
Introducción al mundo de los estándares web

PID_00253483

Mark Norman Francis
Jonathan Lane

Tiempo mínimo previsto de lectura y comprensión: 3 horas



Universitat
Oberta
de Catalunya

Índice

1. La historia de Internet y la web y la evolución de los estándares web	5
1.1. Los orígenes de Internet	5
1.2. La creación de la web mundial	6
1.2.1. Las “guerras de los navegadores”	7
1.3. La aparición de los estándares web	8
1.3.1. La formación del W3C	8
1.3.2. El proyecto de estándares web	9
1.3.3. El auge de los estándares web	9
Resumen	10
Preguntas de repaso	11
Lecturas complementarias	11
2. ¿Cómo funciona Internet?	12
2.1. ¿Cómo se comunican los ordenadores a través de Internet?	12
2.1.1. Disección de un ciclo de solicitud-respuesta	13
2.2. Tipos de contenido	15
2.2.1. Texto normal	15
2.2.2. Estándares web	15
2.2.3. Páginas web dinámicas	16
2.2.4. Formatos que requieren otras aplicaciones o conectores	16
2.3. Páginas web estáticas o dinámicas	17
Resumen	18
Preguntas de repaso	18
Lecturas complementarias	18
3. El modelo de estándares web: HTML, CSS y JavaScript	19
3.1. ¿Por qué separar?	19
3.2. Etiquetado, la base de cada página	20
3.2.1. ¿Qué es el XHTML?	21
3.2.2. Validación, ¿qué es eso?	22
3.3. CSS: añadimos un poco de estilo	22
3.4. JavaScript: adición de comportamiento a las páginas web	24
3.5. Una página de ejemplo	24
3.5.1. index.html	24
3.5.2. styles.css	25
Resumen	27
Preguntas de repaso	28

4. Estándares web: un bonito sueño,	
pero ¿cuál es la realidad?	29
4.1. ¿Cómo se comprueba el cumplimiento de los estándares web?	29
4.2. Compatibilidad de los estándares en las páginas actualmente	30
4.2.1. Amazon: ¿comprar con estándares?	31
4.2.2. CNN: ¿noticias estandarizadas?	31
4.2.3. Apple: el máximo de elegancia en diseño... ¿y en validación?	32
4.2.4. Un pequeño sondeo de compatibilidad de estándares	33
4.3. ¿Por qué hay tan pocas páginas que cumplan los estándares?	33
4.3.1. Educación	34
4.3.2. Motivos empresariales	34
Resumen	35
Preguntas de repaso	36
Lecturas complementarias	36

1. La historia de Internet y la web y la evolución de los estándares web

Mark Norman Francis

“¿Por dónde debo empezar, Majestad?

Empieza por el principio –dijo el rey solemnemente–, y continúa hasta llegar al final. Entonces, detente”.

Alicia en el País de las Maravillas, Lewis Carroll

Todo debe empezar por algún sitio, de manera que nuestro viaje empezará con una lección centrada en la historia. A continuación, realizaremos un breve repaso de la creación de Internet, la web mundial (World Wide Web), y de los estándares web en los que se centra toda esta serie. Creo que es útil e interesante entender cómo hemos llegado hasta donde estamos, pero seremos lo bastante breves como para no agobiaros y poder entrar en detalle de manera rápida y agradable. Si no estáis familiarizados con algún término, no os preocupéis; si son importantes para aprender sobre el desarrollo web, se definirán en los últimos apartados que amplían cada tema, y siempre podéis hacer una busca en Google. Si ya estáis familiarizados con la historia de Internet o de la web mundial, os podéis saltar este apartado sobre los estándares web.

1.1. Los orígenes de Internet

El cuatro de octubre de 1957 sucedió un acontecimiento que cambiaría el mundo. La Unión Soviética lanzó con éxito el primer satélite a la órbita de la Tierra. Se llamaba *Sputnik 1* y sorprendió al mundo, especialmente a Estados Unidos, que tenía en curso su propio programa de lanzamientos de satélites, pero todavía no habían lanzado ninguno.

Este acontecimiento condujo directamente a la creación de la ARPA (Advanced Research Projects Agency, la Agencia de Proyectos de Investigación Avanzada) del Departamento de Defensa de Estados Unidos, a causa de la necesidad reconocida de una organización que pudiera investigar y desarrollar ideas avanzadas y tecnología más allá de las necesidades identificadas actualmente. Quizá su proyecto más famoso (sin duda el más ampliamente utilizado) fue la creación de Internet.

En 1960, el psicólogo y científico informático Joseph Licklider publicó un documento titulado *Simbiosis Hombre-Ordenador*, que articuló la idea de ordenadores en red que proporcionaban un almacenaje y una recuperación avanzada de los datos. En 1962, mientras trabajaba para la ARPA como jefe de la oficina de procesamiento de información, formó un grupo para continuar con la investigación informática, pero lo abandonó antes de que se trabajara en aquella idea.

El plan para esta red de ordenadores (que se denominaría ARPANET) se presentó en octubre de 1967 y en diciembre de 1969 la primera red de cuatro ordenadores ya estaba conectada y en funcionamiento. El principal problema de la creación de una red era cómo conectar redes físicamente separadas sin colapsar los recursos de la red a causa de las conexiones constantes. La técnica que resolvió este problema se conoce como conmutación de paquetes e implica que las solicitudes de datos se dividen en pequeños trozos (paquetes) que se pueden procesar rápidamente sin bloquear la comunicación de los otros. Este principio todavía se utiliza en la actualidad para el funcionamiento de Internet.

Este concepto se adoptó ampliamente con el nacimiento de otras redes que utilizaban la misma técnica de conmutación de paquetes. Por ejemplo, la X.25 (desarrollada por la Unión Internacional de Telecomunicaciones) formó la base de la primera red universitaria del Reino Unido: JANET (que permitía a las universidades del Reino Unido enviar y recibir ficheros) y la red pública norteamericana CompuServe (una empresa comercial que permitía a pequeñas empresas y personas acceder a los recursos informáticos con tiempo compartido, y posteriormente el acceso a Internet). Estas redes, a pesar de tener muchas conexiones, eran más privadas que la Internet actual.

Esta proliferación de diferentes protocolos de red no tardó mucho en convertirse en un problema cuando se intentaba que todas las redes independientes se comunicaran. Sin embargo, había una solución a la vista: Robert Kahn, mientras trabajaba en un proyecto de red de paquetes por satélite para ARPA, empezó a definir algunas reglas para una arquitectura de red más abierta que sustituyera el protocolo actual que se utilizaba en ARPANET. Más adelante, Vinton Cerf –de la Universidad de Stanford– se incorporó al proyecto y ambos crearon un sistema que enmascaraba las diferencias entre protocolos de red utilizando un nuevo estándar. La publicación del borrador de la especificación, en diciembre de 1974, se denominó Programa de Control de Transmisión de Internet.

Esta especificación reducía las funciones de la red y trasladaba la tarea de mantener la integridad de la transmisión al ordenador principal. El resultado final fue que era posible unir fácilmente casi todas las redes entre ellas. ARPA asumió el coste del desarrollo del software y en 1977 se llevó a cabo una demostración de comunicación entre tres redes diferentes. En 1981, la especificación se completó, publicó y adoptó; y en 1982 las conexiones de ARPANET fuera de Estados Unidos se convirtieron para utilizar el nuevo protocolo TCP/IP. Había llegado Internet tal como la conocemos.

1.2. La creación de la web mundial

Gopher* era un sistema de recuperación de información que se utilizaba a principios de los años noventa y que proporcionaba un método de entrega de menús de enlaces a archivos, recursos informáticos y otros menús. Estos menús podían cruzar los límites del ordenador y utilizar Internet para ir a buscar

* [http://en.wikipedia.org/wiki/Gopher_\(protocol\)](http://en.wikipedia.org/wiki/Gopher_(protocol))

menús de otros sistemas. Era muy popular en las universidades, que querían proporcionar información para todo el campus, y organizaciones grandes que querían centralizar el almacenaje y la gestión de documentos.

Gopher fue creado por la Universidad de Minnesota. En febrero de 1993, esta universidad anunció que cobraría licencias por el uso de la implementación de referencia del servidor Gopher. En consecuencia, muchas organizaciones empezaron a buscar alternativas a Gopher.

El Consejo Europeo de Investigación Nuclear (CERN), en Suiza, tenía esta alternativa. Tim Berners-Lee había estado trabajando en un sistema de gestión de información en el que el texto pudiera contener enlaces y referencias a otros trabajos, de manera que permitiera al lector saltar rápidamente de un documento a otro. Había creado un servidor para publicar este estilo de documento (denominado hipertexto) y también un programa para leerlo, al que había denominado World Wide Web. Este software se publicó por primera vez en 1991, pero dos acontecimientos provocaron una explosión de popularidad y, finalmente, la sustitución de Gopher.

El 30 de abril de 1993 el CERN cedió el código fuente del World Wide Web al dominio público, de manera que cualquiera pudiera utilizar o construir sobre el software sin ningún coste.

Así, más tarde, en el mismo año, el NCSA (National Center for Supercomputing Applications, Centro Nacional para Aplicaciones de Supercomputación) publicó un programa que era una combinación de navegador web y cliente Gopher, denominado Mosaic. Originalmente, sólo estaba disponible para equipos Unix y en forma de código fuente, pero en diciembre de 1993 Mosaic ya disponía de una nueva versión con instaladores tanto para Apple Macintosh como para Microsoft Windows. Mosaic aumentó en popularidad rápidamente y, en consecuencia, también la web.

El número de navegadores web disponibles aumentó muchísimo, muchos de ellos creados para proyectos de investigación en universidades y corporaciones, como Telenor (una compañía noruega de comunicaciones), que creó la primera versión del navegador Opera en 1994.

