Diseño y Programación de Páginas Web

Autor: DesarrolloWeb

Editor: Miguel Ángel Pedregosa Pareja

Índice

Capitulo 1
Publicar en Internet

Tratamos los temas más básicos que necesitas saber para publicar una página web, como los materiales necesarios, el lenguaje HTML, editores, elegir alojamiento, subir páginas a los servidores y mucho más.

Pág. 4

Capitulo 2
Introducción al diseño web

Primeros pasos para aquellas personas que deseen crear su propia página web, desde el punto de vista del diseñador. Motivación a seguir, programas a utilizar, etc.

Pág. 12

Capitulo 3
Manual de HTML

HTML es el lenguaje utilizado como base para crear las páginas web. Con este manual puedes aprender a utilizarlo con toda su potencia.

Pág. 15

Capitulo 4
Ayudas técnicas

Las ayudas técnicas son pequeños reportajes de interés general muy útiles para conocer rápidamente diversos temas de interés.

Pág. 75

Capitulo 5
Introducción a la promoción de webs

Una vez hemos construido una página web tenemos que **hacer que esta sea conocida** por todos los medios que estén a nuestro alcance, para atraer visitas a ella y, cuando menos, sentirnos orgullosos de que esta sea popular en la Red. Para conseguir esto tenemos que promocionarla adecuadamente, de manera que su dirección figure en el mayor número de sitios.

Pág. 115

Capitulo 6
La imagen en Internet

Explica cómo debemos crear la imagen en Internet de las empresas, productos o servicios que deseamos vender. No vale clonar lo anterior; hace falta conjugar estrategia, contenido, diseño y tecnología.

Pág. 146

Capitulo 7
Páginas dinámicas

Introducción al concepto de páginas dinámicas, lenguajes de lado cliente y servidor y otra serie de nociones básicas para lanzarse a la aventura de la programación en ASP o PHP. Este manual sirve de introducción a otros más avanzados.

Pág. 150

Capitulo 8
Introducción a los lenguajes del web

Vamos a estudiar de manera global el mundo de la programación de páginas web. Para ello empezaremos estudiando rápidamente algunos conceptos básicos, que seguramente muchos ya sabremos, como el marco donde la web se desarrolla, qué es una página web, cómo se construye una página y el lenguaje HTML. Además veremos qué es una página estática y dinámica distinguiendo entre páginas dinámicas de cliente y servidor.

Pág. 155

Capitulo 9
Programación en ASP

Principios básicos para la programación en ASP, el lenguaje del lado del servidor creado por Microsoft . Manual asequible para no programadores que sienta los fundamentos básicos de este lenguaje. Continuación lógica del manual de páginas dinámicas.

Pág. 169

Capitulo 10
Programación en PHP

Principios básicos para la programación en PHP, el popular lenguaje del lado del servidor. Manual asequible para no programadores que sienta los fundamentos básicos de este lenguaje. Continuación lógica del manual de páginas dinámicas.

Pág. 187

Capitulo 11
Programación en JavaScript

Descubre el lenguaje dinámico de lado cliente por excelencia. Aprende a crear páginas webs con vida propia con nuestro manual de Javascript.

Pág. 224

Capitulo 12
Programación en Javascript II

En este manual explicamos todos los recursos con los que cuenta un programador de Javascript para crear todo tipo de efectos y aplicaciones.

Pág. 258

Capitulo 13
Tutorial de Visual Basic Script

Manual del lenguaje de scripting de Microsoft para páginas web con el que podrás aprender a realizar efectos para el Internet Explorer.
Explora las características del HTML Dinámico con el lenguaje del navegador más habitual.

Pág. 304

Capitulo 14
CSS, hojas de estilos

Manual completo y práctico sobre hojas de estilo en cascada (CSS). Aprende a utilizar esta tecnología que te ayudará a crear páginas más atractivas y precisas. El curso contiene la descrición, uso, sintaxis, y lista de atributos para crear estilos.

Pág. 320

Capitulo 15
Introducción a XML

Una breve intruducción al mundo XML que explica qué es este lenguaje y sus tecnologías relacionadas.

Pág. 337

Capitulo 16
Tutorial de SQL

Aprende a utilizar el estándar utilizado para la consulta de bases de datos. Seleccionar, crear, modificar y borrar registros. Todo lo que necesitas para la creación de tus páginas dinámicas.

Pág. 342

Apéndice I
Qué es cada tecnología

Este apendice trata en cada uno de sus capítulos de introducir cada una de las tecnologías utilizadas en el desarrollo de páginas web.

Pág. 352

Apéndice II
Frontpage 2000 para principiantes

Un análisis en profundidad de Frontpage y la respuesta a las preguntas más habituales planteadas por los usuarios novatos de este programa.

Pág. 373

Apendice III
Instalación de IIS en Windows XP profesional

Descripción detallada del proceso de instalación y configuración de Internet Information Server en Windows XP profesional. Conceptos básicos necesarios para empezar la administración.

Pág. 377

Capitulo 1
Publicar en Internet

Tratamos los temas más básicos que necesitas saber para publicar una página web, como los materiales necesarios, el lenguaje HTML, editores, elegir alojamiento, subir páginas a los servidores y mucho más.

Introducción a Internet

Internet es una red de ordenadores conectados en toda la extensión del Globo Terráqueo que ofrece diversos servicios a sus usuarios como pueden ser el correo electrónico, el chat o la web. Todos los servicios que ofrece Internet son llevados a cabo por miles de ordenadores que están permanentemente encendidos y conectados a Internet, esperando que los usuarios les soliciten los servicios y sirviéndolos una vez son solicitados. Como decimos, hay servidores para todo, los hay que ofrecen correo electrónico, otros hacen posible nuestras conversaciones por chat, otros la transferencia de ficheros o la visita a las páginas web y así hasta completar la lista de servicios de Internet.

También existe otro tipo de servidores que son los que se encargan de proveer de acceso a Internet a nuestros ordenadores, son los proveedores de acceso, los servidores a los que nos conectamos con nuestros módems. Cuando hacemos la llamada con el módem a los servidores que proveen el acceso entramos a formar parte de Internet y mientras mantengamos la conexión podremos acceder a todos los servidores repartidos por todo el mundo y solicitarles sus servicios.

En el momento que pedimos un servicio de Internet nos convertimos en clientes del servidor que lo ofrece. Para solicitar uno de estos servicios es necesario contar con un programa especial que suele ser distinto para cada servicio de Internet. Por ejemplo, para acceder al correo electrónico utilizamos Outlook, para acceder a la web utilizamos Netscape o Internet Explorer o para entrar en el chat utilizamos un programa como Mirc o Pirch. Todos estos programas que nos dan acceso a los servicios de Internet se denominan clientes, como se puede ver, para ser el cliente de un servidor de Internet necesitamos un programa cliente del servicio al que intentamos acceder.

La web es un servicio de Internet

Toda esta introducción sirva para que nos demos cuenta que Internet es un conjunto de servicios y el web, que es lo que tratamos de analizar, no es más que uno de ellos. Probablemente sea el más popular en estos momentos y a veces parezca que Internet se limita al web, como en los anuncios de prensa, donde casi siempre nos venden portales o productos de la web como si ellos fueran lo único que se puede hacer en Internet. La web es entonces un servicio más que consiste en un inmenso conjunto de páginas conectadas unas a otras por un sistema de enlaces.

El sistema con el que está construido el web se llama hipertexto y es un entramado de páginas conectadas con enlaces. Los sistemas de hipertexto se utilizan en otros contextos, como la ayuda del Windows y son muy fáciles de utilizar y de encontrar lo que buscamos rápidamente. La web no solo se limita a presentar textos y enlaces, sino que también puede ofrecernos imágenes, videos, sonido y todo tipo de presentaciones, llegando a ser el servicio más rico en medios que tiene Internet. Por esta razón, para referirnos al sistema que implementa el web (hipertexto), se ha acuñado un nuevo término que es hipermedia, haciendo referencia a que el web permite contenidos multimedia.

Multimedia, por si alguno no lo sabe todavía, hace referencia a muchos medios, solamente quiere decir que se están utilizando muchos medios para presentar información como son el vídeo, el audio o realidad virtual. Cuando nos venden un ordenador multimedia, nos venden un ordenador que está construido para poder trabajar con muchos medios, como imagen, sonido, animación, video, etc.

Cómo es una web por dentro

Una página web la vemos en nuestro navegador, o cliente web, y parece una sola entidad, pero no es así, está compuesta por multitud de diferentes ficheros, como son las imágenes, los posibles vídeos y lo más importante, el código fuente que dice donde colocar cada texto, cada imagen o cada video y la forma que tendrán estos al ser colocados en la página. No es problema que las webs estén compuestas por tantos elementos, ya que rápidamente veremos que su organización es fácil y que no se nos van a perder o escapar ninguno.

Como hemos podido imaginar y a modo de resumen, para publicar en Internet necesitaremos construir unos documentos hipertexto, o hipermedia, con sus correspondientes archivos de imagen o video y colocarlos en unos ordenadores que son servidores de páginas web. Pero esto es algo que vamos a tratar de explicar poco a poco en los siguientes capítulos.

Aunque signifique adelantarse a los acontecimientos, es interesante señalar que con una simple acción podemos ver el código fuente de de las páginas, es decir, cómo están hechas por dentro. Para ello en Internet Explorer se ha de pulsar sobre el menú de Ver > Código fuente y en Netscape se ha de seleccionar Ver > Origen de la página.

Pasos previos I. Pensar un tema

Para publicar una página en Internet debemos seguir unos pasos, primero una planificación, luego la construcción de las páginas y más tarde la promoción y constante actualización de las páginas. En este capítulo vamos a ver cuáles son los pasos previos que debemos realizar para que nuestro esfuerzo y resultados sean lo más óptimos posibles, es decir, vamos a ver cuál es la planificación que hay que hacer antes de ponerse manos a la obra.

Pensar un tema

Puede ser el más importante de los pasos para la creación de un web. Hay que elaborar la idea y documentarse sobre el tema que has elegido para tu página en Internet.

Cuando hacemos incluso una página personal podemos hablar de algún tema interesante como nuestro equipo de fútbol o nuestro cantante favorito, o de un tema que nos conmueva profundamente, como el ecologismo o la historia de nuestra comarca.

También podemos construir una página que trate de nuestra familia o de nuestros perros, o todo junto, pero tenemos que tener en cuenta que el valor de este tipo de páginas es mucho menor y que a los demás usuarios seguramente no les va a interesar. Si deseamos construir una página que algún día sea popular, debemos abordar, como digo, temas que sean de interés para un grupo de gente.

Una vez pensado el tema es muy importante documentarse lo más posible sobre él, aunque muchas de las ideas pueden estar en nuestra cabeza, es importante tomar notas, fotografías u opiniones de otras personas que también conocen el asunto que nos traemos en la cabeza. Insisto, es importante que el material que vamos a publicar tenga el mayor valor posible, así conseguiremos los mejores resultados.

Pasos previos II. Estructura del sitio

Es importante también que pensemos en la estructura de los contenidos de todo el sitio antes de ponernos a diseñarlo. En este punto tenemos que pensar sobre varias cosas, como las distintas secciones que va a contener el sitio, el árbol de las páginas que vamos a tratar, etc.

La manera de que este punto sea realmente útil, es preparar todas estas ideas sobre el papel. Con toda la tranquilidad del mundo y con toda la determinación posible vamos a preparar una serie de diagramas y listados que nos permitan dirigir nuestros esfuerzos de la manera más óptima.

Secciones del sitio: Podremos pensar en qué secciones vamos a poner en el sitio. Una con información general, otra con información de contacto, otra con una visita gráfica a nuestras instalaciones, etc.

Árbol del sitio: Podemos dibujar en un papel cuáles son las distintas secciones del sitio, metidas dentro de cuadrados y representar también los enlaces que hay entre cada sección, mediante líneas que unen esos cuadrados. Es algo muy simple y divertido de hacer, además podemos tener nuevas ideas para el web a medida que lo vamos dibujando.

Esquema de una página: podemos dibujar en papel también una página del sitio que vamos a construir, para que veamos donde colocar la barra de navegación, el logotipo del sitio o la empresa, un posible banner publicitario, etc. Este esquema puede ser también de utilidad a la hora de construir las páginas y siempre es más fácil diseñar sobre el papel que diseñar directamente con el ordenador

Estos pasos son opcionales, cualquiera puede ponerse a construir una página nada más decidir que desea tener una web, pero no todo el mundo es capaz de plantearse bien cuáles son sus objetivos y formar un proyecto con todos los puntos desarrollados de antemano. Los resultados finales son lo más importante y empezar la casa por el tejado no va a ser lo más positivo para nuestra página y los resultados finales mejorarán si hacemos primero una planificación.

Qué necesitas para empezar

Para ponernos a diseñar páginas web nos hace falta realmente poco material. En la mayoría de los equipos que se instalan con Windows 98 se encuentran todos los materiales necesarios para empezar sin necesidad de obtener nuevos programas.

En concreto, los materiales necesarios son un editor de textos, con el que programaremos nuestras páginas y un cliente del web como Internet Explorer o Netscape Navigator. Como un ordenador Windows

98 por defecto siempre tiene instalado el Block de Notas y también el Internet Explorer se puede empezar a construir una página web sin más necesidades que lo que ya tenemos.

En otros sistemas distintos de Windows 98 será también muy fácil obtener un editor de textos sencillo y un navegador con el que ver las páginas que vamos a crear, si es que no están instalados ya. Sitios en la web desde donde se pueden descargar estos programas son Tucows o Download.com, donde tienen software de todos los sistemas operativos como Windows, Mac OS, o Linux.

También en castellano tenemos buenas páginas de descarga de software, la más representativa es Softonic.

Sin duda, en nuestra aventura con el desarrollo del web vamos a necesitar muchos más programas como por ejemplo programas de retoque fotográfico o editores complejos de páginas web, pero no es necesario que los introduzcamos todavía en este manual porque no son necesarios para dar nuestros primeros pasos.

Qué es HTML

HTML es el lenguaje con el que se definen las páginas web. Básicamente se trata de un conjunto de etiquetas que sirven para definir la forma en la que presentar el texto y otros elementos de la página.

El HTML se creó en un principio con objetivos divulgativos. No se pensó que la web llegara a ser un área de ocio con carácter multimedia, de modo que, el HTML se creó sin dar respuesta a todos los posibles usos que se le iba a dar y a todos los colectivos de gente que lo utilizarían en un futuro. Sin embargo, pese a esta deficiente planificación, si que se han ido incorporando modificaciones con el tiempo, estos son los estándares del HTML. Numerosos estándares se han presentado ya. El HTML 4.01 es el último estándar a febrero de 2001.

El HTML es un lenguaje de programación muy fácil de aprender, lo que permite que cualquier persona, aunque no haya programado en la vida pueda enfrentarse a la tarea de crear una web. HTML es fácil y pronto podremos dominar el lenguaje. Más adelante se conseguirán los resultados profesionales gracias a nuestras capacidades para el diseño y nuestra vena artista.

Una vez conocemos el concepto de HTML os vamos a adelantar algunas cosas más. Este lenguaje se escribe en un documento de texto, por eso necesitamos un editor de textos para escribir una página web. Así pues, el archivo donde está contenido el código HTML es un archivo de texto, con una peculiaridad, que tiene extensión .html o .htm (es indiferente cuál utilizar). De modo que cuando programemos en HTML lo haremos con un editor de textos, lo más sencillo posible y guardaremos nuestros trabajos con extensión .html, por ejemplo mipagina.html

Por adelantar un poco cómo se utiliza el HTML os diremos que el lenguaje consta de etiquetas que tienen esta forma <B> o <P>. Cada etiqueta significa una cosa, por ejemplo <B> significa que se escriba en negrita (bold) o <P> significa un párrafo, <A> es un enlace, etc. Casi todas las etiquetas tienen su correspondiente etiqueta de cierre, que indica que a partir de ese punto no debe de afectar la etiqueta. Por ejemplo </B> se utiliza para indicar que se deje de escribir en negrita. Así que el HTML no es más que una serie de etiquetas que se utilizan para definir la forma o estilo que queremos aplicar a nuestro documento. <B>**Esto está en negrita**</B>.

Para aprender HTML en profundidad tenemos un manual en DesarrolloWeb.com. Además se pueden consultar los enlaces a distintos manuales que tenemos en nuestro buscador en la sección de HTML. Hay que hacer en este punto una mención especial a un manual que es muy interesante, aunque un poco anticuado ya, el Webmaestro.

Editores de HTML

Para las personas que no deseen complicarse la vida con el lenguaje HTML, porque no tengan tiempo de aprenderlo o porque se sientan incapaces de hacerlo, hay una posibilidad distinta a programar directamente el HTML a base de texto. Se trata de utilizar un tipo de programas que nos permiten diseñar la página como si estuviéramos escribiendo un documento con un editor del tipo de Word. El editor de HTML es el encargado de vérselas con el lenguaje y programar internamente la página con el código HTML según lo que nosotros estamos diseñando.

Con el editor HTML podemos colocar imágenes, definir estilos, utilizar negritas o cursivas, etc. sin preocuparnos de las etiquetas correspondientes a cada estilo o elemento. Es el editor el que sabe estas etiquetas y las utiliza convenientemente. Este tipo de editores HTML se denominan habitualmente WYSIWYG (What You See Is What You Get) porque cuando trabajas con ellos lo que ves que estás creando con el editor es lo que obtienes luego cuando grabas la página.

Existe entre las personas que se dedican a realizar las páginas web dos tendencias. Por un lado tenemos

a las personas que prefieren crear las páginas programando el HTML y por otro las personas que utilizan editores HTML. Algunas diferencias entre hacerlo de un modo u otro son las siguientes

Escribiendo el HTML	Con un editor WYSIWYG
Dominas con mayor precisión el código de la página, queda más limpio. Si dominas bien el HTML nunca tendrás ningún problema para hacer lo que deseas.	El código de la página tiene peor calidad, incluso puede llegar a tener errores, más o menos visibles, que cuestan arreglar. Es la máquina la que domina el trabajo.
Es más complicado el aprendizaje, más lento y cuando se llega a un nivel avanzado también se hace considerablemente más difícil	El aprendizaje es muy sencillo, tal como puede ser trabajar en Word. Solo se trata de manejar un programa más.
Hacer una página cuesta más trabajo y tiempo.	Es muy rápido.

Cada uno debe escoger el camino que más le convenga o el que le parezca más atractivo. De todos modos siempre se puede empezar de un modo y luego pasar al otro modo sin ningún tipo de problema. Incluso, por adelantarnos a los acontecimientos, diríamos que cuando una persona profundiza en el diseño de páginas web llega un momento en el que le hace falta conocer las dos maneras de construir webs. A los programadores en HTML les hará falta aprender un editor porque eso aumentará su productividad y los que utilizan editores necesitarán aprender un poco de HTML para arreglar alguna cosa que el editor ha hecho mal o realizar alguna cosa que el editor no puede hacer.

Este manual está escrito por una persona que aprendió a realizar sus primeras webs con el Block de notas y algunas veces puede verse mi mayor inclinación a escribir el código HTML uno mismo. Aunque mi consejo es aprender HTML, estoy seguro que muchos de vosotros, maestros diseñadores, obtendréis mejores resultados utilizando un editor HTML WYSIWYG.

En el mercado existen multitud de editores de HTML WYSIWYG, es importante elegir un editor bueno porque nuestros trabajos van a depender de sus resultados. Actualmente el rey de los editores y el que os aconsejaríamos sin duda es el Dreamweaver, fabricado por Macromedia. Otras posibilidades son editores como GoLive de Adobe o Frontpage de Microsoft, aunque este último lo desaconsejamos.

Editores de texto preparados para escribir HTML

Las personas que después de estas líneas han decidido aprender el lenguaje HTML también tienen herramientas muy interesantes para aumentar su productividad sin dejar de escribir ellos mismos el HTML que desean. Se trata de unos editores de texto, como cualquier otro, que están preparados para escribir HTML y por lo tanto ofrecen multitud de ayudas a los diseñadores

- Colorean los códigos de las páginas para hacerlos más comprensibles
- Ofrecen ayudas a la programación
- Completan etiquetas

Y un montón de cosas más que sería demasiado complejo de enumerar aquí. Estos editores son por ejemplo Home Site o UltraEdit y es muy recomendable utilizarlos para sentirnos más a gusto al programar las páginas y poder hacerlas más rápido. Posiblemente sea aconsejable empezar con el Block de notas, por que es lo más sencillo, pero utilizar un programa de estos será imprescindible con el tiempo.

Construir las páginas

Por fin empezamos a trabajar en la página y vamos a ver algunos consejos útiles para hacerlo bien.

Es el momento en el que nos ponemos manos a la obra de una forma más dedicada y tenemos que trabajar más duramente. El programar o diseñar las páginas podrá gustar más o menos que otras acciones como planearlas o promocionarlas más tarde, pero no cabe duda que es el momento más excitante porque nuestros sueños y nuestras ideas empiezan a concretarse en los resultados que esperábamos conseguir.

Si hemos proyectado un sitio compuesto por un gran número de páginas lo más habitual es empezar diseñando una página con el marco del sitio, que luego vamos a repetir a modo de plantilla en todas las páginas. Pero esto son técnicas que aprenderemos con el tiempo.

Para ahorrarnos errores cuando hacemos las páginas podemos seguir una serie de consejos útiles.

- No utilizar espacios en los nombres de los archivos de las páginas o las imágenes. Tampoco utilizar caracteres raros como la ñ o los acentos.
- Tener cuidado con las mayúsculas y las minúsculas en los nombres de los archivos que tratamos. Si las utilizamos equivocadamente la página podrá funcionar en nuestro Windows (por que le dan igual las mayúsculas y las minúsculas), pero al subirla al servidor Linux o Unix podría ser que no funcionase (porque estos sistemas si que distinguen entre mayúsculas y minúsculas).
- Enterarse de cómo funciona el documento por defecto, que nos permitiréis que lo expliquemos en capítulos posteriores.
- Trabajar siempre con una extensión del archivo HTML específica. No mezclar en un mismo proyecto páginas con extensión .html y .htm.

Imágenes y otros recursos

Como se ha podido ver anteriormente, el diseño de una página web implica la creación de un archivo en código HTML, pero no es lo único que debemos crear. En la mayoría de los casos también desearemos incluir imágenes y para ello será necesario crear los correspondientes archivos gráficos.

El proceso para incluir una imagen en una página empieza por la creación de la imagen con un programa de diseño gráfico o mediante su digitalización con un escáner. Será necesario que conozcamos alguno de los programas de diseño gráfico que existen en el mercado. Son muy interesantes Photoshop, Paint Shop Pro o Fireworks. En un principio podremos contentarnos con manejarlos por encima y nos resultará muy fácil, pero según avancemos en el camino como diseñadores web será más necesario dominarlos para obtener resultados más profesionales.

Los tipos de archivos gráficos que soporta Internet son el JPG y el GIF. Tienen características distintas y por tanto usos distintos. Podemos conocer más esto en el reportaje Formatos gráficos de Internet.

Una vez tenemos los archivos gráficos los ponemos en el mismo directorio que los archivos HTML o en un subdirectorio de este y en el código de la página HTML pondremos una etiqueta especial para incluir la imagen, o la insertaremos con nuestro programa editor de HTML.

Lo importante de todo esto es que nos percatemos que el sitio web está compuesto por archivos HTML, GIFs, JPGs e incluso los correspondientes archivos que contengan videos, animaciones Flash, programas Java, etc. Todos estos archivos los tenemos que tener bien localizados dentro de nuestro disco duro y dentro de un mismo directorio. Por supuesto el orden como estén los archivos dentro del directorio es indiferente, pero podrá ser interesante que incluyamos subdirectorios para que quede todo mejor colocadito y su mantenimiento sea más fácil. Por ejemplo, una técnica muy habitual es colocar todas las imágenes dentro de un subdirectorio llamado images.

El objetivo de tanto orden y conocimiento de la localización de todos los archivos es que luego tendremos que subirlos al servidor sin olvidarnos de ninguno. En capítulos posteriores veremos cómo se hace este paso.

Un problema típico con las imágenes y otros archivos externos consiste en que cuando vemos las páginas en nuestro ordenador se ven correctamente y no falta ninguna imagen ni otros posibles elementos. Sin embargo, cuando subimos los archivos al servidor y vemos la página desde Internet esta se muestra con errores en las imágenes y otros elementos, de modo que no se pueden ver. Esto suele llamarse tener una imagen "rota". La razón por la que está rota es que no puede localizarla en el servidor y por tanto no la puede mostrar. Esto puede ser debido a varias razones.

- La imagen no ha sido subida al servidor
- La posición relativa de la imagen con respecto a la página no es la misma en nuestro ordenador (local) y en el servidor (remoto). Por ejemplo, las imágenes en local podrían estar en el directorio images mientras que en remoto podrían estar en el mismo directorio que la página, lo que sería un error. Siempre se debe respetar la estructura de directorios que hay en local y crearla exactamente igual en remoto.
- La imagen que se intenta acceder tiene un camino dirigido a un directorio de nuestro disco duro, como al ver la página desde Internet no se tiene acceso a tu disco duro, los usuarios no podrán ver las imágenes. Cuando trabajamos con un editor de HTML y colocamos imágenes en algunas ocasiones el editor coloca caminos en nuestro disco duro en lugar de caminos relativos. Los caminos relativos son rutas que empiezan en el lugar donde está la página que estamos diseñando.

Hay una forma muy útil de obtener pistas acerca del fallo de una imagen, consiste en pulsar con el botón derecho del ratón sobre ella y seleccionar propiedades. Esto nos muestra información sobre la imagen y nos informa sobre el sitio donde se está intentando encontrarla.

Alojar las páginas

Como hemos dicho, cualquier servicio que se quiere ofrecer en Internet tiene que brindarlo un servidor, que es un ordenador que se encuentra encendido las 24 horas del día y conectado a Internet también permanentemente. En el caso de una página web, existen unos servidores que son los encargados de mandarla cuando se la solicita, son los servidores web. Nuestras páginas tienen que estar alojadas en un servidor web para que puedan estar accesibles desde Internet.

Lo que tendremos que hacer entonces es buscar un lugar para alojar la página que esté acorde con nuestras necesidades, por suerte en muchos de los casos el alojamiento lo podremos conseguir de manera gratuita.

Subir los archivos al servidor

Es una de las tareas que parecen más difíciles cuando te pones a construir las páginas. Igual que cualquier cosa en este mundo, cuando lo has hecho unas cuantas veces el problema se desvanece.

Básicamente lo que tenemos que hacer es tomar todos los archivos que componen nuestro sitio web, incluidas imágenes, animaciones, etc. y subirlas a nuestro servidor web. Para ello primero es tarea imprescindible el identificar dónde están todos los archivos de nuestro web. Si hemos escrito la página con código HTML seguramente sabremos perfectamente donde están nuestros ficheros, pero si la página la hemos hecho con un editor HTML como Frontpage es probable que estén un poco más difíciles de identificar.

Dependiendo del alojamiento que tengamos, la manera de subir los archivos cambiará. Existen, de todos modos, dos maneras de subir los archivos al servidor, por FTP o a través de una interfaz web, de modo que podremos ver aquí todas las formas.

Subir archivos por FTP

La forma más tradicional de subir ficheros es por FTP, que es un servicio más de Internet que se utiliza para transferir ficheros por la red. Como lo que queremos hacer es transferir los ficheros desde nuestro ordenador al servidor, este es el servicio que debemos utilizar.

Cuando tenemos un alojamiento profesional para nuestras webs lo más seguro es que nos proporcionen un acceso por FTP a los servidores para subir las páginas.

Como otros servicios de Internet, para utilizar FTP necesitamos un programa especial que se denomina cliente de FTP. Podemos encontrar en el mercado muchos de estos clientes, algunos populares son Cute FTP o FTP Voyager. Si queremos ver una gama muy amplia de clientes FTP podemos acercarnos por Tucows o Donwload.com y encontraremos muchas opciones interesantes, algunas de ellas gratuitas.

Todos los programas de FTP son parecidos (igual que Internet Explorer es parecido a Netscape), básicamente consisten en una ventana que está partida en dos partes. En una parte podemos ver nuestro disco duro, con sus distintas unidades y carpetas. En la otra parte se puede ver el sistema de archivos del servidor, con sus correspondientes carpetas. Para mover los archivos de un lugar a otro suele bastar con arrastrarlos de una parte de la ventana a la otra.

Una tarea que también puede ser complicada en un principio puede ser el configurar el programa de FTP para que acceda al espacio que tenemos asignado. Los datos de configuración los debes obtener en el lugar donde te ofrecieron el espacio, son estos:

Nombre del servidor FTP: suele tener una forma como ftp.tudominio.com
Usuario: tu nombre de usuario.
Password: tu palabra de clave

Podría haber algún dato adicional, como el directorio por defecto, que es el directorio en el que deseas abrir la sesión, pero no es habitual que te den este dato porque los accesos por FTP suelen estar configurados para que se acceda directamente al directorio donde están tus páginas.

Con estas explicaciones ya verás que cuando vas a hacer un acceso por FTP te suena todo y no te cuesta nada el realizarlo. De todos modos, puedes ver un tutorial de Cute FTP que puede aclararte alguna duda más.

Acceso con interfaz web

Es muy típico que los proveedores de alojamiento gratuito provean de una herramienta de muy fácil uso para subir las páginas. Esta herramienta se accede a través de la web alojador y no es más que un formulario donde se puede elegir los archivos que se desea subir al servidor, se pulsa un botoncito, y todo listo.

Documento por defecto

Es importante conocer este concepto, incluso antes de ponerse ha diseñar la página. El documento por defecto es el archivo que envía el servidor cuando el cliente no especifica que archivo es el que solicita. Esto se puede ver muy fácilmente con un ejemplo.

Cuando escribimos la dirección http://www.desarrolloweb.com/ no estamos especificando ningún nombre de archivo en concreto, entonces el servidor web donde está alojado este sitio web le devolverá el documento por defecto del directorio raíz del dominio. Cada directorio puede tener un documento por defecto, por ejemplo, cuando escribimos www.desarrolloweb.com/manuales se envía el documento por defecto del directorio manuales.

Cada servidor web puede estás configurado de una manera distinta para el documento por defecto, es decir, en cada servidor web el documento por defecto puede ser distinto. Lo más habitual es que se llame index.html, en desarrolloweb es así y en muchos alojadores gratuitos también lo es, pero en otros casos puede variar el documento por defecto y ser lo que los administradores de cada servidor decidan. Otros nombres para el documento por defecto podrían ser default.html o ind.html.

Como decíamos, en desarrolloweb el documento por defecto se llama index.html, podemos probar a acceder a http://www.desarrolloweb.com/index.php o a http://www.desarrolloweb.com/manuales/index.php y veríamos como se accede a las mismas páginas que se accede sin especificar el archivo index.html, usando el documento por defecto.

> Nota: En desarrolloweb el documento por defecto es en realidad index.php. Esto es porque desarrolloweb.com está programado con páginas dinámicas PHP, que son un poco más complejas, pero que también nos permiten hacer cosas más avanzadas. En desarrolloweb podemos encontrar también información adicional sobre páginas dinámicas.

Decimos que es importante saber cuál es nuestro documento por defecto porque es necesario que llamemos a la primera página de nuestro sitio web con ese nombre de archivo, de modo que no tengamos que saber el nombre de ningún archivo de nuestra página para acceder y la dirección será más corta.

Por ejemplo si tenemos espacio en Geocities y nuestro nombre de usuario es pepe, nuestra página se accedería escribiendo www.geocities.com/pepe y se devolvería el documento por defecto. Sin embargo, si no utilizamos el documento por defecto y nuestra home page se llama mipagina.html si intentamos acceder sin poner el nombre de archivo, con la dirección de antes, dará un error de archivo no encontrado y para acceder a nuestra home deberíamos de escribir www.geocities.com/pepe/mipagina.html.

Promoción de las páginas

Con el diseño y la publicación del sitio web en el servidor no se termina el trabajo. Una de las tareas más importantes para el éxito de la web es promocionarla adecuadamente, de manera que su dirección figure en el mayor número de sitios.

Las acciones que se pueden realizar para promocionar una página son muchas y variadas, como hacer que figure su dirección en nuestros correos electrónicos o intercambiar banners, pero la más importante es el registro en los buscadores.

Estas tareas de promoción no son nada complicadas y cualquiera puede realizarlas sin ningún problema, pero conseguir que nuestra web se encuentre situada entre los primeros resultados de la búsqueda reviste más dificultad.

Actualizar las páginas

El último "truco" para que nuestra página sea muy visitada y que los visitantes entren una y otra vez se trata de mantenerla siempre bien actualizada. Hay algunas webs que se prestan más que otras a tener los contenidos actualizados, como son las páginas donde se pueden ver contenidos de actualidad, pero en general todas las páginas pueden mantener secciones con contenidos actualizables.

Este es un aspecto muy importante, pues si el navegante se percata de que los contenidos se renuevan constantemente volverá con el tiempo unas cuantas veces. Para dar una imagen de web actualizada se pueden utilizar imágenes o textos que resalten donde ponga "nuevo" o "new". También será adecuado poner un área en un sitio visible donde se enumeren las novedades del sitio.

Por lo general habrá que volver siempre sobre este punto para ver qué hay de nuevo y dónde se puede renovar el contenido del web. No es bueno tener un enlace que no lleva a ningún sitio, igual que no es bueno tener una dirección de correo que no existe ya. Además, si tenemos un área de noticias no será

bueno que la última noticia sea de hace tres meses porque da la impresión de que nunca renovamos los contenidos.

Si no mantenemos nuestro sitio actualizado el visitante entrará una vez a la página pero no lo volverá a hacer nunca, con lo que habremos perdido una oportunidad de hacernos con un cliente o un visitante asiduo. Supongo que habrá quedado claro lo importante de esta tarea, aunque salta a la vista, creo, para la mayoría de las personas.

Por dónde continuar

En este manual hemos visto lo más básico y sin duda tendremos multitud de trabajo en lo sucesivo para tratar de dominar cada una de las técnicas y programas de los que hemos hablado como el HTML, algún editor, los programas de diseño gráfico, alojar las páginas y subir archivos...

Pero en la carrera de un desarrollador del web queda todo un mundo de posibilidades por delante de las que ni siquiera hemos hablado, como puede ser el tratamiento de imágenes profesional, la programación de páginas avanzadas con Javascript y lenguajes como ASP o PHP que nos permitirán crear portales con páginas que acceden a bases de datos y ofrecen servicios más complicados.

Un punto muy interesante para continuar aprendiendo es la sección Desde 0 que contiene la guía para moverse por los contenidos de DesarrolloWeb para personas novatas. En la sección descubrirás enlaces a un manual de HTML, otro de promoción de páginas y a otros artículos y manuales de interés.

Muchas de estas cosas que acabo de enumerar te sonarán todavía a chino, pero ya verás que con el tiempo y a medida que aumenten tus necesidades llegarás a comprenderlas y estudiarlas con calma. Acerca de todas estas cuestiones se puedes informarte en DesarrolloWeb.com, según tu curiosidad y necesidades se presenten.

Otro manual que te puede introducir en procesos más avanzados de la programación de páginas web es el Manual de Paginas Dinámicas, que te explicará que es una página dinámica, para que la puedes necesitar y cómo llegar a programarlas.

Por ahora, este manual se acaba y solo me queda recomendarte otra serie de sitios donde puedes dirigirte para solucionar tus problemas y dudas.

- Si piensas que a este manual para principiantes le falta algún contenido realmente básico o algo que consideras confuso, te agradecería que escribieses a la dirección eugim@desarrolloweb.com para que publiquemos un artículo al respecto.
- Si deseas preguntar dudas puedes dirigirte a la lista de correo que tenemos a este efecto.
- También puedes participar en los foros para webmasters, donde se puede encontrar mucha información acerca de un montón de tecnologías, servidores, programación, etc.

Esperamos que la suerte te acompañe en tu diseño web.

Capitulo 2
Introducción al diseño web

Primeros pasos para aquellas personas que deseen crear su propia página web, desde el punto de vista del diseñador. Motivación a seguir, programas a utilizar, etc.

Prólogo a los primeros pasos en el diseño web

Hace unos días, durante una cena, mi padre me preguntó qué era exactamente eso de los chats, “o lo que quiera que la gente hace cuando se escribe a través de Internet”. Es un hombre un tanto suspicaz ante las nuevas tecnologías, de manera que me vi obligado a lidiar en tres frentes a la vez, añadiendo además el hecho de que la pregunta, planteada con intenciones solapadas, sonó como un aldabonazo y me desconcertó unos segundos. Las tres líneas de ataque de las que hablo eran las siguientes: una, la que me obligaba a describir de manera superficial las bases tecnológicas de los chats, ya que mi padre los había comparado con sistemas de comunicación más convencionales, y yo debía establecer las distancias adecuadas. En segundo lugar, le expliqué los elementos de la práctica, como por ejemplo la interfaz típica de un software de chat o las funciones básicas: flujo de texto en tiempo real y transmisión de archivos, así como el modo en que los chats se organizan y pueden ser disfrutados en diversas salas; se conoce que cosas muy elementales. En tercer lugar le sugerí la utilidad de los chats, sus ventajas frente a otras formas de comunicación a distancia y le expliqué, de forma más general ya, que pese a su postura, Internet constituye en gran medida el sistema sanguíneo del futuro, aunque en el mejor de los casos se trate arterias de fibra de carbono... De ahí pasamos a conversar acerca de cuestiones mucho más diversas en torno a la Red, pero esto ya no tenía importancia. Un tema u otro era un asunto de formas.

En realidad, yo no siento mucha atracción por los chats genéricos, y si acaso opto por aplicaciones de mensajería instantánea. Pero aquella conversación me reveló algunas de las bases que, después, me ayudarían a responder a una usuaria en cierto foro de diseño web. Su pregunta era: “Aunque no sé mucho de ordenadores, siempre estoy enganchada, y **ahora quiero hacer mi propia página web. ¿Por dónde empiezo?”**

Desde luego eso era toda una declaración de intenciones y no existía, ni existe, una respuesta infalible. El desarrollo web es un proceso creativo e íntimo, como lo es la navegación en Internet; ¿dónde voy si me conecto a la Red? Pues eso depende de ti. Es más: lo probable es que muy pronto ya ni siquiera dejes que otros usuarios compartan y fisguen en tu historial. De la misma manera, una vez que empezamos a diseñar, decidimos que nadie puede entrometerse en nuestra forma de trabajar. ¿Cierto?

Entonces, ¿quiere esto decir que no hay una especie de protocolo de principiantes? Oh, sí que hay un protocolo, un camino de baldosas que todos solemos seguir para llegar a nuestro destino (naturalmente, el destino de cada usuario es distinto, y mientras más diferente e ingenioso, mayor será el aprendizaje).

Introducción al diseño web

Como en el caso de los chats, podemos considerar que existen tres directrices elementales que sirven de guías. Una de ellas es el conocimiento tecnológico de Internet; estamos obligados a saber cómo funciona de manera general, desde las conexiones de ordenador hasta el valor de los servidores, pasando por la arbitraria congestión de las líneas telefónicas y, desde luego, considerando los diversos navegadores que hay en el mercado, además de prestar atención a la necesidad de instruirnos, aunque sea un poco, en el lenguaje HTML. Es muy difícil tomarse en serio a alguien que no tiene constancia de que, echando un vistazo al código fuente de una página, hallará etiquetas tan básicas como <head> y <body>. Estos conocimientos técnicos nos informan de nuestras posibilidades creativas; no basta con crear una preciosa imagen y colgarla en nuestra página principal, si no sabemos que dicha imagen podría tardar un minuto en cargarse... Y a día de hoy ésa sí es una verdad infalible.

En segundo lugar, estamos obligados a saber trabajar. No es más difícil que eso; si trabajas en modo código, debes dominar el HTML y, mejor aún, el HTML y otros lenguajes de uso común en la Red. Si empleas software de tipo WYSIWYG (lo que ves es lo que obtienes, en español), debes ser capaz de conocer las funciones de tu programa. Desde luego que detenerse a leer las instrucciones es aburrido, y provoca lo que se conoce como “curva de aprendizaje”, pero es la única manera de llegar a entender y desarrollar todas las posibilidades técnicas e imaginativas que te permiten las diferentes aplicaciones del mercado. Si usas DreamWeaver, aprende DreamWeaver; puede que conozcas FrontPage, por ejemplo, pero eso no conduce a ninguna parte. Son programas distintos que necesitan procesos de instrucción diversos.

Características especiales de la publicación web

En tercer lugar es fundamental saber que Internet no es una conferencia telefónica, un CD-ROM o televisión a la carta. La Red sufre limitaciones propias y disfruta también de posibilidades más extensas y personales. Así, cuando diseñamos nuestros sitios web, debemos entender que el visitantes no espera lo mismo de nuestras páginas que de un programa de TV, incluso aunque ambos traten el mismo tema.

Tomemos por ejemplo un documental sobre los elefantes africanos. E imaginemos a un ciudadano muy interesado en dichos animales, que en un momento del día debe elegir entre ver la vida de los elefantes en televisión, o buscar información en Internet a propósito de lo mismo. Nosotros tenemos la obligación de diseñar una página web al respecto. ¿Tomaríamos inspiración de la televisión? Quizás alguna, pero si sabemos lo que nos conviene, no pasaríamos de ahí. Por ejemplo, la tele se encuentra a un mínimo de dos metros del televidente; el monitor de la computadora, a unos palmos; el texto debe ser estrecho para no irritar al navegante. En la tele eso da igual. A la tele también le es indiferente mostrar cientos de tomas, cientos de perspectivas y cientos de sonidos emitidos por los elefantes africanos. El web apenas podría hacer eso, en condiciones estándar de conexiones de 56kb. Pero nosotros podemos crear enlaces a páginas que traten todas y cada una de las costumbres del enorme paquidermo. La televisión, no...

La pregunta, sintetizada, es ésta: ¿qué damos al navegante para hacerle fiel a nuestros contenidos aunque tan sólo sea durante una sesión? Le damos personalización, le damos contenidos, le damos libertad, le damos ingenio, le damos belleza, le damos oportunidades, le damos buen gusto, le damos comodidad, le damos datos, le damos síntesis, le damos gráficos, le damos información detallada, le damos esquemas, le damos conocimientos en tres palabras o sabiduría en textos de tres kilómetros, le damos todo y más, le damos, en fin, lo que sea que busca. ¿Estamos en condiciones de hacer estas cosas? Más nos vale.

¿Algo más sobre el diseño web?

Hay más cosas, cierto. El diseño enfocado a la Red suele ser hijo de la receptividad, como cualquier otro tipo de diseño. Más allá del aprendizaje técnico, que consiste en leer informaciones a veces aburridas y engorrosas, existe un modo complementario y muy excitante de aprender. Como decía, hablamos de receptividad. De ver lo que otros muchos hacen con un talento que parece inagotable, y dejar que lo que vemos sedimente en nuestro trabajo; desde el diseño gráfico de los sitios web, desde las imágenes que incluyen y que a veces definen su identidad, hasta la manera en que escriben sus textos. Dejadme decir que pocas cosas me parecen tan divertidas y poco serias como un párrafo mal escrito. **¿Es que desenvolvernos en un entorno digital nos da derecho a escribir mal?** Sin duda que no. Los textos deben ser sugestivos, esclarecedores o enigmáticos, pero siempre llamativos y eficientes.

Hace poco visitaba un sitio web de cierta gran empresa y terminé preguntándome si la redacción había corrido a cargo de algún narrador extranjero. El texto era breve y directo, pero cometía fallos gramaticales tan evidentes que era lo que más llamaba la atención; incluso más que el bonito y “efervescente” diseño. Tengamos en cuenta que si nos consideran poco aplicados, no nos tomarán como una alternativa razonable. **(Y olvidemos la idea de que alguien nos pague por nuestro trabajo.)**

Miremos, analicemos, aprendamos de los demás. Es un proceso basado en la febril actividad de nuestro ratón. Tendremos que hacer clic cientos, miles de veces, antes de que el efecto de los maestros del web deje huella en nosotros. Para empezar podemos observar los estudios y casos de grandes empresas del medio como Adobe y Macromedia, que incluyen secciones ex profeso muy interesantes; y también tenemos la oportunidad de mirar las páginas de las agencias de publicidad y diseño; con frecuencia sus mejores trabajos están hechos para sí mismos, y no para sus clientes.

Herramientas para profesionales del diseño web

Después necesitas el software que interpretará tus deseos, inquietudes y talentos y les dará la forma que quieras —en el mejor de los casos—, de manera que puedas transferirlo todo a un servidor web abierto al público. Tampoco aquí existe acuerdo. Todo dependerá de si trabajas en modo html —1st Page 2000 es una aplicación reconocida y gratuita— o WYSIWYG, cuyo espectro es también muy amplio, desde el popular y doméstico FrontPage de Microsoft, hasta los profesionales y valiosos DreamWeaver o GoLive!, pasando por los accesibles pero potentes Namo Web Editor o NetObjects Fusion...

Tampoco viene mal disponer de algún programa de diseño gráfico. De una manera elemental podemos decir que los hay de dos tipos más o menos puros, y un tercer tipo mixto. En uno de los casos se trata de aplicaciones de dibujo vectorial, es decir, aquellas que generan gráficos desde cero mediante herramientas de trazado geométrico, fundamentalmente, pero también a mano alzada; dicho de una forma más llana, hablamos de dibujos de líneas y curvas. Pero no nos confundamos. Estos programas son madre de muchas de las mejores ilustraciones que vemos tanto en la Red como en publicidad impresa y en los envases de productos cotidianos. Actualmente se asume que Adobe Illustrator es el

mejor en este campo; sin embargo, compite con el clásico indiscutible CorelDraw, además de con Macromedia Freehand. Y, no obstante, existen alternativas baratas y eficientes, bien criticadas y admiradas en los medios, como ZonerDraw 4 o Xara.

El segundo de los casos cuando hablamos de gráficos es el retoque fotográfico; o lo que es lo mismo, el trabajo de diseño visual sobre imágenes que normalmente combinan fotografías reales con objetos diseñados y efectos visuales. También Adobe y Corel se disputan el pastel, de momento favorable al primero. Sus estrellas son Photo Shop y Photo Paint. Las alternativas más accesibles suelen ser Paint Sho Pro de Jasc y Ulead Photo Impact.

Una tercera opción es el software que combina retoque fotográfico y dibujo vectorial; cada vez más, ésta es una posibilidad que traen de serie todos los grandes programas, de manera que no hace falta cambiar de aplicación para mezclar los modos de trabajo gráfico.

Conclusión a la introducción al diseño web

¿Existe vida más allá de los gráficos estáticos?

En realidad sí. La opción más común es Flash de Macromedia, que permite crear películas realmente complejas e interactivas combinando ilustraciones embebidas, gráficos vectoriales, sonidos y ActionScript, el lenguaje de comandos específico de Flash, amén de otras posibilidades muy bien conocidas. Adicionalmente está disponible el formato vectorial SVG.

Conclusión

En conclusión, el trabajo del desarrollador web (esa especie de alquimista obligado a dominar y combinar variables tan dispares, y a veces de apariencia incompatible, como la usabilidad y el atractivo de cara al usuario, y la efectividad técnica de cara a la empresa que paga sus honorarios, como por ejemplo en el caso de tiendas on-line), este trabajo, decía, es un proceso largo que requiere un enorme grado de instrucción y aprendizaje técnico, pero también generosas dosis de intuición y anticipación. A fin de cuentas, la Red muta y crece cada poco tiempo, y no podemos quedar atrás. Buena suerte.

Capitulo 3
Manual de HTML

HTML es el lenguaje utilizado como base para crear las páginas web. Con este manual puedes aprender a utilizarlo con toda su potencia.

Prólogo al manual de HTML

Bienvenidos al manual de HTML de DesarrolloWeb. A través de todos estos capítulos vamos a descubrir el lenguaje utilizado para la creación de páginas web: el **Hyper Text Markup Language**, más conocido como **HTML**.

Puede que en un principio, el hecho de hablar de un lenguaje informático pare los pies a más de uno. No os asustéis, el HTML no deja de ser más que una forma un tanto peculiar de dar formato a los textos e imágenes que pretendemos ver por medio de un navegador.

Antes de entrar en materia, lo cual haremos de una forma directa y practica, os **recomendamos fervorosamente la lectura previa de nuestro manual Publicar en Internet**. A partir de esta guía, aprenderéis los conceptos más básicos necesarios para creación de un sitio web. También os permitirá acceder a este manual con unos conocimientos de base sobre HTML imprescindibles y os dejara bien claro lo que su conocimiento aporta con respecto al simple uso de editores de HTML.

El público al que va enfocado este manual es a todos aquellos que, con conocimientos mínimos de informática, desean hacer mundialmente público un mensaje, una idea o una información usando para ello el medio más práctico, económico y actual: Internet.

Lo que necesitáis como base para llevar a buen término el aprendizaje (aparte de leer el manual Publicar en Internet) es:

- Saber escribir con un teclado
- Saber manejar un ratón
- Tener ganas de aprender

Lo que obtendréis después de haber pasado por estos capítulos:

- Capacidad para crear y publicar vuestro propio sitio web con un mínimo de calidad
- Conocimientos de todo tipo sobre las tecnologías y herramientas empleadas en el ámbito de la Red
- Posiblemente una afición que puede convertirse en pasión y terminar, en algunos casos, siendo un vicio o un oficio.

Introducción al HTML

HTML es el lenguaje con el que se escriben las páginas web. Las páginas web pueden ser vistas por el usuario mediante un tipo de aplicación llamada navegador. Podemos decir por lo tanto que el HTML es el lenguaje usado por los navegadores para mostrar las páginas webs al usuario, siendo hoy en día la interface más extendida en la red.

Este lenguaje nos permite aglutinar textos, sonidos e imágenes y combinarlos a nuestro gusto. Además, y es aquí donde reside su ventaja con respecto a libros o revistas, el HTML nos permite la introducción de referencias a otras páginas por medio de los enlaces hipertexto.

El HTML se creó en un principio con objetivos divulgativos. No se pensó que la web llegara a ser un área de ocio con carácter multimedia, de modo que, el HTML se creó sin dar respuesta a todos los posibles usos que se le iba a dar y a todos los colectivos de gente que lo utilizarían en un futuro. Sin embargo, pese a esta deficiente planificación, si que se han ido incorporando modificaciones con el tiempo, estos son los estándares del HTML. Numerosos estándares se han presentado ya. El HTML 4.01 es el último estándar a septiembre de 2001.

Esta evolución tan anárquica del HTML ha supuesto toda una seria de inconvenientes y deficiencias que han debido ser superados con la introducción de otras tecnologías accesorias capaces de organizar, optimizar y automatizar el funcionamiento de las webs. Ejemplos que pueden sonaros son las CSS, JavaScript u otros. Veremos más adelante en qué consisten algunas de ellas.

Otros de los problemás que han acompañado al HTML es la diversidad de navegadores presentes en el mercado los cuales no son capaces de interpretar un mismo código de una manera unificada. Esto obliga al webmáster a, una vez creada su página, comprobar que esta puede ser leída satisfactoriamente por todos los navegadores, o al menos, los más utilizados.

Además del navegador necesario para ver los resultados de nuestro trabajo, necesitamos evidentemente otra herramienta capaz de crear la página en si. Un archivo HTML (una página) no es más que un texto. Es por ello que para programar en HTML necesitamos un editor de textos.

Es recomendable usar el Bloc de notas que viene con windows, u otro editor de textos sencillo. Hay que tener cuidado con algunos editores más complejos como Wordpad o Microsoft Word, pues colocan su propio código especial al guardar las páginas y HTML es **únicamente texto plano**, con lo que podremos tener problemas.

Existen otro tipo de editores específicos para la creación de páginas web los cuales ofrecen muchas facilidades que nos permiten aumentar nuestra productividad. No obstante, es aconsejable en un principio utilizar una herramienta lo más sencilla posible para poder prestar la máxima atención a nuestro código y familiarizarnos lo antes posible con él. Siempre tendremos tiempo más delante de pasarnos a editores más versátiles con la consiguiente ganancia de tiempo.

Para tener más claro todo el tema de editores y los tipos que existen, visita los artículos:

- Editores de HTML.
- Bloc de notas.
- También puedes acceder a descripciones editores más complejos que el Block de Notas, pero más potentes como Homesite o UltraEdit.

Es importante tener claro todo ello puesto que en función de vuestros objetivos puede que, más que aprender HTML, resulte más interesante aprender el uso de una aplicación para la creación de páginas.

Así pues, una página es un archivo donde está contenido el código HTML en forma de texto. Estos archivos tienen extensión .html o .htm (es indiferente cuál utilizar). De modo que cuando programemos en HTML lo haremos con un editor de textos y guardaremos nuestros trabajos con extensión .html, por ejemplo mipágina.html

> **Consejo:** Utiliza siempre la misma extensión en tus archivos HTML. Eso evitará que te confundas al escribir los nombres de tus archivos unas veces con .htm y otras con .html. Si trabajas con un equipo en un proyecto todavía más importante que os pongáis todos de acuerdo en la extensión.

Sintaxis del HTML

El HTML es un lenguaje que basa su sintaxis en un elemento de base al que llamamos etiqueta. La etiqueta presenta frecuentemente dos partes:

Una apertura de forma general <etiqueta>
Un cierre de tipo </ etiqueta>

Todo lo incluido en el interior de esa etiqueta sufrirá las modificaciones que caracterizan a esta etiqueta. Así por ejemplo:

Las etiquetas y definen un texto en negrita. Si en nuestro documento HTML escribimos una frase con el siguiente código:

```
<b>Esto esta en negrita</b>
```

El resultado Será:

Esto esta en negrita

Las etiquetas y definen un párrafo. Si en nuestro documento HTML escribiéramos:

```
<p>Hola, estamos en el párrafo 1</p>
<p>Ahora hemos cambiado de párrafo</p>
```

El resultado sería:

Hola, estamos en el párrafo 1

Ahora hemos cambiado de párrafo

Partes de un documento HTML

Además de todo esto, **un documento HTML ha de estar delimitado por la etiqueta <html> y </html>**. Dentro de este documento, podemos asimismo distinguir dos partes principales:

El encabezado, delimitado por <head> y </head> donde colocaremos etiquetas de índole informativo como por ejemplo el titulo de nuestra página.

El cuerpo, flanqueado por las etiquetas <body> y </body>, que será donde colocaremos nuestro texto e imágenes delimitados a su vez por otras etiquetas como las que hemos visto.

El resultado es un documento con la siguiente estructura:

```
<html>

<head>
Etiquetas y contenidos del encabezado
Datos que no aparecen en nuestra página pero que son importantes para catalogarla: Titulo, palabras clave,...
</head>

<body>
Etiquetas y contenidos del cuerpo
Parte del documento que será mostrada por el navegador: Texto e imágenes
</body>

</html>
```

Las mayúsculas o minúsculas son indiferentes al escribir etiquetas

A notar que las etiquetas pueden ser escritas con cualquier tipo de combinación de mayúsculas y minúsculas. <html>, <HTML> o <HtMl> son la misma etiqueta. Resulta sin embargo aconsejable acostumbrarse a escribirlas en minúscula ya que otras tecnologías que pueden convivir con nuestro HTML (XML por ejemplo) no son tan permisivas y nunca viene mal coger buenas costumbres desde el principio para evitar fallos triviales en un futuro.

Tu primera página

Podemos ya con estos conocimientos, y alguno que otro más, crear nuestra primera página. Para ello, abre tu editor de textos y copia y pega el siguiente texto en un nuevo documento.

```
<html>

<head>
<title>Cocina Para Todos</title>
</head>

<body>
<p><b>Bienvenido,</b></p>
<p>Estás en la página <b>Comida para Todos</b>.</p>
<p>Aquí aprenderás recetas fáciles y deliciosas.</p>
</body>

</html>
```

Ahora guarda ese archivo con extensión .html o .htm en tu disco duro. Para ello accedemos al menú Archivo y seleccionamos la opción Guardar como. En la ventana elegimos el directorio donde deseamos guardarlo y colocaremos su nombre, por ejemplo mi_pagina.html

> **Consejo:** Utiliza nombres en tus archivos que tengan algunas normas básicas para ahorrarte disgustos y lios.

Nuestro consejo es que no utilices acentos ni espacios ni otros caracteres raros. También te ayudará escribir siempre las letras en minúsculas.

Esto no quiere decir que debes hacer nombres de archivos cortos, es mejor hacerlos descriptivos para que te aclaren lo que hay dentro. Algún caracter como el guión "-" o el guión bajo "_" te puede ayudar a separar las palabras. Por ejemplo quienes_somos.html

Con el documento HTML creado, podemos ver el resultado obtenido a partir de un navegador. Es conveniente, llegado a este punto, hacer hincapié en el hecho de que no todos los navegadores son idénticos. Desgraciadamente, los resultados de nuestro código pueden cambiar de uno a otro por lo que resulta aconsejable visualizar la página en varios. Generalmente se usan Internet Explorer y Netscape como referencias ya que son los más extendidos.

A decir verdad, en el momento que estas líneas son escritas, Internet Explorer acapara la inmensa mayoría de usuarios (90% más o menos) y Netscape esta relegado a un segundo plano. Esto no quiere decir que lo debemos dejar totalmente de lado ya que el 10% de visitas que puede proporcionarnos puede resultar muy importante para nosotros. Por otra parte, parece que se ha hecho publica la intención de Netscape de desviar un poco su temática de negocios hacia otros derroteros y abandonar esta llamada "lucha de navegadores" en la cual estaba recibiendo la peor parte.

Pues bien, volviendo al tema, una vez creado el archivo .html o .htm, podemos visualizar el resultado de nuestra labor abriendo dicha página con un navegador. Para hacerlo, la forma resulta diferente dependiendo del navegador:

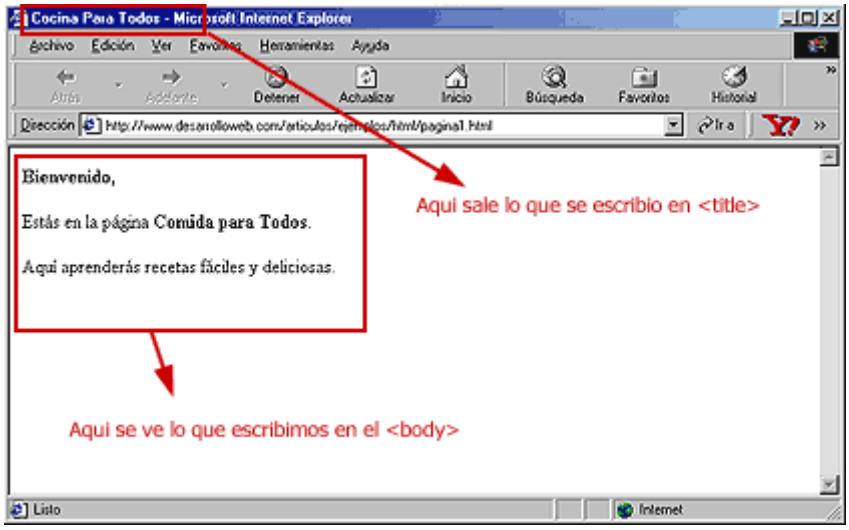
Si estamos empleando el Explorer, hemos de ir al barra de menú, elegir Archivo y seleccionar Abrir. Una ventana se abrirá. Pulsamos sobre el botón Examinar y accederemos a una ventana a partir de la cual podremos movernos por el interior de nuestro disco duro hasta dar con el archivo que deseamos abrir.

La cosa no resulta más difícil para Netscape. En este caso, nos dirigimos también a la barra de menú principal y elegimos File y a continuación Open File. La misma ventana de búsqueda nos permitirá escudriñar el contenido de nuestro PC hasta dar con el archivo buscado.

Nota: También puedes abrir el archivo si accedes al directorio donde lo guardaste. En él podrás encontrar tu archivo HTML y verás que tiene como icono el logotipo de Netscape o el de Internet Explorer. Para abrirlo simplemente hacemos un doble click sobre él.

Una vez abierto el archivo podréis ver vuestra primera página web. Algo sencillita pero por algo se empieza. Ya veréis como en poco tiempo seremos capaces de mejorar sensiblemente.

Fijaos en la parte superior izquierda de la ventana del navegador. Podréis comprobar la presencia del texto delimitado por la etiqueta <title>. Esta es una de las funciones de esta etiqueta, cuyo principal cometido es el de servir de referencia en los motores de búsqueda como Altavista o Yahoo.



Por otro lado, los elementos que colocamos entre la etiqueta <body> y </body> se pueden ver en el espacio reservado para el cuerpo de la página.

Si ahora hacéis click con el botón derecho sobre la página y elegís Ver código fuente (o View page

source) veréis como en una ventana accesoria aparece el código de nuestra página. Este recurso es de extremada importancia ya que nos permite ver el tipo de técnicas empleadas por otros para la confección de sus páginas.

Con todo esto asimilado ya estamos en condiciones de adentrarnos un poco más en la descripción de algunas de las etiquetas más empleadas del HTML.

> **Posible problema:** Al utilizar el Block de Notas en Windows en ocasiones, aunque le digamos que es un archivo .html, el documento se guarda como si fuera un texto y no una página web. Lo que está pasando es que el Block de Notas tiene predeterminado guardar sus archivos con extensión .txt y en realidad lo que está guardando en el disco duro es mi_pagina.html.txt
>
> Para conseguir tener el control de las extensiones en el block de notas y en Windows en general podemos acceder a MI-PC y en el menú de Ver seleccionáis "Opciones de carpeta". En la ventana que sale pulsamos en la solapa "Ver" y nos permite deseleccionar una caja de selección que pone algo como "Ocultar extensiones para los tipos de archivos conocidos". (Así se hace en Win98, puede variar un poco en otras versiones de Windows.)
>
> Con ello conseguiremos que se vea siempre la extensión del archivo con el que estamos trabajando y que el Block de Notas nos haga caso cuando le indicamos que grabe el archivo con otra extensión que no sea .txt

Formato de párrafos en HTML

En los capítulos anteriores hemos presentado a titulo de ejemplo algunas etiquetas que permiten dar formato a nuestro texto. En este capitulo veremos con más detalle las más ampliamente utilizadas y ejemplificaremos algunas de ellas posteriormente.

Formatear un texto pasa por tareas tan evidentes como definir los párrafos, justificarlos, introducir viñetas, numeraciones o bien poner en negrita, itálica...

Hemos visto que para definir los párrafos nos servimos de la etiqueta <p> que introduce un salto y deja una línea en blanco antes de continuar con el resto del documento.

Podemos también usar la etiqueta
, de la cual no existe su cierre correspondiente (</br>), para realizar un simple retorno de carro con lo que no dejamos una línea en blanco sino que solo cambiamos de línea.

> **Nota:** Existen otras etiquetas que no tienen su correspondiente de cierre, como <img> para las imágenes, las veremos más adelante. Esto ocurre porque un salto de línea o una imagen no empiezan y acaban más adelante sino que sólo tienen presencia en un lugar puntual.

Podéis comprobar que cambiar de línea en nuestro documento HTML sin introducir alguna de estas u otras etiquetas no implica en absoluto un cambio de línea en la página visualizada. En realidad el navegador introducirá el texto y no cambiara de línea a no ser que esta llegue a su fin o bien lo especifiquemos con la etiqueta correspondiente.

Los párrafos delimitados por etiquetas <p> pueden ser fácilmente justificados a la izquierda, centro o derecha especificando dicha justificación en el interior de la etiqueta por medio de un atributo align. Un atributo no es más que un parámetro incluido en el interior de la etiqueta que ayuda a definir el funcionamiento de la etiqueta de una forma más personal. Veremos a lo largo de este manual cantidad de atributos muy útiles para todo tipo de etiquetas.

Así, si deseásemos introducir un **texto alineado a la izquierda** escribiríamos:

```
<p align="left">Texto alineado a la izquierda</p>
```

El resultado seria:

Texto alineado a la izquierda

Para una **justificación al centro**:

```
<p align="center">Texto alineado al centro</p>
```

que daría:

Texto alineado al centro

Para **justificar a la derecha**:

```
<p align="right">Texto alineado a la derecha</p>
```

cuyo efecto seria:

Texto alineado a la derecha

Como veis, en cada caso el atributo align toma determinados valores que son escritos entre comillas. En algunas ocasiones necesitamos especificar algunos atributos para el correcto funcionamiento de la etiqueta. En otros casos, el propio navegador toma un valor definido por defecto. Para el caso de align, el valor por defecto es left.

> **Nota:** Los atributos tienen sus valores indicados entre comillas ("), pero si no los indicamos entre comillas también funcionará en la mayoría de los casos. Sin embargo, es aconsejable que pongamos siempre las comillas para acostumbrarnos a utilizarlas, por dar homogeneidad a nuestros códigos y para evitar errores futuros en sistemas más quisquillosos.

El atributo align no es exclusivo de la etiqueta <p>. Otras etiquetas muy comunes, que veremos más adelante, entre las cuales se introducen texto o imágenes, suelen hacer uso de este atributo de una forma habitual.

Imaginemos un texto relativamente largo donde todos los párrafos están alineados a la izquierda (por ejemplo). Una forma de simplificar nuestro código y de evitar introducir continuamente el atributo align sobre cada una de nuestras etiquetas es utilizando la etiqueta .

Esta etiqueta por si sola no sirve para nada. Tiene que estar acompañada del atributo align y lo que nos permite es alinear cualquier elemento (párrafo o imagen) de la manera que nosotros deseemos.

Así, el código:

```
<p align="left">Parrafo1</p>
<p align="left"> Parrafo3</p>
<p align="left"> Parrafo2</p>
```

es equivalente a:

```
<div align="left">
<p>Parrafo1</p>
<p>Parrafo2</p>
<p>Parrafo3</p>
</div>
```

Como hemos visto, la etiqueta marca divisiones en las que definimos un mismo tipo de alineado.

Ejemplo práctico:

Para practicar un poco lo que acabamos de ver vamos a proponer un ejercicio que podéis resolver en vuestros ordenadores. Simplemente queremos construir una página que tenga, por este orden:

2 Párrafos centrados
3 Párrafos alineados a la derecha
Un salto de línea triple
1 párrafo alineado a la izquierda

Encabezados

Existen otras etiquetas para definir párrafos especiales, formateados como títulos. Son los encabezados o Header en inglés. Como decimos, son etiquetas que formatean el texto como un titular, para lo cual asignan un tamaño mayor de letra y colocan el texto en negrita.

Hay varios tipos de encabezados, que se diferencian en el tamaño de la letra que utilizan. La etiqueta en concreto es la <h1>, para los encabezados más grandes, <h2> para los de segundo nivel y así hasta

<h6> que es el encabezado más pequeño.

Los encabezados implican también una separación en párrafos, así que todo lo que escribamos dentro de <h1> y </h1> (o cualquier otro encabezado) se colocará en un párrafo independiente.

Podemos ver cómo se presentan algunos encabezados a continuación.

```
<h1>Encabezado de nivel 1</h1>
```

Se verá de esta manera en la página:

Encabezado de nivel 1

Los encabezados, como otras etiquetas de HTML, soportan el atributo align. Vemos un ejemplo de encabezado de nivel 2 alineado al centro.

```
<h2 align="center">Encabezado de nivel 2</h2>
```

Se verá de esta manera en la página:

Encabezado de nivel 2

Otro ejercicio interesante es construir una página web que contenga todos los encabezados posibles. Se puede ver a continuación.

```
<html>

<head>
<title>Todos los encabezados</title>
</head>

<body>

<h1>Encabezado de nivel 1</h1>
<h2>Encabezado de nivel 1</h2>
<h3>Encabezado de nivel 1</h3>
<h4>Encabezado de nivel 1</h4>
<h5>Encabezado de nivel 1</h5>
<h6>Encabezado de nivel 1</h6>

</body>

</html>
```

Consejo: No debemos utilizar las etiquetas de encabezado para formatear el texto, es decir, si queremos colocar un tipo de letra más grande y en negrita debemos utilizar las etiquetas que existen para ello (que veremos en seguida). Los encabezados son para colocar titulares en páginas web y es el navegador el responsable de formatear el texto de manera que parezca un titular. Cada navegador, pues, puede formatear el texto a su gusto con tal de que parezca un titular.

Formateando el texto

Además de todo lo relativo a la organización de los párrafos, uno de los aspectos primordiales del formateo de un texto es el de la propia letra. Resulta muy común y práctico presentar texto resaltado en negrita, itálica y otros. Paralelamente el uso de índices, subíndices resulta vital para la publicación de textos cient&iaacute;ficos. Todo esto y mucho más es posible por medio del HTML a partir de multitud de etiquetas entre las cuales vamos a destacar algunas.

Negrita

Podemos escribir texto en negrita incluyéndolo dentro de las etiquetas y (bold). Esta misma tarea es desempeñada por y siendo ambas equivalentes. Nosotros nos inclinamos por la primeras por simple razon de esfuerzo.

Escribiendo un código de este tipo:

```
<b>Texto en negrita</b>
```

Obtenemos este resultado:

Texto en negrita

> **Nota:** ¿Qué diferencia hay entre <b> y ?
>
> Aunque las dos etiquetas hacen el mismo efecto, tienen una peculiaridad que las hace distintas. La etiqueta <b> indica negrita, mientras que la etiqueta indica que se debe escribir resaltado. El HTML lo interpretan los navegadores según su criterio, es por eso que las páginas se pueden ver de distinta manera en unos browsers y en otros. La etiqueta <H1> quiere decir "encabezado de nivel 1", es el navegador el responsable de formatear el texto de manera que parezca un encabezado de primer nivel. En la práctica los encabezados de Internet Explorer y Netscape son muy parecidos (tamaño de letra grande y en negrita), pero otro navegador podría colocar los encabezados con subrayado si le pareciese oportuno.
>
> La diferencia entre <b> y se podrá entender ahora. Mientras que <b> significa simplemente negrita y todos los navegadores la interpretarán como negrita, es una etiqueta que significa que se tiene que resaltar fuertemente el texto y cada navegador es el responsable de resaltarlo como desee. En la práctica coloca el texto en negrilla, pero podría ser que un navegador decidiese resaltar colocando negrilla, subrayado y color rojo en el texto.

Itálica

También en este caso existen dos posibilidades, una corta: e (italic) y otra un poco más larga: y . En este manual, y en la mayoría de las páginas que veréis por ahí, os encontraréis con la primera forma sin duda más sencilla a escribir y a acordarse.

He aquí un ejemplo de texto en itálica:

```
<i>Texto en itálica</i>
```

Que da el siguiente efecto:

Texto en itálica

Subrayado

El HTML nos propone también para el subrayado el par de etiquetas: <u> y </u> (underlined). Sin embargo, el uso de subrayados ha de ser aplicado con mucha precaución dado que los enlaces hipertexto van, a no ser que se indique lo contrario, subrayados con lo que podemos confundir al lector y apartarlo del verdadero interés de nuestro texto.

Subíndices y supraíndices

Este tipo de formato resulta de extremada utilidad para textos científicos. Las etiquetas empleadas son:

^y para los supraíndices
_y para los subíndices

Aquí tenéis un ejemplo:

```
La <sup>13</sup>CC<sub>3</sub>H<sub>4</sub>ClNOS es un heterociclo alergeno enriquecido
```

El resultado:

La $^{13}CC_3H_4ClNOS$ es un heterociclo alergeno enriquecido

Anidar etiquetas

Todas estas etiquetas y por supuesto el resto de las vistas y que veremos más adelante pueden ser anidadas unas dentro de otras de manera a conseguir resultados diferentes. Así, podemos sin ningún problema crear texto en negrita e itálica embebiendo una etiqueta dentro de la otra:

```
<b>Esto sólo está en negrita <i>y esto en negrita e itálica</i></b>
```

Esto nos daria:

Esto sólo estáa en negrita *y esto en negrita e itálica*

Consejo: Cuando anides etiquetas HTML hazlo correctamente. Nos referimos a que si abres etiquetas dentro de otra más principal, antes de cerrar la etiqueta principal cierres las etiquetas que hayas abierto dentro de ella.

Debemos evitar códigos como el siguiente:
Esto está en negrita e itálica

En favor de códigos con etiquetas correctamente anidadas:
Esto está en negrita e itálica

Esto es muy aconsejable, aunque los navegadores entiendan bien las etiquetas mal anidadas, por dos razones:

1. Sistemas como XML no son tan permisivos con estos errores y puede que en el futuro nuestras páginas no funcionen correcamente.
2. A los navegadores les cuesta mucho tiempo de procesamiento resolver este tipo de errores, incluso más que construir la propia página y debemos evitarles que sufran por una mala codificación.

Color, tamaño y tipo de letra

A pesar de que por razones de homogeneidad y sencillez de codigo este tipo de formatos son controlados actualmente por hojas de estilo en cascada (de las cuales ya tendremos tiempo de hablar), existe una forma clásica y directa de definir color tamaño y tipo de letra de un texto determinado.

Esto se hace a partir de la etiqueta y su cierre correspondiente. Dentro de esta etiqueta deberemos especificar los atributos correspondientes a cada uno de estos parámetros que deseamos definir. A continuación os comentamos los atributos principales de esta etiqueta:

Atributo face

Define el tipo de letra. Este atributo es interpretado por versiones de Netscape a partir de la 3 y de MSIE 3 o superiores. Otros navegadores las ignoran completamente y muestran el texto con la fuente que utilizan.

Hay que tener cuidado con este atributo ya que cada usuario, dependiendo de la plataforma que utilice, puede no disponer de los mismos tipos de letra que nosotros con lo que, si nosotros elegimos un tipo del que no dispone, el navegador se verá forzado a mostrar el texto con la fuente que utiliza por defecto (suele ser Times New Roman). Para evitar esto, dentro del atributo suelen seleccionarse varios tipos de letra separados por comas. En este caso el navegador comprobará que dispone del primer tipo enumerado y si no es así, pasará al segundo y así sucesivamente hasta encontrar un tipo que posea o bien acabar la lista y poner la fuente por defecto. Veamos un ejemplo.

Este texto tiene otra tipografía

Que se visualizaría así en una página web.

Este texto tiene otra tipografía

Nota: Aquí tenemos un ejemplo de atributo cuyo valor debe estar limitado por comillas ("). Habíamos dicho que las comillas eran opcionales en los atributos, sin embargo esto no es así siempre. Si el valor del atributo contiene espacios, como es el caso de:

face="Comic Sans MS,arial,verdana"

debemos colocar las comillas para limitarlo. En caso de no tener comillas

face=Comic Sans MS,arial,verdana

se entendería que face=Comic, pero no se tendría en cuenta todo lo que sigue, porque HTML no lo asociaría al valor del atributo. En este caso HTML pensaría que las siguientes palabras (después del espacio) son otros atributos, pero como no los conoce como atributos simplemente los desestimará.

Atributo size

Define el tamaño de la letra. Este tamaño puede ser absoluto o relativo.

Si hablamos en términos absolutos, existen 7 niveles de tamaño distintos numerados de 1 a 7 por orden creciente. Elegiremos por tanto un valor size="1" para la letra más pequeña o size="7" para la más grande.

Este texto es más grande

Que se visualizaría así en una página web.

Este texto es más grande

Podemos asimismo modificar el tamaño de nuestra letra con respecto al del texto mostrado precedentemente definiendo el número de niveles que queremos subir o bajar en esta escala de tamaños por medio de un signo + o -. De este modo, si definimos nuestro atributo como size="+1" lo que queremos decir es que aumentamos de un nivel el tamaño de la letra. Si estabamos escribiendo previamente en 3, pasaremos automáticamente a 4.
Los tamaños reales que veremos en pantalla dependerán de la definición y del tamaño de fuente elegido por el usuario en el navegador. Este tamaño de fuente puede ser definido en el Explorer yendo al menu superior, Ver/Tamaño de la fuente. En Netscape elegiremos View/Text Size. Esta flexibilidad puede en más de una ocasión resultarnos embarazosa ya que en muchos casos desearemos que el tamaño del texto permanezca constante para que éste quepa en un determinado espacio. Veremos en su momento que esta prefijación del tamaño puede ser llevada a cabo por las hojas de estilo en cascada.

Atributo color

El color del texto puede ser definido mediante el atributo color. Cada color es a su vez definido por un número hexadecimal que esta compuesto a su vez de tres partes. Cada una de estas partes representa la contribución del rojo, verde y azul al color en cuestión.

Podéis entender cómo funciona esta numeración y cuáles son los colores que resultan más compatibles a partir de este artículo: Los colores y HTML.

Por otra parte, es posible definir de una manera inmediata algunos de los colores más frecuentemente usados para los que se ha creado un nombre más memotécnico:

Nombre	Color
Aqua	
Black	
Blue	
Fuchsia	
Gray	
Green	
Lime	
Maroon	
Navy	
Olive	
Purple	
Red	
Silver	
Teal	
White	
Yellow	

Este texto está en rojo

Que se visualizaría así en una página web.

Este texto está en rojo

Con todo esto estamos ya en disposicion de crear un texto formateado de una forma realmente elaborada.

Pongamos pues en practica todo lo que hemos aprendido en estos capitulos haciendo un ejercicio consistente en una página que tenga las siguientes características:

- Un titular con encabezado de nivel 1, en itálica y color verde oliva.
- Un segundo titular con encabezado de nivel 2, también de color verde oliva.
- Todo el texto de la página deberá presentarse con una fuente distinta de la fuente por defecto. Por ejemplo "Comic Sans MS" y en caso de que ésta no esté en el sistema que se coloque la fuente "Arial".

Atributos para páginas

Las páginas HTML pueden construirse con variedad de atributos que le pueden dar un aspecto a la página muy personalizado. Podemos definir atributos como el color de fondo, el color del texto o de los enlaces. Estos atributos se definen en la etiqueta <body> y, como decíamos son generales a toda la página.

Lo mejor para explicar su funcionamiento es verlos uno por uno.

Atributos para fondos

bgcolor: especificamos un color de fondo para la página. En el capítulo anterior y en el taller de los colores y HTML hemos aprendido a construir cualquier color, con su nombre o su valor RGB. El color de fondo que podemos asignar con bgcolor es un color plano, es decir el mismo para toda la superficie del navegador.

background: sirve para indicar la colocación de una imagen como fondo de la página. La imagen se coloca haciendo un mosaico, es decir, se repite muchas veces hasta ocupar todo el espacio del fondo de la página. En capítulos más adelante veremos como se insertan imágenes con HTML y los tipos de imágenes que se pueden utilizar.

Ejemplo de fondo

Vamos a colocar esta imagen como fondo en la página.

La imagen se llama fondo.jpg y suponemos que se encuentra en el mismo directorio que la página. En este caso se colocaría la siguiente etiqueta <body>

```
<body background="fondo.jpg">
```

Se puede ver el efecto de colocar ese fondo en una página a parte.

> **Consejo**: siempre que coloquemos una imagen de fondo, debemos poner también un color de fondo cercano al color de la imagen.
>
> Esto se debe a que, al colocar una imagen de fondo, el texto de la página debemos colocarlo en un color que contraste suficientemente con dicho fondo. Si el visitante no puede ver el fondo por cualquier cuestión (Por ejemplo tener deshabilitada la carga de imágenes) puede que el texto no contraste lo suficiente con el color de fondo por defecto de la web.
>
> Creo que lo mejor será poner un ejemplo. Si la imagen de fondo es oscura, tendremos que poner un texto claro para que se pueda leer. Si el visitante que accede a la página no ve la imagen de fondo, le saldrá el fondo por defecto, que generalmente es blanco, de modo que al tener un texto con color claro sobre un fondo blanco, nos pasará que no podremos leer el texto convenientemente.
>
> Ocurre parecido cuando se está cargando la página. Si todavía no ha llegado a nuestro sistema la imagen de fondo, se verá el fondo que hayamos seleccionado con bgcolor y es interesante que sea parecido al color de la imagen para que se pueda leer el texto mientras se carga la imagen de fondo.

Color del texto

text: este atributo sirve para asignar el color del texto de la página. Por defecto es el negro.

Además del color del texto, tenemos tres atributos para asignar el color de los enlaces de la página. Ya debemos saber que los enlaces deben diferenciarse del resto del texto de la página para que los usuarios puedan identificarlos fácilmente. Para ello suelen aparecer subrayados y con un color más vivo que el texto. Los tres atributos son los siguientes:

link: el color de los enlaces que no han sido visitados. (por defecto es azul clarito)

vlink: el color de los enlaces visitandos. La "v" viene justamente de la palabra visitado. Es el color que tendrán los enlaces que ya hemos visitado. Por defecto su color es morado. Este color debería ser un poco menos vivo que el color de los enlaces normales.

alink: es el color de los enlaces activos. Un enlace está activo en el preciso instante que se pulsa. A veces es difícil darse cuenta cuando un enlace está activo porque en el momento en el que se activa es porque lo estamos pulsando y en ese caso el navegador abandonará la página rápidamente y no podremos ver el enlace activo más que por unos instantes mínimos.

