zona pertinente. La llamada será la siguiente:

```
wp_register_sidebar_widget(1, __('Mi blogroll'), 'widget_gestorBlogroll');
```

Los parámetros son:

- Identificador único.
- Nombre del *widget*.
- Función que genera el mismo.

Ya sólo nos queda definir la función que creará el *widget*. Tened en cuenta que lo único que debe hacer es un eco en HTML, que muestre el listado de enlaces. Para conseguir una mejor integración con el aspecto gráfico del tema, deberemos hacer uso de una serie de variables que facilitan la integración gráfica en el tema:

- `$before_widget`: Antes del *widget*.
- `$before_title`: Antes del título.

- *\$after_title*: Tras el título.
- *\$after_widget*: Tras el *widgets*.

El tema llamará a estas variables al hacer la llamada al *widgets* y vendrán dentro de un vector de nombre “args”, del que deberemos extraerlas. Ahora en el código fuente veréis el proceso entero.

8.5.2 Manejo de la base de datos

8.5.2.1 CREACIÓN DE LA TABLA

Los enlaces los almacenaremos en una tabla que contendrá los datos relativos a los mismos. Dada la naturaleza del dato a gestionar, sólo deberemos almacenar los siguientes campos:

- *id*: Identificador único, será autoincremental, conformará la clave primaria de la tabla y será el índice de referencia.
- *nombre*: Cadena que contendrá el nombre del sitio. No podrá ser nula.
- *url*: Cadena que contendrá la dirección del sitio. No podrá ser nula.

Esta tabla deberemos crearla al activar el complemento. Tened en cuenta que antes de crearla deberemos comprobar si existe para evitar que MySQL arroje errores. Esta información habría sido imposible gestionarla de forma eficaz haciendo uso de los otros dos métodos que comentamos: la tabla de opciones y los metadatos de las entradas.

Tened en cuenta que siempre que manejemos la base de datos en WordPress a un nivel que podríamos llamar bajo, al menos en comparación con los otros métodos descritos, deberemos declarar la variable global “wpdb”, que contiene el objeto que maneja el motor de la base de datos de WordPress. Es decir, siempre deberemos declararlo de la siguiente manera, antes de hacer llamadas relacionadas con la base de datos: *global \$wpdb*.

La creación de la tabla la haremos directamente en SQL, llamando al método *dbDelta(\$consulta)* presente en el fichero *wp-admin/includes/upgrade.php*, que deberemos incluir en el momento de la llamada. La consulta SQL en cuestión es la siguiente:

```
CREATE TABLE nombre_tabla(  
  
    id mediumint(9) NOT NULL AUTO_INCREMENT,  
  
    nombre VARCHAR(55) NOT NULL,  
  
    url VARCHAR(55) NOT NULL,  
  
    UNIQUE KEY id (id)  
  
);
```

Otro detalle importante es que al hacer referencia al nombre de la tabla, para evitar problemas con que el usuario cambie el prefijo de la tabla de la base de datos WordPress (acción por cierto recomendable para aumentar la seguridad), haremos uso del atributo “prefix” del objeto de manejo de la base de datos.

8.5.2.2 CONSULTA DE ENLACES

Una vez el complemento está activado, se ha creado la tabla y le hemos introducido una serie de datos; el siguiente paso es crear un método que rescate dicha información de la base de datos y los formatee dentro del *widget*. Para representar los datos vamos a usar un listado no ordenado de HTML, cuyos elementos se irán rellenando dentro de un bucle.

Haremos uso del método *get_results(\$consulta)* del objeto para tratamiento de la base de datos *wpdb*, almacenaremos su resultado en un vector y lo iremos recorriendo para mostrar los datos. La consulta será a todos los datos de la tabla: *SELECT nombre, url FROM \$tabla*.

Para introducir los datos en la tabla podéis hacer uso de phpMyAdmin, de la consola de MySQL o de cualquier otro cliente para este sistema gestor de bases de datos, como mostramos en el tercer capítulo. La consulta sería la siguiente:

```
INSERT INTO nombre_tabla ('nombre', 'url') VALUES ('nombre_ejemplo',  
'url_ejemplo');
```