1.2.1. Las “guerras de los navegadores”

La popularización de la web atrajo intereses comerciales. Marc Andreessen abandonó el NCSA y, junto con Jim Clark, fundó Mosaic Communications, que más adelante cambió su nombre por Netscape Communications Corporation, y empezaron a trabajar en lo que acabaría convirtiéndose en el navegador Netscape. La versión 1.0 del software se publicó en diciembre de 1994.

Spyglass Inc. (la rama comercial del NCSA) autorizó la comercialización de su tecnología Mosaic a Microsoft para formar la base de Internet Explorer. La versión 1.0 se publicó en agosto de 1995.

Una rápida escalada siguió a continuación, en la que Netscape y Microsoft intentaban cada uno obtener una ventaja competitiva en cuanto a las funciones que admitían con el fin de atraer desarrolladores. Desde entonces, esta competición se ha conocido como “las guerras de los navegadores”. Opera mantuvo una presencia modesta pero continuada a lo largo de este período e intentó innovar y ser compatible con los estándares web lo mejor posible en aquellos tiempos.

1.3. La aparición de los estándares web

Durante las guerras de los navegadores, Microsoft y Netscape se centraron en la implementación de nuevas funciones en lugar de resolver los problemas de las funciones con las que ya eran compatibles, y también en añadir funciones propias y crear funciones que fueran competencia directa de las existentes en el otro navegador, pero implementadas de manera incompatible.

En aquellos momentos, los desarrolladores se veían forzados a tratar con niveles de confusión cada vez mayores cuando se intentaban construir las páginas web, a veces hasta el punto de haber de construir dos páginas diferentes, pero duplicadas en la práctica, para cada uno de los dos principales navegadores, y otras simplemente optando por ser compatibles sólo con un navegador, de manera que los usuarios que utilizaran el otro no pudieran utilizar sus páginas. Ésta era una manera muy mala de trabajar y la inevitable reacción negativa de los desarrolladores no tardó en producirse.

1.3.1. La formación del W3C

En 1994, Tim Berners-Lee fundó el World Wide Web Consortium (W3C) en el Massachusetts Institute of Technology, con el apoyo del CERN, DARPA (como se había bautizado la ARPA) y la Comisión Europea. La misión del W3C era estandarizar los protocolos y las tecnologías utilizadas para construir la web, de manera que el contenido estuviera disponible para la mayor parte posible de la población del mundo.

Durante los años siguientes, el W3C publicó varias especificaciones (denominadas recomendaciones) incluyendo HTML 4.0, el formato para imágenes PNG y las versiones 1 y 2 de CSS (*cascading style sheets* u hojas de estilo en cascada).

No obstante, el W3C no impone sus recomendaciones. Los fabricantes sólo deben ajustarse a la documentación del W3C si quieren etiquetar su producto como cumplidor del W3C. En la práctica, esto no es un argumento de venta

valioso porque casi todos los usuarios de la web desconocen, y probablemente no les importa, quién es el W3C. En consecuencia, las guerras de los navegadores continuaron sin trabas.

1.3.2. El proyecto de estándares web

En 1998, el mercado de los navegadores estaba dominado por Internet Explorer 4 y Netscape Navigator 4. Se había lanzado una versión beta de Internet Explorer 5 que implementaba un nuevo HTML dinámico de marca registrada. Ello significaba que los desarrolladores web profesionales debían conocer cinco maneras diferentes de escribir JavaScript.

En consecuencia, un grupo de desarrolladores y diseñadores web se asociaron entre ellos. Este grupo se denominaba WaSP (Web Standards Project, Proyecto de estándares web). La idea era que si los documentos del W3C se llamaban *estándares* en vez de *recomendaciones* podrían convencer a Microsoft y Netscape de que les dieran su apoyo.

El antiguo método de realizar un llamamiento a la acción se llevó a cabo mediante una técnica publicitaria tradicional denominada *barricada*, donde una empresa lanza un anuncio en todos los canales de emisión al mismo tiempo, de manera que aunque el espectador cambie de canal, obtendrá exactamente el mismo mensaje. WaSP publicó un artículo simultáneamente en varias páginas centradas en el desarrollo web, como *builder.com*, *Wired online* y algunas listas de correo muy populares.

Otra técnica que utilizaron fue ridiculizar a las empresas que se unían al W3C (y a otros organismos de estándares), pero que después se centraban más en crear nuevas funciones que en hacer que los conceptos básicos para los que se habían comprometido fueran correctos, para empezar.

Todo esto suena un poco negativo, pero los de WaSP no se conformaban con criticar a la gente, también la ayudaban. Siete miembros formaron CSS Samurai, que identificó los diez problemas principales de compatibilidad CSS en Opera e Internet Explorer (Opera resolvió sus problemas y Microsoft no).

1.3.3. El auge de los estándares web

En el 2000, Microsoft lanzó Internet Explorer 5 Macintosh Edition. Fue un hito muy importante, ya que se trataba del navegador que se instalaba entonces de manera predeterminada con el Mac OS, y también tenía un nivel decente de compatibilidad con las recomendaciones del W3C. Junto con el nivel decente de compatibilidad con CSS y HTML, Opera contribuyó a un movimiento positivo general, con el que los desarrolladores y diseñadores web se

sentían cómodos diseñando páginas mediante estándares web por primera vez.

WaSP persuadió a Netscape de retrasar el lanzamiento de la versión 5.0 de Netscape Navigator hasta que fuera mucho más compatible (este trabajo formó la base de lo que ahora es Firefox, un navegador muy popular). WaSP también creó un grupo de trabajo para Dreamweaver, con el fin de animar a Macromedia a cambiar su popular herramienta de autoría web y dar soporte a la creación de páginas web compatibles.

La popular página de desarrollo web *A List Apart* se rediseñó a principios del 2001 y, en un artículo que explicaba cómo y por qué, declaraba:

“En seis meses, un año, o dos años como mucho, todas las páginas se diseñarán con estos estándares. [...] Podemos contemplar cómo nuestras capacidades se quedan obsoletas o podemos empezar a aprender ahora técnicas basadas en estándares”.

Esto era un poco optimista: no todas las páginas, ni siquiera en el año 2009, están hechas con estándares web. Pero muchos les hicieron caso. Los navegadores antiguos redujeron su cuota de mercado y dos páginas web más de perfil muy alto se rediseñaron utilizando estándares web: la revista *Wired* en el 2002 y ESPN en el 2003 se convirtieron en líderes del sector en el soporte a los estándares web y las nuevas técnicas.

También en el 2003, Dave Shea creó una página web denominada CSS Zen Garden. Debía tener más impacto sobre los profesionales web que cualquier otra cosa, e ilustraba, verdaderamente, que todo el diseño podía cambiar sólo modificando el estilo de la página; el contenido podía seguir siendo idéntico.

Desde entonces, en la comunidad de desarrollo web profesional, los estándares web se han convertido en un elemento de rigor. En esta serie os daremos unos excelentes fundamentos en estas técnicas para que podáis crear páginas web tan limpias, semánticas, accesibles y conformes con los estándares como las de las grandes empresas.

Resumen

En este apartado hemos hablado de cómo se creó la Internet moderna como resultado de la carrera espacial, de cómo Tim Berners-Lee definió el hipertexto para una generación y de cómo los intereses comerciales de dos compañías provocaron una de las reacciones más notables de los desarrolladores como nunca se había visto. El término *estándares web* se utiliza ahora más ampliamente entre los profesionales web que cualquier otro término aplicado por el W3C (de hecho, el W3C ha empezado a utilizar el término en sus propias páginas), de manera que por este motivo os enseñaremos el modo de construir páginas web siguiendo los estándares.

Preguntas de repaso

Podrías intentar investigar más respondiendo estas preguntas:

1. ¿Qué navegadores están disponibles actualmente en Internet para los usuarios de Windows, Mac OS X y Linux?
2. ¿Qué porcentaje de usuarios de la web utiliza cada navegador?
3. ¿Qué navegadores utilizan los dispositivos móviles para acceder a las páginas web?
4. ¿Cuántos estándares web ha publicado el W3C y cuáles son ampliamente seguidos por los fabricantes de navegadores actualmente?

Lecturas complementarias

Si queréis saber más, es posible que queráis visitar algunas de las páginas siguientes:

The history of the Internet (wikipedia)

http://en.wikipedia.org/wiki/History_of_the_Internet

The history of the World Wide Web (wikipedia)

http://en.wikipedia.org/wiki/History_of_the_World_Wide_Web

The history of the W3C

<http://www.w3.org/Consortium/history>

El Web Standards Project y su historia

<http://webstandards.org/>

<http://www.webstandards.org/about/history/>

A List Apart

<http://www.alistapart.com>

CSS Zen Garden

<http://www.csszengarden.com>

2. ¿Cómo funciona Internet?

Jonathan Lane

Muy de vez en cuando es posible conseguir ver los engranajes y las correas que actúan entre bastidores. Hoy es vuestro día de suerte. Os llevaré a dar una vuelta tras los telares de una de las tecnologías más interesantes que, posiblemente, ya conocéis bien: la web.

Este apartado trata sobre la tecnología subyacente que hace funcionar la web:

- Lenguaje de etiquetado de hipertexto (HTML).
- Protocolo de transferencia de hipertexto (HTTP).
- Sistema de nombres de dominio (DNS).
- Servidores web y navegadores web.
- Contenido estático y dinámico.

Son materias bastante básicas y, aunque la mayor parte de lo que aquí se explica no os ayudará a construir una página web mejor, sí que os proporcionará el lenguaje adecuado para hablar con clientes y otras personas sobre Internet. Es como lo que una institutriz muy inteligente dijo una vez en *Sonrisas y lágrimas*: “Cuando leemos, empezamos por ABC. Cuando cantamos, empezamos por Do Re Mi”. En este apartado describiremos brevemente cómo los ordenadores se comunican realmente utilizando HTTP y TCP/IP, y después nos fijaremos en los diferentes lenguajes que se combinan para crear las páginas web que conforman Internet.