Ejemplo de color del texto

Vamos a ver una página donde el color de fondo sea negro, y los colores del texto y los enlaces sean claros. Pondremos el color de texto balnco y los enlaces amarillos, más resaltados los que no estén visitados y menos resaltados lo que ya están visitados. Para ello escribiríamos la etiqueta body así:

```
<body bgcolor="#000000" text="#ffffff" link="#ffff33" alink="#ffffcc" alink="ffff00">
```

El efecto se puede ver en una página a parte.

Márgenes

Con otros atributos de la etiqueta <body> se pueden asignan espacios de margen en las páginas, lo que es muy útil para eliminar los márgenes en blanco que aparecen a los lados, arriba y debajo de la página. Estos atributos son distintos para Internet Explorer y para Netscape Navigator, por lo que debemos utilizarlos todos si queremos que todos los navegadores los interpreten perfectamente.

leftmargin: para indicar el margen a los lados de la página. Válido para iexplorer.

topmargin: para indicar el margen arriba y debajo de la página. Para iexplorer.

marginwidth: la contrapartida de leftmargin para Netscape. (Margen a los lados)

marginheight: igual que topmargin, pero para Netscape. (Margen arriba y abajo)

Tenemos un artículo sobre la utilización de estos atributos para hacer diseños avanzados con tablas en distintas definiciones de pantalla, que puede ser interesante de leer.

Un ejemplo de página sin margen es la propia página de DesarrolloWeb.com, que estás visitando actualmente. (Por lo menos a la hora de escribir este artículo) Además, vamos a ver otra página sin márgenes, por si alguien necesita ver el ejemplo en estas líneas.

```
<body topmargin=0 leftmargin=0 marginheight=0 marginwidth=0 bgcolor="ffffff">
<table width=100% bgcolor=ff6666><tr><td>
<h1>Hola amigos</h1>
<br>
<br>
Gracias por visitarme!
</td></tr></table>
</body>
```

Esta página tiene el fondo blanco y dentro una tabla con el fondo rojo. En la página podremos ver que la tabla ocupa el espacio en la página sin dejar sitio para ningún tipo de margen.

Listas I

Las posibilidades que nos ofrece el HTML en cuestión de tratamiento de texto son realmente notables. No se limitan a lo visto hasta ahora, sino que van más lejos todavía. Varios ejemplos de ello son las listas, que sirven para enumerar y definir elementos, los textos preformateados y las cabeceras o títulos.

Las listas son utilizadas para citar, numerar y definir objetos. También son utilizadas corrientemente para desplazar el comienzo de línea hacia la derecha.

Podemos distinguir tres tipos de listas:

- Listas desordenadas
- Listas ordenadas
- Listas de definición

Las veremos detenidamente una a una.

Listas desordenadas

Son delimitadas por las etiquetas <ul> y (unordered list). Cada uno de los elementos de la lista es citado por medio de una etiqueta <li> (sin cierre, aunque no hay inconveniente en colocarlo). La cosa queda así:

```
<p>Países del mundo</p>
<ul>
   <li>Argentina
   <li>Perú
   <li>Chile
</ul>
```

El resultado:

Países del mundo

- Argentina
- Perú
- Chile

Podemos definir el tipo de viñeta empleada para cada elemento. Para ello debemos especificarlo por medio del atributo type incluido dentro de la etiqueta de apertura <ul>, si queremos que el estilo sea válido para toda la lista,o dentro le la etiqueta <li> si queremos hacerlo específico de un solo elemento. La sintaxis es del siguiente tipo:

```
<ul type="tipo de viñeta">
```

donde tipo de viñeta puede ser uno de los siguientes:

circle
disc
square

> **Nota:** En algunos navegadores no funciona la opción de cambiar el tipo de viñeta a mostrar y por mucho que nos empeñemos, siempre saldrá el redondel negro.
>
> En caso de que no funcione siempre podemos construir la lista a mano con la viñeta que queramos utilizando las tablas de HTML. Veremos más adelante cómo trabajar con tablas.

Vamos a ver un ejemplo de lista con un cuadrado en lugar de un redondel, y en el último elemento colocaremos un círculo. Para ello vamos a colocar el atributo type en la etiqueta <ul>, con lo que afectará a todos los elementos de la lista.

```
<ul type="square">
<li>Elemento 1
<li>Elemento 2
<li>Elemento 3
<li type="circle">Elemento 4
</ul>
```

Que tiene como resultado

- Elemento 1
- Elemento 2
- Elemento 3
- Elemento 4

Listas II

Continuamos estudiando las listas de HTML, con las que crear estructuras atractivas para presentar la información.

Listas ordenadas

En este caso usaremos las etiquetas <ol> (ordered list) y su cierre. Cada elemento sera igualmente precedido de su etiqueta <li>.

Pongamos un ejemplo:

```
<p>Reglas de comportamiento en el trabajo</p>
<ol>
<li>El jefe siempre tiene la razón
<li>En caso de duda aplicar regla 1
</ol>
```

El resultado es:

Reglas de comportamiento en el trabajo

1. El jefe siempre tiene la razón
2. En caso de duda aplicar regla 1

Del mismo modo que para las listas desordenadas, las listas ordenadas ofrecen la posibilidad de modificar el estilo. En concreto nos es posible especificar el tipo de numeración empleado eligiendo entre números (1, 2, 3...), letras (a, b, c...) y sus mayúsculas (A, B, C,...) y números romanos en sus versiones mayúsculas (I, II, III,...) y minúsculas (i, ii, iii,...).

Para realizar dicha selección hemos de utilizar, como para el caso precedente, el atributo type, el cual será situado dentro de la etiqueta <ol>. Los valores que puede tomar el atributo en este caso son:

1 Para ordenar por números
a Por letras del alfabeto
A Por letras mayúsculas del alfabeto
i Ordenación por números romanos en minúsculas
I Ordenación por números romanos en mayúsculas

> **Nota:** Recordamos que en algunos navegadores no funciona la opción de cambiar el tipo de viñeta a mostrar

Puede que en algún caso deseemos comenzar nuestra enumeración por un número o letra que no tiene por qué ser necesariamente el primero de todos. Para solventar esta situación, podemos utilizar un segundo atributo, start, que tendra como valor un número. Este número, que por defecto es 1, corresponde al valor a partir del cual comenzamos a definir nuestra lista. Para el caso de las letras o los números romanos, el navegador se encarga de hacer la traducción del número a la letra correspondiente.

Os proponemos un ejemplo usando este tipo de atributos:

```
<p>Ordenamos por numeros</p>
<ol type="1">
<li>Elemento 1
<li> Elemento 2
</ol>

<p>Ordenamos por letras</p>
<ol type="a">
<li>Elemento a
<li> Elemento b
</ol>

<p>Ordenamos por números romanos empezando por el 10</p>

<ol type="i" start="10">
<li>Elemento x
<li> Elemento xi
</ol>
```

El resultado:

Ordenamos por números

1. Elemento 1
2. Elemento 2

Ordenamos por letras

a. Elemento a
b. Elemento b

Ordenamos por numeros romanos empezando por el 10

x. Elemento x
xi. Elemento xi

Listas III

Terminamos el tema de listas estudiando las listas de definición. Veremos también la anidación de listas.

Listas de definición

Cada elemento es presentado junto con su definición. La etiqueta principal es <dl> y </dl> (definition list). La etiquetas del elemento y su definición son <dt> (definition term) y <dd> (definition definition) respectivamente.

Aquí os proponemos un código que podrá aclarar este sistema:

```
<p>Diccionario de la Real Academia</p>
<dl>
   <dt>Brujula
   <dd>Señórula montada en una escóbula
   <dt>Oreja
   <dd>Sesenta minutejos
</dl>
```

El efecto producido:

Diccionario de la Real Academia

Brujula
: Señórula montada en una escóbula

Oreja
: Sesenta minutejos

Fijaos en que cada línea <dd> esta desplazada hacia la izquierda. Este tipo de etiquetas son usadas a menudo con el propósito de crear textos más o menos desplazados hacia la izquierda.

El código:

```
<dl>
<dd>Primer nivel de desplazamiento
   <dl>
     <dd>Segundo nivel de desplazamiento
     <dl>
       <dd>Tercer nivel de desplazamiento
     </dl>
   </dl>
</dl>
```

El resultado:

: Primer nivel de desplazamiento
: Segundo nivel de desplazamiento
: Tercer nivel de desplazamiento

Anidando listas

Nada nos impide utilizar todas estas etiquetas de forma anidada como hemos visto en otros casos. De esta forma, podemos conseguir listas mixtas como por ejemplo:

```
<p>Ciudades del mundo</p>
<ul>
   <li>Argentina
   <ol>
     <li>Buenos Aires
     <li>Bariloche
   </ol>
   <li>Uruguay
   <ol>
     <li>Montevideo
     <li>Punta del Este
   </ol>
</ul>
```

De esta forma creamos una lista como esta:

Ciudades del mundo

- Argentina
    1. Buenos Aires
    2. Bariloche
- Uruguay
    1. Montevideo
    2. Punta del Este

Caracteres especiales

Una página web se ha de ver en paises distintos, que usan conjuntos de caracteres distintos. El lenguaje HTML nos ofrece un mecanismo por el que podemos estar seguros que una serie de caracteres raros se van a ver bien en todos los ordenadores del mundo, independientemente de su juego de caracteres.

Este conjunto son los caracteres especiales. Cuando queremos poner uno de estos caracteres en una página, debemos sustituirlo por su códico.

Por ejemplo, la "**á**" (a minúscula acentuada) se escribe "**á**" de modo que la palabra página se escribiría en una página HTML de este modo: p&aamp;aacute;gina

Caracteres especiales básicos

En realidad estos caracteres se usan en HTML para no confundir un principio o final de etiqueta, unas comillas o un & con su correspondiente caracter.

<	<	>	>
&	&	"	"

Caracteres especiales del HTML 2.0

Á	Á	À	À
É	É	È	È
Í	Í	Ì	Ì
Ó	Ó	Ò	Ò
Ú	Ú	Ù	Ù
á	á	à	à
é	é	è	è
í	í	ì	ì
ó	ó	ò	ò

ú	ú	ù	ù
Ä	Ä	Â	Â
Ë	Ë	Ê	Ê
Ï	Ï	Î	Î
Ö	Ö	Ô	Ô
Ü	Ü	Û	Û
ä	ä	â	â
ë	ë	ê	ê
ï	ï	î	î
ö	ö	ô	ô
ü	ü	û	û
Ã	Ã	å	å
Ñ	Ñ	Å	Å
Õ	Õ	Ç	Ç
ã	ã	ç	ç
ñ	ñ	Ý	Ý
õ	õ	ý	ý
Ø	Ø	ÿ	ÿ
ø	ø	Þ	Þ
Ð	Ð	þ	þ
ð	ð	Æ	Æ
ß	ß	æ	æ

Caracteres especiales del HTML 3.2

¼	¼		
½	½	¡	¡
¾	¾	£	£
©	©	¥	¥
®	®	§	§
ª	ª	¤	¤
²	²	¦	¦
³	³	«	«
¹	¹	¬	¬
¯	¯		
µ	µ	º	º
¶	¶	´	´
·	·	¨	¨
°	°	±	±
¸	¸	»	»
¿	¿		

Otros caracteres especiales

×	×	¢	¢
÷	÷	€	€
“	“	™	™
”	”	‰	‰
Œ	Œ	ƒ	*f*
‡	‡	†	†

Enlaces en HTML

Hasta aquí, hemos podido ver que una página web es un archivo HTML en el que podemos incluir, entre otras cosas, textos formateados a nuestro gusto e imágenes (las veremos enseguida). Del mismo modo, un sitio web podrá ser considerado como el conjunto de archivos, principalmente páginas HTML e imágenes, que constituyen el contenido al que el navegante tiene acceso.

Sin embargo, no podríamos hablar de navegante o de navegación si estos archivos HTML no estuviesen debidamente conectados entre ellos y con el exterior de nuestro sitio por medio de enlaces hipertexto. En efecto, el atractivo original del HTLM radica en la posible puesta en relación de los contenidos de los archivos introduciendo referencias bajo forma de enlaces que permitan un acceso rápido a la información deseada. De poco serviría en la red tener páginas aisladas a las que la gente no puede acceder y desde las que la gente no puede saltar a otras.

Un enlace puede ser fácilmente detectado en una página. Basta con deslizar el puntero del ratón sobre las imágenes o el texto y ver como cambia de su forma original transformándose por regla general en una mano con un dedo señalador. Adicionalmente, estos enlaces suelen ir, en el caso de los textos, coloreados y subrayados para que el usuario no tenga dificultad en reconocerlos. Si no especificamos lo contrario (ya tendremos ocasión de explicar como), estos enlaces texto estarán subrayados y coloreados en azul. En el caso de las imágenes que sirvan de enlace, veremos que están delimitadas por un marco azul por defecto.

Para colocar un enlace, nos serviremos de las etiquetas <a> y . Dentro de la etiqueta de apertura deberemos especificar asimismo el destino del enlace. Este destino será introducido bajo forma de atributo, el cual lleva por nombre href.

La sintaxis general de un enlace es por tanto de la forma:

contenido

Siendo el *contenido* un texto o una imagen. Es la parte de la página que se colocará activa y donde deberemos pulsar para acceder al enlace.

Por su parte, *destino* será una página, un correo electrónico o un archivo.

En función del destino los enlaces son clásicamente agrupados del siguiente modo:

- **Enlaces internos**: los que se dirigen a otras partes dentro de la misma página.
- **Enlaces locales**: los que se dirigen a otras páginas del mismo sitio web.
- **Enlaces remotos**: los dirigidos hacia páginas de otros sitios web.
- **Enlaces con direcciones de correo**: para crear un mensaje de correo dirigido a una dirección.
- **Enlaces con archivos**: para que los usuarios puedan hacer download de ficheros.

Enlaces internos

Son los enlaces que apuntan a un lugar diferente dentro de la misma página. Este tipo de enlaces son esencialmente utilizados en páginas donde el acceso a los contenidos puede verse dificultado debido al gran tamaño de la misma. Mediante estos enlaces podemos ofrecer al visitante la posibilidad de acceder rápidamente al principio o final de la página o bien a diferentes párrafos o secciones.

Para crear un enlace de este tipo es necesario, aparte del enlace de origen propiamente dicho, un segundo enlace que será colocado en el destino. Veamos más claramente como funcionan estos enlaces con un ejemplo sencillo:

Supongamos que queremos crear un enlace que apunte al final de la página. Lo primero será colocar nuestro enlace origen. Lo pondremos aquí mismo y lo escribiremos del siguiente modo:

```
<a href="#abajo">Ir abajo</a>
```

Enlace con final de este documento, para que probéis su funcionamiento:

Ir abajo

Como podéis ver, el contenido del enlace es el texto "Ir abajo" y el destino, abajo, es un punto de la misma página que todavía no hemos definido. Ojo al símbolo #; es él quien especifica al navegador que el enlace apunta a una sección en particular.

En segundo lugar, hay que generar un enlace en el destino. Este enlace llevara por nombre abajo para poder distinguirlo de los otros posibles enlaces realizados dentro de la misma página. En este caso, la etiqueta que escribiremos será ésta:

```
<a name="abajo"></a>
```

A decir verdad, estos enlaces, aunque útiles, no son los más extendidos de cuantos hay. La tendencia general es la de crear páginas (archivos) independientes con tamaños más reducidos enlazados entre ellos por enlaces locales (los veremos enseguida). De esta forma evitamos el exceso de tiempo de carga de un archivo y la introducción de exceso de información que pueda desviar la atención del usuario.

Una aplicación corriente de estos enlaces consiste en poner un pequeño índice al principio de nuestro documento donde introducimos enlaces origen a las diferentes secciones. Paralelamente, al final de cada sección introducimos un enlace que apunta al índice de manera que podamos guiar al navegante en la búsqueda de la información útil para él.

Enlaces locales

Como hemos dicho, un sitio web esta constituido de páginas interconexas. En el capitulo anterior hemos visto como enlazar distintas secciones dentro de una misma página. Nos queda pues estudiar la manera de relacionar los distintos documentos HTML que componen nuestro sitio web.

Para crear este tipo de enlaces, hemos de crear una etiqueta de la siguiente forma:

```
<a href="archivo.html">contenido</a>
```

Por regla general, para una mejor organización, los sitios suelen estar ordenados por directorios. Estos directorios suelen contener diferentes secciones de la página, imágenes, sonidos...Es por ello que en muchos casos no nos valdrá con especificar el nombre del archivo, sino que tendremos que especificar además el directorio en el que nuestro archivo.html esta alojado.

Si habéis trabajado con MS-DOS no tendréis ningún problema para comprender el modo de funcionamiento. Tan solo hay que tener cuidado en usar la barra "/" en lugar de la contrabarra "\".

Para aquellos que no saben como mostrar un camino de un archivo, aquí van una serie de indicaciones que os ayudaran a comprender la forma de expresarlos. No resulta difícil en absoluto y con un poco de practica lo haréis prácticamente sin pensar.

1. Hay que situarse mentalmente en el directorio en el que se encuentra la página con el enlace.
2. Si la página destino esta en un directorio incluido dentro del directorio en el que nos encontramos, hemos de marcar el camino enumerando cada uno de los directorios por los que pasamos hasta llegar al archivo y separándolos por el símbolo barra "/". Al final obviamente, escribimos el archivo.
3. Si la página destino se encuentra en un directorio que incluye el de la página con el enlace, hemos de escribir dos puntos y una barra "../" tantas veces como niveles subamos en la arborescencia hasta dar con el directorio donde esta emplazado el archivo destino.
4. Si la página se encuentra en otro directorio no incluido ni incluyente del archivo origen, tendremos que subir como en la regla 3 por medio de ".." hasta encontrar un directorio que englobe el directorio que contiene a la página destino. A continuación haremos como en la regla 2. Escribiremos todos los directorios por los que pasamos hasta llegar al archivo.

Ejemplo:

Para clarificar este punto podemos hacer un ejemplo a partir de la estructura de directorios de la imagen.

Para hacer un enlace desde index.html hacia yyy.html:

```
<a href="seccion1/paginas/yyy.html">contenido</a>
```

```
Para hacer un enlace desde xxx.html hacia yyy.html:
<a href="../seccion1/paginas/yyy.html">contenido</a>

Para hacer un enlace desde yyy.html hacia xxx.html:
<a href="../../seccion2/xxx.html">contenido</a>
```

Los enlaces locales pueden a su vez apuntar ya no a la página en general sino más precisamente a una sección concreta. Este tipo de enlaces resultan ser un híbrido de interno y local. La sintaxis es de este tipo:

```
<a href="archivo.html#seccion">contenido</a>
```

Como para los enlaces internos, en este caso hemos de marcar la sección con otro enlace del tipo:

```
<a name="seccion"></a>
```

Enlaces externos, de correo y hacia archivos.

Para acabar con los enlaces vamos a ver los últimos 3 tipos de enlaces que habíamos señalado.

Enlaces remotos

Son los enlaces que se dirigen hacia páginas que se encuentran fuera de nuestro sitio web, es decir, cualquier otro documento que no forma parte de nuestro sitio.

Este tipo de enlaces es muy común y no representa ninguna dificultad. Simplemente colocamos en el atributo HREF de nuestra etiqueta <A> la URL o dirección de la página con la que queremos enlazar. Será algo parecido a esto.

```
<a href="http://www.guiarte.com">ir a guiarte.com</a>
```

Sólo cabe destacar que todos las direcciones web (URLs) empiezan por **http://**. Esto indica que el protocolo por el que se accede es HTTP, el utilizado en la web. No debemos olvidarnos de colocarlas, porque si no los enlaces serán tratados como enlaces locales a nuestro sitio.

Otra cosa interesante es que no tenemos que enlazar con una página web con el protocolo HTTP necesariamente. También podemos acceder a recursos a través de otros protocolos como el FTP. En tal caso, las direcciones de los recursos no comenzarán por http:// sino por ftp://.

Enlaces a direcciones de correo

Los enlaces a direcciones de correo son aquellos que al pincharlos nos abre un nuevo mensaje de correo electrónico dirigido a una dirección de mail determinada. Estos enlaces son muy habituales en las páginas web y resultan la manera más rápida de ofrecer al visitante una vía para el contacto con el propietario de la página.

Para colocar un enlace dirigido hacia una dirección de correo colocamos **mailto:** en el atributo href del enlace, seguido de la dirección de correo a la que se debe dirigir el enlace.

```
<a href="mailto:eugim@desarrolloweb.com">eugim@desarrolloweb.com</a>
```

> **Consejo:** Cuando coloques enlaces a direcciones de correo procura indicar en el contenido del enlace (lo que hay entre <A> y </A>) la dirección de correo a la que se debe escribir. Esto es porque si un usuario no tiene configurado un programa de correo en su ordenador no podrá enviar mensajes, pero por lo menos podrá copiar la dirección de mail y escribir el correo a través de otro ordenador o un sistema web-mail.

Además de la dirección de correo del destinatario, también podemos colocar en el enlace el asunto del mensaje. Esto se consigue colocando después de la dirección de correo un interrogante, la palabra subject, un signo igual (=) y el asunto en concreto.

```
<a href="malito:eugim@desarrolloweb.com?subject=contacto a través de la
pagina">eugim@desarrolloweb.com</a>
```

Podemos colocar otros atributos del mensaje con una sintaxis parecida. En este caso indicamos también que el correo debe ir con copia a colabora@desarrolloweb.com.

```
<a href="mailto:eugim@desarrolloweb.com?subject=contacto a través de la
pagina&cc=colabora@desarrolloweb.com">eugim@desarrolloweb.com</a>
```

Nota: El visitante de la página necesitará tener configurada una cuenta de correo electrónico en su sistema para enviar los mensajes. Lógicamente, si no tiene servicio de correo en el ordenador no se podrán enviar los mensajes y este sistema de contacto con el visitante no funcionará.

Enlaces con archivos

Este no es un tipo de enlace propiamente dicho, pero lo señalamos aquí porque son un tipo de enlaces muy habitual y que presenta alguna complicación para el usuario novato.

El mecanismo es el mismo que hemos conocido en los enlaces locales y los enlaces remotos, con la única particularidad de que en vez de estar dirigidos hacia una página web está dirigido hacia un archivo de otro tipo.

Si queremos enlazar con un archivo mi_fichero.zip que se encuentra en el mismo directorio que la página se escribiría un enlace así.

```
<a href="mi_fichero.zip">Descarga mi_fichero.zip</a>
```

Si pinchamos un enlace de este tipo nuestro navegador descargará el fichero, haciendo la pregunta típica de "Qué queremos hacer con el archivo. Abrirlo o guardarlo en disco".

Consejo: No colocar en Internet archivos ejecutables directamente sino archivos comprimidos. Por dos razones:

1. El archivo ocupará menos, con lo que será más rápida su transferencia.
2. Al preguntar al usuario lo que desea hacer con el fichero le ofrece la opción de abrirlo y guardarlo en disco. Nosotros generalmente desearemos que el usuario lo guarde en disco y no lo ejecute hasta que lo tenga en su disco duro. Si se decido a abrirlo en vez de guardarlo simplemente lo pondrá en marcha y cuando lo pare no se quedará guardado en su sistema. Si los archivos están comprimidos obligaremos al usuario a descomprimirlos en su disco duro antes de ponerlos en marcha, con lo que nos aseguramos que el usuario lo guarde en su ordenador antes de ejecutarlo.

Si queremos enlazar hacia otro tipo de archivo como un PDF o un mundo VRML (Realidad virtual para Internet) lo seguimos haciendo de la misma manera. El navegador, si reconoce el tipo de archivo, es el responsable de abrirlo utilizando el conector adecuado para ello. Así, si por ejemplo enlazamos con un PDF pondrá el programa Acrobat Reader en funcionamiento para mostrar los contenidos. Si enlazamos con un mundo VRML pondrá en marcha el plug-in que el usuario tenga instalado para ver los mundos virtuales (Cosmo Player por ejemplo).

Este sería un ejemplo de enlace a un documento PDF.

```
<a href="mi_documento_pdf">Descarga el PDF</a>
```

Imágenes en HTML

Sin duda uno de los aspectos más vistosos y atractivos de las páginas web es el grafismo. La introducción en nuestro texto de imágenes puede ayudarnos a explicar más fácilmente nuestra información y darle un aire mucho más estético. El abuso no obstante puede conducirnos a una

sobrecarga que se traduce en una distracción para el navegante, quien tendrá más dificultad en encontrar la información necesaria, y un mayor tiempo de carga de la página lo que puede ser de un efecto nefasto si nuestro visitante no tiene una buena conexión o si es un poco impaciente.

En este capitulo no explicaremos como crear ni tratar las imágenes, únicamente diremos que para ello se utilizan aplicaciones como Paint Shop Pro, Photoshop o Corel Draw. Tampoco explicaremos las particularidades de cada tipo de archivo GIF o JPG y la forma de optimizar nuestras imágenes. Un capitulo posterior al respecto será dedicado a este menester: Formatos gráficos para páginas web.

Las imágenes son almacenadas en forma de archivos, principalmente GIF (para dibujos) o JPG (para fotos). Estos archivos pueden ser creados por nosotros mismos o pueden ser descargados gratuitamente en sitios web especializados. En desarrolloweb contamos con la mayor base de datos de gifs animados e imágenes de todo tipo en castellano, que nos provee el sitio internacional GOgraph.

Así pues, en estos primeros capítulos nos limitaremos a explicar como insertar y alinear debidamente en nuestra página una imagen ya creada.

La etiqueta que utilizaremos para insertar una imagen es <img> (image). Esta etiqueta no posee su cierre correspondiente y en ella hemos de especificar obligatoriamente el paradero de nuestro archivo grafico mediante el atributo src (source).

La sintaxis queda entonces de la siguiente forma:

```
<img src="camino hacia el archivo">
```

Para expresar el camino, lo haremos de la misma forma que vimos para los enlaces. Las reglas siguen siendo las mismás, lo único que cambia es que, en lugar de una página destino, el destino es un archivo grafico.

Aparte de este atributo, indispensable obviamente para la visualización de la imagen, la etiqueta <img> nos propone otra serie de atributos de mayor o menor utilidad:

Atributo alt

Dentro de las comillas de este atributo colocaremos una brevísima descripción de la imagen. Esta etiqueta no es indispensable pero presenta varias utilidades.

Primeramente, durante el proceso de carga de la página, cuando la imagen no ha sido todavía cargada, el navegador mostrara esta descripción, con lo que el navegante se puede hacer una idea de lo que va en ese lugar.

Esto no es tan trivial si tenemos en cuenta que algunos usuarios navegan por la red con una opción del navegador que desactiva el muestreo de imágenes, con lo que tales personas podrán siempre saber de qué se trata el grafico y eventualmente cambiar a modo con imágenes para visualizarla.

Además, determinadas aplicaciones para discapacitados o teléfonos vocales que no muestran imágenes ofrecen la posibilidad de leerlas por lo que nunca esta de más pensar en estos colectivos.

En general podemos considerar como aconsejable el uso de este atributo salvo para imágenes de poca importancia y absolutamente indispensable si la imagen en cuestión sirve de enlace.

Atributos height y width

Definen la altura y anchura respectivamente de la imagen en pixels.

Todos los archivos gráficos poseen unas dimensiones de ancho y alto. Estas dimensiones pueden obtenerse a partir del propio diseñador grafico o bien haciendo clic con el botón derecho sobre la imagen vista por el navegador para luego elegir propiedades sobre el menú que se despliega.

El hecho de explicitar en nuestro código las dimensiones de nuestras imágenes ayuda al navegador a confeccionar la página de la forma que nosotros deseamos antes incluso de que las imágenes hayan sido descargadas.

Así, si las dimensiones de las imágenes han sido proporcionadas, durante el proceso de carga, el navegador reservara el espacio correspondiente a cada imagen creando una maquetación correcta. El usuario podrá comenzar a leer tranquilamente el texto sin que este se mueva de un lado a otro cada vez que una imagen se cargue.

Además de esta utilidad, el alterar los valores de estos dos atributos, es una forma inmediata de redimensionar nuestra imagen. Este tipo de utilidad no es aconsejable dado que, si lo que pretendemos es aumentar el tamaño, la perdida de calidad de la imagen será muy sensible. Inversamente, si deseamos disminuir su tamaño, estaremos usando un archivo más grande de lo necesario para la

imagen que estamos mostrando con lo que aumentamos el tiempo de descarga de nuestro documento innecesariamente.

Es importante hacer hincapié en este punto ya que muchos debutantes tienen esa mala costumbre de crear gráficos pequeños redimensionando la imagen por medio de estos atributos a partir de archivos de tamaño descomunal. Hay que pensar que el tamaño de una imagen con unas dimensiones de la mitad no se reduce a la mitad, sino que resulta ser aproximadamente 4 veces inferior.

Atributo border

Definen el tamaño en pixels del cuadro que rodea la imagen.

De esta forma podemos recuadrar nuestra imagen si lo deseamos. Es particularmente útil cuando deseamos eliminar el borde que aparece cuando la imagen sirve de enlace. En dicho caso tendremos que especificar border="0".

Atributos vspace y hspace

Sirven para indicar el espacio libre, en pixeles, que tiene que colocarse entre la imagen y los otros elementos que la rodean, como texto, otras imágenes, etc.

Atributo lowsrc

Con este atributo podemos indicar un archivo de la imagen de baja resolución. Cuando el navegador detecta que la imagen tiene este atributo primero descarga y muestra la imagen de baja resolución (que ocupa muy poco y que se transfiere muy rápido). Posteriormente descarga y muestra la imagen de resolución adecuada (señalada con el atributo src, que se supone que ocupará más y será más lenta de transferir).

Este atributo está en desuso, aunque supone una ventaja considerable para que la descarga inicial de la web se realice más rápido y que un visitante pueda ver una muestra de la imagen mientras se descarga la imagen real.

Truco: Utilizar imagenes como enlaces

Ni que decir tiene que una imagen, lo mismo que un texto, puede servir de enlace. Vista la estructura de los enlaces podemos muy fácilmente adivinar el tipo de código necesario:

```
<a href="archivo.html"><img src="imagen.gif"></a>
```

Ejemplo práctico

Resultará obvio para los lectores hacer ahora una página que contenga una imagen varias veces repetida pero con distintos atributos.

- Una de las veces que salga debe mostrarse con su tamaño originar y con un borde de 3 pixeles.
- En otra ocasión la imagen aparecerá sin borde, con su misma altura y con una anchura superior a la original
- También mostraremos la imagen sin borde, con su misma anchura y con una altura superior a la original
- Por último, mostraremos la imagen con una altura y anchura mayores que las originales, pero proporcionalmente igual que antes.

Vamos a utilizar esta imagen para hacer el ejercicio:

Las dimensiones originales de la imagen son 28x21, así que este sería el código fuente:

```
<img src="img1.gif" width="28" height="21" alt="Tamaño original" border="3">
<br>
<br>
<img src="img1.gif" width="68" height="21" alt="Achatada" border="0">
<br>
<br>
```

```
<img src="img1.gif" width="28" height="51" alt="Alargada" border="0">
<br>
<br>
<img src="img1.gif" width="56" height="42" alt="Doble grande" border="0">
```

Alineación de imágenes con HTML

Vimos en su momento el atributo align que nos permitía alinear el texto a derecha, izquierda o centro de nuestra página. Dijimos que este atributo no era exclusivo de la etiqueta <p> sino que podía ser encontrado en otro tipo de etiquetas.

Pues bien, <img> resulta ser una de esas etiquetas que aceptan este atributo aunque en este caso el funcionamiento resulta ser diferente.

Para alinear una imagen horizontalmente podemos hacerlo de la misma forma que el texto, es decir, utilizando el atributo align dentro de una etiqueta <p> o . En este caso, lo que incluiremos dentro de esa etiqueta será la imagen en lugar del texto:

Este código mostrará la imagen en el centro:

```
<div align="center"><img src="logo.gif"></div>
```

Quedaría así:

Sin embargo, ya hemos dicho que la etiqueta <img> puede aceptar el atributo align. En este caso, la utilidad que le damos difiere de la anterior.

El hecho de utilizar el atributo align dentro de la etiqueta <img> nos permite, en el caso de darle los valores left o right, justificar la imagen del lado que deseamos a la vez que rellenamos con texto el lado opuesto. De esta forma embebemos nuestras imágenes dentro del texto de una manera sencilla.

Aquí podéis ver el tipo de código a crear para obtener dicho efecto:

```
<p>
<img src="imagen.gif" align="right">Texto tan extenso como queramos que cubrirá la parte izquierda de la imagen. Sigo poniendo texto para que se vea el efecto, Bla bla bla bla bla bla bla...
</p>
```

Quedaría así:

Texto tan extenso como queramos que cubrirá la parte izquierda de la imagen. Sigo poniendo texto para que se vea el efecto, Bla bla bla bla bla bla bla...

```
<p>
<img src="imagen.gif" align="left">Texto tan extenso como queramos que cubrirá la parte derecha de la imagen. Sigo poniendo texto para que se vea el efecto, Bla bla bla bla bla bla bla...
</p>
```

Quedaría así:

Texto tan extenso como queramos que cubrirá la parte izquierda de la imagen. Sigo poniendo texto para que se vea el efecto, Bla bla bla bla bla bla bla...

Si en algún momento deseásemos dejar de rellenar ese espacio lateral, podemos pasar a una zona libre introduciendo un salto de línea
 dentro del cual añadiremos un atributo: clear

Así, etiquetas del tipo:

<br clear="left">
Saltara verticalmente hasta encontrar el lateral izquierdo libre.
<br clear="right">
Saltara verticalmente hasta encontrar el lateral derecho libre.
<br clear="all">
Saltará verticalmente hasta encontrar ambos laterales libres.

Ejemplo de clear:

Texto tan extenso como queramos que cubrirá la parte izquierda.

Esto está debajo de la imagen.

Existen otro tipo de valores que puede adoptar el atributo align dentro de la etiqueta <img>. Estos son relativos a la alineación vertical de la imagen.
Supongamos que escribimos una línea al lado de nuestra imagen. Esta línea puede quedar por ejemplo arriba, abajo o al medio de la imagen. Asimismo, puede que en una misma línea tengamos varias imágenes de alturas diferentes que pueden ser alineadas de distintas formás.
Estos valores adicionales del atributo align son:

top
Ajusta la imagen a la parte más alta de la línea. Esto quiere decir que, si hay una imagen más alta, ambas imágenes presentaran el borde superior a la misma altura.

bottom
Ajusta el bajo de la imagen al texto.

Absbottom
Colocara el borde inferior de la imagen a nivel del elemento más bajo de la línea.

middle
Hace coincidir la base de la línea de texto con el medio vertical de la imagen.

absmiddle
Ajusta la imagen al medio absoluto de la línea.

Estas explicaciones, que pueden resultar un poco complicadas, pueden ser más fácilmente asimiladas a partir con un poco de practica.

Nos queda explicar como introducir debajo de la imagen un pie de foto o explicación. Para ello tendremos que ver antes de nada las tablas, en el próximos capítulos...

Formátos gráficos para páginas web

El componente gráfico de las páginas web tiene mucha importancia, es el que hace que estas sean vistosas y el que nos permite aplicar nuestra creatividad para hacer del diseño de sitios una tarea agradable. Es también una herramienta para acercar los sitios al mundo donde vivimos, si embargo, es también el causante de errores graves en las páginas y hacer de estas, en algunos casos, un martirio para el visitante.

Las nociones básicas para el uso de archivos gráficos son sencillas, conocerlas, aunque sea ligeramente, nos ayudará a crear sitios agradables y rápidos. No cometer errores en el uso de las imágenes es fundamental, aunque no seas un diseñador y las imágenes que utilices sean feas, utilízalas bien y así estarás haciendo más agradable la visita a tus páginas.

Tipos de archivos

En Internet se utilizan principalmente dos tipos de archivos gráficos GIF y JPG, pensados especialmente para optimizar el tamaño que ocupan en disco, ya que los archivos pequeños se transmiten más rápidamente por la Red.

El formato de archivo GIF se usa para las imágenes que tengan dibujos, mientras que el formato JPG se usa para las fotografías. Los dos comprimen las imágenes para guardarlas. La forma de comprimir la imagen que utiliza cada formato es lo que los hace ideales para unos u otros propósitos.

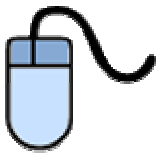
Adicionalmente, se puede usar un tercer formato gráfico en las páginas web, el PNG. Este formato no tiene tanta aceptación como el GIF o JPG por varias razones, entre las que destacan el desconocimiento del formato por parte de los desarrolladores, que las herramientas habituales para tratar gráficos (como por ejemplo Photoshop) generalmente no lo soportan y que los navegadores antiguos también tienen problemas para visualizarlas. Sin embargo, el formato se comporta muy bien en cuanto a compresión y calidad del gráfico conseguido, por lo que resultaría útil si se llega a extender su uso.

GIF

A parte de ser un archivo ideal para las imágenes que estén dibujadas tiene muchas otras características que son importantes y útiles.

Compresión: Es muy buena para dibujos, como ya hemos dicho. Incluso puede ser interesante si la imagen es muy pequeña, aunque sea una foto.

Transparencia: es una utilidad para definir ciertas partes del dibujo como transparentes. De este modo podemos colocar las imágenes sobre distintos fondos sin que se vea el cuadrado donde está inscrito la el dibujo, viendose en cambio la silueta del dibujo en cuestión.
Para crear un gif transparente debemos utilizar un programa de diseño gráfico, con el podemos indicar qué colores del dibujo queremos que sean transparentes. Generalmente, definimos la transparencia cuando vamos a guardar el gráfico.
Colores: Con este formato gráfico podemos utilizar paletas, conjuntos, de 256 colores o menos. Este es un detalle muy importante, puesto que cuantos menos colores utilicemos en la imagen, por lo general, menos ocupará el archivo. En ocasiones, aunque utilicemos menos colores en un gráfico, este no pierde mucho en calidad, llegando a ser inapreciable a la vista.
En algunos programas podemos modificar la cantidad de colores al guardar el archivo, en otros lo hacemos mientras creamos el gráfico.



Parte de esta imagen es transparente

32 colores

16 colores

8 colores

Imagen tomada con distintas paletas de colores. Se puede apreciar como con pocos colores se ve bien el gráfico y como pierde un poco a medida que le restamos colores.

JPG

Veamos ahora cuales son las características fundamentales del formato JPG:

Una fotografía con formato JPG

Compresión: Tal como hemos dicho anteriormente, su algoritmo de compresión hace ideal este formato para guardar fotografías. Además, con JPG podemos definir la calidad de la imagen, con calidad baja el fichero ocupará menos, y viceversa.

Transparencia: Este formato no tiene posibilidad de crear áreas transparentes. Si deseamos colocar una imagen con un área que parezca transparente procederemos así: con nuestro programa de diseño gráfico haremos que el fondo de la imagen sea el mismo que el de la página donde queremos colocarla. En muchos casos los fondos de la imagen y la página parecerán el mismo.
Colores: JPG trabaja siempre con 16 millones de colores, ideal para fotografías.

Optimizar ficheros

Para que las imágenes ocupen lo menos posible y se transfieran rápidamente por la Red debemos aprender a optimizar los ficheros gráficos. Para ello debemos hacer lo siguiente:

Para los archivos GIF: Reduciremos el número de colores de nuestra paleta. Esto se hace con nuestro editor gráfico, en muchos casos podremos hacerlo al guardar el archivo.

GIF 256 colores - 10,8 KB

GIF 32 colores - 5,5 KB

GIF 4 colores - 2 KB

Para los archivos JPG: Ajustaremos la calidad del archivo cuando lo estemos guardando. Este formato nos permite bajar mucho la calidad de la imagen sin que esta pierda mucho en su aspecto visual.


JPG
calidad 0
3 KB


JPG
calidad 20
5,9 KB


JPG
calidad 50
10 KB

Es imprescindible disponer para optimizar la imagen de una herramienta buena que nos permita configurar estas características de la imagen con libertad y fácilmente. Photoshop 5.5 o 6 es un programa bastante recomendable, pues incorpora una opción que se llama "Guardar para el Web" con la que podemos definir los colores del gif, calidad del JPG y otras opciones en varias muestras a la vez. Así con todas las opciones configurables, viendo los resultados a la vez que el tamaño del archivo podemos optimizar la imagen de una manera precisa con los resultados que deseamos.

Photoshop es una herramienta excelente para optimizar ficheros. Viendo varias copias podemos elegir la más adecuada.

También existen en el mercado otros programas que nos permiten optimizar estas imágenes de manera sorprendente. Una vez hemos creado la imagen la pasamos por estos programas y nos comprimen aun más el archivo, haciéndolo rápido de transferir y, por tanto, más optimo para Internet. Al ser estas utilidades tan especializadas los resultados suelen ser mejores que con los programas de edición gráfica.

Ejemplos de optimizadores gráficos:
- WebGraphics Optimizer
- ProJPG, GIF Imantion

Y con versiones Online:
- JPG - GIF Crunchers
- GIF Wizard

Tablas en HTML

Una tabla en un conjunto de celdas organizadas dentro de las cuales podemos alojar distintos contenidos.

En un principio nos podría parecer que las tablas son raramente útiles y que pueden ser utilizadas principalmente para listar datos como agendas, resultados y otros datos de una forma organizada. Nada más lejos de la realidad.

Hoy, gran parte de los diseñadores de páginas basan su maquetación en este tipo de artilugios. En efecto, una tabla nos permite organizar y distribuir los espacios de la manera más optima. Nos puede ayudar a generar texto en columnas como los periódicos, prefijar los tamaños ocupados por distintas secciones de la página o poner de una manera sencilla un pie de foto a una imagen.

Puede que en un principio nos resulte un poco complicado trabajar con estas estructuras pero, si deseamos crear una página de calidad, tarde o temprano tendremos que vérnoslas con ellas y nos daremos cuenta de las posibilidades nos ofrecen.

Para empezar, nada más sencillo que por el principio: las tablas son definidas por las etiquetas <table> y </table>.

Dentro de estas dos etiquetas colocaremos todas las otras etiquetas, textos e imágenes que darán forma y contenido a la tabla.

Las tablas son descritas por líneas de izquierda a derecha. Cada una de estas líneas es definida por otra etiqueta y su cierre: <tr> y </tr>

Asimismo, dentro de cada línea, habrá diferentes celdas. Cada una de estas celdas será definida por otro par de etiquetas: <td> y </td>. Dentro de estas etiquetas será donde coloquemos nuestro contenido.

Aquí tenéis un ejemplo de estructura de tabla:

```
<table>
<tr>
<td>Celda 1, linea 1</td>
<td> Celda 2, linea 1</td>
</tr>
<tr>
<td> Celda 1, linea 2</td>
<td> Celda 2, linea 2</td>
</tr>
</table>
```

El resultado:

Celda 1, linea 1 Celda 2, linea 1
Celda 1, linea 2 Celda 2, linea 2

Nota: Hasta aquí hemos visto todas las etiquetas que necesitamos conocer para crear tablas. Existen otras etiquetas, pero lo que podemos conseguir con éllas se puede conseguir también usando las que hemos visto.

Por poner un ejemplo, señalamos la etiqueta <th>, que sirve para crear una celda cuyo contenido esté formatedo como un título o cabecera de la tabla. En la práctica, lo que hace es poner en negrita y centrado el contenido de esa celda, lo que se puede conseguir aplicando las correspondientes etiquetas dentro de la celda. Así:
<td align="center">contenido de la celda</td>.

A partir de esta idea simple y sencilla, las tablas adquieren otra magnitud cuando les incorporamos toda una batería de atributos aplicados sobre cada tipo de etiquetas que las componen. A lo largo de los siguientes capítulos nos adentraremos en el estudio de estos atributos de manera a proporcionaros los útiles indispensables para una buena puesta en página.

Tablas en HTML. Atributos para filas y celdas.

Hemos visto en el capitulo anterior que las tablas están compuestas de líneas que, a su vez, contienen celdas. Las celdas son delimitadas por las etiquetas <td> o por las etiquetas <th> (si queremos texto en negrita y centrado) y constituyen un entorno independiente del resto del documento. Esto quiere decir que:

- Podemos usar prácticamente cualquier tipo de etiqueta dentro de la etiqueta <td> para, de esta forma, dar forma a su contenido.
- Las etiquetas situadas en el interior de la celda no modifican el resto del documento.
- Las etiquetas de fuera de la celda no son tenidas en cuenta por ésta.

Así pues, podemos especificar el formato de nuestras celdas a partir de etiquetas introducidas en su interior o mediante atributos colocados dentro de la etiqueta de celda <td> o bien, en algunos casos, dentro de la etiqueta <tr>, si deseamos que el atributo sea valido para toda la línea. La forma más útil y actual de dar forma a las celdas es a partir de las hojas de estilo en cascada que ya tendréis la oportunidad de abordar más adelante.

Veamos a continuación algunos atributos útiles para la construcción de nuestras tablas. Empecemos viendo atributos que nos permiten modificar una celda en concreto o toda una línea:

align	Justifica el texto de la celda del mismo modo que si fuese el de un párrafo.
valign	Podemos elegir si queremos que el texto aparezca arriba (top), en el centro (middle) o abajo (bottom) de la celda.
bgcolor	Da color a la celda o línea elegida.
bordercolor	Define el color del borde.

Otros atributos que pueden ser únicamente asignados a una celda y no al conjunto de celdas de una línea son:

background	Nos permite colocar un fondo para la celda a partir de un enlace a una imagen.
height	Define la altura de la celda en pixels o porcentaje.
width	Define la anchura de la celda en pixels o porcentaje.
colspan	Expande una celda horizontalmente.
rowspan	Expande una celda verticalmente.

Nota: El atributo height no funciona en todos los navegadores, además, su uso no está muy extendido. Las celdas por lo general tienen el alto que ncesitan para que quepa todo el contenido que se le haya insertado, es decir, crecen lo suficiente para que quepa lo que hemos colocado dentro.

El atributo width si que funciona en todos los navegadores y lo tendréis que utilizar constantemente. Si le asignamos un ancho a la celda, el ancho será respetado y si dicha celda tiene mucho texto o cualquier otro contenido, la celda crecerá hacia abajo todo lo necesario para que quepa lo que hemos colocado.

Un matiz al último párafo. Se trata de que si definimos una celda de un ancho 100 por ejemplo, y colocamos en la celda un contenido como una imagen que mida más de 100 pixeles, la celda crecerá en horizontal todo lo necesario para que la imagen quepa. Si el elemento, aunque más ancho, fuera divisible (como un texto) el ancho sería respetado y el texto crecería hacia abajo o lo que es lo mismo, en altura, como señalábamos en el anterior párrafo.

Estos últimos cuatro atributos descritos son de gran utilidad. Concretamente, height y width nos ayudan a definir las dimensiones de nuestras celdas de una forma absoluta (en pixels o puntos de pantalla) o de una forma relativa, es decir por porcentajes referidos al tamaño total de la tabla. Podéis leer un articulo interesante a propósito de estas dos modalidades de diseño en nuestro manual de usabilidad.

A titulo de ejemplo:

<td width="80">
Dará una anchura de 80 pixels a la celda. Sin embargo,

<td width="80%">
Dará una anchura a la celda del 80% de la anchura de la tabla.

Hay que tener en cuenta que, definidas las dimensiones de las celdas, el navegador va a hacer lo que buenamente pueda para satisfacer al programador. Esto quiere decir que puede que en algunas ocasiones el resultado que obtengamos no sea el esperado. Concretamente, si el texto presenta una palabra excesivamente larga, puede que la anchura de la celda se vea aumentada para mantener la palabra en la misma línea. Por otra parte, si el texto resulta muy largo, la celda aumentara su altura para poder mostrar todo su contenido.

Análogamente, si por ejemplo definimos dos anchuras distintas a celdas de una misma columna, el navegador no sabrá a cual hacer caso. Es por ello que resulta conveniente tener bien claro desde un principio como es la tabla que queremos diseñar. No esta de más si la prediseñamos en papel si la complejidad es importante. El HTML resulta en general fácil pero las tablas pueden convertirse en un verdadero quebradero de cabeza si no llegamos a comprenderlas debidamente.

Los atributos rowspan y colspan son también utilizados frecuentemente. Gracias a ellos es posible expandir celdas fusionando éstas con sus vecinas. El valor que pueden tomar estas etiquetas es numérico. El número representa la cantidad de celdas fusionadas.

Así,
<td colspan="2">
Fusionara la celda en cuestión con su vecina derecha.

Esta celda tiene un colspan="2"	
Celda normal	Otra celda

Del mismo modo,
<td rowspan="2">

Esta celda tiene rowspan="2", por eso tiene fusionada la celda de abajo.	Celda Normal
	Otra celda normal

Expandirá la celda hacia abajo fusionándose con la celda inferior.

El resto de los atributos presentados presentan una utilidad y uso bastante obvios. Los dejamos a vuestra propia investigación.

Tablas en HTML. Atributos de la tabla y conclusión.

Además de los atributos específicos de cada celda o línea, las tablas pueden ser adicionalmente formateadas a partir de los atributos que nos ofrece la propia etiqueta <table>. He aquí aquellos que pueden parecernos en un principio importantes:

align	Alinea horizontalmente la tabla con respecto a su entorno.
background	Nos permite colocar un fondo para la tabla a partir de un enlace a una imagen.
bgcolor	Da color de fondo a la tabla.
border	Define el número de pixels del borde principal.
bordercolor	Define el color del borde.
cellpadding	Define, en pixels, el espacio entre los bordes de la celda y el contenido de la misma.
cellspacing	Define el espacio entre los bordes (en pixels).
height	Define la altura de la tabla en pixels o porcentaje.
width	Define la anchura de la tabla en pixels o porcentaje.

Los atributos que definen las dimensiones, height y width, funcionan de una manera análoga a la de las celdas tal y como hemos visto en el capitulo anterior. Contrariamente, el atributo align no nos permite justificar el texto de cada una de las celdas que componen la tabla, sino más bien, justificar la propia tabla con respecto a su entorno.

Vamos a poner tres ejemplos de alineado de tablas, centradas, alineadas a la derecha y a la izquierda.

Ejemplo de tabla centrada

Esta tabla está centrada (aling="center"). Solo tiene una celda.

Este sería un texto cualquiera colocado al lado de una tabla centrada

Ejemplo de tabla alineada a la derecha

Esta tabla está alineada a la derecha (aling="right"). Solo tiene una celda.

Para que se vea el efecto de alineado a la tabla debemos colocar un texto al lado y el texto rodeará la tabla, igual que ocurría con las imágenes alineadas a un lado.

Ejemplo de tabla alineada a la izquierda

Esta tabla está alineada a la izquierda (aling="left"). Solo tiene una celda.

Para que se vea el efecto de alineado a la tabla debemos colocar un texto al lado y el texto rodeará la tabla, igual que ocurría con las imágenes alineadas a un lado.

Los atributos cellpading y cellspacing nos ayudaran a dar a nuestra tabla un aspecto más estético. En un principio puede parecernos un poco confuso su uso pero un poco de practica será suficiente para hacerse con ellos.

En la siguiente imagen podemos ver gráficamente el significado de estos atributos.

Podéis comprobar vosotros mismos que los atributos definidos para una celda tienen prioridad con respecto a los definidos para una tabla. Podemos definir, por ejemplo, una tabla con color de fondo rojo y una de las celdas de color de fondo verde y se verá toda la tabla de color rojo menos la celda verde. Del mismo modo, podemos definir un color azul para los bordes de la tabla y hacer que una celda

particular sea mostrada con un borde rojo. (Aunque esto no funcionará en todos los navegadores debido a que algunos no reconocen el atributo bordercolor.

Tabla de color rojo de fondo	El atributo bgcolor de la tabla está en rojo.
Celda normal	Esta celda está en verde. tiene el atributo bgcolor en color verde

Tablas anidadas

Muy útil también es el uso de tablas anidadas. De la misma forma que podíamos incluir listas dentro de otras listas, las tablas pueden ser incluidas dentro de otras. Así, podemos incluir una tabla dentro de la celda de otra. El modo de funcionamiento sigue siendo el mismo aunque la situación puede complicarse si el número de tablas embebidas dentro de otras es elevado.

> **Consejo:** Páginas como DesarrolloWeb.com y muchas otras (La mayoría de las páginas avanzadas) que basan su diseño en tablas, realizan anidaciones de tablas constantemente para meter unos elementos de la página dentro de otros. Se pueden anidar tablas sin límite, sin embargo, en el caso de Netscape 4 hay que tener cuidado con el número de tablas que anidamos, porque a medida que metemos una tabla dentro de otra y otra dentro de esta y otra más, aumentando el grado de anidación sucesivamente... podemos encontrar problemas en su visualización y puede que la página tarde un poco de tiempo más en mostrarse en pantalla.

Vamos a ver un código de anidación de tablas. Veamos primero el resultado y luego el código, así conseguiremos entenderlo mejor.

Celda de la tabla principal	<table><tr><td>Tabla anidada, celda 1</td><td>Tabla anidada, celda 2</td></tr><tr><td>Tabla anidada, celda 3</td><td>Tabla anidada, celda 4</td></tr></table>

Este sería el código:

```
<table cellspacing="10" cellpadding="10" border="3">
<tr>
   <td align="center">
   Celda de la tabla principal
   </td>
   <td align="center">
      <table cellspacing="2" cellpadding="2" border="1">
      <tr>
         <td>Tabla anidada, celda 1</td>
         <td>Tabla anidada, celda 2</td>
      </tr>
      <tr>
         <td>Tabla anidada, celda 3</td>
         <td>Tabla anidada, celda 4</td>
      </tr>
      </table>
   </td>
</tr>
</table>
```

Ejemplos prácticos

Hasta aquí la información que pretendíamos transmitiros sobre las tablas en HTML. Sería importante ahora realizar algún ejemplo de realización de una tabla un poco compleja. Por ejemplo la siguiente:

Animales en peligro de extinción			
Nombre	Cabezas	Previsión 2010	Previsión 2020
Ballena	6000	4000	1500

Oso Pardo	50	0
Lince	10	
Tigre	300	210

Otro ejemplo de tabla con el que podemos practicar:

Climas de América del Sur			
Parte de arriba de América del Sur. Países como:	Venezuela Colombia Ecuador Perú	Parte de abajo de América del Sur. Países como:	Argentina Chile Uruguay Paraguay
Bosque tropical, clima de sabana, clima marítimo con inviernos secos.		Climas marítimos con veranos secos, con inviernos secos, climas frios, clima de estepa, clima desértico.	

Formularios HTML

Hasta ahora hemos visto la forma en la que el HTML gestiona y muestra la información, esencialmente mediante texto, imágenes y enlaces. Nos queda por ver de qué forma podemos intercambiar información con nuestro visitante. Desde luego, este nuevo aspecto resulta primordial para gran cantidad de acciones que se pueden llevar a cabo mediante el Web: Comprar un articulo, rellenar una encuesta, enviar un comentario al autor...

Hemos visto anteriormente que podíamos, mediante los enlaces, contactar directamente con un correo electrónico. Sin embargo, esta opción puede resultar en algunos casos poco versátil si lo que deseamos es que el navegante nos envíe una información bien precisa. Es por ello que el HTML propone otra solución mucho más amplia: Los formularios.

Los formularios son esas famosas cajas de texto y botones que podemos encontrar en muchas páginas web. Son muy utilizados para realizar búsquedas o bien para introducir datos personales por ejemplo en sitios de comercio electrónico. Los datos que el usuario introduce en estos campos son enviados al correo electrónico del administrador del formulario o bien a un programa que se encarga de procesarlo automáticamente.

Usando HTML podemos únicamente enviar el formulario a un correo electrónico. Si queremos procesar el formulario mediante un programa la cosa puede resultar un poco más compleja ya que tendremos que emplear otros lenguajes más sofisticados. En este caso, la solución más sencilla es utilizar los programás prediseñados que nos proponen un gran número de servidores de alojamiento y que nos permiten almacenar y procesar los datos en forma de archivos u otros formatos. Si vuestras páginas están alojadas en un servidor que no os propone este tipo de ventajas, siempre podéis recurrir a servidores de terceros que ofrecen este u otro tipo de servicios gratuitos para webs. Por supuesto, existe otra alternativa que es la de aprender lenguajes como ASP o PHP que nos permitirán, entre otras cosas, el tratamiento de formularios.

Los formularios son definidos por medio de las etiquetas <form> y </form>. Entre estas dos etiquetas colocaremos todos los campos y botones que componen el formulario. Dentro de esta etiqueta <form> debemos especificar algunos atributos:

action

Define el tipo de acción a llevar a cabo con el formulario. Como ya hemos dicho, existen dos posibilidades:

- El formulario es enviado a una dirección de correo electrónico
- El formulario es enviado a un programa o script que procesa su contenido

En el primer caso, el contenido del formulario es enviado a la dirección de correo electrónico especificada por medio de una sintaxis de este tipo:

<form action="mailto:direccion@correo.com" ...>

Si lo que queremos es que el formulario sea procesado por un programa, hemos de especificar la dirección del archivo que contiene dicho programa. La etiqueta quedaría en este caso de la siguiente forma:

<form action="dirección del archivo" ...>

La forma en la que se expresa la localización del archivo que contiene el programa es la misma que la vista para los enlaces.

method

Este atributo se encarga de especificar la forma en la que el formulario es enviado. Los dos valores posibles que puede tomar esta atributo son post y get. A efectos prácticos y, salvo que se os diga lo contrario, daremos siempre el valor post.

enctype

Se utiliza para indicar la forma en la que viajará la información que se mande por el formulario. En el caso más corriente, enviar el formulario por correo electrónico, el valor de este atributo debe de ser "text/plain". Así conseguimos que se envíe el contenido del formulario como texto plano dentro del email.

Si queremos que el formulario se procese automáticamente por un programa, generalmente no utilizaremos este atributo, de modo que tome su valor por defecto, es decir, no incluiremos enctype dentro de la etiqueta <form>

Ejemplo de etiqueta <form> completa

Así, para el caso más habitual -el envío del formulario por correo- la etiqueta de creación del formulario tendrá el siguiente aspecto:

<form action="mailto:direccion@correo.com (o nombre del archivo de proceso)" method="post" enctype="text/plain">

Entre esta etiqueta y su cierre colocaremos el resto de etiquetas que darán forma a nuestro formulario, las cuales serán vistas en capítulos siguientes.

Referencia: Mandar formulario por correo electrónico

Los formularios se utilizan habitualmente para implementar un tipo de contacto con el navegante, que consiste en que éste pueda mandarnos sus comentarios por correo electrónico a nuestro buzón.

Para este tipo de utilización de los formularios hemos publicado hace tiempo en DesarrolloWeb.com un artículo que puede resultar muy interesante para los que deseen un referencia extremadamente rápida para construir un formulario que envíe los datos por correo electrónico al desarrollador de la página.

Elementos de Formularios. Campos de texto

El HTML nos propone una gran diversidad de alternativas a la hora de crear nuestros formularios. Estas van desde la clásica caja de texto hasta la lista de opciones pasando por las cajas de validación.

Veamos en qué consiste cada una de estas modalidades y como podemos implementarlas en nuestro formulario.

Texto corto

Las cajas de texto son colocadas por medio de la etiqueta <input>. Dentro de esta etiqueta hemos de especificar el valor de dos atributos: **type** y **name**.

La etiqueta es de la siguiente forma:

```
<input type="text" name="nombre">
```

De este modo expresamos nuestro deseo de crear una caja de texto cuyo contenido será llamado nombre (por ejemplo). El aspecto de este tipo de cajas es de sobra conocido, aquí lo podéis ver:

El nombre del elemento del formulario es de gran importancia para poder identificarlo en nuestro programa de procesamiento o en el mail recibido. Por otra parte, es importante indicar el atributo type, ya que, como veremos, existen otras modalidades de formulario que usan esta misma etiqueta.

El empleo de estas cajas esta fundamentalmente destinado a la toma de datos breves: palabras o conjuntos de palabras de longitud relativamente corta. Veremos más adelante que existe otra forma de tomar textos más largos a partir de otra etiqueta.

Además de estos dos atributos, esenciales para el correcto funcionamiento de nuestra etiqueta, existen otra serie de atributos que pueden resultarnos de utilidad pero que no son imprescindibles:

size
Define el tamaño de la caja en número de caracteres. Si al escribir el usuario llega al final de la caja, el texto ira desfilando a medida que se escribe haciendo desaparecer la parte de texto que queda a la izquierda.

maxlength
Indica el tamaño máximo del texto que puede ser tomado por el formulario. Es importante no confundirlo con el atributo size. Mientras el primero define el tamaño aparente de la caja de texto, maxlength indica el tamaño máximo real del texto que se puede escribir. Podemos tener una caja de texto con un tamaño aparente (size) que es menor que el tamaño máximo (maxlength). Lo que ocurrirá en este caso es que, al escribir, el texto ira desfilando dentro de la caja hasta que lleguemos a su tamaño máximo definido por maxlength, momento en el cual nos será imposible continuar escribiendo.

value
En algunos casos puede resultarnos interesante asignar un valor definido al campo en cuestión. Esto puede ayudar al usuario a rellenar más rápidamente el formulario o darle alguna idea sobre la naturaleza de datos que se requieren. Este valor inicial del campo puede ser expresado mediante el atributo value. Veamos su efecto con un ejemplo sencillo:

```
<input type="text" name="nombre" value="Perico Palotes">
```

Genera un campo de este tipo:

Nota: estamos obligados a utilizar la etiqueta <form>

Aunque de lo que se lee en estos capítulos sobre formularios se puede entender bien esto, hemos querido remarcarlo para que quede muy claro: Cuando queremos utilizar en cualquer situación elementos de formulario debemos escribirlos siempre entre las etiquetas <form> y </form>. De lo contrario, los elementos se verán perfectamente en Explorer pero no en Netscape.

Dicho de otra forma, en Netscape no se visualizan los elementos de formulario a no ser que esten colocados entre las correspondientes etiquetas de inicio y fin de formulario.

Es por ello que para mostrar un campo de texto no vale con poner la etiqueta <input>, sino que habrá que ponerla dentro de un formulario. Así:

```
<form>
<input type="text" name="nombre" value="Perico Palotes">
```

```
</form>
```

Veremos posteriormente que este atributo puede resultar relevante en determinadas situaciones.

Texto oculto

Podemos esconder el texto escrito por medio asteriscos de manera a aportar una cierta confidencialidad. Este tipo de campos son análogos a los de texto con una sola diferencia: remplazamos el atributo type="text" por type="password":

```
<input type="password" name="nombre">
```

En este caso, podéis comprobar que al escribir dentro del campo en lugar de texto veréis asteriscos.

Estos campos son ideales para la introducción de datos confidenciales, principalmente códigos de acceso. Se ve en funcionamiento a continuación.

Texto largo

Si deseamos poner a la disposición de usuario un campo de texto donde pueda escribir cómodamente sobre un espacio compuesto de varias líneas, hemos de invocar una nueva etiqueta: <textarea> y su cierre correspondiente.

Este tipo de campos son prácticos cuando el contenido a enviar no es un nombre teléfono o cualquier otro dato breve, sino más bien, un comentario, opinión, etc.

Dentro de la etiqueta textarea deberemos indicar, como para el caso visto anteriormente, el atributo name para asociar el contenido a un nombre que será asemejado a una variable en los programás de proceso. Además, podemos definir las dimensiones del campo a partir de los atributos siguientes:

rows
Define el número de líneas del campo de texto.

cols
Define el número de columnas del campo de texto.

La etiqueta queda por tanto de esta forma:

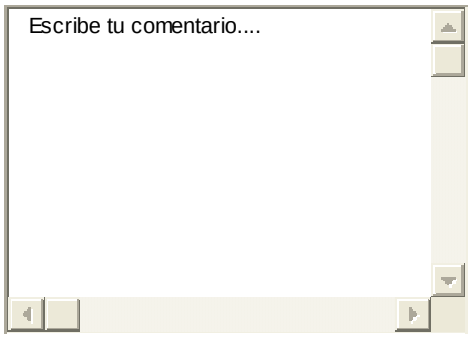
```
<textarea name="comentario" rows="10" cols="40"></textarea>
```

El resultado es el siguiente:

Asimismo, es posible predefinir el contenido del campo. Para ello, no usaremos el atributo value sino que escribiremos dentro de la etiqueta el contenido que deseamos atribuirle. Veámoslo:

```
<textarea name="comentario" rows="10" cols="40">Escribe tu comentario....</textarea>
```

Dará como resultado:



Otros elementos de formulario

Efectivamente, los textos son un manera muy practica de hacernos llegar la información del navegante. No obstante, en muchos casos, los textos son difícilmente adaptables a programás que puedan procesarlos debidamente o bien, puede que su contenido no se ajuste al tipo de información que requerimos. Es por ello que, en determinados casos, puede resultar más efectivo proponer una elección al navegante a partir del planteamiento de una serie de opciones.

Este es el caso de, por ejemplo, ofrecer una lista de países, el tipo de tarjeta de crédito para un pago,...

Este tipo de opciones pueden ser expresadas de diferentes formás. Veamos a continuación cuales son:

Listas de opciones

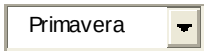
Las listas de opciones son ese tipo de menús desplegables que nos permiten elegir una (o varias) de las múltiples opciones que nos proponen. Para construirlas emplearemos una etiqueta con su respectivo cierre: <select>

Como para los casos ya vistos, dentro de esta etiqueta definiremos su nombre por medio del atributo name. Cada opción será incluida en una línea precedida de la etiqueta <option>.

Podemos ver, a partir de estas directivas, la forma más típica y sencilla de esta etiqueta:

```
<select name="estacion">
<option>Primavera</option>
<option>Verano</option>
<option>Otoño</option>
<option>Invierno</option>
</select>
```

El resultado es:



Esta estructura puede verse modificada principalmente a partir de otros dos atributos:

size
Indica el número de valores mostrados de la lista. El resto pueden ser vistos por medio de la barra lateral de desplazamiento.

multiple
Permite la selección de más varios elementos de la lista. La elección de más de un elemento se hace como con el explorador de Windows, a partir de las teclas ctrl o shift. Este atributo se expresa sin valor alguno, es decir, no se utiliza con el igual: simplemente se pone para conseguir el efecto, o no se pone si queremos una lista desplegable común.

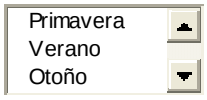
> **Consejo: Si es posible, no uses multiple**

No recomendamos especialmente la puesta en practica de esta opción ya que el manejo de las teclas ctrl o shift para elegir varias opciones puede ser desconocido para el navegante. Evidentemente, siempre cabe la posibilidad de explicarle como funciona aunque no dejara de ser una complicación para más para el visitante.

Veamos cual es el efecto producido por estos dos atributos cambiando la línea:
<select name="estacion">

por:
<select name="estacion" size="3" multiple>

La lista quedara de esta forma:



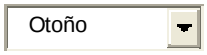
La etiqueta **<option>** puede asimismo ser matizada por medio de **otros atributos**

selected
Del mismo modo que multiple, este atributo no toma ningún valor sino que simplemente indica que la opción que lo presenta esta elegida por defecto.

Así, si cambiamos la línea del código anterior:
<option>Otoño</option>

por:
<option selected>Otoño</option>

El resultado será:



value
Define el valor de la opción que será enviado al programa o correo electrónico si el usuario elige esa opción. Este atributo puede resultar muy útil si el formulario es enviado a un programa puesto que a cada opción se le puede asociar un número o letra, lo cual es más fácilmente manipulable que una palabra o texto. podríamos así escribir líneas del tipo:

<option value="1">Primavera</option>

De este modo, si el usuario elige primavera, lo que le llegara al programa (o correo) es una variable llamada estacion que tendrá com valor 1. En el correo electrónico recibiríamos:

estacion=1

Botones de radio

Existe otra alternativa para plantear una elección, en este caso, obligamos al internauta a elegir únicamente una de las opciones que se le proponen.

La etiqueta empleada en este caso es <input> en la cual tendremos el atributo type ha de tomar el valor radio. Veamos un ejemplo:

```
<input type="radio" name="estacion" value="1">Primavera
<br>
<input type="radio" name="estacion" value="2">Verano
<br>
<input type="radio" name="estacion" value="3">Otoño
<br>
<input type="radio" name="estacion" value="4">Invierno
```

Nota: Hay que fijarse que la etiqueta <input type="radio"> sólo coloca la casilla pinchable en la página. Los textos que aparecen al lado, así como los saltos de línea los colocamos con el correspondiente texto en el código de la página y las etiquetas HTML que necesitemos.

El resultado es el siguiente:

 Primavera
 Verano
 Otoño
 Invierno

Como puede verse, a cada una de las opciones se le atribuye una etiqueta input dentro de la cual asignamos el mismo nombre (name) para todas las opciones y un valor (value) distinto. Si el usuario elige supuestamente Otoño, recibiremos en nuestro correo una línea tal que esta:

estacion=3

Cabe señalar que es posible preseleccionar por defecto una de las opciones. Esto puede ser conseguido por medio del atributo **checked**:

<input type="radio" name="estacion" value="2" checked>Verano

Veamos el efecto:

 Primavera
 Verano
 Otoño
 Invierno

Cajas de validación

Este tipo de elementos pueden ser activados o desactivados por el visitante por un simple clic sobre la caja en cuestión. La sintaxis utilizada es muy similar a las vistas anteriormente:

<input type="checkbox" name="paella">Me gusta la paella

El efecto:

 Me gusta la paella

La única diferencia fundamental es el valor adoptado por el atributo type.

Del mismo modo que para los botones de radio, podemos activar la caja por medio del atributo **checked**.

El tipo de información que llegara a nuestro correo (o al programa) será del tipo:

paella=on (u off dependiendo si ha sido activada o no)

Envio, borrado y demás en formularios HTML

Los formularios han de dar plaza no solamente a la información a tomar del usuario sino también a otra serie de funciones. Concretamente, han de permitirnos su envío mediante un botón. También puede resultar práctico poder proponer un botón de borrado o bien acompañarlo de datos ocultos que puedan ayudarnos en su procesamiento.

En este capitulo, para terminar la saga de formularios, daremos a conocer los medios de instalar todas estas funciones.

botón de envío

Para dar por finalizado el proceso de relleno del formulario y hacerlo llegar a su gestor, el navegante ha de validarlo por medio de un botón previsto a tal efecto. La construcción de dicho botón no reviste ninguna dificultad una vez familiarizados con las etiquetas input ya vistas:

```
<input type="submit" value="Enviar">
```

Con este código generamos un botón como este:

Enviar

Como puede verse, tan solo hemos de especificar que se trata de un botón de envío (type="submit") y hemos de definir el mensaje del botón por medio del atributo value.

botón de borrado

Este botón nos permitirá borrar el formulario por completo en el caso de que el usuario desee rehacerlo desde el principio. Su estructura sintáctica es análoga a la anterior:

```
<input type="reset" value="Borrar">
```

A diferencia del botón de envío, indispensable en cualquier formulario, el botón de borrado resulta meramente optativo y no es utilizado frecuentemente. Hay que tener cuidado de no ponerlo muy cerca del botón de envío y de distinguir claramente el uno del otro.

Datos ocultos

En algunos casos, aparte de los propios datos enviados por el usuario, puede resultar práctico enviar datos definidos por nosotros mismos que ayuden al programa en su procesamiento del formulario. Este tipo de datos, que no se muestran en la página pero si pueden ser detectados solicitando el código fuente, no son frecuentemente utilizados por páginas construidas en HTML, son más bien usados por páginas que emplean tecnologías de servidor. No os asustéis, veremos más adelante qué quiere decir esto. Tan solo queremos dar constancia de su existencia y de su modo creación. He aquí un ejemplo:

```
<input type=hidden name="sitio" value="www.desarrolloweb.com">
```

Esta etiqueta, incluida dentro de nuestro formulario, enviara un dato adicional al correo o programa encargado de la gestión del formulario. podríamos, a partir de este dato, dar a conocer al programa el origen del formulario o algún tipo de acción a llevar a cabo (una redirección por ejemplo).

Botones normales

Dentro de los formularios también podemos colocar botones normales, pulsables como cualquier otro botón. Igual que ocurre con los campos hidden, estos botones por si solos no tienen mucha utilidad pero podremos necesitarlos para realizar acciones en el futuro. Su sintaxis es la siguiente.

```
<input type=button value="Texto escrito en el botón">
```

Quedaría de esta manera:

El uso más frecuente de un botón es en la programación en el cliente. Utilizando lenguajes como Javascript podemos definir acciones a tomar cuando un visitante pulse el botón de una página web.

Ejemplo de formulario

Con este capitulo finalizamos nuestro tema de formularios. Pasemos ahora a ejemplificar todo lo aprendido a partir de la creación de un formulario que consulta el grado de satisfacción de los usuarios de una línea de autobuses ficticia. El formulario está construido para que envíe los datos por correo electrónico a un buzón determinado.

Vemos el formulario en esta página. Vosotros tratar de construirlo para ver si habéis entendido bien los temas sobre formularios.

Nombre

Email @

Población

Sexo

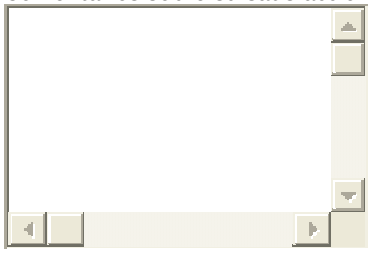
Hombre

Mujer

Frecuencia de los viajes

Varias veces al dia

Comentarios sobre su satisfacción personal



Deseo recibir notificación de las novedades en las líneas de autobuses.

Enviar formulario

Borrar todo

Recordad que podéis ver el código fuente de cualquier página web utilizando los menús de vuestro navegador, así podréis revisar el código que hemos utilizado para construir el formulario.

A continuación también mostraremos el código fuente de este formulario, que es importante que todos le echemos un vistazo, aunque sea rápidamente.

```
<form action="mailto:colabora@desarrolloweb.com" method="post" enctype="text/plain">
Nombre <input type="text" name="nombre" size="30" maxlength="100">
<br>
Email <input type="text" name="email" size="25" maxlength="100" value="@">
<br>
Población <input type="text" name="poblacion" size="20" maxlength="60">
<br>
Sexo
<br>
<input type="radio" name="sexo" value="Varon" checked> Hombre
<br>
<input type="radio" name="sexo" value="Hembra"> Mujer
<br>
<br>
Frecuencia de los viajes
<br>
<select name="utilizacion">
   <option value="1">Varias veces al dia
   <option value="2">Una vez al dia
   <option value="3">Varias veces a la semana
   <option value="4">varias veces al mes
</select>
<br>
<br>
Comentarios sobre su satisfacción personal
<br>
<textarea cols="30" rows="7" name="comentarios"></textarea>
```

```
<br>
<br>
<input type="checkbox" name="recibir_info" checked>
Deseo recibir notificación de las novedades en las líneas de autobuses.
<br>
<br>
<input type="submit" value="Enviar formulario">
<br>
<br>
<input type="Reset" value="Borrar todo">
</form>
```

Para acabar, vamos a ver lo que recibirían por correo electrónico en la empresa de autobuses cuando un usuario cualquiera rellenase este formulario y pulsase sobre el botón de envio.

```
nombre=Federico Mijo Silvestre
email=fede@terramix.com
poblacion=Astorga, León
sexo=Varon
utilizacion=2
comentarios=No creo que sea una buena linea. Poner más autobuses.
recibir_info=on
```

Mapas de imágenes con HTML

En capítulos anteriores hemos podido adentrarnos en el elemento básico de navegación del web: El enlace hipertexto. Hemos visto que estos enlaces son palabras, textos o imágenes que, al pinchar sobre ellos, nos envían a otras páginas o zonas.

Los mapas de imágenes es un nuevo planteamiento de navegación que incorpora una serie de enlaces dentro de una misma imagen. Estos enlaces son definidos por figuras geométricas y funcionan exactamente del mismo modo que los otros enlaces.

En un principio, estos mapas no eran directamente reconocidos por los navegadores y recurrían a tecnologías de lado del servidor para ser visualizados. Hoy en día pueden ser implementados por medio de código HTML tal y como veremos en este capitulo.

Podemos utilizar estos mapas, por ejemplo, en portadas donde damos a conocer cada una de las secciones del sitio por medio de una imagen. También puede ser muy práctico en mapas geográficos donde cada ciudad, provincia o punto cualquiera representa un enlace a una página.

En cualquier caso, el uso de estos mapas ha de estar sistemáticamente acompañado de un texto explicativo que dé a conocer al usuario la posibilidad de hacer clic sobre los distintos puntos de la imagen. Frases como "Haz clic sobre tal icono para acceder a tal información" resultan muy indicativas a la hora de hacer intuitiva la navegación por los mapas de imágenes. Por otro lado, no esta de más introducir esa misma explicación en el atributo alt de la imagen.

Así pues, un mapa de imagen esta compuesto de dos partes:

- La imagen propiamente dicha que estará situada como de costumbre dentro de la etiqueta <body> de nuestro documento HTML.
- Un código, situado en el interior de la etiqueta <map>, que delimitara por medio de líneas geométricas imaginarias cada una de las áreas de los enlaces presentados en la imagen.

Las líneas geométricas que delimitan los enlaces, es decir, las áreas de los enlaces, han de ser definidas por medio de coordenadas. Cada imagen es definida por unas dimensiones de ancho (X) y alto (Y) y cada punto de la imagen puede ser definido por tanto diciendo a que altura (x) y anchura (y) nos encontramos. De este modo, la esquina superior izquierda corresponde a la posición 0,0 y la esquina inferior derecha corresponde a las coordenadas X,Y. Si deseamos saber qué coordenadas corresponden a un punto concreto de nuestra imagen, lo mejor es utilizar un programa de diseño grafico como Photoshop o Paint Shop Pro.

La mejor forma de explicar el funcionamiento de este tipo de mapas es a partir de un ejemplo práctico. Supongamos que tenemos una imagen con un mapa como esta:

Dentro de ella queremos introducir un enlace a cada uno de los elementos que la componen. Para ello, definiremos nuestros enlaces como zonas circulares de pequeño tamaño que serán distribuidas a lo largo y ancho de la imagen.

Veamos a continuación el código que utilizaremos:

```
<table border=0 width=450><tr><td align="center">
<map name="mapa1">
<area alt="Pulsa para ver la página de mis amigos" shape="CIRCLE" coords="44,36,29" href="#">
<area alt="Pulsa para ver mi novia" shape="CIRCLE" coords="140,35,31" href="#">
<area alt="Pulsa para conocer a mi Familia" shape="circle" coords="239,37,30" href="#">
<area alt="Pulsa para conocer mi trabajo" shape="CIRCLE" coords="336,36,31" href="#">
</map>
<img src="images/html/mapa1.gif" width="380" height="72" alt="Mapa de imágenes. Pulsa en cada una de los círculos." border="0" usemap="#mapa1">
<br>
Pulsa en los círculos para acceder a las secciones!
</td></tr></table>
```

> **Nota: Los href de las áreas van a #**
> Este es un ejemplo parcial de utilización de los mapas, faltaría colocar los href con valores reales y no con la #. Cada uno de los enlaces de las áreas -atributo href de la etiqueta <area>- deberían llevar a una página web. El ejemplo quedaría completo si creasemos todas las páginas donde enlazar las áreas y colocasemos los href dirigidos hacia dichas páginas. Como no hemos hecho las páginas "destino" hemos colocado enlaces que no llevan a ningún sitio, que, como puedes ver, se indica con el caracter "#".

Podéis observar, tal y como hemos explicado antes, que nuestro mapa consta de dos partes principales: la imagen y la etiqueta <map> que define las áreas de cada enlace.

Cada área se indica con una etiqueta <area>, que tiene los siguientes atributos:

alt
Para indicar un texto que se mostrará cuando situemos el ratón en el área.

shape
Indica el tipo de área.

coords
Las coordenadas que definen el área. Serán un grupo de valores numéricos distintos dependiendo del tipo de área (shape) que estemos definiendo.

href
Para indicar el destino del enlace correspondiente al área.

En este caso hemos utilizado unas áreas circulares (shape="CIRCLE"), que se definen indicando el centro del círculo -una coordenada (X,Y) y el radio, que es un número entero que se corresponde con el número de pixels desde el centro hasta el borde del círculo.

Tipos de áreas: shape distintas.