Ya tenemos información suficiente para mostrar el código fuente del *plugin* y que se entienda al completo.

```
<?php
```

```
/*
```

```
    Plugin Name: GestorBlogroll
```

```
    Plugin URI: http://tu_url.com/plugin
```

```
    Description: Un plugin para presentar el blogroll almacenado en la base  
de datos en forma de widget
```

```
    Version: 1.0
```

```
    Author: Francisco Javier Carazo Gil
```

```
    Author URI: http://www.jcarazo.com
```

```
    License: GPLv3
```

```
*/
```

```
function rellenarEnlaces(){
```

```
    global $wpdb;
```

```
    $tabla = $wpdb->prefix . "gestorblogroll";
```

```
    $enlaces = $wpdb->get_results("SELECT nombre, url FROM $tabla");
```

```
    echo "<ul>\n";
```

```
    foreach ($enlaces as $enlace)
```

```
        echo "                "<li><a                href='<?php echo $enlace->url';>$enlace->nombre</a></li>\n";
```

```
    echo "</ul>\n";
```

```
}
```



```

function widget_gestorBlogroll($args) {
    extract($args);
    echo $before_widget;
    echo $before_title;?>Mi blogroll<?php echo $after_title;
    rellenarEnlaces();
    echo $after_widget;
}

function gestorBlogroll_init(){
    wp_register_sidebar_widget(1, __('Mi blogroll'), 'widget_gestorBlogroll');
}

function gestorBlogroll_activa(){
    global $wpdb;
    $tabla = $wpdb->prefix . "gestorblogroll";
    if($wpdb->get_var("SHOW TABLES LIKE '$tabla'") != $tabla){
        $sql = "CREATE TABLE " . $tabla . " (
            id mediumint(9) NOT NULL AUTO_INCREMENT,
            nombre VARCHAR(55) NOT NULL,
            url VARCHAR(55) NOT NULL,
            UNIQUE KEY id (id));";
        require_once(ABSPATH . 'wp-admin/includes/upgrade.php');
        dbDelta($sql);
    }
}

```

```
}  
  
register_activation_hook(__FILE__, 'gestorBlogroll_activa');  
  
add_action("plugins_loaded", "gestorBlogroll_init");  
  
?>
```

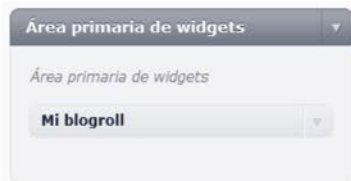


Figura 8.5. Nuestro widget activado en el panel de administración

Activamos el *plugin*, creamos los datos de ejemplo e insertamos el *widget*. Dentro del panel de administración, dentro de la sección de “Apariencia”, en “Widgets”, podremos arrastrar y soltar “Mi blogroll” al área de *widgets* que más nos guste. Una vez activado y situado el *widgets* podremos verlo en funcionamiento en nuestro blog.

Mi blogroll

- Linux Hispano
- JCarazo
- Editorial Ra-Ma

Figura 8.6. Aspecto del widget en la barra lateral del blog

8.6 CONCLUSIONES

A diferencia de los temas, donde hay una mayor proporción de código HTML y CSS de maquetación, en los *plugins* se interactúa más tiempo con la API de WordPress y el código PHP. Como habréis podido comprobar, desde el “¡Hola Mundo!” a este último ejemplo, las posibilidades que ofrece la creación de complementos son enormes, a la vez que el desarrollo de los mismos no es ni mucho menos complejo.

Cuando digo que no es complejo no me refiero a que el desarrollo de un complemento de cierto tamaño y con una funcionalidad avanzada no tenga complicación, sino que la parte de integración dentro del sistema es muy clara y no oscurece el resto del código, a diferencia de otros sistemas similares.

Como ya he comentado en este mismo capítulo, las posibilidades que se abren al poder integrar librerías, tanto en el lado cliente como en el del servidor, junto con el acceso a diferentes API que no son sólo la de WordPress, hace que tengamos un panorama muy amplio para poder seguir desarrollando.

Gracias a la filosofía de las licencias libres, siempre podremos aprender de los complementos que hemos visto en el capítulo de optimización, o de cualquier otro que encontremos por la red. Siempre deberemos empezar por complementos de menor tamaño, que no por ello menos importantes, para luego ir viendo códigos más largos, de forma que podamos llegar a comprender y aprender cómo se ha dado solución a diferentes problemas que se nos pueden presentar. También es muy posible que tengamos la necesidad de modificar, aunque sea sólo un detalle de algún complemento que usemos en nuestros blogs, por lo que siempre viene bien conocer el funcionamiento de acciones, filtros y otros mecanismos propios de la API que le dan vida a todo este entorno.

CONCLUSIONES

A lo largo de los ocho capítulos en los que está dividido el libro, he tratado de mostrar toda la evolución de lo que es un cuaderno de bitácora en la red. Desde qué es en sí un blog a, finalmente, la creación de temas y complementos, pasando por todos los puntos intermedios: qué es WordPress, probar WordPress en el servicio de Automattic, instalarlo y configurarlo en nuestro equipo, migrarlo a un servidor, para, finalmente, optimizarlo en muchos campos y pasar a describir su funcionamiento interno.