2.1. ¿Cómo se comunican los ordenadores a través de Internet?

Por suerte, hemos mantenido las cosas simples para los ordenadores. Cuando se trata de la web, la mayoría de las páginas están escritas utilizando el mismo lenguaje, el HTML, que pasa de un sitio a otro utilizando un protocolo común: HTTP (*hypertext transfer protocol* o protocolo de transferencia de hipertexto). El HTTP es el dialecto (especificación) común de Internet que permite, por ejemplo, que un equipo con Windows cante en armonía con un ordenador que ejecute la versión más reciente y más fantástica de Linux (¡Do Re Mi!). Mediante el uso de un navegador web –un software especial que interpreta el HTTP y entrega el HTML en una manera legible para los humanos–, las páginas web creadas con HTML y con cualquier tipo de ordenador se pueden leer en cualquier medio, incluyendo teléfonos, PDA e incluso en los sistemas de videojuegos más populares.

Aunque hablen el mismo lenguaje, los diferentes dispositivos que acceden a la web han de tener algunas normas establecidas para poder hablar entre

ellos; es como aprender a alzar la mano para preguntar en clase. El HTTP establece estas normas básicas para Internet. Gracias al HTTP, un equipo cliente (como vuestro ordenador) sabe que debe ser él quien inicie una petición de una página web desde un **servidor**. Un servidor es un ordenador donde residen las páginas web; cuando escribís una dirección web en vuestro navegador, un servidor recibe la petición, encuentra la página web que deseáis y la envía a vuestro ordenador para que se vea en vuestro navegador.

2.1.1. Disección de un ciclo de solicitud-respuesta

Ahora que hemos visto todas las partes que permiten a los ordenadores comunicarse a través de Internet, trataremos con más detalle el ciclo de solicitud-respuesta de HTTP. A continuación, se presentan una serie de pasos numerados para que podáis trabajar siguiéndolos y, así, os podremos demostrar algunos conceptos de manera más eficaz:

1) Todas las solicitudes-respuestas empiezan cuando se escribe un URL (*universal resource locator*, localizador universal de recursos) en la barra de direcciones del navegador web, como `http://dev.opera.com`. Abrid un navegador para hacerlo ahora mismo.

Algo que quizá no sabéis es que los navegadores realmente no utilizan los URL para solicitar páginas web en los servidores; utilizan el **Protocolo de Internet** o **direcciones IP** (que son casi como números de teléfono o direcciones postales que identifican los servidores.) Por ejemplo, la dirección IP de `http://dev.opera.com` es 213.236.208.98.

2) Abrid una nueva pestaña o ventana del navegador, escribid `http://www.apple.com` y pulsad *intro*; a continuación, escribid `http://17.149.160.10/` y pulsad *intro*: llegaréis al mismo sitio. Escribid `http://213.236.208.98` en la barra de direcciones y pulsad *intro*: iréis a parar al mismo sitio que en el paso 1, pero obtendréis un error 403 “Acceso Denegado”, esto es porque no tenéis permiso para acceder a la raíz real de este servidor.

`http://www.apple.com` actúa básicamente como un alias para `http://17.149.160.10/`, pero ¿por qué? ¿Y cómo? Ello se debe a que a las personas les resulta más fácil recordar palabras que largas cadenas de números. El sistema que realiza este trabajo se denomina *Domain name system* (DNS) o Sistema de nombres de dominio, que es esencialmente un directorio automático completo de todos los ordenadores conectados a Internet. Cuando escribís `http://dev.opera.com` en la barra de direcciones y apretáis *intro*, esta dirección se envía a un servidor de nombres que intenta asociarla a vuestra dirección IP. Hay muchos ordenadores conectados a Internet, y no todos los servidores DNS tie-

nen un listado de cada ordenador que hay conectado, de manera que hay un sistema creado donde se puede dirigir la solicitud al servidor correcto para atenderla.

Así pues, el sistema DNS busca la página web `www.opera.com`, averigua que se encuentra en `17.149.160.10` y devuelve la dirección IP al navegador.

El ordenador envía una solicitud al ordenador de la dirección IP especificada y espera obtener una respuesta. Si todo va bien, el ordenador del servidor envía un breve mensaje de retorno al cliente que dice que todo es correcto (podéis ver la figura 1) seguido de la propia página web. Este tipo de mensaje está incluido en un **encabezamiento HTTP**.

Figura 1



En este caso todo es correcto y el servidor devuelve la página web correcta.

Si algo va mal, por ejemplo, si se escribe incorrectamente el URL, en su lugar se obtendrá un **error HTTP**: el famoso error 404 “no se encuentra la página” es el ejemplo más común que se puede encontrar.

3) Escribid `http://dev.opera.com/joniscool.html`: la página no existe, o sea que obtendréis un error 404. Probadlo con algunas páginas, en diferentes páginas web que no existan y os devolverán páginas diferentes. Esto se debe al hecho de que algunos desarrolladores web han dejado que el servidor web sólo emita la página de error predeterminada y otros han codificado páginas de error personalizadas para que aparezcan cuando se devuelve una página no existente. Se trata de una técnica avanzada que no trataremos en esta asignatura, pero que afortunadamente se verá pronto en un artículo aparte en `dev.opera.com`.

Por último, una nota sobre los URL: generalmente el primer URL al que se accede en una página web no tiene un nombre de archivo real al final (por ejemplo, `http://www.mysite.com/`), y a continuación las páginas subsiguientes a veces tienen y a veces no. Siempre se accede a archivos reales, pero en ocasiones el desarrollador web ha configurado el servidor web para que no muestre

los nombres de archivo en el URL; esto a menudo permite conseguir URL más limpios y fáciles de recordar, que conducen a una mejor experiencia para el usuario de vuestra página web.

2.2. Tipos de contenido

Ahora que ya os hemos enseñado una solicitud-respuesta HTTP, queremos que os fijéis en los diferentes tipos de contenido que pueden encontrarse en Internet. Los hemos agrupado en 4 tipos: texto normal, estándares web, páginas web dinámicas y formatos que requieren otras aplicaciones o conectores.

2.2.1. Texto normal

Durante los primeros días de Internet, antes de que apareciera cualquier estándar web o conector, Internet era principalmente imágenes y texto normal, archivos con una extensión .txt o similar. Cuando se encuentra un texto normal en Internet, el navegador lo muestra tal como es, sin ningún tipo de proceso. Todavía pueden encontrarse ficheros de texto normal en páginas web universitarias.

2.2.2. Estándares web

Las herramientas de construcción básicas de la web son los tres principales estándares web: HTML (o XHTML, aquí utilizaremos ambos indistintamente para nuestras finalidades), CSS y JavaScript:

a) **Lenguaje de marcado de hipertexto (HTML)**, es un nombre realmente bueno con respecto a la definición de su objetivo. El HTML se utiliza para dividir un documento, especificar sus contenidos y su estructura, y definir el significado de cada parte (es lo que incluye todo el texto, etc., que se ve en las páginas web). Utiliza elementos para identificar los diferentes componentes de una página.

b) Las **hojas de estilo en cascada (CSS)** dan un control total sobre cómo se visualiza un elemento.

Existen muchas ventajas para separar la estructura del formato y lo veremos con más detalle en el siguiente apartado. Para demostrar la potencia de HTML y CSS utilizados de manera combinada, la figura 2 muestra a la izquierda HTML normal, sin ningún formato añadido, mientras que a la derecha puede verse exactamente el mismo HTML con algunos estilos de CSS aplicados.

Cambios de formato

Utilizando declaraciones de estilo es muy sencillo cambiar todos los párrafos para que queden a doble espacio:

```
line-height: 2em;
```

O hacer que todos los encabezamientos de segundo nivel sean verdes:

```
color: green;
```

Figura 2



HTML normal a la izquierda, HTML con CSS aplicado a la derecha.

c) Por último, el **lenguaje JavaScript** aporta funciones dinámicas a las páginas web. Se pueden escribir pequeños programas en JavaScript que se ejecutarán en el ordenador cliente y que no requieren que haya ningún software especial instalado en el servidor. JavaScript permite añadir algunas funciones básicas e interactividad a las páginas web, pero tiene sus limitaciones, lo que nos lleva a hablar de lenguajes de programación del lado del servidor y a páginas web dinámicas.

2.2.3. Páginas web dinámicas

En ocasiones, cuando navegáis por Internet os encontraréis páginas web que no tienen una extensión .html: es posible que tengan una extensión .php, .asp, .aspx, .jsp, u otras extensiones extrañas. Todos son ejemplos de tecnologías web dinámicas que se pueden utilizar para crear páginas web que tengan secciones dinámicas: código que muestra diferentes resultados según los valores que reciba, por ejemplo, de una base de datos, de un formulario o de otra fuente de datos. Trataremos estos tipos de páginas web en el subapartado “Páginas web estáticas o dinámicas”, a continuación.

2.2.4. Formatos que requieren otras aplicaciones o conectores

Como los navegadores web sólo están equipados para interpretar y mostrar determinadas tecnologías como estándares web, si solicitáis un URL que apunta a un formato de archivo complejo o a una página web que contiene una tecnología que requiere conectores (*plugins*), se descargará en vuestro ordenador o se abrirá utilizando el conector necesario si el navegador lo tiene instalado.

Ejemplos de conectores

Si os encontráis un documento de Word, de Excel, PDF, un fichero comprimido (por ejemplo, ZIP o SIT), un fichero de imágenes complejas como Photoshop PSD, u otro fichero complejo que el navegador no comprende, generalmente el navegador os preguntará si deseáis descargar o abrir el fichero. Ambas acciones normalmente tienen resultados similares, excepto que la última hará que el fichero se descargue y a continuación se abra con una aplicación que lo comprende, si está instalada.