Existen tres tipos de áreas distintas, suficientes para hacer casi cualquier tipo de figura. En el dibujo que acompaña estas líneas se puede ver una representación de las áreas, que detallamos a continuación.

shape="RECT"

Crea un área rectangular. Para definirla se utilizan las coordenadas de los puntos de la esquina superior izquierda y la esquina inferior derecha. Tal como están nombradas dichas coordenadas en nuestro dibujo, el área tendría la siguiente etiqueta:

```
<area shape="RECT" coords="X1,Y1,X2,Y2" href="#">
```

shape="CIRCLE"

Crea un área circular, que se indica con la coordenada del centro del círculo y el radio. A la vista de nuestro dibujo, la etiqueta de un área circular tendría esta forma:

```
<area shape="CIRCLE" coords="X1,Y1,R" href="#">
```

shape="POLY"

Este tipo de área, poligonal, es la más compleja de todas. Un polígono queda definido indicando todos sus puntos, pero atención, los tenemos que indicar en orden, siguiendo el camino marcado por el perímetro del polígono. A la vista del dibujo y los nombres que hemos dado a los puntos del polígono, la etiqueta <area> quedaría de esta forma.

<area shape="POLY" coords=" X1,Y1, X2,Y2, X3,Y3, X4,Y4" href="#">

Frames en HTML

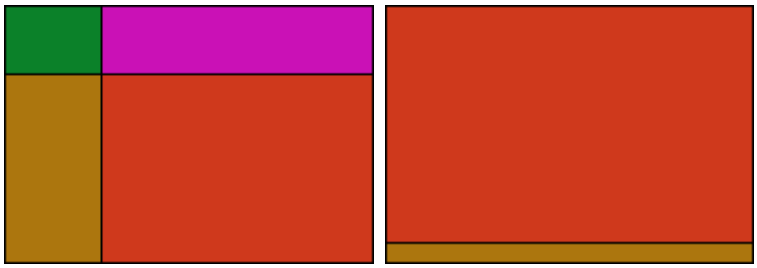
Una de las más modernas características de HTML son los frames, que se añadieron, tanto en Netscape Navigator como en Internet Explorer, a partir de sus versiónes 2.0. Los frames -que significan en castellano marcos- son una manera de partir la página en distintos espacios independientes los unos de los otros, de modo que en cada espacio se coloca una página distinta que se codifica en un fichero HTML distinto.

Al principio se crearon como etiquetas propietarias del navegador Netscape y rápidamente la potencia del recurso hizo que el uso de frames se extendiera por toda la web. Poco tardaría Internet Explorer en incluirlos, para que no se le escapase una novedad tan popular de su competidor. Finalmente, como respuesta a la popularidad entre los desarrolladores de los frames, el estándar HTML 4.0 incluyó estas etiquetas dentro de las permitidas.

Los frames, como decíamos, nos permiten partir la ventana del navegador en diferentes áreas. Cada una de estas áreas son independientes y han de ser codificadas con archivos HTML también independientes. Como resultado, cada frame o marco contiene las propiedades específicas que le indiquemos en el código HTML a presentar en ese espacio. Así mismo, y dado que cada marco es independiente, tendrán sus propias barras de desplazamiento, horizontales y verticales, por separado.

Existen en la web muchas páginas que contienen frames y seguro que todos hemos tenido la ocasión de conocer algunas. Se suelen utilizar para colocar en una parte de la ventana una barra de navegación, que generalmente se encuentra fija y permite el acceso a cualquier zona de la página web. Una de las principales ventajas de la programación con frames viene derivada de la independencia de los distintos frames, pues podemos navegar por los contenidos de nuestro web con la barra de navegación siempre visible, y sin que se tenga que recargar en cada una de las páginas que vamos visitando.

Un ejemplo de las áreas que se pueden construir en una construcción de frames se puede ver en las imágenes siguientes.



Frames - Explicación básica

Las páginas web que están hechas con frames se componen de una declaración de los marcos y tantas páginas en formato HTML corriente como distintas divisiones hemos definido. La declaración o definición de frames es la única página que realmente debemos aprender, puesto que las páginas que se van a visualizar en cada uno de los cuadros son ficheros HTML de los que venimos aprendiendo anteriormente en este manual.

Dicha definición está compuesta por etiquetas <FRAMESET> y <FRAME>, con las que se indicamos la disposición de todos los cuadros. La etiqueta <FRAMESET> indica las particiones de la ventana del navegador y la etiqueta <FRAME> indica cada uno de los cuadros donde colocaremos una página independiente.

Las particiones que se pueden hacer con un <FRAMESET> son en filas o columnas. Por ejemplo, podríamos indicar que deseamos hacer una división de la página en dos filas, o dos columnas, tres filas, etc. Para indicar tanto la forma de partir la ventana -en filas o columnas- como el número de particiones que pretendemos hacer, se ha de utilizar el atributo COLS o ROWS. El primero sirve para indicar una partición en columnas y el segundo para una partición en filas.

Nota: Es importante indicar que no se puede hacer una partición en filas y columnas a la vez, sino que debemos escoger en partir la ventana en una de las dos disposiciones. Más adelante indicaremos cómo partir la ventana tanto en filas como en columnas, que se hace con la anidación de frames.

En el atributo COLS o ROWS -sólo podemos elegir uno de los dos- colocamos entre comillas el número de particiones que deseamos realizar, indicando de paso el tamaño que va a asignarse a cada una. Un valor típico de estos atributos sería el siguiente:

cols="20%,80%"
Indica que se deben colocar dos columnas, la de la izquierda tendría un 20% del espacio total de la ventana y la de la derecha un 80%.

rows="15%,60%,25%"
Así indicamos que deseamos tres filas, la de arriba con un 15% del espacio total, la del medio con un espacio correspondiente al 60% del total y la de abajo con un 25%. En total suman el 100% del espacio de la ventana.

Además del porcentaje para indicar el espacio de cada una de las casillas, también podemos indicarlo en pixeles. De esta manera.

cols="200,600"
Para indicar que la columna de la izquierda debe tener 200 pixels de ancho y la de la derecha 600. Esto está bien si nuestra ventana tiene 800 pixels de ancho, pero esto no tiene porque ser así en todos los monitores de los usuarios, por lo que este modo de expresar los marcos es importante que se indique de la siguiente manera.

cols="200,*"
Así indicamos que la primera columna ha de medir 200 pixels y que el resto del espacio disponible -que será mayor o menor dependiendo de la definición de la pantalla del usuario- se le asignará a segunda columna.

En la práctica podemos mezclar todos estos métodos para definir los marcos de la manera que deseemos, con pocentaje, con pixels o con el comodín (*). No importa cómo se definan, la única recomendación es que uno de los valores que indiquemos sea un asterisco, para que el área correspondiente a dicho asterisco o comodín sea más o menos grande dependiendo del espacio que tenga la ventana de nuestro navegador. Otros métodos de definir filas y columnas, atendiendo a este consejo, serían los siguientes:

rows="100,*,12%"
Definimos tres filas, la primera con 100 pixels de ancho, la segunda con el espacio que sobre de las otras dos, y la tercera con un 12% del espacio total.

cols="10%,50%,120,*"
Estamos indicando cuatro columnas. La primera del 10% del espacio de la ventana, la segunda con la mitad justa de la ventana, la tercera con un espacio de 120 pixels y la última con la cantidad de espacio que sobre al asignar espacio a las demás particiones.

Una vez hemos indicado el número de filas o columnas y el espacio reservado a cada una con la etiqueta <FRAMESET>, debemos especificar con la etiqueta <FRAME> la procedencia de cada uno de los frames en los que hemos partido la ventana.

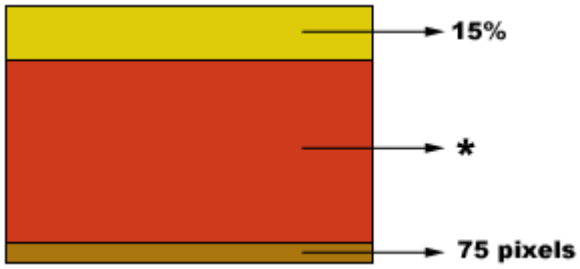
Para ello, disponemos del atributo SRC, que se ha de definir para cada una de las filas o columnas. De esta manera.

```
<FRAME src="marco1.html">
```

Así queda indicado que el frame que estamos definiendo debe mostrar la página marco1.html en su interior.

Frames - Creación de una estructura simple

Para ilustrar todo lo que venimos explicando podemos ver el ejemplo sobre cómo se crearía la definición de frames de la imagen que podemos ver a continuación.




```
<html>
<head>
   <title>Definición de Frames</title>
</head>
<frameset rows="15%,*,75">
   <frame src="pagina1.html">
   <frame src="pagina2.html">
   <frame src="pagina3.html">
</frameset>
</html>
```

Además tenemos algunas consideraciones que hacer para terminar de comprender este ejemplo:

- El título de la definición de frames es el que hereda toda la página web, por ello, no es buena idea titular como "definición de frames" por ejemplo, ya que entonces toda nuestra página se titularía así y seguramente no sea muy descriptivo. Si estuviésemos haciendo una página para la carnicería pepe sería mejor titular a la definición de frames algo como "Carnicería Pepe, las mejores carnes en Madrid".
- La página que define los frames no tiene body. HTML puede arrojarnos un error si lo incluimos.
- Las páginas "pagina1.html", "pagina2.html"y "pagina3.html" han de escribirse en archivos independientes con el nombre indicado. En este ejemplo, dichas páginas deberían encontrarse en el mismo directorio que la declaración de frames. Si especificamos una ruta para acceder al archivo podemos colocarlo en el directorio que deseemos.
- Los colores de cada uno de los frames los hemos colocado con el atributo bgcolor colocado en la etiqueta <BODY> de cada una de las páginas que se muestran en los marcos.

Frames - Una página en cada marco

Las páginas que mostraremos en cada marco son documentos HTML iguales a los que venimos creando anteriormente. Podemos colocar cualquier elemento HTML de los estudiados en este manual, como etiquetas de párrafo, imágenes, colores de fondo, etc.

Cada documento, como ya hemos indicado, se escribe por separado en su propio archivo HTML. Para el ejemplo del capítulo anterior podemos definir los archivos HTML de la siguiente manera.

pagina1.html
Es la página que contiene el titular de la web. Simplemente se trata de una etiqueta <H1> de titular. La página tiene su propio título, con la etiqueta <TITLE>, que no se podrá visualizar por ningún sitio a no ser que se muestre esta página sin los frames, ya que las páginas dentro de los marcos heredan el título de la definición de los frames.

```
<html>
<head>
   <title>Titulo Carnicería Pepe</title>
</head>

<body bgcolor="#DECC09">
<h1 align=center>Carnicería PEPE</h1>
</body>
</html>
```

pagina2.html
Es la página que se presentará en el área principal de la definición de frames, es decir, la página que tiene más espacio para visualizarse y donde pondremos los contenidos de la web. En este caso muestra un mensaje de bienvenida a la web, que hará las veces de portada.

```
<html>
<head>
   <title>Portada de Carnicería PEPE</title>
</head>

<body bgcolor="#CF391C" text="#ffffff">
<h1 align="center">Bienvenidos a nuestra web</h1>
<br>
<br>
La carnicería PEPE, con más de 100 años de experiencia, es la mejor fuente de carnes de vacuno y cerdo de la comunidad.
<br>
<br>
Tanto en invierno como en verano puede encontrar nuestras ofertas de temporada de primera calidad.
</body>
</html>
```

pagina3.html
En esta página se mostrará la barra de navegación por los contenidos del sitio. Contiene enlaces que deberían actualizar el contenido del área principal de la declaración de frames, para mostrar los distintos contenidos del sitio, por ejemplo, la portada, los productos, la página de contacto, etc.

```
<html>
<head>
   <title>Barra de navegación de carnicería PEPE</title>
</head>

<body bgcolor="#AC760E" link="ffffcc" vlink="ffffcc">
<div align="center">
<b>
<a href="pagina2.html">Portada</a> |
<a href="productos.html">Productos</a> |
<a href="contacto.html">Contacto</a>
</b>
</div>
</body>
</html>
```

Frames - Dirigir los enlaces

La única particularidad destacable en el ejemplo del capítulo anterior, y en el manejo de frames en general, se trata de que **cada uno de los enlaces que colocamos en las páginas actualizan el frame donde está colocado este enlace**. Por ejemplo, si tenemos enlaces en la parte inferior de la ventana, en el espacio correspondiente al tercer marco, actualizarán los contenidos del tercer frame, que es donde están situados los enlaces.

Lo lógico es que al pulsar sobre un enlace de la barra de navegación actualicemos el frame principal, que es donde habíamos planeado colocar los contenidos, en lugar del frame donde colocamos la barra de navegación, que debería mantenerse fija. Para conseguir este efecto debemos hacer un par de cosas:

1. **Darle un nombre al frame que deseamos actualizar**
   Dicho nombre se indica en la etiqueta <FRAME> de la definición de frames. Para ello utilizamos el atributo name, igualado al nombre que le queremos dar a dicho marco.
2. **Dirigir los enlaces hacia ese frame**
   Para ello debemos colocar en el atributo target de los enlaces -etiqueta <A>- el nombre del frame que deseamos actualizar al pulsar el enlace.

Después de darle un nombre al frame principal, nuestra declaración de frames quedaría de la siguiente manera.

```
<frameset rows="15%,*,75">
   <frame src="pagina1.html">
   <frame src="pagina2.html" name="principal">
   <frame src="pagina3.html">
</frameset>
```

Además, deberíamos colocar el atributo target a los enlaces, tal como sigue.

```
<a href="pagina2.html" target="principal">Portada</a> |
```

```
<a href="productos.html" target="principal">Productos</a> |
<a href="contacto.html" target="principal">Contacto</a>
```

Una vez realizados este par de cambios podemos ver como los enlaces de la barra de navegación sí actualizan la página que deben.

Valores para el atributo target

Como hemos visto, con el atributo target de la etiqueta <A> podemos indicar el nombre del frame que deseamos que actualice ese enlace. Sin embargo, no es este el único valor que podemos aplicarle al atributo. Tenemos algunos valores adicionales que podemos asignar a cualquier enlace en general.

_blank
Para hacer que ese enlace se abra en una ventana a parte. Nuestros ejemplos en este manual se suelen abrir en una ventana a parte, colocando este valor en el target de los enlaces que llevan a los ejemplos.

_self
Se actualiza el frame donde está situado el enlace. Es el valor por defecto.

_parent
El enlace se actualiza sobre su padre o sobre la ventana que estamos trabajando, si es que no hay un padre.

_top
La página se carga a pantalla completa, es decir, eliminando todos los frames que pudiera haber. Este atributo es muy importante porque si colocamos en nuestra página con frames un enlace a una página externa, se abriría en uno de los frames y se mantendrían visibles otros frames de la página, haciendo un efecto que suele ser poco agradable, porque parece que están evitando que nos escapemos.

La sintaxis de uno de estos valores de atributos colocados en un enlace sería la siguiente.

```
<A href="http://www.guiarte.com" target="_top">Acceder a guiarte.com</A>
```

Frames - Anidar frames

Para crear estructuras de marcos en las que se mezclen las filas y las columnas debemos anidar etiquetas <FRAMESET>. Empezando por la partición de frames más general, debemos colocar dentro las particiones de frames más pequeñas. La manera de indicar esto se puede ver fácilmente con un ejemplo.

En la imagen se puede ver el resultado final acompañada de la representación sobre la manera de definirlos. En primer lugar definimos una estructura de frames en dos columnas y dentro de la primera columna colocamos otra parción de frames en dos filas. El código necesario es el siguiente.

```
<frameset cols="200,*">
   <frameset rows="170,*">
     <frame src="pagina1.html">
     <frame src="pagina2.html">
   </frameset>
   <frame src="pagina3.html">
</frameset>
```

> **Nota:** hemos colocado un margen en cada una de las líneas de esta definición de frames para conseguir un código más entendible visualmente. Estos márgenes no son en absoluto necesarios, simplemente nos sirven para ver en qué nivel de anidación nos encontramos.

El ejemplo anterior se puede complicar un poco más si incluimos más particiones. Vamos a ver algo un poco más complicado para practicar más con las anidaciones de frames.

En la imagen se observa que el primer frameset a definir se compone de dos filas. Posteriormente, dentro de la segunda fila del primer frameset, tenemos otra partición en dos columnas, dentro de las que colocamos un tercer nivel de frameset con una defincón en filas en los dos casos. El código se puede ver a continuación.

```
<frameset rows="60,*">
   <frame src="pagina1.html">
   <frameset cols="200,*">
     <frameset rows="*,150">
       <frame src="pagina2.html">
       <frame src="pagina3.html">
     </frameset>
     <frameset rows="*,60">
       <frame src="pagina4.html">
       <frame src="pagina5.html">
     </frameset>
   </frameset>
</frameset>
```

Hasta aquí hemos visto la parte más básica de la creación de frames. En los siguientes capítulos podremos aprender a configurar los marcos para variar su apariencia y, entre otras cosas, eliminar las barras que separan cada uno de los distintos frames.

Frames - Atributos avanzados

Aparte de la creación de los marcos propiamente dicha, existen muchos atributos con los que configurar su apariencia. Para ello, tanto la etiqueta <frameset> como <frame> admiten diversos atributos que permiten especificar la forma de elementos como los bordes de los frames, el margen, la existencia o no de barras de desplazamiento, etc.

Atributos para la etiqueta <frameset>

Ya hemos conocido el atributo cols y rows, que sirven para indicar si la distribución en marcos se hará horizontalmente o verticalmente. Sólo se puede utilizar uno de ellos y se iguala a las dimensiones de cada uno de las divisiones, separadas por comas.

border="número de pixels"
Permite especificar de manera global para todo el frameset el número de pixels que ha de tener el borde de los frames.

bordercolor="#rrggbb"
Con este atributo podemos modificar el color del borde de los frames, también de manera global a todo el frameset.

frameborder="yes|no|0"
Sirve para mostrar o no el borde del frame. Sus posibles valores son "yes" (para que se vean los bordes) y "no" o "0" (para que no se vean). En la práctica elimina el borde, pero permanece una línea de separación de los frames.

framespacing="número de pixels"
Para determinar la anchura de la línea de separación de los frames. Se puede utilizar en Internet Explorer y junto con el atributo frameborder="0" sirve para eliminar los bordes de los marcos.

Atributos para la etiqueta <frame>

Para esta etiqueta hemos señalado en capítulos anteriores los atributos src, que sirve para indicar el archivo que contiene el marco y name, para darle un nombre al marco y luego dirigir los enlaces hacia el. Veamos ahora otros atributos disponibles.

marginwidth="número de pixels"
Define el número de pixels que tiene el margen del frame donde se indica. Este margen se aplica a la página que pretendemos ver en ese marco, de modo que si colocamos 0, los contenidos del página en ese marco estarán pegados por completo al borde del margen y si indicamos un valor de 10, los contenidos de la página estarían separados del borde 10 pixels.

marginheight="número de pixels"
Lo mismo que el anterior atributo, pero para el margen vertical.

scrolling="yes|no|auto"
Sirve para indicar si queremos que haya barras de desplazamiento en los distintos marcos. Si indicamos "yes" siempre saldrán las barras, si indicamos "no" no saldrán nunca y si colocamos "auto" saldrán sólo si son necesarias. Auto es el valor por defecto.

Consejo: hay que tener cuidado si eliminamos los bordes de los frames, puesto que la página web puede tener dimensiones distintas dependiendo de la definición de pantalla del visitante. Si el espacio de la ventana se ve reducido, podría verse reducido el espacio para el frame y puede que no quepan los elementos que antes si que cabían y si hemos eliminado las barras de desplazamiento puede que el visitante no pueda ver todo el contenido del marco.

Este mismo consejo se puede aplicar al redimensionamiento de frames, que veremos en el siguiente atributo. Si hacemos que los marcos no sean redimensionables probablemente tengamos una declaración de frames demasiado rígida, que puede verse mal en algún tipo de pantalla.

noresize
Este atributo no tiene valores, simplemente se pone o no se pone. En caso de que esté presente indica que el frame no se puede redimensionar. Como hemos podido ver, al colocar el ratón sobre el borde de los marcos sale un cursor que nos señala que podemos mover dicho borde y redimensionar así los frames. Por defecto, si no colocamos nada, los marcos si se pueden redimensionar.

frameborder="yes|no|0"
Este atributo permite controlar la aparición de los bordes de los frames. Con este atributo igualado a "0" o "no" los bordes se eliminan. Sin embargo, quedan los feos márgenes en el borde. Por lo que hemos podido comprobar funciona mejor en Netscape que en Internet Explorer. De todos modos, tenemos una nota un poco más adelante para explicar los frames sin bordes.

Nota: los atributos de frames no funcionan siempre bien en todos los navegadores. Es recomendable que hagamos un test sobre lo que estamos diseñando en varios navegadores para comprobar que nuestros frames se ven bien en todas las plataformas.

bordercolor="#rrggbb"
Permite especicicar el color del borde del marco.

Ventajas e inconvenientes del uso de frames

El diseño con frames es un asunto bastante controvertido, ya que distintos diseñadores tendrán unas u otras opiniones.

Referencia: Si deseas saber qué son los frames y cómo crearlos consulta los capítulos de Frames de nuestro manual de HTML.

En mi caso, pienso que es preferible no utilizarlos, aunque eso depende del tipo de sitio web que estés construyendo, ya que en algunos casos sí que sería muy adecuado su uso.

Voy a colocar unas ventajas e inconvenientes del uso de marcos (frames). Siempre es a mi entender, otros pueden tener otras opiniones.

Ventajas de usar frames

- La navegación de la página será más rápida. Aunque la primera carga de la página sería igual, en sucesivas impresiones de páginas ya tendremos algunos marcos guardados , que no tendrían que volverse a descargar.
- Crear páginas del sitio sería más rápido. Como no tenemos que incluir partes de código como la barra de navegación, título, etc. crear nuevas páginas sería un proceso mucho más rápido.
- Partes de la página (como la barra de navegación) se mantienen fijas y eso puede ser bueno, para que el usuario no las pierda nunca de vista.
- Estas mismas partes visibles constantemente, si contienen enlaces, pueden servir muy bien para mejorar la navegación por el sitio.
- Mantienen una identidad del sitio donde se navega, pues los elementos fijos conservan la imagen siempre visible.

Inconvenientes de usar frames

- Quitan espacio en la pantalla. El espacio ocupado por los frames fijos se pierde a la hora de hacer páginas nuevas, porque ya está utilizado. En definiciones de pantalla pequeña o dispositivos como Palms, este problema se hace más patente.

- Fuerzan al visitante a entrar por la declaración de frames. Si no lo hacen así, sólo se vería una página interior sin los recudros. Estos recuadros podrían ser insuficientes para una buena navegación por los contenidos y podrían no conservar una buena imagen corporativa.
- La promoción de la página sería, en principio, más limitada. Esto es debido a que sólo se debería promocionar la portada, pues si se promocionan páginas interiores, podría darse en caso de que los visitantes entrasen por ellas en lugar de por la portada, creandose el problema descrito en el punto anterior.
- A mucha gente les disgustan pues no se sienten libres en la navegación, pues entienden que esas partes fijas están limitando su movilidad por la web. Este efecto se hace más patente si la página con frames tiene enlaces a otras páginas web fuera del sitio y, al pulsar un enlace, se muestra la página nueva con los marcos de la página que tiene frames.
- Algunos navegadores no los soportan. Esto no es muy habitual, pero si estamos haciendo una página que queramos que sea totalmente accesible deberíamos considerarlo importante.
- Los bookmarks o favoritos no funcionan correctamente en muchos casos. Si queremos incluir un favorito a una página de un frame que no sea la portada podemos encontrar problemas.
- Puede que el botón de atrás del navegador no se comporte como deseamos.
- Si quieres actualizar más de un frame con la pulsación de un enlace deberás utilizar Javascript. Además los scripts se pueden complicar bastante cuando se tienen que comunicar varios frames entre si.

Conclusión

El trabajo con frames puede ser más bueno o más malo dependiendo de las características de la página a desarrollar, es tu tarea saber si en tu caso debes utilizarlos o no.

Las nuevas etiquetas de HTML 4.0

Introducción. Cuando Internet empezaba su imparable escalada, la versión del estándar HTML que circulaba era la 2.0, el cuál siguen soportando los navegadores más actuales. Pero las herramientas de que se disponía no ofrecían un control preciso de los documentos.

Pero como por aquel entonces el objetivo de Internet estaba fundamentalmente orientado al ámbito académico y no al de diseño, no se le dio demasiada importancia a la cuestión de lanzar una versión mejorada del estándar hasta que Netscape, que por aquel entonces era la empresa líder en el sector, tomó la iniciativa de incluir nuevas etiquetas pensadas para mejorar el aspecto visual de las páginas web.

Por este motivo el IETF (Internet Engineering Task Force) http://www.ietf.cnri.reston.va.us/, o lo que es lo mismo, Grupo de Trabajo en Ingeniería de Internet, comenzó a elaborar nuevos estándares, los cuales dieron como fruto el HTML 3.0, que resultó ser demasiado grande para las infraestructuras que había en ese momento, lo cual dificultó su aceptación.

Así pues, una serie de compañías (entre las que estaban Netscape, Sun Microsystems o Microsoft, entre otras), se unieron para crear lo que hoy se denomina W3C (o lo que es lo mismo, Consorcio para la World Wide Web), que fue fundado en octubre de 1.994 para conducir a la World Wide Web a su máximo potencial, desarrollando protocolos de uso común, para normalizar el uso de la web en todo el mundo.

El compromiso del W3C de encaminar a la Web a su máximo potencial incluye promover un alto grado de accesibilidad para las personas con discapacidades. El grupo de trabajo permanente Web Accessibility Initiative (WAI, Iniciativa para la Accesibilidad de la Red), en coordinación con organizaciones alrededor de todo el mundo, persigue la accesibilidad de la Web a través de cinco áreas de trabajo principales: Tecnología, directrices, herramientas, formación, difusión, e investigación y desarrollo.

De esta iniciativa nació el borrador de HTML 3.2 y en su versión definitiva se introdujeron cambios esenciales para las posibilidades que empezaban a ofrecer los navegadores, estas inclusiones fueron las tablas, los applets, etc.

En julio de 1.997 nace el borrador del HTML 4.0 y finalmente se aprueba en diciembre de 1.997 este estándar incluía como mejoras los marcos (frames), las hojas de estilo y la inclusión de scripts en páginas web, entre otras cosas.

Las nuevas etiquetas de HTML 4.0 (1)

Entre el estándar del HTML 3.2 al 4.0 se introdujeron ocho nuevas etiquetas de las cuales daremos una breve explicación.

<Q>... </Q>

Las etiquetas <Q> y </Q> actúan de forma muy parecida a <BLOCKQUOTE> pero con la particularidad de que añade un sangrado en párrafos más pequeños y sin necesidad de romper el párrafo.

Según el W3C, la etiqueta <BLOCKQUOTE> es para añadir sangrados largos y <Q>, para sangrados más pequeños, sin necesidad de romper el párrafo.

> Nota: En el HTML 4.0 es imprescindible poner la etiqueta de apertura y la de clausura <Q>.... </Q>.

<ACRONYM>... </ACRONYM>

Las etiquetas <ACRONYM>... </ACRONYM>, indican que hay un acrónimo en el texto. Un acrónimo es un pequeño texto que ayuda a explicar la estructura del texto una frase.

<INS>... </INS> y **<DEL>... </DEL>**

Utilice < INS>...</INS> para marcar las partes de un documento que se han agregado desde la versión pasada del documento. <DEL>... </ DEL> marca de manera similar un texto de un documento que se ha suprimido desde la versión anterior.

<COLGROUP>... </COLGROUP>

Se utiliza para tener un mejor control sobre un el formato de las tablas especificando las características que comparten como: anchura, altura y alineación.
Cada tabla debe tener por lo menos un <COLGROUP>; sin especificar ninguna característica de < COLGROUP >. HTML 4.0 asume que una tabla contiene un solo grupo de columnas y que este contiene todas las columnas de una tabla.
Por ejemplo, esto nos serviría para crear una tabla con una celda en la que puede incluirse una descripción y después seguido de check boxes para seleccionar las opciones deseadas.

Código: <TABLE> <COLGROUP span="10" width="30"> <COLGROUP span="1" width="0*"> <THEAD> <TR>... </ TABLE>

De esta forma, <COLGROUP> proporciona un formato más agradable a los check boxes sin necesidad de especificar, propiedades identicas para cada fila.

La etiqueta de inicio < COLGROUP >, requiere otra de cierre.
Con el que obtenemos: (en Nestcape sólo se verá la tabla, no el botón).

Las nuevas etiquetas de HTML 4.0 (2)

<FIELDSET>... </FIELDSET>

Hasta ahora, no disponíamos de ninguna manera de agrupar visualmente varios controles, si no echábamos mano de elementos que no son del formulario, como tablas o imagenes.
Ahora, si encerramos una parte de un formulario dentro de la etiqueta FIELDSET se mostrara un rectángulo alrededor de los mismos. Además, podemos indicar un título por medio de la etiqueta LEGEND, que admite el parámetro align="left / center / right / top / bottom", lo que nos permite alinear el título horizontal y verticalmente. La única pega es que deberemos introducir el conjunto en una celda de tabla con un ancho determinado, ya que si no lo hacemos así el recuadro abarcara todo el ancho de pantalla disponible.

Ejemplo.- (Sólo para I. Explorer)

```
<form action="cgi-bin/control.exe" method="post" enctype="text/plain" name="miform">
<table width="200">
<tr>
<td>
<fieldset>
<legend align="left"><font color="red">Caja de texto</font></legend>
pon tu nombre:
<input type="text" size="15">
</fieldset>
</td>
</tr>
<table>
</form>
```

<LABEL>... </LABEL>

Hasta no hace mucho los campos de entrada no estaban asociados a ellos mismos. Por ejemplo; a la hora de pulsar sobre un campo de confirmación, ¡ no sucedía nada! Pero ahora, sí lo pulsamos el control cambiará de estado.

Ejemplo:

```
<form action="cgi-bin/micontrol.exe" method="post" enctype="text/plain" name="un ejemplo más">
<label>
<input type="checkbox" name="email">
Le deseamos un feliz año nuevo
</label>
</form>
```

<BUTTON>... </BUTTON>

A partir de la implementación de los estándares HTML 4.0 contamos con varias etiquetas nuevas para construir formularios, siendo BUTTON una de ellas, bastante útil por cierto. La pega es que las versiones de 4 de Nestcape se lanzaron antes de estas implementaciones, por lo que estas nuevas etiquetas sólo se pueden visualizar correctamente con Internet Explorer 4 y superiores.
Esta etiqueta proporciona un método único para la implementación de cualquier tipo de botón de formulario. Sus principales atributos son:

- type= " tipo ", que puede tomar los ya conocidos valores submit (por defecto), reset y button.
- name= " nombre ", que asigna un nombre identificador único al botón.
- value= " texto ", que define el texto que va a aparecer en el botón.

La principal ventaja que aporta estas etiquetas es que ahora vamos a poder introducir dentro de ellas cualquier elemento de HTML, como imagenes y tablas.

Ejemplos.

```
<form action="cgi-bin/control.exe" method="post" enctype="text/plain" name="miform">

<button name="boton_1" type="button">
<table width="10" cellspacing="0" cellpadding="2" border="1">
<tr>
<td>uno</td>
<td>dos</td>
</tr>
<tr>
<td>tres</td>
<td>cuatro</td>
</tr>
</table>
</button>
</form>
```

Sonido en HTML I, introducción

En su corta pero rápida vida, las páginas web han pasado a ser no ya unos meros documentos textuales a los que se puede acceder por Internet, sino unas verdaderas presentaciones multimedia, que combinan textos con imágenes, sonidos, videos y elementos de realidad virtual.

Si el primer paso que se dio fue añadir imágenes a las páginas web, tanto estáticas como dinámicas GIF animados), el siguiente paso consistió en introducir sonidos en las mismas, consiguiendo con ellos el apelativo de “multimedia”. Y nos referiremos en lo sucesivo cuando hablemos de sonido tanto a sonido sintetizado como a verdaderas grabaciones de audio, de calidad muy elevada.

Ahora bien, aunque los navegadores han sido capaces de interpretar los ficheros de sonido adecuados desde hace ya algunas versiones, es cierto que la aplicación de sonidos a las páginas web ha estado limitada desde siempre por el ancho de banda necesario en las conexiones a Internet para poder descargar de forma adecuada dichos ficheros, debido al tamaño “excesivo” de los mismos.

Otra de las limitaciones importantes que encontramos a la hora de incluir ficheros de sonido en nuestras páginas es la diferentes implementación que hacen de ellos los navegadores web más usados. En efecto,

no sólo deberemos usar etiquetas HTML distintas para Internet Explorer que para Netscape Navigator, sino que a veces la forma misma de interpretar el sonido puede diferir de uno a otro navegador.

Por último, hay que destacar que a la hora de incluir ficheros de audio en nuestras páginas debemos ser conscientes que muchos de los formatos usados, sobre todo en grabaciones de calidad, precisan un plugin o programa especial para su reproducción en el navegador cliente. Y si es cierto que actualmente hay ciertos plugins se han transformado casi en un estándar en Internet (como el de Real Audio o el de MP3), hay otros posibles que no es normal tener instalados, por lo que si incluimos ficheros de esos tipos obligaremos al usuario a tener que instalarlos, cosa a la que suele ser reacio.

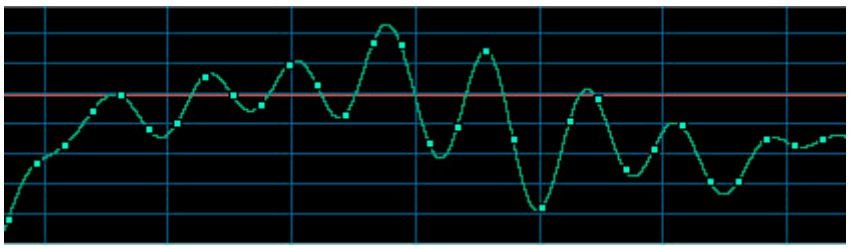
Sonido en HTML II, características del sonido digital

Vamos a estudiar algunos de los conceptos básicos del sonido digital, aunque sin entrar en demasiadas consideraciones técnicas. Para aquellos que deseen más información, existen multitud de sitios web que estudian específicamente el sonido digital y el hardware necesario para su captura y reproducción.

El sonido tiene una naturaleza ondulante, es decir, se propaga en forma de ondas analógicas desde el objeto que lo produce. Las características propias de cualquier sonido (desde el producido por un automóvil hasta una bella canción), sus diferentes tonos y notas dependen precisamente de las propiedades físicas de las ondas que lo forman.

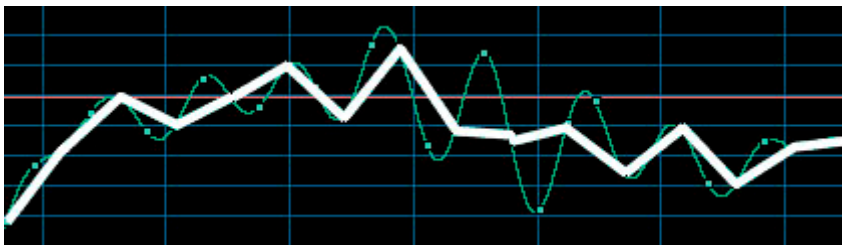
Para poder viajar desde el emisor al receptor, las ondas de sonido precisan de un medio físico de soporte, ya sea el aire de la atmósfera, al agua, etc. Tanto es así que en el espacio exterior, donde no hay medio físico soporte, no se pueden transmitir sonidos.

Si representamos en un gráfico un sonido complejo, obtendremos la siguiente figura:



En la que podemos apreciar los diferentes valores de onda que va tomando el sonido.

Todos sabemos que los equipos informáticos no trabajan con datos analógicos, sino que lo hacen con datos digitales, formados por estados binarios. Por lo tanto, para representar un sonido, desde el punto de vista informático, es preciso capturarlo en una naturaleza binaria, para lo que se hace un **muestreo** del mismo, tomando determinados valores de las ondas y representando dichos valores en formato digital. En cada captura obtendremos un punto de la gráfica anterior.



Pero, ¿Cuántas muestras deberemos tomar?. Este es el verdadero meollo de la cuestión, ya que cuantas más muestras tomemos, más fiel será el sonido capturado respecto al original, con lo que tendrá más calidad.

Para medir el número de capturas utilizamos la frecuencia del muestreo. Como un Herzio es un ciclo por segundo, la frecuencia de una captura en Herzios representa el número de capturas que realizamos en un segundo. Así, una frecuencia de muestreo de 20 KHz (20 Kilo Herzios = 20000 Herzios) realizará 20000 capturas de puntos cada segundo.

El oído humano es captar de captar la asombrosa cantidad de 44000 sonidos por segundo, es decir, 44 KHz. Por lo tanto, para que un sonido digital tenga suficiente calidad deberá estar basado en una frecuencia similar a ésta. En general, el valor estándar de captura de sonidos de calidad es de 44,1 Khz (calidad CD), aunque hay capturadoras de sonido profesionales que llegan hasta los 100 Khz, con objeto

de obtener un mayor número de puntos sobre la muestra, consiguiendo una calidad máxima.

Otro concepto del que habréis oído hablar en torno al sonido digital es el número de bits de una tarjeta de sonido. El origen de esta magnitud es que, a la hora de capturar el sonido, no sólo es importante el número de muestreos tomados, sino también la cantidad de información capturada en cada uno de esos muestreos.

Una vez capturado el sonido, para su posterior reproducción en un equipo informático es necesario mandar una serie de impulsos o posiciones a los altavoces para que creen el sonido a partir de ellos. ¿Cómo?. Bien, produciendo a partir de esas posiciones movimientos de las membranas de los altavoces, movimientos que transforman de nuevo el sonido digital en analógico, estado en el que es capaz de viajar por el aire y producir los estímulos necesarios en nuestros tímpanos, con lo que somos capaces de percibir el sonido “original”. Cuantas más posiciones de información se envíen a los altavoces, mejor calidad tendrá el sonido reproducido.

Con estas bases, se define el número de bits de un sonido digital como el número de impulsos de información (posiciones) que se envían a los altavoces para su transformación en ondas analógicas.

Las tarjetas de sonido actuales trabajan normalmente con 8 bits de información, con los que se pueden obtener 28=256 posiciones (ceros y unos binarios), aunque hay algunas de mayor calidad que son capaces de trabajar con capturas de 16 bits, que originan 216 = 65536 posiciones de información.

Como dato de referencia, los CDs actuales están basados en sonido grabado a 44 Khz y con un tamaño de muestra de 16 bits. Estas medidas se conocen con el nombre de sonido de calidad CD.

Por último, una vez que el sonido digital llega a nuestros oídos, impactan contra los tímpanos, verdaderas membranas especializadas que vuelven a transformar las ondas analógicas en impulsos eléctricos, que viajan hasta nuestro cerebro, donde son interpretados y producen las sensaciones auditivas que todos conocemos.

Una excepción al sonido anteriormente descrito, que podemos denominar "de datos de sonido", es el sonido sintetizado, en el que no se realiza ninguna captura de ondas sonoras reales, sino que es sonido totalmente digital, generado directamente en el equipo informático por en reproductor digital conocido con el nombre de MIDI (Music Instrument Digital Interface). Cuando se desea reproducir una nota musical concreta, se envía un comando MIDI al chip sintetizador, que se encarga de traducir ese comando en una vibración especial que produce la nota. Mediante este sistema es posible crear melodías bastante aceptables, aunque nunca tendrán la calidad ni riqueza de una onda sonora natural capturada.

Sonido en HTML (III)

Formatos de sonido

A la hora de incluir ficheros de sonido en nuestras páginas web debemos distinguir entre los que pueden ser directamente ejecutados por el navegador y aquellos que deben ser abiertos por un programa propio, que deberá tener el usuario instalado en su equipo para poder reproducir el fichero.

De forma general, podemos incluir en la web los siguietes tipos de ficheros de audio.

- **WAV** (Wave form Audio File format): formato típico de la casa Windows, de elevada calidad, usado en las grabaciones de CDs, que trabaja a 44 Khz y a 16 bits. Consta básicamente de tres bloques: el de identificación, el que especifica los parámetros del formato y el que contiene las muestras. Su principal inconveniente es el elevado peso de los ficheros, por lo que su uso queda limitado en Internet a la reproducción de ruidos o frases cortas. La extensión de estos ficheros es .wav. Es soportado por Internet Explorer y Netscape 4x.

- **AU** (Audio File format): formato creado por la casa Apple para plataformas MAC, cuyos ficheros se guardan con la extensión .au

- **MIDI** formato de tabla de ondas, que no guardan el sonido a reproducir, sino un código que nuestra tarjeta de sonido tendrá que interpretar. Por ello, este tipo de ficheros no puede almacenar sonidos reales, como voces o música rela grabada; sólo puede contener sonidos almacenables en tablas de ondas. Como contarpartida, los ficheros MIDI, que se guardan con extensión .mid, son de pequeño tamaño, lo que los hace idóneos para la web. Es soportado por Internet Explorer y Netscape 4x.

- **MP3** (MPEG 1 Layer 3): desarrollado por el MPEG (Moving Picture Expert Group), obtiene una alta compresión del sonido y una muy buena calidad basándose en la eliminación de los componentes del sonido que no estén entre 20 hz y 16 Kh (los que puede oir el ser humano normal). Tiene en cuenta el sonido envolvente (surround) y la extensión multilingüe, y guarda

los ficheros con la extensión .mp3, y permite configurar el nivel de compresión, consiguiéndose calidades similares a las del formato WAVE pero con hasta 10 veces menos tamaño de fichero. Es soportado directamente sólo por Internet Explorer 5.5 y superiores.

- **MOD** especie de mezcla entre el formato MIDI y el formato WAV, ya que por un lado almacena el sonido en forma de instrucciones para la tarjeta de sonido, pero por otro puede almacenar también sonidos de dintrumentos musicales digitalizados, pudiendo ser interpretados por cualquier tarjeta de sonido de 8 bits. No es un formato estándar de Windows, por lo que su uso es más indicado para sistemas Mac, Amiga o Linux. La extensión de los ficheros es .mod

- **µ-Law Format** de calidad similar al formato WAV, es original de las máquinas NeXt, y guarda sus ficheros con la extensión .au

- **Real Audio** de calidad media, aunque permite ficheros muy comprimidos, que guarda con extensión .rmp o .ra. Para su reproducción hace falta tener instalado el plugin Real Audio.

A la hora de trabajar con estos formatos de sonido, deberemos tener en cuenta las limitaciones en su uso, ya que muchos de ellos no pueden ser reproducidos más que en sistemas operativos concretos, y aún así, con plugins o programas específicos.

En busca de la compatibilidad, si usamos Windows como sistema operativo conviene usar para ficheros musicales a reproducir directamente en el navegador los formatos WAV y MIDI, que son los más compatibles.

En cambio, si lo que deseamos es poder brindar a nuestros visitantes la opción de navegar con música ejecutable desde un programa externo, lo mejor es usar ficheros en formato MP3, ya que en la actualidad la mayoría de los navegantes tienen instalado en su equipo algún programa reproductor adecuado, pudiendo valer desde software incluido en Windows, como Windows Media Player, hasta aplicaciones esternas, como Winamp. En este caso, bsata colocar un enlace normal en nuestras páginas, apuntando al fichero de sonido.

Como ejemplo, si queremos enlazar en nuestra página un fichero MP3, bastaría con escribir:

```
<a href="sonidos/mp3.mp3" target="_blank"> Pincha aquí para oir la música. </a>
```

Que nos da:

Pincha aquí para oir la música

Con esto, al pinchar el usuario el enlace, se lanzará la aplicación que tenga asociada con el tipo de fichero MP3, que dependerá de la configuración interna de cada navegador y usuario.

Un caso especial es Netscape 6x. Casi no admite directamente ningún tipo de formato de sonido incrustado en la página, al no venir configuradas por defecto las aplicaciones o plugins necesarios. Y en el caso de ficheros enlazados, Nestscape 6x suele lanzar su propio reproductor, que suele ser de la casa AOL, precisando para la ejecución una serie de pasos para darse de alta en esa compañia como usuario del software.

Resumiendo: cada usuario tendrá configurada su máquina de forma particular, soliendo prevalecer el último software de sonido instalado, ya que estos programas suelen adueñarse de ciertos tipos de ficheros para su ejecución automática. Entre las apliaciones posibles de ejecución de ficheros de audio, bien de forma directa o en forma de plugina para los navegadores, destacan Windos Media Player, Real Player, Winamp, Quick time, etc.

Sonido en HTML (IV)

Incluir sonidos en la web.

Una vez elegidos nuestros ficheros de sonido, es hora de incluirlos en nuestra página web. Lógicamente, para que un fichero de audio pueda ser reproducido por un navegador es necesario que su máquina tenga incluida una tarjeta de sonido y un par de altavoces.

Existen diversas formas de incluir un fichero de audio en una página, formas que dependen del tipo de fichero y del navegador usado, y podemos usar diferentes etiquetas para cada una de ellas.

BGSOUND

La etiqueta bgsound incorpora sonidos de fondo en una página web, sonidos que se ejecutan

automáticamente al cargarse la página. Es una etiqueta propietaria de Microsoft, por lo que sólo es interpretada por Internet Explorer, admitiendo los formatos de audio MID y WAV, aunque generalmente también acepta AU y MP3, en versiones actuales del nevagador o mediante plugins de uso general.

Su sintaxis general, con sus atributos más importantes, es del tipo:

```
<bgsound src="ruta_fichero" loop="l" balance="b" volume="v"></bgsound>
```

Donde:

- **src="ruta_fichero"** fija la ruta en la que se encuentra el fichero de audio a reproducir. La ruta puede ser relativa a nuestro sistema de carpetas local, absoluta respecto el sistema de carpetas del servidor web o una URL completa que localice el fichero en Internet.

- **loop="l"** determina el número de veces (l) que se debe ejecutar el fichero de audio. Si le damos el valor infinite, el fichero se reproducirá indefinidamente.

- **balance="b"** determina el balance del sonido entre los dos altavoces del equipo, es decir, la potencia o intensidad con que se oirá en cada uno de ellos (derecho e izquierdo). Sus valores pueden estar entre -10,000 y +10,000, correspondiendo el valor 0 a un balance equilibrado entre los dos altavoces.

- **volume="v"** fija el volumen al que se oirá el sonido, y sus valores pueden variar entre -10,000 (mínimo) y 0 (máximo). No es soportado por los equipos MAC.

Ejemplo:

```
<bgsound src="../sonidos/wav.wav" balance=0 volume=0></bgsound>
```

Que podéis ver funcionando en esta ventana (sólo Internet Explorer).

La etiqueta bgsound admite muchas más propiedades (disabled, delay, id, class, controls, etc.). Asímismo, esta etiqueta es accesible en Internet Explorer mediante código JavaScript, pudiendo modificar en tiempo real sus propiedades balance, loop, src, y volume, aunque ésta última sólo es accesible en plataformas PC. Para una información completa sobre todas las propiedades y funcionalidades de este etiqueta podéis visitar la página correspondiente de Microsoft:

http://msdn.microsoft.com/library/default.asp?url=/workshop/author/dhtml/reference/objects/bgsound.asp

EMBED

Nestcape Navigator implementó la etiqueta embed para incorporar ficheros de audio. Es ésta una etiqueta de caracter general, que se usa para la inclusión en las páginas web de todos aquellos archivos ajenos al navegador y que necesitan por lo tanto la ejecución de algún plugin para su interpretación.

Paradójicamente, Internet Explorer asumió después el uso de esta etiqueta para la inclusión de ficheros de audio, para llegar a interpretarla mejor y ampliarla con más atributos y propiedades, de tal forma que la ejecución de sonidos con embed es actualmente más cómoda con este navegador, al incorporar la suite de Microsoft sus propios plugins para la interpretación de los diferentes formatos de audio. En cambio, si usamos Netscape Navigator nos encontraremos en muchos casos con un fallo en la reproducción o con un engorroso mensaje de necesidad de algún plugin especial (sobre todo en las versiones 6x), lo que nos obligará a visitar la página de Netscape para su descarga e instalación, que muchas veces no será efectiva.

Sea como sea, hay que indicar que esta etiqueta nos va a incluir en la página web un objeto especial, una especie de consola de mando, denominada Crescendo, que consta de tres botones, similares al de cualquier reproductor de audio: un botón Play, para comenzar la reproducción (si no está establecida a automática), un botón Pause, para detenerla momentáneamente y un botón Stop, para detenerla definitivamente (puesta a cero). Esta consola es diferente según el navegador usado; en el caso de Internet Explorer se muestra la típica consola de Windows Media, cuyo tamaño podemos configurar, mientras que en Netscape se muestra una consola propia, de tamaño fijo definido.

La sintaxis general de la etiqueta embed es del tipo:

```
<embed atributo1="valor1" atributo2="valor2"...atributoN="valorN"></embed>
```

Y en el caso que nos ocupa, de la inclusión de ficheros de audio, los atributos podemos dividirlos en dos tipos:

1. Atributos referentes al sonido:

- **src="ruta_fichero"**, que fija la ruta en la que se encuentra el fichero de audio a reproducir. La ruta puede ser relativa a nuestro sistema de carpetas local, absoluta respecto el sistema de carpetas del servidor web o una URL completa que localice le fichero en Internet.

- **loop="l/true/false"**, que determina el número de veces que se debe ejecutar el fichero de audio. Los valores admitidos son l (número entero de veces), true (infinitas veces) y false (sólo una vez). Sólo es reconocida por Netscape Navigator.

- **playcount="n"**, que define el número de veces (n) que se debe ejecutar en fichero de audio en el caso de Internet Explorer.

- **type="tipo_fichero"**, atributo importante, que declara el tipo de fichero de audio que estamos usando, con lo que el navegador web puede ejecutar el programa o plugin adecuado para la reproducción del fichero. Puede ser audio/midi, audio/wav, etc.

- **autostart="true/false"**, que determina si el fichero de audio debe empezar a reproducirse por sí sólo al cargarse la página o si por el contrario será preciso la actuación del usuario (o de código de script) para que comience la audición.

- **pluginspage="URL"**, que establece, en caso de ser necesario un plugin especial para reproducir el fichero, la página web donde se puede descragar el mismo. Sólo se activa en el caso de que el navegador no sea capaz de reproducir el fichero por sí mismo, y es soportada tan sólo por Netscape Navigator.

- **name="nombre"**, que asigna un nombre identificador (debe ser único en la página) a una etiqueta embed determinada, con objeto de ser accedida luego por lenguajes de script.

- **volume="v"**, que determina el volumen de reproducción del sonido, y que puede variar entre 0 y 100. Es sólo soportada por Netscape Navigator, que en la consola muestra el valor establecido en su indicador de volumen, siendo su valor por defecto 50. En en caso de Internet Explorer, el valor del volumen por defecto es 50 en plataformas PC, y 75 en MAC, siendo necesario actuar sobre el control de volumen de la consola para modificarlo.

2. Atributos referentes a la consola:

- **hidden="true/false"**, que establece si la consola va a ser visible (false) o no (true). Es éste un aspecto polémico, ya que si ocultamos la consola obligamos al usuario a oir nuestro fichero, sin posibilidad de detenerlo ni de modificar el volumen, y si la mostramos estaremos incrustando en la pantalla un objeto que muchas veces nos romperá el esquema de diseño de nuestra página. Queda determinar su uso en cada caso concreto.

- **width="w"**, que determina el ancho visible de la consola, en pixels. height="h", que determina el alto visible de la consola, en pixels. Estos atributos son también muy importantes, caso de que hayamos establecido hidden= "false", ya que de su valor va a depender la correcta visulazación de la consola. En el caso de Internet Explorer, que muestra un logo de Windows Media sobre los controles, el tamaño mínimo aceptable debe ser de 140x100 pixels, ya que si no la consola saldrá deformada en exceso o recortada. Y en el caso de Netscape Navigator, deberemos asignar unos valores de 145x60 pixels, que es lo que ocupa la consola; si ponemos un tamaño menor, la consola será recortada, perdiendo funcionalidades, y si asignamos un tamaño mayor, aparecerán espacios grises alrededor de la consola, afeando el aspecto de la página. Si no especificamos estos atributos y tampoco hidden, nos aparecerán en la página tan sólo los mandos de la consola, sin logotipos añadidos (Internet Explorer) o la consola recortada (Netscape Navigator).

- **align="top/bottom/center/baseline/left/right/ texttop/middle/absmiddle/absbotom"**, análogo al de la etiqueta IMG, define la alineación horizontal o vertical de la consola respecto de los elementos de la página.

- **hspace="hs"**, que establece la separación horizontal, vspace="vs", que establece la separación vertical, en pixels, entre la consola y los elementos de la página que la redean. Análoga a sus equivelentes de la etiqueta IMG.

Estos son los atributos principales, aunque podemos encontrar referencias de otros admitidos, aunque no suelen ser operativos en la realidad, ya que no suelen funcionar de forma correcta o son específicos de Nestcape (como toda la serie de atributos que configuran los controles de la consola.

Ejemplo sin consola:

```
<embed src="../sonidos/mid.mid" hidden="true" type="audio/midi" autostart="true"></embed>
```

Que podemos ver en funcionamiento en esta ventana.

Ejemplo con consola:

```
<embed src="../sonidos/mid.mid" hidden="false" type="audio/midi" autostart="false" width="150"
height="100"></embed>
```

Sonido en HTML (V)

La etiqueta OBJECT.

Con objeto de normalizar la inclusión de ficheros no nativos en los navegadores web se decidió sustituir las diferentes etiquetas que realizaban este papel (APPLET, BGSOUND, EMBED, etc.), y que no pertenecían a los estándares web, por una etiqueta general, que fuera capaz de incrustar en el navegador todo tipo de ficheros. La etiqueta elegida en el estándar HTML 4.0 fué OBJECT, a la que se dotó de suficientes atributos y flexibilidad para poder realizar correctamente su trabajo. Debido a esto, la propuesta ha sido usar la etiqueta object también para incluir ficheros de audio de todo tipo en las páginas web.

Ahora bien, la aceptación e implementación que la misma a tenido varía según el navegador en particular, así como en función del objeto a incrustar. De este forma, Internet Explorer a realizado su propia implementación de la etiqueta object, incluyendo en ella referencias a filtros y componentes ActiveX específicos para los ficheros de audio. Por su lado, los navegadores Netscape no soportan correctamente este etiqueta para ficheros de este tipo.

Restringiéndonos a Internet Explorer, la polémica sigue, ya que en diferentes manuales nos encontraremos diferentes formas de incrustar sonidos mediante object, unas que funcionan bien, y otras que no. ¿Porqué sucede esto?. Yo creo que porque Microsoft ha ido usando la etiqueta object para implementar todo un grán conjunto de componentes propios, que además han ido adaptándose a las diferentes versiones de Internet Explorer.

Como regla general, válida no sólo para incrustar ficheros de sonido, sino también para otros tipos, la etiqueta object va a definir un objeto o componente externo encargado de la reproducción del fichero, que en el caso de Internet Explorer suele ser algún tipo de control ActiveX. Mediante object se instancia el objeto, se declara su URL y sus principales propiedades generales, y mediante un conjunto de etiquetas especiales, PARAM, se le van pasando los valores que necesita para su correcto funcionamiento o para su configuración deseada.

La sintaxis general de la etiqueta object, para el caso de ficheros de sonido, es del tipo:

```
<object atributo1="valor1" atributo2="valor2" ... atributoN="valorN">
<param name="nombre" value="valor">
<param name="nombre" value="valor">
...
</object>
```

Los principales atributos de object, en referencia a ficheros de audio, son:

- **classid="identificador_objeto"**, que fija la URL del objeto o componenteexterno necesario para reproducir el fichero de audio, y la implementación CLSID de los controles ActiveX necesarios.
- **type="tipo_fichero"**, atributo importante, que declara el tipo de fichero de audio que estamos usando.
- **width="w"**, que determina el ancho visible de la consola, en pixels.
- **height="h"**, que determina el alto visible de la consola, en pixels.

- **align="top/bottom/center/baseline/left/right /texttop/middle/absmiddle/absbotom"**, análogo al de la etiqueta IMG, define la alineación horizontal o vertical de la consola respecto de los elementos de la página.

- **hspace="hs"**, que establece la separación horizontal, vspace="vs", que establece la separación vertical, en pixels, entre la consola y los elementos de la página que la redean. Análoga a sus equivelentes de la etiqueta IMG.

- **autostart="true/false"**, que determina si el fichero de audio debe empezar a reproducirse por sí sólo al cargarse la página o si por el contrario será preciso la actuación del usuario (o de código de script) para que comience la audición.

- **standby="mensaje"**, que presenta en pantalla un mensaje al usuario mientras el fichero se carga.

  En cuanto a los elementos param, los más importantes son:

- **param name="FileName" value="ruta_fichero"**, determina la ruta o URL del fichero de audio a reproducir. No es necesario utilizar sólo ficheros WAV o MID, pudiendo reproducirse también ficheros MP3 o Real Audio. El reproductor del primero lo incluye Explorer en ActiveMovie (componente de Windows Media).

- **param name="autostart" value="true/false"**, indica al navegador si se debe empezar a reproducir el sonido automáticamente al cargar la página o si por el contrario será preciso que el usuario pulse el botón Play para ello.

No son estos todos los atributos y parámetros posibles. Es más, en cuanto nos metemos en componentes Microsoft, podemos encontrarnos multitud de configuraciones posibles, que nos van a permitir fijar muchos aspectos de los mismos. Dejo a cada uno la posibilidad de profundizar en el estudio de aquellos componentes y propiedades que necesite, pero sabiendo que con los elementos vistos arriba tenemos más que suficiente para presentar un fichero de audio en nuestra página web.

xEjemplo:

```
<object classid="CLSID:05589FA1-C356-11CE-BF01-00AA0055595A" width="150" height="175"
type="audio/midi">

<param name="FileName" value="../sonidos/xfiles.mid">
<param name="autostart" value="true">
</object>>
```

La etiqueta A.

Si hasta ahora hemos visto cómo podemos incluir en nuestras páginas sonidos de fondo o inicializados por el usuario mediante interacción con la consola Crescendo, vamos a ver ahora cómo podemos implementar audio mediante el uso de una de las etiquetas más polivalentes en HTML: la etiqueta A.

Efectivamente, los enlaces son la base del hipertexto, base a su vez de la web, y dentro de sus múltiples usos podemos considerar el enlace a ficheros de audio. El fichero enlazado puede ser interpretado directamente por el navegador (porque sea de reproducción directa o se tenga instalado el plugin adecuado) o puede ser ejecutado por un programa independiente que se abra automáticamente (Winamp, Real Audio, etc.), siendo este el caso más común. Si el usuario no dispone del programa o plugin adecuado, se le abrirá una ventana de descarga del fichero, con lo que podrá guardarlo hasta disponer de la aplicación necesaria para su reproducción.

La sintaxis general en este caso será del tipo:

```
<a href="ruta_fichero">Mensaje</a>
```

Ejemplo de fichero MID:

```
<a href="../sonidos/watermark.mid">Música para tí</a>
```

Ejemplo de fichero MP3:

```
<a href="../sonidos/mp3.mp3">Madonna</a>
```

Capitulo 4
Ayudas técnicas

Las ayudas técnicas son pequeños reportajes de interés general muy útiles para conocer rápidamente diversos temas de interés.

Enlaces sin subrayado

Tienes dos maneras de aprender a hacer esto:

- Por la vía rápida, más fácil hoy pero peor para en el futuro.
- Aprendiendo bien el asuntillo, desde el principio.

RÁPIDO

solo ponle el atributo - **style="text-decoration:none"** - al enlace que quieras sin subrayar. Ejemplo:
desarrolloweb

APRENDIENDO BIEN

Eso se hace con **Hojas de Estilo en Cascada**, tambien llamadas **CSS**, incluso la forma en la rápida estás utilizandolas, posiblemente sin saberlo.
Si aprendes CSS podras saber cómo quitar el subrayado de manera más eficaz, por ejemplo, quitándole de una sola vez los subrayados a todos los enlaces de la pagina o del sitio entero. Además aprenderás muchas otras cosas muy importantes y chulas.

Tu eliges, **pero no se te ocurra hacer un sitio entero**, o incluso una pagina grande donde todos los enlaces esten sin subrayar, **con el método rápido** porque a la larga en más lento y costoso. El manual que te propongo es corto de leer pero preciso y fácil de entender, te ayudara mucho en tu evolución como diseñador web.

Formátos gráficos para páginas web

El componente gráfico de las páginas web tiene mucha importancia, es el que hace que estas sean vistosas y el que nos permite aplicar nuestra creatividad para hacer del diseño de sitios una tarea agradable. Es también una herramienta para acercar los sitios al mundo donde vivimos, si embargo, es también el causante de errores graves en las páginas y hacer de estas, en algunos casos, un martirio para el visitante.

Las nociones básicas para el uso de archivos gráficos son sencillas, conocerlas, aunque sea ligeramente, nos ayudará a crear sitios agradables y rápidos. No cometer errores en el uso de las imágenes es fundamental, aunque no seas un diseñador y las imágenes que utilices sean feas, utilízalas bien y así estarás haciendo más agradable la visita a tus páginas.

Tipos de archivos

En Internet se utilizan principalmente dos tipos de archivos gráficos GIF y JPG, pensados especialmente para optimizar el tamaño que ocupan en disco, ya que los archivos pequeños se transmiten más rápidamente por la Red.

El formato de archivo GIF se usa para las imágenes que tengan dibujos, mientras que el formato JPG se usa para las fotografías. Los dos comprimen las imágenes para guardarlas. La forma de comprimir la imagen que utiliza cada formato es lo que los hace ideales para unos u otros propósitos.

Adicionalmente, se puede usar un tercer formato gráfico en las páginas web, el PNG. Este formato no tiene tanta aceptación como el GIF o JPG por varias razones, entre las que destacan el desconocimiento del formato por parte de los desarrolladores, que las herramientas habituales para tratar gráficos (como por ejemplo Photoshop) generalmente no lo soportan y que los navegadores antiguos también tienen problemas para visualizarlas. Sin embargo, el formato se comporta muy bien en cuanto a compresión y calidad del gráfico conseguido, por lo que resultaría útil si se llega a extender su uso.

GIF

A parte de ser un archivo ideal para las imágenes que estén dibujadas tiene muchas otras características que son importantes y útiles.

Compresión: Es muy buena para dibujos, como ya hemos dicho. Incluso puede ser interesante si la imagen es muy pequeña, aunque sea una foto.

Transparencia: es una utilidad para definir ciertas partes del dibujo como transparentes. De este modo podemos colocar las imágenes sobre distintos fondos sin que se vea el cuadrado donde está inscrito la el dibujo, viendose en cambio la silueta del dibujo en cuestión.
Para crear un gif transparente debemos utilizar un programa de diseño gráfico, con el podemos indicar qué colores del dibujo queremos que sean transparentes.
Generalmente, definimos la transparencia cuando vamos a guardar el gráfico.

Parte de esta imagen es transparente

Colores: Con este formato gráfico podemos utilizar paletas, conjuntos, de 256 colores o menos. Este es un detalle muy importante, puesto que cuantos menos colores utilicemos en la imagen, por lo general, menos ocupará el archivo. En ocasiones, aunque utilicemos menos colores en un gráfico, este no pierde mucho en calidad, llegando a ser inapreciable a la vista.
En algunos programas podemos modificar la cantidad de colores al guardar el archivo, en otros lo hacemos mientras creamos el gráfico.

32 colores

16 colores

8 colores

Imagen tomada con distintas paletas de colores. Se puede apreciar como con pocos colores se ve bien el gráfico y como pierde un poco a medida que le restamos colores.

JPG

Veamos ahora cuales son las características fundamentales del formato JPG:

Compresión: Tal como hemos dicho anteriormente, su algoritmo de compresión hace ideal este formato para guardar fotografías. Además, con JPG podemos definir la calidad de la imagen, con calidad baja el fichero ocupará menos, y viceversa.

Transparencia: Este formato no tiene posibilidad de crear áreas transparentes. Si deseamos colocar una imagen con un área que parezca transparente procederemos así: con nuestro programa de diseño gráfico haremos que el fondo de la imagen sea el mismo que el de la página donde queremos colocarla. En muchos casos los fondos de la imagen y la página parecerán el mismo.
Colores: JPG trabaja siempre con 16 millones de colores, ideal para fotografías.

Optimizar ficheros

Para que las imágenes ocupen lo menos posible y se transfieran rápidamente por la Red debemos aprender a optimizar los ficheros gráficos. Para ello debemos hacer lo siguiente:

Para los archivos GIF: Reduciremos el número de colores de nuestra paleta. Esto se hace con nuestro editor gráfico, en muchos casos podremos hacerlo al guardar el archivo.

GIF 256 colores - 10,8 KB

GIF 32 colores - 5,5 KB

GIF 4 colores - 2 KB

Para los archivos JPG: Ajustaremos la calidad del archivo cuando lo estemos guardando. Este formato nos permite bajar mucho la calidad de la imagen sin que esta pierda mucho en su aspecto visual.


JPG
calidad 0
3 KB


JPG
calidad 20
5,9 KB


JPG
calidad 50
10 KB

Es imprescindible disponer para optimizar la imagen de una herramienta buena que nos permita configurar estas características de la imagen con libertad y fácilmente. Photoshop 5.5 o 6 es un programa bastante recomendable, pues incorpora una opción que se llama "Guardar para el Web" con la que podemos definir los colores del gif, calidad del JPG y otras opciones en varias muestras a la vez. Así con todas las opciones configurables, viendo los resultados a la vez que el tamaño del archivo podemos optimizar la imagen de una manera precisa con los resultados que deseamos.

También existen en el mercado otros programas que nos permiten optimizar estas imágenes de manera sorprendente. Una vez hemos creado la imagen la pasamos por estos programas y nos comprimen aun más el archivo, haciéndolo rápido de transferir y, por tanto, más optimo para Internet. Al ser estas utilidades tan especializadas los resultados suelen ser mejores que con los programas de edición gráfica.

Ejemplos de optimizadores gráficos:
- WebGraphics Optimizer
- ProJPG, GIF Imantion

Y con versiones Online:
- JPG - GIF Crunchers
- GIF Wizard

Icono en favoritos

En este artículo vamos a aprender a colocar un icono en el menú de favoritos, de modo que cuando alguien guarde la página en sus favoritos se vea la página con el icono que nosotros deseemos, en lugar del icono de Internet Explorer. El icono de nuestra página también se podrá ver en la barra de direcciones del explorador, siempre a partir del momento en que un usuario guarde la página en favoritos.

Es un efecto muy sencillo y vistoso, pero tiene un serio inconveniente: solo funciona en Internet Explorer versión 5. No funciona en Netscape ni en versiones distintas de Explorer.

Cómo conseguir un icono

Un icono es un archivo especial, que tiene extensión .ico. Los editores de gráficos habituales no trabajan con este formato, por lo que se requerirá del uso de una herramienta especial para la creación y tratamiento de la imagen que vamos a utilizar como icono.

Una popular herramienta útil para trabajar con iconos es Microangelo. Permite la creación de gráficos que luego se pueden guardar como .ico. También permite convertir a iconos gráficos en formato GIF.

Donde colocar el icono

Para conseguir nuestro objetivo tenemos dos posibilidades. La primera se trata de colocar un icono a todo un sitio web entero. Con la segunda opción podremos asignar un icono en una página de manera independiente.

Colocar un icono en un sitio, de manera global

Para ello debemos conseguir un archivo ico y darle el nombre "favicon.ico". Este archivo lo colocaremos en el directorio raiz del dominio. Cuando se introduzca en favoritos cualquier página del sitio se vera el icono seleccionado.

No siempre tenemos acceso al directorio raiz del dominio. Imaginemos que publicamos una web en Geocities, en este caso nunca tendremos acceso al directorio raiz, www.geocities.com, sino que tendremos acceso al directorio que nos hayan asignado, www.geocities.com/mipagina. Para este caso, tenemos una segunda manera de insertar el icono.

Colocar un icono específico en una página

En este caso tenemos que colocar una etiqueta en la cabecera del documento HTML (entre y). Esta etiqueta la tenemos que colocar en cada página que queramos asociar con un icono.

<link rel="SHORTCUT ICON" href="myicono.ico">

Por supuesto, en el atributo href de la etiqueta anterior, se ha de colocar el nombre del icono que queremos utilizar y la ruta para acceder a el. En el caso del ejemplo anterior se supone que un archivo llamado myicono.ico estará en el mismo directorio que la página web.

GIFs animados

No sabes hacer un círculo con el PaintBrush? no te preocupes, igualmente puedes hacer magníficas animaciones. Eso si, necesitarás la ayuda de alguna aplicación que haga algo de trabajo por ti. En la sección de recursos puedes encontrar lo que necesitas. Para la realización de los ejemplos hemos utilizado el Animator Shop 2 que acompaña al Paint Shop 6.0.

Lo mas fácil

El primer paso es tener una idea, cosa nada fácil. Se pueden hacer infinidad de maravillas, lo mejor es buscar buenos resultados con poco trabajo. Hay que recordar siempre las limitaciones de tamaño en internet, con lo que se debe trabajar siempre con poca profundidad (cantidad de colores) y tamaños

reducidos. También es mejor usar un buen programa que optimize el tamaño de la animación.

Para el que realmente tenga problemas para hacer un círculo puede crear una animación con cualquier imagen GIF, aplicandole alguno de los efectos que ofrecen las aplicaciones. Un ejemplo de la aplicación de un simple efecto a una simple imagen, que se puede hacer con cuatro clicks:

Tus propias animaciones

Mucho más interesante es el realizar uno mismo su propia animación para eso solo hay que seguir unos sencillos pasos. Primero debes determinar el tamaño de la imagen animada, cada uno de los fotogramas debe respetar ese tamaño. También debes encontrar el número mínimo de fotogramas que necesitas para crear el realismo deseado, este número no suele ser muy alto. Debes seguidamente realizar cada uno de los fotogramas, la mejor manera es hacer pequeños cambios en cada uno de ellos. Después los reunimos en un programa de animación y los montamos. Una muestra de lo que se puede realizar con unas decenas de clicks:

Realizada simplemente girando cada una de las letras de forma diferente. Sobra deciros que para completar el ciclo correctamente es necesario que el último fotograma sea similar al primero. Gracias a la baja profundida (16 colores) y a que utilizamos solo cuatro fotogramas la animacón solo ocupa 988 bytes!

Ultimas sugerencias

Esta animación ha necesitado unos centenares de clicks. Ha estado realizada en dos partes: la animación de las bandas laterales y la de la imagen interior. Aún con la mínima profundidad (2 colores) el archivo ocupa 11k ya que esta compuesto de 27 fotogramas.

Hay otros importantes factores sobre los que podemos jugar, almenos si nuestro software lo permite. Uno es la duración de los fotogramas, podemos ralentar o acelerar las animaciones. Pero cuidado, porque fotogramas lentos disminuyen la fluidez de la animación, recordamos que lo importante es que el ojo no llegue a "ver" cada uno de los fotogramas por separado.

Otra posibilidad es la de no repetir indefinidamente la animación, sino pasarla una sola vez o un numero determinado de veces. Muy útil como inicio de vuestras páginas. Por último deciros que las animaciones también se pueden utilizar como fondos, estas animaciones deben ser muy sutiles para no cargar demasiado la página.

Programas para crear GIFs animados

Hay multitud de programas para la creación de gifs animados. Podemos hacernos una idea del número de aplicaciones existentes visitando el directorio de animación de imágenes del sitio de descargas Tucows, donde hay muchas utilidades para animación en general y específicas para gifs animados.

De todos modos, aquí tenemos una serie de recomendaciones de programas.

- Photoshop, a partir de su versión 5.5, incorpora una utilidad complementaria llamada Image Ready para hacer, entre otras cosas, gifs animados. http://www.adobe.es/products/photoshop/
- el Xara3d, es bueno para hacer titulos sencillos, en 3d y animados. http://www.xara.com/products/xara3d/
- Tambien tenes el "unlead gif animator", para algo mas elaborado. http://www.ulead.com/ga/
- Con el "firewoks", podes hacer gif animados de muy buena resolucion y poco peso. http://www.macromedia.com/es/software/fireworks/
- Si necesitas movimientos mas complejos podes hacerlos en Flash y lo exportas como .gif animado. http://www.desarrolloweb.com/articulos/338.php

Nota: Esta referencia a los programas para hacer gifs animados se la debemos a Miguel Angel Alvarez y Quo Vadis Pepus. Sin ella, este artículo no quedaría completo.

Los colores y HTML

El la composición de webs juegan un papel muy importante los colores. Se indican en valores RGB, es decir, que para conseguir un color cualquiera mezclaremos cantidades de Rojo, Verde y Azul.

Los valores RBG se indican en numeración hexadecimal, en base 16. (Los dígito pueden crecer hasta 16. Como no hay tantos dígitos numéricos se utilizan las letras de la A a la F.

Tabla de color

0=0	4=4	8=8	C=12
1=1	5=5	9=9	D=13
2=2	6=6	A=10	E=14
3=3	7=7	B=11	F=15

Para conseguir un color, mezclaremos valores de esta manera:

RRGGBB
Donde cada valor puede crecer desde 00 hasta FF.

Ejemplo: Cómo se cambiaría la fuente para escribir en rojo:

```
<font color="#FF0000">Rojo</font>
```

Al Atributo **color** le damos un valor RGB en formato hexadecimal. El caracter # se coloca al principio de la cadena.

Otros colores:

Naranja	**#FF8000**
Verde turquesa	**#339966**
Azul oscuro	**#000080**

Colores compatibles en todos los sistemas

Como las páginas web las tienen que ver todos los usuarios, y los sistemas que utilizan para entrar son distintos, hay que utilizar colores compatibles con la paleta de todos ellos.

La forma de conseguir esto es limitando nuestros colores alos que se pueden conseguir utilizando la siguiente norma:

Utilizaremos siempre estos valores:
00
33
66
99
CC
FF

Ejemplos: #3366FF #FF9900 #666666

Se consiguen los colores siguientes:

#000000	#000033	#000066	#000099	#0000CC	#0000FF
#003300	#003333	#003366	#003399	#0033CC	#0033FF
#006600	#006633	#006666	#006699	#0066CC	#0066FF
#009900	#009933	#009966	#009999	#0099CC	#0099FF
#00CC00	#00CC33	#00CC66	#00CC99	#00CCCC	#00CCFF
#00FF00	#00FF33	#00FF66	#00FF99	#00FFCC	#00FFFF
#330000	#330033	#330066	#330099	#3300CC	#3300FF
#333300	#333333	#333366	#333399	#3333CC	#3333FF
#336600	#336633	#336666	#336699	#3366CC	#3366FF
#339900	#339933	#339966	#339999	#3399CC	#3399FF
#33CC00	#33CC33	#33CC66	#33CC99	#33CCCC	#33CCFF
#33FF00	#33FF33	#33FF66	#33FF99	#33FFCC	#33FFFF
#660000	#660033	#660066	#660099	#6600CC	#6600FF
#663300	#663333	#663366	#663399	#6633CC	#6633FF
#666600	#666633	#666666	#666699	#6666CC	#6666FF
#669900	#669933	#669966	#669999	#6699CC	#6699FF
#66CC00	#66CC33	#66CC66	#66CC99	#66CCCC	#66CCFF
#66FF00	#66FF33	#66FF66	#66FF99	#66FFCC	#66FFFF
#990000	#990033	#990066	#990099	#9900CC	#9900FF
#993300	#993333	#993366	#993399	#9933CC	#9933FF
#996600	#996633	#996666	#996699	#9966CC	#9966FF
#999900	#999933	#999966		#9999CC	#9999FF
#99CC00	#99CC33	#99CC66	#99CC99	#99CCCC	#99CCFF
#99FF00	#99FF33	#99FF66	#99FF99	#99FFCC	#99FFFF
#CC0000	#CC0033	#CC0066	#CC0099	#CC00CC	#CC00FF
#CC3300	#CC3333	#CC3366	#CC3399	#CC33CC	#CC33FF
#CC6600	#CC6633	#CC6666	#CC6699	#CC66CC	#CC66FF
#CC9900	#CC9933	#CC9966	#CC9999	#CC99CC	#CC99FF
#CCCC00	#CCCC33	#CCCC66	#CCCC99	#CCCCCC	#CCCCFF
#CCFF00	#CCFF33	#CCFF66	#CCFF99	#CCFFCC	#CCFFFF
#FF0000	#FF0033	#FF0066	#FF0099	#FF00CC	#FF00FF
#FF3300	#FF3333	#FF3366	#FF3399	#FF33CC	#FF33FF
#FF6600	#FF6633	#FF6666	#FF6699	#FF66CC	#FF66FF
#FF9900	#FF9933	#FF9966	#FF9999	#FF99CC	#FF99FF
#FFCC00	#FFCC33	#FFCC66	#FFCC99	#FFCCCC	#FFCCFF
#FFFF00	#FFFF33	#FFFF66	#FFFF99	#FFFFCC	#FFFFFF

Definiciones de pantalla

Cuando trabajas a **distintas definiciones**, como en la web, debes elegir el **publico objetivo** de la página y construirla para que ese público la vea bien, pero no te debes olvidar de los demás, de modo que estos también la visualicen, a ser posible, sin ningún problema.

El público objetivo actual de cualquier página debería de ser el que tiene la definición de **800x600**, ya que la mayoría de usuarios utilizan esta resolución de pantalla. También navega por Internet mucha gente a **640x480**, pero cada vez son menos el número de personas que utilizan esta definición. Para el caso de **1024x780**, esta definición es demasiado grande y exclusiva de usuarios con buenas máquinas, configuradas de manera avanzada, como deja fuera muchas personas no debemos utilizarla todavía, además, estos usuarios podrán ver también la página a definición menor sin demasiado perjuicio. Una última consideración sería que si la página a desarrollar es muy corporativa, de modo que pertenezca a una empresa importante y se tenga que ver bien en todas las máquinas posibles, sería razonable utilizar la definición de 640x480, pues esta nos asegura que todo el mundo podrá ver bien el web.

Una vez escogido el público objetivo, se han de componer las páginas y las imágenes para se vean bien

en la definición de pantalla que utilicen estos, de modo que nunca salgan las feas barras de desplazamiento horizontales. Para ello calcularemos el tamaño de los elementos de la página concienzudamente.

Otra cosa que se puede hacer es aplicar a los elementos de la página los tamaños (atributo width) utilizando porcentaje, de este modo, se ajustarán al tamaño de la definición del usuario automáticamente. Sin embargo, las imágenes no soportan el tamaño en porcentajes, es decir, no podemos ajustarlas así automáticamente, y en cualquier caso, no desearemos que la imagen se deforme al alterar artificialmente sus tamaños a través de estos atributos. Así pues, para el caso de las imágenes seguiremos en la necesidad de crearlas sin que exceda su tamaño horizontal la definición horizontal de público objetivo.

El que firma este artículo siempre utiliza tablas para maquetar las páginas que realiza. El método utilizado es el siguiente:

1. Hablamos de un diseño con tablas porque casi siempre es más adecuado que un diseño con frames, pero esto es otra discusión.
2. **Crear una tabla al principio de la página, que incluya toda la página**, y le asignamos el tamaño en pixels. Dependiendo de la resolución del tarjet de audiencia, el tamaño de la tabla variará, pero siempre serán 20 pixels menos que la definición objetivo. Estos pixels sobrantes se utilizan para las barras de desplazamiento verticales. Por ejemplo, si diseñamos para una resolución de pantalla de 800x600, el tamaño de la tabla será de 780 pixels.
   También se pueden crear las tablas en porcentajes, pero esto hará que la página se estire y se encoja dependiendo de la definición. No es un efecto muy deseable para los diseñadores, porque habrá que conseguir que la página se vea bien en todas las definiciones, y seguro que no se ve tan bien como lo que nosotros teníamos con la definición en la que estabamos diseñando, además de significar un esfuerzo extraordinario.
3. **Incluimos en la etiqueta <BODY> de la página los atributos: topmargin=0 leftmargin=0 marginheight=0 marginwidth=0**. Estos atributos sirven para eliminar los márgenes de la página y que las tablas se sitúen ocupando todo el espacio de la página. Los dos primeros atributos son para Internet Explorer, los dos siguientes son para Netscape Navigator, tenemos que poner los cuatro para asegurarnos que se vea bien en todos los navegadores más importantes.
4. Si deseamos incluir **márgenes en las páginas**, podemos jugar con los atributos **cellspacing y cellpadding**, en la tabla principal, para conseguirlos. En caso de que no nos guste este método podemos incluir **celdas adicionales en la tabla principal, que sean transparentes** (no meterles contenido), a las que les asignamos los tamaños deseados para los márgenes. El método que utilizo habitualmente es este último y casi siempre pongo los atributos cellspacing y cellpadding a cero.