Sin lugar a dudas, aunque hay corrientes en la red que apuntan a que las redes sociales y el *microblogging* se llevarán parte del nicho de mercado de los blogs, otros muchos, pensamos justo lo contrario. No hablamos del mismo servicio, ni siquiera del mismo grupo de usuarios.

Los blogs se están convirtiendo, día tras día, en la mejor alternativa de los medios de comunicación impresos. Los periódicos y las revistas tienen los días contados, sus mismos propietarios lo saben. Sólo hay que ver plataformas como Orbyt de *El Mundo*, que intenta anticiparse a lo que pasará a medio plazo. Los blogs son la mayor amenaza en este sentido. Hay que tener en cuenta que los periódicos en línea seguirán existiendo. Ningún *blogger* puede sustituir el trabajo realizado por una redacción al completo. Sin embargo, las revistas temáticas no tienen el futuro asegurado. Al menos si no se pasan a un formato electrónico que, en el fondo, no es otra cosa que un blog.

Desde que el mundo de los blogs empezó a rentabilizarse económicamente y comenzaron a nacer y llegar empresas a este mercado, las cosas han cambiado. Sólo un ejemplo, si antes las pruebas de los nuevos modelos de vehículos sólo las

proporcionaban a las grandes revistas de papel de automóviles, con el tiempo, diferentes medios de comunicación en línea sobre el tema, muchos de ellos blogs, ya tienen acceso y pueden escribir entradas de pruebas de nuevos modelos, que poco tienen que envidiar a sus homólogos de papel.



Figura 9.1. Logo de WordPress con las letras oficiales

Tened en cuenta que el progreso natural de una revista temática de papel es el blog, pero para mantener la calidad hace falta que se retribuya económicamente el trabajo a los editores. El modelo de negocio actual, basado sólo en publicidad en línea, quizás no sea capaz de aguantar esta presión económica, pero por un lado o por otro el mercado, con el tiempo, se ajustará. Veremos este ajuste y este progreso en ciertos temas que son más favorables a estos cambios tecnológicos: los propios medios que hablan sobre tecnología y los medios más cercanos a los jóvenes.

El progreso de la telefonía móvil y la mayor presencia de *smartphone* en el mercado hará que este progreso se acelere. Asimismo, si se consigue mejorar la simbiosis entre los medios de comunicación en línea y las redes sociales, el proceso será más rápido. Sin embargo, aún queda y, en algunos temas, mucho, por lo que todavía el mercado de los blogs está bastante abierto.

De hecho, podría deciros que si ponéis un blog de temas que aún no están demasiado copados (la tecnología es uno de ellos), conseguiréis un rápido crecimiento y probablemente os convirtáis en una referencia en la red. Siempre que optimicéis correctamente y creéis buen contenido.

El mundo del *blogging* y la plataforma que centra este libro, WordPress, cambian muy rápidamente, pero los conceptos expuestos a lo largo de estas páginas, os aseguro que serán los mismos. Habrá mejoras y modificaciones, algunos de los *plugins* comentados dejarán de existir como tales para integrarse dentro del núcleo del sistema gestor de contenidos, pero, en el fondo, el entorno seguirá siendo el mismo.

Es buen momento para introducirse en este mundo. Puedes desarrollarte personalmente con tu blog personal, intentar sacar partido a un tema que te guste desarrollando un blog temático, e incluso rentabilizar lo que sabes de blogs implementando cuadernos de bitácora para empresas. Tened en cuenta que es tal la cantidad de complementos y herramientas libres que existen que la cantidad de servicios y productos que podemos ofrecer es enorme.

Como siempre se dice en las Ciencias de la Computación, “no hay que reinventar la rueda”, por lo que sólo deberéis desarrollar vosotros mismos cuando estéis completamente seguros de que nadie lo ha hecho. WordPress nos pone la herramienta, el repositorio de temas y el de complementos. Nosotros sólo tenemos que poner las ganas y los conocimientos mínimos para echar a andar. El resto os aseguro que vendrá solo.

EJEMPLOS PRÁCTICOS

Antes de dar por finalizado el texto, vamos a proceder a mostrar una serie de ejemplos prácticos de la aplicación de WordPress a problemas y necesidades concretas, tanto de particulares como de empresas. Sin entrar en mucho detalle, nos limitaremos a recomendar una serie de temas gráficos, complementos y posibles *widgets* a incluir en cada uno de los casos que se muestran.