Si os encontráis una página que contenga una película Flash, MP3 u otro formato de música, MPEG u otro formato de vídeo, el navegador lo reproducirá utilizando un conector, si hay uno instalado. En caso contrario, se proporcionará un enlace para instalar el conector necesario, o el archivo se descargará y buscará una aplicación de escritorio para ejecutarlo.

Naturalmente, hay algunas áreas grises: por ejemplo, SVG (*scalable vector graphics*, gráficos vectoriales escalables) es un estándar web que se ejecuta de manera nativa en algunos navegadores, como Opera, pero no en otros, como las versiones anteriores a la 9 de Internet Explorer. Esas versiones de IE necesitan un conector para comprender los SVG. Hay una serie de navegadores que incluyen algunos conectores previamente instalados, de modo que es posible que no seáis conscientes de que el contenido se está visualizando mediante un conector y de manera no nativa en el navegador.

2.3. Páginas web estáticas o dinámicas

Así pues, ¿qué son las páginas web estáticas y dinámicas y cuál es la diferencia entre ambas? Como en una caja de bombones, todo se basa en el relleno.

Una **página web estática** es una página web donde el contenido, el HTML y los gráficos, son siempre estáticos –se sirve a cualquier visitante de la misma manera, a no ser que la persona que ha creado la web decida cambiar manualmente su copia en el servidor–, exactamente lo que hemos estado repasando en la mayor parte de este apartado.

Por el contrario, en una **página web dinámica**, el contenido del servidor es el mismo, pero en vez de ser sólo HTML, también contiene código dinámico, que puede mostrar datos diferentes según la información que suministre a la página web.

Página web dinámica

Podéis ver un ejemplo de página web dinámica: id a Amazon* con vuestro navegador web y buscad cinco productos diferentes. Amazon no os ha enviado cinco páginas diferentes, os ha enviado la misma página cinco veces, pero con diferente información dinámica completada cada vez. Esta información diferente se guarda en una base de datos, que entrega la información correspondiente cuando se solicita, y la envía al servidor web para insertarla en la página dinámica.

* www.amazon.com

Otra cosa que cabe tener en cuenta es que se debe instalar un software especial en el servidor para crear una página web dinámica. Mientras que los ficheros HTML estáticos normales se guardan con una extensión de fichero .html, estos ficheros contienen código dinámico especial además del HTML y se guardan con extensiones de archivo especiales para indicarle al servidor web que necesitan un procesamiento adicional antes de enviarlos al cliente (como, por ejemplo, que se inserten los datos desde la base de datos); los archivos PHP, por ejemplo, generalmente tienen una extensión de archivo .php.

Hay muchos lenguajes dinámicos que se pueden elegir: el PHP que hemos mencionado antes y otros como Python, Ruby on Rails, ASP.NET y Coldfusion. En definitiva, todos estos lenguajes tienen más o menos las mismas capacidades, como hablar con bases de datos, validar la información introducida en los formularios, etc., pero hacen las cosas de manera ligeramente diferente y tienen algunas ventajas e inconvenientes. Todo se reduce a la forma más sencilla que mejor se adapte.

Resumen

Hasta aquí el recorrido por la sala de máquinas de Internet. Este apartado realmente sólo trata de pasada muchos de los temas que incluye, pero resulta útil porque los pone en perspectiva entre ellos y muestra cómo se relacionan y funcionan entre sí. Todavía queda mucho por aprender sobre la sintaxis real del lenguaje que conforma el HTML, el CSS y JavaScript, y esto es lo que haremos a continuación: el apartado siguiente se centra en el modelo de estándares web HTML, CSS y JavaScript de desarrollo web, y da un vistazo al código de la página web.

Preguntas de repaso

1. Realizad una breve descripción de HTML y HTTP y explicad la diferencia entre los dos.
2. Explicad la función de un navegador web.
3. Navegad por Internet durante 5-10 minutos e intentad encontrar algunos tipos diferentes de contenidos: texto normal, imágenes, HTML, páginas dinámicas como páginas PHP y .NET (.aspx), PDF, documentos de Word, películas Flash, etc. Accedid a algunos de estos contenidos y pensad cómo os los muestra el ordenador.
4. ¿Cuál es la diferencia entre una página estática y una página dinámica?
5. Encontrad una lista de códigos de error HTTP, enumerad cinco y explicad qué significa cada uno de ellos.

Lecturas complementarias

En esta asignatura ya no volveremos a hablar de los lenguajes dinámicos, pero hemos creado una lista de recursos en caso de que queráis leerlos:

Rails: Fernandez, Obie. (2007). *The Rails Way*. Addison-Wesley Professional Ruby Series.

Rails screencasts

<http://www.rubyonrails.org/screencasts>

PHP: Powers, David (2006). *PHP Solutions: Dynamic web development made easy, friends of ED*.

Documentación en línea de PHP

<http://www.php.net/docs.php>

ASP.NET: Lorenz, Patrick. (2003). *ASP.NET 2.0 Revealed*. Apress.

ASP.NET: documentación y guías en línea de ASP.NET

<http://asp.net/>

3. El modelo de estándares web: HTML, CSS y JavaScript

Jonathan Lane

En el último apartado hemos visto brevemente las herramientas de construcción básicas de la web: HTML (o XHTML), CSS y JavaScript. Ahora es el momento de ampliar un poco más y observarlos individualmente: qué hacen y cómo interaccionan entre ellos para crear una página web.

3.1. ¿Por qué separar?

Ésta es, generalmente, la primera pregunta que se formula sobre los estándares web. Se puede crear el contenido, el estilo y el formato sólo utilizando HTML: elementos de tipo de letra para el estilo y tablas HTML para el formato, de manera que, ¿por qué preocuparse de este asunto de XHTML/CSS? El uso de tablas para el formato, etc., es como se solía hacer en los malos tiempos del diseño web, y mucha gente todavía lo hace de esta manera (aunque no se debería de hacer) y, de hecho, es uno de los motivos por los que hemos creado esta asignatura. Aquí no trataremos estos métodos. Éstas son las razones más importantes para utilizar CSS y HTML en vez de métodos obsoletos:

1) **Eficiencia del código:** cuanto mayores sean los archivos, más tardarán en descargarse y más dinero le costará a algunas personas (algunas personas todavía pagan por megabyte descargado, y las tarifas móviles acostumbran a tener límites estrictos). Por lo tanto, no se debe malgastar el ancho de banda con páginas grandes abarrotadas de información de estilo y de formato en cada archivo HTML. Una alternativa mucho mejor es que los archivos HTML estén desglosados y limpios, e incluir la información de estilo y de formato sólo una vez en un archivo CSS independiente o en varios.

2) **Facilidad de mantenimiento:** en relación con el último punto, si la información de estilo y formato sólo se especifica en un sitio, quiere decir que sólo habrá que hacer actualizaciones en un lugar si se quiere cambiar el aspecto de la página web. ¿Preferiríais tener que actualizar esta información en cada página de vuestra web? No lo creo.

3) **Accesibilidad:** los usuarios de la web con problemas visuales pueden utilizar un tipo de software conocido como “lector de pantalla” para acceder a la información mediante el sonido en vez de la vista; literalmente, les lee la página. Además, el software de introducción de datos mediante la voz que utilizan las personas con problemas de movilidad también se beneficia de las páginas web con una semántica bien construida. De manera muy parecida al lector de pantalla que utiliza las instrucciones del teclado

Eficiencia del código

Para ver un caso real de este hecho en acción, podéis ver el artículo sobre la reescritura de Slashdot*, en A List Apart, donde el autor tomó una página web muy popular y la reescribió en XHTML/CSS.

* <http://www.alistapart.com/articles/slashdot>

para navegar por los encabezamientos, enlaces y formularios, un usuario que interaccione mediante voz utilizará instrucciones con su voz. Los documentos web marcados semánticamente, en vez de presentacionalmente, pueden resultar más fáciles de navegar y la información que contienen es más fácil de encontrar por parte del usuario. En otras palabras, cuanto más rápidamente “entre en materia” (el contenido), mejor. Los lectores de pantalla no pueden acceder al texto dentro de imágenes y encuentran confusos algunos usos de JavaScript. Aseguraos de que el contenido más importante está disponible para todo el mundo.

4) Compatibilidad de dispositivos: ya que vuestra página XHTML es simplemente etiquetado normal, sin información de estilo, se puede reformatear para diferentes dispositivos con atributos ampliamente variados (por ejemplo, el tamaño de pantalla), simplemente aplicando una hoja de estilos alternativa; podéis hacerlo de varias maneras. Podéis ver los artículos de [dev.opera.com*](http://dev.opera.com/articles/mobile/) para obtener recursos al respecto. CSS también permite especificar hojas de estilo diferentes a nivel nativo para distintos métodos de presentación o tipos de soporte (por ejemplo, visualización en pantalla, impresión, visualización en un dispositivo móvil, etc.).

* <http://dev.opera.com/articles/mobile/>

5) Motores de búsqueda en la web: probablemente os interesará que vuestras páginas web sean fáciles de encontrar al buscar en Google o en otros motores de busca. Un motor de búsqueda utiliza un *crawler* o araña web, que es una pieza de software especializada para leer vuestras páginas. Si esta araña tuviera problemas para encontrar el contenido de vuestras páginas o interpretara mal lo que es importante porque no habéis definido los encabezamientos como encabezamientos, etc., entonces seguro que vuestra posición en los resultados de busca se verá afectada.