Eso es todo, recuerda que siempre puedes ver el código fuente de las páginas web que te gustan, para ver cómo lo han hecho otros diseñadores.

Elegir alojamiento gratuito

Publicar una página web es el objetivo de la mayoría de las personas que visitan desarrolloweb.com. Para conseguirlo lo más imprescindible es **tener un servidor donde colocarlas**, accesible desde Internet, aparte de construir las propias páginas.

> **Nota:** Este artículo fué escrito durante el año 2000, por lo que la evaluación de los distintos alojadores gratuitos puede que no esté del todo actualizada, debido a que constantemente cambian sus servicios y a nosotros nos es imposible estar todo el tiempo revisando los distintos alojadores gratuitos comentados para actualizar el artículo con los cambios que ellos realicen.
>
> De todos modos, el artículo se mantiene más o menos actualizado gracias a la colaboración de muchos de los lectores y usuarios de los servicios, que nos han mandado sus observaciones sobre los servicios que están utilizando, cuando los datos que figuraban en este artículo estaban desfasados. Gracias a todos ellos y si ves alguna cosa incorrecta, por favor, envíanos el comentario con el enlace que hay al lado de la firma del artículo.

Dependiendo de cuál sean nuestras intenciones y deseos para el futuro de nuestra página deberemos elegir **un proveedor gratuito o uno de pago**, Será recomedable optar por uno de pago, en un proveedor seleccionado, si tratamos de hacer una web de una empresa o una que pretenda ser muy visitada, una web que ofrezca unos servicios y prestaciones más elevados tanto al administrador como a los visitantes. En este caso, conviene leer mejor el reportaje Cómo elegir un proveedor de pago,

Si por el contrario, deseamos publicar un trabajo con menores prestanciones, o simplemente no queremos gastar dinero en el hosting, será indicado optar por un **servicio de alojamiento gratuito**. Por suerte, existe una gran variedad de sitios que ofrecen a sus visitantes la oportunidad de publicar sus paginas personales o de empresa de manera gratuita y muchas de estas opciones tienen muy buena calidad.

Donde elegir un espacio

En un principio, el mejor espacio para publicar una web que podemos elegir puede ser el proporcionado por **nuestro proveedor de acceso a Internet**. Muchas veces nuestro proveedor, a la vez que ofrece la conexión a Internet, también ofrece un espacio para publicar, por el mismo precio que el acceso. Esta opción tiene sus ventajas:

- Suelen estar libres de publicidad
- En muchos casos, existen menos limitaciones por el tipo de contenidos
- Casi siempre tienen buena velocidad.

El único inconveniente a destacar -aparte de los problemas puntuales que tenga cada proveedor de acceso- es que **abandonemos algún día el proveedor y nos retiren las páginas publicadas**, pero esto se puede evitar teniendo un servicio de redirección que nos de una URL (dirección de web) para toda la vida, que redireccione a la página que esté activa en ese momento.

En el caso de que nuestro proveedor no ofrezca servicio de publicación de webs o no nos guste el servicio que ofrece, tenemos la opción de publicar las páginas en alguna **web que ofrezca espacio gratuito**. Este tipo de servicio es muy popular, exisiten muchas ofertas en Internet, algunas de ellas mejores de las proporcionadas por los proveedores de acceso, incluso por los proveedores de alojamiento.

En qué debemos fijarnos

Vamos continuar explicando algunos puntos en los que fijarnos para poder elegir un proveedor gratuito. Dando algunos ejemplos de sitios que ofrecen espacio con buenas prestaciones.

Recusos permitidos

Existen muchos sitios donde publicar una página, se diferencian fácilmente por los recursos que ponen a nuestra disposición para construirla. Cuando hablamos de recursos nos referimos a la **cantidad de herramientas y programas que nos dejan utilizar en nuestra web**. Ejemplos de recursos son: estadísticas, contadores, direcciones web cortas, mail gratuito, CGI, bases de datos... Los recursos se hacen necesarios cuando construimos una web avanzada, con secciones dinámicas como un buscador, un portal, o en general, una página que se alimenta de cualquier base de datos o programa para mostrar su contenido.

En la gama de recursos que ofrecen estos sitios gratuitos es muy variada. Algunos sitios casi no ofrecen ningún recurso, simplemente podemos publicar la página, y en otros sitios ofrecen más servicios que en algunos proveedores de alojamiento de pago.

Al final del reportaje podremos ver una comparativa sobre la cantidad de recursos ofrecida por algunos sitios proveedores de espacio gratuito para publicar webs.

Velocidad del servidor

Para elegir un servicio suficientemente veloz tenemos la oportunidad de **probar la rapidez con que bajan las páginas de alguna web que esté alojada en ese servidor**. Es decir, visitamos una web de Geocities.com y vemos lo rápido que llega, una web de Galeon.com y vemos la rapidez, comparando nosotros mismos. En un último caso podemos registrarnos nosotros mismos y probar la velocidad del espacio con nuestros propios contenidos. No pasa nada por registrarse en un sitio, probar cómo funciona y si no nos gusta simplemente nos vamos a otro, es gratis.

Para poder comprobar la velocidad de varios sitios de alojamiento gratuito hemos creado un web y lo hemos subido a servidores distintos. Con los siguientes enlaces podemos comprobar en qué servidor es más rápido para nosotros:

Galeón: http://www.galeon.com/ipaginate
Iespana: http://ipaginate.iespana.es
Geocities: http://www.geocities.com/michel98.geo/testvelocidad/index.html
Freeservers: http://quintointento.freeservers.com
Metropoli2000: http://desarrolloweb.metropoli2000.net

Un dato a considerar es que, dependiendo de dónde está situado físicamente el servidor que alojará nuestras páginas, la velocidad será distinta para nosotros. Es decir, un español ve más rápido un servidor español, pero un americano navegará con más velocidad si el servidor está en América. Esto, se lleva al límite cuando es nuestro proveedor de acceso el que nos ofrece el espacio, en este caso, seremos los que navegen más rápido por esa web, por estar tan proximos los dos sitios, tu acceso a Internet y el servidor de tu página.

Publicidad insertada en las páginas

También se pueden **clasificar los sitios para publicar en Internet por la publicidad que insertan**. Los sitios de espacio gratuito suelen subvencionarse mediante la inserción de anuncios en las páginas de los usuarios del servicio. Los tipos de publidad que podemos encontrar son los siguientes:

- **Publicidad en un frame inferior**. Es tal vez la más agresiva de las publicidades que pueden hacer, pues el banner permanece siempre visible en la página y es un poco feo para las páginas.
- **Publicidad dentro de la página, utilizando el banner superior**. Es el método de inserción de banners más habitual. Como desventaja tiene que el banner se exibe en la propia página, modificando nuestra web.
- **Publicidad en una ventana aparte**. Esta publicidad aparece en una ventanita aparte del explorador. Como ventaja está que no utiliza el espacio de nuestra página. Como desventaja contamos que estas ventanas son muy molestas para muchos usuarios.
- **Publicidad en la propia página, en un grafiquito flotando en la propia página.** Es muy interesante, pues ocupa muy poco espacio comparado con otros mecanismos y no molesta en exceso.
- **Sin publicidad**, sin duda el mejor de los casos. Lo que no nos asegura que el sitio no ponga publicidad algún día y deje de ser tan interesante. Actualmente, muy pocos sitios ofrecen espacio gratuito sin colocar algún tipo de publicidad.

Hay muchos tipos de publicidad insertada habitualmente, como se puede ver. Eso no quita que algunos sitios empiecen a utilizar varios sistemas a la vez.

Los ejemplos de sitios que estamos evaluando tienen la siguiente forma de insertar la publicidad en las páginas de los usuarios:

Galeon: En el momento de escribir este reportaje no colocaban banners en las páginas alojadas en su servidor. Sin embargo, actualmente si colocan publicidad en una capa que acompaña tu navegación y permanece siempre visible.

Metropoli 2000: Colocan un banner en la parte de arriba de la página. Es una opción interesante, porque reparten los ingresos por publicidad con el dueño de la página.

Iespana: Los banners de este sitio se colocan en un frame en la parte inferior de la página. Pero con un simple aviso podemos pasar cambiar esto para que la publicidad salga en una ventana aparte.

Geocities: Colocan un banner cuadrado que se puede incluso borrar de la web a base de clicks. El problema es que este banner se coloca con Javascript y en determinados navegadores da errores, lo que es peor incluso que un banner grande. En mi opinión un fallo de Javascript es algo bastante molesto y en mi Internet Explorer 4 fallan casi todas las páginas de Geocities.

Freeservers: Colocan un banner en la parte superior de la página, con enlaces a otras partes de Freeservers. Además, también colocan publicidad en ventanas a parte. Todos estos males se pueden eliminar si el usuario paga una cuota mensual. Según nos ha informado, también se puede configurar para que aparezcan flotando arriba, en la izquiera, derecha, abajo, etc.

Por manera de colgar los archivos

Existen diferentes maneras de subir los archivos. Por ejemplo, **los podemos subir con FTP o por medio de una página donde existe un formulario y donde podemos seleccionar los archivos que deseamos subir**. En la descripción de características del alojamiento suelen indicar acceso por FTP cuando esta permitido realizar este tipo de actualizaciones.

Para los **principiantes** será más cómodo subir las páginas por medio de **un formulario de una web**, pero para las personas **experimentadas**, a la larga, les será más sencillo y tendrá menos trabajo el realizar las actulizaciones del sitio **via FTP**.

Como ventaja de subir archivos con un formulario tenemos que no será necesario disponer de un programa de FTP ni tampoco saber utilizarlo. Como ventaja del FTP destacamos la rapidez de subida de archivos, ya que podemos seleccionar varios archivos de una sola vez y no tenemos que esperar a que se cargue la página web para que tengamos acceso al formulario, subir un directorio en una sola acción y otra mejora importante, que consiste en que podremos utilizar el programa con el que diseñamos las páginas web (tipo Dreamweaver o Homesite) para acceder al servidor y actualizar los contenidos automáticamente.

Nuestro consejo es aprender a utilizar FTP, si es que deseamos trabajar de manera más profesional y ahorrar tiempo a la larga.

Comparativa y conclusión

Aquí podemos ver una **tabla con las principales características** de los sitios que ofrecen alojamiento gratuito destacados en este reportaje.

	Geocities	**Metropoli 2000**	**Iespana**	**Galeon**	**Freeservers**
Espacio	15M	Ilimitado	Ilimitado	Ilimitado	12M
Correo	SI	Ilimitado	SI	SI	Sólo redirección
Estadísticas	SI	SI	SI	NO	SI
Dominio	NO	Pagando	Pagando	NO	Pagando y gratuitos
Subdominio	NO	SI	SI	NO	Varios a elegir
Páginas dinámicas	NO	PHP	PHP	NO	NO
Bases de datos	NO	MySQL	MySQL	NO	NO
Multimedia	Típica	Típica	Real Networks	Típica	Típica
CGI instalados	Alguno	Numerosos	Numerosos	NO	Algunos
CGI propios	NO	SI	NO	NO	NO
Herramientas componer y publicar	SI	?	SI	SI	SI
Acceso por FTP	SI	SI	SI	SI	NO
Publicidad	Cuadrado flotante	Banner superior ingresos a compartir	Frame inferior o ventana aparte	Cuadro flotante y popus abusivos	Banner superior
Transferencia*	Sin dato. limitada?	Ilimitada	Ilimitada	Sin dato. ilimitada?	512 MB/mes
Observa-ciones	Tiene errores javascript	Solo para contenidos de calidad	Para todos los públicos, buenas herramientas	La única sin publicidad	Nos costó mucho registrarnos

*__Transferencia__: Es la cantidad de información que recibes y mandas cada mes el servidor cuando le piden tu página. Empiezan a haber servicios gratuitos que limitan la transferencia porque al fin y al cabo es lo que cuesta dinero del hecho de ofrecer espacio gratuito y poco a poco se cierra el grifo de la Internet gratuita. Tener cuidado con este detalle, porque puede que el alojador gratuito elegido pueda ser inutil debido a que tenga mucho exito tu web y tenga limitado el tráfico.

Los Sitios que ofrecen **mayores recursos para hacer una página web son Iespana.es y Metropoli2000.net**, con la principal diferencia de que

En **Metropoli2000 sólo aceptan webs con contenidos de calidad**, sin despreciar al diseño. Es decir, necesitas tener una web en Internet previamente publicada y estos te dicen si es de suficiente calidad para poder publicarla en su servidor. Sin embargo, ellos te remuneran por los ingresos de publicidad, aunque actualmente parece ser que el mercado les ha hecho desistir de esta posibilidad. Para colmo de males, la publicidad cada vez es más abusiva y puede llegar a ser bastante tedioso el vistar una página en este servidor.

Iespana es bastante versatil, por permitir todo tipo de contenidos, incluso comerciales o de adultos. Además nos permiten programar en PHP 4 con base de datos MySQL (pagando), lo que hace muy interesante para aquellos desarrolladores que deseen programar aplicaciones de servidor. La velocidad del servicio, por lo menos con nuestra conexión a Internet, no es del todo agradable.

Galeón no nos gusta mucho. Cuando escribimos el artículo era un lugar interesante, gracias a su ausencia de publicidad, aunque ahora sí que la ponen y cada vez más abusiva. Actualmente saltan ventanas secundarias por todas partes y encima los cuadros flotantes son cada vez más grandes.

Freeservers no nos funcionó del todo bien, sobretodo en el registro, que tuvimos que intentarlo durante un par de días en un total de 5 ocasiones antes de conseguir ver la hoja final de registro. De todos modos, hemos recibido buenas críticas de este servidor.

Para mi, **Geocities una opción bastante atractiva**. Funciona bastante rápido con relación con otros sitios, por lo menos en mi caso. Pertenece al grupo de Yahoo! y eso siempre es una garantía de continuidad. Como contrapartida, en algunos casos limitaban la transferencia, con lo que es posible que a mitad de mes tu página se deje de transferir, si es que recibe muchas visitas. En lo que respecta a la publicidad, no abusan demasiado aunque puede ser un poco molesta por ponerse encima de los contenidos de nuestra web.

Direcciones de espacio gratuito

Iespana: http://www.iespana.es
Metropoli 2000: http://www.metropoli2000.net
Freeservers: http://www.freeservers.com
Geocities: http://www.geocities.com
Galeón: http://www.galeon.com

Cómo elegir un alojamiento

Para publicar una página en Internet debemos colocarla en un servidor. Los servidores son ordenadores conectados permanentemente a la Red que envían las páginas cuando los exploradores la piden.

Antes de nada vamos a diferenciar entre dos formas de alojar una web, en un servidor gratuito o en un servidor de pago. Los servidores de pago suelen tener mayores prestaciones y recursos disponibles que los gratuitos. En este reportaje vamos a tratar de orientar a las personas que tienen que contratar un servicio de pago, más indicado para negocios y páginas que pretendamos que sean populares.

Definir los recursos

Para encontrar un alojamiento para nuestra página primero nos tenemos que plantear los objetivos que queremos cumplir y los recursos que necesitaremos. Según sea el proyecto que vamos a publicar necesitaremos más o menos personalidad propia o determinados recursos como acceso a bases de datos , estadísticas, etc.

El primer paso podría ser decidir si vamos a necesitar un dominio propio para nuestro proyecto. Un dominio propio le dará más personalidad al sitio, se asociará mejor con el nombre de la empresa, a la vez que se hace más accesible la página. Con el reducido precio que tienen actualmente los dominios sería muy indicado adquirir uno para nuestra página. En muchos casos al contratar un espacio para alojar la web nos van a ofrecer directamente nuestro propio dominio, si no es así, lo podemos solicitar expresamente.

Lo más habitual es que en el proveedor donde alojemos la página se encargue también del registro del dominio, sin prácticamente otro coste adicional que las tasas del NIC, 30$ aproximadamente. Para conocer más sobre los dominios y su registro se puede consultar un reportaje también publicado en desarrolloweb.

Aparte del dominio existen otros recursos podemos necesitar, como:

- Programación en lenguajes específicos
- Acceso a bases de datos
- Administración de las DNS...

Deberemos definir los recursos a utilizar y buscar un proveedor que los acepte todos. Si estás proyectando una página sencilla, probablemente no te preocupará el número de recursos avanzados que posea tu servidor, pero siempre será interesante evaluar otros recursos más básicos como:

- El número de direcciones de correo que te van a proporcionar
- Los CGIs que tenga instalados, como contadores, envío de formularios por correo electrónico, etc.
- Estadísticas...

Velocidad

Es una pena que no exista un método realmente fiable para controlar uno de los aspectos que más preocupa a todos los que buscamos un alojamiento. Para conocer la calidad el servicio que nos va a ofrecer el proveedor deberíamos preguntarles a ellos sobre sus líneas y la capacidad que tienen de transferencia con Internet. Pero no solo eso sino también preguntarles por la saturación que tienen sus líneas. Pueden ser muy potentes pero estar super-utilizadas. Como los proveedores no van a revelar estos detalles tal como a nosotros nos gustaría, lo más probable es que solo nos quede la posibilidad de probar la velocidad por la práctica. Si navegamos por el sitio de la empresa que estamos evaluando y comprobamos la cantidad de información (Kb) que recibimos por segundo podremos saber cómo va de rápido la página.

Otro factor que también condiciona la velocidad es la cercanía del servidor. Ten en cuenta que si estás en España, un servidor situado en tu país será más rápido para ti, en la misma condición de líneas, que uno que esté en Estados Unidos. Cuanto más lejos tenga que viajar la información, tarda más. Pero un detalle, si un usuario de Colombia visita una página que está en un servidor español, para él tu servidor será más lento que uno de su país.

Transferencia

Otro aspecto que tenemos que evaluar a la hora de contratar un espacio para nuestra web es la cantidad de megas de transferencia mensual que el proveedor permite realizar desde nuestro dominio hacia fuera. Es decir, las páginas web que manda el servidor tienen un peso en Kbytes y el proveedor los va contando, cuando pasamos el límite que le han asignando al dominio nos cobran el exceso según un precio.

Así pues, a la hora de contratar un dominio merece la pena enterarse sobre este aspecto y evaluar también las condiciones que nos ofrecen. Si encontramos un servidor que ofrece transferencia ilimitada no nos tiene que decir tampoco mucho de él, el control de la transferencia les permite ajustar su servicio a unos niveles de calidad superiores. Además, en la mayoría de los casos no nos tendrá que preocupar el superar las tasas de transferencia pues las megas que transfiere una web normal nunca superará los niveles propuestos, a no ser que se trate de un portal o una página muy visitada.

Precio

El último factor a considerar, y el más fácil de comparar, es el precio. Lo importante es comparar la calidad del servicio y el precio del mismo. Seguro que podrás realizar esta comparación sabiendo los datos que has de evaluar.

Buscador de Alojamiento

En DesarrolloWeb podemos ofreceros una gama de alojamientos de las más altas prestaciones. Desde el registro de dominios o alojamientos en Unix y NT con los recursos más avanzados, hasta servidores privados para administrar tu página y la de todos tus clientes.

Dominios y cómo registrarlos

Un dominio es una forma sencilla de identificar un ordenador en Internet de manera única, a partir del cual se encontrarán las páginas pertenecientes a la institución que lo posee.

Los dominios tienen un nombre y una terminación que indica su actividad o procedencia territorial. Por ejemplo, **yahoo.com**: **yahoo** es el nombre y **.com** expresa el ámbito de esa página, comercial en este caso.

Escoger bien el nombre del dominio es fundamental, si este tiene gancho podremos ser fácilmente recordados e identificados en Internet. A veces no es tan sencillo realizar una buena elección del nombre del dominio pues puede que los que nos gusten ya hayan sido registrados por otras personas.

En muchos casos, el registro de los dominios ha sido realizado como un auténtico pillaje, solicitando los usuarios de Internet miles de nombres con el **único objetivo de poseer ese dominio con gancho**, pensando en futuras compensaciones, y sin presentar ninguna información en sus páginas web. Los nombres de las empresas importantes son los que más han sufrido este pillaje, que ha obligado a las empresas a **comprar los dominios "raptados" por un precio astronómico** a sus dueños.

Tipos de dominos

En Internet existen varios tipos de terminaciones de dominios o, mejor dicho, **dominios de primer nivel**. Estos son los.com, .org, .es, etc.

Como decíamos, los dominios de primer nivel **indican el ámbito al que pertenecen**, hay principalmente dos grupos, **genéricos y territoriales**.

Dominios genéricos

Son dominios que se otorgan a nivel internacional, para empresas y personas de todo el mundo. Los vamos a enumerar aquí, indicando el tipo de institución al que van dirigidos.

.com Para empresas o en general para cualquier web que tenga carácter comercial. En un principio, quería decir que ese dominio que se trataba de una **compañía** estadounidense, pero en la práctica cualquiera ha tenido acceso a estos dominios que se han hecho muy populares y los preferidos para cualquier tipo de fin.

.net Indica una **red** en Internet, la de un proveedor de servicios por ejemplo. Una opción que a la larga también se ha convertido en válida para cualquier tipo de propósito.

.org Destinado para **organizaciones**, asociaciones, fundaciones y demás entidades muchas veces con fines benéficos o si ánimo de lucro.

.gov Es para las páginas del **gobierno** de los Estados Unidos.

.edu Reservado para las instituciones relativas a la **educacion**, pero solo las de los Estados Unidos.

.mil Se utiliza para instituciones **militares** de los estados unidos.

.int Que pertenece a la **Unión Internacional de Telecomunicaciones**, y en el que se pueden encontrar organismos que se hayan creado con acuerdos internacionales, como las Naciones Unidas.

> **Nota:** Actualmente, se han aprobado 7 dominios nuevos. Se pueden ver unas descripciones de estos en este artículo.

Hay un organismo llamado **NIC** que es el que se encarga de **regular el registro de los dominios** a nivel mundial. Se encarga de indicar para que se utiliza cada dominio, quien está autorizado a registrarlo, y quien puede ser el registrador. Esta entidad delega en otras para desempeñar todo el trabajo de organización que conlleva la administración de los dominios, sobretodo en los distintos países, como más adelante se verá.

Para registrar un dominio

Anteriormente, el registro de dominios sólo lo podía realizar una empresa llamada Network Solutions, que mantuvo el monopolio hasta el verano de 1999. Actualmente se encuentra liberalizado este mercado y **existen muchas más empresas registradoras de dominios**. (Ver el listado) De todos modos, a través de estas empresas operan muchos más intermediarios y encontrar un registrador cercano a nosotros puede ser muy sencillo.

Sólo los dominios del tipo **.com .net y .org** (y ahora también los .info .biz y .edu) se encuentran al alcance de cualquier persona. Para registrarlos podemos acceder a las páginas de las empresas que están capacitadas para ello. En DesarrolloWeb.com podemos ayudaros también en esta tarea.

En nuestro servicio de alojamiento puedes solicitar un dominio y registrarlo para cuanto tiempo desees. Además, en la parte de la derecha puedes ver un formulario donde escribir y buscar el dominio que deseas. Posteriormente, si es que tu dominio está libre, podrás registrarlo a través de nuestras páginas. En estas páginas debemos realizar una búsqueda para saber si se encuentra disponible el dominio que deseamos. Una vez hemos comprobado que no pertenece ya a ninguna persona podemos pasar a su registro, que se realiza a través de unos formularios online en esas mismas páginas.

Si vamos a colocar unas páginas web en el dominio y vamos a contratar los servicios de hosting en algún **proveedor**, puede ser aconsejable que este **sea el encargado de la labor de registro**. Lo hará a través de una de esas empresas capacitadas para ello (señaladas arriba) y, generalmente, sin un coste adicional, o en el caso de haberlo, no será muy elevado. (probablemente el mismo precio que tenga trasladar ese dominio a sus servidores)

Hay unos costes relacionados al registro de los dominios, costes que no se pueden evitar y que no se los queda el proveedor, sino que son destinados al NIC. Para los dominios **.com, .net y .org**, el coste del registro es de **35 dólares**. Esta tasa permite mantener el dominio durante 1 año. Posteriormente hay que pagar 35 dólares por año para seguir manteniendo el nombre.

Dominios territoriales

También existen **dominios de primer nivel que indican el territorio** de origen de la página. Estos dominios solo se le otorgan a **empresas o personas de los países relacionados** con el dominio.

Como ejemplos de dominios territoriales podemos señalar **.es** para España, **.fr** para Francia, **.mx** para México...

El registro de los dominios territoriales es regulado en base a unas normas específicas para cada país. Los encargados de crear estas normas para el registro son los distintos **delegados del NIC de cada país**. De este modo, el **ES-NIC (www.nic.es)** es el encargado en España, mientras que **MX-NIC (www.nic.mk)** es el de México, o **AR-NIC (www.nic.ar)** el de Argentina.

España, por ejemplo, las normas para el registro de un dominio .es son bastante restrictivas, y no todo el mundo puede registrarlos. En concreto para acceder a estos debemos **ser una empresa y nuestro nombre debe de ser igual al del dominio que queremos registrar**, o muy parecido. También es posible el registro si se posee una marca registrada en España con ese nombre, siempre llevado a cabo por una empresa y no por una persona física.

Para consultar los requisitos para el registro de dominios en otros países podemos **visitar las páginas de sus NIC** correspondientes, donde siempre estará todo bien explicado y con la información más actualizada.

Registrar un dominio

En este caso, **los NIC de los distintos paises son los que dicen cómo realizar esta labor, y los que realizan el registro**. De todos modos debemos apoyarnos en las mismas empresas que nos servían para el registro de dominios genéricos. También nos pueden ayudar **nuestros proveedores de hosting**, que conocerán bien las normas de registro de los países donde trabajan.

Con respecto al registro de los **.es**, que es el que conocemos, puesdes registrarlos también en por medio de nuestro servicio de registro de dominios y haremos de intermediarios para registrarlo a través de ES-NIC.

Para los dominios **.es**, el coste del registro viene a costar 72,12 € (IVA incluido). Esta tasa permite mantener el dominio durante 1 año natural (es decir, sólo hasta que se termina el año en curso). Posteriormente hay que pagar 48,08 € por año (IVA incluido) para seguir manteniendo el nombre.

Donde conseguir dominios gratis

Es dificil conseguir un dominio gratuito, además, sería un regalo envenenado, ya que la empresa que lo registra gratuitamente se reserva todos los derechos sobre los dominios, como pueden ser la propiedad, el decidir qué uso se le da o la colocación de publicidad, muchas veces abusiva.

Hace tiempo, las empresas que ofrecían estos dominios gratis eran:

- Namezero.
- DomainZero, aunque sólo para ciudadanos de Estados Unidos.

En la actualidad, estas ofertas se han acabado o son de pago (Hay que enterarse de el estado de la oferta en las páginas señaladas, pero no esperéis mucho).

Sin embargo, hay otra opción para conseguir dominios gratuitos muy interesante. Se trata de **DOT.TK**, un dominio asignado a una isla "perdida" que ofrece, para particulares, la posibilidad de registrar gratuitamente nombres de dominio con la terminación .tk.

Para los que desean dominios gratuitos, insistir en que el registro de un dominio generalmente tiene un coste, por lo que no suele ser muy habitual que los regalen. También está la posibilidad de conseguir un subdominio, que no tiene, en teoría, coste de setup, razón por la cuál son regalados muy a menudo.

> **Nota:** Un subdominio es un "dominio dentro de un dominio". Por ejemplo, usuarios.desarrolloweb.com sería un subdominio del dominio principal desarrolloweb.com. www.ciberpais.elpais.es es un subdominio del dominio principal elpais.es.

Los 7 dominios nuevos

Los dominios son una manera de identificar un ordenador en Internet, un servidor donde al que se accede, para leer una página web, entrar en un chat, etc.

Existen unas terminaciones para cada dominio que indican el ámbito del servidor donde se está accediendo. Los dos tipos de ámbito más comunes son el regional, que indica el pais del servidor, y el genérico, que lo que indica es el tipo de información que se va a encontrar.

La comisión que se encarga de gestionar la administración de dominos ICANN -Internet Corporation for Assigned Names and Numbers- ha aprobado la creación de 7 nuevos dominios genéricos durante última reunión del sigo XX.

Veamos cuáles son estos nuevos dominios:

Estos dominios de primer nivel se suman a los actuales, de los que tenemos la descripción y procedimiento de registro en otro reportaje.

La fuente desde donde la hemos recogido esta completa información es Nominalia.es

Descripción de los nuevos dominios

.biz
Este dominio es abierto (en el sentido que no hacen falta requisitos específicos previos para el registro, del tipo .com, .net y .org) y el de significado más genérico, por lo que es el que recibirá, probablemente, un volumen de registros mayor. Es la abreviación anglosajona, en pronunciación figurada de business.

.info
Este dominio es también abierto (sin requisitos específicos para registrar) pero el significado parece, a priori, un poco menos genérico que .biz o .com. De todas formas, será también un dominio de gran volumen de registros, seguramente por encima del millón de nombres ya en el primer año.

.name
Este dominio es (bastante) abierto, pero para un uso específico, de carácter personal. Está reservado a los individuos, que podrán reservar su nombre con la estructura MiNombre.MiApellido.name, ej: (pedro.garcia.name). El tercer nivel, correspondiente al nombre de pila, es exclusivo del titular, pero el segundo nivel, correspondiente a los apellidos, es compartido con todos los que ostenten dicho apellido.

En este caso, Pedro no podría impedir el registro de miguel.garcia.name, por ejemplo. Son 14 millones de nombres en 5 años los previstos por Global Name Registry.

.pro
Este dominio es para un uso específico reservado a profesionales de determinadas categorías, agrupados en subdominios: inicialmente serán .med.pro (médicos), .law.pro (abogados) y .cpa.pro (auditores; cpa significa chartered public accountant). En el tercer nivel estará el nombre del profesional en cuestión que deberá acreditar su pertenencia al colegio u organización profesional correspondiente.

.coop
Este dominio está reservado a las cooperativas. Un dominio claramente restringido en sus políticas de registro (hace falta demostrar la cualidad de cooperativa a través las organizaciones locales correspondientes). El nombre de dominio debe ser necesariamente el de la cooperativa. Este dominio tendrá un periodo de lanzamiento/test de seis meses, con procesos aún más restringidos, por lo que no va a tener un gran volumen de entradas en el primer momento.

.aero
Este dominio, también de uso restringido, es para la industria de los servicios aéreos: compañías aéreas; compañías aeronáuticas; aeropuertos y servicios aéreos. El volumen esperado por el registro es de entre 100.000 y 300.000 nombres en cuatro años.

.museum
Este dominio es de uso restringido para la comunidad de museos. La posibilidad de registrar en el segundo nivel (mnac.museum) o en el tercero (mnac.bcn.museum o mnac.sp.museum) según clasificaciones geográficas todavía está por definir definitivamente. Los promotores esperan unos 50.000 nombres en este dominio.

Enlaces de interés:
ICANN: http://www.icann.org/
Descripción de dominios y su registro

Secretos de los buscadores

Los buscadores y los directorios son herramientas programadas por seres humanos especializadas en buscar información en la Red por medio de sus robots de búsqueda, también llamados Spider o Arañas. De esta manera cualquier internauta que solicite información mediante la introducción de palabras claves o frases cortas en un buscador, obtendrá inmediatamente una lista de páginas web relacionadas con la palabra clave escogida.

Buscadores y directorios

Hay que aclarar la diferencia entre buscador y directorio, son conceptos que normalmente se confunden.

Un directorio como Yahoo depende de humanos o editores que indexan cada URL manualmente, siguiendo su propio criterio de valoración y colocando cada URL en la categoría y subcategoría adecuadas. Por ello el alta en un directorio es un proceso lento y puede llegar a tardar hasta 2 meses.

Una buena página con un buen contenido tiene más posibilidades de ser indexada en un directorio con un ranking alto que una página pobre, sin información, que incluso puede ser rechazada. Los directorios suelen ser muy selectivos a la hora de indexar nuevas páginas. Prefieren calidad y no cantidad de páginas web.

Ejemplos de grandes directorios son Yahoo, Terra y DMoz.

Envía a los directorios solamente tu página principal, redactando un buen título y descripción pensado de antemano a conciencia, utilizando las palabras claves más relevantes y repitiéndolas con frecuencia. El texto que introduzcas al hacer el alta será el que el editor utilizará a la hora de catalogar tu página en la categoría correspondiente.

Los buscadores como Altavista o Excite utilizan un proceso totalmente diferente para indexar las páginas web. Crean sus listados automáticamente por medio de los motores de búsqueda.

El Spider visita tu web, la lee y sigue todos los links a otras subpáginas que vaya encontrando. El Spider volverá a visitar tu página en un mes o dos y si has hecho cambios importantes en la página, el buscador reconoce estos cambios que pueden afectar el posicionamiento de tu web.

Las Metatags son también un elemento a tener en cuenta en un buscador a la hora de extraer la

información de la página

(no todos los buscadores soportan Metatags)

En un buscador se pueden destacar tres partes: El Spider que visita las URL, el índice o catálogo que contiene una copia de cada url que ha sido encontrada por el Spider y el software.

Este es el programa que filtra la información grabada en el índice o catálogo, extrae las búsquedas y posiciona cada URL siguiendo un orden o criterio.

Todos los buscadores contienen estas tres partes básicas pero ninguno de ellos funciona igual. Por eso una búsqueda por una palabra clave en particular en Altavista no produce los mismos resultados que la misma palabra clave en Excite. Estos buscadores siguen criterios diferentes a la hora de indexar las páginas web. Si tu página web tiene predeterminado el criterio que sigue el buscador para indexar las URLs puedes alcanzar unos de los primeros puestos en el buscador. Es fundamental tener una buena organización HTML y saber cómo funcionan los buscadores en general para lograr tener un buen posicionamiento en el buscador.

Factores que afectan al ranking de tu URL

- Utilizando tu propio dominio tendrás muchas más posibilidades de obtener una buena posición y tu página estará más valorada. Si tienes posibilidades nu dudes en invertir en tu propio dominio.

- El título es el primer criterio que valora un buscador. El título es imprescindible y recuerda colocar tu palabra clave más relevante en él.

- Tus palabras claves deberían ser frases cortas. Pon tus palabras claves en el título y descripción con frecuencia.

- Utiliza Metatags para que tu página sea correctamente indexada por los buscadores. Las MetaTags comunes son el título, descripción y palabras claves. Una buena combinación de texto en el título y descripción y la elaboración de Metatags es muy recomendable para mejorar el posicionamiento en buscadores que soportan Meta Tags

- No recomendamos la utilización de Frames, pero si lo haces deberás incluir los Metatags en la página que distribuye los frames porque será la única información disponible para indexar. Puedes incluir en el NOFRAMES un párrafo descriptivo del contenido del sitio, así como links a las páginas interiores para facilitar el recorrido del robot.

- Utiliza la MetaTag Robots.txt para dar instrucciones al Spider que recorra los links encontrados en tu Sitio Web e indexe todas las demás páginas. Asegúrate que no hay ningún enlace muerto en la página.

<META name="robots" content="All">

- Utiliza la Metatag Revisit para indicar al Spider cuándo debe volver a visitar tu página web e indexar los cambios hechos de manera que tu página web esté siempre actualizada y activa en el buscador.

<META NAME="revisit" CONTENT="15 days">
(también 30 days)

- Otra Tag útil es el ALT text que va asociado a las imagénes o gráficos, este texto forma parte de la página, así que utiliza tus palabras claves más relevantes en este texto.

ALT="myrasoft ofrece software de promocion web"

- Incrementa la popularidad de tu página web mediante la colocación de links a otras páginas. La popularidad se mide por el número de enlaces a una determinada página.

Puedes medir la popularidad de tu página y la de tu competencia gratuitamente en: http://www.linkpopularity.com.

La mayoría de los buscadores analizan cuantos links hay hacia tu web, valoran tu popularidad e incrementan el ranking. Además cuantos más links tengas mayores son las posibilidades de que tu página sea visitadas por los Spiders. Puedes negociar links de calidad con otras páginas web que estén relacionadas con el tema de tu sitio web y que contengan palabras claves semejantes, a la larga generarán un tráfico importante de nuevos visitantes y clientes potenciales.

- Haz un Mapa de tu Sitio Web con links a todas las páginas interiores. Puedes enviar esta página a los buscadores, esta es una buena táctica para ser localizado en los buscadores. Un ejemplo sería:

http://www.myrasoft.com/websiteindexsp.htm

- No hagas Spam. Spam es enviar tu página una y otra vez al mismo buscador diariamente o semanalmente. Los buscadores tienen limitaciones y penalizan a las páginas que hacen Spam, borrándolas de su índice. Altavista e Infoseek solamente aceptan una página al día por dominio. Yahoo solamente la página principal. Puedes enviar diferentes URLs en días consecutivos, envía tan sólo las más importantes. Otro tipo de Spam es utilizar texto invisible en letra pequeña del mismo color que el fondo de la página, con la misma palabra clave una y otra vez. Los buscadores penalizan este tipo de diseño, obtendrás un ranking bajo o tu página no será indexada. No utilices palabras claves que no estén relacionadas con el contenido de tu página y no las repitas en exceso. Puedes repetirlas una o dos veces, pero intercalándolas.

- Envía tu página una vez al mes o siempre que hayas hecho cambios importantes. Una vez que has enviado tu URL hay que ser paciente, los grandes buscadores indexarán tu página en un promedio de dos a ocho semanas. No todos envían un email de confirmación de que tu página ha sido añadida satisfactoriamente, deberás dejar pasar un tiempo y hacer una búsqueda por dominio o por palabra clave. Una aproximación del tiempo que tardan los grandes buscadores que tienen enlaces en castellano en indexar las URLs:

- Altavista, es el más rápido de 2-3 días a dos semanas.
- Excite de 2-3 semanas
- Lycos de 4 a 6 semanas
- Terra es prácticamente inmediato, de 2 a tres días

La mayoría de los buscadores de lengua castellana son directorios. Terra es un ejemplo. Sigue las recomendaciones descritas para los directorios cuando vayas a dar de alta tu URL. Asegúrate que eliges la categoría adecuada al hacer el alta, intenta ser lo más preciso posible. Parece una tontería pero hay muchas páginas que finalmente no se indexan o que no son indexadas correctamente porque no siguieron las instrucciones adecuadamente al realizar el alta, cometiendo errores.

Hay muchos buscadores que están incorporando una opción geográfica. Si a la hora de dar de alta tu Sitio Web eliges tu región el alta será mucho más rápida de validar y obtendrás mejores resultados. Te recuerdo que el contenido del texto de tu web es fundamental.

El término Portal no es lo mismo que un buscador o directorio, estos términos se confunden porque la mayoría de los portales tienen incorporado un buscador. Los Portales incorporan información adicional como noticias, foros, chats...etc.

Espero que esta información te sirva de ayuda en tu estrategia de márketing en Internet.

Registro en buscadores

Una vez tienes realizada tu web el objetivo es que sea visitada. Existen muchas maneras de conseguir esto, una de las más fáciles e inmediatas es el registro en buscadores.

Para conseguir que tu web sea incluida en un buscador debes rellenar un formulario con los datos de la web que deseas registrar. Cada buscador tiene un formulario especifico y a menudo el proceso de registro es diferente.

Para encontrar el formulario lo más fácil es que entres en la página principal del buscador y busques un enlace que ponga **Añadir página**, **Add URL**, **nueva dirección**, o algo parecido. Ese enlace te llevara al formulario de registro o, en su defecto, a la página donde explica cómo has de registrarte en ese buscador en concreto.

Aun así, existen unos procedimientos básicos de registro que suelen ser repetirse en los buscadores:

- **Para los índices**, los buscadores que tienen categorías y podemos encontrar páginas clasificadas dependiendo de su temática. Se suele navegar a la categoría en la que se desea incluir el web. Allí se busca el botón "añadir página"(o algo parecido) y se encuentra el formulario para el registro. Indices son Yahoo! o Terra
- **Para los motores de búsqueda**, que son los buscadores que no tienen porque mantener un índice y que tienen robots que constantemente recorren Internet en busca de nuevas páginas para incluirlas en el buscador. Estos buscadores suelen tener un formulario accesible desde la página inicial, con el enlace correspondiente. No hay que navegar las categorías para acceder al formulario. Motores de búsqueda típicos son Altavista o Sol.es.

Salta a la vista que registrarte en varios buscadores puede ser una tarea un poco costosa y otro tanto pesada, pues tienes que ir buscador a buscador introduciendo una y otra vez los mismos datos.

Afortunadamente existen herramientas multiregistro. Estas herramientas nos permiten registrar nuestra página en distintos buscadores introduciendo una sola vez los datos de la página web.

Existen también dos tipos de herramientas multiregistro:

- **Aplicaciones windows**, como cualquier otro programa, pero que su objetivo es registrar en buscadores. Este tipo de herramientas suelen tener que comprarse, puedes encontrarlas shareware en www.tucows.com por ejemplo, pero suelen estar limitadas en su uso.
- **Herramientas online**, estas nos permiten desde Internet, y por lo general gratuitamente, registrarnos en varios buscadores. No suelen ser tan potentes como las anteriores, pero si más útiles y accesibles por ser gratuitas. En el **buscador de desarrolloweb.com, en la sección de promoción** puedes encontrar alguna de estas herramientas. Nosotros estamos preparando una **herramienta multiregistro online**, que ya se encuentra operativa y creciendo en el numero de buscadores contemplados.

En desarrolloweb.com disponemos de una **sección dedicada a la promoción de páginas web**. Amplia en muchisimo la información recogida en este informe y recoje unas herramientas para facilitarte la labor de que tu web sea más vistada.

Contacto con navegante

Bien sabido es que una de las tareas más importantes y laboriosas del ciclo de vida de una página web es su mantenimiento, para conservar la web, su tráfico y su importancia con el tiempo. Pero mantener una web no es sólo publicar nuevos contenidos y actualizar los anteriores, entre estas tareas se encuentra una que es fundamental: proporcionar mecanismos para que los visitantes se puedan poner en contacto contigo, y, por supuesto, contestar a cada uno de los mensajes que recibes.
Existen muchas formas de comunicarnos con el navegante, vamos a ver los mecanismos que están más a nuestro alcance para que a ninguna de nuestras webs les falte la posibilidad de comunicarse con sus visitantes.

Uno de los errores graves en una web es que no se brinde la oportunidad de contactar con el creador o responsable de contenidos, pero es aun peor que los mensajes no se respondan pues crea una gran sensación de vacio

Hemos ordenado los siguientes puntos por orden de facilidad en su uso, así los primeros son más asequibles y los siguientes más difíciles de conseguir, aunque a menudo mejores.

Correo electrónico

Lo que nunca debe faltar en una página, por su sencillez y utilidad, por que es la forma más adecuada en muchos casos... Siempre tenemos que incluir una dirección de correo, fácilmente localizable, para que los visitantes puedan comunicarse. Es facilísimo hacer un enlace con una dirección de correo:

Se ha de poner un enlace convencional, pero su atributo **HREF va direccionado** a una dirección de **correo** con la palabra **mailto:**.

```
<A HREF="mailto:eugim@desarrolloweb.com">Escríbeme</A>
```

Quedaría así: Escríbeme

Poner un enlace a una dirección de correo en un editor de HTML, para los que trabajáis con herramientas de edicion, también es pareceido a como lo hacéis para los enlaces normales, pero con esas diferencias.

Formulario de contacto

Podemos utilizar un formulario para comunicarnos. En el formulario nuestro visitante puede introducir sus datos y la consulta o sugerencia que desea realizar, para, pulsando luego un botón, enviar por correo electrónico a la persona de contacto todos los datos.

Las ventajas de un formulario con respecto a una dirección de correo pueden ser:

- Ayudar al visitante a componer el mensaje, y así incentivar su comunicación.
- Forzar a que el visitante introduzca cierta información que te pueda resultar importante.
- Ofrecer más mecanismos de comunicacion, cada uno elegirá el que más le convenga.

Veamos brevemente ahora cómo construir un formulario en una página web.

Un formulario se coloca entre las etiquetas <form> y </form>. A esta etiqueta le tenemos que incluir varios atributos, estos son:

- ACTION="mailto:eugim@desarrolloweb.com" Para indicarle a qué dirección de correo enviar los resultados del formulario.
- METHOD="post" Para que lo envie por metodo post, esencial para enviarlo por e-mail.
- ENCTYPE="text/plain" Por que lo que vamos a enviar es texto.

En resumen, la etiqueta del formulario quedaría así:

```
<FORM ACTION="mailto:eugim@desarrolloweb.com"
        METHOD="post"
        ENCTYPE="text/plain">
<!--  CAMPOS DEL FORMULARIO  -->
</FORM>
```

Ahora veamos cómo colocar campos en el formulario, por lo menos los más fáciles:

Campos de texto
Se utiliza la etiqueta <INPUT> de esta manera: **<INPUT TYPE="text" NAME="nombre_del_campo" SIZE=10>** El atributo TYPE indica que es un campo de texto. NAME es el nombre del campo, tiene que describir lo que hay dentro. SIZE te permite ajustar el tamaño del campo. La etiqueta NO tiene cierre con </INPUT>
Areas de texto
Se utiliza la etiqueta <TEXTAREA> de esta manera: **<TEXTAREA NAME="nombre_del_campo" COLS="20" ROWS="4>** **Texto inicial dentrodel textarea< BR></TEXTAREA>** El atributo NAME es el nombre del campo. COLS indica el número de columnas del área de texto y ROWS en de filas.
Botón de envío
Se utiliza la etiqueta <INPUT> de esta manera: **<INPUT TYPE="submit" VALUE="Envíalo YA!">** El atributo TYPE indica que es un botón submit (de envio). VALUE indica lo que va escrito dentro del botón. La etiqueta NO tiene cierre con </INPUT>, NO hace falta darle un nombre con NAME

No te olvides colocar antes de </FORM> el botón de envío y ya tendrás el formulario listo para que tus navegantes te manden sus ruegos y preguntas.

Puedes ver un ejemplo de formulario aquí.

Referencia: Tenemos una serie de capítulos en el manual de HTML que tratan sobre la creación de formularios de manera detalladísima.

Tenemos también una serie de consejos para hacer formularios más rápidos, fáciles y agradables.

Lista de correo

Una lista de correo es una dirección de e-mail a la que si enviamos un correo nos lo hace llegar a todos los integrantes de una lista de direcciones.

Con un ejemplo estará más claro. Si todos los integrantes de una lista, cuando quieren conversar entre ellos, en vez de mandar un correo electrónico a todos ellos lo mandan a un servidor y este se encarga de que le llegue el mensaje a todos los de la lista, este servidor era una lista de correo ;-)

Las listas de correo pueden ser muy útiles para que los usuarios que llegen a las páginas se apunten y nosotros, como administradores del sitio web mandaremos correos electrónicos periodicamente a las lista para mantenerlos informados de noticias, cambios en el web y todo aquello que queramos enviarles.

Para tener una lista de correo en tu página puedes contratar el servicio con un proveedor, pero es mucho más fácil y económico usar una lista de correo de las que regalan en varios servidores. Para

montarla en tu web, simplemente has de seguir las instrucciones que te brinde el proveedor. Puedes conocer varios enlaces a listas de correo para páginas web en nuestro buscador.

Libro de visitas

Ahora veremos algún mecanismo adicional para comunicarse con el cliente, pero en estos casos no se trata tanto de una comunición entre ellos y tu, sino más es una forma de comunición te todos para todos.

En el caso del libro de visitas está bien claro, es una herramienta donde los visitantes pueden dejar los mensajes que deseen para que estos queden reflejados en la web y así, no sólo los leas tu, sino que también los pueden leer todos los dem´sa usuarios de tus páginas. Esto le da agilidad a la página, dinamismo y hace que los visitantes se sientan integrados en el proyecto, colaboren y vuelvan para ver sus "huellas". Las ventajas son muchas.

En la mayoría de los casos no dispondremos de la tecnología ni conocimientos para implementar esta tecnología, pero existen en el mercado varios libros de visitas personalizabes e integrables dentro de webs sencillas, es decir, tu no tienes que hacer nada, ellos te proporcionan todos los recursos y explicaciones para montar un libro de visitas en tu web. Aquí puyedes ver varios enlaces a estos sitios:

- Melody Soft: http://www.melodysoft.com/ (En castellano)
- Cambia.net: http://libros.cambia.net (En castellano)
- Otros enlaces a libros de visitas en nuestro buscador

Forums de discusión

En algún caso podemos pasar a una opción más avanzada de lo que nos ofrecia libro de visitas. Estos son los Forums de discusión, donde la gente puede opinar sobre temas y otros navegantes contestarles, etc. Es muy útil e interesante. No cabe ya destacar sus múltiples ventajas y hits que recibiremos si se tratan temas de interés.

Si no tenemos la infraestructura suficiente para montar nosotros el forum, podemos utilizar los servicios, muchas veces gratuitos, de otros servidores.

Aquí se pueden seguir varios enlaces que nos llevarán a soluciones para implementar estos forums en tu página.

- Melody Soft: http://www.melodysoft.com/ (En castellano)
- BoardHots: http://www.boardhost.com/
- Otros enlaces a foros en nuestro buscador

Conclusión

Hemos visto un mostón de ideas para incluir en nuestras páginas, todas ellas con el objetivo de establecer contacto con el visitante, aunque al final se nos ha ido el tema del planteamiento inicial y hemos incluido mucho más que meras herramientas de marketing. Porque no olvides que todo esto es para hacerte conocer y hacer que los visitantes te recuerden con alegría y a menudo.

Permíteme por último ofrecerte un enlace, corresponde con la sección de elementos para incluir en páginas web de nuestro buscador, donde están todos los enlaces enumerados antes y donde irán apareciendo nuevos.

Caracteres especiales

Una página web se ha de ver en paises distintos, que usan conjuntos de caracteres distintos. El lenguaje HTML nos ofrece un mecanismo por el que podemos estar seguros que una serie de caracteres raros se van a ver bien en todos los ordenadores del mundo, independientemente de su juego de caracteres.

Este conjunto son los caracteres especiales. Cuando queremos poner uno de estos caracteres en una página, debemos sustituirlo por su códico.

Por ejemplo, la "**á**" (a minúscula acentuada) se escribe "**á**" de modo que la palabra página se escribiría en una página HTML de este modo: p&aamp;aacute;gina

Caracteres especiales básicos

En realidad estos caracteres se usan en HTML para no confundir un principio o final de etiqueta, unas comillas o un & con su correspondiente caracter.

<	<	>	>
&	&	"	"

Caracteres especiales del HTML 2.0

Á	Á	À	À
É	É	È	È
Í	Í	Ì	Ì
Ó	Ó	Ò	Ò
Ú	Ú	Ù	Ù
á	á	à	à
é	é	è	è
í	í	ì	ì
ó	ó	ò	ò
ú	ú	ù	ù
Ä	Ä	Â	Â
Ë	Ë	Ê	Ê
Ï	Ï	Î	Î
Ö	Ö	Ô	Ô
Ü	Ü	Û	Û
ä	ä	â	â
ë	ë	ê	ê
ï	ï	î	î
ö	ö	ô	ô
ü	ü	û	û
Ã	Ã	å	å
Ñ	Ñ	Å	Å
Õ	Õ	Ç	Ç
ã	ã	ç	ç
ñ	ñ	Ý	Ý
õ	õ	ý	ý
Ø	Ø	ÿ	ÿ
ø	ø	Þ	Þ
Ð	Ð	þ	þ
ð	ð	Æ	Æ
ß	ß	æ	æ

Caracteres especiales del HTML 3.2

¼	¼		
½	½	¡	¡
¾	¾	£	£
©	©	¥	¥
®	®	§	§
ª	ª	¤	¤
²	²	¦	¦
³	³	«	«
¹	¹	¬	¬
¯	¯		
µ	µ	º	º
¶	¶	´	´
·	·	¨	¨
°	°	±	±
¸	¸	»	»
¿	¿		

Otros caracteres especiales

×	×	¢	¢
÷	÷	€	€
“	“	™	™
”	”	‰	‰
Œ	Œ	ƒ	*ƒ*
‡	‡	†	†

Usabilidad para PDAs

Una guía para el correcto diseño de sitios webs accesibles desde PDAs. (Personal Digital Assitants)

1. **Ser consciente de las limitaciones de los PDAs.**
   Los ordenadores de bolsillo tienen una pantalla más reducida, menos memoria y menos velocidad de proceso que los ordenadores convencionales.

2. **Definir cuidadosamente la estructura del site.**
   Con las secciones más importantes y enlaces a las mismas desde todas las páginas. La primera página ya ha de mostrar información útil para el usuario, evitando páginas de bienvenida o de selección de idioma.

3. **Evitar el uso de tablas.**
   En la mayoría de los casos basta con dividir el texto con saltos de línea y párrafos. El procesado de las tablas ralentiza la velocidad de carga de la página. Si se utilizan, especificar las dimensiones en porcentajes y sin sobrepasar los 150 pixels de ancho.

4. **No utilizar marcos (frames).**
   Si ya de por sí se ha de intentar evitar su uso en el Internet convencional, con más razón en el Internet móvil: la mayoría de los navegadores no los soportan y restan mucho espacio en la pantalla.

5. **Publicar contenidos concisos.**
   Por la limitación de la memoria y de la pantalla, escoger sólo la información más importante y esencial.

6. **Organizar la información cuidadosamente.**
   Minimizando la longitud del texto (máximo 3 pantallas de longitud) y escogiendo una distribución óptima de los links de navegación (enlaces a otras secciones). Si el texto debe ser extenso, incluir enlaces a distintas partes del mismo para mejorar la navegación.

7. **Optimizar los gráficos.**
   - En dimensión: las pantallas de dispositivos Palm OS tienen un tamaño de 150x150, y los Windows CE/Pocket PC de unos 240x320.
   - **En tamaño**: escogiendo el número de colores visibles en el PDA. (16 y 8 bits de color; 16, 4 y 2 escalas de grises).

8. **Incluir texto alternativo en todas las imágenes.**
   Con el tag *alt* ofrecemos información a los usuarios que han deshabilitado la carga de imágenes en su navegador.

9. **Añadir el tag *handheldfriendly* al comienzo de todas las páginas.**
   AvantGo sabrá que la página está optimizada para PDAs.
   <meta name="HandheldFriendly" content="true"> .

10. **Utilizar convenientemente el *caching* de las páginas.**
    Para aumentar la velocidad de carga de las páginas. No guardar en caché las páginas que se actualizan todos los días (página de noticias), y guardar en cache las páginas que se actualizan raramente (página de créditos).

Mp3 vs. Servidores gratuitos

En Internet hay muchas formas por las que los archivos mp3 se han ido difundiendo: software como Napster, Gnutella, Win Mp3 Locator... también a través de buscadores de mp3 como audiofind.com o audiogalaxy.com, pero el mayor medio de difusión es -siempre lo ha sido- las páginas personales de los cibernavegantes.

La red se ha plagado de páginas personales que contienen mp3, y ésto ha generado un gran conflicto con los servidores, al ser ellos en parte "responsables" por alojar archivos de este tipo.

Qué hacen los servidores gratuitos

Al principio uno de los únicos servidores gratuitos en donde se encontraban los mp3 era Geocties.com, pero ahora casi no quedan empresas de servicios en Internet que no ofrezcan hosting gratuito, por lo que uno se puede encontrar con millones y millones de archivos de audio por toda la red. Ésto ha despertado a los servidores como Xoom.com, Demasiado.com, Tripod.com... a hacer un boycot a los uploaders.

Entonces por ejemplo cuando en nuestro servidor FTP tratamos de uploadear un archivo con extensión mp3 a una cuenta de Xoom no salta un ventana diciéndonos que ese tipo de archivos no se permiten en su servidor. U otro caso peor es el de Geocities, que nos permite subirlo, pero a la hora de haberlo hecho nos anulan la cuenta, entonces el que no lo sabe tal vez uploadea 10 temas en mp3 y en una hora se los borran. No es que esté del lado de los uploaders, pero podrían avisar, no?

Uploaders son usuarios de Internet que suben contenidos a los servidores, en este caso canciones MP3.

Qué hacen los uploaders

Pero los uploaders no se quedan atrás e idearon dos formas de pasar por alto estos problemas: el primer método fue comprimiendo el archivo a formato zip, rar o hqx (con el WinZip), cambiándole entonces la extensión de mp3 a esas extensiones, "engañando" a los servidores.

El otro método, tal vez más difícil, es renombrando el archivo a otra extensión (por ejemplo class, bin, pl...), produciendo el mismo efecto que el método anterior.

Conclusión

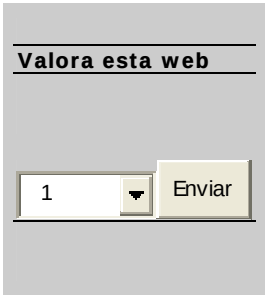
Pero la guerra continua, pues algunos servidores realizan inspecciones mensuales en las que detectan archivos comprimidos que poseen mp3 o archivos que pueden ser renombrados a mp3. La vulnerabilidad de los servidores cada vez es más poca, pero igual los uploaders siempre seguiran protagoizando su

papel de pícaros usuarios de la red a los que no hay barrera que los detenga.

Valorar una web

El objetivo de este taller de HTML consiste en colocar un pequeño formulario para que las personas que visitan nuestro web puedan valorarlo rápidamente. Se trata de un ejemplo de recurso sencillísimo que se puede obtener con sólo utilizar un poco de HTML. La sencillez es belleza y compatibilidad con los distintos navegadores, así que veamos sin más el efecto que pretendemos conseguir.

Como se puede ver a la derecha, el formulario nos propone que valoremos la página y al lado tenemos un botón para mandar la dicha valoración al webmaster del sitio.



Nota: este ejemplo sólo funciona si el visitante tiene configurado el correo electrónico en su ordenador.

La valoración se enviará por correo electrónico a la persona que se indique en el formulario, tal como veremos más adelante. Si un usuario no tiene correo electrónico configurado en su ordenador con el Outlook Express, Netscape Composer, Eudora o similar el mensaje no se podrá enviar.

Puede darse el caso de que el mensaje se cree, pero no se enviará si no tenemos email... así que probablemente lo podamos encontrar en la bandeja de salida del programa de correo.

Para crear un sistema que no funcione a través del correo electrónico del visitante y conseguir así que incluso los que no tienen correo puedan valorarte, necesitaremos utilizar algún recurso avanzado como CGI, ASP o PHP, pero eso es otro tema.

No pretendo explicar en este artículo los formularios en HTML, ya que en DesarrolloWeb.com hay descripciones suficientes sobre su funcionamiento. Para el que no conozca los formularios recomiendo la lectura varios artículos en el manual de HTML.

El código de este ejemplo es tan sencillo como este:

```
<form action="mailto:xxx@tudominio.es" method="post" enctype="text/plain">
  Valora esta web<br>
   <select name="Valoracion">
     <option>1</option>
     <option>2</option>
     <option>3</option>
     <option>4</option>
     <option>5</option>
     <option>6</option>
     <option>7</option>
     <option>8</option>
     <option>9</option>
     <option>10</option>
   </select>
   <input type="submit" value="Enviar">
</form>
```

Se puede copiar y pegar en la página que lo desees. La única línea que habría que modificar es la de la etiqueta <FORM>, donde tenemos que cambiar la dirección de email del atributo action por la dirección donde queramos que llegue el correo con la valoración.

Si quisiésemos que el correo le llegase a yo@midominio.com pondríamos así nuestro atributo action:

action="malito:yo@midominio.com"

Frames sin bordes

Este taller de HTML explica cómo realizar una declaración de frames sin bordes. Para los lectores que no conozcan lo que son los frames o como se definen sería necesario que estudiasen el manual de HTML, por lo menos los capítulos dedicados a frames.

Hemos visto que hay muchos atributos que sirven para eliminar los bordes de los marcos, tal vez demasiados y haya quedado poco claro cual sería la forma exacta de eliminar todos los frames de una vez. Resulta que no se hace igual en todos los navegadores, aunque incluyendo los atributos adecuados para cada navegador estaremos seguros que los bordes no se verán nunca.

En Netscape, simplemente necesitamos especificar el atributo border="0" en el primer frameset. Esta opción también funciona en las versiones más modernas de Internet Explorer.

En Internet Explorer, debemos especificar dos atributos también en el primer frameset. frameborder="0" y framespacing="0".

Si colocamos los tres atributos a la vez en el primer frameset estaremos seguros que no hay bordes, por lo menos en los navegadores más habituales.

La etiqueta frameset con los tres atributos quedaría así:

```
<frameset cols="90,*" border="0" frameborder="0" y framespacing="0">
```

Ejemplo práctico

Vamos a escribir por completo una declaración de frames sin bordes, para que quede todo suficientemente claro.

```
<html>
<head>
   <title>Definición de Frames</title>
</head>
<frameset cols="200,*" border="0" frameborder="0" y framespacing="0">
   <frameset rows="170,*">
      <frame src="pagina1.html">
      <frame src="pagina2.html">
   </frameset>
   <frame src="pagina3.html">
</frameset>
</html>
```

Se puede ver la página de frames sin bordes en este enlace.

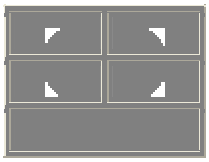
Tabla con las esquinas redondeadas

En este taller de HTML vamos a crear una tabla con las esquinas redondeadas, que nos podrá servir para destacar alguna información en el texto o crear una barra de enlaces lateral un poco más vistosa. A la derecha aparece una tabla como la que pretendemos conseguir.

Tabla con esquinas redondeadas

Cuerpo de la tabla. Podemos colocar el texto que deseemos, que la tabla crecerá en altura lo suficiente para contenerlo.

El ejemplo no resulta nada complejo. Simplemente se crea una tabla normal, en la que colocamos en cada una de sus esquinas un gráfico que hace la forma redondeada. Los gráficos que utilizamos en esta ocasión tienen una parte de color y otra transparente. La parte de color es la que dibuja el borde redondeado y la parte transparente deja ver el color de fondo que hayamos colocado en la tabla. Las imágenes se pueden ver a continuación. Para guardarlas utiliza el botón derecho del ratón encima de la imagen y selecciona la opción que pone "Guardar imagen como..." o algo parecido. También podrás descargar las imágenes y el ejemplo completo en un archivo comprimido.



En nuestro ejemplo hemos creado imágenes que tienen la parte no transparente de color blanco, que corresponde con el color de fondo de la página donde queremos colocar la tabla. Si queremos colocar

una tabla como esta sobre un fondo distinto al blanco deberíamos crear unos gráficos que tengan el mismo color que el fondo, en lugar de blanco.

Lo bueno de que el otro color utilizado en la imagen sea transparente es que la tabla que creamos puede tener el color de fondo que se desee. Esta otra tabla -a la derecha- se crea con las mismas imágenes del ejemplo y, como se puede ver, tiene otro color de fondo que la anterior.

Tabla con esquinas redondeadas

Esta tabla tiene otro color de fondo, pero está creada con las mismas imágenes que la tabla anterior.

Creación de la tabla

Ahora vamos a estudiar el código HTML que hace falta para crear esta tabla con esquinas redondeadas. Probablemente con otro código HTML más simple también se podría construir, pero hemos preferido añadirle un pequeño exceso de atributos y etiquetas que servirá para estar seguros de que se puede visualizar correctamente en todos los navegadores.

La tabla que utilizamos contiene varias celdas dispuestas en tres filas y tres columnas. En las celdas de las esquinas es donde colocamos las imágenes que hacen que los bordes aparezcan redondeados. En el resto de celdas de la tabla que forman el borde, para asegurarnos de que tienen el tamaño correcto, colocamos imágenes de un píxel transparente con sus correspondientes atributos de anchura y altura modificados a lo que necesitamos. En la celda del centro es donde colocamos el cuerpo de la tabla, con todo el texto que queremos que vaya dentro, sus imágenes, etc.

```
<table width=300 cellspacing=0 cellpadding=0 bgcolor="#333399" border=0>
<tr>
<td width=11><img src="sup-izq.gif" width=11 height=11></td>
<td width=278><img src="pixeltrans.gif" width=278 height=1></td>
<td width=11 align=right><img src="sup-der.gif" width=11 height=11></td>
</tr>
<tr>
<td><img src="pixeltrans.gif" width=1 height=1></td>
<td><font color="#ffffff" face="verdana,arial,helvetica" size=2>
<b>Tabla guay</b>
<br>
<br>
Este es el texto que quieras ponerle a la tabla. Puedes poner tanto texto como desees, que la tabla se hará lo suficientemente grande como para que quepa todo.

</font></td>
<td><img src="pixeltrans.gif" width=1 height=1></td>
</tr>
<tr>
<td width=11><img src="inf-izq.gif" width=11 height=11></td>
<td width=278><img src="pixeltrans.gif" width=278 height=1></td>
<td width=11 align=right><img src="inf-der.gif" width=11 height=11></td>
</tr>
</table>
```

Si alguno desea utilizar este código para crear sus propias tablas únicamente debería modificar unos pocos datos:

- Texto del cuerpo de la tabla.
- En caso de que se desee modificar el ancho de la tabla
  + Tamaño de la tabla. Atributo width de la etiqueta <table>
  + Tamaño de los píxeles transparentes, en la primera y última fila. Atributo width de las etiquetas <img> de los pixels transparentes.

Los anchos de los pixels transparentes (en la primera y última fila de la tabla) tienen que ser el ancho de la tabla menos el ancho de las dos imágenes que aparecen en las esquinas. En nuestro código, como el ancho de la tabla es de 300 píxel y el ancho de las dos imágenes de los bordes es de 11 píxel, el ancho de la imagen de píxel transparente será 300 - 11 x 2 = 300 - 22 = 278
Para acabar, recordamos que se pueden descargar las imágenes, así como el código fuente de la tabla con esquinas redondeadas, en un archivo comprimido.

Tabla con las esquinas redondeadas, tipo 2

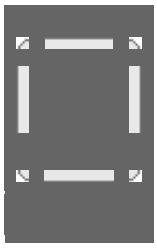
Cuerpo de la tabla

Vamos a ver como realizar con HTML una tabla con las esquinas redondeadas y con un pequeño marco de un píxel. Es un ejemplo de tabla con los bordes redondeados como cualquier otro, de hecho, ya hemos visto un ejemplo sobre este asunto en nuestro anterior artículo Tabla con esquinas redondeadas. Como siempre, lo mejor para darse cuenta de lo que pretendemos construir es verlo en un ejemplo y al lado de estas mismas líneas podemos verlo.

En este caso utilizaremos las siguientes imágenes, que podemos guardar pulsando sobre ellas con el botón derecho del ratón y seleccionando "Guardar imagen como...". También podemos descargar todo el código y las imágenes en un archivo comprimido.

Aquí podremos escribir lo que deseemos, que la tabla crecerá lo suficiente para contener todo el texto que coloquemos, incluso imágenes.

Espero que parezca un diseño interesante, aunque seguro que los hay mejores...



Creación de la tabla

Vamos a ver el código HTML necesario para crear este ejemplo. Hemos de tener en cuenta que el código se podría haber creado de varias maneras, aunque nosotros presentamos la que consideramos más versátil.

Hemos construido la tabla con las correspondientes etiquetas de tablas de HTML. Como consideración cabe señalar que las etiquetas de las imágenes que se colocan dentro de las celdas tienen que estar pegadas a la etiqueta </TD>, que se utiliza para cerrar la tabla. Si no es así puede que nuestro ejemplo quede descuadrado.

El tamaño de la tabla en anchura se puede definir perfectamente en el atributo width de la etiqueta <TABLE>. La altura será la suficiente para que quepan todos los contenidos de la tabla.

Este es el código en cuestión:

```
<TABLE WIDTH=300 BORDER=0 CELLPADDING=0 CELLSPACING=0>
<TR>
<TD>
<IMG SRC="images/tablita_01.gif" WIDTH=6 HEIGHT=6></TD>
<TD background="images/tablita_02.gif">
<IMG SRC="espacio.gif" WIDTH=1 HEIGHT=6></TD>
<TD>
<IMG SRC="images/tablita_03.gif" WIDTH=6 HEIGHT=6></TD>
</TR>
<TR>
<TD background="images/tablita_04.gif">
<IMG SRC="espacio.gif" WIDTH=6 HEIGHT=1></TD>
<TD bgcolor=E8E8E8 valign=top>
Cuerpo esto es el Cuerpo esto es el Cuerpo esto es el Cuerpo...
</TD>
<TD background="images/tablita_06.gif">
<IMG SRC="espacio.gif" WIDTH=6 HEIGHT=1></TD>
</TR>
<TR>
<TD>
<IMG SRC="images/tablita_07.gif" WIDTH=6 HEIGHT=6></TD>
<TD align=center background="images/tablita_08.gif">
<IMG SRC="espacio.gif" WIDTH=1 HEIGHT=6></TD>
<TD>
<IMG SRC="images/tablita_09.gif" WIDTH=6 HEIGHT=6></TD>
</TR>
</TABLE>
```

No hay mucho que explicar... simplemente que recojas las imágenes descargando el archivo comprimido (a notar que han sido colocadas en un directorio llamado images y que si no están allí no funcionará el ejemplo) y que la imagen que se llama espacio.gif es simplemente un píxel transparente.

Algunos consejos para webmasters

Algunos consejos para webmasters distribuídos por el autor del artículo en la lista de correo de ayuda de DesarrolloWeb.com.

1.- JAMAS programen directo sobre la PC, diagramen todo lo que vayan a hacer, en terminos informaticos se llama "algoritmizar", y a nivel empresarial se llama "trabajo de mesa", ese es el verdadero corazon del trabajo....nuncan dejen que los problemas les soprendan a mitad del trabajo, eso desmoraliza al programador perdiendo asi su interes en el trabajo y conllevando a una mala realizacion del proyecto.

2.- TODOS los nombres que usen traten de que sean lo mas relacionado con el contenido de los que trata, eso mejora a la hora de analizar los enlaces y/o la estructuración del sitio; de igual manera pasa con las variables a la hora de paginas activas en cualquiera de los lenguajes, y en el caso especifco de variables traten de definirlas antes de usarlas, o sea, forcen la definicion de las variables cada vez que puedan asi a la hora de analizar los errores sera mas comodo.

3.- El diseño de un sitio es todo un arte, JAMAS se lo tomen a la ligera. NO se debe programar NUNCA para un determinado tipo de explorador, eso demerita a los clientes cuando entran. La SOBRIEDAD en la sitios es algo digno y no un mal diseño, los diseños BARROCOS se deben usar solo para sitios donde tienen la VERDADERA necesidad de mostrar grandes volumenes de informaciones al mismo tiempo, si su sitio no admite mas contenido NO lo presione. Como norma se establece no mas de tres colores y no mas de tres tipografias en cada pagina, lo mas recomendable es usar un "sello único" a la hora de diseñar, o sea, que el cliente se acomode perfectamente de un solo vistazo a su sitio.

Cómo colocar un foro en tu página web

Para disponer de unos foros en tu página web tienes dos opciones principales. La primera es utilizar un servicio de foros de otra página web. La segunda es instalar unos scripts y una base de datos en nuestro alojamiento que hagan en trabajo desde nuestro propio dominio. Esta segunda opción es más interesante pero, lógicamente, requerirá más recursos a tener en nuestro sitio web y más conocimientos para ponerlo en marcha.

Utilizar un servicio de otro sitio web

Hay muchos sitios web que ofrecen foros para colocar en nuestras páginas. Estos foros suelen ser ligeramente configurables e integrables en nuestro sitio rápidamente y con poco esfuerzo. Como contrapartida, la empresa que ofrece el servicio suele incluir publicidad en las páginas del foro y la integración con nuestro sitio será menor. Otra desventaja, más obvia pero no menos importante es que las opciones para mejorar el servicio están fuera de nuestro alcance, ya que la administración del servidor de foros corre por cuenta de la empresa que ofrece el servicio.

En este sentido, en DesarrolloWeb.com podemos encontrar muchos enlaces de sitios que ofrecen foros para nuestra web. Los veremos todos en nuestra sección de recursos_gratis > elementos_para_webs > foros.

Instalar una aplicación de foros

Si deseamos ofrecer unos foros a lo grande, con muchas opciones de configuración y de adaptación a nuestro sitio, con posibilidad de administración y, en definitiva, con un control total de la aplicación, lo mejor es que instalemos una aplicación de servidor.

Es necesario que contemos con un alojamiento que permita algún tipo de programación en el servidor y alguna base de datos. Como posibles tecnologías de servidor podemos señalar CGI, JSP, ASP o PHP, si bien las dos últimas cuentan con muchas más posibilidades y desarrollos gratuitos y listos para usar. Las bases de datos podrán ser variadas, pero dependerán de la tecnología con la que estemos trabajando.

> **Referencia:** como ya hemos dicho, instalar un foro ya creado requerirá que sepamos un poquito sobre algún lenguaje del servidor. Si no sabemos nada de ellos, probablemente lo que digamos a continuación y la propia instalación del foro, no tendrá mucho sentido. Si es así, queremos apuntar los enlaces más interesantes de DesarrolloWeb para encontrar documentación sobre las tecnologías de servidor.
> ASP a fondo
> PHP a fondo

Categoría JSP de nuestro directorio
Categoría CGI de nuestro directorio

Cuando descarguemos alguna aplicación de estas, generalmente, nos encontraremos con un archivo comprimido Zip que contendrá las páginas web que conforman el foro y la base de datos que se utiliza en el desarrollo. Las páginas habrá que subirlas por FTP y la base de datos colocarla en el servidor, aunque la realización de esta segunda tarea dependerá de qué base de datos sea la que deseamos subir.

Generalmente también nos encontraremos con algún o algunos archivos de configuración del foro, que tendremos que editar para que funcione todo en nuestro servidor y con las características que deseemos. Pero todo esto seguramente vendrá explicado perfectamente en las instrucciones que seguro acompañan al foro en el archivo comprimido.

Para encontrar foros ya listos para usar podemos acceder a páginas como Hotscripts (buscar por Discussion Boards dentro de los scripts de la tecnología que estemos tratando) o http://sourceforge.net/ (podemos hacer una simple búsqueda por la palabra forum).

El posicionamiento en los buscadores de Internet

Todas las empresas con cierta entidad, hoy en día disponen de un sitio web, pero muchas de ellas se plantean si esta presencia en la Red ha sido una inversión o un mero gasto de representación.

Para que nuestra presencia en Internet sea rentable es necesaria la promoción de nuestro sitio, y una de las herramientas más rentables para dar a conocer nuestro site y generar visitas son los buscadores.

Aparecer en los principales buscadores internacionales, nacionales y en los específicos de nuestro sector es principal, pero no lo es menos aparecer en un lugar destacado según ciertas palabras clave que definan nuestro negocio en la mente de nuestras audiencias. Puesto la mayoría de los internautas se conforman con los primeros resultados proporcionados por su buscador favorito.

Y para aparecer en un lugar destacado en estas útiles herramientas de búsqueda debemos conocer su funcionamiento interno. Los buscadores se dividen en dos grandes grupos: los índices y los motores de búsqueda.

Referencia: en nuestro manual de promoción de páginas web podemos encotrar gran parte de la información de este artículo, relatada con mayor detalle y más calmadamente.

Los índices dividen la información en un árbol temático de categorías y subcategorías. Aquí el ejemplo paradigmático sería Yahoo!, que nos presenta una serie de grandes categorías temáticas entre las que encontramos la subcategoría "Economía y Negocios", dentro de ella "Empresas", y esta a su vez contiene entre otras "productos y servicios para empresas" y así sucesivamente hasta ir acotando la amplitud de la categoría de sitios web, ya que no se nos permitirá proponer la inclusión de nuestro sitio web en una categoría demasiado amplia. En los índices lo esencial es encontrar la rama ideal de este árbol temático en la cual ubicar nuestro sitio web, y digo nuestro sitio, puesto que en los índices sólo es posible incluir una página (normalmente la principal) a su directorio, aunque a menudo es posible incluirlo en dos o tres categorías. Para encontrar esta categoría ideal en la cual debería estar nuestro web, la estrategia a seguir es puramente marketiniana; ponerse en la piel de nuestro público y pensar en qué categoría nos buscará. Para ello, podemos ayudarnos de un estudio de mercado, y como no, del sentido común y de la observación de en qué categoría se encuentran ubicados nuestros principales competidores. Pero cuidado, quizá ellos no lo hayan echo tan bien y no se encuentren en la categoría ideal. Lógicamente dependiendo de la amplitud de nuestros productos o servicios, será más obvio o más difícil hallar esta categoría ideal. No obstante tras nuestra petición de alta existe un proceso de revisión humano e incluso podemos proponer una nueva categoría si no nos encontramos debidamente definidos por ninguna de las existentes.

Y sobre este proceso de revisión humana es sobre el que quiero hablar a continuación, puesto que es este el segundo factor que más diferencia a los índices de Internet de los motores de búsqueda. Cuando proponemos el alta de nuestro sitio en el índice de turno, se nos pide toda una serie de datos, como: Título de la página, URL, Definición, Ubicación geográfica, persona de contacto y correo electrónico... Y finalmente nuestro site es revisado por un surfer (un especialista en catalogar recursos) del índice que considera si nuestro site cumple con los estándares de calidad requeridos y si está bien clasificado en la categoría elegida por nosotros.

Vemos que esto es lo único que conoce el índice de nuestro sitio web; los datos suministrados en el formulario de petición de alta en el buscador. Por lo que debemos ser extremadamente cuidadosos en la definición que enviamos de nuestro site.

El caso de los motores de búsqueda es bien distinto. Podemos tomar como ejemplo a Google, y veremos que la única información que proporcionamos a un motor es la dirección URL (por ejemplo: www.miempresa.com) y quizá una dirección de correo electrónico. El resto del proceso se realiza de forma automática, ya que nuestra petición de alta en el buscador entrará en la cola de trabajo de un programa de software llamado spider (araña) que visitará la página que hemos dado de alta y a partir de ella todas las que se encuentren enlazadas y así sucesivamente. Simultáneamente nuestras páginas serán indexadas utilizando complejos algoritmos, para ser devueltas como resultado cuando un internauta utilizando el buscador, introduzca un término que se encuentre en alguna de ellas y haga una petición de extracción de información de su ingente base de datos. Vemos de esta forma que nuestro web puede aparecer en algún motor de búsqueda por la simple razón de que otra página de un tercero que está incluida en el buscador enlaza a ella en Internet.

Así, en los motores de búsqueda, para obtener una notable posición, lo esencial es el código de nuestras páginas, algo que era verdaderamente indiferente en el caso de los índices.

Teóricamente con sólo dar de alta nuestra página principal el buscador indexará todas las páginas que cuelgan de ella, pero habitualmente nos encontraremos con problemas derivados de la ventaja que se concede a las altas de pago frente a las gratuitas; el primero es el tiempo a esperar para que nuestro sitio sea introducido en la base de datos del motor de búsqueda, que puede variar entre varias semanas a varios meses según el motor en cuestión. E incluso a menudo, tras este dilatado periodo de tiempo, no seremos indexados en sus bases de datos. Y esto, en los motores que aún admiten el alta gratuita.

Como recomendación, si nuestro tiempo y energías son limitadas deberemos optar por el alta de pago en algunos buscadores.

Si conocemos las interacciones entre los distintos buscadores de Internet, descubriremos que la inclusión en alguno de ellos puede suponer la sucesiva inclusión en otros que a menudo son más "duros" con las admisiones.

Una vez que conseguimos que nuestro web aparezca en los buscadores, nuestro trabajo no habrá hecho más que empezar, ya que lo realmente valioso es aparecer en los primeros lugares por aquellas palabras clave que nuestros públicos utilizan para buscar nuestra categoría de productos, y esta sí que es una verdadera guerra, puesto que en esa lucha estamos frente a nuestros principales competidores, que también batallarán por mejorar la posición de sus páginas frente a las nuestras y las de otros competidores. Desde luego, que la complejidad dependerá de la popularidad de las palabras clave por las que queramos aparecer de forma destacada en el buscador.

Realmente cada buscador valora de distinta forma el código de nuestras páginas para ubicarla en una u otra posición de su ranking, así por ejemplo Google valora especialmente cuantas y que tipo de páginas apuntan hacia las nuestras, aplicando una lógica bastante humana, según la cual si muchos y especialmente importantes hablan de uno, es que uno es importante. Otros motores como AltaVista valoran los Meta Tags (unas líneas de código que informan al motor acerca del contenido de nuestras páginas), etc.

En general, los buscadores se fijan en la frecuencia o densidad y ubicación con la que aparecen ciertos términos en nuestras páginas, para ubicarlas en un lugar superior de sus resultados frente a otras páginas, en las cuales la frecuencia y ubicación de este término que el navegante ha introducido en la caja de búsqueda del motor aparece.