A.1 BLOG PERSONAL

El uso primero de los cuadernos de bitácora en la red fue, sin lugar a dudas, el de blog personal, donde su creador expone opiniones, anécdotas o relatos de forma periódica. Hoy en día, este concepto se ha extendido y no sólo personas muy metidas en el mundo de la red hacen uso de blogs personales. Políticos, periodistas, economistas, científicos... incluso deportistas de élite hacen uso de este medio de comunicación. Es probable que éste también sea el uso que queráis darle muchos de vosotros. Sin lugar a dudas, todos podemos tener un blog personal, donde, además de contar nuestras opiniones y anécdotas, hablar sobre temas de los que sepamos, por ejemplo, las aficiones.

A.1.1 Ejemplos

Veamos un listado de blogs personales de cierto renombre que ahora mismo podemos encontrar en la red:

- Blog de Matt Mullenweg, padre de WordPress y presidente de Automattic:
 - <http://ma.tt/>
- Blog de Tim Berners Lee, científico considerado padre de la web:
 - <http://dig.csail.mit.edu/breadcrumbs/blog/4>
- Blog de Enrique Dans, una de las personalidades más influyentes de la red en España:
 - <http://www.enriquedans.com/>
- Blog de Ignacio Escolar, periodista:
 - <http://www.escolar.net/>
- Blog de Jordi Sevilla, ex ministro:
 - <http://blog.jordisevilla.org/>
- Blog de Jesús Encinar, emprendedor en la red y CEO de Idealista:
 - <http://www.jesusencinar.com/>
- Blog de Mark Shuttleworth, fundador de Ubuntu y presidente de Canonical. Una de las personalidades más influyentes del mundo del software libre:
 - <http://www.markshuttleworth.com/>
- “A pesar del gobierno”, blog de Carlos Rodríguez Braun, economista, profesor y tertuliano:
 - <http://www.larazon.es/blogs/show/a-pesar-del-gobierno>
- “Merodeando”, blog de Julio Alonso, fundador y CEO de Weblogs SL, la referencia empresarial en el mundo del *blogging* en español:
 - <http://www.merodeando.com/>

A.1.2 Aspecto gráfico

Si visitáis diferentes blogs personales en la red, comprobaréis que, por lo general, todos ellos tienen temas gráficos relativamente simples, poco recargados, con fondos blancos que permiten una correcta visualización del texto en cualquier dispositivo. Por lo general, se le da mucha importancia al contenido y poca al entorno. Prácticamente todos permiten comentar y sus entradas no suelen tener demasiado contenido gráfico.

En WordPress, el tema por defecto, Twenty Ten, cumple todos estos requisitos. También hay multitud de otros temas en el repositorio oficial que pueden ser buenos candidatos: Thematic, Simplr, Journalist o Elegant-Grunge. Siempre podréis modificar estos temas o ser originales y escoger otro estilo de plantilla para vuestro blog personal.

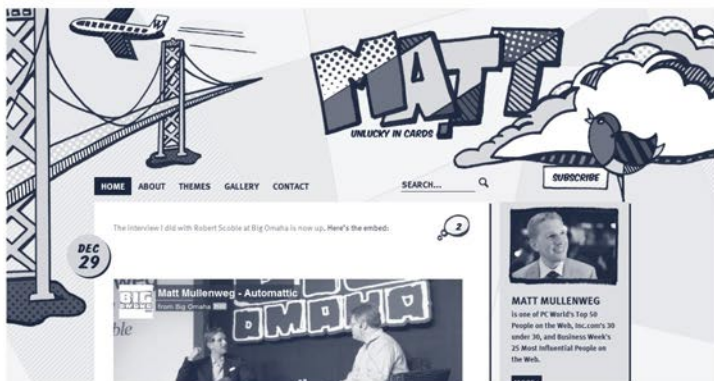


Figura A.1. Blog personal de Matt Mullenweg

A.1.3 Funcionalidad añadida

Los blogs personales suelen enfocar su acción a promover el diálogo, por lo que deberemos prestar especial atención a este punto. Necesitaremos los complementos que comentamos en el capítulo dedicado a la optimización, más concretamente: WP-RecentComments y Surprise To Comments Reloaded.

Así mismo, será conveniente tener un formulario de contacto, que permita a los visitantes comunicarse con nosotros de forma simple. Para esa labor nada mejor que un complemento como Contact Form 7.