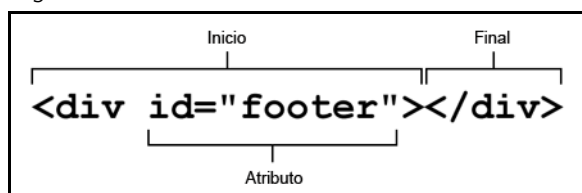
6) Es simplemente una **buena práctica**: se trata de un motivo un poco del tipo “porque lo digo yo”, pero hablad con cualquier desarrollador o diseñador web profesional consciente de los estándares y seguro que os dice que separar el contenido, el estilo y el comportamiento es la mejor manera de desarrollar una aplicación.

3.2. Etiquetado, la base de cada página

HTML y XHTML son lenguajes de etiquetado formados por elementos que contienen atributos (algunos opcionales y otros obligatorios). Estos elementos se utilizan para etiquetar los diferentes tipos de contenidos en los documentos, especificando que cada trozo de contenido se debe entregar supuestamente como en los navegadores web (por ejemplo, encabezamientos, párrafos, tablas, listas con viñetas, etc.).

Como sería de esperar, los **elementos** definen el tipo de contenido actual, mientras que los **atributos** definen información adicional sobre estos elementos, como una ID para identificar el elemento o una ubicación para que le señale un enlace. Deberíais tener en cuenta que se supone que el etiquetado es el más semántico posible, es decir, se supone que debe describir la función del contenido de la manera más cuidada posible. La figura 1 muestra la anatomía de un elemento (X)HTML característico.

Figura 1



Anatomía de un elemento (X)HTML.

Teniendo esto en cuenta, ¿cuál es la diferencia entre HTML y XHTML?

3.2.1. ¿Qué es el XHTML?

La X de XHTML quiere decir “extensible”, es decir, ampliable. Una de las preguntas más habituales para los que empiezan es: “¿he de utilizar HTML o XHTML, y qué diferencia hay?”. Son casi lo mismo; la principal diferencia radica en la estructura. Podéis consultar la tabla 1 para ver las diferencias.

Tabla 1. Diferencias entre HTML y XHTML

HTML	XHTML
Los elementos y atributos no distinguen entre mayúsculas y minúsculas, <h1> es lo mismo que <H1>.	Los elementos y atributos distinguen entre mayúsculas y minúsculas; todos están en minúsculas.
Algunos elementos no necesitan una etiqueta de cierre (por ejemplo, párrafos, <p>), mientras que otros (denominados <i>elementos vacíos</i>) prohíben la marca de cierre (por ejemplo, las imágenes,).	Todos los elementos se deben cerrar claramente (por ejemplo, <p>A paragraph</p>). Los elementos sin contenido se pueden cerrar utilizando una barra inclinada en la marca inicial (por ejemplo, <hr></hr> y <hr/> significan lo mismo). Si servís el texto XHTML como text/html, deberéis utilizar la sintaxis abreviada en todos los elementos que están definidos como “vacíos” ^{**} y colocar un espacio antes de la barra inclinada. Deberíais utilizar la forma larga (con marcas inicial y final independientes) en cualquier elemento que no esté definido como vacío, aunque no tenga ningún contenido.
Algunos valores de atributos pueden estar escritos sin estar entre comillas.	Los valores de los atributos se han de incluir entre comillas.
Algunos atributos se pueden abreviar (por ejemplo, <option selected>).	Se debe utilizar la forma de atributo entera para todos los atributos (por ejemplo, <option selected="selected">).
Los servidores habrían de servir el HTML al cliente con un tipo de medio text/html.	El XHTML debería utilizar el tipo de medio application/xhtml+xml pero se puede utilizar application/xml, text/xml o text/html. Si se utiliza text/html, se deberían seguir las directrices de compatibilidad de HTML ^{**} porque los navegadores lo tratarán como HTML (y utilizar la recuperación de errores para representar las diferencias entre idiomas).

* <http://www.cs.tut.fi/~jkorpela/html/empty.html#html>

** <http://www.w3.org/TR/xhtml1/guidelines.html>

Por ahora, os recomendamos que no os preocupéis mucho sobre si estáis escribiendo HTML o XHTML. Seguid los consejos que se dan a lo largo de esta asignatura y utilizad el tipo de documento HTML e iréis por el buen camino.

Para obtener más información sobre tipos de documentos HTML, podéis consultar el apartado 3 del módulo "Fundamentos de HTML".

3.2.2. Validación, ¿qué es eso?

Como HTML y XHTML son estándares establecidos (y CSS también, en realidad), el World Wide Web Consortium (W3C) ha creado una gran herramienta denominada **validador** que comprueba automáticamente las páginas que queráis y señala cualquier problema-error que pueda tener vuestro código, como la falta de etiquetas de cierre o la falta de comillas alrededor de los atributos.

Encontraréis más información sobre la validación en el apartado 6 del módulo "El texto central de HTML". También encontraréis más información sobre los tipos de documentos en el apartado 3 del módulo "Fundamentos de HTML".

- El validador de HTML está disponible en línea en <http://validator.w3.org/>. Detectará automáticamente si estáis utilizando HTML o XHTML y qué tipo de documento estáis usando.
- Si queréis comprobar el CSS, el validador está disponible en la red.*

* <http://jigsaw.w3.org/css-validator/>

3.3. CSS: añadimos un poco de estilo

Las **hojas de estilo en cascada** permiten un control preciso sobre el formato y la disposición del documento. Podéis cambiar o añadir colores, fondo, tipo de letra, tamaños y estilos de tipo de letra, e incluso la posición de elementos de vuestra página web en diferentes lugares.

Hay tres maneras básicas de aplicar estilos mediante CSS: redefinición de un elemento, aplicación de un estilo a una ID o aplicación de un estilo a una clase. Echemos un vistazo a cada uno de ellos:

1) **Redefinición de un elemento.** Podéis cambiar la forma en la que cualquier elemento de (X)HTML se muestra mediante la definición de una regla de estilo.

Ejemplo de redefinición de un elemento

Si queréis que todos los párrafos sean a doble espacio y en verde, podéis añadir esta declaración en CSS:

```
p {  
  line-height: 2;  
  color: green;  
}
```

Ahora, todo el contenido incluido entre las etiquetas `<p></p>` tendrá una altura de línea doble y será de color verde.

2) Definición de una ID. Podéis darle a un elemento un atributo id para identificarlo de manera única en una página (cada ID se puede utilizar sólo una vez por página), por ejemplo `id="navigation_menu"`. Esto os permite un control más preciso sobre el formato de una página.

Ejemplo de definición de una ID

Si sólo queréis que un párrafo determinado sea a doble espacio y destacado con texto verde, asignad una ID.

```
<p id="highlight">Paragraph content</p>
```

Y entonces aplicadle una regla CSS, tal como se indica a continuación:

```
#highlight {  
  line-height: 2;  
  color: green;  
}
```

Esto sólo aplicará la regla CSS al párrafo de la página con un atributo id del tipo highlight (el símbolo de almohadilla es sólo una convención de CSS para indicar que se trata de una ID).

3) Definición de una clase. Las clases son como las ID, excepto que se puede tener más de un elemento de la misma clase en cada página.

Ejemplo de definición de una clase

Siguiendo con nuestro ejemplo de doble espacio, si queréis que los dos primeros párrafos de una página sean a doble espacio y estén destacados, les habríais de añadir clases como:

```
<p class="highlight">Paragraph content</p>  
<p class="highlight">The content of the second paragraph</p>
```

Y entonces aplicadles una regla CSS como se indica a continuación:

```
highlight {  
  line-height: 2;  
  color: green;  
}
```

En este caso, highlight es una clase, no una ID: el punto sólo es una convención de CSS para indicar que se trata de una clase.

El ejemplo siguiente os dará una idea mejor de cómo CSS aplica estilos a HTML.

Empezaremos a ver el CSS con más detalle en el apartado 8 del módulo "El texto central de HTML".

3.4. JavaScript: adición de comportamiento a las páginas web

Por último, JavaScript es el lenguaje de *script* que se utiliza para añadir comportamiento a sus páginas web.

JavaScript se puede utilizar para validar los datos que se introducen en un formulario (discriminar si están en el formato correcto o no), para ofrecer la funcionalidad de arrastrar y soltar, para cambiar estilos sobre la marcha, para animar elementos de las páginas como los menús, para tratar las funciones de los botones y un millón de cosas más.

La mayoría del JavaScript moderno funciona localizando un elemento HTML de destino y después haciéndole algo, igual que el CSS, pero la manera de funcionamiento, la sintaxis, etc., es bastante diferente.

El JavaScript es un tema más complicado y extenso que el HTML y el CSS, de modo que para poner las cosas simples y evitar confusiones tan pronto no lo incluimos en el ejemplo siguiente.

De hecho, no volveréis a ver el JavaScript hasta una asignatura posterior.

3.5. Una página de ejemplo

Hay muchos detalles que no hemos incluido, pero lo trataremos todo durante esta asignatura de diseño web. Por ahora, os presentaremos una página real de ejemplo, sólo para que veáis una pequeña muestra de aquello con lo que trabajaréis durante el resto de apartados.

El ejemplo que presentamos a continuación es una página de referencias que se puede utilizar para citar referencias al final de, pongamos por caso, un ensayo psicológico sobre la dinámica de grupo de un equipo de desarrollo web, o un informe para trabajar sobre el uso de Internet de banda ancha en Estados Unidos. Tened en cuenta que si sois muy estrictos con la redacción académica canónica, este ejemplo muestra el formato APA (tenía que elegir uno). Descargaos el código*.