Así, una palabra que está presente en nuestra misma dirección de Internet (www.palabra.com) indica un elevado nivel de coincidencia si es el término buscado por el internauta. Después es especialmente valorada esta palabra, en el título del documento, en el primer párrafo más que en el segundo... Si está en mayúscula es más valorado, al igual que si está en negrita... Si aparece dos veces en una frase más que si aparece una, etc.

A estas alturas, seguro que a más de un lector se le ha pasado por la cabeza, la idea de llenar de términos clave la página para que aparezca en las primeras posiciones del buscador, pero lamentablemente esto ya esta contemplado por estas herramientas que si encuentran demasiadas palabras repetidas o un texto de tamaño muy pequeño o con el mismo color que el fondo, etc, penalizarán nuestras páginas o incluso las eliminarán de la base de datos por tratarse de técnicas de spam (técnicas de promoción ilícitas)

Creación de un menú desplegable en Dreamweaver

Muchas veces, nuestras páginas tienen tantos contenidos que si quisiéramos que apareciesen todos estos en un determinado lugar de nuestra página, en un menú, este abarcaría casi la totalidad del espacio que tenemos para nuestra web. En estos casos es muy útil insertar uno o varios de estos menús desplegables, apareciendo por encima del propio contenido de la página para mostrar todas sus partes y desapareciendo posteriormente.

A lo largo de este artículo vamos a desarrollar la forma de crear uno de estos menús con DreamWeaver de una manera genérica para que cada uno lo amplíe a su gusto.

En primer lugar, tenemos que saber que este menú está construido casi en su totalidad por capas, a las cuales les atribuiremos unos comportamientos específicos para que aparezcan y desaparezcan a nuestro gusto.

Otra cosa que debemos tener en cuenta es la condición de capa absoluta o capa relativa, ya que a las capas absolutas se les tiene que dar unas coordenadas de posicionamiento a raíz de la esquina superior izquierda de nuestra web, coordenadas que no nos sirven para nada en el caso de que nuestra página tenga los contenidos centrados, ya que la posición en la que va a aparecer esta, dependerá de la configuración del monitor desde el que se visualice dicha página.

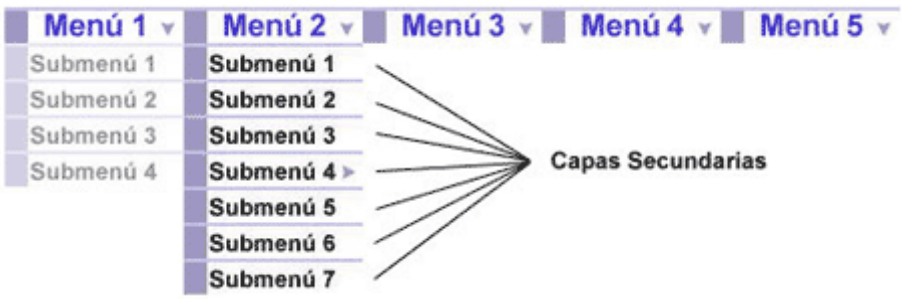
Menú Principal

El primer paso que tenemos que dar es desarrollar mentalmente un esquema del menú que queremos realizar, o mejor, sobre papel. Nosotros nos hemos decantado por hacer un menú principal compuesto por 5 partes en posición horizontal, para lo cual crearemos una tabla con 5 celdas y les daremos a cada una de ellas un determinado el tamaño de píxel, en nuestro caso 100px - 20px. A continuación haremos clic en el icono de capa y dibujaremos una en cualquier lado, seguidamente arrastraremos el símbolo de capa (que determina el lugar donde el programa introducirá la línea de código HTML, que por defecto creará dentro de la etiqueta "Body") dentro una de las celdas de la tabla que hemos generado y a continuación modificaremos en la ventana propiedades, los campos "Iz" (izquierda) y "Sup" (superior) dejándolos en blanco, al hacer esto el programa engancha la capa en la esquina superior izquierda del recipiente en el que se encuentra, en este caso, la celda en la que hemos introducido la capa. Después daremos un valor a los campos de "An" (ancho) y "Al" (alto) En el ejemplo que estamos creando serán 100px y 20px respectivamente, este paso lo tenemos que repetir para cada una de las 5 celdas (en nuestro caso), de las que se compone nuestro menú principal.

Una vez terminado con este paso procederemos a crear otras capas dentro de las que ya hemos establecido, a estas capas tendremos que darles unos valores de tamaño dependiendo de las distintas partes que queramos introducir dependientes de cada menú (nosotros daremos 100px - 80px en el primer desplegable, 100px - 140px en el segundo, 100px - 100px en el tercero, 100px - 80px en el cuarto y 100px - 140px en el quinto). Para colocar una capa dentro de otra podemos hacerlo: 1º, arrastrando como hemos hecho anteriormente esta nueva capa dentro de la anterior, o 2º, presionando la tecla F2 nos aparecerá la ventana "capas" donde podremos ver un esquema de las capas que tenemos en nuestra web, cogiendo una de las capas que aparecen y arrastrándola encima de otra mientras que presionamos la tecla "ctrl", introduciremos la capa arrastrada dentro de la que hayamos seleccionado.

Estas subcapas que hemos creado, que dependen de las principales, por defecto tienen la propiedad "default" que deja la capa visible en todo momento y nos viene bien para trabajar, pero antes de darles un comportamiento a estas, cuando tengamos terminado por completo la estructura de nuestro menú, deberemos cambiar este tributo "default" de las capas secundarias (las que se encuentran dentro de las 5 capas principales) por "hidden", que las hace invisibles, dándonos la posibilidad de trabajar con los comportamientos haciéndolas aparecer y desaparecer a nuestro gusto.

Dentro de estas capas secundarias introduciremos tablas con el numero de celdas que hayamos calculado con el tamaño de cada capa. Una vez hecho esto en todas las capas deberíamos colocar las imágenes o las palabras de nuestro menú para poder configurar los comportamientos, ya que si no colocamos nada dentro de las capas, estas no se verán cuando aparecen o desaparecen por ser transparentes. Otra solución es darles un color de fondo.



El último paso para terminar nuestro menú será dar a cada capa un comportamiento. Para eso necesitamos la ventana comportamiento que, si no la tenemos ya a la vista, presionando la tecla F3 aparecerá. Para poder jugar con los comportamientos de "mostrar u ocultar capa" debemos trabajar con

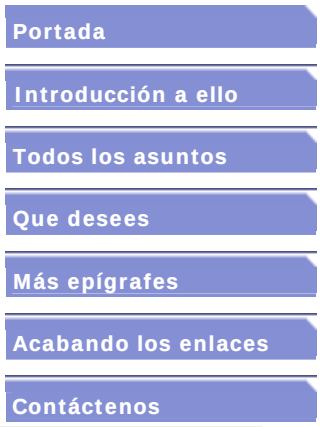
un vinculo (estos vínculos serán las imágenes que hemos colocado dentro de las celdas y en su defecto las palabras, para hacer este vinculo introduciremos en el espacio reservado para los vínculos de la ventana propiedades el símbolo "#" creando así un vínculo en blanco, a continuación presionaremos el vinculo creado (en la parte inferior de la página principal aparecerá el símbolo de vínculo "<a>" en negrita) después iremos a la ventana comportamiento y presionaremos el símbolo "+" y pincharemos en "Mostrar - Ocultar capas" y nos aparecerá una ventana con un listado de todas las capas que tenemos creadas, lo que tenemos que hacer es presionar la capa que se tendría que desplegar cuando pasásemos el ratón por encima del vinculo que estamos modificando y apretar el botón "Mostrar" y las demás capas pincharlas y ocultarlas de la misma forma que hemos hecho antes pero en este caso apretando el botón de "Ocultar".

Este paso lo repetiremos para cada una de los cinco vínculos principales de los que se nos desplegarán los correspondientes submenús.

Tabla mejorada con imagenes para barra de navegación

En este taller de HTML vamos a ver como una pequeña imagen puede resultar muy vistosa para construir una barra de navegación para nuestro sitio web.

Vamos a construir una tabla como la que se puede ver en la parte de la derecha, donde podremos observar la utilización de imágenes para camuflar el hecho que las celdas son siempre rectangulares. Las imágenes aplican un leve biselado y eliminan una esquina, con lo que las tablas mejoran sensiblemente su apariencia.



Nota: Un efecto como este o parecido se puede conseguir de muchas maneras, así que nos tenemos que tomar este taller como tan sólo una idea de las posibilidades y el modo de construir las tablas.

Las imágemes

Creo que viendo las imágenes que hemos colocado en la tabla se comprenderá un poco la idea sobre la que hemos trabajado. Las imágenes están ampliadas para que se pueda observar mejor sus líneas. Se pueden crear con cualquier editor gráfico del que dispongamos.


Esta es la imagen que colocamos entre dos celdas de texto.


Esta es la imagen que colocamos en la parte de arriba de la celda superior. No podemos colocar la misma que la de en medio porque queda un poco mal.

Consideraciones para crear la tabla

Vamos a colocar cada elemento en la tabla en una celda independiente. En la primera celda colocaremos la imagen destinada para la parte de arriba, en la segunda el texto del primer enlace, luego la imagen que colocamos en medio de cada celda de texto, seguida por otra celda con el texto del siguiente enlace, luego otra vez la imagen, luego texto, etc.

La tabla se tiene que crear de modo que no quede separación entre celdas ni márgenes, pues si la hubiera no parecería que las celdas de la imagen y las del texto forman un mismo bloque y aparecería deslabazada. Los atributos cellspacing y cellpadding quedarían a cero.

Además, las etiquetas <TD> y las de las imágenes, <IMG>, tienen que estar en el código sin espacios entre medias, pues si no fuera así tampoco conseguiríamos que las celdas quedasen pegadas unas a otras.

Por lo demás, decir que las celdas de texto les hemos aplicado estilos utilizando CSS (Hojas de estilo en cascada), que son mucho más cómodos y nos permiten definir una única vez el estilo para todas las celdas en lugar de repetir las etiquetas y atributos HTML para cada una.

Código de la tabla

Colocamos el código de toda la página en lugar de solamente la tabla para que se puedan ver las etiquetas para colocar los estilos CSS, que aparecen en la cabecera.

```
<html>
<head>
   <title>Tabla enlaces guay</title>
   <style>
   .celda {background-color:#848ED3;font-size:8pt;font-family:verdana,arial;color:#ffffff;font-
weight:bold;padding-left:3px;padding-bottom:2px;}
   </style>
</head>

<body>

<table cellspacing="0" cellpadding="0" border="0">
<tr>
   <td><img src="arriba.gif" width="155" height="6" alt="" border="0"></td>
</tr>
<tr>
   <td class="celda">Portada</td>
</tr>
<tr>
   <td><img src="medio.gif" width="155" height="8" alt="" border="0"></td>
</tr>
<tr>
   <td class="celda">Introducción a ello</td>
</tr>
<tr>
   <td><img src="medio.gif" width="155" height="8" alt="" border="0"></td>
</tr>
<tr>
   <td class="celda">Todos los asuntos</td>
</tr>
<tr>
   <td><img src="medio.gif" width="155" height="8" alt="" border="0"></td>
</tr>
<tr>
   <td class="celda">Que desees</td>
</tr>
<tr>
   <td><img src="medio.gif" width="155" height="8" alt="" border="0"></td>
</tr>
<tr>
   <td class="celda">Más epígrafes</td>
</tr>
<tr>
   <td><img src="medio.gif" width="155" height="8" alt="" border="0"></td>
</tr>
<tr>
   <td class="celda">Acabando los enlaces</td>
</tr>
<tr>
   <td><img src="medio.gif" width="155" height="8" alt="" border="0"></td>
</tr>
<tr>
   <td class="celda">Contáctenos</td>
</tr>
</table>

</body>
</html>
```

Parcheando aplicaciones

Bien saben los técnicos de sistemas que hay que estar siempre al tanto de las actualizaciones de los programas de ordenador, ya que incluyen solución a los agujeros de seguridad que se van detectando. Sin los parches somos vulnerables a los virus y programas malignos. Ningún usuario, por muy novato que sea, debe olvidarlo.

Durante el transcurso de un año, podemos encontrar del orden de 10 a 20 actualizaciones críticas en software de uso común que solucionan agujeros de seguridad importantes y que nadie debería dejar de instalar, igual que nadie se olvida sus zapatos para no lesionarse las plantas de los pies al pasear por la calle.

Hay que estar al tanto de las actualizaciones y para ello podemos visitar las páginas web de los desarrolladores del software que utilizamos habitualmente, brindando especial atención al sistema operativo o los programas de Internet. Resultará también muy útil leer cada cierto tiempo publicaciones con noticias tecnológicas o especializadas en seguridad, como Hispasec. http://www.hispasec.com/

Parchear el sistema operativo

El caso más clásico es el de Windows, el sistema operativo de la mayoría de los ordenadores personales del mundo y también una de las mayores fuentes de vulnerabilidades. Sólo tomando este sistema operativo en su versión más moderna de la actualidad, Windows XP, podremos encontrar 5 o 6 actualizaciones críticas, aun siendo un producto presentado hace relativamente poco tiempo. Si nos referimos a otros sistemas operativos de Microsoft, como Windows 98, el número de parches críticos que podremos encontrar supera la decena, debido a que este sistema es mucho más antiguo y, por lo tanto, ha habido mucho más tiempo para detectar agujeros de seguridad.

Microsoft por lo menos cuida a sus usuarios y pone muy a la vista sus actualizaciones, criticas o no tanto. Accediendo a la página http://windowsupdate.microsoft.com podemos encontrar una aplicación web que revisa nuestro sistema operativo Windows, independientemente del modelo o versión y nos informa de las actualizaciones que debemos introducir. El informe recoge tanto las actualizaciones críticas, que no debemos dejar de instalar, como las actualizaciones importantes, que no se refieren a problemas de seguridad, sino a detalles relevantes sobre el trabajo con el ordenador, como puede ser una adaptación del sistema al Euro.

Aunque no todo son bondades en los sistemas de actualización de Microsoft. Para empezar, el simple hecho de disponer de tantas actualizaciones resulta cuando menos polémico. Sin embargo, la nota más negativa resulta que Microsoft va "jubilando" sus sistemas operativos con el tiempo. Por ejemplo, con la entrada de 2003 ha calificado de obsoletos a sus sistemas MS-DOS, Windows 3.x, Windows 95 y Windows NT 3.5. Windows ME, con tan sólo unos pocos años de vida, se jubilará el 31 de diciembre de 2003. A partir de entonces, Microsoft no ofrecerá más asistencia ni actualizaciones para ellos, lo que puede dejar desprotegidos a miles de usuarios de todo el mundo, que deberán actualizar su sistema operativo para disponer actualizaciones de seguridad.

Windows no es el único sistema operativo donde podemos encontrar vulnerabilidades. En realidad, cualquier sistema construido por el hombre está potencialmente afectado y Linux o Mac OS no son una excepción. Bien es cierto que las vulnerabilidades encontradas son en número menores, pero también es verdad que el ímpetu con el que se buscan los agujeros en Windows es mucho mayor, debido a que algunos informáticos utilizan buena parte de su tiempo y se divierten buscando manchas en el expediente de Microsoft.

Por otro lado, decir que hay menos vulnerabilidades en Linux que en Windows, puede resultar erróneo para la opinión de ciertas empresas, o por lo menos, una verdad a medias. Al cabo del año se realizan muchos informes por consultoras que sitúan por delante, en la dudosa clasificación de agujeros de seguridad, a unos u otros sistemas, dependiendo, lógicamente, de quién encargue el informe. Es un tema al que es preferible no entrar, dado que cada cuál tiene sus impresiones y, como vemos, hay muchas opiniones contrastadas. Lo que sí esta claro es que existen muchos menos virus y usuarios con problemas de seguridad en Linux y que la comunidad que desarrolla el sistema está siempre atenta a los agujeros que se puedan encontrar para solucionarlos a las pocas horas de haber sido detectados.

Parchear programas

No sólo los sistemas operativos están afectados por las vulnerabilidades, también los programas, muchos de ellos de uso habitual, pueden afectar nuestros ordenadores, brindando a las personas maliciosas agujeros de seguridad por los que acceder a nuestros equipos y realizar tareas dañinas.

Cuanto más cerca de Internet están los programas o los ordenadores, más problemas de seguridad pueden aparecer. Uno de los ejemplos de programas con serias vulnerabilidades que se detectan periódicamente es Flash, una herramienta para la visualización de contenido dinámico y multimedia en páginas web que tienen instalada el 97% de los ordenadores, según Macromedia, la compañía que desarrolla el producto. Hace poco en Flash fueron encontradas algunas vulnerabilidades para las que es necesario actualizar el visor a la versión más moderna para poder evitarlas. Macromedia no ofrece una información clara en su página para que los usuarios actualicen sus programas, podríamos decir que esconde sus vulnerabilidades y con ello la posibilidad de que los usuarios las subsanen rápido.

Otros programas para los que se han detectado vulnerabilidades importantes son los llamados P2P (Peer TO Peer) que sirven para compartir, de igual a igual, programas y ficheros multimedia entre los usuarios de Internet. Estos ficheros algunas veces están infectados por virus que utilizan vulnerabilidades para ejecutar código dañino, agujeros que las discográficas siempre reciben con buenos ojos puesto que los programas para compartir información, generalmente música, resultan muy contraproducentes para sus actividades.

El paquete de herramientas Office, también de Microsoft, para el trabajo de oficina presenta también varias vulnerabilidades importantes para las que existen los correspondientes parches. Microsoft informa sobre ellos en la página web http://office.microsoft.com/ProductUpdates/

Conclusión

Es normal que, una vez lanzado un producto software, se encuentren vulnerabilidades importantes que no se había tenido en cuenta antes. Debido a que la creación y distribución de nuevas ediciones de los programas resulta muchas veces muy costosa y es casi imposible hacer llegar a todos los usuarios un parche o una nueva versión del programa comprado, debemos estar al tanto de las actualizaciones que se van presentando en el software. Es importante instalar las actualizaciones y, de ese modo, estar protegidos contra posibles agujeros de seguridad.

Las actualizaciones sólo se presentan cuando se ha detectado un problema y puede que, incluso antes de que se distribuya un parche para corregir el error, existan programas o virus que utilicen los agujeros de seguridad para hacernos padecer serios quebraderos de cabeza con nuestros ordenadores.

Desabilitar la barra de imágenes de Internet Explorer

¿Cuantas veces nos hemos quejado o hemos deseado que no apareciese esa incómoda barrita en Internet Explorer (Versión 6) cuando pasamos el puntero por encima de una imagen?. Pues la solución es extremadamente simple...

Imagen que muestra la barra emergente.

Tan sólo, lo que tenemos que incluir en la cabecera de nuestra página, es una escueta etiqueta META que automáticamente nos deshabilitará la susodicha opción.

También disponemos de un comportamiento de Dreamweaver que nos ahorrará el tener que escribir dicha META. Pero cuando veáis lo reducida que es, probablemente paséis de descargaros el comportamiento. De todas formas para todos aquellos que deseéis tenerla, la podréis adquirir en la web de Macromedia Exchange.

Esta es la etiqueta META que deberéis incluir entre <HEAD> y </HEAD>

```
<meta http-equiv="imagetoolbar" content="no">
```

Y lo veremos un poquito mejor en un pequeño ejemplo de código HTML, de modo que no quede ningún tipo de duda.

Ej:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//SP">
<html>
<head>
<!-- #BeginEditable "doctitle" -->
```

```
<title>:::: Mi página web::::</title>
<!-- #EndEditable -->
<META NAME="TITLE" CONTENT="Nombre">
<meta http-equiv="Content-Type" content="text/html; charset=windows-1252">
<meta name="Description" content="descripción de nuestra web.">
<meta name="Keywords" content="palabras relacionadas con el contenido de nuestra web.">
<meta name="author" content="Nombre de la empresa.">
<META HTTP-EQUIV="EXPIRES" CONTENT="Mon, 31 Dec 2054 00:00:01 PST">
<meta name="reply-to" content="correo electrónico de la empresa">
<meta name="owner" content="Propietario de la empresa.">
<META HTTP-EQUIV="VW96.OBJECT TYPE" CONTENT="Tipo de web">
<META NAME="RATING" CONTENT="General">
<meta name="robots" content="index, follow">
<META NAME="REVISIT-AFTER" CONTENT="7 days">
<meta http-equiv="imagetoolbar" content="no">
</head>
<body>

El contenido de nuestra web.

</body>
</html>
```

Recordad que no importa donde coloquéis la etiqueta META, siempre y cuando esta se encuentre entre el <HEAD> y </HEAD>.

Cómo evitar que una página se imprima

Para ello, hay que echar mano de las hojas de estilo. Tanto si el documento tiene una hoja ya asociada como sino, lo que vamos a hacer es asociarle un nueva hoja de estilos. Dicha hoja contendrá un único estilo, con el código necesario para ocultar un elemento:

```
.nover{
visibility:hidden
}
```

A la hora de asociar la hoja de estilos, se le añade un modificador a la etiqueta HTML que enlaza con el fichero .css que permite especificar para qué tipo de medio se aplicará esta hoja. En nuestro caso, se aplica en el ámbito de la impresión, por lo que se utiliza el atributo media="print".

```
<link href="nombre_hoja" rel="stylesheet" type="text/css" media="print">
```

Una vez hecho esto, basta que toda nuestra página este dentro de un elemento div, que pertenezca a la clase nover.

```
<body>
<div class="nover">

-- Contenido --


</div>
</body>
```

Al hacer esto se provoca que en pantalla se visualice la página, pero que si por el contrario se decide imprimir, se le aplicará la hoja de estilos de impresión, en la que el elemento esta puesto como no visible, por lo que no se imprimirá.

Codigo Completo:

Veamos el código íntegro de la página web y la hoja de estilos asociada.

Página HTML

```
<html>
<head>
<link href="estilos.css" rel="stylesheet" type="text/css" media="print">
</head>
<body>
<div class="nover">
```

```
... contenido de la pagina

</div>
</body>
</html>
```

Hoja estilos: estilos.css

```
.nover{
visibility:hidden
}
```

> **Nota:** Esta catacterística de las hojas de estilos funciona con éxito en navegadores IExplorer 6, Netscape 7 y Opera 7. No la hemos probado en otras versiones.

Presentar un presupuesto cautivador

Son muchos los profesionales del diseño y desarrollo web que no parecen dar la importancia que debería tener el primer contacto con un cliente cuando, este último, solicita presupuesto a un proyecto. Cada vez más, el usuario de Internet tiene más conocimiento sobre el funcionamiento del mismo y la presentación de un presupuesto escueto en información, no ayudará en nada a que nos encarguen el proyecto.

Así pues, está claro que algo hay que cambiar. De nada le servirá al buen desarrollador o diseñador web que su trabajo sea espléndido y profesional si no es capaz de demostrar esa profesionalidad en el primer contacto con el cliente. ¿Cómo conseguir difundir seriedad y confianza a un posible cliente?. Veamos...

Cuando va a comprar una vivienda antes de ser construida, lo primero que el constructor nos enseña son los planos del proyecto inmobiliario. De este modo, sin poder ver la vivienda, pues aún no existe, sabremos cuantos habitáculos tendrá, los metros de cada uno, su disposición, y antes de decidir nada el constructor nos indicará las fechas de ejecución, las garantías, los plazos de pago y el precio. Bien, así debería ser un presupuesto de un proyecto web.

Es preciso que un cliente no experimentado en Internet visualice mentalmente su futuro sitio para poder ilusionarlo y esto será posible incluyendo un completo plan que describa las fases y elementos que compondrán su proyecto. También es conveniente que no se cometa el error de dar exclusivamente un precio a aquellos cliente que saben lo que quieren para su sitio web, también estos últimos agradecerán el recibir información sobre el tratamiento que recibirá él y su proyecto.

Un esquema básico de un presupuesto (mejor llamado plan de proyecto), podría ser este:

- **Introducción**: Una pequeña introducción a como hemos enfocado las necesidades del proyecto y que motivos nos han llevado a decidir que este presupuesto es el más conveniente para el proyecto.
- **Descripción estructural del sitio**: Explicación sobre la organización del sitio en secciones y breve comentario informativo sobre lo que contendrá cada una de ellas.
- **Descripción funcional del sitio**: Aquí informaremos sobre la interactividad del sitio, tanto a nivel de usuario como de administración.
- **Metodología de trabajo**.
- **Tiempos de ejecución**: Definimos las fases de presentación y le damos al cliente fecha de finalización de cada una de ellas. No es recomendable establecer aquí fechas fijas (a no ser que haya contrato de por medio).
- **Garantía y soporte**.
- **Tiempos y formas de pago**.
- **Presupuesto detallado** (Con especial énfasis en detallado).

No se ha de tener miedo de que nuestros presupuestos sean como contratos, jamás perderás un cliente porque le ofrezcas una extensa información de donde y como está gastando su dinero. Más bien al contrario.

Como proteger el código fuente de una web

Tras intentarlo con muchos métodos, scripts y demás he llegado a la conclusión de que ningún método es perfecto y de que todos estos scripts que dicen que protegen el código fuente en realidad lo único que hacen es bloquear el botón derecho del ratón y el teclado.

De todos estos scripts y para el que le guste usarlos este es el que me parece el más simple y mejor:

```
<body oncontextmenu="return false" onkeydown="return false">
```

simplemente hay que poner estos atributos en el body y el botón derecho del ratón y el teclado quedarán inutilizados, sin que salgan esas molestas ventanas de alerta diciendo, "el botón derecho ha sido inhabilidado" o "las imágenes están protegidas"....

Estos scripts puede que protejan las imágenes de usuarios inexpertos que simplemente saben navegar por internet y poco más. Pero no impedirán a un usuario experto conseguir copiarse estas imágenes a su ordenador. Salvar estos scripts es tan fácil como darle: "Archivo>Guardar como..." y guardarse la página completa con todas sus imágenes.

En el caso de que el script solo proteja el botón derecho incluso podemos seleccionar la imagen y hacer Ctrl+C y luego pegarla en cualquier editor gráfico.

Por si todo esto no funcionara también queda siempre la posibilidad de imprimir pantalla, con el botón: "Impr Pant Pet Sis" y luego Ctrl+V en cualquier editor gráfico.

Capitulo 5
Introducción a la promoción de webs

Una vez hemos construido una página web tenemos que **hacer que esta sea conocida** por todos los medios que estén a nuestro alcance, para atraer visitas a ella y, cuando menos, sentirnos orgullosos de que esta sea popular en la Red. Para conseguir esto tenemos que promocionarla adecuadamente, de manera que su dirección figure en el mayor número de sitios.

Existen muchas maneras de promocionar una página,

- Podemos enviar correos anunciando su puesta en marcha
- Registrarla en los buscadores
- Incluir la dirección en el membrete del papel de nuestro negocio
- Anunciarla por televisión, por poner varios ejemplos.
- Una vez la web ya lleva un tiempo online podemos hacer que los visitantes vuelvan a ella mediante técnicas adicionales como listas de correo que informen a los usuarios de las novedades.

Este manual pretende dar a conocer las técnicas para conseguir los anteriores objetivos, centrándonos en el más importante: **el registro en buscadores**, que requiere una serie de indicaciones muy específicas para que nuestra página este presente y ocupe los **primeros puestos en los resultados de la búsqueda**.

Pero atención, no es solo nuestro trabajo promocionando una página el que va a atraer las visitas, también debemos mantener la web con buena presencia, que sea rápida y que tenga unos buenos contenidos. En otras palabras, con la misma promoción, una buena página web atrae más visitas que una mala, algo muy lógico si lo piensas.

Este manual está compuesto por varios capítulos. En los primeros vamos a conocer al detalle todos los mecanismos y los trucos relativos a los buscadores. En capitulos más avanzados veremos otras maneras de promocionar la página.

Tipos de buscadores

Existen varios tipos de buscadores en Internet. En primer lugar, podemos distinguirlos por su forma de trabajo, esto es importante, dado que **la manera de registrar una dirección en los buscadores es diferente según el tipo**.

Indices

Son los buscadores que mantienen una **organización de las páginas** incluidas en su base de datos **por categorías**, es decir, tienen un directorio navegable de temas. Dentro de cada directorio podemos encontrar páginas relacionadas con ese tema. Para mantener esta organización, los buscadores tienen unos administradores humanos que se encargan de visitar las páginas y vigilan que todas se encuentren clasificadas en su lugar correcto. Índices típicos son **Yahoo**, **Terra** o **TodoEnlaces**.

Para que una página quede registrada en un índice debemos **mandarles la dirección a los administradores** humanos de ese índice, generalmente acompañada de una serie de datos que les ayuden a clasificar la página de una forma correcta, como la descripción, temática, titulo, lenguaje, etc. Además, si queremos que varias páginas de nuestro sitio web estén en el buscador, **deberemos registrarlas todas ellas una a una**.

Motores de búsqueda

Son buscadores que basan su **recolección de páginas en un robot**, denominado araña, que recorre constantemente Internet en busca de páginas nuevas que va **introduciendo en su base de datos automáticamente**. Los motores de búsqueda, no tienen porque tener un índice, aunque cada vez es más habitual que dispongan de uno. Motores de búsqueda típicos son **Altavista** o **Sol**.

Los motores de búsqueda, como se puede haber deducido, no necesitan que les mandemos la dirección de nuestra página para tenerla en su base de datos, puesto que el robot puede haberla encontrado previamente. De todos modos, nosotros podemos **mandarles la dirección si no deseamos esperar a que el robot nos encuentre**, practica muy habitual.

Para clasificar una página, los motores de búsqueda son capaces de leer el contenido de esta y encontrar

aquellos datos que permitan su catalogación. Por esto es que cuando registramos una página en un motor de búsqueda generalmente no nos piden información adicional, como ocurría con los índices.

Cuando un robot recorre nuestra página guarda sus datos, y luego se dirige a las distintas páginas que están enlazadas a esta. De este modo, solo hace falta registrar la página inicial de un sitio web, pues el motor de búsqueda se encargará de **recorrer todo el sitio de manera automática**. Adicionalmente, estos motores o arañas, volverán a recorrer las páginas de su base de datos en busca de cambios que se hayan producido en estas, con objetivo de mantener su información lo más actualizada posible.

Multibuscadores

Estos últimos no tienen una base de datos propia, lo que hacen es buscar la **página en unos cuantos motores de búsqueda** e índices y combinar los resultados de la búsqueda en esos buscadores. Como ejemplos de multibuscadores podemos destacar **Meta buscador de desarrolloweb**, en castellano, o **Metacrawler**, en inglés.

Para registrar una dirección de manera que aparezca en un multibuscador debemos mandársela a los algún buscador donde este va a recoger los resultados.

Otras clasificaciones

Aparte de la clasificación anterior de los buscadores, también se pueden distinguir de otras formas.

Por su ámbito: los hay internacionales, nacionales, incluso de regiones más pequeñas, como provincias o ciudades.

Por el tema: existen buscadores genéricos, donde podemos encontrar todo tipo de páginas, y también los hay temáticos, donde solo hay páginas que tratan sobre una temática específica.

A la hora de registrar una página debemos comprobar que pertenece al ámbito o temática del buscador, o de lo contrario es casi seguro que no la aceptarán.

En el siguiente capítulo vamos a ver cómo se ha de registrar una página en cada tipo de buscador.

Cómo registrar una página

Para conseguir que tu web sea incluida en un buscador debes **rellenar un formulario con los datos de la web** que deseas registrar. Cada buscador tiene un formulario especifico y a menudo **el proceso de registro es diferente**.

Para encontrar el formulario lo más fácil es que entres en la página principal del buscador y busques un enlace que ponga **Añadir página, Add URL, nueva dirección**, o algo parecido. Ese enlace te llevara al formulario de registro o, en su defecto, a la página donde explica cómo has de registrarte en ese buscador en concreto.

Aun así, existen unos procedimientos básicos de registro que suelen repetirse en los buscadores:

Para los índices

Los buscadores que tienen categorías donde podemos encontrar páginas clasificadas según su temática. Por lo general se ha de **navegar a la categoría** en la que se desea incluir el web. Allí se busca el botón "añadir página"(o algo parecido) y se encuentra el formulario para el registro.

Para los motores de búsqueda

Que son los buscadores que no tienen porque mantener un índice y que tienen robots que constantemente recorren Internet en busca de nuevas páginas para incluirlas en el buscador. Estos buscadores suelen tener un formulario accesible desde la página inicial, con el enlace correspondiente. **No hay que navegar las categorías** para acceder al formulario.

Salta a la vista que registrarte en varios buscadores puede ser una tarea un poco costosa y otro tanto pesada, pues tienes que ir **buscador a buscador introduciendo una y otra vez los mismos datos**. Afortunadamente existen herramientas multiregistro. Estas herramientas nos permiten registrar nuestra página en distintos buscadores introduciendo una sola vez los datos de la página web.

Existen dos tipos de herramientas multiregistro:

Aplicaciones windows, como cualquier otro programa, pero que su objetivo es registrar en buscadores. Este tipo de herramientas suelen estar a la venta, o puedes encontrarlas shareware (en www.tucows.com por ejemplo), pero suelen estar limitadas en su uso.

Herramientas online, estas nos permiten desde Internet, y por lo general gratuitamente, registrarnos en varios buscadores. No suelen ser tan potentes como las anteriores, pero si más útiles y **accesibles** por ser **gratuitas**. En desarrolloweb disponemos de una **herramienta multiregistro online**, vigilada de cerca para que todo funcione bien. Existen otras de estas herramientas en Internet, se pueden ver en la **sección de promoción de nuestro buscador**.

Por otro lado, hay que destacar que existen una serie de métodos que nos pueden ayudar a que nuestra página se encuentre mejor situada entre los resultados. pero esto es otro tema, que se tratará en el siguiente capítulo.

Trucos para el registro

Antes de empezar el proceso de registro en buscadores es muy recomendable, incluso absolutamente necesario, que las páginas **hayan sido revisadas y modificadas, incluyendo todas aquellas ayudas que permitan una mejor clasificación** de las mismas. De este modo, conseguiremos que nuestras páginas sean fácilmente localizables entre los resultados del buscador.

Existen unas reglas y recomendaciones básicas para conseguir este objetivo. Elegir bien el título, descripción o palabras clave son algunos ejemplos. Vamos a ver con detenimiento en este reportaje cada una de estas técnicas. No nos tenemos que olvidar tampoco que el uso excesivo de estas puede ser contraproducente ya que muchos motores de búsqueda interpretan que la **página esta haciendo "trampa"** para situarse entre los primeros puestos de manera injusta. Cuando nuestra página es catalogada como "spam page" o "página tramposa" (en adelante **spam**), es confinada a los últimos puestos en los resultados de manera automática. Hay que utilizar estas técnicas con moderación, como se puede ver, por regla general.

Antes de abordar el tema sería adecuado conocer **varios conceptos que vamos a utilizar**:

Palabras clave

Son palabras sueltas que describen una página web, las que pensamos que van a **escribir los usuarios en los buscadores** para encontrar una página como la nuestra.

En este manual, cuando hablemos de palabras clave nos referimos a **palabras sueltas**, pero en la practica, tanto en la busqueda de documentos como en la preparación de nuestra página para ser reconocida por los buscadores eficientemente, se utilizan las frases clave, descritas a continuación.

Frases clave

Son las **frases que van a escribir los usuarios** en los buscadores para encontrar un tipo de página. Si buscan mecánicos de coches de Vallecas, una población de la comunidad de Madrid, podrán escribir algo como mecánicos de vallecas madrid, reparación coche vallecas madrid, taller mecánico vallecas madrid, taller coche vallecas...

Las frases clave, como se ve, deben incluir las **distintas combinaciones de palabras clave** que se supone que van a utilizar los distintos usuarios para localizar nuestra página. Escogerlas bien es importante y para ello podemos valernos de un buscador. Haciendo pruebas entre distintas combinaciones de frases clave podemos comprobar cuales son las que ofrecen unos resultados mejores, en las que los usuarios, es de suponer, que se fijarán más, rebuscando incluso en páginas sucesivas de resultados.

Es interesante el uso de frases clave por que son utilizadas habitualmente por los usuarios en sus búsquedas. Además, si utilizas frases clave es más posible que alguien te encuentre pues posiblemente no tengas tantos competidores como los que tendrías con una palabra simple.

Descripción

Es una **frase que describe el contenido de la página**. Se corresponde con la descripción que un buscador ofrece de cada página en los resultados de la búsqueda.

Caracteres especiales

Existe una duda generalizada sobre la utilización de los **acentos, mayusculas, minusculas** y otros caracteres. Nosotros recomendamos **escribir todas las palabras y frases correctamente, usando los acentos y las mayusculas donde sea necesario**. Esto es así porque cuando buscamos, en Altavista por ejemplo, una palabra y la buscamos con acento sólo nos devuelve las páginas que contemplan esa palabra con los acentos, mientras que si la buscamos sin acento nos las ofrece todas, las que lo tienen como las que no.

Ahora veamos cómo hacer llegar a un motor de búsqueda todos los conceptos anteriores de palabras y frases clave, así como la descripción.

Dónde colocar las palabras clave

Ahora vamos a ver **donde situar** las palabras y frases clave, así como la descripción dentro de la página web.

El Título de la página

Lo primero que debemos hacer es escoger un titulo apropiado, es decir, un titulo que describa bien la página donde se encuentra y contemple otras pequeñas consideraciones:

- **Tiene que ser muy descriptivo**. Algo como Taller mecánico no es suficiente. Debería indicar también la localización del taller o cualquier información adicional que pudiera enriquecer la descripción, como el nombre del taller o su especialidad.
- Ha de **incluir las palabras clave** que definen nuestro sitio. Además debería contener las palabras más importantes al principio, pues muchos buscadores solo se fijan en las primeras palabras del título.
- La longitud del título también es otro parámetro que hay que ajustar. Un titulo corto es poco adecuado, pero los buscadores pueden pensar que les estas haciendo spam si encuentran uno demasiado largo. Hay muchas consideraciones al respecto, pero podemos indicar que lo más adecuado es que sea de **15 a 20 palabras de longitud, o del orden de los 50 a 100 caracteres**.

Un título apropiado para el mecánico de Vallecas sería:

Taller mecanico de coches en Vallecas, Madrid, Todo Auto. Reparacion de lunetas limpiaparabrisas electricidad recambios aceite

Primeros párrafos

Es otra cosa en la que se fijan mucho los motores de búsqueda al indexar una página web. Si nuestra web lo permite, debemos colocar un primer párrafo que sirva de **introducción a todo lo que podemos encontrarnos en la página** o en la web. Este primer párrafo,

- Deberá estar colocado dentro del <BODY>, lo **mas cerca posible** de este. Si es posible, evitaremos que entre la etiqueta y el primer párrafo existan otras etiquetas, como imágenes.
- Deberá **ser legible**. Este texto se verá en la página, igual que cualquier párrafo, por eso debe leerse bien.

Podríamos poner en la página del mecánico este texto después de la etiqueta <BODY>

<H1 align=center> Todo Auto</H1>
En Vallecas, situado en la carretera de Valencia a 3 km de Madrid, se encuentra nuestro taller de reparación de automóviles y recambios. Estaremos gustosos de atenderles y ofrecerles nuestros servicios de todo tipo para su coche, como cambio de aceite, reparación de lunetas termicas, y otros.

De manera adicional, y sobretodo durante los primeros párrafos podemos utilizar otras técnicas, de más dudosa utilidad.

- **Utilizar los encabezamientos** (etiquetas <H1> y similares) para destacar los títulos de las páginas siempre que se pueda.
- Definir los atributos **ALT de las imágenes** con palabras claves

Estas técnicas han de usarse siempre con moderación, evitando cualquier uso excesivo que acabe perjudicando la posición del sitio.

Etiquetas META

Muchos buscadores, sobretodo los motores de búsqueda, leen determinadas etiquetas en la página para encontrar la descripción y las palabras clave de esta, se tratan de las etiquetas META. Son de vital importancia para estar bien catalogados en los buscadores, se describirán con detalle en el siguiente capítulo.

Etiquetas META

Muchos buscadores, sobretodo los motores de búsqueda, **leen determinadas etiquetas en la página para encontrar la descripción y las palabras clave** que se le asocian. Estas son las **Etiquetas META**, o META Tags.

Es muy recomendable construirlas de manera adecuada, aunque no todos los buscadores hagan uso de ellas. En concreto, los índices no suelen consultar estas etiquetas, ya que extraen esta información generalmente de los formularios que rellenamos para registrar la página.

Las etiquetas META se sitúan **en la cabecera del documento HTML**, entre las etiquetas <HEAD> y </HEAD>. Como se indico anteriormente, se han de introducir con estas etiquetas tanto la descripción como las palabras clave, cada una con una etiqueta distinta.

La etiqueta META con la descripción tiene esta sintaxis:

```
<META name="description" content="descripcion de la página aquí">
```

Mientras que **la etiqueta META con las palabras clave** tiene esta otra forma, muy parecida:

```
<META name="keywords" content="palabra clave 1,palabra 2,palabra3">
```

Ahora veamos algunas recomendaciones para construir estas etiquetas.

La descripción

- Puede ser parecida al titulo, si cabe un poco más **descriptiva y siempre debe ser una frase lógica** completamente legible.
- El tamaño puede estar **entre 150 o 200 caracteres**.
- **No debemos repetir la misma palabra** en la descripción, pues pueden pensar que hacemos spam.

Una descripción adecuada para el mecánico de Vallecas podría ser la siguiente: **Taller mecánico de Vallecas, Madrid. Reparación de vehículos, lunetas térmicas, cambio de aceite y otros servicios. Distribuidor autorizado Seat - Audi - Volkswagen.**

Las palabras clave

En la etiqueta META donde deberían ir las palabras clave, nosotros recomendamos colocar **frases clave**. Una detras de otra, por orden de importancia.

- **No se debe repetir una frase clave**.
- Una misma **palabra no se debe repetir más de 5 veces**, en distintas frases claves, se entiende. Esto es debido a que si la repetimos excesivamente el motor de búsqueda puede pensar que estamos haciendo spam. Las palabras como "el", "de", "y", y otras parecidas generalmente no serán tenidas en cuenta por los motores de busqueda, y no pasara nada por repetirlas más de 5 veces.
- Como se ha visto, las frases clave han de estar **separadas por comas**. Muchos buscadores tratan a las comas de la misma manera que los espacios en blanco, con lo que si ponemos dos frases clave seguidas que tienen sentido si las leemos como si fuese una, tanto mejor. Conviene, por esto último, que no tengamos dos frases clave con casi las mismas palabras muy próximas entre si.
- El tamaño recomendado para el texto con las palabras clave está **entre 200 y 400 caracteres**. No es problema de todos modos en excederse, pues cada buscador leerá un determinado número de palabras y las del final las desechará. Lo que si es problemático es repetir muchas veces la misma palabra clave, como ya se ha dicho.

Para nuestro ejemplo, podriamos poner todo este texto como palabras clave:
taller coche Vallecas Madrid,reparación vehículo Vallecas Madrid,mecánico de automóviles Vallecas Madrid,recambios para vehículos Madrid Vallecas,servicios reparación de automóviles Madrid Vallecas,taller reparación de coches,lunetas térmicas para automoviles,cambio de aceite,mecánico y recambios para vehículos,mecánico coche servicios

automóvil

Otras etiquetas META

La descripción y palabras clave no son las únicas etiquetas META que podemos incluir en una página web, aunque si las más importantes. Otros ejemplos de etiquetas META son el autor de la página, el idioma, email de contacto, etc. Podemos ver estas etiquetas y cómo se construyen con nuestro **generador de etiquetas META**, que puede serte muy útil para crear las de tu página web sin ningún esfuerzo. En el siguiente capítulo vamos a reunir todas estas ideas y ver cómo se construiría una página web que las contemplase.

Todo el HTML

Veamos ahora un ejemplo de **página HTML que recoje todo** lo dicho anteriormente sobre el título etiquetas META y primeros párrafos.