También será conveniente crear una página de “Acerca de” lo más completa posible. Tened en cuenta que un blog personal es una ventana abierta al mundo. El SEO que proporciona WordPress, unido al contenido que podamos generar, será un escaparate para salidas profesionales o para darnos a conocer en los círculos sociales convenientes. Por esto mismo, también será conveniente instalar los *plugins* que se relacionan con las redes sociales. Especial mención merece uno que no hemos comentado, GD LinkedIn Badge, que nos relaciona directamente con nuestro perfil público dentro de la red social profesional LinkedIn. Esta red social está orientada a los contactos profesionales y a través de ella podemos tener un currículum vitae en línea, al mismo tiempo que nos relacionamos, y pedimos recomendaciones, con otros profesionales del mundo entero. Es un medio interesante para buscar oportunidades profesionales o para buscar posibles candidatos.

Por supuesto, también deberemos incluir los complementos que son necesarios para todo blog, sea de la temática que sea: los relacionados con mejora del SEO, los de seguimiento, sindicación, mejora de la seguridad... Otros como los de monetización del blog serán una elección personal en estos casos, ya que parte de nuestro público, quizás, pueda sentirse molesto al ver publicidad.

A.2 REVISTA ELECTRÓNICA, BLOG TEMÁTICO

Ni revista electrónica, por un lado, ni blog temático, por el otro, completan del todo el sentido del sector del *blogging* al que hago referencia. Las formas de comunicación han cambiado y lo que antes eran revistas especializadas, se han convertido en portales y blogs temáticos. Aunque los blogs personales o de empresa también pueden rentabilizarse, no suele ser el objetivo de los mismos, por lo que la mayor parte de los blogs que buscan un beneficio económico son de este tipo.

Sólo hay que ver el abanico de blogs que ofrecen las grandes empresas del sector en el mercado hispanohablante. Una de las pioneras, y la más importante hoy en día, es Weblogs S. L. Esta empresa tiene blogs dedicados a la política, la economía, la tecnología, la fotografía, el deporte, a los móviles... y, en cada área de éstas, podemos encontrar varios blogs especializados en subgéneros de los que acabamos de citar.

¿Qué tienen en común todos ellos? Se basan en dar información muy relevante de su área (que suele ser muy específica), para así crear un interés por parte de los usuarios, provocando que visiten y participen en el blog de forma activa. Los blogs personales también pueden tener entradas dedicadas a ciertas temáticas, pero no de forma habitual, sino más bien a modo de *hobby*. Lo mismo podemos decir de los blogs empresariales, pero la diferencia radica en que un blog como éste nace, sólo y exclusivamente, para dar información de un tema y no lo hace como una actividad accesoria.

A.2.1 Ejemplos

Podemos encontrar multitud de ejemplos. Vamos a mostrar algunos de los más relevantes en lengua hispana:

- “Linux Hispano”. Noticias, tira cómica, *podcast*, tutoriales y manuales sobre Linux y software libre:
 - <http://www.linuxhispano.net/>
- “Xataka”. *Gadgets* y tecnología:
 - <http://www.xataka.com/>
- “Genbeta”. Software y web:
 - <http://www.genbeta.com/>
- “Directo al paladar”. Recetas de cocina y gastronomía:
 - <http://www.directoalpaladar.com/>
- “Alt1040”. La guía del *geek*:
 - <http://alt1040.com/>
- “Gizmodo”. El blog de los *gadgets*:
 - <http://www.gizmodo.es/>
- “Diario Motor”. Diario sobre el mundo del motor:
 - <http://www.diarimotor.com/>

A.2.2 Aspecto gráfico

En este género, el aspecto gráfico varía enormemente de unos a otros blogs, pero en todos ellos se procura hacer atractiva la visualización del mismo, dejando de lado la simplicidad que suelen mostrar los blogs personales. Los blogs que intentan imitar más a las revistas de papel físico hacen uso de temas como Magazine Basic, que están específicamente diseñados con este objetivo. Cualquier tema, que sea llamativo y que presente claramente la información, será buen candidato, más aún si se le incluyen algunas modificaciones.

Aparte del tema gráfico, si os fijáis, todos ellos están bastante cargados de *widgets* de todo tipo. Desde los paneles de “Me gusta” de Facebook a los contadores de suscriptores de FeedBurner, podemos ver en la barra lateral de todos ellos multitud de funcionalidades extra.

Otro punto importante a destacar es el color predominante de todos ellos. Cuando dos o más blogs se hacen competencia en un nicho muy concreto de mercado, por ejemplo el automóvil, podéis observar que cada uno de los contrincantes usa un esquema de color diferenciado, pero que tiene mucho que ver con su temática. Por ejemplo, en el caso del mundo del motor, se usan colores relacionados con este sector: rojo o azul. Dependiendo de la temática, el abanico de colores será mayor, pero a la hora de competir, hace falta diferenciarse claramente, y el color principal del tema es uno de los aspectos más importantes en esta labor.