* http://mosaic.uoc.edu/ac/le/operafiles/article4_examples.zip

3.5.1. index.html

```
1  <!DOCTYPE html>
2
3  <html>
4  <head>
5    <meta charset="utf-8" />
6    <title>References</title>
7    <style type="text/css">
8      @import url("styles.css");
9    </style>
10 </head>
```



```

11 <body>
12 <div id="bggraphic"></div>
13 <div id="header">
14 <h1>References</h1>
15 </div>
16 <div id="references">
17 <cite class="article">Adams, J. R. (2008). The Benefits of Valid Markup: A Post-Modernistic
    Approach to Developing Web Sites. <em>The Journal of Awesome Web Standards, 15:7,</em>
    57-62.</cite>
18 <cite class="book">Baker, S. (2006). <em>Validate Your Pages.... Or Else!</em> Detroit,
    MI: Are you out of your mind publishers.</cite>
19 <cite class="article">Lane, J. C. (2007). Dude, HTML 4, that's like so 2000. <em>The
    Journal that Publishes Genius, 1:2, </em> 12-34.</cite>
20 <cite class="website">Smith, J. Q. (2005). <em>Web Standards and You.</em> Retrieved
    May 3, 2007 from <a href="http://www.webstandardsandyou.com/">Web standards and you</a>
    .</cite>
21 </div>
22 <div id="footer">
23 <p>The content of this page is copyright © 2007
24 <a href="mailto:jonathanlane@gmail.com">J. Lane</a></p>
25 </div>
26 </body>
27 </html>

```

No haremos una disección de este archivo línea por línea, ya que veréis muchos ejemplos en futuros apartados; sin embargo, a continuación se indican algunos elementos importantes que cabe tener en cuenta.

La línea 1 es lo que se denomina *declaración de tipo de documento* o *doctype*. En este caso, la página es "HTML5" (aunque no explota todas las características de HTML5). El tipo de documento especifica una serie de reglas que debe seguir el etiquetado y gracias a las que se puede validar.

Podéis ver el apartado 3 del módulo "Fundamentos de HTML" para encontrar más información sobre tipo de documentos.

Las líneas 5 a 7 importan un archivo CSS a la página: los estilos contenidos en este archivo se aplicarán a los diferentes elementos de la página. En el siguiente subapartado veréis el contenido del archivo CSS que trata todo el formato de la página.

Hemos asignado una clase diferente a cada uno de los diferentes tipos de referencia. Hacerlo de esta manera nos permite aplicar un estilo diferente a cada tipo de referencia, así, en nuestro ejemplo hemos puesto un color diferente a la derecha de cada referencia para que sea más fácil analizar la lista.

Ahora demos un vistazo al CSS que aplica estilo al HTML.

3.5.2. styles.css

```

body{
  background: #fff url('images/gradbg.jpg') top left repeat-x;
  color: #000;
  margin: 0;
  padding:0;
  border: 0;
  font-family: Verdana, Arial, sans-serif; font-size: 1em;
}

```

```
#references cite {
  margin: 1em 0 0 3em;
  text-indent: -3em;
  display: block;
  font-style: normal;
  padding-right: 3px;
}

.website {
  border-right: 5px solid blue;
}

.book {
  border-right: 5px solid red;
}

.article {
  border-right: 5px solid green;
}

#footer {
  font-size: 0.5em;
  border-top: 1px solid #000;
  margin-top: 20px;
}

#footer a {
  color: #000;
  text-decoration: none;
}

#footer a:hover{
  text-decoration: underline;
}

body{
  background: #fff url('images/gradbg.jpg') top left repeat-x;
  color: #000;
  margin: 0;
  padding:0;
  border: 0;
  font-family: Verdana, Arial, sans-serif; font-size: 1em;
}

div {
  width: 800px;
  margin: 0 auto;
}

#bggraphic {
  background: url('images/pen.png') top left no-repeat;
  height: 278px;
  width: 362px;
  position: absolute;
  left: 50%;
  z-index: 100;
}
```

Hemos exagerado un poco con el estilo de esta página y hemos añadido algunos bonitos efectos de fondo para mostraros algunas de las cosas que se pueden conseguir con el CSS.

La línea 1 establece algunos valores predeterminados para el documento, como el color del texto y del fondo, el ancho del borde que se debe añadir en torno al texto, etc. Algunas personas no se preocuparán de indicar explícita-

mente valores predeterminados como éstos, y la mayoría de los navegadores modernos aplicarán sus propios valores predeterminados, pero es una buena idea, ya que le permite un control mayor sobre cómo se verá vuestra página web en diferentes navegadores.

En la línea siguiente hemos establecido la anchura de página en 800 px (aunque en este caso podríamos haber especificado un porcentaje para que la página se expanda/contraiga en función del tamaño de la ventana del navegador). La configuración de márgenes que hemos utilizado garantizará que el contenido de la página siempre se mantenga centrado en la ventana.

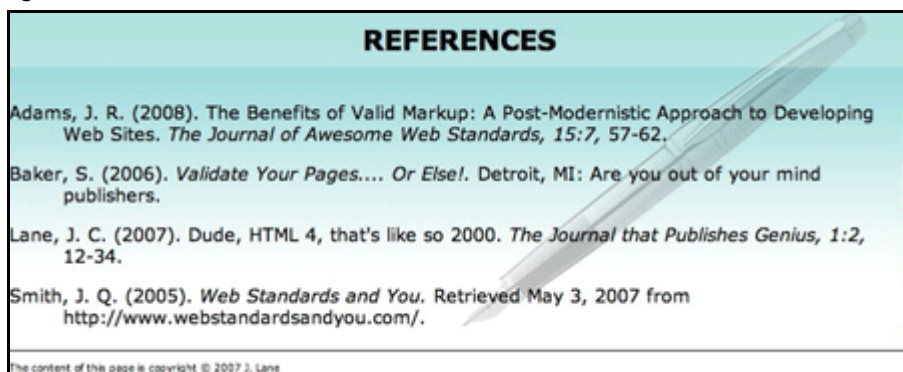
Las imágenes PNG semitransparentes no funcionan en Internet Explorer 6 o versiones anteriores, pero funcionan prácticamente en todos los navegadores modernos; podéis consultar la corrección de JavaScript para IE* por parte de Dean Edward para una solución para IE 6 a este problema, que también resuelve algunos de los problemas de compatibilidad con CSS de IE 6.

* <http://code.google.com/p/ie7-js/>

Pasemos ahora a fijarnos en las imágenes de fondo utilizadas en la página (se aplican utilizando las declaraciones de fondo: url). En esta página tenemos 3 elementos de fondos diferentes. El primero es una gradación de azul a blanco que empieza desde la parte superior de la página. En segundo lugar, hemos utilizado un PNG semitransparente para el gráfico de la pluma para crear un poco de contraste con el texto que tiene encima y respetar la gradación. Por último, hemos utilizado otro PNG para el fondo del encabezamiento de página. Le da al título un poco más de contraste y ofrece un efecto de calidad.

El ejemplo queda como se ve en la figura 2.

Figura 2



El ejemplo acabado con los estilos aplicados.

Resumen

XHTML, CSS y JavaScript no tienen nada de complicado. Son simplemente una evolución de la web. Si ya habéis tenido un poco de contacto con HTML, lo podéis aprovechar perfectamente. Todo lo que sabéis sigue siendo válido, sólo hay que tratar algunos conceptos de manera diferente (y tener un poco más de cuidado con el etiquetado).

Aparte de tener la satisfacción de un trabajo bien hecho, el desarrollo de los estándares web es totalmente lógico. Los argumentos en contra del desarrollo con estándares son que se debe invertir mucho tiempo y es muy pesado hacer que un diseño funcione en distintos navegadores. Se podría utilizar el argumento contrario para hacer que un formato no basado en estándares funcione en diferentes dispositivos y con futuras versiones de navegadores. ¿Cómo se puede tener la certeza de que un etiquetado que se sostiene con agujas se verá bien en Opera 12.0, Firefox 5.0 e IE 10.0? No se puede, a no ser que se sigan algunas reglas.

Preguntas de repaso

1. ¿Cuál es la diferencia entre una clase y una ID?
2. ¿Para qué sirven XHTML, CSS y JavaScript en una página web?
3. Tomad el archivo index.html del ejemplo y cambiad el formato utilizando sólo CSS (os recomendamos editar el archivo utilizando un editor de texto como el Bloc de Notas o Text Wrangler). No hagáis ningún cambio al HTML.
 - a) Añadid un icono para cada uno de los diferentes tipos de referencia (un icono diferente para artículos, libros y recursos web). Generad vuestros propios iconos para este propósito y haced que aparezcan junto con los diferentes tipos de referencia utilizando sólo CSS.
 - b) Ocultad el aviso de *copyright* de la parte inferior de la página.
 - c) Cambiad el aspecto del título, conseguid que destaque.
4. ¿Qué tipo de cosas podríais hacer con CSS para lograr que este ejemplo funcione bien con un navegador de teléfono móvil?

4. Estándares web: un bonito sueño, pero ¿cuál es la realidad?

Jonathan Lane

Hasta ahora, hemos hablado sobre el maravilloso ideal de los estándares web. Los estándares web permiten una interoperabilidad entre todos los navegadores web, en todos los sistemas operativos e incluso en todos los dispositivos electrónicos disponibles. Pero ¿es ésta la realidad? ¿Todos los navegadores cumplen los estándares al 100%? ¿Están utilizando adecuadamente los estándares web todos los desarrolladores web? ¿Construyen los desarrolladores web una página utilizando estándares web y ya se quedan tranquilos pensando que su diseño funcionará en todas partes?

La respuesta realmente sencilla a esta última pregunta es no; aunque se trata de una situación ideal, está muy alejada de la realidad.

4.1. ¿Cómo se comprueba el cumplimiento de los estándares web?