```
<html>
<head>
<title>Taller mecanico de coches en Vallecas, Madrid, Todo Auto. Reparación de lunetas limpiaparabrisas electricidad recambios aceite</title>
<META name="description" content="Taller mecánico de Vallecas, Madrid. Reparación de vehículos, lunetas térmicas, cambio de aceite y otros servicios. Distribuidor autorizado Seat - Audi - Volkswagen.">
<META name="keywords" content="taller coche Vallecas Madrid,reparación vehículo Vallecas Madrid,mecánico de automóviles Vallecas Madrid,recambios para vehículos Madrid Vallecas,servicios reparación de automóviles Madrid Vallecas,taller reparación de coches,lunetas térmicas para automoviles,cambio de aceite,mecánico y recambios para vehículos,mecánico coche servicios automóvil">
</head>
<body>
<H1 align=center> Todo Auto</H1>
En Vallecas, situado en la carretera de Valencia a 3 km de Madrid, se encuentra nuestro taller de reparaci&oacute;n de autom&oacute;viles y recambios.
Estaremos gustosos de atenderles y ofrecerles nuestros servicios de todo tipo para su coche, como cambio de aceite, reparaci&oacute;n de lunetas t&eacute;rmicas, y otros.
<br>
<br>
Resto de la p&aacute;gina web...
</body>
</html>
```

Así es como debería quedar la página, requiere bastante trabajo, pero los resultados serán sufientes para compensarlo.

En el siguiente capítulo vamos a conocer otros trucos para el registro, que nunca debemos utilizar, pues resultan cotraproducentes para nuestros objetivos.

Falsos trucos

Además existen una serie de trucos adicionales, que se pueden utilizar y lo más seguro es que sean contraproducentes. Los destacamos aquí para que **NO los pongáis en práctica** pues pueden trataros de pagina spam, con el consiguiente perjuicio en su posición entre los resultados.

- Poner un **párrafo entre comentarios HTML** con la descripción del sitio, o repitiendo palabras clave de la página.
- Poner un **texto del mismo color que el fondo** (con lo que no se ve) con las palabras clave.
- **Repetir excesivamente** las palabras clave en el texto.
- Utilizar **palabras clave que luego no figuren** por ningún sitio **en la página**.
- En general, **cualquier técnica que sea fraudulenta** para conseguir estar mejor situados, pues poco a poco se van descubriendo estas técnicas y los buscadores las van penalizando a medida que las detectan.

Como se debe haber comprendido, al final, lo más recomendable es realizar la página, la creación de etiquetas META y demás, de una manera legal, **procurando facilitarle las cosas al buscador y sin abusar en el uso de técnicas** para situarlo más arriba en los resultados.

Esperamos, con los primeros capítulos de este manual, podais registrar páginas y que estas se encuentren bien posicionadas en los buscadores. En proximos capítulos vamos a tratar de describir otras técnicas de promoción distintas a los buscadores.

Otros métodos de promoción

Podemos hacer muchas cosas para que nuestra página sea más popular, básicamente se trata de hacer que la dirección de nuestra página figure en mayor número de sitios. Vamos a proporcionar a continuación una serie de ideas:

Primero es aconsejable que **conozcáis el medio en el que os encontráis**. Existen en Internet muchas ideas para promocionar una página, muchos lo están haciendo muy bien, incluso entre las páginas que tratan vuestro mismo tema podéis encontrar ideas para enriquecer vuestra web. Por otro lado también podéis aprender así de los errores de otras personas, y no cometerlos vosotros, por supuesto.

Poner la **dirección en la firma de vuestros correos electrónicos**. Así tendréis un posible acceso con cada correo que mandéis. Si incluís una breve descripción del sitio tanto mejor, pues puede aumentar el interés por la página.

Del mismo modo, es aconsejable que **pongáis en otros lugares la dirección** de la página. Si se trata de una empresa, por ejemplo, se puede poner el URL en el membrete del papel de oficina, en los sobres, carpetas, etc.

Daros un paseo por las **listas de correo o las news de temas relacionados** con vuestra página para sugerir la dirección. Esto se debe hacer con cautela, y solo cuando puede servir nuestra página para algo útil: No debemos presentar nuestra dirección como si fuera publicidad puesto que, por un lado, no conseguiremos que la gente se fije en nosotros y pulse en nuestro enlace, y por otro, conseguiremos el efecto contrario al deseado al crear animadversión contra la página por estar haciendo spam a los usuarios de la lista o news.

Es muy importante que nuestra página **figure en índices especializados en la temática de nuestra web.** No son páginas tan importantes como los buscadores, pero reciben visitas que pueden estar muy interesadas en nuestro sitio. Por ejemplo si es un manual, este debe figurar en todos los índices de manuales que se pueda, como http://www.ciberteca.net/ y otros más modestos que existen. Si es un recurso para crear páginas web, debería figurar en el buscador de http://www.desarrolloweb.com/ y en otras listas de recursos que existen.

Siempre es buena idea **intercambiar links con páginas que traten la misma temática**, para crear trafico de unas a otras. No hay que mirar a las demás páginas como competidoras sino como enriquecimiento de vuetros contenidos o servicios

También debemos preocuparnos de que las **páginas personales de los usuarios, que muchas veces tienen listas de links**, contengan el nuestro. Sería adecuado crear un pequeño **botón para sugerirles a estos que lo incluyan en su sitio** e indicarles cómo deben hacerlo.

Anunciarla en otros medios de comunicación. Estamos sugiriendo que se haga publicidad en televisión, radio, prensa, etc. Esto, lógicamente, no está al alcance de todos pero evidentemente sería útil, sobretodo porque esta publicidad se encuentra "fuera de la Red" y puede llegar a otro tipo de personas distintas de las que podemos llegar a través de ella.

También podemos **promocionarla fuera de Internet** con otros medios más económicos: en el pueblo donde vivo he recibido **panfletos** donde se anuncia una web de servicios relacionados con esta población. Estos no son muy caros y además en target es de lo más apropiado. También se me ocurre colocar carteles en lugares frecuentados por esta gente con la dirección de la página y las "delicias" que ofrece, como las paradas de autobús.

Se puede, y esto es gratis, **enviar notas de prensa a los distintos medios** comentando la creación del sitio. Si alguno de ellos se hace eco de esta noticia ganaríamos muchas visitas. Del mismo modo, podemos mandar nuestra dirección a **revistas de Internet**, que suelen tener una sección donde presentan webs atractivas, y en muchos casos , las páginas personales de sus lectores.

Montar una **lista de correo** para que la gente que visite la página se pueda apuntar. Así podemos conseguir una buena cantidad de personas interesadas en nuestro web y **mandarle novedades del sitio**. Con cada correo que enviamos tenemos muchas posibilidades de conseguir una visita.

También podemos esforzarnos un poco más para **crear un boletín que podemos mandar a los componentes de una lista periódicamente**. Este boletín puede contener noticias del día, reportajes, direcciones que ver...

El **intercambio de banners** es otra buena manera de incluir referencias a nuestra página. Para crear un banner debemos ser muy cuidadosos y hacerlo de manera que este sea lo más llamativo posible. Se pueden encontrar muchas páginas de intercambio en nuestro buscador. También podemos **pagar por exposiciones de nuestros banners** en otras páginas, o por cliks. Pero esto solo vale para el que se lo pueda permitir.

Otra posibilidad para conseguir visitas es estar **catalogado dentro de hits**, o lugares donde se presentan las páginas más visitadas. Normalmente para acceder a estos hits debemos registrarnos y únicamente competimos entre las páginas que también se encuentran registradas.

Para ir acabando, también se nos ocurre crear un **sistema de afiliados** de cualquier tipo. Esto es que nuestra página ofrezca dinero otros por promocinarnos de alguna forma, por ejemplo, teniendo nuestra página como página de inicio en los exploradores, pagando a los usuarios por que pongan nuestro servicio de búsqueda en su página, o por usuario registrado a nuestro servicio que viniese de su web.

Como se puede comprobar, existen **multitud de vías para la promoción** de un sitio. Nosotros hemos enumerado unas cuantas, reconocemos que algunas son un poco rebuscadas o no están al alcance de la mayoría, pero hay que contar con **otras tantas que se podrían añadir con un poco más de imaginación**.

Por último, señalar que otra labor muy importante **es seguir promocionando el web, sin parar** de hacerlo nunca. Nuevos esfuerzos en promociones del web se traducen en usuarios que te recuerdan y grupos de gente que aun no te conocían por no haber llegado a ellos el mensaje anteriormente.

En el siguiente capítulo veremos algunos consejos para que nuestro web esté preparado para asimilar todos nuestros esfuerzos en su promoción.

Consejos para el web

Es importante que nuestro web pueda asimilar todos los esfuerzos que hacemos en promocionarlo, para que estos se traduzcan en usuarios satisfechos y nuevas visitas cuando vuelvan. Al final, lo que te da tráfico de verdad en la web es que la gente vuelva a menudo a ella porque le estás ofreciendo algo que le resulta útil, y se lo estás ofreciendo en bandeja de plata.

Veamos algunos consejos que debéis seguir y nuevas ideas para vuestros webs.

Renovar la información cada cierto tiempo, incluyendo nuevas páginas con nuevos contenidos. Además es importante hacerle llegar **a la gente el mensaje de que estás dedicándote a ello**, poniendo, por ejemplo, una sección donde se muestran las novedades del sitio o gráficos indicando los recursos que son novedad en el web.

Revisar tu web en algún servicio de los que te dan informes sobre su estado, por ejemplo http://www.websitegarage.com/. Estos sitios te pueden chequear el web e informar de las cosas que estás haciendo bien o mal en variados aspectos como la compatibilidad con los navegadores, si está bien hecha para una correcta indexación en los buscadores, si tienes algún link roto, etc.

Poner en algún sitio visible una **página de contacto con el visitante**, donde puedan realizar preguntas, sugerencias u otro tipo de comentarios. No se os olvide contestar a todos los correos que recibais. En desarrolloweb hemos publicado un reportaje sobre contacto con el navegante que te puede ayudar en este sentido.

Ofrecer servicios desde tu web es otra forma de hacer que la gente vuelva a menudo. Si puedes poner cualquier cosa como un buscador, correo gratuito, un servicio de alertas, estarás haciendo que muchos usuarios vuelvan a la página para utilizarlo.

En general, como hemos ido diciendo, se trata de tener una página web sin errores clara, con buen diseño y con contenidos y servicios interesantes, es decir, **una página web cuidada al detalle**.

En el siguiente capítulo vamos a hacer una recapitulación de todo lo visto hasta ahora en este manual y a proponeros varias herramientas para la promoción.

Recapitulación

Haciendo una recapitulación de lo visto hasta ahora en este manual, vamos a proponer unas herramientas para la correcta y rápida promoción del sitio y a ofrecer en un único archivo todos los textos de los distintos capítulos que llevamos.

Servicios

En desarrolloweb.com disponemos de dos servicios útiles para promocionar tu página.

MULTIREGISTRO EN BUSCADORES www.desarrolloweb.com/promocion/multiregistro
Es una herramienta que te permite registrar una web en varios buscadores a la vez con solo introducir una vez los datos de la página. Con una clasificación de los buscadores por distintos criterios, para que te registres en aquellos que te interesan. Revisada constantemente para que todo funcione bien.

CREACIÓN DE ETIQUETAS META www.desarrolloweb.com/etiquetasmeta
Generador de etiquetas META para tu web. Para que tengas las etiquetas META en tu página web sin errores y no sea una excusa el no saber hacerlas para no tenerlas.

Cómo ayuda un dominio

Disponer de un **dominio propio** para la web que queremos promocionar puede ser de gran utilidad, ya que le dará a nuestras páginas un nombre que las identifique perfectamente en la Red, de una manera personalizada y muy profesional, **con un aire de "marca en Internet"**.

Escoger bien el nombre del dominio es fundamental, si este tiene gancho podremos ser **fácilmente recordados**, y conseguir muchas ventajas:

- Nuestros visitantes pueden volver **fácilmente a nuestro sitio** con solo introducir una dirección URL corta y sencilla.
- **Promueve la imagen** del producto, servicio o empresa en Internet y le da un valor adicional.
- Nuestra web tiene un nombre equiparable al de otras grandes empresas y proyectos en Internet. **Estamos al mismo nivel que los mejores**.
- Cuando se ve un URL que únicamente consta del dominio en cuestión en los buscadores se piensa, en un principio, que se ha encontrado una página con **mucha información** del tema que se buscaba.

Por el contrario a lo que se podía pensar, tener un dominio con las palabras claves de nuestra temática no nos va a ayudar a estar mejor situados entre los resultados de la búsqueda - para estar más arriba entre los resultados hay que seguir nuestros consejos explicados al principio de este manual -. Es decir, si nuestra dirección web es www.manuales_de_diseno_web_html_css_asp_php_java.com no tiene por que estar posicionada mejor que si se llamase www.geocities.com/petardo, ya que los **buscadores usualmente no miran en la dirección de la página para extraer las palabras clave** ni ningún otro contenido que la haga más representativa con respecto a un tema.

Para obtener más informacion sobre los dominios en Internet, los distintos tipos, métodos de registrarlos, y donde conseguir dominios gratuitos se puede consultar el reportaje **Los dominios, cómo registrarlos**, publicado en desarrolloweb.

También resulta interesante, llegado a este punto, un artículo que se ha publicado dentro de este mismo manual relativo a la elección adecuada de un nombre de dominio.

Profesionales I: Promocionar proyectos

Como dar a conocer un proyecto a los medios off-line.

Debemos distinguir 3 tipos de canales de comunicación:

- Publicidad
- Información inducida (comunicados de prensa)
- Noticias

La seducción se afianza en el primer nivel

La manipulación en el segundo.

La **credibilidad** se afianza, sólo, en el tercero.

En los 60 Mac Luhan (Marshall) lanzo la frase "El medio es el mensaje", me podéis llamar nostálgico, pero creo que es totalmente vigente.

Según el canal que se utilice no solo aumenta o disminuye el valor de la información, sino que cambia la naturaleza misma del mensaje transmitido.

Una de las especifidades de la información, a diferencia de otro tipo de producto de la economía tradicional es la siguiente:

Todo producto tiene un valor de uso y un valor de cambio; ejemplo: el agua a diferencia del oro tiene una gran valor de uso y un escaso valor de cambio.

La información sólo adquiere valor cuando se usa. Por tanto la adjudicación de valor dependerá de quien la use (o sea del receptor).

Una misma información puede tener gran valor para una persona y ninguno para otra.

Es decir, no hay forma objetiva de asignar valor a la información, ya que el valor lo da el sujeto receptor de acuerdo con sus necesidades y circunstancias.

Como no es posible probar la información sin acceder al contenido, la única referencia de valor para el receptor es la identificación que se produce entre valor del contenedor de la misma y el valor del contenido.

Dicho de otra forma, el valor de la información viene determinado por el valor de su contenedor.

Cuando el receptor al que no dirigimos es un periodista debemos tener en cuenta sus especifidades profesionales.

- El criterio informativo siempre será suyo, es un receptor activo
- Los periodistas aceptan el valor de determinadas fuentes y desprecian el de otras. *(en el argot periodístico información de "buena fuente")*.
- Están literalmente bombardeados por centenares de mensajes interesados.
- Su criterio de adjudicación de valor dependerá de la naturaleza misma del mensaje, su riqueza, su alcance, y muy especialmente de la fuente que lo transmite.

Para establecer una buena política informativa debe existir una perfecta correlación entre:

- Mensaje
- Canal
- Fuente

Cuando alguno de estos factores falla o no encaja con el resto, el sistema se cae.

Consejos para una start-up que quiera establecer una política de comunicación en medios tradicionales:

- Comience conociendo los medios de proximidad a su negocio, el periódico, la emisora local, establezca con ellos una relación de confianza.
- Localice a los corresponsales locales de los grandes medios, trabaje en la misma línea.
- Utilice la prensa del sector, siempre más sensible a esta tipología de proyectos.
- Si la situación de su negoció se lo permite contrate los servicios de un gabinete de prensa solvente.
- Desconfíe de los servicios de emisión de comunicados de prensa indiscriminados, la relación personal con el periodista es mucho más eficaz.
- Huya de los proyectos de comunicación de no se estructuren sobre los canales tradicionales de recepción de noticias. (Ejemplo, los Webs de comunicados de prensa son poco eficaces).

- Los periodistas, en nuestro país, tienen fundamentalmente una pantalla de referencia que es la que esta conectada con Agencia EFE. Cualquier medio alternativo o paralelo carece de valor.

Fiebre comunicadora en la llamada nueva economía.

El binomio sagrado: "Hacerlo bien y hacerlo saber"

Muchas veces, especialmente en el momento actual, influidos por esta fiebre comunicadora no dejamos llevar por el segundo factor del binomio.

Participando de la inflación comunicativa conseguimos que "comunicar" se vuelva un "verbo intransitivo". (Ignacio Ramonet)

A este mal uso contribuye hoy el circulo perverso que se establece entre burbuja financiera y burbuja informativa.

Estamos convencidos que cuanto mayor sea nuestra presencia mediática mayor será nuestro valor.

Entramos en una cultura en la que el simulacro se vuelve potente (incluso, no lo niego, a veces productivo)

A esta situación Freud la calificaría de delirante.

Y nos instalamos todos en este delirio colectivo, mediante el cual una imagen bien orquestada adquiere más valor (bursátil) que un proyecto sólido.

Una start-up el énfasis debe ponerlo en "hacerlo bien", basándonos en tres factores:

- Producto sólido
- Trabajo
- Músculo (conceptual y financiero)

A esto debe añadirse una gestión del proyecto eficaz y sostenida en el tiempo.

Y convertir esto en un circulo retroalimentado permanentemente.

El énfasis en comunicación en una start-up debe dirigirse principalmente al acercamiento selectivo y gradual a los inversores. (angels investors, seep capital, estructuradores financieros, private equity, venture capitalist, etc.)

En este sentido son especialmente valiosas las iniciativas como forum.org y gentemprendedora, que no sólo propician este tipo de contactos, sino que asesoran tanto en el diseño proyecto como en su estrategia de presentación a posibles inversores.

La cultura del "pelotazo" afortunadamente esta desapareciendo y emerge un nuevo sentido del valor trabajo, de los proyectos sólidos y de la gestión eficaz..

Deben denunciarse las campañas perfectamente orquestadas, en connivencia con círculos mediáticos, en las que la información tendenciosa, la imagen de diseño y el marketing, crean un sobrevalor al generar en el mercado falsas expectativas sobre determinados proyectos.

El nuevo modelo de desarrollo en el que nos encontramos llamado, interesadamente, "nueva economía" debe afianzarse en valores menos volátiles y recuperar de forma renovada el viejo concepto de la producción.

Profesionales II: Promoción en la Red

¿Publicidad on o off line?:

La publicidad off line es cara y escasamente efectiva, en un país donde el 23 % de la población no conoce ni ha oído hablar de Internet y el 60 % ha oído hablar pero no lo utiliza nunca.

Debe aprovecharse la interantividad y la posibilidades de la propia red para promocionar el negocio.

Es más rentable invertir en publicidad on –line, especialmente cuando se disponen de escasos recursos las acciones on –line permiten rentabilizar mejor la imaginación (recurso escaso pero gratuito, cuando se posee)

Tipos de promoción en la red :

Primer consejo: darse de alta en el buscador (servicio gratuito).

Los buscadores araña realizan automáticamente el alta del site, registrando utomáticamente las páginas de internet.

Problemas: el registro de nuestra web puede aparecer el número 586 de la lista de resultados y las posibilidades de ser visitado son mínimas.

Soluciones al problema:

Realizar un alta escogiendo cuidadosamente las palabras clave, la sección donde aparecerá, la descripción del web, colocar bien los meta tags y aprovechar las particularidades de cada buscador.

Actualizar las altas de forma periódica.

Pagar al buscador para mejorar nuestra posición en la lista de recursos.

MARKETING VIRAL:

Como conseguir que mi site se convierta en un virus que se reproduzca automáticamente en la red, via e-mail por ejemplo (campañas piramidales de producto Ericsson, Nokia, ofrecer correo electrónico gratuito, avisos a móvil, etc)

Que debemos tener en cuenta al diseñar una campaña en la red:

Planificación de Medios en Internet

Proceso:

Definición de objetivos y presupuesto de la campaña.

Selección de soportes.

Planificación y compra.

Seguimiento de campaña.

Reconstrucción de campaña.

Objetivos y estrategias publicitarias

Generación de tráfico

Branding

Generar respuesta directa

Venta (e-commerce)

Generación de base de datos

Creación de comunidades

Generación de tráfico

Aportación a través de campañas de banners de público al site del anunciante desde un site de alto tráfico donde se encuentre representado el público objetivo.

El banner/promoción es la primera etapa de la pieza publicitaria, ha de generar el interés de la audiencia para conseguir la acción.

Hay que considerar el conjunto banner/site como una única pieza de comunicación.

El ratio medio de respuesta (click throught) a las campañas de banners en nuestro país rondaba el 2%, ahora es prácticamente del 0%.

Branding

La publicidad on-line es una poderosa herramienta de branding, es evidente que la audiencia que no interactúa con la publicidad on-line está recibiendo impactos publicitarios tan útiles como los producidos por los demás medios tradicionales.

Un estudio sobre la efectividad de la publicidad "on-line" realizado por iab en 1997 indica que las marcas que realizan publicidad en la red incrementan en un 30% el grado de conocimiento de la marca entre los usuarios de la misma y que la percepción sobre la marca puede mejorar un 55%.

Respuesta Directa

A través de campañas de publicidad podemos conseguir respuesta directa de la audiencia en tiempo real.

Es una gran solución para conocer la opinión sobre un producto o un tema concreto por parte del público objetivo.

e-commerce

Realizar el proceso de la venta de forma directa (publicidad/presentación de producto/venta) y en un solo paso.

Podemos realizar campañas de publicidad dirigidas a aportar compradores a una solución de comercio electrónico o que todo el proceso se realice en el banner

Objetivos y estrategias publicitarias

Promociones Puntuales

Los site de tercera generación son los sites de producto (los sites institucionales son cada día menos utilizados). microsites que tienen una vida muy limitada y que presentan un producto o servicio.

El extremo máximo de este tipo de sites son las campañas banner/site creados para difundir una promoción puntual.

El bajo coste de producción y la velocidad de la respuesta de dichas campañas hacen de internet un medio insuperable para este tipo de promociones.

Generación de Base de Datos

El coste por registro de una base de datos generada en internet es realmente competitivo (alrededor de 400.- ptas).

Es muy importante la creación de dichas bases de datos para crear sistemas de fidelización de usuarios a nuestro site o simplemente para realizar ofertas puntuales a través del correo electrónico.

Profesionales III: Tipos de publicidad

Tipos de Publicidad on-line:

El Banner

Efectividad en claro descenso.

El formato estándar de la publicidad en internet.

A esta pieza publicitaria se le pide en la mayoría de los casos que genere tráfico hacia el website del anunciante.

Hay que valorar su potencialidad como herramienta de branding.

El formato más utilizado es el gif animado y las medidas 468x60 píxeles

Banners

Elementos que influyen en el éxito de la campaña:

La creatividad.

La planificación de la campaña.

El conjunto campaña de banners/website

Según el estudio "banner ad location efectiveness" (1997 athenia associates) hay dos localizaciones preferentes:

- En el tercio superior de la página
- En el lateral derecho

El mismo estudio indica que más de un banner en la misma página no mejora el ratio de click. se divide entre dos creatividades el número de clicks.

Creatividad del Banner

Mensajes cortos claros y legibles.

Crear un sentido de urgencia.

Mensajes enigmáticos.

Formular una pregunta.

Utilizar mensajes como "gratis", "gana"...

Invitar al clic.

Integrar en el mensaje del banner la invitación a la acción.

Animación de los banners.

Múltiples creatividades.

Planificación de la Campaña

Siempre que sea posible, personalizar el mensaje para cada uno de los targets a los que nos dirigimos y teniendo en consideración las posibilidades de segmentación de los soportes

Nuevos formatos.

Para contrarrestar la caída del porcentaje de respuesta de los banners en formato tradicional (gif) aparecen en el mercado nuevos formatos que aumentan la interactividad y notoriedad de los banners.

Estas tecnologías son:

banners html y dhtml
banners flash
banners en java

Aparecen nuevas propuestas publicitarias en la red que intentan potenciar la notoriedad como su efectividad de respuesta.

Estas propuestas en algunos casos topan con el problema de la falta de ancho de banda.

En otros casos, pueden ser considerados por los internautas como demasiado intrusivos.

Estos formatos son las pop-up windows, los intersticials, los banners desplegables o los cursores personalizados

Ventanas flotantes

Aumentan la notoriedad presentándose en un navegador nuevo.

Problemas en navegación lenta.

Intersticials

El **interstitial** intenta recrear el spot televisivo en internet.

Se enfrentan al problema de la lentitud en internet.

Pueden ser considerados por el usuario como demasiado intrusivos.

Banners Extensibles.

Una de las últimas propuestas aparecidas en la red.

El banner dispone de la posibilidad de extenderse y disponer de más espacio para plantear la oferta o diversificarla antes de remitirte a la página del anunciante.

Cursores Animados.

Introducir el mensaje o animaciones del anunciante en los cursores del ordenador del usuario.

Este sistema puede reforzar la notoriedad de banners, patrocinios....

Comet systems es la compañía que los ha lanzado al mercado.

Acciones especiales:

Ventajas del Patrocinio

Potencia la imagen de marca del anunciante.

Ofrece información de valor añadido al usuario e integrarla en un entorno que conoce.

Conoce con más exactitud el perfil de la audiencia que nos visita.

Crea bases de datos de prospects.

Audiencia asegurada desde el primer día.

Posibilita el acceso a anunciantes que no disponen de web site.

Objetivos:

Generar tráfico al site

Construir imagen de marca

Potenciar la interacción con los usuarios.

Potenciación de los contenidos de la misma.

Con una campaña de banners en los principales websites españoles, atrayendo tráfico no a las páginas del anunciante, sino a la sección patrocinada, que se hallaba en otro website.

Por contenidos del web site.

Las temáticas del site nos permiten intuir el tipo de público que visita la página.

Es UN sistema mUY utilizado actualmente.

Los costes de inserción aumentan al comprar sites con públicos muy definidos.

Existen sites nicho donde podemos encontrar targets muy definidos. un ejemplo claro sería un site de contenido para aparejadores y arquitectos.

El idioma del site nos puede segmentar públicos en ocasiones.

Por usuario registrado / datamining.

Los sites donde los usuarios están registrados permiten dirigir campañas a perfiles sociodemográficos muy precisos.

Ejemplos de estos tipos de site son msn y financial times

El seguimiento de los perfiles de los usuarios que se están realizando por parte de las grandes redes publicitarias permitirán próximamente comprar perfiles de usuarios independientemente de los contenidos del site que visite el usuario. Por el momento, estos sistemas de datamining no permiten procesamientos suficientemente rápidos para permitir servir el banner en décimas de segundo.

Adserving.

Los servidores de publicidad identifican al usuario que accede a un site y le sirven los banners dirigidos a su perfil.

De esta forma se sirve la publicidad adecuada a cada usuario.

Los adservers permiten la compra por impresiones, utilizar más de un banner en la campaña y conocer los datos en tiempo real de la evolución de la campaña.

Algunos adservers están auditados por terceras partes (abc electronics, net ratings).

Adserving.

Las segmentaciones que permiten son:

- País (dominios o sistemas mixtos ip/dominio)
- Día y hora
- Dominio
- Sistema operativo
- Navegador
- Por banner
- Frecuencias...

Cookies.

Fichero de texto instalado por el servidor web en el navegador que contiene información sobre las preferencias del usuario y otros datos diversos, que activan ciertas respuestas en los sistemas a los que se conecta. Es el sistema que utilizan los servidores de publicidad para identificar máquinas únicas, activar el sistema de frecuencias o recopilar información sobre el perfil del usuario de la máquina. (http://www.cookiecentral.com/)

Profesionales IV: Mercado y soportes.

mercado y selección de soportes

Grandes sites con departamento comercial propio (yahoo, olé!, ciudades netropolis...)

Versiones on-line de medios de comunicación tradicionales (el país, recoletos, idg, grupo correo...)

Redes publicitarias, exclusivistas que aportan a sites medianos/grandes tecnología de adserving y equipos comerciales para la comercialización de los espacios publicitarios (doubleclick...).

Formas de Compra:

Espacio tiempo:

Contratar un banner durante un mes en la página principal de un site.

No hay garantía de audiencia.

No hay segmentación

No se dispone de datos sobre la evolución de la campaña.

formato poco utilizado.

Por Impresiones:

Una impresión = un banner cargado en un ordenador.

Sólo se paga por la audiencia que ve nuestro anuncio.

Posibilidad de segmentación (por país...)

Resultados de campaña en 24h.

Formato de compra más utilizado.

Coste de la impresión de 3 a 10 pesetas.

Por Click:

Un click = un usuario pulsa sobre el banner y visita la página del anunciante.

Sólo se paga por la audiencia que va a nuestro site.

Posibilidad de segmentación (por país...)

Resultados de campaña en 24h.

Formato de compra poco utilizado.

Coste del click de 160 a 200 pesetas.

Por transacción:

El site cobra un porcentaje por las ventas generadas por las campañas publicitarias.

Existen soluciones de adserving que permiten controlar este tipos de campaña con total transparencia.

Seguimiento, control y valoración de los resultados de campaña

Acceso a las estadísticas on-line:

Diariamente se actualizan los datos de la evolución de las campañas.

En el caso de campañas contratadas site a site es importante valorar los resultados de forma individualizada en cada uno de los soportes e intentar optimizar los resultados tras disponer de datos con suficiente peso estadístico.

Acceso a las estadísticas on-line:

Hay que valorar los ratios de respuesta según el día de la semana, la hora del día y según los diferentes banners.

La mayoría de los sistemas permiten realizar reports específicos con estas variables y realizar modificaciones de campaña en 24h.

Informe final de campaña:

Los informes de final de campaña nos permiten valorar si hemos conseguido los resultados definidos en la estrategia de campaña.

La valoración se ha de realizar de forma conjunta con la información del servidor de publicidad, el análisis de los logs y los resultados cuantificables de respuesta directa.

Campañas de éxito:

variables

Tráfico generado hacia el site (cuantitativamente y cualitativamente).

Prospects generados. calidad de los mismos.

Ventas generadas.

Comparación del tráfico tras la campaña con los datos anteriores al inicio de la misma.

Popularidad de un sitio web

La popularidad de un sitio es un concepto muy amplio, en general mide lo conocido que sea entre los internautas. En el caso de páginas especializadas, como desarrolloweb, su popularidad significa lo conocido que sea entre los usuarios que se interesan por el desarrollo de páginas.

Cuando nosotros promocionamos un sitio procuramos que este sea conocido, buscamos su popularidad. En muchos casos, el simple hecho de que nuestro sitio sea popular nos ayudará también a estar mejor considerados entre la oferta de sitios de Internet. Los buscadores nos admitirán las páginas con mayor probabilidad y nos posicionarán mejor entre los resultados.

Por todo esto, debemos dar una imagen de "sitio popular", esto se consigue definiendo una buena imagen de marca y, por supuesto, ofreciendo buenos contenidos y servicios.

Los buscadores ponderan según la popularidad

Existe una nueva generación de buscadores que ponderan los resultados de sus búsquedas en función de lo popular que sea un sitio. Éstos realizan un sencillo cálculo que consiste en contar los enlaces que tienen otras páginas hacia un sitio web determinado. Los sitios que tengan más enlaces dirigidos hacia ellos tendrán una puntuación más alta y saldrán más arriba en los resultados.

Pongamos un ejemplo: buscando la página de la petrolífera Repsol-Ypf podríamos introducir repsol ypf en el buscador. Éste, buscará entre sus páginas y extraerá muchas webs con estas palabras. Una de ellas será la compañía que buscamos y seguro que habrá muchas páginas que no sean la web corporativa que estamos buscando, pero muy probablemente contendrán enlaces a la página de Repsol-Ypf. Así, podremos imaginar que la página que estará más enlazada de todas las páginas de los resultados será la web corporativa, por lo tanto saldrá en el primer resultado de la búsqueda.

Medir los enlaces que hay hacia tu sitio

Para medir los enlaces que hay en otras páginas hacia tu sitio debes utilizar los propios buscadores. En muchas ocasiones tienen una herramienta de búsqueda avanzada o alguna ayuda que explica como puedes hacer esta búsqueda. Por ejemplo, Altavista incluye esta opción y para utilizarla deberemos buscar por **link: seguido por la dirección que quieres contar el número de enlaces**, de esta forma: ***link:http://www.tudominio.com***

También existe la posibilidad de hacer esta búsqueda en http://www.linkpopularity.com/ a través de una página que te devuelve el número de páginas que te enlazan en distintos motores de búsqueda.

Cómo tener un web-negocio y no morir en el intento

Internet y las nuevas tecnologías han supuesto para muchas personas emprendedoras y/o PYMES un nuevo modelo de negocio. Al principio todo parecía ir bien y 'viento en popa": eran los tiempos del boom de la nueva economía y la implantación de grandes e-empresas que gastaban ingentes cantidades de dinero en publicidad, marketing,... prometiendo conquistar un nuevo "el dorado". Actualmente **la crisis del sector, la bancarrota de la bolsa y los nuevos valores tecnológicos y la desilusión de los inversionistas está produciendo un receso en el desarrollo de web-negocios** exitosos.

Hoy día crear una web con la finalidad de convertirla en una empresa capaz de generar los suficientes ingresos como para vivir de ella se ha convertido en algo sumamente complicado, así que ahí van algunas sugerencias para todos aquellos emprendedores/as que tienen una idea, quieren llevarla a Internet... pero no quieren morir en el intento.

1.- **Reunir una cantidad de dinero suficientemente importante como para afrontar, al menos, dos años de gastos**. Esa idea de que en Internet "de la noche a la mañana" uno es rico y puede irse a

las Bahamas mientras los ordenadores trabajan por él/ella es una falacia. Sin dinero previo no hay nada que hacer.

2.- **Disponer de una idea ligeramente diferente a lo ya existente pero no extremadamente distinta**. Si vas a crear algo, que sea nuevo, o que tenga algo diferente a la competencia, pero no crees algo tan nuevo que no lo entienda nadie. ¿Dónde hay más consumidores/clientes?, ¿qué es lo que necesitan/piden?, pues ahí hay que aplicar la idea.

3.- **Contar con un equipo que te complemente**. Si eres un especialista que ha desarrollado un innovador programa pero no cuentas con una persona que se encarge del área administrativa y de marketing, tienes tres meses de vida. Nadie puede hacer por sí solo todo completamente bien, y eso es justamente lo que le pedirán sus clientes: quiero lo mejor al mejor precio.

4.- **Disponga de un saquito con bastante dinero para invertir en publicidad y marketing**. En Internet cada vez hay más productos/servicios y hay que "hacerse ver". Ni el mejor producto/servicio puede triunfar si no se dedica una cantidad de dinero mensual a la publicidad. Las estrategias ya las conocemos: banner, newsletter, email, offline, patrocinio,...

5.- **La atención al cliente es fundamental**. Ante precios similares, su cliente siempre preferirá estar con alguien que sabe atenderle correctamente (aunque sea un poco más caro): en Internet esto quiere decir "ahora mismo". Por supuesto, además cuentan todas las normas de cortesía y management que podamos aprender.

6.- **Tenga paciencia** si en los primeros meses "no vende una rosquilla". Probablemente todavía no sabe nadie que usted (su web) existe. Sea perseverante y márquese un plan de trabajo con submetas. Esto le ayudará a valorar si la evolución que sigue es la correcta.

7.- **Procure ser el mejor en lo suyo pero no se especialice demasiado**, a no ser que disponga de muuucho dinero. Ofrecer un producto/servicio de calidad, ajustado en su precio y con algo distintivo es vital pero además procure rodear su venta con otros "añadidos". ¡Ojo! Estos añadidos deben ser de interés para el cliente aunque a penas le generen beneficio a usted. Un cliente satisfecho es el mejor agente comercial que podemos tener.

8.- Si ha realizado todo lo anterior ya habrá pasado más de un año por lo que estará en el momento crítico: invierto un poco más o "tiro la toalla". No lo dude: **siempre mire hacia delante**. Es la etapa crítica para establecer con solidez el futuro de su web-negocio. Porque en Internet al igual que en el mundo tradicional, crear un web-negocio es fácil, lo difícil es mantenerse, continuarlo en el tiempo.

El banner perfecto

Resumen.

1. El banner funciona. Si bien como soporte de un modelo de negocio esta cuestionado por el ingresos que es capaz de generar, como medio de comunicacion es algo que se debe aprovechar.
2. En este articulo veremos puntos clave a la hora de diseñar nuestro banner para lograr una comunicacion efectiva. No loop, si animaci—n, menor peso, mas grande.

1. El banner funciona.

En un estudio de Site Usability se hace referencia a la ceguera al banner. El estudio muestra que los usuarios no son tan ciegos al banner como parece y la impresion del mensaje es efectiva.

Recuerdo del banner:
43% algo.
57% nada.

Fuente: Site Usability.

Desde el punto de vista del anunciante, el banner puede funcionar. Cada vez internet resta mas tiempo a la television (evidentemente mas que los periodicos y revistas) y en internet se encuentran nichos de mercado que en el mundo real son dificiles de seguir.
Por otro lado, en internet tenemos la oportunidad de acceder a un rango de poblacion mas amplio que un medio con limitaciones geograficas, temporales, etc.

Incluido el 27.5.2001: Los datos de Site Usability sobre la ceguera al banner despertaron dudas por parte de algunos lectores por lo que se hizo una encuesta propia para saber hasta que punto este dato era cierto. La encuesta que hice da que un 25% de los usuarios recuerdan algun banner de su ultima sesion. Est dato da muy pocas esperanzas al banner actual con lo que debemos pensar en ir cambiando los banners. Otra idea es ir haciendo las paginas que dependen de ingresos por publicidad mas "ligeras" para que el banner pueda brillar lo mas posible y capte la atencion del usuario.

2. No siempre debe llevar a una pagina.
El dato que se toma para medir la eficacia de un banner es el click through. Esto es correcto pero no simpre es valido.

Los anuncios sirven para ofrecer informacion, posicionar o darle valor a una marca y para despertar la necesidad de adquirir un determinado producto. El click through esta una dimension mas alla de estos datos ya que ofrece acceso instantaneo a la adquisicion de este producto.

El problema es que muchos productos que se anuncian en banners no se pueden adquirir directamente. Un banner sobre el estreno de una pelicula pude ofrecer informacion sobre el dia del estreno y como opcion el visitar la pagina. En este ejemplo, la efectividad del banner no se mide solamente por el click through. La impresion correcta del mensaje es suficiente para justificar la inversion.

Este banner sirve para comunicar la fecha de estreno de la pelicula y como opcion de enlace nos invita a ver los grupos de discusion.

3. Debe llevar a una pagina de interes.
El enlace que contiene el banner debe llevar a una pagina diseñada especificamente para este fin. Si el banner lleva a la portada, el usuario puede sentirse confundido o no encontrar aquello que vio en el banner.

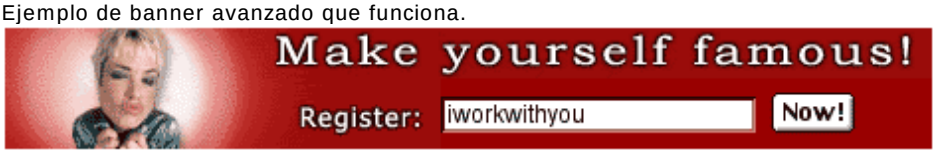
Si lo que estamos comunicando es la existencia de nuestro site, se puede preparar una pagina de bienvenida especial donde se explica el proyecto, objetivos, etc.

En el caso de este banner de Mixmail.com, pese a invitarnos a darnos de alta en su servicio, el banner te lleva a la portada general.

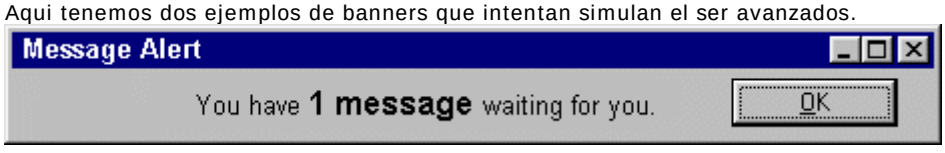
4. Banners avanzados son interesantes pero la gente no los usa por que no sabe si debe pinchar o usarlo.
Estos banners que contienen combos, campos de texto o botones, donde el usuario puede realizar parte de la transaccion antes de ver la pagina. Estan bien en concepto, pero el problema es que mucha gente abusa de este tipo de banners, engañando al usuario haciendo parecer botones del sistema, etc.

No son muy recomendables.

Ejemplo de banner avanzado que funciona.



Aqui tenemos dos ejemplos de banners que intentan simulan el ser avanzados.



5. El peso importa.

En general haciendo un bonito gif, el banner no deberia pasar de 4kbs.

Esto es evidente, pero hay que recordarlo. Haz tu banner ultra ligero. El modelo google de banners en formato texto es el paso mas avanzado en este campo. Quizas sea algo extremo, pero en el mundo google funciona (academido, profesional, usuarios avanzados). Personalmente creo que es otra dimension de la publicidad a la que mucha gente no llega.

Este es el ejemplo de banner que se puede encontrar en google.com

Imagen reducida, este enlace permite verla a tamaño real.

Si bien este modelo de publicidad es demasiado avanzado, lo que si hay que destacar de google es la forma de comprar publicidad. Todo en formato web bien sencillo. Aqui puedes ver la opinion de gente que ha revisado este sistema.

6. Animacion si, pero procura fijar el mensaje desde el primer momento.
Que el gif este animado es parte de la "sorpresa" que ayuda a fijar el mensaje. Procura que la animacion ocurra en una parte del banner mientras el resto permanece estatico ofreciendo claramente el mensaje que quieres transmitir.

Pese a esto no se deben hacer loops y se debe fijar el mensaje en el primer frame.

Los loops hacen que el mensaje se distorsione al perder la identificacion de "conclusion" e "inicio". Si la gente no sabe donde esta el final, no entenderan cual es el sentido del banner con lo que parte de la efectividad se puede perder.

Desde el primer frame se debe fijar el mensaje. Muchas veces la parte superior de la pagina esta presente unos segundos hasta que el usuario hace scroll o cambia de pagina. No pierdas esos segundos contado una historia.

Aqui tenemos un par de banners donde la animacion que conduce al mensaje es muy larga.

Estos dos banners de Marca (periodico deportivo español) fallan en casi todos los puntos que hemos visto hasta ahora. Son un loop inifinito, no fijan el mensaje, la duracion es exesiva, etc.

7. Procura que el mensaje no solo sea el logo.
Todavia quedan algunos banners donde el frame (entiendase como fotograma) final es el logo de la compañia anunciante. Esto es un error. El frame final debe servir para ofrecer toda la informacion necesaria para trasmitir el mensaje. El incluir la url suele ser parte de este mensaje.

El frame final puede ser imprimido, enviado por mail y debe seguir transmitiendo el mensaje con claridad.

8. El banner puede ser un soporte de modelo de negocio.
Lo que falla ahora mismo es el formato. Banner que no son rectangulares y que no estan en la parte superior de la pagina.

La busqueda de formatos y tecnicas que permitan crear una mejor impresion del mensaje en el usuario es un camino a explorar. Tambien hay que contar con la dimension que internet añade donde el mensaje se puede continuar hasta la compra directa del producto.

En un estudio de webreference se vio como colocando el banner un tercio mas abajo de lo normal, el click through se incrementaba un 77%.

Seguramente banners gigantes con flash tipo CNET sean un camino a seguir. Permiten explorar internamente el banner, ofrecen mas informacion y no hace falta abandonar la pagina en la que estamos

con lo que podemos seguir con nuestra tarea.
Lo que tiene de bueno este banner es el concepto de "poner un trozo de mi web en la tuya". El banner ya no es un anuncio, si no que pasa a ser un trozo de mi web donde te puedes completar una accion sin abandonar el site en el que estas actualmente. Esto tiene mucho sentido si pensamos en la compra de productos. No hace falta abandonar el site en el que aparece el anuncio, puedo comprar el producto directamente sobre el anuncio.

Enlaces relacionados:

- Site Usability. http://wsupsy.psy.twsu.edu/surl/usabilitynews/2S/banners.htm. Somos ciegos a los banners.

- Webreference. http://webreference.com/dev/banners/onethird.html. Resumen: Based on the above results, we conclude that the placement of the ad 1/3 down the screen, increased click-through 77% (for the Photodisc ad). The Webreference ad showed the same trend, but the result was not statistically significant.

- Useit. http://www.useit.com/alertbox/9709a.html. Resumen: "... only the top 0.01% of websites can generate sufficient revenues from advertising: in the larger picture, advertising is almost irrelevant for the success of the Web."
- Useit. http://www.useit.com/alertbox/9704b.html. Resumen: "... In fact, the model predicts that the largest website will run at a rate of about 200 billion pageviews per year by the end of the Year 2000... "

- BannerTips. http://www.bannertips.com/bannerdesign.shtml. Banners design. Consejos de todo tipo.
- BannerTips. http://www.bannertips.com/makersForHire.shtml. Listado de empresa que recomiendan para crear tu banner.

- Listado de los banners mas imprimidos.
http://209.249.142.27/nnpm/owa/NRpublicreports.topbannerweekly

La inteligencia emocional aplicada a Internet

Para lograr el éxito en Internet hoy día no basta con poseer un alto coeficiente intelectual (CI) y unos conocimientos técnicos muy grandes. Los cambios del siglo veintiuno están siendo testigos del predominio del coeficiente emocional (CE), un concepto nuevo que incluye el autoconocimiento y autodominio, el celo y la persistencia, la capacidad de motivarse uno mismo y de lograr resultados en conjunto con otros.

Todavía quedan algunas personas que desechan del todo la importancia de las emociones; las ven como un campo minado que se debe evitar a toda costa. En muchos casos son precisamente esas personas que, con su énfasis en los números fríos y el balance de utilidades, están más desconectados del motor del corazón que impulsa al capital humano y produce el trabajo creativo exponencial que se requiere para que una organización vaya a la cabeza en su campo en Internet.

La Inteligencia emocional es una forma de interactuar con el mundo que tiene muy en cuenta los sentimientos, y engloba habilidades tales como el control de los impulsos, la autoconciencia, la motivación, el entusiasmo, la perseverancia, la empatía, la agilidad mental, etc. Ellas configuran rasgos de carácter como la autodisciplina, la compasión o el altruismo, que resultan indispensables para una buena y creativa adaptación social.

Por otra parte no hay que olvidar que el éxito en Internet depende del más fundamental de todos los principios comerciales: satisfacer al cliente/visitante. Para ello, teniendo en cuenta las claves que nos aporta el estudio de la Inteligencia Emocional, los aspectos que debemos tener en cuenta son los siguientes:

1. **Confianza**. Generar la sensación de controlar y dominar el web site, la propia evolución y los contenidos aportados. La sensación de que los servicios/productos tienen muchas posibilidades de éxito y que usted cree en ellos debe transmitirla a sus clientes, tanto desde la propia página web como en los mensajes de correo electrónico.
2. **Curiosidad**. Tener en cuenta la sensación de que el hecho de descubrir algo es positivo y placentero. Aporte algo nuevo o una nueva vision del algo ya existente o un modo de interactuar/probar el servicio que desea vender: habrá despertado curiosidad.
3. **Intencionalidad**. Mostrar un claro deseo y la capacidad de lograr algo y de actuar en consecuencia. Esta habilidad está ligada a la sensación y a la capacidad de sentirse competente, de ser eficaz. Tenga claro cuál es su objetivo, su intención con el web y busque la manera de ser competente en su desarrollo.
4. **Autocontrol**. La capacidad de modular y controlar los productos/servicios en una forma apropiada; la sensación de control interno. ¿Es usted quien controla la evolución de sus

servicios o depende de las fluctuaciones de variables externas?. Cuanto más control tenga sobre sus productos/servicios más fácil será satisfacer al cliente.

5. **Relación**. La capacidad de relacionarse con los clientes, una capacidad que se basa en el hecho de comprenderles y de ser comprendido por ellos. Póngase en el papel del cliente y piense como él ¿Qué pediría, qué necesitaría?. Fomente una relación lo más personal posible (hay que hacer humano un medio que es totalmente anónimo).
6. **Capacidad de comunicar**. El deseo y la capacidad de intercambiar verbalmente o por escrito ideas, sentimientos y conceptos con los demás. Esta capacidad exige la confianza en los clientes y el placer de relacionarse con ellos. Hable, comuníquese, no se esconda tras un perfil. Los demás le quieren oir y saber lo que piensa.
7. **Cooperación**. La capacidad de armonizar las propias necesidades con las de los clientes. Recuerde que usted buscar una ganancia (económica) pero el cliente también persigue un beneficio. Y nunca olvidad el soporte y ayuda post-compra.

A nivel personal, todo administrador de un web site debe plantearse la urgencia en "alfabetizarse" emocionalmente. Algunas propuestas que pueden servirle de reflexión son:

- **Autoconciencia emocional**: El conocimiento de nuestros sentimientos y sus causas nos ayuda a mejorar modelos de conducta y relaciones.
- **Automotivación**: No esperemos siempre estímulos externos. La capacidad de generarlos nos hará más independientes y capaces en situaciones adversas.
- **Intuición**: Desarrollemos nuestra propia y natural capacidad para captar e interpretar las cosas. Confiemos más en ella.
- **Toma de decisiones personales**: Nuestra opinión nos hace únicos. Atrevámonos a ser un poco más nosotros mismos.
- **Conciencia de uno mismo**: Sólo desde ella, podremos tomar auténticas decisiones y vivir de forma consciente y adulta.
- **Capacidad de manejar el estrés**: Fundamental en la vorágine de muchas vidas cotidianas.
- **Empatía**: Captar la sintonía con otras personas, aprender a ajustar el ritmo según el interlocutor, adaptarse... ayudará en cualquier contexto de relaciones humanas.

Las siete reglas de Oro de un dominio

¿Qué hace de un dominio un gran dominio? ¿Cómos saber si el nombre de dominio que estoy pensando en comprar o registrar es realmente el adecuado para mí? Muy fácil, sólo hay que tener en cuenta estas siete reglas de oro.

1. Ponlo fácil. Ésta es la gran regla básica en el mundo de los dominios en particular y en el del márqueting y comunicación en general. A ningún publicista se le olvida la conocida regla KISS (Keep it Simple Stupid) y desde luego nuestro nombre de dominio es la primera forma de publicitar nuestro negocio en la red.

Hay que intentar que nuestro nombre de nuestro dominio le resulte fácil de recordar al usuario. El motivo es bien sencillo. Imaginémosnos que, navegando por la red, nos encontramos con dos páginas que nos encantan: eco.com y emeronion.com. ¿Cuál es más probable que volvamos a visitar?

2. Lo breve... dos veces bueno. El dicho se cumple una vez más en el mundo de los dominios. Cuantos menos caracteres tenga un nombre, mucho más fácil de recordar y también más cómodo y rápido de teclear. Desgraciadamente, estos dominios son ya muy escasos. El 100% de los dominios con tres caracteres o menos hace tiempo que está registrado. Comprarlos puede ser una solución pero los dominios breves son, sin duda, los más valisosos por lo escaso y, por tanto, los más caros.

3. Dando sentido. Varios estudios han demostrado que las palabras con significado se recuerdan prácticamente un 80% más que aquellas que no lo tienen. Buscar un nombre que realmente diga algo puede resultar complicado pero merece la pena. Es cierto que hay empresas como Yahoo que han conseguido grandes resultados con un nombre sin significado pero ¿cuánto dinero han tenido que invertir en publicidad?

4. Sin confusión. Uno de los problemas comunes con los que se encuentran las empresas de internet es el del tráfico desviado desde su página a causa de un error de tecleo por parte del usuario. Los juegos de palabras ingeniosos pueden resultar muy divertidos y útiles para captar la atención pero, en general, son poco efectivos en la red y mucho menos si utilizamos números y guiones. El significado del dominio "love-2u.com" puede ser muy sencillo de recordar pero el usuario se encontrará con la duda de si debía escribir "lovetoyou.com" o "loveto-u.com" o quién sabe cuántas combinaciones más.

5. ¿Somos .com? A la hora de escoger una extensión para nuestro dominio el dilema es el siguiente: ¿qué dominio se adapta mejor a nuestro mercado? Está claro que el estándar de facto en la red es el .com. Ésta es sin duda la extensión más codiciada. Pero, si no operamos a nivel internacional, también puede resultar una buena idea pensar en los dominios regionales de nuestro campo de actuación: .es, com.ar, com.mx, etc...

Otra alternativa son las extensiones .net y .org consolidadas desde hace tiempo aunque con bastante menos tirada que la .com. Si decidimos tener algo de paciencia, también podemos apostar por los .info o .biz que, aunque todavía tienen poca implantación, prometen ser importantes en un futuro.

6. ¿Qué dice de lo que hacemos? Ajustar el nombre de nuestro dominio a nuestra actividad es muy conveniente para conseguir que el usuario nos asocie con un determinado producto o servicio, y de ese modo, se pueda acordar de nosotros cada vez que lo necesite. Calor.com quizá es un gran nombre de dominio pero, si el negocio para el que que lo utilizamos es una tienda de zapatos on-line, probablemente no nos va a resultar de mucha utilidad.

7. Sin malas lecturas. El último aspecto a tener en cuenta a la hora de elegir un dominio son las connotaciones asociadas al nombre que elijamos. Es importante asegurarse de que nuestro dominio no tiene connotaciones negativas que puedan quedar asociadas a nuestros productos. Y si operamos en un entorno internacional tenemos que asegurarnos de que esto no sucede con nuestra lengua ni tampoco con el resto que podamos considerar importantes para nuestro negocio. Desde luego, "estohuele.com" no sería precisamente la elección más adecuada para nuestra tienda de perfumes.

Errores frecuentes al redactar el título de una web

Casi todos los buscadores consideran la "MetaTag Title" como la más importante del código HTML de una página web.

La MetaTag Title es además la primera información que aparece cuando se realiza una búsqueda en la mayoría de los grandes buscadores, haciéndola mucho más relevante, ya que es la primera información que el Internauta ve acerca de nuestra página web y decidirá hacer click o no dependiendo de su contenido.

Hay muchas páginas web que utilizan fabulosos títulos para crear una gran campaña de marketing en los buscadores y recibir tráfico completamente gratuito, pero también hay páginas web que ni siquiera tienen un título y se muestran como "Untitle". ¿Quién va a hacer click en un documento donde la información que aparece es simplemente "Untitle"?

Que no debes hacer al redactar un título

1- Nunca utilices una lista de palabras claves separadas por comas. Es díficil de leer y no es profesional. Debes de redactar una frase lógica utilizando siempre las palabras clave.

2- No utilices palabras complicadas y difíciles, a no ser que tu página web sea de un tema muy específico y busques tráfico selecto.

3- No repitas en exceso tus palabras clave, una vez es suficiente.

4- No utilices siempre mayúsculas en el título, es difícil de leer. Utiliza las mayúsculas sólo cuando correspondan.

5- Dedica tiempo a redactar un título que sea efectivo, exactamente igual a cualquier campaña de publicidad.

6- Redacta un título corto y fácil de leer. El título es como un anuncio de tu página web.

7- No olvides incluir el título en TODAS tus páginas interiores y si son títulos diferentes mucho mejor. La "MetaTag Title" debe ser siempre la primera del código HTML.

8- No utilices únicamente el nombre de tu empresa en el título. Si quieres incluir el nombre de tu empresa debe de ir siempre acompañada de tus palabras clave más importantes.

9- No utilices caracteres como !!!! o AAA para aparecer el primero de la lista en orden alfabético. Este recurso ha sido altamante utilizado y puedes ser penalizado por el buscador.

10- Por último no cometas faltas de ortografía ni palabras mal escritas que den aspecto poco profesional a tu página web. Dedica tiempo a redactar el título de tu página web.

¿Son eficaces los sistemas multiregistro en buscadores?

Un sistema multiregistro es un programa o aplicación web con el que podemos registrar una web en muchos buscadores introduciendo una sola vez los datos de la página, en lugar de ir buscador a buscador introduciendo los mismos datos para realizar todos los registros.

El registro en buscadores es una tarea muy importante si se quiere hacer una buena promoción de la página, puesto que no es lo mismo aparecer, entre los resultados de la búsqueda, en un puesto que 100 posiciones más abajo. Por esta razón, deberíamos ser muy concienzudos en el registro, eligiendo las palabras clave y la categoría del posible directorio donde incluir nuestro sitio, por lo menos en los buscadores más importantes, como Yahoo.

Sin embargo, existen cientos de buscadores donde querremos figurar, aunque puede que nos importe un poco menos la colocación entre los resultados... y puede que haya cientos o miles de buscadores donde querremos figurar...

Así pues, el dilema estaría entre:

- Registrar tu página a mano, asegurándote de la calidad del registro, pero utilizando mucho tiempo.
- Registrar tu página en poco tiempo, pero con una calidad menor en el registro y la colocación de la página en el directorio del buscador.

La recomendación sería hacer a mano los registros más importantes y tener una herramienta para realizar un registro masivo de la página web. Elegimos los directorios donde registrarnos a mano -de 10 a 20 buscadores- y ponemos en marcha un multiregistro para los cientos o miles que faltan.

El hecho de que en muchos buscadores haya que introducir la página en una categoría y que estas categorías varíen mucho de un buscador a otro es otra de las razones por las que un multiregistro automático puede resultar problemático. Como podéis imaginar, a los buscadores les llegan cientos de páginas cada día y si ven algunas que llegan automatizadas, donde la página no haya sido bien encajada entre las categorías, pueden desestimarla, dejando de lado su registro. Teniendo en cuenta que los sistemas multiregistro no pueden mantener al día todas las categorías de todos los buscadores, la idea de registrar la página manualmente, por lo menos en los buscadores más importantes, parece todavía más razonable.

Tipos de multiregistro y su eficacia

Interfaz web
Tenemos un directorio donde podemos encontrar varias de estas herramientas gratuitas. Son páginas web donde el creador se ha preocupado por reunir una buena cantidad de buscadores y automatizar su registro.

En DesarrolloWeb tenemos una herramienta multiregistro en buscadores gratuita y con interfaz web. El objetivo de esta herramienta es el de ofreceros la posibilidad, de dar de alta gratis vuestras páginas en una treintena de buscadores. Un servicio como éste cuesta mucho de mantener ya que los buscadores actualizan sus formularios de registro, desaparecen o ponen mecanismos para evitar que estas herramientas funcionen correctamente. Pese a ello creemos que se trata de una opción interesante para aquellos que deseen dar a conocer su página personal o no comercial sin tener que pagar por ello. Por supuesto, este tipo de servicio no ofrece ninguna garantía en lo que respecta su eficacia y, para proyectos comerciales, aconsejamos el uso de herramientas profesionales de pago.

Programas windows
También existen varios programas que se pueden comprar para realizar un registro múltiple en buscadores. La ventaja de estos programas con respecto a las versiones online con interfaz web, consiste en que detrás de los programas comerciales se encuentra una empresa que se lucra de vender su producto. Por esta razón, la empresa se preocupa mucho de mantener actualizada la base de datos de buscadores, de modo que las herramientas tengan la mayor efectividad posible.

Existen varias opciones de programas de este estilo, como Active WebTraffic, EspAñadir o SubmitWolf. Si bien nosotros os podemos recomendar Active WebTraffic, ya que es el que utilizamos para realizar nuestros registros masivos.

Respecto al envío de altas a directorios donde el árbol de categorías es muy amplio, donde un envio automático resulta ineficaz, como es el caso de Yahoo, Open Directory o Terra directorio, es recomendable darse de alta manualmente para elegir la categoría más adecuada al contenido de nuestra web. Por eso en la última versión del programa Active WebTraffic se ha incluído la opción de

"Buscadores manuales", que facilita la inclusión manual es estos directorios donde el registro manual es importante.

> **Nota:** para que el registro de la página sea correcto no sólo es necesario elegir la categoría adecuada, también habrá que seguir las normas del directorio para asegurarnos que el alta será aceptada por el editor que revise nuestra solicitud de alta. En el caso de Yahoo estas normas son muy extrictas y el alta manual tiene que ser perfecta para que la página sea incluída.
>
> Al utilizar un software de multiregistro también es necesario rellenar todos los campos correctamente siguiendo los consejos que, en el caso de Active WebTraffic, se envían al propietario del software para que el alta sea eficaz. Nunca se debe utilizar el software indiscriminadamente ya que lo único que se consigue es que nuestra web sea penalizada y no aceptada por el buscador. Como regla general una página al día por dominio es suficiente, hay que ser conservador.
>
> Si se ha introducido toda la información correctamente y se utiliza el software con prudencia, la página web será aceptada, exactamente igual a si se hubiera hecho un alta manual.

El posicionamiento en los buscadores de Internet

Todas las empresas con cierta entidad, hoy en día disponen de un sitio web, pero muchas de ellas se plantean si esta presencia en la Red ha sido una inversión o un mero gasto de representación.

Para que nuestra presencia en Internet sea rentable es necesaria la promoción de nuestro sitio, y una de las herramientas más rentables para dar a conocer nuestro site y generar visitas son los buscadores.

Aparecer en los principales buscadores internacionales, nacionales y en los específicos de nuestro sector es principal, pero no lo es menos aparecer en un lugar destacado según ciertas palabras clave que definan nuestro negocio en la mente de nuestras audiencias. Puesto la mayoría de los internautas se conforman con los primeros resultados proporcionados por su buscador favorito.

Y para aparecer en un lugar destacado en estas útiles herramientas de búsqueda debemos conocer su funcionamiento interno. Los buscadores se dividen en dos grandes grupos: los índices y los motores de búsqueda.

> **Referencia:** en nuestro manual de promoción de páginas web podemos encotrar gran parte de la información de este artículo, relatada con mayor detalle y más calmadamente.

Los índices dividen la información en un árbol temático de categorías y subcategorías. Aquí el ejemplo paradigmático sería Yahoo!, que nos presenta una serie de grandes categorías temáticas entre las que encontramos la subcategoría "Economía y Negocios", dentro de ella "Empresas", y esta a su vez contiene entre otras "productos y servicios para empresas" y así sucesivamente hasta ir acotando la amplitud de la categoría de sitios web, ya que no se nos permitirá proponer la inclusión de nuestro sitio web en una categoría demasiado amplia. En los índices lo esencial es encontrar la rama ideal de este árbol temático en la cual ubicar nuestro sitio web, y digo nuestro sitio, puesto que en los índices sólo es posible incluir una página (normalmente la principal) a su directorio, aunque a menudo es posible incluirlo en dos o tres categorías. Para encontrar esta categoría ideal en la cual debería estar nuestro web, la estrategia a seguir es puramente marketiniana; ponerse en la piel de nuestro público y pensar en qué categoría nos buscará. Para ello, podemos ayudarnos de un estudio de mercado, y como no, del sentido común y de la observación de en qué categoría se encuentran ubicados nuestros principales competidores. Pero cuidado, quizá ellos no lo hayan echo tan bien y no se encuentren en la categoría ideal. Lógicamente dependiendo de la amplitud de nuestros productos o servicios, será más obvio o más difícil hallar esta categoría ideal. No obstante tras nuestra petición de alta existe un proceso de revisión humano e incluso podemos proponer una nueva categoría si no nos encontramos debidamente definidos por ninguna de las existentes.

Y sobre este proceso de revisión humana es sobre el que quiero hablar a continuación, puesto que es este el segundo factor que más diferencia a los índices de Internet de los motores de búsqueda. Cuando proponemos el alta de nuestro sitio en el índice de turno, se nos pide toda una serie de datos, como: Título de la página, URL, Definición, Ubicación geográfica, persona de contacto y correo electrónico... Y finalmente nuestro site es revisado por un surfer (un especialista en catalogar recursos) del índice que considera si nuestro site cumple con los estándares de calidad requeridos y si está bien clasificado en la categoría elegida por nosotros.

Vemos que esto es lo único que conoce el índice de nuestro sitio web; los datos suministrados en el formulario de petición de alta en el buscador. Por lo que debemos ser extremadamente cuidadosos en la definición que enviamos de nuestro site.

El caso de los motores de búsqueda es bien distinto. Podemos tomar como ejemplo a Google, y veremos que la única información que proporcionamos a un motor es la dirección URL (por ejemplo: www.miempresa.com) y quizá una dirección de correo electrónico. El resto del proceso se realiza de forma automática, ya que nuestra petición de alta en el buscador entrará en la cola de trabajo de un programa de software llamado spider (araña) que visitará la página que hemos dado de alta y a partir de ella todas las que se encuentren enlazadas y así sucesivamente. Simultáneamente nuestras páginas serán indexadas utilizando complejos algoritmos, para ser devueltas como resultado cuando un internauta utilizando el buscador, introduzca un término que se encuentre en alguna de ellas y haga una petición de extracción de información de su ingente base de datos. Vemos de esta forma que nuestro web puede aparecer en algún motor de búsqueda por la simple razón de que otra página de un tercero que está incluida en el buscador enlaza a ella en Internet.

Así, en los motores de búsqueda, para obtener una notable posición, lo esencial es el código de nuestras páginas, algo que era verdaderamente indiferente en el caso de los índices.

Teóricamente con sólo dar de alta nuestra página principal el buscador indexará todas las páginas que cuelgan de ella, pero habitualmente nos encontraremos con problemas derivados de la ventaja que se concede a las altas de pago frente a las gratuitas; el primero es el tiempo a esperar para que nuestro sitio sea introducido en la base de datos del motor de búsqueda, que puede variar entre varias semanas a varios meses según el motor en cuestión. E incluso a menudo, tras este dilatado periodo de tiempo, no seremos indexados en sus bases de datos. Y esto, en los motores que aún admiten el alta gratuita.

Como recomendación, si nuestro tiempo y energías son limitadas deberemos optar por el alta de pago en algunos buscadores.

Si conocemos las interacciones entre los distintos buscadores de Internet, descubriremos que la inclusión en alguno de ellos puede suponer la sucesiva inclusión en otros que a menudo son más "duros" con las admisiones.

Una vez que conseguimos que nuestro web aparezca en los buscadores, nuestro trabajo no habrá hecho más que empezar, ya que lo realmente valioso es aparecer en los primeros lugares por aquellas palabras clave que nuestros públicos utilizan para buscar nuestra categoría de productos, y esta sí que es una verdadera guerra, puesto que en esa lucha estamos frente a nuestros principales competidores, que también batallarán por mejorar la posición de sus páginas frente a las nuestras y las de otros competidores. Desde luego, que la complejidad dependerá de la popularidad de las palabras clave por las que queramos aparecer de forma destacada en el buscador.

Realmente cada buscador valora de distinta forma el código de nuestras páginas para ubicarla en una u otra posición de su ranking, así por ejemplo Google valora especialmente cuantas y que tipo de páginas apuntan hacia las nuestras, aplicando una lógica bastante humana, según la cual si muchos y especialmente importantes hablan de uno, es que uno es importante. Otros motores como AltaVista valoran los Meta Tags (unas líneas de código que informan al motor acerca del contenido de nuestras páginas), etc.

En general, los buscadores se fijan en la frecuencia o densidad y ubicación con la que aparecen ciertos términos en nuestras páginas, para ubicarlas en un lugar superior de sus resultados frente a otras páginas, en las cuales la frecuencia y ubicación de este término que el navegante ha introducido en la caja de búsqueda del motor aparece.

Así, una palabra que está presente en nuestra misma dirección de Internet (www.palabra.com) indica un elevado nivel de coincidencia si es el término buscado por el internauta. Después es especialmente valorada esta palabra, en el título del documento, en el primer párrafo más que en el segundo... Si está en mayúscula es más valorado, al igual que si está en negrita... Si aparece dos veces en una frase más que si aparece una, etc.

A estas alturas, seguro que a más de un lector se le ha pasado por la cabeza, la idea de llenar de términos clave la página para que aparezca en las primeras posiciones del buscador, pero lamentablemente esto ya esta contemplado por estas herramientas que si encuentran demasiadas palabras repetidas o un texto de tamaño muy pequeño o con el mismo color que el fondo, etc, penalizarán nuestras páginas o incluso las eliminarán de la base de datos por tratarse de técnicas de spam (técnicas de promoción ilícitas)

El PageRank de Google

El motor de búsqueda de Google utiliza un concepto llamado PageRank como base de su tecnología de búsqueda. Vamos a contar algunas cosas con respecto a este concepto, que seguro que resultarán interesantes para los lectores centrados en temas de promoción de páginas web.

PageRank es el corazón del software de Google dedicado a posicionar las páginas web entre los resultados. Fue desarrollado por los creadores-fundadores del motor de búsqueda en la Universidad de

Stanford. Aunque actualmente hay decenas de desarrolladores que trabajan día a día en el desarrollo de la herramienta de búsqueda de Google, PageRank sigue siendo la base del software de búsquedas del afamado buscador.

PageRank se basa en la misma estructura de la World Wide Web para decidir la valoración de las páginas que la forman. Para ello utiliza los enlaces, que son también la parte más característica del sistema hipertexto, que es el sistema utilizado en los documentos de la WWW.

Los enlaces, para PageRank, significan votos: si una página enlaza con otra, considera que está dando un voto a esa página que vincula. Según el número de votos (o enlaces) recibidos por una página su posición variará. Lógicamente, a mayor número de votos, mejor posición entre los resultados.

Pero Google no mira simplemente en número de votos totales de cada página para decidir su posición, también analiza la página que otorga el voto. Si la página que realiza el enlace hacia otra página (o otorga el voto) es importante, el voto también tiene más importancia. Por tanto, es fundamental que nos enlacen páginas consideradas por Google como importantes. Aunque aun tendríamos que matizar una cuestión en este apartado: la importancia de la página no significa nada si la información que contiene no está relacionada con la información que se está buscando. Por ello, lo mejor es disponer de enlaces en páginas que traten sobre el mismo tema que la nuestra y que sean importantes para Google.

Además de PageRank, Google combina en sus búsquedas diversas técnicas que rastrean coincidencias de las palabras buscadas entre las páginas de su base de datos. Las búsquedas de coincidencias de textos, como ya se sabrá, abarcan gran cantidad de lugares, como el título, etiquetas META, cuerpo de la página y además valoran cada aparición según donde se produzca y en que condiciones. Las búsquedas de coincidencias también se extienden a las páginas que enlazan con la página que Google pretende posicionar, es decir, también busca coincidencias en las páginas que enlazan con otra para valorar si ese voto otorgado tiene más o menos importancia.

Caso "go to hell"

Aunque las palabras "go to hell" no tienen nada que ver con Microsoft, durante un tiempo Google estuvo ofreciendo la web de la multinacional -www.microsoft.com- como primer resultado de la búsqueda.

Este caso tiene mucho que ver con lo que estábamos comentando acera del PageRank y los enlaces como ponderadores de los resultados en Google. Microsoft por su parte no tiene mucha relación con infierno, pero parece que los enlaces con el texto "Go to hell" de webmasters particulares, descontentos con la compañía, eran muy populares y esto hacía que, a pesar de que esas palabras no figuraran en la web de Microsoft por ningún sitio, la búsqueda colocase su web por encima de otros resultados como hell.com.

Esto quiere decir dos cosas:

- El texto de los enlaces que nos ponen hacia nuestro web tiene mucha importancia.

- Google es un sistema vivo y en desarrollo, lo que hoy puede ser importante para ponderar los resultados mañana puede ser considerado de distinta forma. El caso es que Google ya no muestra la web de Microsoft al escribir esas palabras.

Conocer el PageRank de las páginas que visitamos

Si queremos conocer el valor de PageRank de las páginas que visitamos, aunque sólo sea por curiosidad, podemos descargar la barra de herramientas de Google que, aparte de permitir realizar búsquedas en dicho motor rápidamente, muestra en un campo de la barra el valor asignado al PageRank de la página que se está mostrando en la ventana de nuestro Internet Explorer.

Una imagen de la barra y el PageRank se pueden ver a continuación:

Conclusión

Es fácil extraer conclusiones por nosotros mismos sobre el funcionamiento de Google y la manera de ponderar sus resultados. Mejorar nuestro PageRank para disfrutar de una mejor posición en los buscadores puede ser complicado, ya que Google dispone de mecanismos para detectar cuando una página está utilizando técnicas ilegales para mejorar su posición. De todos modos, lo más útil es que muchas páginas nos enlacen, a ser posible, de webs bien valoradas y que traten nuestra misma temática.

La promoción de webs con frames

Conseguir que una web se encuentre entre los primeros puestos de los buscadores no es una tarea fácil y, si la web ha sido creada por medio de frames, puede complicarse todavía un poco más. En este pequeño artículo vamos a dar algunas razones para justificar la afirmación anterior, junto con consejos que pueden ayudarnos a conseguir mejores posiciones en las webs con frames.

> **Referencia:** Tratamos los temas de promoción de webs y de frames (también llamados marcos) en general en los manuales de promoción y HTML de DesarrolloWeb.com.
>
> Manual de promoción
> Capítulos de frames del manual de HTML

La dificultad de promocionar una web con frames se basa en dos puntos, aunque probablemente otras personas encuentren otras ventajas e inconvenientes que no señalemos aquí:

- La página principal del frame no tiene texto y tan solo etiquetas <frameset> y <frame>, así pues, los motores de búsqueda como Google, no pueden extraer palabras clave o una descripción de nuestra página.
- Las páginas hechas con frames están compuestas de varias páginas distintas y, según nuestra interpretación, si el texto estuviera en un mismo archivo HTML, contendría más palabras de donde el motor pueda extraer una buena información de la página. En este punto también cabe destacar que la navegabilidad del sitio decrece si no se accede a la web entrando por la declaración de frames y sólo se visualiza un archivo HTML que debería mostrarse dentro de un frame, ya que ese archivo seguramente contenga menos enlaces y referencias a otras partes del web. Esto motiva que la impresión de un visitante por la página pueda ser peor que si accede visualizando los marcos completos.

Consejos para mejorar la promoción con frames

1.- Titular correctamente la declaración de frames
Hay muchos desarrolladores que titulan el archivo HTML con la declaración de frames de forma poco descriptiva o incluso con palabras como "frames", "index.html" o "marcos". Esto es un error muy grave porque todas las páginas de nuestro sitio heredan el título de los frames y no deseamos que nuestra web o la del cliente se llame "marcos", sino algo descritivo como "Almacén de materiales de construcción Hermanos Gutierrez".

2.- Colocar las etiquetas META en la declaración de frames

Ya que no se pueden extraer palabras clave o descripciones de una declaración de frames porque no tiene texto, es aconsejable que las introduzcamos dentro de las etiquetas META en la cabecera de la página.

3.- Cuidar el texto de la etiqueta <NOFRAMES>
Algunos motores de búsqueda extraen el texto para palabras clave o descripción de la página de lo que el desarrollador haya colocado entre <noframes> y </noframes>, así que no coloques únicamente algo como "Esta página utiliza frames pero su navegador no los admite". Mejor coloca una página de inicio completa, con texto, etiquetas <b>, <h1> o imágenes, o lo que desees para que la portada quede bien para ese tipo de exploradores y se puedan extraer palabras clave y descripciones adecuadas.

4.- Las páginas interiores también importan
Los motores de búsqueda también pueden indexar las páginas interiores de los sitios con frames, tratándolas como páginas independientes a pesar que el desarrollador las hayas concebido con marcos. Muchas veces estas páginas son muy adecuadas para obtener una excelente posición en buscadores, ya que contienen texto sobre el tema de la página sin información superflua alrededor. Así que cuida mucho el modo de realización de estas páginas colocando títulos correctos, etiquetas META, un buen conjunto de palabras clave repetidas entre el texto de la página, etc.

5.- Añadir navegabilidad y datos de contacto en las páginas interiores
Dado que las páginas incluidas dentro de los frames también serán un punto de entrada habitual en la web, añade siempre enlaces para poder volver a la portada, otros contenidos del sitio y a la declaración de frames en si para que el visitante pueda ver la web con todos sus elementos. También es importante colocar información sobre el cliente o la nuestra propia para que se puedan poner en contacto con nosotros y el visitante pueda conocer el propietario de la información a la que está accediendo. Además, en la dirección o datos de la empresa o particular, en muchos de los casos, tenemos palabras clave que pueden ser muy interesantes, como nuestra localidad o actividades.

8.- Registra también las páginas interiores que te interese promocionar
Algunos buscadores recorren la estructura de tu web entera indexando las distintas páginas. Otros buscadores, generalmente los directorios, ni siquiera tienen esas funcionalidades, pero para los dos casos es interesante registrar las páginas interiores para asegurarnos de que figuren también. Este consejo puede servir para todo tipo de webs, pero en el caso de los frames puedes construirte diversas entradas repitiendo la declaración de frames en diversos archivos que muestren en el cuerpo de la página una información distinta.

7.- Crea un "tunel de entrada"
Para que tu página principal no sea una declaración de frames, poco promocionable, puedes crear una portada para la página en un archivo HTML normal, con texto, imágenes y demás elementos donde introduzcas el contenido de la web, cuidando especialmente en colocar las palabras clave y descripciones que desees conseguir. Esta página podría contener el enlace a la declaración de frames, o enlaces en el caso de que tengamos varias declaraciones.

Puede haber otros consejos para promocionar páginas con frames, de hecho, en general todos los consejos para promocionar webs en páginas sin marcos pueden ayudar a una página que sí los tiene. Esperamos que estas notas sirvan a la correcta realización de las páginas con frames desde el punto de vista de su promoción en buscadores. Si alguien le interesa añadir algo puede hacerlo comentando el artículo con un enlace de en la parte de abajo.

Capitulo 6
La imagen en Internet

Explica cómo debemos crear la imagen en Internet de las empresas, productos o servicios que deseamos vender. No vale clonar lo anterior; hace falta conjugar estrategia, contenido, diseño y tecnología.

La imagen en Internet

El desarrollo imparable de internet en los inicios del siglo XXI obliga a introducir un capítulo de este ámbito, porque ninguna dirección de comunicación puede ni debe ignorar su importancia

Lo más peligroso ante la eclosión de Internet puede ser que la empresa se duerma y quiera ignorar su existencia. Será un buen paso para perder oportunidades y liderazgos. Aunque también puede errar quien se equivoque de estrategia.

La primera duda que puede plantearse al comunicador o al empresario es si debe construir o no una página web. La tendencia general es apostar por ella, aunque es cierto que si lo único que se busca es notoriedad tal vez sea efectivo y barato acudir a lugares de gran tráfico, fundamentalmente portales, y situar allí el mensaje de forma atractiva.

No obstante, es básico mantener un lugar de referencia, eso sí, suficientemente atendido, que sirva adecuadamente a nuestras estrategias de comunicación y mercado.

En los medios informativos se había especulado en los años pasados sobre el crecimiento de la publicidad en Internet. La realidad es que ésta crece pero a un ritmo menor de lo previsto. No es por esta línea por donde se presenta la real subversión de las nuevas tecnologías.

Es por los aspectos logísticos y de venta por donde asoma el cambio. Libros, Música, Viajes, Juguetes, etc., se comercializan cada día más a través de la Red, con lo que proveedores clásicos padecen un deterioro de su línea de negocio, ante jóvenes y ágiles competidores que –sin necesidad de tiendas ni casi de empleados- acaparan o van a acaparar crecientes volúmenes de negocio.

Entre los casos llamativos figura el conocido en octubre de 1999, cuando la empresa informática Dell Computer se situó por primera vez como la compañía que más ordenadores personales (PC) vendió en Estados Unidos al superar a su rival Compaq en el tercer trimestre de este año. En el fondo estaba un hecho que los expertos habían detectado desde tiempo atrás, como lo atestiguó la propia sustitución del presidente de Compaq, E. Pfeiffer, ante la mala marcha del negocio electrónico.

Los nuevos canales electrónicos están reordenando las cuotas de mercado. Y es algo imparable. Algunos jugueteros, en las Navidades de 1999 comentaban que los niños consultaban Internet antes de acudir a la juguetería, pero luego querían tocar físicamente el objeto de sus sueños en el establecimiento. Es media verdad. Niños y padres contactan cada vez más con Internet para ver juguetes… Y compran. Y los directivos de Toys R Us veían como eToys.com acaparaba una creciente tarta del mercado desde su pantalla electrónica. Luego vinieron problemas para la firma de comercio electrónico, pero la realidad está marcada.

Y esa evolución será creciente. Aunque sólo un dos por ciento de los ciudadanos de América Latina utilizan habitualmente Internet en el inicio del 2000, la cifra crece en torno a un 30 por ciento anual. La mejora de estructuras telefónicas, la adecuación de los precios y el arrastre de las nuevas tecnologías asegurará que lo que hoy es una minoría –eso sí, de gran poder económico y prescriptora de opinión- sea mañana la generalidad.

Todos somos medios

En el libro de Vendedores de Imagen sostuve que todo comunica, desde la sonrisa del dependiente al logo de la empresa.

Además, con Internet hay que añadir algo importante: todas las webs son medios. Definitivamente se ha roto la idea de los "medios de comunicación" en el sentido clásico que hemos conocido durante todo el siglo XX.

Hasta ahora entendíamos como medios a la prensa, la radio, la televisión y las revistas. Todos eran productos emanados desde las editoriales. Ahora se ha roto el rol de la editorial.

Tomemos un ejemplo. La cerveza XXX decide abrir una página en Internet. Es algo que han hecho muchas empresas en los últimos años, porque cuando el uso de la red empezó a popularizarse, entidades y empresas empezaron pronto a ocupar un espacio, situando allí un mensaje institucional o comercial.

El primer paso fue colocar una web institucional. La cerveza XXX colocó una página oscura donde mostraba el grupo empresarial que la fabricaba, un poco de su historia y un par de fotografías de la factoría principal.

Como segundo paso, y para modernizar su imagen, en la parte inferior de las páginas publicitarias que situaba en los medios e incluso en sus anuncios de vallas y televisión, comenzó a poner un WWW.cervezaXXX.com. En realidad no sabía exactamente que significaba aquello, únicamente intuía que daba un tono moderno al anuncio.

Al año siguiente, la web de XXX avanzó un poco más. Aparte del contenido institucional del grupo cervecero alegró su imagen con unas pinceladas de juventud y humor, dejó que unas burbujas correteasen sobre una jarra rebosante de apetitosa bebida. Dio un contenido similar al de la imagen publicitaria. En realidad, transformó su imagen de Internet en mensaje publicitario clásico.

En el 2.000, la empresa decidió avanzar un nuevo paso. Descubrió que mucha gente que entraba una vez en su web no volvía. Una vez vista la misma, el navegante de Internet no experimentaba la necesidad de regresar. Es lógico. Nadie sueña con ponerse ante el televisor y contemplar varias veces el mismo noticiario.

A partir de este descubrimiento, los hombres de XXX dedujeron que había que darle contenidos cambiantes a la cerveza. Identificaron que el navegante era joven y quisieron llegar a él. Por ello contactaron con EFE, primer proveedor mundial de noticias en castellano, y le dijeron lisa y llanamente: queremos que nos prepares un periódico con noticias y fotografías del mundo del ocio: fundamentalmente música y películas.

Y a partir de ese día, XXX generó –sin necesidad de redacción- una revista sobre ocio en permanente renovación de contenidos; algo que indujo a los jóvenes navegadores a situar entre sus "favoritos" a WWW.cervezaXXX.com. Simplemente, con una revista de ocio habían logrado darle a la web una nueva salida, superando la presencia estática anterior.

Pero ahí no acabó la evolución, porque con un par de toques adicionales introdujeron unos juegos, concursos e incluso permitieron que el navegante se hiciera una página adecuada a sus deseos. Pulsó un botón que indicaba "Personalizar" y a través de las sencillas opciones que se le presentaron cambió el color del fondo de la pantalla y seleccionó las noticias de su grupo musical favorito. La propia web de XXX empezó a ser su suministradora de información sobre discos y música preferida…un servicio más de una cerveza que tal vez de esta forma entró en su hábito de consumo.

XXX dejó de ser simplemente una cerveza para transformarse en un medio informativo en materia de ocio, una cerveza amiga y confidente, que conocía los gustos del navegante.

Medios en Internet

En Internet están apareciendo portales de música, jurídicos, financieros, culturales…. Cualquiera que pretende entrar con fuerza en la red acaba necesitando un contenido informativo o de diversión. Y la Red está trocando las costumbres de los ciudadanos. En los albores del 2001, multitud de grandes bancos están penetrando al unísono en el ámbito de los negocios financieros por la red. Pero cada banco que intenta avanzar por esta senda –y es una senda cada vez más usada- sitúa al servicio de su cliente la información. Información en tiempo casi instantáneo de evolución de cotizaciones; datos de magnitudes monetarias, cambios, informes económicos, movimientos empresariales, etc.

En estas páginas se está encontrando una información económica y financiera básica, los consejos de las sociedades de bolsa, las calificaciones de solvencia de los agentes económicos, etc. Las páginas se han trocado en auténticos diarios de información económica.

No sólo los medios informativos clásicos salen de su sus soportes de papel o sus ondas para abrir webs en Internet, surgen nuevos medios específicos para la red. Pero además, las empresas e instituciones de todo tipo avanzan en su papel tradicional y se convierten en poseedores y difusores de información.

Hasta la cerveza XXX ha empezado a ser una revista de ocio…

La publicidad

Al finalizar el siglo XX, los editores de los países desarrollados estaban pasando un calvario por temor al impacto de Internet en las cuentas de resultados de sus compañías. Todos temían que el crecimiento vertiginoso del uso de la Red posibilitara un súbito desajuste, cuando no un derrumbamiento de los ingresos por publicidad.

El temor se ha ido aplacando progresivamente, porque no ha habido un vuelco de las cifras tan vertiginoso como se esperaba.

El informe anual de inversiones publicitarias que en España realiza Infoadex dice que en el ejercicio de 1999 apenas el 1,5 por ciento de la facturación en medios convencionales y no convencionales correspondió a Internet, aunque con un crecimiento que evalúa en el 600 por ciento respecto al ejercicio precedente.

La cifra es baja, si bien hay que tener en cuenta que parte de la inversión en medios está vinculada a Internet, tanto en su oferta de productos y servicios como en la promoción de páginas web. De todas formas, la publicidad de la Red aún se mueve en unos niveles relativamente pequeños.

Los problemas para el crecimiento de la publicidad en Internet son varios, entre ellos está la direccionalidad con la que opera quien entra en la red y el sentido del tiempo del internauta.

Cuando lector clásico de periódico abre el ejemplar, sentado en la terraza de su casa, en el jardín o en el mismo tren, no sólo mira el ejemplar sin prisa –siempre se lo puede llevar a otro lado para continuar la lectura- sino que a veces se recrea en ver los anuncios de ropa, de automóviles o de pisos. Frente a la ordinariez de tanta letra, un anuncio atractivo provoca también placer estético.

Cuando el internauta se sienta ante su ordenador, busca algo; pero generalmente no busca anuncios. Es más, al llegar a una página se exaspera ante los segundos que tarda en abrirse y la mayor parte de las veces ni se le ocurre pinchar en el baner que le ofrece un producto financiero o de otro tipo. El internauta busca información concreta, busca contactos con gente, pero no presta gran atractivo a la publicidad. Es más, se están diseñando navegadores de páginas HTML que evitan mostrar los anuncios en forma de baners.

De momento, los expertos piensan que Internet está rentabilizándose más por el comercio y negocio financiero que genera y facilita que por los ingresos publicitarios.

Pero el interés publicitario de Internet es real. El secreto para sacarle partido es presentar la publicidad en el lugar en el que se halla el potencial cliente.

Miguel Angel Alvarez Sánchez es uno de los locos que apostó un día por crear una web temática. Por ella, desarrolloweb.com, pasan los muchachos o mayores que andan por el mundo tratando de encontrar información sobre construcción de webs, programas, consejos, recursos gratuitos, etc. Muchos cientos de miles de páginas de su web se consultan mensualmente. Quien quiera encontrar a ese público - programadores, diseñadores, webmasters, etc.- saben que está allí, en desarrolloweb, y que no va a estar en otros portales generalistas…. Es un ejemplo.

La publicidad en Internet es rentable, siempre que el buen publicista sepa localizar donde andan sus potenciales clientes, para colocarles allí su mensaje.

Consejos para actuar en la Red

1º Dar lo que busca el usuario

El usuario que llega a una página lo puede hacer por muchas razones, desde la simple consulta del baner hasta su curiosidad ante algo que ha leído o escuchado en los medios clásicos. En ese momento hay que lograr que el impacto sea positivo.

Los vendedores saben que es muy importante la primera impresión en cualquier contacto. Esa primera impresión posibilita que "el otro" se sitúe frente a nosotros con ánimo de colaboración u oposición. De esa primera impresión depende en gran parte la marcha del resto del encuentro o de las negociaciones.

Los vendedores dicen que "solo una vez se genera la primera impresión", por ello es absolutamente necesario aprovechar el primer encuentro, un encuentro en el que no cabe defraudar al internauta.

2º Dar alegría y dinamismo

Una página triste, estática o confusa es la primera condición para no seguir. También lo es una página que presenta complicaciones tecnológicas. Si cuando accedes a ella te encuentras con un "descargue el programa…. Para ver en su PC la página de tal producto…." El internauta se desanima. Incluso puede temer que con la descarga entre algún virus en su ordenador, también puede temer enfrentarse a un proceso técnico que le robe bastantes minutos… Y frecuentemente vuelve atrás y regresa hacia algo conocido y familiar.

3º Fidelizar

Aquí entra en juego el contenido. Sólo conseguiremos que el internauta vuelva si ofrecemos un contenido apetente. Ocurre igual que quien pone un restaurante: si el restaurador ofrece una comida oriental extraña a los gustos de la clientela del lugar lo tiene difícil a la hora de rentabilizar la inversión.

Para fidelizar hay que dar contenidos de calidad y novedades. Lo mismo que no leemos un gran libro todas las semanas, no entraremos en una excelente página de Internet todas las semanas si no esperamos que en la misma haya algo novedoso o sumamente entretenido. La página Web puede tener noticias, juegos, e incluso sencillos sistemas de participación en sorteos que permiten una fidelización.

Como ejemplo se puede poner la web presentada por Coca Cola en el inicio de 2000, www.cocacola.com en la que el usuario personaliza la información que quiere recibir sobre cine, música, deporte y ocio, contenido que se le selecciona y presenta de forma divertida. El internauta accede a su propia habitación, que puede decorar y cambiar de color, en un edificio donde convive con otros personajes.

En esas fechas, Coca Cola anunció también la celebración de sorteos y concursos a través de la red, sin descartar la entrada en el comercio electrónico. Y todo sin perder el humor. Se trata de un intento de acceder de forma eficaz hacia la juventud, empleando sus mismos códigos que el cliente al que Coca Cola quiere seducir. Como pero a este producto cabe señalar la complejidad técnica: la necesidad de descargar un programa adicional y recordar una complicada clave. Una clave suele ser siempre un estorbo.

Consejos para actuar en la Red II

4º Innovar

En este sector en el que todos convergen es preciso acudir con calidad e imaginación. Se trata de aportar ideas originales que cautiven al personal, o estrategias nuevas. Una de estas, por ejemplo, es en el momento en que se escriben estas líneas la colaboración entre empresas distintas para crear fuertes promociones conjuntas, galerías de compra o parques de ocio virtual. No vale sólo copiar lo conocido, hay que avanzar.

5º Cuidar el factor humano. Respuesta e interactividad

Es básico dar una personalidad a la nuestra página web. En la página hay que transmitir un mensaje hacia el internauta, un mensaje con emoción. Esa emoción tan conseguida en numerosos anuncios vistos en otros formatos, especialmente la televisión, se puede transmitir y compartir.

El compromiso conjunto se asienta sobre varios elementos, uno de ellos es la respuesta eficaz a los correos, a las demandas o a llamadas telefónicas. El efecto de la acción conjunta página-correo electrónico-teléfono es excelente.

6º Establecer una comunidad

La efectividad de la comunicación a través de Internet puede ser alta, si se consigue el efecto comunidad. Cuando el navegante accede a la página diseñada y conecta con los temas de la misma se produce un efecto identificación que posibilita compartir futuro. Eso es particularmente visible y fácil en el caso de las webs que vinculan grupos sociales de cualquier índole: los aficionados de un club de fútbol, los nacidos en La Cepeda o Montevideo, los admiradores del Museo del Prado o los combatientes en las Malvinas. Cierto que a medida que la colectividad es mayor se hace más difícil el mantenimiento de vínculos. Pero la creación de colectividades permite una gran eficacia y rentabilidad comunicativa.

Capitulo 7
Páginas dinámicas

Introducción al concepto de páginas dinámicas, lenguajes de lado cliente y servidor y otra serie de nociones básicas para lanzarse a la aventura de la programación en ASP o PHP. Este manual sirve de introducción a otros más avanzados.

Concepto de páginas dinámicas

Muy probablemente, estimado lector, seas una persona más o menos familiarizada con el lenguaje HTML y con toda seguridad hayas desarrollado algún sitio basado en esta herramienta. En realidad el HTML no es lenguaje de programación sino, más bien, se trata de un lenguaje descriptivo que tiene como objeto dar formato al texto y las imágenes que pretendemos visualizar en el navegador.

A partir de este lenguaje somos capaces de introducir enlaces, seleccionar el tamaño de las fonts o intercalar imágenes, todo esto de una manera prefijada y en ningún caso inteligente. En efecto, el HTML no permite el realizar un simple cálculo matemático o crear una página de la nada a partir de una base de datos. A decir verdad, el HTML, aunque muy útil a pequeña escala, resulta bastante limitado a la hora de concebir grandes sitios o portales.

Es esta deficiencia del HTML la que ha hecho necesario el empleo de otros lenguajes accesorios mucho más versátiles y de un aprendizaje relativamente más complicado, capaces de responder de manera inteligente a las demandas del navegador y que permiten la automatización de determinadas tareas tediosas e irremediables como pueden ser las actualizaciones, el tratamiento de pedidos de una tienda virtual...

Estos lenguajes capaces de recrear a partir de ciertos "scripts" un sinfín de páginas automatizadas son los protagonistas de este concepto de páginas dinámicas.

Este manual, que no es mas que una introducción a otros manuales en curso de redacción, esta destinado a aquellos que sienten que el HTML se les queda corto para realizar sus proyectos y que, sin tener ni idea de programar, desean dar el paso y darle un nuevo aire a sus páginas sin por ello pasar por experiencias traumáticas debidas a su poca afinidad con los lenguajes informáticos.

El contenido ha sido, por lo tanto, deliberadamente simplificado y será ampliado posteriormente a partir de anexos y artículos de mayor nivel para que pueda ser también utilizado por aquellos que ya están familiarizados con este tipo de lenguajes.

Páginas dinámicas vs HTML

A pesar de que las páginas dinámicas nos puedan en un principio limitar a causa de su mayor complejidad con respecto al HTML, todas las ventajas que nos ofrecen compensan con creces este esfuerzo inicial.

No obstante, hay que ser consciente del posible interés que pueda tener para uno el lanzarse en esta aventura de aprender un nuevo lenguaje y volver a rediseñar su propio sitio.

Si la página en la que estamos pensando o que queremos rediseñar es relativamente pequeña, no necesita estar al día continuamente sino que sus contenidos son perennes y no hemos previsto el pagar por mantenerla, el empleo de páginas dinámicas puede quedarse grande y resultar a todas luces improductivo.

Por el contrario, si el sitio es extenso y sus contenidos cambian rápidamente, nos interesa el automatizar en la medida de lo posible todas las tareas de tal forma que podamos gestionar su explotación de la manera más óptima.

Para dejar más claro hasta que punto resulta útil utilizar páginas dinámicas lo mejor será ejemplificarlo a partir de un sitio web modelo.

Supongamos que hemos decidido realizar un portal de televisión donde una de las informaciones principales a proveer podría ser la programación semanal. Efectivamente, esta información suele ser dada por las televisiones con meses de antelación y podría ser muy fácilmente almacenada en una base de datos. Si trabajásemos con páginas HTML, tendríamos que construir una página independiente para cada semana en la cual introduciríamos "a mano" cada uno de los programas de cada una de las cadenas. Asimismo, cada semana nos tendríamos que acordar de descolgar la página de la semana pasada y colgar la de la anterior. Todo esto podría ser fácilmente resuelto mediante páginas dinámicas.

En este caso, lo que haríamos sería crear un programa (solo uno) que se encargaría de recoger de la base de datos de la programación aquellos programas que son retransmitidos en las fechas que nos interesan y de confeccionar una página donde aparecerían ordenados por cadena y por hora de retransmisión. De este modo, podemos automatizar un proceso y desentendernos de un aspecto de la página por unos meses.

Este hecho lo podríamos aplicar a otras situaciones: podemos preparar el horóscopo de todos los días, las promociones de un sitio de e-comercio...

Además, tampoco resultaría complicado el introducir una pequeña caja de búsqueda que nos permitiera dar rápidamente con el programa que queremos ver, saber a qué hora y en qué cadena se emite.

Volviendo a nuestro portal de televisión, en él hay una sección en la cual presentamos todas las series actualmente emitidas con comentarios sobre ella, fotos, etc. Podríamos, en lugar de hacer una página HTML por serie, hacer una única página dinámica en contacto con una base de datos en la cual visualizamos las fotos y comentarios relativos a la serie que nos interesa. Asimismo, si lo que buscamos es modificar el formato del texto de dicha sección, podemos automatizar este proceso sin necesidad de cambiar a mano cada una de las etiquetas font y sin hacer uso de la hojas de estilo las cuales no son reconocidas por la totalidad de los navegadores.

Otra serie de aspectos tales como la gestión de las lenguas, podrían ser fácilmente resueltos sin para ello duplicar el número de páginas y buscar los textos a traducir penosamente entre el código HTML.

En realidad, a partir de estas herramientas, podemos plantearnos cuantas cosas queramos. El único límite... nuestra imaginación

Lenguajes de lado servidor o cliente

El navegador es una especie de aplicación capaz de interpretar las órdenes recibidas en forma de código HTML fundamentalmente y convertirlas en las páginas que son el resultado de dicha orden.

Cuando nosotros pinchamos sobre un enlace hipertexto, en realidad lo que pasa es que establecemos una petición de un archivo HTML residente en el servidor (un ordenador que se encuentra continuamente conectado a la red) el cual es enviado e interpretado por nuestro navegador (el cliente).

Sin embargo, si la página que pedimos no es un archivo HTML, el navegador es incapaz de interpretarla y lo único que es capaz de hacer es salvarla en forma de archivo. Es por ello que, si queremos emplear lenguajes accesorios para realizar un sitio web, es absolutamente necesario que sea el propio servidor quien los ejecute e interprete para luego enviarlos al cliente (navegador) en forma de archivo HTML totalmente legible por él.

De modo que, cuando pinchamos sobre un enlace a una pagina que contiene un script en un lenguaje comprensible unicamente por el servidor, lo que ocurre en realidad es que dicho script es ejecutado por el servidor y el resultado de esa ejecucion da lugar a la generacion de un archivo HTML que es enviado al cliente.

Así pues, podemos hablar de lenguajes de lado servidor que son aquellos lenguajes que son reconocidos, ejecutados e interpretados por el propio servidor y que se envían al cliente en un formato comprensible para él. Por otro lado, los lenguajes de lado cliente (entre los cuales no sólo se encuentra el HTML sino también el Java y el JavaScript los cuales son simplemente incluidos en el código HTML) son aquellos que pueden ser directamente "digeridos" por el navegador y no necesitan un pretratamiento.

Cada uno de estos tipos tiene por supuesto sus ventajas y sus inconvenientes. Así, por ejemplo, un lenguaje de lado cliente es totalmente independiente del servidor, lo cual permite que la página pueda ser albergada en cualquier sitio sin necesidad de pagar más ya que, por regla general, los servidores que aceptan páginas con scripts de lado servidor son en su mayoría de pago o sus prestaciones son muy limitadas. Inversamente, un lenguaje de lado servidor es independiente del cliente por lo que es mucho menos rígido respecto al cambio de un navegador a otro o respecto a las versiones del mismo. Por otra parte, los scripts son almacenados en el servidor quien los ejecuta y traduce a HTML por lo que permanecen ocultos para el cliente. Este hecho puede resultar a todas luces una forma legítima de proteger el trabajo intelectual realizado.

Lenguajes de lado servidor

Existe una multitud de lenguajes concebidos o no para Internet. Cada uno de ellos explota más a fondo ciertas características que lo hacen más o menos útiles para desarrollar distintas aplicaciones.

La versatilidad de un lenguaje está íntimamente relacionada con su complejidad. Un lenguaje complicado en su aprendizaje permite en general el realizar un espectro de tareas más amplio y más profundamente. Es por ello que a la hora de elegir el lenguaje que queremos utilizar tenemos que saber

claramente qué es lo que queremos hacer y si el lenguaje en cuestión nos lo permite o no.

En el dominio de la red, los lenguajes de lado servidor más ampliamente utilizados para el desarrollo de páginas dinámicas son el ASP, PHP y PERL.

El ASP (Active Server Pages) es un lenguaje derivado del Visual Basic desarrollado por Microsoft. Evidentemente su empleo se realiza sobre plataformas funcionando bajo sistema Windows NT.

El PHP podría ser considerado como el lenguaje análogo al ASP utilizado en plataformas Unix y Linux.

Estos dos lenguajes resultan bastante útiles para la explotación de bases de datos y su aprendizaje resulta accesible para una persona profana de la programación. Cualquiera de ellos resultaría la opción ideal a la hora de hacer evolucionar un sitio web realizado en HTML.

Por otra parte, el PERL es un lenguaje más rápido y potente que requiere obviamente un aprendizaje más largo y resulta más reservado para personas ya familiarizadas con la verdadera programación.

Algunos aspectos prácticos previos

Antes de lanzarnos en las consideraciones teóricas relativas a la programación, resultaría interesante aclarar algunas dudas que puede presentarse referentes a cómo escribir y publicar páginas dinámicas.

Para escribir una página dinámica podemos hacerlo del mismo modo que si lo hiciésemos en HTML. En realidad, el código está constituido exclusivamente de texto y lo único que tenemos que hacer por lo tanto es guardar el archivo texto con una extensión que pueda ser reconocida posteriormente por el servidor. Así, por ejemplo, las páginas de ASP son reconocidas por su extensión "asp" del mismo modo que las de PHP lo son a partir de extensiones "php" u otras en las que se especifica la versión utilizada ("php3" o "php4"). En muchos casos el servidor nos permite seleccionar qué tipo de extensión debe ser reconocida para un determinado lenguaje por lo que estas extensiones no están totalmente generalizadas aunque son sin duda las más utilizadas.

Dado que se trata únicamente de archivos texto, es posible crear páginas dinámicas a partir del Bloc de Notas o cualquier otro procesador de texto plano (Texto ASCII, sin códigos raros como los que pone MS Word). También podemos utilizar los editores clásicos empleados para el HTML aunque en este caso, estamos obligados a trabajar en modo editar y no en modo gráfico. Esta última posibilidad resulta tanto menos aconsejable cuanto que la mayoría de estos editores no están preparados para la programación en estos lenguajes y algunos de ellos (Frontpage en sus versiones anteriores a la 2000, sin ir más lejos) están dispuestos a borrar aquellos textos que no es capaz de interpretar.

> Existe un artículo en desarrolloweb que explica de manera más detallada el mundo de los editores de páginas

Existen sin embargo algunos editores de HTML que si ofrecen ventajas al editar scripts. Tal es el caso del Homesite que muestra coloraciones diferentes en función de la sintaxis del programa lo cual permite una lectura más fácil. Además, hay otra serie de editores más pensados para páginas dinámicas en general o para algún lenguaje en particular.

Una vez el programa realizado, el paso inmediato es el de ejecutarlo. Como ya ha sido explicado, los lenguajes de lado servidor ejecutan los scripts en el propio servidor y envían el resultado en forma de código HTML al cliente (navegador). Resulta obvio que para probar entonces el programa es necesario colgar por FTP los archivos que lo componen en el servidor y hacer la petición desde el navegador. En principio, no es por tanto posible el trabajar offline a partir de archivos alojados en el disco duro tal y como hacíamos con el HTML. Esto en realidad no es completamente cierto ya que existe la posibilidad de convertir nuestro propio ordenador en servidor web personal de manera que podemos trabajar en local sin necesidad de estar conectados continuamente lo cual podría representar un problema para aquellos que tengan que pagar una factura telefónica al estar conectados por modem además de resultar más juicioso puesto que un servidor no es el sitio ideal para hacer nuestros pinitos en un lenguaje que no controlamos suficientemente. Bucles infinitos, variables no cerradas y otra serie de irregularidades pueden estar consumiendo recursos importantes en perjuicio de los usuarios que estén accediendo a otras paginas albergadas por este servidor.

Como puede verse, la forma de operar resulta casi análoga a lo que hacíamos para nuestro sitio estático y no presenta ninguna complicación aparente. Cabe destacar que, como ya se ha dicho anteriormente, para poder servirse de estos lenguajes de lado servidor, es imprescindible que el servidor esté preparado para leer las páginas programadas en un lenguaje no comprendido por el navegador. Dichos servidores son en su gran mayoría de pago lo cual añade ciertas limitaciones económicas al proyecto.

Conceptos básicos de programación I

Antes de abordar en detalle las particularidades de estos lenguajes, es importante guardar en espíritu toda una serie de nociones básicas comunes. Estos aspectos son sin duda conocidos por aquellos que hayan programado alguna vez y pueden ser muy rápidamente asimilados por todos los que estén familiarizados con las matemáticas. Teniendo en cuenta esto, hemos querido acercar estos conceptos a cualquier persona proponiendo definiciones poco rigurosas y carentes de detalles pero que en contrapartida permiten ser digeridas con más facilidad.

Así pues, aquellos sugerimos el pasar directamente al siguiente capítulo a todos aquellos que consideren conocer perfectamente los conceptos de variable y función aunque siempre puede resultar interesante volver a recordarlo visto desde el prisma de otra definición.

Variable

Una variable consiste en un elemento al cual le damos un nombre y le atribuimos un determinado tipo de información. Las variables pueden ser consideradas como la base de la programación.

De este modo podríamos escribir en un lenguaje ficticio:

```
a="perro"
b="muerde"
```

La variable que nosotros llamamos "a" posee un elemento de información de tipo texto que es "perro". Asimismo, la variable "b" contiene el valor "muerde".

Podríamos definir una tercera variable que fuese la suma de estas dos:

```
c=a+b
```

Si introdujésemos una petición de impresión de esta variable en nuestro lenguaje ficticio:

```
imprimir(c)
```

El resultado podría ser:

```
perro muerde
```

Podríamos de la misma forma trabajar con variables que contuviesen números y construir nuestro programa:

```
a=3
b=4
c=a+b
imprimir(c)
```

El resultado de nuestro programa sería:

```
7
```

La utilidad de estas variables quedará de relieve en el transcurso de los siguientes capítulos.

Conceptos básicos de programación II

Funciónes y procedimientos

La función podría ser definida como un conjunto de instrucciones que permiten procesar las variables para obtener un resultado. Puede que esta definición resulte un poco vaga si no nos servimos de un ejemplo para ilustrarla.

Supongamos que queremos calcular el valor total de un pedido a partir de la simple suma de los precios de cada uno de los artículos. Podríamos definir una función suma en nuestro lenguaje ficticio:

```
definir funcion suma(art1,art2,art3)
suma=art1+art2+art3
imprimir(suma)
fin funcion
```

Este supuesto programa nos permitiría calcular la suma de tres elementos e imprimir el resultado en

pantalla. Lo interesante de utilizar este tipo de funciones es que ellas nos permiten su utilización sistemática tantas veces como queramos sin necesidad de escribir las instrucciones tantas veces como veces queremos utilizarla. Por supuesto, podemos prescindir de esta declaración de función e introducir una línea del siguiente tipo:

imprimir(art1+art2+art3)

Evidentemente, cuanto más complicada sea la función y más a menudo la utilicemos en nuestros scripts más útil resulta definirlas.

Esta función suma podría ser utilizada en cualquier lugar de nuestro script haciendo una llamada del siguiente tipo:

ejecuta suma(4,6,9)

Cuyo resultado sería:

19

Del mismo modo, los procedimientos son parecidos a las funciones. La diferencia consiste tan solo en que en estos últimos el interés no radica en el resultado obtenido sino más bien en las operaciones realizadas al ejecutarla (creación de un archivo, reenvío a otra página,...). En lenguajes como el PHP las funciones y los procedimientos son considerados como la misma cosa y para definirlos se hace usando los mismos comandos.

Tanto las variables como las funciones y los procedimientos deben ser nombradas sin servirse de acentos, espacios ni caracteres especiales para no correr riesgos de error .

Estos conceptos son básicos para una comprensión de la programación. No obstante, es posible que si es la primera vez que oímos hablar de ellos, su asimilación puede resultar parcial o nula. En realidad esto no es preocupante ya que a partir de los ejemplos de los capítulos siguientes y con la práctica de uno mismo se irán consolidando poco a poco.

Para nada hay que desanimarse si después de leer este capítulo algunas dudas quedan en el aire.

El paso siguiente es continuar el aprendizaje de un lenguaje de programación que nos sirva para construir las páginas dinámicas, con el manual de ASP o el manual de PHP, según nuestros gustos o necesidades.

Capitulo 8
Introducción a los lenguajes del web

Vamos a estudiar de manera global el mundo de la programación de páginas web. Para ello empezaremos estudiando rápidamente algunos conceptos básicos, que seguramente muchos ya sabremos, como el marco donde la web se desarrolla, qué es una página web, cómo se construye una página y el lenguaje HTML. Además veremos qué es una página estática y dinámica distinguiendo entre páginas dinámicas de cliente y servidor.

Nos detendremos con mayor profundidad en la presentación de cada uno de los lenguajes que tenemos a nuestra disposición para construir páginas web, enmarcados en el ámbito donde se ejecutan: cliente o servidor.

Para finalizar, conoceremos el lenguaje XML y las tecnologías relacionadas con él, para entender porqué es tan importante este lenguaje y cómo se desarrollan las webs que lo utilizan.

Introducción a la web

La web se encuadra dentro de Internet, no es más que un servicio de los muchos que presta la Red, entre los que podemos encontrar

- Correo electrónico
- IRC o chat
- FTP
- El propio web

1.1 Web es un sistema Hipertexto/Hipermedia

El sistema con el que está construido el web se llama hipertexto y es un entramado de páginas conectadas con enlaces.

Los sistemas de hipertexto se utilizan en otros contextos aparte del web, como la ayuda del Windows. Son muy fáciles de utilizar y también es muy fácil encontrar lo que buscamos rápidamente, gracias a que pulsando enlaces vamos accediendo a la información que más nos interesa.

La web no solo se limita a presentar textos y enlaces, sino que también puede ofrecernos imágenes, videos, sonido y todo tipo de presentaciones, llegando a ser el servicio más rico en medios que tiene Internet. Por esta razón, para referirnos al sistema que implementa el web (hipertexto), se ha acuñado un nuevo término que es hipermedia, haciendo referencia a que el web permite contenidos multimedia.

> **Referencia:** Esta introducción está extraída del Manual de Publicar en Internet. En dicho manual se puede encontrar una introducción a la web más amplia.
>
> Ir al Manual de Publicar en Internet.

Lenguaje HTML

Una página web la vemos en nuestro navegador, o cliente web, y parece una sola entidad, pero no es así, está compuesta por multitud de diferentes ficheros, como son las imágenes, los posibles vídeos y lo más importante: el código fuente.

El código de las páginas está escrito en un lenguaje llamado HTML, que indica básicamente donde colocar cada texto, cada imagen o cada video y la forma que tendrán estos al ser colocados en la página.

El HTML se creó en un principio con objetivos divulgativos. No se pensó que la web llegara a ser un área de ocio con carácter multimedia, de modo que, el HTML se creó sin dar respuesta a todos los posibles usos que se le iba a dar y a todos los colectivos de gente que lo utilizarían en un futuro.

el lenguaje consta de etiquetas que tienen esta forma <B> o <P>. Cada etiqueta significa una cosa, por ejemplo <B> significa que se escriba en negrita (bold) o <P> significa un párrafo, <A> es un enlace, etc. Casi todas las etiquetas tienen su correspondiente etiqueta de cierre, que indica que a partir de ese punto no debe de afectar la etiqueta. Por ejemplo </B> se utiliza para indicar que se deje de escribir en

negrita. Así que el HTML no es más que una serie de etiquetas que se utilizan para definir la forma o estilo que queremos aplicar a nuestro documento. <B>Esto está en negrita</B>.

Partes de un documento HTML

Un documento HTML ha de estar delimitado por la etiqueta <html> y </html>. Dentro de este documento, podemos asimismo distinguir dos partes principales:

El encabezado, delimitado por <head> y </head> donde colocaremos etiquetas de índole informativo como por ejemplo el titulo de nuestra página.

El cuerpo, flanqueado por las etiquetas <body> y </body>, que será donde colocaremos nuestro texto e imágenes delimitados a su vez por otras etiquetas como las que hemos visto.

El resultado es un documento con la siguiente estructura:

```
<html>

<head>
  Etiquetas y contenidos del encabezado
  Datos que no aparecen en nuestra página pero que son importantes para catalogarla: Titulo, palabras clave,...
</head>

<body>
  Etiquetas y contenidos del cuerpo
  Parte del documento que será mostrada por el navegador: Texto e imágenes
</body>

</html>
```

Con todo lo que conocemos ya sobre HTML podemos construir una página web que ya tiene bastante sentido. Vemos un ejemplo a continuación.