Figura A.2. Aspecto de Linux Hispano

A.2.3 Funcionalidad añadida

En este caso, el uso de *plugins* es absolutamente necesario. Todos los grandes blogs que he comentado utilizan la práctica totalidad de complementos que se han comentado en el capítulo de optimización. En algunos casos quizás no sean los mismos, pero sí usarán otros que cumplan la misma funcionalidad. En el caso de la monetización, se desarrollan *plugins* o *widgets* propios, ya que, en muchos casos, los contratos de publicidad no suelen ser con el gigante Google AdSense, sino con otras empresas con las que no se dispone de complementos para las mismas en el repositorio oficial.

Otros campos, como la integración en redes sociales, no se dejan sólo a algunos complementos como ic BeSocial, sino que se crean perfiles y páginas en todas las redes, para sacar aún más rendimiento a estas plataformas.

Sin lugar a dudas, son la máxima expresión del aprovechamiento de las posibilidades de difusión y de rentabilización de los cuadernos de bitácora en línea, por lo que si queremos adentrarnos en este mundo con esta intención, deberemos fijarnos bien, no sólo en qué complementos o *widgets* usamos, sino en otros detalles como la forma de poner los títulos, la longitud de las entradas, las palabras destacadas de cada *post*... porque nadie mejor que ellos sabe cómo maximizar las visitas.

A.3 BLOG DE EMPRESA U ORGANIZACIÓN

Llegamos al último punto de este anexo: los blogs de empresas o de organización. ¿Cuál es el cometido de este tipo de blogs? Informar de la actualidad de la empresa u organización. Desde un nuevo producto, a la presentación de un nuevo servicio, pasando por la explicación de unos datos u estadísticas de la misma.

Aunque en un comienzo eran exclusivos de empresas de carácter tecnológico, con el paso del tiempo se han ido extendiendo a todos los sectores. Tened en cuenta que, antes de su existencia, era muy frecuente la presencia de revistas corporativas, por lo que la llegada de los blogs ha sustituido este concepto por otro mucho más barato y con mayor difusión. Además, el aspecto tecnológico de los cuadernos de bitácora en línea le da un valor añadido extra a este servicio.

A.3.1 Ejemplos

- Blog oficial de Google:
 - <http://googleblog.blogspot.com/>
- Blog oficial de Facebook:
 - <http://blog.facebook.com/>
- Blog oficial de Telefónica España:
 - <http://www.blogtelefonica.es/>
- Blog oficial de Abengoa:
 - <http://blog.abengoa.es/>
- Blogosfera del BBVA:
 - <https://bbvablogs.com/>
- El blog de Renfe:
 - <http://renfeblog.com/>



Figura A.3. El blog de Renfe

A.3.2 Aspecto gráfico

Como podéis comprobar, el aspecto gráfico es mucho más similar al de los blogs personales. Tened en cuenta que el blog no es en sí la empresa, sino un medio de comunicación de la misma hacia el exterior. En el caso de los blogs temáticos, el blog es rentable y, como tal, tiene que ser atractivo y sacrificar parte de su espacio para fines publicitarios. Aquí es justo lo contrario, el blog de empresa trata de mostrar al mundo una imagen de la empresa lo más formal y seria posible. El objetivo no es ni mucho menos rentabilizarlo económicamente, para eso están las otras actividades de la empresa. Podemos comprobarlo hasta en el caso de empresas de blogs. El blog corporativo de las mismas sigue estas directrices, mientras que el resto, que son el sustento económico de la misma, se aprovechan al máximo.

Si tenéis una empresa, asociación, organización, agrupación... siempre será conveniente tener un blog de este estilo. Tened en cuenta que aporta un valor añadido a la imagen de la organización, a la vez que se puede presentar como un gancho comercial más. Si no tenéis empresa, éste es un campo que explotar si os metéis en el mundo del desarrollo de blogs. Las pequeñas empresas no serán el objetivo directo, pero prácticamente todas las organizaciones de cierto tamaño tendrán un cuaderno de bitácora en línea tarde o temprano, y vosotros podéis ser los encargados de ponerlo en marcha.

A.3.3 Funcionalidad añadida

De igual forma que en el caso de los blogs temáticos el uso de complementos y *widgets* era extensivo, en este caso, al igual que en el de los blogs personales, no lo es tanto. De hecho, podéis llegar a comprobar en el código fuente de algún que otro blog corporativo de alguna empresa que usan una versión desactualizada de WordPress (doblemente mal, primero por usar una versión antigua y segundo por mostrar el número de versión de WordPress que se usa).