Antes de continuar, la pregunta que probablemente os esté pasando por la cabeza es: ¿cómo se sabe si una página web utiliza estándares web? ¿Tiene un aspecto diferente a cualquier otra página web? Sí y no. Las páginas web que cumplen los estándares web, si están correctamente desarrolladas, no deben parecer diferentes de las páginas web codificadas mediante un etiquetado que es un batiburrillo o que se sujeta con pinzas. No obstante, el código fuente de la página web puede parecer bastante diferente (intentad hacer clic con el botón secundario del ratón o pulsando la tecla Ctrl sobre una página web y seleccionad la opción “Código fuente” o “Ver el código fuente”; la terminología exacta depende del navegador).

Una página web que cumpla los estándares tendrá un etiquetado ordenado y claro con poco o sin formato incrustado en la misma página. Es posible que os resulte difícil notarlo a primera vista pero, creedme, las personas con dificultades visuales que utilizan lectores de pantalla y motores de búsqueda lo notarán enseguida. En el último apartado ya hemos hablado de las ventajas de utilizar los estándares web.

La manera más sencilla de comprobar la compatibilidad de los estándares es utilizar una práctica herramienta, disponible en línea, denominada **validador**.

- El World Wide Web Consortium (W3C) tiene disponible el validador gratuitamente* (podéis consultar la figura 1). Podéis (y sería necesario) utilizar esta

* <http://validator.w3.org/>

herramienta para comprobar errores de HTML o XHTML en cualquier página web que estéis desarrollando.

- El CSS se puede comprobar utilizando el validador de CSS*.

* Disponible en
<http://jigsaw.w3.org/css-validator/>

Visitad libremente estos enlaces y probad algunas de las páginas web que más os gusten.

Figura 1

El servicio de validación de etiquetado del W3C comprueba vuestras páginas e indica cualquier error de etiquetado.

Validar siempre las páginas que se crean es sólo la mitad del proceso. ¿Cómo podemos comprobar si los navegadores cumplen los estándares? El Proyecto de estándares web ha desarrollado una serie de pruebas denominadas pruebas Acid que utilizan algunas reglas complejas de HTML y CSS (además de otro etiquetado y código), para ver si un navegador puede reproducir correctamente diferentes pantallas de prueba. La última versión de la prueba Acid está disponible en acidtests.org.

Podéis leer más cosas sobre las pruebas Acid en la página web de acidtests.org* y también podéis visitar las páginas de prueba reales para poner a prueba vuestro navegador.
*<http://www.acidtests.org/>

4.2. Compatibilidad de los estándares en las páginas actualmente

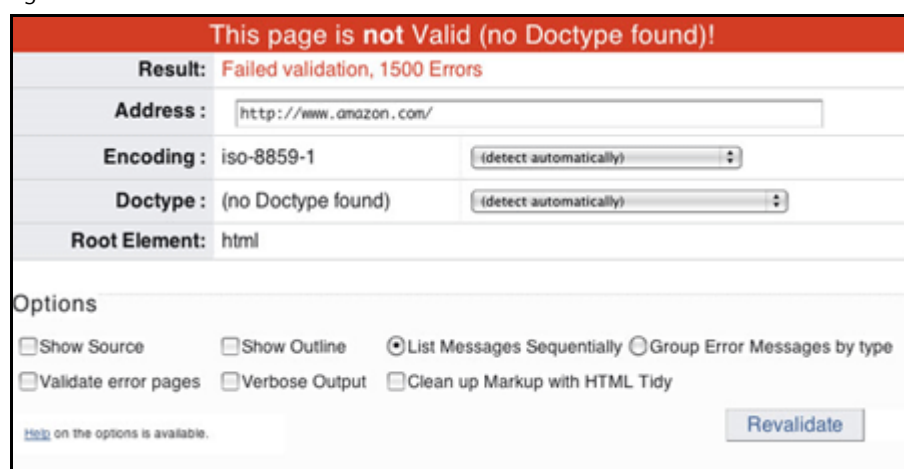
¿Utilizan las principales webs los estándares o simplemente recogen cuatro cosas y ya está? Echemos una ojeada a unas cuantas empresas y veamos qué puntuación reciben mediante el servicio de validación de etiquetado del W3C. Os sorprendería cuántas grandes páginas web no pasan las pruebas de validación de estándares; no os desaniméis porque no hay motivo para no poder hacerlo mejor y que vuestras páginas sí se validen correctamente. Cuando leáis los si-

guientes informes, tened en cuenta que cuanto mayor y complicada sea la página web, más difícil será validarla, en general (hay otros factores que se deben tener en cuenta, como las tecnologías utilizadas).

4.2.1. Amazon: ¿comprar con estándares?

Es posible que si alguna vez habéis realizado alguna compra en línea hayáis visitado Amazon.com (o una de las webs específicas por país). Amazon es un megaalmacén del ciberespacio que ofrece de todo, desde libros hasta CD e incluso alimentos en algunas zonas. Ahora bien: ¿pasaría Amazon.com la prueba de validación? Podéis comprobarlo en la figura 2.

Figura 2



The screenshot shows a web validation tool interface. At the top, a red banner reads "This page is not Valid (no Doctype found)!". Below this, the "Result" is "Failed validation, 1500 Errors". The "Address" field contains "http://www.amazon.com/". The "Encoding" is "iso-8859-1" with a dropdown menu set to "(detect automatically)". The "Doctype" is "(no Doctype found)" with a dropdown menu set to "(detect automatically)". The "Root Element" is "html". Under the "Options" section, there are several checkboxes: "Show Source" (unchecked), "Show Outline" (unchecked), "List Messages Sequentially" (checked), "Group Error Messages by type" (unchecked), "Validate error pages" (unchecked), "Verbose Output" (unchecked), and "Clean up Markup with HTML Tidy" (unchecked). A "Revalidate" button is at the bottom right. A small link "Help on the options is available." is at the bottom left.

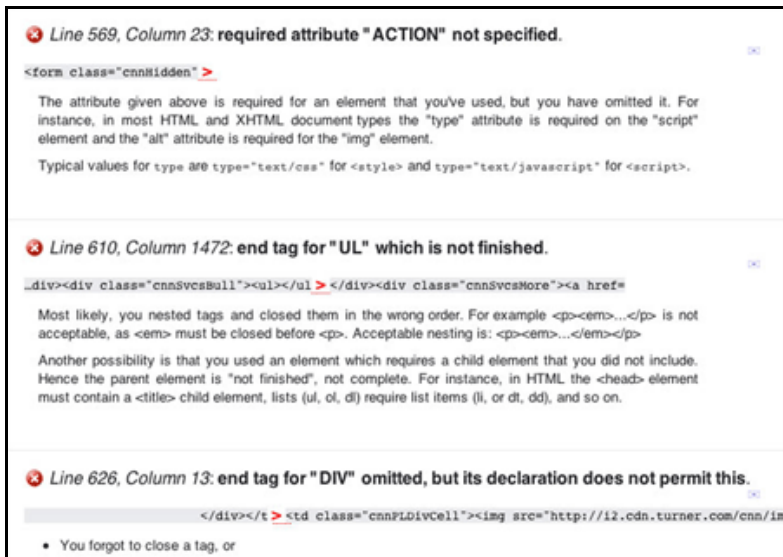
¡Amazon.com falla estrepitosamente! No sólo tiene etiquetado no válido, sino que ni siquiera declaran un tipo de documento (explicando qué versión de HTML o XHTML utilizan).

A Amazon todavía le queda mucho por hacer con respecto a compatibilidad con los estándares. Desconocemos qué sucede en Amazon, pero si se nos permite especular un poco, diríamos que, teniendo en cuenta que Amazon ya hace tiempo que funciona, probablemente han utilizado el mismo software para su página web durante toda su existencia. Como los estándares web no han recibido la suficiente atención hasta comienzos de este siglo, es muy probable que el sistema que utiliza Amazon para vender productos en línea se desarrollara mucho antes, cuando los estándares web no eran realmente aceptados ni publicitados. No lo sabemos a ciencia cierta, pero intuimos que Amazon se encuentra en el caso de aquellos que se mantienen con lo que les funciona.

4.2.2. CNN: ¿noticias estandarizadas?

¿Seguro que las organizaciones de noticias son entes semánticos? CNN.com es una de las páginas web de medios mayor de todo el mundo. Al tener recursos globales e informar sobre las noticias a medida que suceden, seguramente tendrán un equipo de especialistas internos para garantizar que su web tenga un etiquetado válido, ¿no? Podéis comprobarlo en la figura 3.

Figura 3



CNN.com (el 15 de abril del 2008) no supera la validación con 33 errores. Se menciona un tipo de documento HTML 4.01 transicional, pero gran parte de su etiquetado se parece mucho a XHTML.

33 errores no está mal, tratándose de una web de dimensiones y complejidad como las de CNN. Puede que estos 33 errores se produzcan (y de nuevo estamos especulando) por que el sistema de gestión de contenidos que utiliza no esté del todo preparado para producir etiquetado que cumpla los estándares, o podría tratarse de una colección de errores de etiquetado por parte de un personal de producción que se especializa en redactar noticias, y no en producir webs; cualquier explicación es verosímil.

4.2.3. Apple: el máximo de elegancia en diseño... ¿y en validación?

Apple es famosa por sus productos de hardware y software bonitos y funcionales. Sus anuncios de productos son casi como experiencias religiosas para multitudes de fieles seguidores. La web de Apple* (podéis ver la figura 4) a menudo se considera muy bien diseñada y organizada, sin embargo... ¿pasaría la prueba de validación?