```
<html>

<head>
  <title>Cocina Para Todos</title>
</head>

<body>
  <p><b>Bienvenido,</b></p>
  <p>Estás en la página <b>Comida para Todos</b>.</p>
  <p>Aquí aprenderás recetas fáciles y deliciosas.</p>
</body>

</html>
```

Podemos ver esa página en marcha para hacernos una idea exacta de los resultados.

> **Referencia:** Para aprender todos los detalles del lenguaje HTML tenemos en DesarrolloWeb.com un Manual de HTML que hará las delicias de sus lectores.
>
> Ir al Manual de HTML.

Páginas estáticas Vs. dinámicas

En la web podemos encontrar, o construir, dos tipos de páginas:

- Las que se presentan sin movimiento y sin funcionalidades más allá de los enlaces
- Las páginas que tienen efectos especiales y en las que podemos interactuar.

Las primeras páginas son las que denominamos páginas estáticas, se construyen con el lenguaje HTML, que no permite grandes florituras para crear efectos ni funcionalidades más allá de los enlaces.

Estas páginas son muy sencillas de crear, aunque ofrecen pocas ventajas tanto a los desarrolladores

como a los visitantes, ya que sólo se pueden presentar textos planos acompañados de imágenes y a lo sumo contenidos multimedia como pueden ser videos o sonidos

El segundo tipo de páginas se denomina página dinámica. Una página es dinámica cuando se incluye cualquier efecto especial o funcionalidad y para ello es necesario utilizar otros lenguajes de programación, aparte del simple HTML.

Mientras que las páginas estáticas todo el mundo se las puede imaginar y no merecen más explicaciones, las páginas dinámicas son más complejas y versátiles. Para aclarar este concepto, veremos con detalle a continuación qué son las páginas dinámicas.

> **Referencia:** En DesarrolloWeb.com tenemos otro manual que explica también las diferencias entre páginas estáticas y dinámicas, pero desde otro punto de vista. El manual también introduce seriamente en el concepto de página dinámica y puede ser de utilidad su lectura.
>
> Ir al Manual de páginas dinámicas.

Páginas dinámicas

Como hemos visto, una página es dinámica cuando realiza efectos especiales o implementa alguna funcionalidad o interactividad.

Además, hemos visto que para programar una página dinámica necesitaremos otros lenguajes aparte del HTML. Sin embargo, nunca hay que olvidarse del HTML, ya que éste es la base del desarrollo web: generalmente al escribir una página dinámica el código de los otros lenguajes de programación se incluye embebido dentro del mismo código HTML.

Una razón por la que construiremos una página dinámica es la simple vistosidad que pueden alcanzar los trabajos, ya que podemos hacer presentaciones más entretenidas de las que se consiguen utilizando únicamente HTML. Pero vamos a ver con calma algunas razones menos obvias pero más importantes.

Supongamos que hemos decidido realizar un portal de televisión donde una de las informaciones principales a proveer podría ser la programación semanal. Efectivamente, esta información suele ser dada por las televisiones con meses de antelación y podría ser muy fácilmente almacenada en una base de datos. Si trabajásemos con páginas HTML, tendríamos que construir una página independiente para cada semana en la cual introduciríamos "a mano" cada uno de los programas de cada una de las cadenas. Asimismo, cada semana nos tendríamos que acordar de descolgar la página de la semana pasada y colgar la de la actual. Todo esto podría ser fácilmente resuelto mediante páginas dinámicas. En este caso, lo que haríamos sería crear un programa (solo uno) que se encargaría de recoger de la base de datos de la programación aquellos programas que son retransmitidos en las fechas que nos interesan y de confeccionar una página donde aparecerían ordenados por cadena y por hora de retransmisión. De este modo, podemos automatizar un proceso y desentendernos de un aspecto de la página por unos meses.

Este hecho lo podríamos aplicar a otras situaciones: podemos preparar el horóscopo de todos los días, las promociones de un sitio de e-comercio...

Podemos hacer una clasificación a las páginas dinámicas en función de dónde se lleva a cabo el procesamiento de la página, es decir, el computador que cargará con el peso adicional que supone que la página realice efectos y funcionalidades.

> **Referencia:** Cabe señalar también en este punto la existencia de otro manual que explica el concepto de página dinámica. Puede ser de utilidad su lectura.
>
> Ir al Manual de páginas dinámicas.

Páginas dinámicas de cliente

Son las páginas dinámicas que se procesan en el cliente. En estas páginas toda la carga de procesamiento de los efectos y funcionalidades la soporta el navegador.

Usos típicos de las páginas de cliente son efectos especiales para webs como rollovers o control de ventanas, presentaciones en las que se pueden mover objetos por la página, control de formularios, cálculos, etc.

El código necesario para crear los efectos y funcionalidades se incluye dentro del mismo archivo HTML y

es llamado SCRIPT. Cuando una página HTML contiene scripts de cliente, el navegador se encarga de interpretarlos y ejecutarlos para realizar los efectos y funcionalidades.

Las páginas dinámicas de cliente se escriben en dos lenguajes de programación principalmente: Javascript y Visual Basic Script (VBScript), que veremos en detalle más adelante. También veremos el concepto de DHTML y conoceremos las CSS.

> **Nota:** Flash es una tecnología, y un programa, para crear efectos especiales en páginas web. Con Flash también conseguimos hacer páginas dinámicas del lado del cliente. Como este manual explica los lenguajes del web, no hemos incluido el Flash por ninguna parte, porque no es un lenguaje. Sin embargo, si tuvieramos que catalogarlo en algún sitio quedaría dentro del ámbito de las páginas dinámicas de cliente.

Las páginas del cliente son muy dependientes del sistema donde se están ejecutando y esa es su principal desventaja, ya que cada navegador tiene sus propias características, incluso cada versión, y lo que puede funcionar en un navegador puede no funcionar en otro.

Como ventaja se puede decir que estas páginas descargan al servidor algunos trabajos, ofrecen respuestas inmediatas a las acciones del usuario y permiten la utilización de algunos recursos de la máquina local.

Páginas dinámicas de servidor

Podemos hablar también de páginas dinámicas del servidor, que son reconocidas, interpretadas y ejecutadas por el propio servidor.

Las páginas del servidor son útiles en muchas ocasiones. Con ellas se puede hacer todo tipo de aplicaciones web. Desde agendas a foros, sistemas de documentación, estadísticas, juegos, chats, etc. Son especialmente útiles en trabajos que se tiene que acceder a información centralizada, situada en una base de datos en el servidor, y cuando por razones de seguridad los cálculos no se pueden realizar en el ordenador del usuario.

Es importante destacar que las páginas dinámicas de servidor son necesarias porque para hacer la mayoría de las aplicaciones web se debe tener acceso a muchos recursos externos al ordenador del cliente, principalmente bases de datos alojadas en servidores de Internet. Un caso claro es un banco: no tiene ningún sentido que el cliente tenga acceso a toda la base de datos, sólo a la información que le concierne.

Las páginas dinámicas del servidor se suelen escribir en el mismo archivo HTML, mezclado con el código HTML, al igual que ocurría en las páginas del cliente. Cuando una página es solicitada por parte de un cliente, el servidor ejecuta los scripts y se genera una página resultado, que solamente contiene código HTML. Este resultado final es el que se envía al cliente y puede ser interpretado sin lugar a errores ni incompatibilidades, puesto que sólo contiene HTML

Luego es el servidor el que maneja toda la información de las bases de datos y cualquier otro recurso, como imágenes o servidores de correo y luego envía al cliente una página web con los resultados de todas las operaciones.

Para escribir páginas dinámicas de servidor existen varios lenguajes, que veremos con detenimiento más adelante. Common Gateway Interface (CGI) comúnmente escritos en Perl, Active Server Pages (ASP), Hipertext Preprocesor (PHP), y Java Server Pages (JSP).

Las ventajas de este tipo de programación son que el cliente no puede ver los scripts, ya que se ejecutan y transforman en HTML antes de enviarlos. Además son independientes del navegador del usuario, ya que el código que reciben es HTML fácilmente interpretable.

Como desventajas se puede señalar que será necesario un servidor más potente y con más capacidades que el necesario para las páginas de cliente. Además, estos servidores podrán soportar menos usuarios concurrentes, porque se requerirá más tiempo de procesamiento para cada uno.

Qué es Javascript

Javascript es un lenguaje de programación utilizado para crear pequeños programitas encargados de realizar acciones dentro del ámbito de una página web.

Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.

Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Javascript es el siguiente paso, después del HTML, que puede dar un programador de la web que decida mejorar sus páginas y la potencia de sus proyectos. Es un lenguaje de programación bastante **sencillo y pensado para hacer las cosas con rapidez**, a veces con ligereza. Incluso las personas que no tengan una experiencia previa en la programación podrán aprender este lenguaje con facilidad y utilizarlo en toda su potencia con sólo un poco de práctica.

Entre las acciones típicas que se pueden realizar en Javascript tenemos dos vertientes. Por un lado los **efectos especiales** sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo. Por el otro, javascript nos permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que podemos crear **páginas interactivas** con programas como calculadoras, agendas, o tablas de cálculo.

Javascript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Además, Javascript pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente.

Con Javascript el programador, que se convierte en el verdadero dueño y controlador de cada cosa que ocurre en la página cuando la está visualizando el cliente.

> **Ver también:** Para que quede claro el lenguaje y alguna aplicación práctica, que se puede hacer y entender rápidamente, se puede acceder al artículo Efectos Rápidos con Javascript. En dicho artículo veremos la implementación de un botón de volver y la muestra de la última modificación de una página web.

Referencias

En DesarrolloWeb.com hemos publicado un manual de programación en Javascript, donde explicamos toda la sintaxis y metodología de programaión.

Además, podemos acceder al Manual de Javascript II, donde vamos a tratar de acercarnos a este lenguaje en profundidad y conocer todos sus secretos y recursos disponibles.

También hemos hecho una recopilación de scripts interesantes para hacer directamente una variada gama de efectos y utilidades para páginas web. En nuestro Taller de Javascript.

Además de estos manuales, tenemos muchos otros recursos interesantes sobre el lenguaje en nuestro buscador, en la sección Javascript.

Qué es Visual Basic Script

Es un lenguaje de programación de scripts del lado del cliente, pero sólo compatible con Internet Explorer. Es por ello que su utilización está desaconsejada a favor de Javascript.

Está basado en Visual Basic, un popular lenguaje para crear aplicaciones Windows. Tanto su sintaxis como la manera de trabajar están muy inspirados en él. Sin embargo, no todo lo que se puede hacer en Visual Basic lo podremos hacer en Visual Basic Script, pues este último es una versión reducida del primero.

El modo de funcionamiento de Visual Basic Script para construir efectos especiales en páginas web es muy similar al utilizado en Javascript y los recursos a los que se puede acceder también son los mismos: el navegador.

Como decimos, no debemos utilizar este lenguaje en la mayoría de las ocasiones, aunque un caso donde tendría sentido utilizar Visual Basic Script sería la construcción de una Intranet donde sepamos con toda seguridad que los navegadores que se van a conectar serán siempre Internet Explorer. En este caso, un programador habitual de Visual Basic tendría más facilidades para realizar los scripts utilizando Visual Basic Script en lugar de Javascript.

> **Nota:** El popular ASP (Active Server Pages) es una tecnología de programación del lado del servidor. Habitualmente, los scripts ASP se escriben con Visual Basic Script también y eso no nos debe liar. Visual Basic Script, por tanto, es un lenguaje que se puede utilizar para la

programación en el cliente, pero también para la programación en el servidor.

En este artículo hemos hablado del lenguaje en su faceta del lado del cliente, puesto que en la faceta del servidor tenemos muchos manuales, pero están englobados dentro de la programación en ASP.

Referencias

En DesarrolloWeb.com tenemos un manual de Visual Basic Script, que resulta ser uno de los pocos que se encuentran el la Red sobre esta materia. Está orientado a enseñar la sintaxis y la programación del lado del cliente con el lenguaje.

DHTML

DHTML no es precisamente un lenguaje de programación. Más bien se trata de una nueva capacidad de la que disponen los navegadores modernos, por la cual se puede tener un mayor control sobre la página que antes.

Cualquier página que responde a las actividades del usuario y realiza efectos y funcionalidades se puede englobar dentro del DHTML, pero en este caso nos referimos más a efectos en el navegador por los cuales se pueden mostrar y ocultar elementos de la página, se puede modificar su posición, dimensiones, color, etc.

DHTML nos da más control sobre la página, gracias a que los navegadores modernos incluyen una nueva estructura para visualizar en páginas web denominada capa. Las capas se pueden ocultar, mostrar, desplazar, etc.

Para realizar las acciones sobre la página, como modificar la apariencia de una capa, seguimos necesitando un lenguaje de programación del lado del cliente como Javascript o VBScript.

Aclaración: DHTML también puede englobar la programación en el servidor.

Depende del autor que esté describiendo lo que es DHTML, muchas veces hace también referencia a la programación en el servidor y no sólo a la del cliente, como hemos apuntado en este artículo. Nosotros también pensamos que en cierto modo debería incluirse ese tipo de programación y así lo hemos constatado en un artículo publicado con anterioridad en DesarrolloWeb sobre Qué es DHTML.

Dicho de otro modo y para que quede claro. Las fronteras del DHTML quedan poco definidas. Las que marcamos en el presente artículo son sólo las que engloban a los procesos en el cliente, pero también podríamos decir que DHTML es cualquier cosa que hace una página dinámica, ya sea en el cliente, el servidor o las dos cosas.

Dentro del concepto de DHTML se engloban también las Hojas de Estilo en Cascada o CSS (Cascade Style Sheets), que veremos a continuación.

Qué es CSS

CSS, es una tecnología que nos permite crear páginas web de una manera más exacta. Gracias a las CSS somos mucho más dueños de los resultados finales de la página, pudiendo hacer muchas cosas que no se podía hacer utilizando solamente HTML, como incluir márgenes, tipos de letra, fondos, colores...

CSS son las siglas de Cascading Style Sheets, en español Hojas de estilo en Cascada. En este reportaje vamos a ver algunos de los efectos que se pueden crear con las CSS sin necesidad de conocer la tecnología entera.

Para empezar

Las Hojas de Estilo en Cascada se escriben dentro del código HTML de la página web, solo en casos avanzados se pueden escribir en un archivo a parte y enlazar la página con ese archivo. En un principio vamos a utilizar la manera más directa de aplicar estilos a los elementos de la página, mas adelante veremos la declaración en archivos externos. Para ello, y esto es la primera lección de este artículo, vamos a conocer un nuevo atributo que se puede utilizar en casi todas las etiquetas HTML: style.

Ejemplo:

```
<p style="color:green;font-weight:bold">El párrafo saldrá con color verde y en negrita</p>
```

Dentro del atributo style se deben indicar los atributos de estilos CSS separados por punto y coma (;). Durante este artículo vamos a conocer muchos atributos de CSS, los dos primeros que hemos visto aquí son:

Color: indica el color del contenido la etiqueta donde estemos utilizándolo, generalmente indica el color del texto.

Font-weight: indica el grosor del texto. Bold sirve para poner en negrita.

Color en los enlaces

Con HTML definimos el color de los enlaces en la etiqueta <body>, con lo atributos link, vlink y alink. Esto nos permite cambiar el color de los enlaces para todo el documento, pero ¿Y si queremos cambiar el color de un enlace en concreto, para que tenga otro color que el definido en la etiqueta <body>?

Para hacer esto utilizaremos el atributo style dentro del enlace:

```
<a href="mienlace.html" style="color:red">
```

Así saldrá el enlace en color rojo, independientemente de lo definido para todo el documento.

Espaciado entre líneas

Con CSS podemos definir el espacio que hay entre cada línea del documento, utilizando el atributo line-height. Por ejemplo, podemos definir que para todo un párrafo el espacio entre cada una de sus líneas sea 25 pixels:

```
<p style="line-height: 25px;">
Un párrafo normal en el que cada una de las líneas está separada 25 pixels de la otra. Hay que poner suficiente texto como para que se vean 2 líneas, así saldrán separadas
</p>
```

Espaciado entre caracteres

Se puede definir también el espacio entre cada carácter. Esto se hace con el atributo de CSS letter-spacing. Veamos un ejemplo:

```
<p style="letter-spacing:12cm">
Este párrafo tiene las letras espaciadas por 1 centímetro.
</p>
```

Este atributo, al igual que ocurre con muchos otros de CSS, no está soportado por todos los navegadores. En concreto Netscape, en su versión 4 todavía no lo incluye.

Enlaces sin subrayado

Uno de los efectos más significativos y fáciles de realizar con CSS es eliminar el subrayado de los enlaces de una página web. Existe un atributo que sirve para definir la decoración de un texto, si está subrayado, tachado, o si no tiene ninguna de estas "decoraciones". Es el atributo text-decoration, en este caso indicaremos en un enlace que no queremos decoración:

```
<a href="mipagina.html" style="text-decoration:none">
```

Incluir estilos para todo un sitio web

Una de las características más potentes de la programación con hojas de estilo consiste en definir los estilos de todo un sitio web. Esto se consigue creando un archivo donde tan sólo colocamos las

declaraciones de estilos de la página y enlazando todas las páginas del sitio con ese archivo. De este modo, todas las páginas comparten una misma declaración de estilos y, por tanto, si la cambiamos, cambiarán todas las páginas.

Veamos ahora todo el proceso para incluir estilos con un fichero externo.

1- Creamos el fichero con la declaración de estilos

Es un fichero de texto normal, que puede tener cualquier extensión, aunque le podemos asignar la extensión .css para aclararnos qué tipo de archivo es. El texto que debemos incluir debe ser escrito exclusivamente en sintaxis CSS, es un poco distinta que la sintaxis que utilizamos dentro del atributo style. Sería erróneo incluir código HTML en este archivo: etiquetas y demás. Podemos ver un ejemplo a continuación.

```
P {
font-size : 12pt;
font-family : arial,helvetica;
font-weight : normal;
}
H1 {
font-size : 36pt;
font-family : verdana,arial;
text-decoration : underline;
text-align : center;
background-color : Teal;
}
BODY {
background-color : #006600;
font-family : arial;
color : White;
}
```

2- Enlazamos la página web con la hoja de estilos

Para ello vamos a colocar la etiqueta <LINK> con los atributos

- rel="STYLESHEET" indicando que el enlace es con una hoja de estilo.
- type="text/css" porque el archivo es de texto, en sintaxis CSS.
- href="estilos.css" indica el nombre del fichero fuente de los estilos.

Veamos una página web entera que enlaza con la declaración de estilos anterior:

```
<html>
<head>
<link rel="STYLESHEET" type="text/css" href="estilos.css">
<title>Página que lee estilos</title>
</head>
<body>
<h1>Página que lee estilos</h1>
<p>
Esta página tiene en la cabecera la etiqueta necesaria para enlazar con la hoja de estilos. Es muy fácil.
</p>
</body>
</html>
```

Las CSS tienen mucho más jugo

Las Hojas de Estilo en Cascada son un estándar muy amplio, con unas especificaciones y posibilidades muy grandes. En este artículo hemos visto unos cuantos efectos interesantes que realizar aunque no tengamos ningún conocimiento previo. Sin embargo, lo mejor para trabajar con esta tecnología es conocerla bien, gracias a ello, los resultados serán mucho más sorprendentes.

Para ampliar esta información y conocer más sobre CSS se puede encontrar un manual en desarrolloweb.com www.desarrolloweb.com/manuales/2

Qué son los Applets de Java

Es otra manera de incluir código a ejecutar en los clientes que visualizan una página web. Se trata de

pequeños programas hechos en Java, que se transfieren con las páginas web y que el navegador ejecuta en el espacio de la página.

Los applets de Java están programados en Java y precompilados, es por ello que la manera de trabajar de éstos varía un poco con respecto a los lenguajes de script como Javascript. Los applets son más difíciles de programar que los scripts en Javascript y requerirán unos conocimientos básicos o medios del lenguaje Java.

La principal ventaja de utilizar applets consiste en que son mucho menos dependientes del navegador que los scripts en Javascript, incluso independientes del sistema operativo del ordenador donde se ejecutan. Además, Java es más potente que Javascript, por lo que el número de aplicaciones de los applets podrá ser mayor.

Como desventajas en relación con Javascript cabe señalar que los applets son más lentos de procesar y que tienen espacio muy delimitado en la página donde se ejecutan, es decir, no se mezclan con todos los componentes de la página ni tienen acceso a ellos. Es por ello que con los applets de Java no podremos hacer directamente cosas como abrir ventanas secundarias, controlar Frames, formularios, capas, etc.

Cómo es posible la multiplataforma en Java

Java es compatible con todos los sistemas porque basa su funcionamiento en los Byte Codes, que no es más que una precompilación del código fuente de Java.

Estos Byte Codes no son el programa en Java propiamente dicho, sino un archivo que contiene un código intermedio que puede manejar la Máquina Virtual de Java. Cada sistema operativo dispone de una Maquina Virtual de Java que puede interpretar los Byte Codes y transformarlos a sentencias ejecutables en el sistema en cuestión.

Qué es CGI

Es el sistema más antiguo que existe para la programación de las páginas dinámicas de servidor. Actualmente se encuentra un poco desfasado por diversas razones entre las que destaca la dificultad con la que se desarrollan los programas y la pesada carga que supone para el servidor que los ejecuta.

Los CGI se escriben habitualmente en el lenguaje Perl, sin embargo, otros lenguajes como C, C++ o Visual Basic pueden ser también empleados para construirlos.

El funcionamiento básico de un programa CGI es parecido al apuntado para el conjunto de las páginas dinámicas del servidor, con algunas particularidades.

1. Se realiza una petición http, a la que pueden acompañar datos llegados o bien por un formulario o bien a través de la URL.
2. El servidor ejecuta los programas CGI a los que se accede y trabaja con los recursos necesarios para llevar a cabo las acciones, como por ejemplo bases de datos.
3. El programa CGI va escribiendo en la salida estándar el resultado de la ejecución del CGI, que incluye etiquetas HTML, ya que lo que se escribe es una página web.

Algunas desventajas de la programación en CGI son las siguientes:

- Los resultados se escriben directamente con el CGI, así que el código del programa se mezcla con el del HTML haciendo difícil su comprensión y mantenimiento.
- Cada programa CGI que se pone en marcha lo hace en un espacio de memoria propio. Así, si tres usuarios ponen en marcha un CGI a la vez se multiplicará por tres la cantidad de recursos que ocupe ese CGI. Esto significa una grave ineficiencia.

Para completar esta información sería interesante acceder a la sección CGI de nuestro directorio de enlaces, donde podemos encontrar sitios en Internet que ofrecen tutoriales sobre la tecnología y directorios de programas CGI ya creados para hacer cosas tan variadas como tiendas, foros, envio de formularios, etc.

Qué es Perl

Es un lenguaje de programación muy utilizado para construir aplicaciones CGI para el web. Perl es un acrónimo de Practical Extracting and Reporting Languaje, que viene a indicar que se trata de un lenguaje de programación muy práctico para extraer información de archivos de texto y generar informes a partir

del contendido de los ficheros.

Es un lenguaje libre de uso, eso quiere decir que es gratuito. Antes estaba muy asociado a la plataforma Uníx, pero en la actualidad está disponible en otros sistemas operativos como Windows.

Perl es un lenguaje de programación interpretado, al igual que muchos otros lenguajes de Internet como Javascript o ASP. Esto quiere decir que el código de los scripts en Perl no se compila sino que cada vez que se quiere ejecutar se lee el código y se pone en marcha interpretando lo que hay escrito. Además es extensible a partir de otros lenguajes, ya que desde Perl podremos hacer llamadas a subprogramas escritos en otros lenguajes. También desde otros lenguajes podremos ejecutar código Perl.

Perl está inspirado a partir de lenguajes como C, sh, awk y sed (algunos provenientes de los sistemas Uníx), pero está enfocado a ser más práctico y fácil que estos últimos. Es por ello que un programador que haya trabajado con el lenguaje C y los otros tendrá menos problemas en entenderlo y utilizarlo rápidamente. Una diferencia fundamental de Perl con respecto a los otros lenguajes es que no limita el tamaño de los datos con los que trabaja, el límite lo pone la memoria que en ese momento se encuentre disponible.

Si queremos trabajar con Perl será necesario tener instalado el interprete del lenguaje. A partir de ese momento podemos ejecutar CGIs en nuestros servidores web. El proceso para conseguirlo puede variar de unos servidores a otros, pero se suelen colocar en un directorio especial del servidor llamado cgi-bin donde hemos colocado los correspondientes permisos CGI. Además, los archivos con el código también deberán tener permiso de ejecución.

Este informe se complementa con los enlaces que podéis encontrar en la sección Perl de nuestro buscador y ocasionalmente la sección CGI.

Qué es ASP

ASP (Active Server Pages) es la tecnología desarrollada por Microsoft para la creación de páginas dinámicas del servidor. ASP se escribe en la misma página web, utilizando el lenguaje Visual Basic Script o Jscript (Javascript de Microsoft).

Un lenguaje del lado del servidor es aquel que **se ejecuta en el servidor web**, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores. Podemos saber algo más sobre la programación del servidor y del cliente en el artículo qué es DHTML.

El tipo de servidores que emplean este lenguaje son, evidentemente, todos aquellos que funcionan con sistema Windows NT, aunque también se puede utilizar en un PC con windows 98 si instalamos un servidor denominado Personal Web Server. Incluso en sistemas Linux podemos utilizar las ASP si intalamos un componente denominado Chilisoft, aunque parece claro que será mejor trabajar sobre el servidor web para el que está pensado: Internet Information Server.

Con las ASP podemos realizar muchos tipos de aplicaciones distintas. Nos permite acceso a bases de datos, al sistema de archivos del servidor y en general a todos los recursos que tenga el propio servidor. También tenemos la posibilidad de comprar componentes ActiveX fabricados por distintas empresas de desarrollo de software que sirven para realizar múltiples usos, como el envio de correo, generar gráficas dinámicamente, y un largo etc.

Actualmente se ha presentado ya la segunda versión de ASP, el ASP.NET, que comprende algunas mejoras en cuanto a posibilidades del lenguaje y rapidez con la que funciona. ASP.NET tiene algunas diferencias en cuanto a sintaxis con el ASP, de modo que se ha de tratar de distinta manera uno de otro.

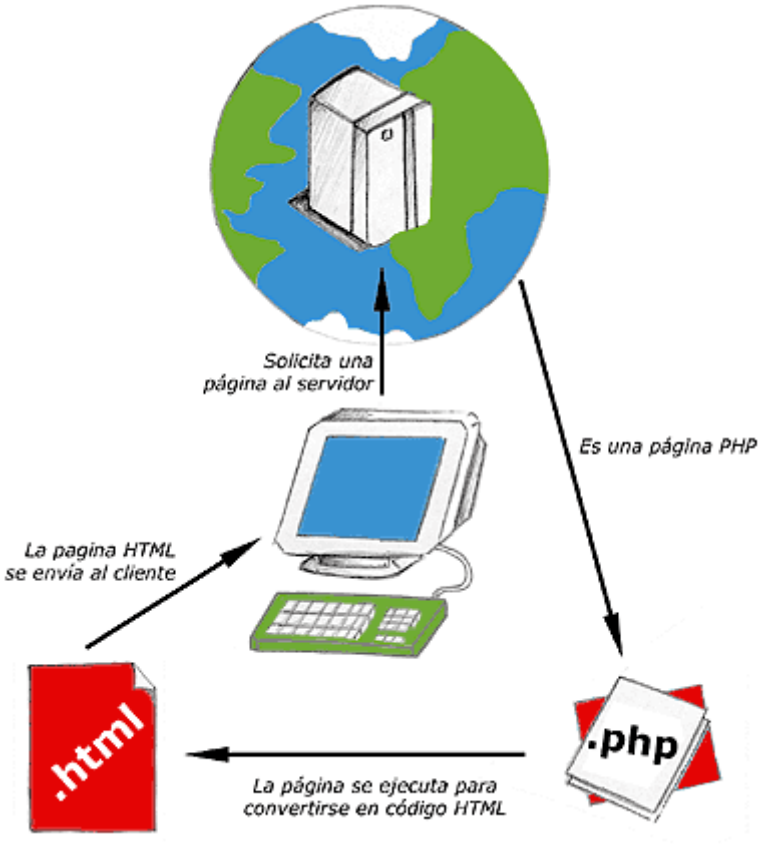
Para enterarnos bien de lo que es ASP y aprender a programar en el lenguaje disponemos de una **sección en DesarrolloWeb dedicada por entero a tratar ASP a fondo**.

Qué es PHP

PHP es el acrónimo de Hipertext Preprocesor. **Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma**, rápido, con una gran librería de funciones y mucha documentación.

Un lenguaje del lado del servidor es aquel que **se ejecuta en el servidor web**, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden

realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores. Podemos saber algo más sobre la programación del servidor y del cliente en el artículo qué es DHTML.



Esquema del funcionamiento de las páginas PHP.

Una vez que ya conocemos el concepto de lenguaje de programación de scripts del lado del servidor podemos hablar de PHP. **PHP se escribe dentro del código HTML**, lo que lo hace realmente fácil de utilizar, al igual que ocurre con el popular ASP de Microsoft, pero con algunas ventajas como su gratuidad, independencia de plataforma, rapidez y seguridad. Cualquiera puede descargar a través de la página principal de PHP http://www.php.net/ y de manera gratuita, un módulo que hace que nuestro servidor web comprenda los scripts realizados en este lenguaje. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante, ya que permite portar el sitio desarrollado en PHP de un sistema a otro sin prácticamente ningún trabajo.

PHP, en el caso de estar montado sobre un servidor Linux u Unix, es más rápido que ASP, dado que se ejecuta en un único espacio de memoria y esto evita las comunicaciones entre componentes COM que se realizan entre todas las tecnologías implicadas en una página ASP.

Por último señalábamos la seguridad, en este punto también es importante el hecho de que en muchas ocasiones PHP se encuentra instalado sobre servidores Unix o Linux, que son de sobra conocidos como más veloces y seguros que el sistema operativo donde se ejecuta las ASP, Windows NT o 2000. Además, PHP permite configurar el servidor de modo que se permita o rechacen diferentes usos, lo que puede hacer al lenguaje más o menos seguro dependiendo de las necesidades de cada cual.

Fue creado originalmente en 1994 por Rasmus Lerdorf, pero como **PHP está desarrollado en política de código abierto**, a lo largo de su historia ha tenido muchas contribuciones de otros desarrolladores. Actualmente PHP se encuentra en su **versión 4, que utiliza el motor Zend**, desarrollado con mayor meditación para cubrir las necesidades de las aplicaciones web actuales.

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red, por poner dos ejemplos.

Algunas de las más importantes capacidades de PHP son: **compatibilidad con las bases de datos** más comunes, como MySQL, mSQL, Oracle, Informix, y ODBC, por ejemplo. Incluye funciones para el **envío de correo electrónico**, **upload de archivos**, crear **dinámicamente en el servidor imágenes en formato GIF**, incluso animadas y una lista interminable de utilidades adicionales.

Se puede obtener mucha más información de PHP en los diversos manuales de PHP que hemos publicado en desarrolloweb. Manuales que atienden a los diversos niveles de cada programdor:

- Programación en PHP: Aprende PHP desde el principio con este manual que toca las prestaciones más imprescindibles.
- Programación en PHP II: Aprende a hacer una aplicación práctica en PHP. Un sistema de postales.
- Taller de PHP: otros artículos prácticos más avanzados con el lenguaje.

Qué es JSP

JSP es un acrónimo de Java Server Pages, que en castellano vendría a decir algo como Páginas de Servidor Java. Es, pues, una tecnología orientada a crear páginas web con programación en Java.

> **Bibliografía:** Esta descripción de JSP está extraída de un PDF en inglés muy completo que introduce la tecnología, que se puede encontrar el la página corporativa de Java de Sun Microsystems, en su sección de aprendizaje online. A su vez, dicho manual proviene del portal Java **jGuru**.
>
> jGuru: Introduction to JavaServer Pages technology

Con JSP podemos crear aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java. Por tanto, las JSP podremos escribirlas con nuestro editor HTML/XML habitual.

Motor JSP

El motor de las páginas JSP está basado en los servlets de Java -programas en Java destinados a ejecutarse en el servidor-, aunque el número de desarrolladores que pueden afrontar la programación de JSP es mucho mayor, dado que resulta mucho más sencillo aprender que los servlets.

En JSP creamos páginas de manera parecida a como se crean en ASP o PHP -otras dos tecnologías de servidor-. Generamos archivos con extensión .jsp que incluyen, dentro de la estructura de etiquetas HTML, las sentencias Java a ejecutar en el servidor. Antes de que sean funcionales los archivos, el motor JSP lleva a cabo una fase de traducción de esa página en un servlet, implementado en un archivo class (Byte codes de Java). Esta fase de traducción se lleva a cabo habitualmente cuando se recibe la primera solicitud de la página .jsp, aunque existe la opción de precompilar en código para evitar ese tiempo de espera la primera vez que un cliente solicita la página.

Ejemplo de página JSP

En la imagen siguiente se puede ver un ejemplo extremadamente simple de una página JSP y el esquema de conversión de esa página en un servlet.

Prerequisitos

Para aprender JSP, aparte de conocer HTML, será necesario comprender y tener algo de experiencia en la programación en Java, que es un lenguaje de programación Orientado a Objetos por completo. Una vez conocida la programación en Java se puede estudiar por encima el sistema de Servlets, lo que nos dará una mejor idea del funcionamiento interno del motor JSP.

Para aprender Java podemos consultar algunos enlaces del correspondiente directorio de nuestro buscador de enlaces.

Además, necesitaremos descargar e instalar Tomcat, el contenedor de servlets usado en la referencia oficial de implementación de JSP. Podemos acceder a un ejercicio para aprender a realizar esta instalación, disponible también en la referencia de aprendizaje de la página de Java.

Referencias JSP

Hemos creado una nueva sección en nuestro directorio dedicada por completo a las páginas JSP, que será muy interesante para todo aquel que desee profundizar en el tema.

Comparando JSP con ASP

JSP y ASP sirven para hacer, más o menos, el mismo tipo de aplicaciones web. Sin embargo, en el fondo tienen bastantes diferencias. Después de mi experiencia en el trabajo con JSP, un día un cliente me preguntó por qué no programaba la página en ASP en lugar de JSP, ya que había oido hablar que el sistema de Microsoft tenía unas características muy apropiadas para su modelo de negocio. A partir de esta sugerencia, y para que mi cliente quedase satisfecho con la tecnología JSP -que es la que prefiero utilizar-, preparé una lista de ventajas de utilizar páginas dinámicas Java frente a las de Microsoft.

Plataforma e independencia del servidor

JSP sigue la filosofía de la arquitectura JAVA de "escribe una vez ejecuta donde quieras". La implantación de ASP está limitada para arquitecturas basadas en tecnología Microsoft.

Así, JSP se puede ejecutar en los sistemas operativos y servidores web más populares, como por ejemplo Apache, Netscape o Microsoft IIS. Mientras que ASP sólo tiene soporte nativo para los servidores IIS y Personal Web Server, que son los dos servidores web para sistemas Microsoft, el primero con tecnología NT y el segundo para sistemas Windows 98 y similares.

Proceso de desarrollo abierto (open source)

El API JSP se beneficia de la extendida comunidad JAVA existente, por el contrario la tecnología ASP es específica de Microsoft que desarrolla sus procesos internamente.

TAGS

Mientras que tanto JSP como ASP usan una combinación de tags y scripts para crear paginas web dinámicas, la tecnología JSP permite a los desarrolladores crear nuevos tags. Así los desarrolladores pueden crear nuevos tags y no depender tanto de los scripts.

Reusabilidad entre plataformas.

Los componentes JSP son reusables en distintas plataformas (UNIX, Windows).

La ventaja Java

La tecnología JSP usa Java como lenguaje de Script mientras que ASP usa VBScript o Jscript. Java es un lenguaje mas potente y escalable que los lenguajes de Script. Las páginas JSP son compilados en Servlets por lo que actúan como una puerta a todos los servicios Java de Servidor y librerías Java para aplicaciones http. Java hace el trabajo del desarrollador más fácil p. e. ayuda a proteger el sistema contra las "caídas" mientras que las aplicaciones ASP sobre sistemas NT son más susceptibles a sufrirlas, también ayuda en el manejo de la memoria protegiendo contra fallos de memoria y el duro trabajo de buscar los fallos de perdida de punteros de memoria que pueden hacer mas lento el funcionamiento de una aplicación.

Mantenimiento

Las aplicaciones que usan JSP tiene un mantenimiento más fácil que las que usan ASP.

- Los lenguajes de Script están bien para pequeñas aplicaciones, pero no encajan bien para aplicaciones grandes. Java es un lenguaje estructurado y es más fácil de construir y mantenimientos grandes como aplicaciones modulares.
- La tecnología JSP hace mayor énfasis en los componentes que en los Scripts, esto hace que sea más fácil revisar el contenido sin que afecte a la lógica o revisar la lógica sin cambiar el contenido.
- La arquitectura EJB encapsula la lógica de p. e.: acceso a BD, seguridad, integridad transaccional y aislamiento de la aplicación.
- Debido a que la tecnología JSP es abierta y multiplataforma, los servidores web, plataformas y otros componentes pueden ser fácilmente actualizados o cambiados sin que afecte a las aplicaciones basadas en la tecnología JSP.

Conclusión

Las ventajas sobre utilizar la tecnología Java con respecto a la propietaria de Microsoft (ASP) son, como se ha podido ver, diversas e interesantes. Sin embargo, podemos apuntar una ventaja de la

programación en ASP, pues resulta bastante más fácil de aprender que JSP, por lo menos si no se tiene una experiencia previa en programación. Esto es debido a que Java es un lenguaje muy potente, pero un poco más complicado de usar porque es orientado a objetos y la manera de escribir los programas es más rígida.

Qué es XML

XML es una tecnología en realidad muy sencilla que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Vamos a ver a lo largo de varios capítulos una introducción al mundo XML, es decir, al lenguaje así como a las tecnologías que trabajan con él, sus usos, ventajas y modos de llevar a cabo las tareas.

XML, con todas las tecnologías relacionadas, representa una manera distinta de hacer las cosas, más avanzada, cuya principal novedad consiste en permitir compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. Así pues, el XML juega un papel importantísimo en este mundo actual, que tiende a la globalización y la compatibilidad entre los sistemas, ya que es la tecnología que permitirá compartir la información de una manera segura, fiable, fácil. Además, XML permite al programador y los soportes dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corre a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello.

Vemos que XML no está sólo, sino que hay un mundo de tecnologías alrededor de él, de posibilidades, maneras más fáciles e interesantes de trabajar con los datos y, en definitiva, un avance a la hora de tratar la información, que es en realidad el objetivo de la informática en general. XML, o mejor dicho, el mundo XML no es un lenguaje, sino varios lenguajes, no es una sintaxis, sino varias y no es una manera totalmente nueva de trabajar, sino una manera más refinada que permitirá que todas las anteriores se puedan comunicar entre si sin problemas, ya que los datos cobran sentido. Todo esto lo veremos con calma en la Introducción a XML.

XML es interesante en el mundo de Internet y el e-bussiness, ya que existen muchos sistemas distintos que tienen que comunicarse entre si, pero como se ha podido imaginar, interesa por igual a todas las ramas de la informática y el tratamiento de datos, ya que permite muchos avances a la hora de trabajar con ellos.

En la introducción a XML, a lo largo de los siguientes capítulos, vamos a ver algunas características importantes de la tecnología que nos permitirán comprender mejor el mundo XML y cómo soluciona nuestros problemas a la hora de trabajar con los datos.

Capitulo 9
Programación en ASP

Principios básicos para la programación en ASP, el lenguaje del lado del servidor creado por Microsoft . Manual asequible para no programadores que sienta los fundamentos básicos de este lenguaje. Continuación lógica del manual de páginas dinámicas.

Introducción a la programación en ASP

Tal como hemos explicado, **ASP (Active Server Pages) es la tecnología para la creación de páginas dinámicas del lado del servidor desarrollada por Microsoft**.

El tipo de servidores que emplean este lenguaje son aquellos que funcionan con sistema operativo de la familia de Windows NT. Afortunadamente, también podemos visualizar páginas ASP sobre Windows 95/98, pero esto lo veremos más adelante.

Para escribir páginas ASP utilizamos un lenguaje de scripts, que se colocan en la misma página web junto con el código HTML. Comúnmente este lenguaje de scripts es **Visual Basic Script**, que deriva del conocido Visual Basic, aunque también se pueden escribir los scripts ASP en otro lenguaje: JScript, que deriva a su vez del conocido Javascript.

Existe una versión de Visual Basic Script en el lado cliente y otra en el lado del servidor. En los dos casos, como su nombre indica, el lenguaje de base es Visual Basic por lo que su aprendizaje puede ser perfectamente coordinado, ya que las sentencias y las sintaxis son prácticamente las mismas. En ASP, al estar programando páginas del lado del servidor, utlizaremos Visual Basic Script del lado del servidor y en este manual nos centraremos en este punto. El lector interesado por la sintaxis de Visual Basic Script y su programación del lado del cliente puede encontrar en este mismo sitio otro manual para tratar exclusivamente Visual Basic Script.

Este manual va destinado a aquellos que quieren comenzar de cero el aprendizaje de este lenguaje y que buscan en él la aplicación directa a su proyecto de sitio o a la mejora de su sitio HTML. Los capítulos son extremadamente simples, sino simplistas, buscando ser accesibles a la mayoría. Ellos serán complementados posteriormente con otros artículos de mayor nivel destinados a gente más experimentada.

Antes de comenzar a leer este manual es altamente aconsejable, sino imprescindible, haber leído previamente el manual sobre páginas dinámicas en el cual se explica a grandes rasgos qué es el ASP, algunos conceptos útiles sobre el modo de trabajar con páginas dinámicas al mismo tiempo que nos introduce algunos elementos básicos de la programación como pueden ser las variables y las funciones.

Del mismo modo, puede resultar extremadamente útil el haber leído o leer inmediatamente después el manual de Visual Basic Script en el cual se explica más en profundidad el lenguaje Visual Basic que resulta ser utilizado en la mayoría de scripts ASP.

Otra referencia a la cual haremos alusión es el tutorial de SQL que nos será de gran ayuda para el tratamiento de bases de datos.

Nuestra intención es la de ir publicando paulatinamente diferentes capítulos de modo que rogamos un poco de paciencia a aquellos que estén esperando la continuación. Todo irá llegando.

Esperamos que este manual resulte de vuestro agrado y que corresponda a nuestras expectativas: El poder acercar este lenguaje a todos aquellos amantes del desarrollo de webs que quieren dar el paso hacia las webs "profesionales".

Los scripts que usamos en estos primeros ejemplos pueden ser descargados aquí.

Si te interesa trabajar con un editor específico de ASP te recomendamos el MS Visual Interdev. Otra posibilidad es el Drumbeat de Macromedia aunque para empezar ninguno de los dos resulta absolutamente indispensable. También podemos elegir Homesite, un editor que no es específico para las ASP, pero que se comporta bastante bien y ofrece ayudas interesantes.

Pasos previos I: Instalación del PWS

En capítulos anteriores hemos explicado que, dada la naturaleza de los lenguajes de lado servidor, nos es imposible trabajar offline como hacíamos para el caso de las páginas HTML que almacenábamos en

nuestro disco duro. También dijimos que esto no era completamente cierto ya que podíamos resolver este eventual problema instalándonos en nuestro PC un servidor propio. Este servidor distribuido por Microsoft tiene dos versiones diferentes que son utilizadas dependiendo del equipo que estemos utilizando. Para los usuarios de W95 o W98, la versión disponible se llama Personal Web Server (PWS).

Si trabajamos bajo sistema Windows NT, o las versiones Profesional y Server de Windows 2000 y XP, el servidor a instalar es el Internet Information Server (IIS). En este caso os referimos a nuestro manual sobre la Instalación de IIS en Windows XP profesional.

Existe también la posibilidad de trabajar en plataformas UNIX empleando en este caso el ChilisoftASP.

Los usuarios de W95 tienen varias posibilidades para hacerse con el PWS: Descargarlo del sitio Microsoft , a partir de una antigua versión de Frontpage 98, instalándolo desde la opción pack de W-NT 4.0 o desde el CD de instalación de W98 (directorio add-ons/pws).

Los usuarios de Windows ME no tienen soporte para PWS, aunque se pueden probar una serie de pasos para lograr utilizarlo en el sistema. Este documento de Microsoft explica mejor este asunto.

Por otro lado, los usuarios de Windows 2000 deben utilizar IIS 5.0, que se encuentra en la instalación. Es recomendable que leáis también las notas de los visitantes al pie de página, porque encontraréis muchos más datos sobre problemas en distintas versiones y compatibilidades con otros sistemas que van apareciendo.

Algunas versiones del PWS anteriores a la 4.0 requieren un archivo adicional asp.exe para poder reconocer páginas ASP.

PWS podría ser considerado como una versión "light" del IIS4. En realidad en PWS no es suficientemente versátil para ejercer de servidor de un sitio de un tamaño mediano aunque si que podría en un momento dado hacerse cargo de un sitio de tamaño reducido y no muy concurrido. De todas formas, la utilidad del PWS radica sobre todo en que nos permite realizar las pruebas del sitio que vayamos a desarrollar en "local" sin necesidad de colgar nuestros archivos en el servidor que alberga nuestro sitio cada vez que queramos hacer una prueba sobre una pequeña modificación introducida. Esto resulta a todas luces práctico sobre todo para principiantes que necesitan hacer pruebas con una relativa frecuencia permitiendo el ahorro de mucho tiempo.

Dado que la mayoría de los posibles lectores de este manual trabajan en entorno W95 y 98, en este capítulo nos limitaremos a la descripción del PWS dejando el IIS4 para futuros capítulos. Sin embargo, las explicaciones que damos pueden ser igualmente útiles para quienes tengan que utilizar este último el cual presenta un funcionamiento básico análogo. El uso del PWS es extremadamente fácil. Una vez instalado, podemos observar la introducción de un nuevo icono en la barra de tareas así que en el menú de inicio correspondientes a la aplicación. A partir de cualquiera de ellos podemos tener acceso a la página principal o gestionario.

El siguiente paso es crear un directorio virtual dentro del cual alojaremos nuestra página. Hablamos de directorio virtual debido a que nuestra página puede estar alojada en cualquier parte de nuestro disco duro, donde a nosotros nos plazca y con un nombre de directorio que tampoco tiene por qué parecerse al que incluiremos en la URL cuando queramos ejecutar la página. De hecho, la URL que debemos introducir en el navegador para visualizar nuestra página ASP es del tipo:
http://localhost/nombre_del_directorio virtual/archivo.asp

Como se puede observar, en este tipo de dirección no se especifica el camino en el disco duro donde se encuentran nuestros archivos.

Volviendo a la creación de nuestro directorio virtual, para hacerlo debemos pinchar sobre el icono "Avanzado" el cual nos da acceso a las opciones avanzadas del PWS. Una vez ahí, el siguiente paso es "Agregar" un directorio virtual. Una ventana en la que tendremos que introducir el nombre de dicho directorio virtual y especificar en qué carpeta del disco duro tenemos guardados los archivos y carpetas de la página aparecerá.

Como puede verse, la cosa es fácil. Ahora no queda más que introducir en el navegador el tipo de URL mencionada anteriormente para ejecutar los scripts creados.

Una opción interesante en el menú avanzado es la selección del tipo de archivo que será ejecutado por defecto. Aquí podríamos poner archivos con nombre index.html o index.asp o bien con el nombre default o home...

Pasos previos II: Conexión a BD

El siguiente paso, una vez instalado el servidor que nos permite trabajar en local, es crear los vínculos con las bases de datos que explotaremos en nuestros scripts. En efecto, la utilización de páginas dinámicas está muy frecuentemente asociada con el empleo de bases de datos.

Una base de datos es sencillamente un conjunto de tablas en las que almacenamos distintos registros (artículos de una tienda virtual, proveedores o clientes de una empresa, películas en cartelera en el cine...). Estos registros son catalogados en función de distintos parámetros que los caracterizan y que presentan una utilidad a la hora de clasificarlos. Así, por ejemplo, los artículos de una tienda virtual podrían catalogarse a partir de distintos campos como puede ser un número de referencia, nombre del artículo, descripción, precio, proveedor...

Las bases de datos son construidas sirviéndose de aplicaciones tales como el Microsoft Access o el MySQL las cuales resultan bastante sencillas de utilizar con unos conceptos mínimos.

Nuestro objeto aquí no es explicar la forma de explotarlas sino cómo establecer una conexión entre la base de datos, almacenada en cualquier lugar del disco duro y nuestra página web alojada también en cualquier parte y reconocida por nuestro servidor personal a partir del directorio virtual.

Para crear este vínculo, nos servimos de los conectores ODBC (Open DataBase Connectivity) los cuales establecen el enlace con la base de datos.

El primer paso para crear esta conexión es ir al panel de configuración y abrir el icono ODBC 32bits. Dentro de él, deberemos crear un DSN (Data Source Name) de tipo sistema o usuario. Para ello nos colocamos en la solapa correspondiente (DSN sistema o DSN usuario) y seleccionamos "Añadir". A continuación se nos pedirá seleccionar los controladores de la aplicación que hemos utilizado para crear la base de datos, el nombre que le queremos asignar (aquel que empleemos en nuestros scripts) y el camino para encontrarla en el disco duro.

Esta DSN permite en realidad definir la base de datos que será interrogada sin necesidad de pasar por la aplicación que hayamos utilizado para construirla, es decir, con simples llamadas y órdenes desde nuestros archivos ASP podremos obtener los datos que buscamos sin necesidad de ejecutar el Access o el MySQL los cuales, evidentemente, no tendrán por qué encontrarse en el servidor donde trabajemos.

Inicio a la programación en ASP

A lo largo de los capítulos precedentes nos ha quedado claro que el ASP es un lenguaje orientado a las aplicaciones en red creado por Microsoft que funciona del lado servidor. Es en efecto el servidor quien se ocupa de ejecutarlo, interpretarlo y enviarlo al cliente (navegador) en forma de código HTML.

ASP es principalmente utilizado sirviéndose del lenguaje Visual Basic Script que no es más que una versión light del Visual Basic. Sin embargo, es posible programar páginas ASP en Java Script. Lo único que hay que hacer es especificar en la propia página qué tipo de lenguaje estamos utilizando.

Dado que el lenguaje ASP está muy frecuentemente embebido dentro del código HTML, es importante poder marcar al servidor qué partes están escritas en un lenguaje y cuáles en otro. Es por ello que todas las partes del archivo que están escritas en ASP estarán siempre delimitadas por los símbolos: <% y %>.

De este modo, cuando realicemos nuestros scripts, lo primero que debemos definir es el tipo de lenguaje utilizado, lo cual se hace del siguiente modo:

<% @ LANGUAGE="VBSCRIPT" %> Para el caso en el que programemos en Visual Basic Script

<% @ LANGUAGE="JSCRIPT" %> Si nos servimos del Java Script en servidor para programar en ASP

Los scripts que serán presentados en este manual estarán basados en el VBS, el cual presenta toda una serie de prestaciones que lo hacen sin duda más accesible y apto para ASP. No es por nada que es el propio Microsoft quien ha creado ambos.

Con los elementos que hemos presentado hasta ahora, ya estamos en situación de poder escribir nuestro primer programa en ASP. Vamos a crear un programa que calcule el 20% de impuestos que habría que añadir a una serie de artículos. Para plasmar el concepto de función, explicado en el manual de páginas dinámicas, vamos a definir una función "impuesto" que emplearemos sucesivas veces. El programa podría resultar algo así:

```
<% @ LANGUAGE="VBSCRIPT" %>
```

```
<HTML>
<HEAD>
<TITLE>Funcion impuesto</TITLE>
</HEAD>
<BODY>
<%Function impuesto(precio_articulo)
precio_final=precio_articulo+precio_articulo*20/100
Response.Write precio_final
End Function%>
Un libro de 3500 ptas. se quedará en un precio de <% impuesto(3500) %>
<br>
Una camisa de 6000 ptas. tendrá un precio final de <% impuesto(6000) %>
<br>
Un CD de música de 2000 ptas. costaría <% impuesto(2000) %> ptas.
</BODY>
</HTML>
```

Si quieres ver el efecto que produce pincha aquí

Como puede verse, el script contiene dos partes fundamentales: Una primera en la que definimos la función que llamamos impuesto que depende únicamente de una variable (precio_articulo). Impuesto permite añadir un 20% al precio del artículo e imprimir el resultado en pantalla (Response.Write). En la segunda parte nos servimos de la función para realizar los cálculos necesarios y mostrarlos en pantalla acompañados de texto.

Resulta muy interesante, una vez ejecutado el script, ver el código fuente. Como puede verse, el código HTML que muestra el browser no coincide con el que nosotros hemos escrito. Algo que no debe sorprendernos ya que, como ya hemos explicado, el servidor se encarga de procesarlo y hacerlo comprensible al navegador.

Bucles y condiciones I

La programación exige en muchas ocasiones la repetición de acciones sucesivas o la elección de una determinada secuencia y no de otra dependiendo de las condiciones específicas de la ejecución.

Como ejemplo, podríamos hacer alusión a un script que ejecute una secuencia diferente en función del día de la semana en el que nos encontramos.

Este tipo de acciones pueden ser llevadas a cabo gracias a una paleta de instrucciones presentes en la mayoría de los lenguajes. En este capítulo describiremos someramente algunas de ellas propuestas por el VBS y que resultan de evidente utilidad para el desarrollo de páginas ASP.

Para evitar el complicar el texto, nos limitaremos a introducir las más importantes dejando de lado otras cuantas que podrán ser fácilmente asimilables a partir de ejemplos prácticos. Para una mayor información sobre este tipo de elementos así como sobre los tipos de variables, referirse al manual de VBScript que trata más detenidamente estos aspectos.

Las condiciones: IF

Cuando queremos que el programa, llegado a un cierto punto, tome un camino determinado en determinados casos y otro diferente si las condiciones de ejecución difieren, nos servimos del conjunto de instrucciones If, Then y Else. La estructura de base de este tipo de instrucciones es la siguiente:

```
IF condición THEN
   Instrucción 1
   Instrucción 2
   ...
ELSE
   Instrucción A
   Instrucción B
   ...
END IF
```

Llegados a este punto, el programa verificará el cumplimiento o no de la condición. Si la condición es cierta las instrucciones 1 y 2 serán ejecutadas. De lo contrario (Else), las instrucciones A y B serán llevadas a cabo.

Una vez finalizada la estructura, deberemos cerrar con un End If.

Esta estructura de base puede complicarse un poco más si tenemos cuenta que no necesariamente todo es blanco o negro y que muchas posibilidades pueden darse. Es por ello que otras condiciones pueden plantearse dentro de la condición principal. Hablamos por lo tanto de condiciones anidadas que tendrían una estructura del siguiente tipo:

```
IF condición THEN
   Instrucción 1
   Instrucción 2
   ...
ELSE
   IF condición2 THEN
      Instrucción A
      Instrucción B
      ...
   ELSE
      Instrucción X
      ...
   END IF
END IF
```

De este modo podríamos introducir tantas condiciones como queramos dentro de una condición principal. En este tipo de estructuras es importante cerrar correctamente cada uno de los IF con sus END IF correspondientes. De gran ayuda es la instrucción ELSE IF que permite en una sola línea y sin necesidad de añadir un END IF introducir una condición anidada.

El uso de esta herramienta resultará claro con un poco de práctica. Pongamos un ejemplo sencillo de utilización de condiciones. El siguiente programa permitiría detectar la lengua empleada por el navegador y visualizar un mensaje en dicha lengua.

```
<% @ LANGUAGE="VBSCRIPT" %>
<HTML>
<HEAD>
<TITLE>Detector de Lengua</TITLE>
</HEAD>
<BODY>
<%
'Antes de nada introducimos mensajes en forma de variables
   espanol="Hola"
   ingles="Hello"
   frances="Bonjour"

'Ahora leemos del navegador cuál es su lengua oficial
   idioma=Left(Request.ServerVariables("HTTP_ACCEPT_LANGUAGE"),2)

'Formulamos las posibilidades que se pueden dar
If idioma="es" Then
   Response.Write espanol
ElseIf idioma="fr" Then
   Response.Write frances
Else
   Response.Write ingles
End If %>
</BODY>
</HTML>
```

Para poder ver el funcionamiento de este script es necesario cambiar el idioma preferido lo cual puede ser realizado a partir del menú de opciones del navegador.

Si quieres ver el efecto que produce pincha aquí

Como puede verse, las variables que contienen texto son almacenadas entre comillas.

Para leer la lengua aceptada por el navegador lo que hacemos es definir una variable (idioma) que recoge las dos primeras letras empezando por la izquierda del idioma aceptado por el navegador ("HTTP_ACCEPT_LANGUAGE"). Este idioma aceptado puede ser requerido como una variable del objeto ServerVariables. Por el momento dejaremos esto tal cual, ya nos encargaremos de verlo más detenidamente en otros capítulos.

La tercera parte de script se encarga de ver si el navegador está en español (es), francés (fr) o en cualquier otro idioma que no sea ninguno de estos dos y de imprimir cada uno de los mensajes que proceda en cada caso.

Otro punto a comentar es el hecho de poder comentar los programas. Como puede observarse, dentro del script hemos introducido unos mensajes que nos sirven para leerlo más fácilmente. Estos mensajes no ejercen ninguna influencia en el desarrollo del mismo. Para introducirlos es necesario escribirlos detrás de un apóstrofe: '

Los comentarios son de gran utilidad cuando tratamos con programas muy extensos y complicados los cuales. En estos casos, resultan de gran ayuda a la hora de depurar fallos o introducir modificaciones. Es altamente aconsejable el acostumbrarse a utilizarlos.

Bucles y condiciones II

Los bucles FOR

En muchas ocasiones resulta necesario el ejecutar un conjunto de instrucciones un número definido de veces. Esto puede ser llevado a cabo a partir de la instrucción FOR/NEXT.

La estructura clásica:

```
FOR contador=número inicial to número final STEP incremento
   Instrucción 1
   Instrucción 2
   ...
NEXT
```

A partir de este tipo de estructuras ejecutamos las instrucciones contenidas entre el FOR y el NEXT un cierto número de veces definido por el número inicial, final y el incremento. El incremento resulta de 1 por defecto.

Pongamos un ejemplo:

```
<% @ LANGUAGE="VBSCRIPT" %>
<HTML>
<HEAD>
<TITLE>Bucle for/next</TITLE>
</HEAD>
<BODY>

<%For i=1 to 5%>
<font size=<%Response.Write i%>>Vuelta número <%Response.Write i%></font><br>
<%Next

For i=5 to 1 Step -1%>
<font size=<%Response.Write i%>>Contamos atrás: <%Response.Write i%></font><br>
<%Next%>

</BODY>
</HTML>
```

Este script compuesto de dos bucles cuenta primero de 1 a 5. La variable i toma por lo tanto todos los valores enteros comprendidos entre estos dos números y puede ser utilizada dentro del bucle como lo hacemos en este caso para aumentar el tamaño de la letra. El segundo bucle realiza el proceso inverso (el incremento es negativo) produciendo una disminución del tamaño de la letra.

Si quieres ver el efecto que produce pincha aquí

Lo que puede resultar interesante para ver hasta qué punto el programar páginas dinámicas puede ahorrarnos texto con respecto a la misma página programada en código HTML, es mirar el código fuente de la página a partir del navegador.

Bucles y condiciones III

Los bucles DO WHILE/LOOP

Otra forma de realizar este tipo de secuencias bucle es a partir de la instrucción DO WHILE. En este caso lo que especificamos para fijar la extensión del bucle no es el número de vueltas sino el que se cumpla o no una condición. La estructura de este tipo de bucles es análoga a la de los bucles FOR/NEXT:

```
DO WHILE condición
  Instrucción 1
   Instrucción 2
   ...
LOOP
```

El bucle se dará mientras la condición propuesta siga siendo válida.

Como se verá en ejemplos posteriores, este tipo de bucles resulta muy práctico para la lectura de bases de datos.

Todo este tipo de controladores de flujo (condiciones y bucles) pueden ser matizados y optimizados a partir del uso de operadores lógicos. Así, podemos exigir que sean dos las condiciones que se den para llevar a cabo un conjunto de instrucciones:

IF condición 1 AND condición 2 THEN ...

También podemos requerir que sea una de las dos:

IF condición 1 OR condición 2 THEN...

Del mismo modo, es posible exigir que la condición de un bucle DO sea la inversa a la enunciada:

DO WHILE NOT condición

He aquí, en conclusión, un conjunto de recursos básicos para controlar el desarrollo de programas. Su utilidad resultará más que patente y su uso irá haciéndose intuitivo a medida que nos familiaricemos con el lenguaje.

Los objetos ASP

El ASP es un lenguaje diseñado para la creación de aplicaciones en internet. Esto quiere decir que existen toda una serie de tareas bastante corrientes a las cuales debe dar un tratamiento fácil y eficaz. Nos referimos por ejemplo al envío de e-mails, acceso a archivos, gestión de variables del cliente o servidor como pueden ser su IP o la lengua aceptada...

El lenguaje VB propiamente dicho no da una solución fácil y directa a estas tareas sino que invoca a los denominados objetos que no son más que unos módulos incorporados al lenguaje que permiten el desarrollo de tareas específicas. Estos objetos realizan de una manera sencilla toda una serie de acciones de una complejidad relevante. A partir de una llamada al objeto este realizará la tarea requerida. En cierta forma, estos objetos nos ahorran el tener que hacer largos programas para operaciones sencillas y habituales.

Algunos de estos objetos están incorporados en el propio ASP, otros deben de ser incorporados como si se tratase de componentes accesorios. Por supuesto, no podríamos ejecutar correctamente un script en el cual tuviésemos que llamar a un objeto que no estuviese integrado en el servidor. Este tipo de "plug-in" son generalmente comprados por el servidor a empresas que los desarrollan.

Como todo objeto del mundo real, los objetos del mundo informático tienen sus propiedades que los definen, realizan un cierto numero de funciones o métodos y son capaces de responder de una forma definible ante ciertos eventos.

Dado el nivel de esta obra, la descripción de la totalidad de objetos con sus métodos y propiedades resulta ciertamente fuera de lugar. Nos contentaremos pues con ir describiendo los más frecuentemente utilizados y ejemplificarlos de la manera más práctica dejando la enumeración exhaustiva en forma de apéndice.

Objeto Request I

Bucles y condiciones son muy útiles para procesar los datos dentro de un mismo script. Sin embargo, en un sitio internet, las páginas vistas y los scripts utilizados son numerosos. Muy a menudo necesitamos

que nuestros distintos scripts estén conectados unos con otros y que se sirvan de variables comunes. Por otro lado, el usuario interacciona por medio de formularios cuyos campos han de ser procesados para poder dar una respuesta. Todo este tipo de factores dinámicos han de ser eficazmente regulados por un lenguaje como el ASP.

Como veremos, todo este tipo de aspectos interactivos pueden ser gestionados a partir del objeto Request.

El objeto Request nos devuelve informaciones del usuario que han sido enviadas por medio de formularios, por URL o a partir de cookies (veremos de qué se tratan seguidamente). También nos informa sobre el estado de ciertas variables del sistema como pueden ser la lengua utilizada por el navegador, el número IP del cliente...

Transferir variables por URL

Para pasar las variables de una página a otra lo podemos hacer introduciendo dicha variable en la dirección URL de la página destino dentro del enlace hipertexto. La sintaxis sería la siguiente:

```
<a href="destino.asp?variable1=valor1&variable2=valor2&..."></a>
```

Para recoger la variable en la página destino lo hacemos por medio del objeto Request con el método Querystring:

```
Request.querystring("variable1")
Request.querystring("variable2")
```

Las dos páginas serían así:

```
<HTML>
<HEAD>
<TITLE>Página origen.asp</TITLE>
</HEAD>
<BODY>
<a href="destino.asp?saludo=hola&texto=Esto es una variable texto">Paso variables saludo y texto a la página destino.asp</a>
</BODY>
</HTML>
```

```
<HTML>
<HEAD>
<TITLE>destino.asp</TITLE>
</HEAD>
<BODY>
Variable saludo: <%Response.Write Request.Querystring("saludo")%><br>
Variable texto: <%Response.Write Request.Querystring("texto")%><br>
</BODY>
</HTML>
```

Objeto Request II

Transferir variables por formulario

El proceso es similar al explicado para las URLs. Primeramente, presentamos una primera página con el formulario a rellenar y las variables son recogidas en una segunda página que las procesa:

```
<HTML>
<HEAD>
<TITLE>formulario.asp</TITLE>
</HEAD>
<BODY>
<FORM METHOD="POST" ACTION="destino2.asp">
Nombre<br>
<INPUT TYPE="TEXT" NAME="nombre"><br>
Apellidos<br>
```

```
<INPUT TYPE="TEXT" NAME="apellidos"><br>
<INPUT TYPE="SUBMIT">
</FORM>
</BODY>
</HTML>


<HTML>
<HEAD>
<TITLE>destino2.asp</TITLE>
</HEAD>
<BODY>
Variable nombre: <%=Request.Form("nombre")%><br>
Variable apellidos: <%=Request.Form("apellidos")%>
</BODY>
</HTML>
```

Otras utilidades de Request: las ServerVariables

El objeto Request nos da acceso a otras informaciones relativas al cliente y el servidor las cuales pueden resultar de una gran utilidad. Estas informaciones son almacenadas como variables las cuales son agrupadas en una colección llamada ServerVariables.

Dentro de esta colección tenemos variables tan interesantes como:

HTTP_ACCEPT_LANGUAGE	Nos informa de la lengua preferida por el navegador
HTTP_USER_AGENT	Indica cuál es el navegador utilizado.
PATH_TRANSLATED	Nos devuelve el path físico del disco duro del servidor en el que se encuentra nuestro script
SERVER_SOFTWARE	Nos dice qué tipo de software utiliza el servidor

Para visualizar en pantalla alguna de estas variables, debemos escribir algo como:

Response.write request.servervariables("nombre de la variable")

Una forma rápida de visualizar todas estas variables es a partir de un script con esta secuencia:

```
<%
For Each elemento in Request.ServerVariables
   Response.Write elemento&" : "&Request.ServerVariables(elemento)& "<br>"
Next
%>
```

Esto nos daría por un lado el nombre de la variable y del otro su valor. Este tipo de bucle For Each/Next se parece a otros ya vistos. En este caso, el bucle se realiza tantas veces como elementos tiene la colección (ServerVariables) que no es más que el conjunto de elementos comprendidos en la extensión del objeto (Request). Este tipo de bucle es aplicable a otras colecciones de éste y otros objetos como por ejemplo los Request.Form o Request.Querystring o las cookies. De esta forma seríamos capaces de visualizar el nombre y contenido de tales colecciones sin necesidad de enunciarlas una por una.

Objeto Response

Tal y como se ha visto, el objeto Request gestiona todo lo relativo a entrada de datos al script por parte del usuario (formularios), provenientes de otra URL, del propio servidor o del browser.

Nos queda pues por explicar cómo el script puede "responder" a estos estímulos, es decir, cómo, después de procesar los debidamente los datos, podemos imprimir estos en pantalla, inscribirlos en las cookies o enviar al internauta a una u otra pagina. En definitiva, queda por definir la forma en la que ASP regula el contenido que es enviado al navegador.

Esta función es tomada en cargo por el objeto Response el cual ha sido ligeramente mencionado en capítulos precedentes concretamente en asociación al método Write.

En efecto, sentencias del tipo:

<%Response.Write "Una cadena de texto"%>

o bien,

<%Response.Write variable%>

Tienen como cometido imprimir en el documento HTML generado un mensaje o valor de variable. Este método es tan comúnmente utilizado que existe una abreviación del mismo de manera a facilitar su escritura:

<% = variable %> es análogo a <%response.write variable%>

Es importante precisar el hecho que imprimir en el documento HTML no significa necesariamente visualizar en pantalla ya que podríamos servirnos de estas etiquetas para crear determinadas etiquetas HTML. He aquí un ejemplo de lo que se pretende decir:

<% path="http://www.misitio.com/graficos/imagen.gif" %>

<img src="<% =path %>">

Este fragmento de script nos generaría un código HTML que sería recibido en el navegador del siguiente modo:

<img src="http://www.misitio.com/graficos/imagen.gif">

Otro elemento interesante de este objeto Response es el método Redirect. Este método nos permite el enviar automáticamente al internauta a una página que nosotros decidamos.

Esta función, a todas luces práctica, puede ser empleada en scripts en los que enviamos al visitante de nuestro sitio a una u otra página en función del navegador que utiliza o la lengua que prefiere. También es muy útil para realizar páginas "escondidas" que realizan una determinada función sin mostrar ni texto ni imágenes y nos envían a continuación a otra página que es en realidad la que nosotros recibimos en el navegador.

Aquí se puede ver una secuencia de script que nos permitiría usar este método para el caso en el que tengamos páginas diferentes para distintas versiones de navegadores. En ella emplearemos un objeto que nos debe parecer familiar ya que lo acabamos de ver (Request.ServerVariables):

```
<%
tipo_navegador = Request.ServerVariables("HTTP_USER_AGENT")
If Instr(1, tipo_navegador, "MSIE 3", 1) <> 0 then
Response.Redirect "MSIE3.asp"
ElseIf Instr(1, tipo_navegador, "MSIE 4", 1) <> 0 then
Response.Redirect "MSIE4.asp"
ElseIf Instr(1, tipo_navegador, "MSIE 5", 1) <> 0 then
Response.Redirect "MSIE5.asp"
ElseIf Instr(1, tipo_navegador, "Mozilla/4", 1) <> 0 then
Response.Redirect "Netscape4.asp"
Else
Response.Redirect "cualquiera.asp"
%>
```

Por supuesto, acompañando este script, debemos tener los correspondientes archivos MSIE3.asp, MSIE4.asp, MSIE5.asp... Cada uno de ellos con las particularidades necesarias para la buena visualización de nuestra página en tales navegadores.

Las famosas cookies

Sin duda este término resultara familiar para muchos. Algunos lo habrán leído u oído pero no saben de qué se trata. Otros sin embargo sabrán que las cookies son unas informaciones almacenadas por un sitio web en el disco duro del usuario. Esta información es almacenada en un archivo tipo texto que se guarda cuando el navegador accede al sitio web.

Es posible, por supuesto, ver estos archivos. Para abrirlos hay que ir al directorio C:\Windows\Cookies para los usuarios de IE 4+ o a C:\...\Netscape\Users\defaultuser para usuarios de Netscape. Como podréis comprobar, en la mayoría de los casos la información que se puede obtener es indescifrable.

La utilidad principal de las cookies es la de poder identificar al navegador una vez éste visita el sitio por segunda vez y así, en función del perfil del cliente dado en su primera visita, el sitio puede adaptarse dinámicamente a sus preferencias (lengua utilizada, colores de pantalla, formularios rellenados total o parcialmente, redirección a determinadas páginas...).

Para crear un archivo cookies, modificar o generar una nueva cookie lo podemos hacer a partir del objeto Response con una sintaxis como la siguiente:

Response.Cookies("nombre de la cookie") = valor de la cookie

El valor de la cookie puede ser una variable, un numero o una cadena delimitada por comillas.

Es importante saber que las cookies tienen una duración igual a la sesión, es decir, a menos que lo especifiquemos, el archivo texto generado se borrará desde el momento en que el usuario haya abandonado el sitio por un tiempo prolongado (generalmente unos 20 minutos) o que el navegador haya sido cerrado. Podemos arreglar este supuesto inconveniente mediante la propiedad Expires:

Response.Cookies("nombre de la cookie").expires = #01/01/2002#

Esto nos permite decidir cual es la fecha de caducidad de la cookie. Hay que tener en cuenta que esto no es más que hipotético ya que el usuario es libre de borrar el archivo texto cuando le plazca.

Por otra parte, es interesante señalar que el hecho de que definir una cookie ya existente implica el borrado de la antigua. Del mismo modo, el crear una primera cookie conlleva la generación automática del archivo texto.

Para leer las cookies, nada más fácil que usando el objeto Request de la siguiente forma:

variable = Request.Cookies("nombre de la cookie")

Si por ejemplo quisiéramos recuperar del archivo txt la cookie correspondiente a la lengua del cliente y almacenarlo en nuestro script para futuros usos, podríamos escribir:

lengua=Request.Cookies("lengua")

Las cookies son una herramienta fantástica para personalizar nuestra página pero hay que ser cautos ya que, por una parte, no todos los navegadores las aceptan y por otra, se puede deliberadamente impedir al navegador la creación de cookies. Es por ello que resultan un complemento y no una fuente de variables infalible para nuestro sitio.

Objeto Session

En los programas que hemos visto hasta ahora, hemos utilizado variables que solo existían en el archivo que era ejecutado. Cuando cargábamos otra página distinta, los valores de estas variables se perdían a menos que nos tomásemos la molestia de pasarlos por la URL o inscribirlos en las cookies o en un formulario para su posterior explotación. Estos métodos, aunque útiles, no son todo lo prácticos que podrían en determinados casos en los que la variable que queremos conservar ha de ser utilizada en varios scripts diferentes y distantes los unos de los otros.

Podríamos pensar que ese problema puede quedar resuelto con las cookies ya que se trata de variables que pueden ser invocadas en cualquier momento. El problema, ya lo hemos dicho, es que las cookies no son aceptadas ni por la totalidad de los usuarios ni por la totalidad de los navegadores lo cual implica que una aplicación que se sirviera de las cookies para pasar variables de un archivo a otro no sería 100% infalible.

Nos resulta pues necesario el poder declarar ciertas variables que puedan ser reutilizadas tantas veces como queramos dentro de una misma sesión. Imaginemos un sitio multilingüe en el que cada vez que queremos imprimir un mensaje en cualquier página necesitamos saber en qué idioma debe hacerse. Podríamos introducir un script identificador de la lengua del navegador en cada uno de los archivos o bien declarar una variable que fuese valida para toda la sesión y que tuviese como valor el idioma reconocido en un primer momento.

Estas variables que son válidas durante una sesión y que luego son "olvidadas" son definidas con el objeto Session de la siguiente forma:

Session("nombre de la variable") = valor de la variable

Una vez definida, la variable Session, será almacenada en memoria y podrá ser empleada en cualquier script del sitio web.

La duración de una sesión viene definida por defecto en 20 minutos. Esto quiere decir que si en 20 minutos no realizamos ninguna acción, el servidor dará por finalizada la sesión y todas las variables Session serán abandonadas. Esta duración puede ser modificada con la propiedad Timeout:

Session.Timeout = n° de minutos que queramos que dure

Una forma de borrar las variables Session sin necesidad de esperara que pase este plazo es a partir del método Abandon:

Session.Abandon

De este modo todas las variables Session serán borradas y la sesión se dará por finalizada. Este método puede resultar practico cuando estemos haciendo pruebas con el script y necesitemos reinicializar las variables.

Lo que se suele hacer es crear un archivo en el que se borran las cookies y se abandona la sesión. Este archivo será ejecutado cuando queramos hacer borrón y cuenta nueva:

```
<% @ LANGUAGE="VBSCRIPT" %>
<HTML>
<HEAD>
<TITLE>Puesta a cero</TITLE>
</HEAD>
<BODY>
<%
For Each galleta in Response.Cookies
  Galleta=""
Next
Session.Abandon
%>
Borrón y cuenta nueva!!<br>
<a href="url de la página de inicio">Volver al principio</a>
</BODY>
</HTML>
```

Trabajar con bases de datos en ASP

Una de las principales ventajas que presenta el trabajar con páginas dinámicas es el poder almacenar los contenidos en bases de datos. De esta forma, podemos organizarlos, actualizarlos y buscarlos de una manera mucho más simple.

ASP nos ofrece una forma muy eficaz de interaccionar con estas bases de datos gracias al uso del componente ADO (ActiveX Data Objects) el cual permite acceder a dichas bases de una forma sencilla.

Este ADO no es más que un conjunto de objetos que, utilizados en conjunto, nos permiten explotar de una forma muy versátil las bases de datos de nuestra aplicación. No entraremos por el momento en consideraciones teóricas al respecto.

Por otra parte, lo scripts ASP deben establecer un dialogo con la base de datos. Este dialogo se lleva a cabo a partir de un idioma universal: el SQL (Structured Query Language) el cual es común a todas las bases de datos. Este lenguaje resulta, como veremos en el manual de SQL, muy potente y fácil de aprender.

En este manual de ASP nos limitaremos a utilizar las instrucciones básicas que serán aprendidas a medida que explicamos las diferentes formas de actuar sobre una base de datos a partir de páginas ASP.

La base de datos que ha sido utilizada en estos ejemplos es MS Access. No es por supuesto la única si bien es la más corriente en pequeños PCs y resulta absolutamente operativa siempre que las tablas no sean astronómicamente grandes. Esperamos poder ofreceros próximamente también un pequeño curso de Access en el que explicar los principios rudimentarios necesarios para poder servirnos de él. No obstante, esta aplicación resulta suficientemente fácil e intuitiva como para poder prescindir de dicho curso por el momento.

Así pues, para sumergirnos en estos capítulos siguientes, hemos de cumplir los siguientes requisitos técnicos:

-Instalar el PWS
-Enlazar la base de datos con ODBC
-Instalar el MS Access (viene dentro del pack Office)

-Disponer de un browser y un editor (MS IE y Home Site son nuestras humildes recomendaciones)

Selecciones en una tabla

Dentro de una base de datos, organizada por tablas, la selección de una tabla entera o de un cierto numero de registros resulta una operación rutinaria.

A partir de esta selección se puede posteriormente efectuar toda una serie de cambios o bien realizar una simple lectura.

El siguiente script nos permite realizar la lectura de la tabla clientes contenida en nuestra base de datos. A primera vista todo nos puede parecer un poco complejo, pero nada más lejos de la realidad.