A excepción de Google, que usa su plataforma Blogger, y alguna que otra empresa como Facebook o Abengoa, que usan soluciones propias, la mayor parte de los blogs corporativos de las grandes multinacionales utilizan WordPress. Los complementos que suelen usar son los básicos: All In One SEO Pack, Google Analyticator y poco más. Como veis, podrían explotar algo más esta parcela de sus blogs.

ÍNDICE ALFABÉTICO

A

Acción 19, 143, 171, 184, 186, 202, 203, 208, 210
 Actualización 25, 29, 55, 79, 84, 85, 93, 94, 95, 96, 97, 108, 111, 144, 155, 205, 207
 Add action 171, 172, 184, 203, 206, 208, 213
 AdSense 20, 50, 51, 121, 123, 125, 126, 127, 129, 131
 Akismet 29, 46, 89, 90, 92, 103, 104, 111
 All In One SEO Pack 227
 Analytics 89, 120, 121, 122, 123, 125, 127
 Android 127, 131
 Apache 55, 56, 57, 58, 107
 API 79, 88, 103, 141, 142, 144, 158, 159, 161, 170, 171, 173, 179, 194, 195, 200, 204, 213, 214
 Automattic 23, 33, 47, 48, 49, 52, 53, 90, 97, 103

B

Base de datos 25, 54, 58, 59, 60, 61, 62, 63, 64, 65, 70, 71, 72, 75, 76, 104, 136, 143, 144, 145, 146, 148, 198, 200, 205, 207, 208, 209, 210, 211
 Bing 105, 107, 114
 Blackberry 131
 Blog 16, 17, 18, 19, 20, 21, 22, 24, 25, 28, 29, 30, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 46, 48, 49, 50, 51, 52, 53, 57,

62, 63, 64, 66, 72, 77, 79, 81, 82, 89, 92, 93, 94, 96, 97, 99, 100, 102, 103, 104, 105, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 120, 122, 123, 124, 125, 129, 130, 131, 132, 134, 146, 147, 157, 158, 164, 168, 169, 177, 178, 184, 187, 195, 203, 207, 213, 219
 Blogger 19, 20, 23, 33, 34, 49, 50, 51, 52
 Buddy Press 117

C

Caché 89, 137, 138
 Captcha 101
 Categoría 28, 37, 39, 45, 46, 58, 107, 109, 113, 131, 143, 149, 153, 154, 155, 180, 181, 186, 188, 189, 193
 Chmod 86
 CMS 24, 25, 27, 30, 31, 32, 33, 60, 62, 64, 72, 83, 88, 93, 94, 119, 136, 137, 142, 148, 149, 151, 157, 180, 194, 195
 Código fuente 22, 38, 61, 169, 170, 173, 176, 178, 181, 185, 187, 190, 191, 193, 202, 205, 209, 210
 Comentario 43, 46, 99, 100, 101, 103, 104, 132, 135, 143, 152, 153, 181, 182
 Consola 57, 59, 60, 67, 74, 86, 210
 Contact Form 7 89, 222
 Cookies 156, 157, 158
 CPU 66, 113
 CSS 38, 42, 48, 79, 142, 143, 161, 164, 167, 168, 169, 170, 186, 188, 193, 213

D

Digg.....115
 Drupal.....24, 25, 27, 30, 32

E

Enlaces permanentes.....106, 107, 108, 118,
 147, 154
 Etiquetas.....37, 109, 167, 198

F

Facebook.....81, 115, 116, 117, 204, 224,
 226, 227
 Favicon.....171, 172, 198, 204, 205, 206, 207
 FeedBurner.....123, 125, 127, 224
 FileZilla.....55, 67, 68, 69, 71, 72, 87
 Footer.....164, 165, 173, 174, 175, 183, 184,
 191, 193, 203
 FTP.....55, 66, 67, 68, 71, 73, 77, 83, 85, 86,
 95
 Funciones.....36, 114, 118, 158, 159, 161,
 165, 166, 170, 171, 172, 173, 174, 178,
 179, 180, 181, 185, 187, 194, 195, 198,
 199, 203, 204, 205, 208

G

Gancho.....171, 198, 202, 207, 208
 Geany.....61, 77
 Gedit.....61
 GET.....106
 Gimp.....161, 162
 Google.....20, 22, 28, 33, 49, 50, 51, 61, 89,
 92, 105, 106, 108, 112, 114, 115, 117, 120,
 121, 122, 123, 125, 126, 127, 131, 132,
 225, 226, 227
 GPL.....22, 25, 49, 164, 193
 Gravatar.....132, 133, 179

H

Header.....165, 169, 173, 174, 175, 185, 190
 ¡Hola Mundo!.....37, 202, 203, 213
 Hook.....171, 198, 205, 206, 213
 HTML.....38, 64, 124, 128, 129, 137, 138,
 142, 143, 155, 161, 171, 173, 179, 183,
 184, 185, 186, 187, 193, 203, 205, 208,
 210, 213