* <http://apple.com/>

Figura 4



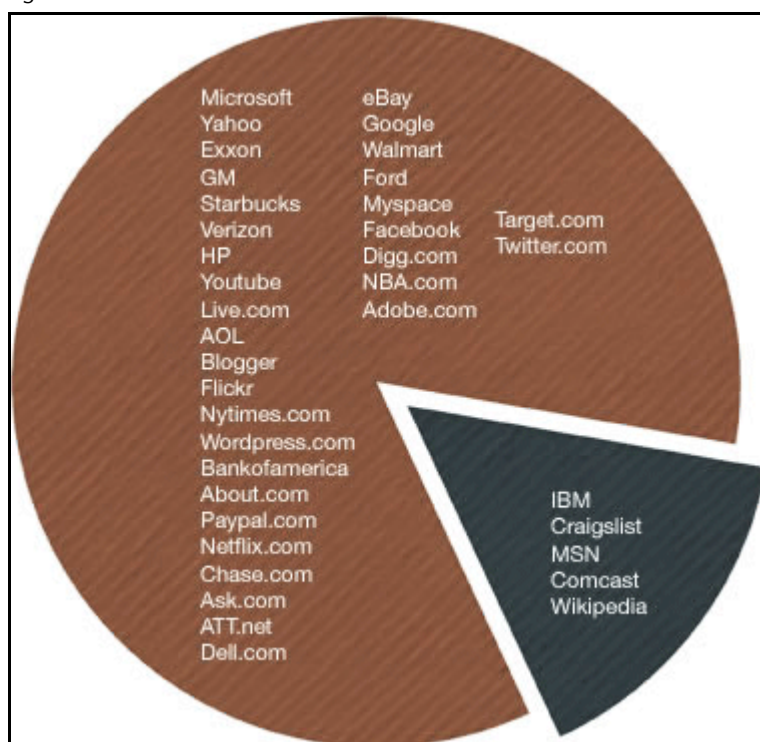
A Apple.com le falta muy poco para tener un etiquetado HTML 4.01 transicional válido. Entre los seis errores que tiene hay una mezcla de errores de etiquetado y un caso de uso de etiquetas específicas de Safari.

La web de Apple se acerca mucho a la validación. Realmente, sólo sería necesario que alguien le dedicara cinco minutos para corregir los errores y hacer que la página supere la prueba de validación. No obstante, el único error que mencionaríamos brevemente es que Apple ha decidido utilizar un atributo específico de Safari en su cuadro de busca (dando al cuadro de busca el atributo `type="search"`). En Safari, esto permitirá ver una lista de búsquedas recientes pinchando sobre un pequeño icono de lupa. En otros navegadores, como Opera o Internet Explorer, se verá sólo como un cuadro de texto normal.

4.2.4. Un pequeño sondeo de compatibilidad de estándares

En vez de ver muchos ejemplos como éste, hemos comprimido el resto de webs sondeadas en una práctica gráfica de tarta. Hemos mirado unas 40 webs corporativas de la lista Fortune 500 y también las clasificaciones Alexa de webs con más tráfico, y en la figura 5 se muestra lo que hemos averiguado:

Figura 5



El 85% de las webs sondeadas no pasaron la validación de ninguna manera. Algunas presentaban fallos espectaculares de hasta mil errores; otras tenían sólo un par de errores aquí y allí.

4.3. ¿Por qué hay tan pocas páginas que cumplan los estándares?

Nos preguntamos: ¿por qué, por qué no validan? Esto puede ser un poco dramático, pero como mínimo tiene un tono similar a la pregunta que os estáis haciendo en este momento. ¿Por qué se validan tan pocas webs? Ya hemos mencionado algunas de las razones posibles –cosas como sistemas de comercio electrónico o sistemas de gestión de contenidos heredados– pero también hay otros motivos subyacentes.

4.3.1. Educación

La facultad en la que estudié tenía un programa de sistemas de información de gestión (*management information systems*), un programa de ciencias informáticas y un programa de nuevos medios, cada uno de los cuales tenía asignaturas relacionadas con la producción de webs pero, aunque enseñaban muchas cosas de manera eficiente, en ninguna de éstas había realmente mucha cobertura sobre cómo codificar realmente una web. La sensación general que tengo cuando reviso muchas asignaturas universitarias es que los lenguajes web como el HTML, el CSS y JavaScript se encuentran bajo el umbral técnico de la mayoría de los programas de ciencias informáticas y por encima del de la mayoría de los programas de MIS/Nuevos medios.

A donde quiero ir a parar en este caso es que muchos cursos educativos no cubren este tipo de materias con mucho detalle. Apostaría a que si le preguntáis a diez desarrolladores que trabajen con estándares web dónde aprendieron a utilizarlos, nueve de ellos responderían que son autodidactas (y el otro no respondería porque tiene demasiado trabajo intentando que su web se visualice bien con el IE 6).

El World Wide Web Consortium (W3C), que es el grupo responsable de desarrollar los estándares, y la Web Standards Project (WaSP) se están tomando seriamente este reto y están presionando realmente para que mejore la aceptación de los estándares web, tanto por parte de los fabricantes de navegadores como por parte de los desarrolladores.

De hecho, el verdadero motivo por el que se creó esta asignatura que estáis leyendo es proporcionar un conjunto adecuado de materiales de aprendizaje para los estándares web y un medio para utilizar este material para aprender, de manera gratuita. En definitiva, intentamos eliminar algunos motivos más (dudamos de si utilizar la palabra *excusas* en este caso...) por los que la gente no está adoptando los estándares web. Realmente no hay excusa para no utilizarlos, teniendo en cuenta las ventajas que suponen (tal como se ha visto en el subapartado 3.1).

4.3.2. Motivos empresariales

Una web que visitamos frecuentemente, destinada a emprendedores implicados en poner en marcha nuevas iniciativas basadas en la web, ha alojado una serie de discusiones sobre el uso de los estándares web en “aplicaciones web 2.0”. Generalmente, existe un intercambio interesante entre aquellos que creen que se deberían utilizar los estándares web porque tiene sentido (por todos los motivos que hemos tratado anteriormente), y aquellos que simplemente dicen que no importa.

El fondo de la cuestión es que los navegadores web interpretarán el código por malo que sea. No es necesario validar vuestras páginas para que se visualicen correctamente en la mayoría de los navegadores. Desde una perspectiva empresarial, donde el tiempo es oro, ¿para qué preocuparse de invertir más tiempo a fin de que se validen? Si podéis arreglar cuatro códigos en forma de tabla en 30 minutos, o pasaros 30 minutos codificando vuestra página en HTML y CSS y 30 minutos para aseguráros de que se valida y funciona correctamente en cualquier navegador, y el resultado final tiene el mismo aspecto en la mayoría de los navegadores web, ¿qué os parece más fácil y rápido?

Mucha gente de mi generación (casi llego a la treintena en el momento de redactar esto) aprendió a crear webs utilizando tablas para el formato y etiquetas de tipo de letra para la tipografía. Puede parecer abrumador volver a aprender a hacer algo cuando lo que estás haciendo todavía “funciona” (todavía se ve bien en la mayoría de los navegadores web). Los empresarios generalmente no distinguen la diferencia; no me he encontrado nunca con que un director me hablara de la calidad de mi etiquetado durante una revisión de rendimiento. De manera que, realmente, ¿cuál es el incentivo?

Pues lo que yo creo es (ya podéis adivinar cuál es mi posición) que el planteamiento del código desordenado o mal hecho tiene muy poca visión de futuro. Según mi experiencia, rediseñar una web basada en estándares es mucho más fácil que tener que convertir un caos de páginas incorrectamente codificadas (he hecho las dos cosas). Todavía debe llegar la utopía que prometen XHTML y CSS en la que sólo se ha de tocar el CSS para rediseñar una web, pero me he acercado mucho.

También hay que tener en cuenta que veréis muchos más anuncios de trabajo actualmente pidiendo conocimientos de estándares web que nunca.

También hay algunos motivos empresariales directos que se deben tener en cuenta. En general, el uso de estándares web mejora la posición de una web en las clasificaciones de los motores de búsqueda (el orden en el que aparecen las webs cuando se realiza una busca de Google) y sobre todo mejora la facilidad de encontrarla. Además, el uso de estándares y buenas prácticas generalmente hará que la página web sea más accesible para las personas con discapacidades que intentan utilizarla y para los usuarios que intentan acceder a la web con teléfonos móviles. Más usuarios y un aumento de la visibilidad siempre es bueno para el negocio.

Resumen

En este apartado hemos hablado sobre el estado actual de la adopción de estándares web, sobre cómo comprobar si los estándares se están utilizando correctamente en una web, cuántos utilizan los estándares web correctamente y

los motivos por los que la gente no adopta los estándares. Como ya habéis visto anteriormente, los motivos no son tan imperativos y deberían ser fáciles de superar.

Así pues, ¿qué debe hacer un desarrollador web emprendedor? ¿Pensáis que son importantes los estándares web (y seguir leyendo esta asignatura), o sois de los que ponéis en marcha un editor gráfico y empezáis a maquetar la web con tablas?

Digámoslo de esta manera: la queja individual mayor que he leído de personas que dicen que el desarrollo basado en estándares es una pérdida de tiempo es que se tarda demasiado en aprender los estándares web en vez de los métodos obsoletos y desarrollar webs que funcionen con todos los navegadores. Entonces, ¿por qué no empezar aprendiendo la manera correcta de hacerlo y ahorrarse unos cuantos problemas? Habéis decidido aprender cómo crear webs y debéis hacerlo de una manera u otra; por lo tanto, ¿por qué no aprender a hacerlo correctamente?

Preguntas de repaso

1. Hemos visto muchas “grandes” webs y si validan o no. Pasad por el validador algunas de las webs que visitáis a menudo. ¿Validan? Si no es así, mirad algunos de los errores para haceros una idea de por qué fallan.
2. ¿Qué es un tipo de documento (*doctype*)? ¿De qué se encarga?
3. ¿Qué argumentos podéis dar a favor de los estándares web para los negocios?

Lecturas complementarias

Servicio de validación de marcado del W3C.

<http://validator.w3.org/>

Web del W3C (con información sobre distintos estándares y recomendaciones para el futuro).

<http://www.w3.org/>

El Proyecto de estándares web

<http://www.webstandards.org/>