HTTP.....156, 184

I

IC BeSocial.....116, 117, 225
 iPhone.....131, 132

J

Java.....105, 143
 Joomla!.....24, 25, 27, 30, 31, 32

L

Linux.....23, 26, 32, 54, 57, 59, 65, 67, 81,
 84, 85, 86, 107
 Login.....148, 157, 159
 Loop.....165, 173, 174, 177, 178, 181, 187,
 193

M

Matt Mullenweg.....22, 23, 29, 33, 103
 Microsoft Internet Explorer.....22
 Movable Type.....21, 24, 30, 31, 39
 Mozilla Firefox.....22, 88
 MySQL.....54, 55, 56, 57, 58, 59, 60, 61, 62,
 63, 65, 70, 71, 73, 74, 76, 130, 143, 144,
 201, 209, 210

N

NET.....65

P

Palm.....131
 Password.....60, 62, 71, 150, 158, 182
 Permalink.....106, 188
 PHP.....24, 27, 31, 54, 55, 57, 63, 65, 73,
 74, 136, 137, 140, 142, 143, 146, 158, 161,
 164, 169, 170, 173, 174, 177, 181, 193,
 195, 196, 197, 198, 201, 203, 205, 213
 PHPBB.....24, 119
 PHPMyAdmin.....55, 59, 74, 76, 77, 210
 Pingback.....39, 116, 120, 152, 153, 186
 Plugin.....25, 27, 79, 80, 86, 88, 89, 90, 91,
 92, 95, 102, 103, 104, 111, 112, 114, 116,
 118, 121, 122, 123, 127, 128, 131, 134,
 135, 136, 137, 138, 195, 196, 197, 198,

Y

Yahoo!.....28, 105, 106, 114

Z

ZIP55, 58, 82, 90, 91, 162

Domine WordPress

Manual práctico

Hace diez años los blogs apenas eran conocidos en la sociedad. Hoy en día, es raro encontrar a alguien que no visite un blog. Los hay que lo hacen a diario, otros llegan a ellos derivados por Google o cualquier otro buscador; pero al final, gran parte de los usuarios de la red leen entradas de los mismos y dejan sus comentarios. Es tal el desarrollo de esta herramienta de comunicación en línea que existen empresas que se dedican a esta actividad; y no son pocos los políticos, profesionales de renombre, famosos, empresas y organizaciones de todo tipo, que tienen su propio blog oficial. El concepto inicial de escribir entradas de forma periódica y poder comentar sobre las mismas se ha ampliado de tal manera, que mediante un blog podemos llegar a crear una comunidad virtual en la que los usuarios hablen entre sí y compartan información.

Lo mejor de todo, es que gracias al uso de tecnologías libres como la que presentamos en este libro, WordPress, tenemos en nuestra mano hacer todo esto y mucho más en muy poco tiempo. No hace falta escribir una sola línea de código para tener una plataforma llena de funcionalidades. Tampoco hace falta saber nada en especial para poder generar contenido para la plataforma. Sólo hay que tener ganas, poner empeño en el contenido que se escribe y seguir las directrices expuestas en este texto; así, la interacción con las redes sociales y, sobre todo, el posicionamiento en los buscadores, hará que recibamos visitas de miles de usuarios.

Podemos optar por escribir nuestro propio blog personal. Comentar nuestras opiniones, poner fotos de nuestros viajes o hablar sobre algo muy específico para gente que comparta nuestras aficiones.

Otra opción es desarrollarnos profesionalmente en este campo. Las empresas y organizaciones necesitan tener presencia en la red y muchas de ellas querrán cambiar su página estática por algo más dinámico como un blog. No hablemos ya de las que directamente no tienen presencia en la red.

Y finalmente, podemos tratar de ganarnos la vida, o al menos un extra, hablando de lo que nos gusta. ¿Te gusta tal tema y quieres compartir información con el resto? Sólo tienes que montar un blog y hablar sobre tu *hobby*. Pueden ser los coches, los Scalextric, un lenguaje de programación, la cocina, los videojuegos, la fotografía, la pintura, el cine... Seguro que en la red hay miles de usuarios deseosos de leerle. Todo esto redundará en una dinámica en la que al final, el número de visitas hace que te motives, e incluso podemos llegar a tener hasta un beneficio económico.

Los blogs son uno de los medios de comunicación del futuro y leyendo este texto tendrás la información y las herramientas necesarias para saber dominarlos.



ra-ma.es



Ra-Ma®