```
<HTML>
<HEAD>
<TITLE>Lectura de registros de una tabla</TITLE>
</HEAD>
<BODY>
<h1><div align="center">Lectura de la tabla</div></h1>
<br>
<br>
<%
'Antes de nada hay que instanciar el objeto Connection
Set Conn = Server.CreateObject("ADODB.Connection")

'Una vez instanciado Connection lo podemos abrir y le asignamos la base de datos donde vamos a
efectuar las operaciones
Conn.Open "Mibase"

'Ahora creamos la sentencia SQL que nos servira para hablar a la BD
sSQL="Select * From Clientes Order By nombre"

'Ejecutamos la orden
set RS = Conn.Execute(sSQL)

'Mostramos los registros%>
<table align="center">
<tr>
<th>Nombre</th>
<th>Teléfono</th>
</tr>
<%
Do While Not RS.Eof
%>
<tr>
<td><%=RS("nombre")%></td>
<td><%=RS("telefono")%></td>
</tr>
<%
RS.MoveNext
Loop

'Cerramos el sistema de conexion
Conn.Close
%>

</table>

<div align="center">
<a href="insertar.html">Añadir un nuevo registro</a><br>
<a href="actualizar1.asp">Actualizar un registro existente</a><br>
<a href="borrar1.asp">Borrar un registro</a><br>
</div>

</BODY>
</HTML>
```

Antes de nada, si lo que deseamos es interrogar una base de datos, lo primero que hay que hacer es

obviamente establecer la conexión con ella. Esto se hace a partir del objeto Connection el cual es "invocado" o más técnicamente dicho instanciado por medio de la primera instrucción en la cual el objeto toma el nombre arbitrario de la variable Conn .

El paso siguiente es abrir el objeto y asignarle la base de datos con la que debe contactar. En este caso, la base la hemos llamado Mibase. Este debe de ser el mismo nombre con el que la hemos bautizado cuando hemos configurado los conectores ODCB, además, este nombre no tiene por qué coincidir necesariamente con el nombre del archivo.

Una vez creada la conexión a nuestra base de datos, el paso siguiente es hacer nuestra petición. Esta petición puede ser formulada primeramente y almacenada en una variable (sSQL) para, a continuación, ser ejecutada por medio de la instrucción siguiente.

La petición que hemos realizado en este caso es la de seleccionar todos los campos que hay en la tabla clientes (* es un comodín) y ordenar los resultados por orden alfabético con respecto al campo nombre. Podemos ver más detalladamente este tipo de instrucciones SQL en nuestro manual de SQL.

El resultado de nuestra selección es almacenado en la variable RS en forma de tabla. Para ver la tabla lo que hay que hacer ahora es "pasearse" por esta tabla "virtual" RS la cual posee una especie de cursor que, a menos que se especifique otra cosa, apunta al primer registro de la selección. El objetivo ahora es hacer desplazarse al cursor a lo largo de la tabla para poder leerla en su totalidad. La forma de hacerlo es a partir de un bucle Do While el cual ya ha sido explicado anteriormente y que lo único que hace es ejecutar las instrucciones comprendidas entre el Do y el Loop siempre que la condición propuesta (Not RS.Eof) sea verdadera. Esto se traduce como "Ejecutar este conjunto de instrucciones mientras que la tabla de resultados (RS) no llegue al final (Eof, End of File).

Las instrucciones incluidas en el bucle son, por un lado, la impresión en el documento de los valores de determinados campos (=RS("nombre del campo")) y por otro, saltar de un registro al otro mediante la instrucción RS.MoveNext.

Todo este conjunto de instrucciones ASP viene en combinación con un código HTML que permite su visualización en forma de tabla. Además, se han incluido unos enlaces que apuntan hacia otra serie de scripts que veremos más adelante y que formaran en conjunto una aplicación.

Es interesante ver el código fuente resultante de este script. En este caso el código que ve el cliente resulta sensiblemente más sencillo.

Creación de un nuevo registro

En este caso lo que buscamos es crear, a partir de los datos recibidos de un formulario, un nuevo registro en nuestra tabla clientes. Tendremos pues dos archivos diferentes, uno que podría ser un HTML puro en el que introducimos el formulario a rellenar y que nos envía al segundo, un script muy parecido al previamente visto para realizar una selección. He aquí los dos scripts:

```
<HTML>
<HEAD>
<TITLE>Insertar.html</TITLE>
</HEAD>
<BODY>
<div align="center">
<h1>Insertar un registro</h1>
<br>
<FORM METHOD="POST" ACTION="insertar.asp">
Nombre<br>
<INPUT TYPE="TEXT" NAME="nombre"><br>
Teléfono<br>
<INPUT TYPE="TEXT" NAME="telefono"><br>
<INPUT TYPE="SUBMIT" value="Insertar">
</FORM>
</div>
</BODY>
</HTML>
```

```
<HTML>
<HEAD>
<TITLE>Insertar.asp</TITLE>
</HEAD>
```

```
<BODY>

<%
'Recogemos los valores del formulario
nombre=Request.Form("nombre")
telefono= Request.Form("telefono")

'Instanciamos y abrimos nuestro objeto conexion
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "Mibase"

'Ahora creamos la sentencia SQL
sSQL="Insert Into Clientes (nombre,telefono) values ('" & nombre & "','" & telefono & "')"

'Ejecutamos la orden
set RS = Conn.Execute(sSQL)
%>

<h1><div align="center">Registro Insertado</div></h1>
<div align="center"><a href="lectura.asp">Visualizar el contenido de la base</a></div>

<%
'Cerramos el sistema de conexion
Conn.Close
%>

</BODY>
</HTML>
```

Como puede verse, la forma de operar es idéntica a la vista anteriormente para el display de una tabla. En este caso hemos introducido un enlace a este primer script de lectura para ver cómo los cambios se han hecho efectivos.

La construcción de la sentencia SQL se hace por fusión de los distintos elementos constitutivos. La forma de fusionarlos mediante el símbolo **&**. Todo lo que sea texto tiene que ir entre comillas. Sería interesante introducir una línea suplementaria en vuestro código para imprimir la sSQL formada. La línea sería del siguiente tipo:

Response.Write sSQL

Esta línea iría situada evidentemente después de haber construido la sentencia.

Actualización de un registro existente

Para mostrar cómo se actualiza un registro presente en nuestra base de datos, vamos a hacerlo a partir de un caso un poco más complejo para que empecemos a familiarizarnos con estas operaciones. Realizaremos un par de scripts que permitan cambiar el numero de teléfono de las distintas personas presentes en nuestra base. El nombre de estas personas, así como el nuevo numero de teléfono, serán recogidos por medio de un formulario.

El archivo del formulario va a ser esta vez un script ASP en el que efectuaremos una llamada a nuestra base de datos para construir un menú desplegable donde aparezcan todos los nombres. La cosa quedaría así:

```
<HTML>
<HEAD>
<TITLE>Actualizar1.asp</TITLE>
</HEAD>
<BODY>
<div align="center">
<h1>Actualizar un registro</h1>
<br>

<%
'Instanciamos y abrimos nuestro objeto conexion
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "Mibase"
```

```
%>

<FORM METHOD="POST" ACTION="actualizar2.asp">
Nombre<br>
<%
'Creamos la sentencia SQL y la ejecutamos
sSQL="Select nombre From clientes Order By nombre"
set RS = Conn.Execute(sSQL)
%>
<select name="nombre">
<%
'Generamos el menu desplegable
Do While not RS.eof%>
   <option><%=RS("nombre")%>
   <%RS.movenext
Loop
%>
</select>
<br>
Teléfono<br>
<INPUT TYPE="TEXT" NAME="telefono"><br>
<INPUT TYPE="SUBMIT" value="Actualizar">
</FORM>
</div>

</BODY>
</HTML>
```

La manera de operar para construir el menú desplegable es la misma que para visualizar la tabla. De nuevo empleamos un bucle Do While que nos permite mostrar cada una de las opciones.

El script de actualización será muy parecido al de inserción:

```
<TITLE>Actualizar2.asp</TITLE>
</HEAD>
<BODY>

<%
'Recogemos los valores del formulario
nombre=Request.Form("nombre")
telefono= Request.Form("telefono")

'Instanciamos y abrimos nuestro objeto conexion
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "Mibase"

'Ahora creamos la sentencia SQL
sSQL="Update Clientes Set telefono='" & telefono & "' Where nombre='" & nombre & "'"

'Ejecutamos la orden
set RS = Conn.Execute(sSQL)
%>

<h1><div align="center">Registro Actualizado</div></h1>
<div align="center"><a href="lectura.asp">Visualizar el contenido de la base</a></div>

<%
'Cerramos el sistema de conexion
Conn.Close
%>

</BODY>
</HTML>
```

Nada que comentar al respecto salvo la estructura de la sentencia SQL que en este caso realiza un

Update en lugar de un Insert. Aconsejamos, como para el caso precedente imprimir el valor de sSQL de manera a ver cómo queda la sentencia una vez construida.

Borrado de un registro

Otra de las operaciones elementales que se pueden realizar sobre una base de datos es el borrar un registro. Para hacerlo, SQL nos propone sentencias del tipo Delete. Veámoslo con un ejemplo aplicado a nuestra agenda. Primero, crearemos un menú desplegable dinámico como para el caso de las actualizaciones:

```
<HTML>
<HEAD>
<TITLE>Borrar1.asp</TITLE>
</HEAD>
<BODY>
<div align="center">
<h1>Borrar un registro</h1>
<br>
<%
'Instanciamos y abrimos nuestro objeto conexion
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "Mibase"
%>

<FORM METHOD="POST" ACTION="borrar2.asp">
Nombre<br>
<%
'Creamos la sentencia SQL y la ejecutamos
sSQL="Select nombre From clientes Order By nombre"
set RS = conn.execute(sSQL)
%>
<select name="nombre">
<%
'Generamos el menu desplegable
Do While not RS.eof%>
   <option><%=RS("nombre")%>
   <%RS.movenext
Loop
%>
</select>
<br>
<INPUT TYPE="SUBMIT" value="Borrar">
</FORM>
</div>

</BODY>
</HTML>
```

El siguiente paso es hacer efectiva la operación a partir de la ejecución de la sentencia SQL que construimos a partir de los datos del formulario:

```
<HTML>
<HEAD>
<TITLE>Borrar2.asp</TITLE>
</HEAD>
<BODY>
<%
'Recogemos los valores del formulario
nombre=Request.Form("nombre")

'Instanciamos y abrimos nuestro objeto conexion
Set Conn = Server.CreateObject("ADODB.Connection")
Conn.Open "Mibase"

'Ahora creamos la sentencia SQL
sSQL="Delete From Clientes Where nombre='" & nombre & "'"

'Ejecutamos la orden
```

```
set RS = Conn.Execute(sSQL)
%>

<h1><div align="center">Registro Borrado</div></h1>
<div align="center"><a href="lectura.asp">Visualizar el contenido de la base</a></div>

<%
'Cerramos el sistema de conexion
Conn.Close
%>

</BODY>
</HTML>
```

Capitulo 10
Programación en PHP

Principios básicos para la programación en PHP, el popular lenguaje del lado del servidor. Manual asequible para no programadores que sienta los fundamentos básicos de este lenguaje. Continuación lógica del manual de páginas dinámicas.

Introducción a la programación en PHP

PHP es uno de los lenguajes de lado servidor más extendidos en la web. Nacido en 1994, se trata de un lenguaje de creación relativamente creciente que ha tenido una gran aceptación en la comunidad de webmasters debido sobre todo a la potencia y simplicidad que lo caracterizan.

PHP nos permite embeber su pequeños fragmentos de código dentro de la página HTML y realizar determinadas acciones de una forma fácil y eficaz sin tener que generar programas programados íntegramente en un lenguaje distinto al HTML. Por otra parte, y es aquí donde reside su mayor interés con respecto a los lenguajes pensados para los CGI, PHP ofrece un sinfín de funciones para la explotación de bases de datos de una manera llana, sin complicaciones.

Podríamos efectuar la quizás odiosa comparación de decir que PHP y ASP son lenguajes parecidos en cuanto a potencia y dificultad si bien su sintaxis puede diferir sensiblemente. Algunas diferencias principales pueden, no obstante, mencionarse:

-PHP, aunque multiplataforma, ha sido concebido inicialmente para entornos UNIX y es en este sistema operativo donde se pueden aprovechar mejor sus prestaciones. ASP, siendo una tecnología Microsoft, esta orientado hacia sistemas Windows, especialmente NT.

-Las tareas fundamentales que puede realizar directamente el lenguaje son definidas en PHP como funciones mientras que ASP invoca más frecuentemente los objetos. Por supuesto, esto no es más que una simple cuestión de forma ya que ambos lenguajes soportan igualmente ambos procedimientos.

-ASP realiza numerosas tareas sirviéndose de componentes (objetos) que deben ser comprados (o programados) por el servidor a determinadas empresas especializadas. PHP presenta una filosofía totalmente diferente y, con un espíritu más generoso, es progresivamente construido por colaboradores desinteresados que implementan nuevas funciones en nuevas versiones del lenguaje.

Este manual va destinado a aquellos que quieren comenzar de cero el aprendizaje de este lenguaje y que buscan en él la aplicación directa a su proyecto de sitio o a la mejora de su sitio HTML. Los capítulos son extremadamente simples, sino simplistas, buscando ser accesibles a la mayoría. Ellos pueden ser complementados posteriormente con otros **artículos de mayor nivel** destinados a gente más experimentada.

La forma en la que hemos redactado este manual lo hace accesible a cualquier persona no familiarizada con la programación. Sin embargo, es posible que en determinados momentos alguien que no haya programado nunca pueda verse un poco desorientado. Nuestro consejo es el de no querer entender todo antes de pasar al siguiente capítulo sino intentar asimilar algunos conceptos y volver atrás en cuanto una duda surja o hayamos olvidado algún detalle. Nunca viene mal leer varias veces lo mismo hasta que quede bien grabado y asimilado.

Antes de comenzar a leer este manual es altamente aconsejable, sino imprescindible, haber leído previamente el manual sobre **manual sobre páginas dinámicas** en el cual se explica a grandes rasgos qué es el PHP, algunos conceptos útiles sobre el modo de trabajar con páginas dinámicas al mismo tiempo que nos introduce algunos elementos básicos de la programación como pueden ser las variables y las funciones.

Otra referencia a la cual haremos alusión es el **tutorial de SQL** que nos será de gran ayuda para el tratamiento de bases de datos.

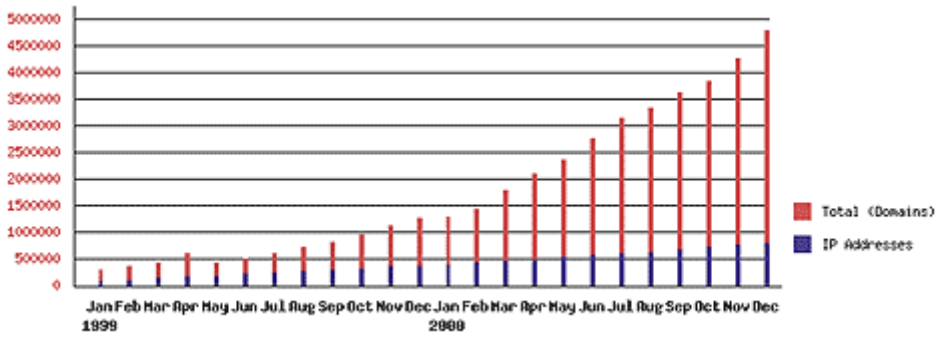
Esperamos que este manual resulte de vuestro agrado y que corresponda a nuestras expectativas: El poder acercar PHP a todos aquellos amantes del desarrollo de webs que quieren dar el paso hacia las webs "profesionales".

Breve historia de PHP

PHP es un lenguaje creado por una gran comunidad de personas. El sistema fue desarrollado originalmente en el año 1994 por Rasmus Lerdorf como un CGI escrito en C que permitía la

interpretación de un número limitado de comandos. El sistema fue denominado Personal Home Page Tools y adquirió relativo éxito gracias a que otras personas pidieron a Rasmus que les permitiese utilizar sus programas en sus propias páginas. Dada la aceptación del primer PHP y de manera adicional, su creador diseñó un sistema para procesar formularios al que le atribuyó el nombre de FI (Form Interpreter) y el conjunto de estas dos herramientas, sería la primera versión compacta del lenguaje: PHP/FI.

La siguiente gran contribución al lenguaje se realizó a mediados del 97 cuando se volvió a programar el analizador sintáctico, se incluyeron nuevas funcionalidades como el soporte a nuevos protocolos de Internet y el soporte a la gran mayoría de las bases de datos comerciales. Todas estas mejoras sentaron las bases de PHP versión 3. Actualmente PHP se encuentra en su versión 4, que utiliza el motor Zend, desarrollado con mayor meditación para cubrir las necesidades actuales y solucionar algunos inconvenientes de la anterior versión. Algunas mejoras de esta nueva versión son su rapidez -gracias a que primero se compila y luego se ejecuta, mientras que antes se ejecutaba mientras se interpretaba el código-, su mayor independencia del servidor web -creando versiones de PHP nativas para más plataformas- y un API más elaborado y con más funciones.



Gráfica del número de dominios y direcciones IP que utilizan PHP.
Estadística de Netcraft.

En el último año, el número de servidores que utilizan PHP se ha disparado, logrando situarse cerca de los 5 millones de sitios y 800.000 direcciones IP, lo que le ha convertido a PHP en una tecnología popular. Esto es debido, entre otras razones, a que PHP es el complemento ideal para que el tándem Linux-Apache sea compatible con la programación del lado del servidor de sitios web. Gracias a la aceptación que ha logrado, y los grandes esfuerzos realizados por una creciente comunidad de colaboradores para implementarlo de la manera más óptima, podemos asegurar que el lenguaje se convertirá en un estándar que compartirá los éxitos augurados al conjunto de sistemas desarrollados en código abierto.

Tareas principales del PHP

Poco a poco el PHP se va convirtiendo en un lenguaje que nos permite hacer de todo. En un principio diseñado para realizar poco más que un contador y un libro de visitas, PHP ha experimentado en poco tiempo una verdadera revolución y, a partir de sus funciones, en estos momentos se pueden realizar una multitud de tareas útiles para el desarrollo del web:

Funciones de correo electrónico
Podemos con una facilidad asombrosa enviar un e-mail a una persona o lista parametrizando toda una serie de aspectos tales como el e-mail de procedencia, asunto, persona a responder...

Otras funciones menos frecuentes pero de indudable utilidad para gestionar correos electrónicos son incluidas en su librería.

Gestión de bases de datos
Resulta difícil concebir un sitio actual, potente y rico en contenido que no es gestionado por una base de datos. El lenguaje PHP ofrece interfaces para el acceso a la mayoría de las bases de datos comerciales y por ODBC a todas las bases de datos posibles en sistemas Microsoft, a partir de las cuales podremos editar el contenido de nuestro sitio con absoluta sencillez.

Gestión de archivos
Crear, borrar, mover, modificar...cualquier tipo de operación más o menos razonable que se nos pueda ocurrir puede ser realizada a partir de una amplia librería de funciones para la gestión de archivos por PHP. También podemos transferir archivos por FTP a partir de sentencias en nuestro código, protocolo para el cual PHP ha previsto también gran cantidad de funciones.

Tratamiento de imágenes
Evidentemente resulta mucho más sencillo utilizar Photoshop para una el tratamiento de imágenes

pero...¿Y si tenemos que tratar miles de imágenes enviadas por nuestros internautas?

La verdad es que puede resultar muy tedioso uniformar en tamaño y formato miles de imágenes recibidas día tras día. Todo esto puede ser también automatizado eficazmente mediante PHP.

También puede parecer útil el crear botones dinámicos, es decir, botones en los que utilizamos el mismo diseño y solo cambiamos el texto. Podremos por ejemplo crear un botón haciendo una única llamada a una función en la que introducimos el estilo del botón y el texto a introducir obteniendo automáticamente el botón deseado.

A partir de la librería de funciones graficas podemos hacer esto y mucho más.

Muchas otras funciones pensadas **para Internet** (tratamiento de cookies, accesos restringidos, comercio electrónico...) o para **propósito general** (funciones matemáticas, explotación de cadenas, de fechas, corrección ortográfica, compresión de archivos...) son realizadas por este lenguaje. A esta inmensa librería cabe ahora añadir todas las funciones personales que uno va creando por necesidades propias y que luego son reutilizadas en otros sitios y todas aquellas intercambiadas u obtenidas en foros o sitios especializados.

Como puede verse, las posibilidades que se nos presentan son sorprendentemente vastas. Lo único que se necesita es un poco de ganas de aprender y algo de paciencia en nuestros primeros pasos. El resultado puede ser muy satisfactorio.

Instalación de PHP en nuestro servidor

Como todo lenguaje de lado servidor, PHP, requiere de la instalación de un servidor en nuestro PC para poder trabajar en local. Este modo de trabajo resulta a todas luces más práctico que colgar los archivos por FTP en el servidor y ejecutarlos desde Internet.

Así pues, antes comenzar a crear nuestros programas en PHP, es necesario:

- Convertir nuestro ordenador en un servidor. Esto se hace instalando uno de los varios servidores disponibles para el sistema operativo de nuestra máquina.

- Introducir en nuestro servidor los archivos que le permitirán la comprensión del PHP. Estos archivos pueden ser descargados, en su versión más actual, de la **página oficial de PHP**.

Para conocer la forma de instalar PHP sobre cada servidor de cada sistema operativo podemos dirigirnos al apartado de **documentación de la página oficial de PHP** donde disponemos de un manual en HTML de rápida consulta y un enorme manual en PDF de casi 1000 páginas traducido al castellano donde explican minuciosamente y entre otras cosas, los pasos a seguir para cada caso particular. De todos modos, nosotros vamos a ofrecer algunas ayudas para configurar PHP en los sistemas más habituales.

La elección de vuestro programa servidor tendrá mucho que ver con el sistema operativo que tengáis corriendo en vuestro ordenador. Estas serían algunas posibilidades de sistemas opertativos y soluciones que funcionan bien.

Windows 95/98

Si estáis trabajando en Windows 95 o Windows 98 y para desarrolladores principiantes, podría ser recomendable utilizar el servidor Personal Web Ser. En este caso necesitaríais:

- Personal Web Server de Microsoft como servidor el cual os sirve además para el aprendizaje en ASP. Tenéis una **guía de instalación y configuración** en esta misma web.

- Una instalación autoextraible de la versión más reciente de PHP que, además de tardar menos en descargarse, os guiará paso a paso en el proceso de instalación. Esta versión no incluye todas las funcionalidades de PHP, pero os servirá para aprender hasta un buen nivel.

Hay que señalar que, para el caso de PHP en PWS, además de todo lo dicho en capítulo de instalación, es importante al crear el directorio virtual permitir la ejecución de scripts validando la caja correspondiente.

En Windows 95/98 también podremos utilizar el servidor Apache y puede que sea una opción todavía más completa que la de utilizar PWS. A continuación explicamos más sobre ello.

Windows ME y XP Home edition

No hemos probado PHP en estas plataformas, pero en principio no tienen compatibilidad con Personal Web Server, por lo que deberíamos decantarnos por otro servidor.

Otra posibilidad para los usuarios de Windows en general es instalar **Apache** como servidor web lo cual puede resultar ventajoso con respecto al uso del PWS ya que PHP está principalmente diseñado para correr en este servidor. Esto quiere decir que, aunque en principio todo debería funcionar correctamente sobre ambos servidores, es posible que algún bug no corregido haga fallar uno de nuestros scripts si trabajamos para con un servidor cuyas actualizaciones son menos frecuentes y detalladas.

Apache ha sido especialmente pensado para plataformas Unix-Linux, aunque recientemente, con la Apache 2.0, han desarrollado una versión específica para Windows.

Disponemos de un artículo para aprender a configurar PHP sobre Apache en Windows.

Windows NT, Windows 2000 y XP en sus versiones Profesional y Server

Para estos sistemas tenemos dos posibilidades muy interesantes, ya que podremos instalar PHP sobre Internet Information Server o sobre Apache con todas las garantías. Si hubiese que recomendar una de las dos opciones, nos decantaríamos por Apache debido a que, como decíamos, PHP está pensado para trabajar sobre Apache. Podría ser interesante IIS en el caso de que deseemos correr ASP y PHP sobre el mismo servidor, ya que, en principio, Apache no es compatible con ASP.

Unix - Linux

Hay que decir, no obstante, que las mejores prestaciones de este lenguaje son obtenidas trabajando en entorno Unix o Linux y con un servidor Apache, la combinación más corriente en la mayoría de los servidores de Internet que trabajan con PHP.

Conclusión

En cualquier caso, para fines de desarrollo en local, podemos contentarnos en un principio de trabajar con cualquier sistema. Solamente en casos de programación realmente avanzada podremos confrontarnos con problemas relacionados con el sistema operativo utilizado o el servidor en el que hacemos correr nuestras páginas. Hay que pensar también que, en casos puntuales para los que nuestro PC pueda quedarse corto, podemos hacer directamente nuestras pruebas en el servidor donde alojamos nuestro sitio el cual será muy probablemente, como hemos dicho, un Unix o Linux funcionando con Apache.

Configuración de PHP con Apache en Windows

El presente artículo trata de cómo **configurar PHP y Apache** para que trabajen conjuntamente en un sistema **Windows**. Además, este articulo asume que hay un servidor Apache configurado en el Windows, y que funciona correctamente.

> **Referencia:** Si deseamos conocer las distintas posibilidades para la instalación de PHP en los distintos sistemas operativos y servidores, puede ser de utilidad la lectura del artículo Instalación de PHP en nuestro servidor.

Existen dos formas de configurar PHP para trabajar con Apache, instalar como un módulo o instalar como un CGI.

Para instalar PHP como un CGI hay que seguir los siguientes pasos:

En primer lugar, hay que descargarse PHP desde la página de php.net. Existen dos versiones, una que tiene un instalador, y otra que es un fichero ZIP. Hay que descargarse esta última.

Una vez descargado, hay que descomprimirlo dentro de una carpeta, esta no tiene que estar bajo el árbol de directorios de Apache. El artículo asumirá que se descomprime dentro de la carpeta C:\PHP. Comprobar que los contenidos del archivo ZIP no quedan en un subdirectorio de la carptea C:\PHP, sino directamente en dicha carpeta.

Dentro de la carpeta c:\PHP se encuentra un fichero llamado PHP4ts.dll, hay que mover el fichero dentro de la carpeta: c:\windows\system ó c:\winnt\system

A continuación, dentro de la carpeta c:\php se encuentra un fichero llamado php.ini-recomended. Hay que copiar este fichero dentro de la carpeta c:\Windows, y renombrarlo a php.ini.

En este fichero se encuentra toda la configuración de PHP, y las modificaciones en la configuración de PHP (mostrar Errores, variables globales etc...) se encuentra dentro del mismo.
Es muy recomendable cambiar la directiva display_errors que por defecto esta en OFF, y pornerla en ON, para poder ver los errores que se producen en las páginas durante el desarrollo. Para un servidor en producción en conveniente dejarla en OFF.

Una vez se han hecho estos cambios, queda indicarle al Apache, donde se encuentra instalado el PHP, para ello hay que editar el fichero httpd.conf que se encuentra dentro de la carpeta conf, en la carpeta de instalación del apache (por defecto c:\archivos de programa\apache group\apache2\conf)

Abrir el fichero, y situarse al final del mismo, y escribir las siguientes líneas:

```
ScriptAlias /php/ "c:/php/"
AddType application/x-httpd-php .php
Action application/x-httpd-php "/php/php.exe"
```

En ellas se indica donde se encuentra el ejecutable de php, y lo asocia a los ficheros .php que se encuentren dentro de apache.

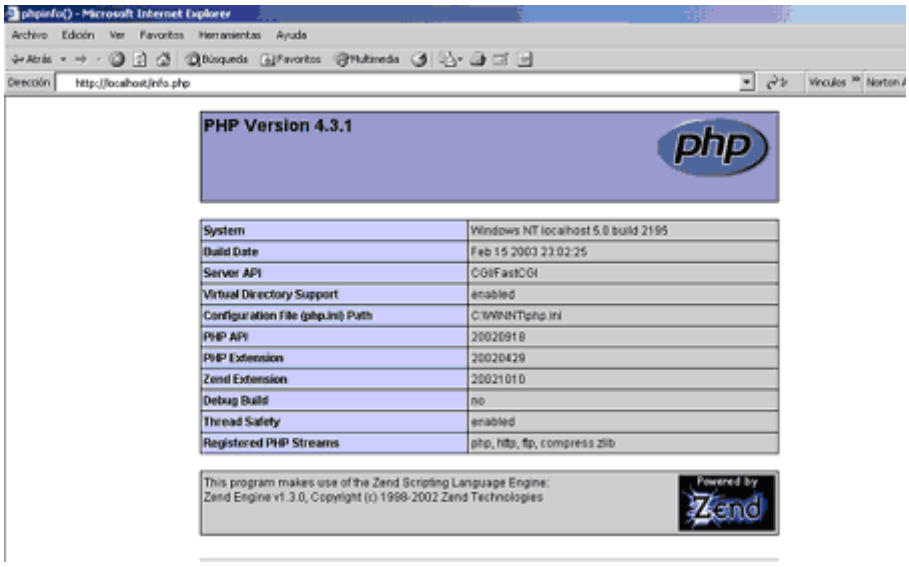
A continuación reiniciar el servidor Apache, y ya esta!

Por último, indicar que para probar la nueva instalación, es recomendable crear un fichero php con el siguiente contenido:

```
<? phpinfo();?>
```

Luego lo guardamosdentro de la carpeta raíz de documentos del Apache (por defecto c:\archivos de programa\apache group\apache2\htdocs), con un nombre terminado en .php, por ejemplo info.php

Para ejecutarlo, a través de un navegador, escribir la dirección http://localhost/info.php.
Debería aparecer una pantalla como la que se muestra a continuación.



Si la vemos correctamente es que todo ha ido bien y que tenemos perfectamente instalado PHP en nuestro servidor Apache.

Instalación de MySQL en Windows

Uno de los puntos fuertes de las páginas en PHP es la posibilidad de explotar bases de datos mediante funciones de una simplicidad y potencia muy agradecidas. Estas bases de datos pueden servir a nuestro sitio para almacenar contenidos de una forma sistemática que nos permita clasificarlos, buscarlos y editarlos rápida y fácilmente.

Una base de datos es sencillamente un conjunto de tablas en las que almacenamos distintos registros (artículos de una tienda virtual, proveedores o clientes de una empresa, películas en cartelera en el cine...). Estos registros son catalogados en función de distintos parámetros que los caracterizan y que presentan una utilidad a la hora de clasificarlos. Así, por ejemplo, los artículos de una tienda virtual podrían catalogarse a partir de distintos campos como puede ser un número de referencia, nombre del artículo, descripción, precio, proveedor...

La base de datos más difundida con el tandem UNIX-Apache es sin duda MySQL. Como para el caso de Apache, una versión para Windows está disponible y puede ser **descargada** gratis.

Su puesta a punto no entraña mucha dificultad. Una vez instalado el programa podemos ejecutar nuestras ordenes en modo MS-DOS. Para ello abrimos una ventana MS-DOS y nos colocamos en el

directorio *bin* de *mysql*. En este directorio se encuentran los archivos ejecutables. Aquí habrá que encontrar un archivo llamado *mysqld*. En el caso de la versión más actual durante la redacción de este articulo este archivo es llamado mysqld-shareware. Una vez ejecutado este archivo podemos ejecutar el siguiente: *mysql*.

Llegados a este punto veremos cómo un mensaje de bienvenida aparece en nuestra pantalla. En estos momentos nos encontramos dentro de la base de datos. A partir de ahí podemos realizar todo tipo de operaciones por sentencias SQL.

No vamos a entrar en una explicación pormenorizada del funcionamiento de esta base de datos ya que esto nos daría para un manual entero. Daremos como referencia nuestro **tutorial de SQL** a partir del cual se puede tener una idea muy práctica de las sentencias necesarias para la **creación** y edición de las tablas. También existe una documentación extensa en inglés en el directorio *Docs* de MySQL. A modo de resumen, aquí os proponemos además las operaciones más básicas que, combinadas nuestro **tutorial de SQL** pueden dar solución a gran parte de los casos que se os presenten:

Instrucción	Descripción
Show databases;	Muestra el conjunto de bases de datos presentes en el servidor
Use nombre_de_la_base	Determina la base de datos sobre la que vamos a trabajar
Create Database nombre_de_la_base;	Crea una nueva bd con el nombre especificado
Drop Database nombre_de_la_base;	Elimina la base de datos del nombre especificado
Show tables;	Muestra las tablas presentes en la base de datos actual
Describe nombre_de_la_tabla;	Describe los campos que componen la tabla
Drop Table nombre_de_la_tabla;	Borra la tabla de la base de datos
Load Data Local Infile "archivo.txt" Into Table nombre_de_la_tabla;	Crea los registros de la tabla a partir de un fichero de texto en el que separamos por tabulaciones todos los campos de un mismo registro.
Quit	Salir de MySQL

Para evitarnos el tener que editar nuestras tablas directamente sobre archivos de texto, puede resultar muy práctico usar cualquier otra base de datos con un editor y exportar a continuación la tabla en un archivo de texto configurado para dejar tabulaciones entre cada campo. Esto es posible en Access por ejemplo pinchando con el botón derecho sobre la tabla que queremos convertir y eligiendo la opción exportar. Una ventana de dialogo aparecerá en la que elegiremos guardar el archivo en tipo texto. El paso siguiente será elegir un formato delimitado por tabulaciones sin cualificador de texto.

Otra posibilidad que puede resultar muy práctica y que nos evita trabajar continuamente tecleando órdenes al estilo de antaño es servirse de programas en PHP o Perl ya existentes y descargables en la red. El más popular sin duda es **phpMyAdmin**. Este tipo de scripts son ejecutados desde un navegador y pueden ser por tanto albergados en nuestro servidor o empleados en local para, a partir de ellos, administrar MySQL de una forma menos sufrida.

Asimismo, dentro del directorio bin de MySQL, podemos encontrar una pequeña aplicación llamada MySqlManager. Se trata de una interface windows, más agradable a la vista y al uso que la que obtenemos ejecutando el archivo *mysql*. En este caso, las sentencias SQL deben realizarse sin el punto y coma final.

Introducción a la sintaxis PHP

PHP se escribe dentro de la propia página web, junto con el código HTML y, como para cualquier otro tipo de lenguaje incluido en un código HTML, en PHP necesitamos especificar cuáles son las partes constitutivas del código escritas en este lenguaje. Esto se hace, como en otros casos, delimitando nuestro código por etiquetas. Podemos utilizar distintos modelos de etiquetas en función de nuestras preferencias y costumbres. Hay que tener sin embargo en cuenta que no necesariamente todas están configuradas inicialmente y que otras, como es el caso de sólo están disponibles a partir de una determinada versión (3.0.4.).

Estos modos de abrir y cerrar las etiquetas son:

```
<?      y     ?>
<%      y     %>
```

```
<?php   y     ?>
<script languaje="php">
```

Este ultimo modo está principalmente aconsejado a aquellos que tengan el valor de trabajar con Front Page ya que, usando cualquier otro tipo de etiqueta, corremos el riesgo de que la aplicación nos la borre sin más debido a que se trata de un código incomprensible para ella.

El modo de funcionamiento de una página PHP, a grandes rasgos, no difiere del clásico para una página dinámica de lado servidor: El servidor va a reconocer la extensión correspondiente a la página PHP (phtml, php, php4,...) y antes de enviarla al navegador va a encargarse de interpretar y ejecutar todo aquello que se encuentre entre las etiquetas correspondientes al lenguaje PHP. El resto, lo enviara sin más ya que, asumirá que se trata de código HTML absolutamente comprensible por el navegador.

Otra característica general de los scripts en PHP es la forma de separar las distintas instrucciones. Para hacerlo, hay que acabar cada instrucción con un punto y coma ";". Para la ultima expresión, la que va antes del cierre de etiqueta, este formalismo no es necesario.

Incluimos también en este capitulo la sintaxis de comentarios. Un comentario, para aquellos que no lo sepan, es una frase o palabra que nosotros incluimos en el código para comprenderlo más fácilmente al volverlo a leer un tiempo después y que, por supuesto, el ordenador tiene que ignorar ya que no va dirigido a él sino a nosotros mismos. Los comentarios tienen una gran utilidad ya que es muy fácil olvidarse del funcionamiento de un script programado un tiempo atrás y resulta muy útil si queremos hacer rápidamente comprensible nuestro código a otra persona.

Pues bien, la forma de incluir estos comentarios es variable dependiendo si queremos escribir una línea o más. Veamos esto con un primer ejemplo de script:

```
<?
$mensaje="Tengo hambre!!"; //Comentario de una linea
echo $mensaje; #Este comentario también es de una linea
/*En este caso
mi comentario ocupa
varias lineas, lo ves? */
?>
```

Si usamos doble barra (//) o el símbolo # podemos introducir comentarios de una línea. Mediante /* y */ creamos comentarios multilínea. Por supuesto, nada nos impide de usar estos últimos en una sola línea.

No os preocupéis si no comprendéis el texto entre las etiquetas, todo llegará. Os adelantamos que las variables en PHP se definen anteponiendo un símbolo de dólar ($) y que la instrucción *echo* sirve para sacar en pantalla lo que hay escrito a continuación.

Recordamos que todo el texto insertado en forma de comentario es completamente ignorado por el servidor. Resulta importante acostumbrarse a dejar comentarios, es algo que se agradece con el tiempo.

Variables en PHP

En el manual de páginas dinámicas hemos introducido el **concepto de variable**. En el capítulo anterior comentábamos que, para PHP, las variables eran definidas anteponiendo el símbolo dólar ($) al nombre de la variable que estábamos definiendo.

Dependiendo de la información que contenga, una variable puede ser considerada de uno u otro tipo:

Variables numéricas Almacenan cifras		
Enteros	$entero=2002;	Numeros sin decimales
Real	$real=3.14159;	Numeros con o sin decimal

Variables alfanuméricas Almacenan textos compuestos de números y/o cifras

Cadenas	Almacenan variables alfanuméricas	$cadena="Hola amigo";

Tablas Almacenan series de informaciones numéricas y/o alfanuméricas		
Arrays	Son las variables que guardan las tablas	$sentido[1]="ver"; $sentido[2]="tocar"; $sentido[3]="oir"; $sentido[4]="gusto"; $sentido[5]="oler";

Objetos Se trata de conjuntos de variables y funciones asociadas. Presentan una complejidad mayor que las variables vistas hasta ahora pero su utilidad es más que interesante.

A diferencia de otros lenguajes, PHP posee una gran flexibilidad a la hora de operar con variables. En efecto, cuando definimos una variable asignándole un valor, el ordenador le atribuye un tipo. Si por ejemplo definimos una variable entre comillas, la variable será considerada de tipo cadena:

```
$variable="5"; //esto es una cadena
```

Sin embargo si pedimos en nuestro script realizar una operación matemática con esta variable, no obtendremos un mensaje de error sino que la variable cadena será asimilada a numérica:

```
<?
$cadena="5"; //esto es una cadena
$entero=3; //esto es un entero
echo $cadena+$entero
?>
```

Este script dará como resultado "8". La variable cadena ha sido asimilada en entero (aunque su tipo sigue siendo cadena) para poder realizar la operación matemática. Del mismo modo, podemos operar entre variables tipo entero y real. No debemos preocuparnos de nada, PHP se encarga durante la ejecución de interpretar el tipo de variable necesario para el buen funcionamiento del programa.

Sin embargo, en contraste, hay que tener cuidado en no cambiar mayúsculas por minúsculas ya que, en este sentido, PHP es sensible. Conviene por lo tanto trabajar ya sea siempre en mayúsculas o siempre en minúsculas para evitar este tipo de malentendidos a veces muy difíciles de localizar.

Variables de sistema

Dada su naturaleza de lenguaje de lado servidor, PHP es capaz de darnos acceso a toda una serie de variables que nos informan sobre nuestro servidor y sobre el cliente. La información de estas variables es atribuida por el servidor y en ningún caso nos es posible modificar sus valores directamente mediante el script. Para hacerlo es necesario influir directamente sobre la propiedad que definen.

Existen multitud de variables de este tipo, algunas sin utilidad aparente y otras realmente interesantes y con una aplicación directa para nuestro sitio web. Aquí os enumeramos algunas de estas variables y la información que nos aportan:

Variable	Descripción
$HTTP_USER_AGENT	Nos informa principalmente sobre el sistema operativo y tipo y versión de navegador utilizado por el internauta. Su principal utilidad radica en que, a partir de esta información, podemos redireccionar nuestros usuarios hacia páginas optimizadas para su navegador o realizar cualquier otro tipo de acción en el contexto de un navegador determinado.
$HTTP_ACCEPT_LANGUAGE	Nos devuelve la o las abreviaciones de la lengua considerada como principal por el navegador. Esta lengua o lenguas principales pueden ser elegidas en el menú de opciones del navegador. Esta variable resulta también extremadamente útil para enviar al internauta a las páginas

	escritas en su lengua, si es que existen.
$HTTP_REFERER	Nos indica la URL desde la cual el internauta ha tenido acceso a la página. Muy interesante para generar botones de "Atrás" dinámicos o para crear nuestros propios sistemas de estadísticas de visitas.
$PHP_SELF	Nos devuelve una cadena con la URL del script que está siendo ejecutado. Muy interesante para crear botones para recargar la página.
$HTTP_GET_VARS	Se trata de un array que almacena los nombres y contenidos de las variables enviadas al script por URL o por formularios GET
$HTTP_POST_VARS	Se trata de un array que almacena los nombres y contenidos de las variables enviadas al script por medio de un formulario POST
$HTTP_COOKIES_VARS	Se trata de un array que almacena los nombres y contenidos de las cookies. Veremos qué son más adelante.
$PHP_AUTH_USER	Almacena la variable usuario cuando se efectúa la entrada a páginas de acceso restringido. Combinado con $PHP_AUTH_PW resulta ideal para controlar el acceso a las páginas internas del sitio.
$PHP_AUTH_PW	Almacena la variable password cuando se efectúa la entrada a páginas de acceso restringido. Combinado con $PHP_AUTH_USER resulta ideal para controlar el acceso a las páginas internas del sitio.
$REMOTE_ADDR	Muestra la dirección IP del visitante.
$DOCUMENT_ROOT	Nos devuelve el path físico en el que se encuentra alojada la página en el servidor.
$PHPSESSID	Guarda el identificador de sesión del usuario. Veremos más adelante en qué consisten las sesiones.

No todas estas variables están disponibles en la totalidad de servidores o en determinadas versiones de un mismo servidor. además, algunas de ellas han de ser previamente activadas o definidas por medio de algún acontecimiento. Así, por ejemplo, la variable $HTTP_REFERER no estará definida a menos que el internauta acceda al script a partir de un enlace desde otra página.

Tablas o Arrays en PHP

Un tipo de variable que ya hemos descrito pero puede ser relativamente complicado a asimilar con respecto a la mayoría son los arrays. Un array es una variable que está compuesta de varios elementos cada uno de ellos catalogado dentro de ella misma por medio de una clave.

En el capítulos anteriores poníamos el ejemplo de un array llamado sentido que contenía los distintos sentidos del ser humano:

```
$sentido[1]="ver";
$sentido[2]="tocar";
$sentido[3]="oir";
$sentido[4]="gustar";
$sentido[5]="oler";
```

En este caso este array cataloga sus elementos, comunmente llamados valores, por números. Los números del 1 al 5 son por lo tanto las claves y los sentidos son los valores asociados. Nada nos impide emplear nombres (cadenas) para clasificarlos. Lo único que deberemos hacer es entrecomillarlos:

```
<?
$moneda["espana"]="Peseta";
$moneda["francia"]="Franco";
$moneda["usa"]="Dolar";
?>
```

Otra forma de definir idénticamente este mismo array y que nos puede ayudar para la creación de arrays más complejos es la siguiente sintaxis:

```
<?
```

```
$moneda=array("espana"=> "Peseta","francia" => "Franco","usa" => "Dolar");
?>
```

Una forma muy practica de almacenar datos es mediante la creación de arrays multidimensionales (tablas). Pongamos el ejemplo siguiente: Queremos almacenar dentro de una misma tabla el nombre, moneda y lengua hablada en cada país. Para hacerlo podemos emplear un array llamado país que vendrá definido por estas tres características (claves). Para crearlo, deberíamos escribir una expresión del mismo tipo que la vista anteriormente en la que meteremos una array dentro del otro. Este proceso de incluir una instruccion dentro de otra se llama anidar y es muy corriente en programación:

```
<?
$pais=array
(
"espana" =>array
  (
  "nombre"=>"España",
  "lengua"=>"Castellano",
  "moneda"=>"Peseta"
  ),
"francia" =>array
  (
  "nombre"=>"Francia",
  "lengua"=>"Francés",
  "moneda"=>"Franco"
  )
);
echo $pais["espana"]["moneda"] //Saca en pantalla: "Peseta"
?>
```

Antes de entrar en el detalle de este pequeño script, comentemos algunos puntos referentes a la sintaxis. Como puede verse, en esta secuencia de script, no hemos introducido punto y coma ";" al final de cada línea. Esto es simplemente debido a que lo que hemos escrito puede ser considerado como una sola instrucción. En realidad, somos nosotros quienes decidimos cortarla en varias líneas para, así, facilitar su lectura. La verdadera instrucción acabaría una vez definido completamente el array y es precisamente ahí donde hemos colocado el único punto y coma. Por otra parte, podéis observar cómo hemos jugado con el tabulador para separar unas líneas más que otras del principio. Esto también lo hacemos por cuestiones de claridad, ya que nos permite ver qué partes del código están incluidas dentro de otras. Es importante acostumbrarse a escribir de esta forma del mismo modo que a introducir los comentarios ya que la claridad de los scripts es fundamental a la hora de depurarlos. Un poco de esfuerzo a la hora de crearlos puede ahorrarnos muchas horas a la hora de corregirlos o modificarlos meses más tarde.

Pasando ya al comentario del programa, como podéis ver, éste nos permite almacenar tablas y, a partir de una simple petición, visualizarlas un determinado valor en pantalla.

Lo que es interesante es que la utilidad de los arrays no acaba aquí, sino que también podemos utilizar toda una serie de funciones creadas para ordenarlos por orden alfabético directo o inverso, por claves, contar el numero de elementos que componen el array además de poder movernos por dentro de él hacia delante o atrás.

Muchas son las **funciones** propuestas por PHP para el tratamiento de arrays, no vamos a entrar aquí en una descripción de las mismas. Sólo incluiremos esta pequeña tabla que puede ser complementada, si necesario, con la **documentación** que ya hemos mencionado.

Función	Descripción
array_values (mi_array)	Lista los valores contenidos en mi_array
asort(mi_array) y arsort(mi_array)	Ordena por orden alfabético directo o inverso en función de los valores
count(mi_array)	Nos da el numero de elementos de nuestro array
ksort(mi_array) y krsort(mi_array)	Ordena por orden alfabético directo o inverso en función de las claves
list ($variable1, $variable2...)=mi_array	Asigna cada una variable a cada uno de los valores del array
next(mi_array), prev(mi_array),	Nos permiten movernos por dentro del array con un

reset(mi_array) y end(mi_array)	puntero hacia delante, atras y al principio y al final.
each(mi_array)	Nos da el valor y la clave del elemento en el que nos encontramos y mueve al puntero al siguiente elemento.

De gran utilidad es también el bucle **foreach** que recorre de forma secuencial el array de principio a fin.

Trabajo con tablas o arrays en PHP

Vamos a ver varios ejemplos de trabajo con arrays (arreglos, vectores, matrices o tablas en castellano) en PHP que ilustrarán un poco el funcionamiento de algunas de las funciones de arrays más populares que trae consigo PHP.

Sin más, vamos a introducirnos en materia con varios ejemplos interesantes de manejo de vectores.

> **Referencia:** Los arrays en PHP se explican en el artículo Tablas o Arrays en PHP.

Modificar el número de elementos de un array

Ahora vamos a ver varios ejemplos mediante los cuales nuestros arrays pueden aumentar o reducir el número de casillas disponibles.

Reducir el tamaño de un array

array_slice()
Para disminuir el número de casillas de un arreglo tenemos varias funciones. Entre ellas, array_slice() la utilizamos cuando queremos recortar algunas casillas del arreglo, sabiendo los índices de las casillas que deseamos conservar.

Recibe tres parámetros. El array, el índice del primer elemento y el número de elementos a tomar, siendo este último parámetro opcional.

En el ejemplo siguiente tenemos un array con cuatro nombres propios. En la primera ejecución de array_slice() estamos indicando que deseamos tomar todos los elementos desde el índice 0 (el principio) hasta un número total de 3 elementos.

El segundo array_slice() indica que se tomen todos los elementos a partir del índice 1 (segunda casilla).

```
<?
$entrada = array ("Miguel", "Pepe", "Juan", "Julio", "Pablo");

//modifico el tamaño
$salida = array_slice ($entrada, 0, 3);
//muestro el array
foreach ($salida as $actual)
   echo $actual . "<br>";

echo "<p>";

//modifico otra vez
$salida = array_slice ($salida, 1);
//muestro el array
foreach ($salida as $actual)
   echo $actual . "<br>";
?>
```

Tendrá como salida:

Miguel
Pepe
Juan

Pepe
Juan

array_shift()
Esta función extrae el el primer elemento del array y lo devuelve. Además, acorta la longitud del array eliminando el elemento que estaba en la primera casilla. Siempre hace lo mismo, por tanto, no recibirá más que el array al que se desea eliminar la primera posición.

En el código siguiente se tiene el mismo vector con nombres propios y se ejecuta dos veces la función array_shift() eliminando un elemento en cada ocasión. Se imprimen los valores que devuelve la función y los elementos del array resultante de eliminar la primera casilla.

```
<?
$entrada = array ("Miguel", "Pepe", "Juan", "Julio", "Pablo");

//quito la primera casilla
$salida = array_shift ($entrada);
//muestro el array
echo "La función devuelve: " . $salida . "<br>";
foreach ($entrada as $actual)
   echo $actual . "<br>";

echo "<p>";

//quito la primera casilla, que ahora sería la segunda del array original
$salida = array_shift ($entrada);
echo "La función devuelve: " . $salida . "<br>";
//muestro el array
foreach ($entrada as $actual)
   echo $actual . "<br>";
?>
```

Da como resultado:

```
La función devuelve: Miguel
Pepe
Juan
Julio
Pablo

La función devuelve: Pepe
Juan
Julio
Pablo
```

unset()
Se utiliza para destruir una variable dada. En el caso de los arreglos, se puede utilizar para eliminar una casilla de un array asociativo (los que no tienen índices numéricos sino que su índice es una cadena de caracteres).

Veamos el siguiente código para conocer cómo definir un array asociativo y eliminar luego una de sus casillas.

```
<?
$estadios_futbol = array("Barcelona"=> "Nou Camp","Real Madrid" => "Santiago Bernabeu","Valencia"
=> "Mestalla","Real Sociedad" => "Anoeta");

//mostramos los estadios
foreach ($estadios_futbol as $indice=>$actual)
   echo $indice . " -- " . $actual . "<br>";

echo "<p>";

//eliminamos el estadio asociado al real madrid
unset ($estadios_futbol["Real Madrid"]);

//mostramos los estadios otra vez
foreach ($estadios_futbol as $indice=>$actual)
echo $indice . " -- " . $actual . "<br>";
?>
```

La salida será la siguiente:

```
Barcelona -- Nou Camp
Real Madrid -- Santiago Bernabeu
Valencia -- Mestalla
Real Sociedad -- Anoeta

Barcelona -- Nou Camp
```

Valencia -- Mestalla
Real Sociedad -- Anoeta

Aumentar el tamaño de un array

Tenemos también a nuestra disposición varias funciones que nos pueden ayudar a aumentar el número de casillas de un arreglo.

array_push()
Inserta al final del array una serie de casillas que se le indiquen por parámetro. Por tanto, el número de casillas del array aumentará en tantos elementos como se hayan indicado en el parámetro de la función. Devuelve el número de casillas del array resultante.

Veamos este código donde se crea un arreglo y se añaden luego tres nuevos valores.

```
<?
$tabla = array ("Lagartija", "Araña", "Perro", "Gato", "Ratón");

//aumentamos el tamaño del array
array_push($tabla, "Gorrión", "Paloma", "Oso");

foreach ($tabla as $actual)
   echo $actual . "<br>";
?>
```

Da como resultado esta salida:

Lagartija
Araña
Perro
Gato
Ratón
Gorrión
Paloma
Oso

array_merge()
Ahora vamos a ver cómo unir dos arrays utilizando la función array_merge(). A ésta se le pasan dos o más arrays por parámetro y devuelve un arreglo con todos los campos de los vectores pasados.

En este código de ejemplo creamos tres arrays y luego los unimos con la función array_merge()

```
<?
$tabla = array ("Lagartija", "Araña", "Perro", "Gato", "Ratón");
$tabla2 = array ("12","34","45","52","12");
$tabla3 = array ("Sauce","Pino","Naranjo","Chopo","Perro","34");

//aumentamos el tamaño del array
$resultado = array_merge($tabla, $tabla2, $tabla3);

foreach ($resultado as $actual)
   echo $actual . "<br>";
?>
```

Da como resultado:

Lagartija
Araña
Perro
Gato
Ratón
12
34
45
52
12
Sauce
Pino
Naranjo
Chopo
Perro
34

Una última cosa. También pueden introducirse nuevas casillas en un arreglo por los métodos habituales de asignar las nuevas posiciones en el array a las casillas que necesitemos.

En arrays normales se haría así:

```
$tabla = array ("Sauce","Pino","Naranjo");
$tabla[3]="Algarrobo";
```

En arrays asociativos:

```
$estadios_futbol = array("Valencia" => "Mestalla","Real Sociedad" => "Anoeta");
$estadios_futbol["Barcelona"]= "Nou Camp";
```

Veremos más adelante otras posibilidades del trabajo con arrays.

Cadenas

Una de las variables más corrientes a las que tendremos que hacer frente en la mayoría de nuestros scripts son las cadenas, que no son más que información de carácter no numérico (textos, por ejemplo).

Para asignar a una variable un contenido de este tipo, lo escribiremos entre comillas dando lugar a declaraciones de este tipo:

```
$cadena="Esta es la información de mi variable"
```

Si queremos ver en pantalla el valor de una variable o bien un mensaje cualquiera usaremos la instrucción *echo* como ya lo hemos visto anteriormente:

```
echo $cadena //sacaría "Esta es la información de mi variable"
```

```
echo "Esta es la información de mi variable" //daría el mismo resultado
```

Podemos yuxtaponer o concatenar varias cadenas poniendo para ello un punto entre ellas:

```
<?
$cadena1="Perro";
$cadena2=" muerde";
$cadena3=$cadena1.$cadena2;
echo $cadena3 //El resultado es: "Perro muerde"
?>
```

También podemos introducir variables dentro de nuestra cadena lo cual nos puede ayudar mucho en el desarrollo de nuestros scripts. Lo que veremos no es el nombre, sino el valor de la variable:

```
<?
$a=55;
$mensaje="Tengo $a años";
echo $mensaje //El resultado es: "Tengo 55 años"
?>
```

La pregunta que nos podemos plantear ahora es...¿Cómo hago entonces para que en vez del valor "55" me salga el texto "$a"? En otras palabras, cómo se hace para que el símbolo $ no defina una variable sino que sea tomado tal cual. Esta pregunta es tanto más interesante cuanto que en algunos de scripts este símbolo debe ser utilizado por una simple razón comercial (pago en dólares por ejemplo) y si lo escribimos tal cual, el ordenador va a pensar que lo que viene detrás es una variable siendo que no lo es.

Pues bien, para meter éste y otros caracteres utilizados por el lenguaje dentro de las cadenas y no confundirlos, lo que hay que hacer es escribir una contrabarra delante:

Carácter	Efecto en la cadena
\$	Escribe dólar en la cadena

\"	Escribe comillas en la cadena
\\	Escribe contrabarra en la cadena
\8/2	Escribe 8/2 y no 4 en la cadena

Además, existen otras utilidades de esta contrabarra que nos permiten introducir en nuestro documento HTML determinados eventos:

Carácter	Efecto en la cadena
\t	Introduce una tabulación en nuestro texto
\n	Cambiamos de línea
\r	Retorno de carro

Estos cambios de línea y tabulaciones tienen únicamente efecto en el código y no en el texto ejecutado por el navegador. En otras palabras, si queremos que nuestro texto ejecutado cambie de línea hemos de introducir un *echo "
"* y no *echo "\n"* ya que este ultimo sólo cambia de línea en el archivo HTML creado y enviado al navegador cuando la página es ejecutada en el servidor. La diferencia entre estos dos elementos puede ser fácilmente comprendida mirando el código fuente producido al ejecutar este script:

```
<HTML>
<HEAD>
<TITLE>cambiolinea.php</TITLE>
</HEAD>
<BODY>
<?
echo "Hola, \n sigo en la misma línea ejecutada pero no en código fuente.<br>Ahora cambio de línea ejecutada pero continuo en la misma en el código fuente."
?>
</BODY>
</HTML>
```

El código fuente que observaríamos seria el siguiente:

```
<HTML>
<HEAD>
<TITLE>cambiolinea.php</TITLE>
</HEAD>
<BODY>
Hola,
 sigo en la misma línea ejecutada pero no en código fuente.<br>Ahora cambio de línea ejecutada pero continuo en la misma en el código fuente.</BODY>
</HTML>
```

Las cadenas pueden asimismo ser tratadas por medio de funciones de todo tipo. Veremos estas funciones más adelante con más detalle. Tan sólo debemos retener que existen muchas posibles acciones que podemos realizar sobre ellas: Dividirlas en palabras, eliminar espacios sobrantes, localizar secuencias, remplazar caracteres especiales por su correspondiente en HTML o incluso extraer las etiquetas META de una página web.

Funciones

En nuestro manual de páginas dinámicas vimos el **concepto de función**. Vimos que la función podría ser definida como un conjunto de instrucciones que explotan ciertas variables para realizar una tarea más o menos elemental.

PHP basa su eficacia principalmente en este tipo de elemento. Una gran librería que crece constantemente, a medida que nuevas versiones van surgiendo, es complementada con las funciones de propia cosecha dando como resultado un sinfín de recursos que son aplicados por una simple llamada.

Las funciones integradas en PHP son muy fáciles de utilizar. Tan sólo hemos de realizar la llamada de la forma apropiada y especificar los parámetros y/o variables necesarios para que la función realice su

tarea.

Lo que puede parecer ligeramente más complicado, pero que resulta sin lugar a dudas muy práctico, es crear nuestras propias funciones. De una forma general, podríamos crear nuestras propias funciones para conectarnos a una base de datos o crear los encabezados o etiquetas meta de un documento HTML. Para una aplicación de comercio electrónico podríamos crear por ejemplo funciones de cambio de una moneda a otra o de calculo de los impuestos a añadir al precio de articulo. En definitiva, es interesante crear funciones para la mayoría de acciones más o menos sistemáticas que realizamos en nuestros programas.

Aquí daremos el ejemplo de creación de una función que, llamada al comienzo de nuestro script, nos crea el encabezado de nuestro documento HTML y coloca el titulo que queremos a la página:

```
<?
Function hacer_encabezado($titulo)
{
$encabezado="<html>\n<head>\n\t<title>$titulo</title>\n</head>\n";
echo $encabezado;
}
?>
```

Esta función podría ser llamada al principio de todas nuestras páginas de la siguiente forma:

```
$titulo="Mi web";
hacer_encabezado($titulo);
```

De esta forma automatizamos el proceso de creación de nuestro documento. Podríamos por ejemplo incluir en la función otras variables que nos ayudasen a construir la etiquetas meta y de esta forma, con un esfuerzo mínimo, crearíamos los encabezados personalizados para cada una de nuestras páginas. De este mismo modo nos es posible crear cierres de documento o formatos diversos para nuestros textos como si se tratase de hojas de estilo que tendrían la ventaja de ser reconocidas por todos los navegadores.

Por supuesto, la función ha de ser definida dentro del script ya que no se encuentra integrada en PHP sino que la hemos creado nosotros. Esto en realidad no pone ninguna pega ya que puede ser incluida desde un archivo en el que iremos almacenando las definiciones de las funciones que vayamos creando o recopilando.

Estos archivos en los que se guardan las funciones se llaman librerías. La forma de incluirlos en nuestro script es a partir de la instrucción *require* o *include*:

```
require("libreria.php") o include("libreria.php")
```

En resumen, la cosa quedaría así:

Tendríamos un archivo libreria.php como sigue

```
<?
//función de encabezado y colocación del titulo
Function hacer_encabezado($titulo)
{
$encabezado="<html>\n<head>\n\t<title>$titulo</title>\n</head>\n";
echo $encabezado;
}
?>
```

Por otra parte tendríamos nuestro script principal página.php (por ejemplo):

```
<?
include("libreria.php");
$titulo="Mi Web";
hacer_encabezado($titulo);
?>
<body>
El cuerpo de la página
</body>
</html>
```

Podemos meter todas las funciones que vayamos encontrando dentro de un mismo archivo pero resulta muchísimo más ventajoso ir clasificándolas en distintos archivos por temática: Funciones de conexión a bases de datos, funciones comerciales, funciones generales, etc. Esto nos ayudara a poder localizarlas antes para corregirlas o modificarlas, nos permite también cargar únicamente el tipo de función que necesitamos para el script sin recargar éste en exceso además de permitirnos utilizar un determinado tipo de librería para varios sitios webs distintos.

También puede resultar muy práctico el utilizar una nomenclatura sistemática a la hora de nombrarlas: Las funciones comerciales podrían ser llamadas com_loquesea, las de bases de datos bd_loquesea, las de tratamiento de archivos file_loquesea. Esto nos permitirá reconocerlas enseguida cuando leamos el script sin tener que recurrir a nuestra oxidada memoria para descubrir su utilidad.

No obstante, antes de lanzarnos a crear nuestra propia función, merece la pena echar un vistazo a la **documentación** para ver si dicha función ya existe o podemos aprovecharnos de alguna de las existentes para aligerar nuestro trabajo. Así, por ejemplo, existe una función llamada header que crea un encabezado HTML configurable lo cual nos evita tener que crearla nosotros mismos.

Como puede verse, la tarea del programador puede en algunos casos parecerse a la de un coleccionista. Hay que ser paciente y metódico y al final, a base de trabajo propio, intercambio y tiempo podemos llegar poseer nuestro pequeño tesoro.

Control del flujo en PHP: Condiciones IF

La programación exige en muchas ocasiones la repetición de acciones sucesivas o la elección de una determinada secuencia y no de otra dependiendo de las condiciones específicas de la ejecución.

Como ejemplo, podríamos hacer alusión a un script que ejecute una secuencia diferente en función del día de la semana en el que nos encontramos.

Este tipo de acciones pueden ser llevadas a cabo gracias a una paleta de instrucciones presentes en la mayoría de los lenguajes. En este capítulo describiremos someramente algunas de ellas propuestas por PHP y que resultan de evidente utilidad.

Para evitar el complicar el texto, nos limitaremos a introducir las más importantes dejando de lado otras cuantas que podrán ser fácilmente asimilables a partir de ejemplos prácticos.

Las condiciones if
Cuando queremos que el programa, llegado a un cierto punto, tome un camino concreto en determinados casos y otro diferente si las condiciones de ejecución difieren, nos servimos del conjunto de instrucciones *if, else* y *elseif*. La estructura de base de este tipo de instrucciones es la siguiente:

```
if (condición)
{
   Instrucción 1;
   Instrucción 2;
   ...
}
else
{
   Instrucción A;
   Instrucción B;
   ...
}
```

Llegados a este punto, el programa verificará el cumplimiento o no de la condición. Si la condición es cierta las instrucciones 1 y 2 serán ejecutadas. De lo contrario (*else*), las instrucciones A y B serán llevadas a cabo.

Esta estructura de base puede complicarse un poco más si tenemos cuenta que no necesariamente todo es blanco o negro y que muchas posibilidades pueden darse. Es por ello que otras condiciones pueden plantearse dentro de la condición principal. Hablamos por lo tanto de condiciones anidadas que tendrían una estructura del siguiente tipo:

```
if (condición1)
{
   Instrucción 1;
   Instrucción 2;
   ...
}
```

```
else
{
  if (condición2)
  {
    Instrucción A;
    Instrucción B;
    ...
  }
  else
  {
    Instrucción X
    ...
  }
}
```

De este modo podríamos introducir tantas condiciones como queramos dentro de una condición principal.

De gran ayuda es la instrucción *elseif* que permite en una sola línea introducir una condición adicional. Este tipo de instrucción simpifica ligeramente la sintaxis que acabamos de ver:

```
if (condición1)
{
  Instrucción 1;
  Instrucción 2;
  ...
}
elseif (condición2)
{
  Instrucción A;
  Instrucción B;
  ...
}
else
{
  Instrucción X
  ...
}
```

El uso de esta herramienta resultará claro con un poco de práctica. Pongamos un ejemplo sencillo de utilización de condiciones. El siguiente programa permitiría detectar la lengua empleada por el navegador y visualizar un mensaje en dicha lengua.

```
<HTML>
<HEAD>
<TITLE>Detector de Lengua</TITLE>
</HEAD>
<BODY>
<?
//Antes de nada introducimos mensajes en forma de variables
$espanol="Hola";
$ingles="Hello";
$frances="Bonjour";

//Ahora leemos del navegador cuál es su lengua oficial
$idioma=substr($HTTP_ACCEPT_LANGUAGE,0,2);

//Formulamos las posibilidades que se pueden dar
if ($idioma == "es")
{echo "$espanol";}
elseif ($idioma=="fr")
{echo "$frances";}
else
{echo "$ingles";}
?>
</BODY>
</HTML>
```

Para poder ver el funcionamiento de este script es necesario cambiar el idioma preferido lo cual puede

ser realizado a partir del menú de opciones del navegador.

Para leer la lengua aceptada por el navegador lo que hacemos es definir una variable (*$idioma*) y, mediante la función *substr*, recogemos las dos primeras letras del código correspondiente al idioma aceptado por el navegador ($HTTP_ACCEPT_LANGUAGE).

La tercera parte de script se encarga de ver si el navegador está en español (es), francés (fr) o en cualquier otro idioma que no sea ninguno de estos dos y de imprimir el mensaje que proceda en cada caso.

A notar que, cuando se trata de comparar variables, ponemos un doble igual "==" en lugar de un simple "=". Este último queda reservado exclusivamente para asignar valores a variables

Control del flujo en PHP: Bucles I

Los ordenadores, como cualquier máquina, están diseñados para realizar tareas repetitivas. Es por ello que nuestros programas pueden aprovecharse de este principio para realizar una determinada secuencia de instrucciones un cierto número de veces. Para ello, utilizamos las estructuras llamadas en bucle que nos ayudan a, usando unas pocas líneas, realizar una tarea incluida dentro del bucle un cierto numero de veces definido por nosotros mismos.

PHP propone varios tipos de bucle cada uno con características especificas:

Bucle while
Sin duda el bucle más utilizado y el más sencillo. Lo usamos para ejecutar las instrucciones contenidas en su interior siempre y cuando la condición definida sea verdadera. La estructura sintáctica es la siguiente.

```
while (condición)
{
  instruccion1;
  instruccion2;
  ...
}
```

Un ejemplo sencillo es este bucle que aumenta el tamaño de la fuente en una unidad a cada nueva vuelta por el bucle:

```
<?
$size=1;
While ($size<=6)
{
  echo"<font size=$size>Tamaño $size</font><br>\n";
  $size++;
}
?>
```

A modo de explicación, diremos que, antes de nada, hemos de definir el valor de la variable que vamos a evaluar en la condición. Algo absolutamente obvio pero fácil de olvidar. En este caso le hemos atribuido un valor de 1 que corresponde a la letra más pequeña.

El paso siguiente es crear el bucle en el que imponemos la condición que la variable no exceda el valor de 6.

La instrucción a ejecutar será imprimir en nuestro documento un código HTML en el que la etiqueta *font* y el mensaje que contiene varían a medida que *$size* cambia su valor.

El siguiente paso es incrementar en una unidad el valor de $size. Esto se puede hacer con una expresión como la mostrada en el bucle que en realidad es sinónima de:

$size=$size+1

Veremos otras de estas abreviaciones más adelante.

Bucle do/while
Este tipo de bucle no difiere en exceso del anterior. La sintaxis es la siguiente:

```
do
{
```

```
  instruccion1;
  instruccion2;
  ...
}
while (condición)
```

La diferencia con respecto a los bucles *while* es que este tipo de bucle evalúa la condición al final con lo que, incluso siendo falsa desde el principio, éste se ejecuta al menos una vez.

Control del flujo en PHP: Bucles II

Bucle for
PHP está provisto de otros tipos de bucle que también resultan muy prácticos en determinadas situaciones. El más popular de ellos es el bucle *for* que, como para los casos anteriores, se encarga de ejecutar las instrucciones entre llaves. La diferencia con los anteriores radica en cómo se plantea la condición de finalización del bucle. Para aclarar su funcionamiento vamos a expresar el ejemplo de bucle *while* visto en el capítulo anterior en forma de bucle *for*:

```
<?
For ($size=1;$size<=6;$size++)
{
  echo"<font size=$size>Tamaño $size</font><br>\n";
}
?>
```

Las expresiones dentro del paréntesis definen respectivamente:

-Inicialización de la variable. Valida para la primera vuelta del bucle.

-Condición de evaluación a cada vuelta. Si es cierta, el bucle continua.

-Acción a realizar al final de cada vuelta de bucle.

Bucle foreach
Este bucle, implementado en las versiones de PHP4, nos ayuda a recorrer los valores de un array lo cual puede resultar muy útil por ejemplo para efectuar una lectura rápida del mismo. Recordamos que un array es una variable que guarda un conjunto de elementos (valores) catalogados por claves.

La estructura general es la siguiente:

```
Foreach ($array as $clave=>$valor)
{
  instruccion1;
  instruccion2;
  ...;
}
```

Un ejemplo práctico es la lectura de un array lo cual podría hacerse del siguiente modo:

```
<?
$moneda=array("España"=> "Peseta","Francia" => "Franco","USA" => "Dolar");
Foreach ($moneda as $clave=>$valor)
{
  echo "Pais: $clave Moneda: $valor<br>";
}
?>
```

Este script se encargaría de mostrarnos por pantalla el contenido del array *$moneda*. No resultaría mala idea crear una función propia basada en este bucle que nos permitiese visualizar arrays monodimensionales y almacenarla en nuestra librería. Esta función podría ser definida de esta forma:

```
Function mostrar_array ($array)
{
Foreach ($array as $clave=>$valor)
{echo "$clave=>$valor<br>";}
}
```

Break y continue
Estas dos instrucciones se introducen dentro de la estructura y nos sirven respectivamente para escapar

del bucle o saltar a la iteración siguiente. Pueden resultarnos muy prácticas en algunas situaciones.

Operadores

Las variables, como base de información de un lenguaje, pueden ser creadas, modificadas y comparadas con otras por medio de los llamados operadores. En los capítulos anteriores hemos utilizado en nuestros ejemplos algunos de ellos.

En este capítulo pretendemos listar los más importantes y así dar constancia de ellos para futuros ejemplos.

Operadores aritméticos
Nos permiten realizar operaciones numéricas con nuestras variables

+	Suma
-	Resta
*****	Multiplicación
/	División
%	Devuelve el resto de la división

Referencia: El operador aritmético que puede resultar más desconocido para los lectores es el operador %. Explicamos con mayor detenimiento su funcionamiento y un ejemplo en el que es útil en el taller: Listas de elementos con colores alternos en PHP.

Operadores de comparación
Se utilizan principalmente en nuestras condiciones para comparar dos variables y verificar si cumple o no la propiedad del operador.

==	Igualdad
!=	Desigual
<	Menor que
<=	Menor igual que
>	Mayor que
>=	Mayor igual que

Operadores lógicos
Se usan en combinación con los operadores de comparación cuando la expresión de la condición lo requiere.

And	Y
Or	O
!	No

Operadores de incremento
Sirven para aumentar o disminuir de una unidad el valor de una variable

++$variable	Aumenta de 1 el valor de $variable
--$variable	Reduce de uno el valor de $variable

Operadores combinados
Una forma habitual de modificar el valor de las variables es mediante los operadores combinados:

$variable **+=** 10	Suma 10 a $variable
$variable **-=** 10	Resta 10 a $variable

$variable **.=** "añado"	Concatena las cadenas $variable y "añado"

Este tipo de expresiones no son más que abreviaciones de otras formas más clásicas:

$variable += 10

es lo mismo que:

$variable = $variable+10

Pasar variables por la URL

Bucles y condiciones son muy útiles para procesar los datos dentro de un mismo script. Sin embargo, en un sitio Internet, las páginas vistas y los scripts utilizados son numerosos. Muy a menudo necesitamos que nuestros distintos scripts estén conectados unos con otros y que se sirvan de variables comunes. Por otro lado, el usuario interacciona por medio de formularios cuyos campos han de ser procesados para poder dar una respuesta. Todo este tipo de factores dinámicos han de ser eficazmente regulados por un lenguaje como PHP.

Es posible que ya os hayáis percatado de que las variables de un script tienen una validez exclusiva para el script y que nos resulta imposible conservar su valor cuando ejecutamos otro archivo distinto aunque ambos estén enlazados. Existen varias formas de enviar las variables de una página a otra de manera a que la página destino reconozca el valor asignado por el script de origen:

Pasar variables por URL

Para pasar las variables de una página a otra lo podemos hacer introduciendo dicha variable dentro del enlace hipertexto de la página destino. La sintaxis sería la siguiente:

```
<a href="destino.php?variable1=valor1&variable2=valor2&...">Mi enlace</a>
```

Podéis observar que estas variables no poseen el símbolo $ delante. Esto es debido a que en realidad este modo de pasar variables no es específico de PHP sino que es utilizado por otros lenguajes.

Ahora nuestra variable pertenece también al entorno de la página *destino.php* y está lista para su explotación.

> **Nota:** No siempre se definen automáticamente las variables recibidas por parámetro en las páginas web, depende de una variable de configuración de PHP: register_globals, que tiene que estar activada para que así sea. Ver comentarios del artículo al final de la página para más información.

Para aclarar posibles dudas, veamos esto en forma de ejemplo. Tendremos pues dos páginas, *origen.html* (no es necesario darle extensión PHP puesto que no hay ningún tipo de código) y *destino.php*:

```
<HTML>
<HEAD>
<TITLE>origen.html</TITLE>
</HEAD>
<BODY>
<a href="destino.php?saludo=hola&texto=Esto es una variable texto">Paso variables saludo y texto a la página destino.php</a>
</BODY>
</HTML>
```

```
<HTML>
<HEAD>
<TITLE>destino.php</TITLE>
</HEAD>
<BODY>
<?
echo "Variable \$saludo: $saludo <br>\n";
echo "Variable \$texto: $texto <br>\n"
?>
</BODY>
```

```
</HTML>
```

$HTTP_GET_VARS

Recordamos que es posible recopilar en una variable tipo array el conjunto de variables que han sido enviadas al script por este método a partir de la variable de sistema $HTTP_GET_VARS, que es un array asociativo. Utilizándolo quedaría así:

```
<?
echo "Variable \$saludo: $HTTP_GET_VARS["saludo"] <br>\n";
echo "Variable \$texto: $HTTP_GET_VARS["texto"] <br>\n"
?>
```

> **Nota:** Aunque podamos recoger variables con este array asociativo o utilizar directamente las variables que se definen en nuestra página, resulta más seguro utilizar $HTTP_GET_VARS por dos razones, la primera que así nos aseguramos que esa varible viene realmente de la URL y la segunda, que así nuestro código será más claro cuando lo volvamos a leer, porque quedará especificado que esa variable estamos recibiéndola por la URL.

$_GET

A partir de la versión 4.1.0 de PHP se ha introducido el array asociativo $_GET, que es idéntico a $HTTP_GET_VARS, aunque un poco más corto de escribir.

Procesar variables de formularios

Este tipo de transferencia es de gran utilidad ya que nos permite interaccionar directamente con el usuario.

El proceso es similar al explicado para las URLs. Primeramente, presentamos una primera página con el formulario clásico a rellenar y las variables son recogidas en una segunda página que las procesa:

> **Nota:** No siempre se definen automáticamente las variables recibidas por el formulario en las páginas web, depende de una variable de configuración de PHP: register_globals, que tiene que estar activada para que así sea. Ver comentarios del artículo al final de la página para más información.

```
<HTML>
<HEAD>
<TITLE>formulario.html</TITLE>
</HEAD>
<BODY>
<FORM METHOD="POST" ACTION="destino2.php">
Nombre<br>
<INPUT TYPE="TEXT" NAME="nombre"><br>
Apellidos<br>
<INPUT TYPE="TEXT" NAME="apellidos"><br>
<INPUT TYPE="SUBMIT">
</FORM>
</BODY>
</HTML>
```

```
<HTML>
<HEAD>
<TITLE>destino2.php</TITLE>
</HEAD>
<BODY>
<?
echo "Variable \$nombre: $nombre <br>\n";
echo "Variable \$apellidos: $apellidos <br>\n"
?>
```

```
</BODY>
</HTML>
```

$HTTP_POST_VARS

Recordamos que es posible recopilar en una variable tipo array el conjunto de variables que han sido enviadas al script por este método a partir de la variable de sistema $HTTP_POST_VARS.

```
echo "Variable \$nombre: " . $HTTP_POST_VARS["nombre"] . "<br>\n";
```

> **Nota:** Aunque podamos recoger variables con este array asociativo o utilizar directamente las variables que se definen en nuestra página, resulta más seguro utilizar $HTTP_POST_VARS por dos razones, la primera que así nos aseguramos que esa varible viene realmente de un formulario y la segunda, que así nuestro código será más claro cuando lo volvamos a leer, porque quedará especificado que esa variable estamos recibiéndola por un formulario.

$_POST

A partir de PHP 4.1.0 se pueden recoger las variables de formulario utilizando también el array asociativo $_POST, que es el mismo que $HTTP_POST_VARS, pero más corto de escribir.

Utilización de las cookies

Sin duda este término resultara familiar para muchos. Algunos lo habrán leído u oído pero no saben de qué se trata. Otros sin embargo sabrán que las cookies son unas informaciones almacenadas por un sitio web en el disco duro del usuario. Esta información es almacenada en un archivo tipo texto que se guarda cuando el navegador accede al sitio web.

> **Referencia:** Una explicación de las cookies más detallada se puede encontrar en el artículo Qué son las cookies, publicado en DesarrolloWeb.com.

Es posible, por supuesto, ver estos archivos. Para abrirlos hay que ir al directorio C:\Windows\Cookies para los usuarios de IE 4+ o a C:\...\Netscape\Users\defaultuser para usuarios de Netscape. Como podréis comprobar, en la mayoría de los casos la información que se puede obtener es indescifrable.

La utilidad principal de las cookies es la de poder identificar al navegador una vez éste visita el sitio por segunda vez y así, en función del perfil del cliente dado en su primera visita, el sitio puede adaptarse dinámicamente a sus preferencias (lengua utilizada, colores de pantalla, formularios rellenados total o parcialmente, redirección a determinadas páginas...).

Para crear un archivo cookies, modificar o generar una nueva cookie lo podemos hacer a partir de la función SetCookie:

```
setcookie("nombre_de_la_cookie",valor,expiracion);
```

Pongamos un ejemplo sencillo. Imaginemos que queremos introducir en una variable cookie el nombre del visitante. El nombre ha podido ser previamente recogido por un formulario tal y como hemos visto:

```
setcookie("persona",$nombre,time()+86400*365);
```

De este modo hemos creado una cookie llamada persona que tiene como valor el contenido de la variable $nombre y tendrá una duración de 1 año a partir de su creación (el tiempo time() actual en segundos sumado a un año en segundos).

Es importante que la creación de la cookie sea previa a la apertura del documento HTML. En otras palabras, las llamadas a la función setcookie() deben ser colocadas antes de la etiqueta HTML.

Por otra parte, es interesante señalar que el hecho de que definir una cookie ya existente implica el borrado de la antigua. Del mismo modo, el crear una primera cookie conlleva la generación automática del archivo texto.

Para utilizar el valor de la cookie en nuestros scripts tan sólo tendremos que llamar la variable que define la cookie. ¡Realmente sencillo!

Hay que tener cuidado sin embargo de no definir variables en nuestro script con el mismo nombre que las cookies puesto que PHP privilegiará el contenido de la variable local con respecto a la cookie y no dará un mensaje de error. Esto nos puede conducir a errores realmente difíciles de detectar.

Recordamos que es posible recopilar en una variable tipo array el conjunto de cookies almacenadas en el disco duro del internauta mediante la variable de servidor $HTTP_COOKIES_VARS

Las cookies son una herramienta fantástica para personalizar nuestra página pero hay que ser cautos ya que, por una parte, no todos los navegadores las aceptan y por otra, se puede deliberadamente impedir al navegador la creación de cookies. Es por ello que resultan un complemento y no una fuente de variables infalible para nuestro sitio.

Sesiones I

En los programas que hemos visto hasta ahora, hemos utilizado variables que sólo existían en el archivo que era ejecutado. Cuando cargábamos otra página distinta, los valores de estas variables se perdían a menos que nos tomásemos la molestia de pasarlos por la URL o inscribirlos en las cookies o en un formulario para su posterior explotación. Estos métodos, aunque útiles, no son todo lo prácticos que podrían en determinados casos en los que la variable que queremos conservar ha de ser utilizada en varios scripts diferentes y distantes los unos de los otros.

Podríamos pensar que ese problema puede quedar resuelto con las cookies ya que se trata de variables que pueden ser invocadas en cualquier momento. El problema, ya lo hemos dicho, es que las cookies no son aceptadas ni por la totalidad de los usuarios ni por la totalidad de los navegadores lo cual implica que una aplicación que se sirviera de las cookies para pasar variables de un archivo a otro no sería 100% infalible. Es importante a veces pensar en "la inmensa minoría", sobre todo en aplicaciones de comercio electrónico donde debemos captar la mayor cantidad de clientes posibles y nuestros scripts deben estar preparados ante cualquier eventual deficiencia del navegador del cliente.

Nos resulta pues necesario el poder declarar ciertas variables que puedan ser reutilizadas tantas veces como queramos dentro de una misma sesión. Imaginemos un sitio multilingüe en el que cada vez que queremos imprimir un mensaje en cualquier página necesitamos saber en qué idioma debe hacerse. Podríamos introducir un script identificador de la lengua del navegador en cada uno de los archivos o bien declarar una variable que fuese valida para toda la sesión y que tuviese como valor el idioma reconocido en un primer momento.

Pensemos también en un carrito de la compra de una tienda virtual donde el cliente va navegando por las páginas del sitio y añadiendo los artículos que quiere comprar a un carrito. Este carrito podría ser perfectamente una variable de tipo array (tabla) que almacena para cada referencia la cantidad de artículos contenidos en el carrito. Esta variable debería ser obviamente conservada continuamente a lo largo de todos los scripts.

Este tipo de situaciones son solventadas a partir de las variables de sesión. Una sesión es considerada como el intervalo de tiempo empleado por un usuario en recorrer nuestras páginas hasta que abandona nuestro sitio o deja de actuar sobre él durante un tiempo prolongado o bien, sencillamente, cierra el navegador.

PHP nos permite almacenar variables llamadas de sesión que, una vez definidas, podrán ser utilizadas durante este lapso de tiempo por cualquiera de los scripts de nuestro sitio. Estas variables serán especificas del usuario de modo que varias variables sesión del mismo tipo con distintos valores pueden estar coexistiendo para cada una de las sesiones que están teniendo lugar simultáneamente. Estas sesiones tienen además su propio identificador de sesión que será único y específico.

Algunas mejoras referentes al empleo de sesiones han sido introducidas con PHP4. Es a esta nueva versión a la que haremos referencia a la hora de explicar las funciones disponibles y la forma de operar. Para los programadores de PHP3 la diferencia mayor es que están obligados a gestionar ellos mismos las sesiones definir sus propios identificadores de sesión.

Veamos en el siguiente capítulo la forma de plasmar esta necesidad técnica en nuestros scripts a partir de las funciones que gestionan las sesiones en PHP.

Sesiones II

Hemos dicho en el capítulo anterior que la variables de sesión se diferencian de las variables clásicas en que éstas residen en el servidor, son especificas de un solo usuario definido por un identificador y pueden ser utilizadas en la globalidad de nuestras páginas.

Para iniciar una sesión podemos hacerlo de dos formas distintas:

-Declaramos abiertamente la apertura de sesión por medio de la función *session_start()*. Esta función crea una nueva sesión para un nuevo visitante o bien recupera la que está siendo llevada a cabo.

-Declaramos una variable de sesión por medio de la función *session_register('variable')*. Esta función, además de crear o recuperar la sesión para la página en la que se incluye también sirve para introducir

una nueva variable de tipo sesión.

Las sesiones han de ser iniciadas al principio de nuestro script. Antes de abrir cualquier etiqueta o de imprimir cualquier cosa. En caso contrario recibiremos un error.

Con lo visto, vamos a proponer el ejemplo clásico de utilización de una sesión: un contador. Este contador deberá aumentar de una unidad cada vez que recargamos la página o apretamos al enlace:

```
<?
session_register('contador');
?>
<HTML>
<HEAD>
<TITLE>contador.php</TITLE>
</HEAD>
<BODY>
<?
If (isset($contador)==0)
{$contador=0;}
++$contador;
echo "<a href=\"contador.php\">Has recargado esta página $contador veces</a>";
?>
</BODY>
</HTML>
```

La condición *if* tiene en cuenta la posibilidad de que la variable *$contador* no haya sido todavía inicializada. La función *isset* se encarga de dar un valor cero cuando una variable no ha sido inicializada.

Otras funciones útiles para la gestión de sesiones son:

Función	Descripción
Session_id()	Nos devuelve el identificador de la sesión
Session_destroy()	Da por abandonada la sesión eliminando variables e identificador.
Session_unregister('variable')	Abandona una variable sesión

Trabajar con bases de datos en PHP

Una de las principales ventajas que presenta el trabajar con páginas dinámicas es el poder almacenar los contenidos en bases de datos. De esta forma, podemos organizarlos, actualizarlos y buscarlos de una manera mucho más simple.

El lenguaje PHP, ya hemos dicho, ofrece interfaces para el acceso a la mayoría de las bases de datos comerciales y por ODBC a todas las bases de datos posibles en sistemas Microsoft, a partir de las cuales podremos editar el contenido de nuestro sitio con absoluta sencillez.

Esta interacción se realiza, por un lado, a partir de las funciones que PHP nos propone para cada tipo de base de datos y, por otro estableciendo un diálogo a partir de un idioma universal: SQL (Structured Query Language) el cual es común a todas las bases de datos. Este lenguaje resulta, como veremos en el **tutorial de SQL**, muy potente y fácil de aprender.

En este manual de PHP nos limitaremos pues a la utilización las instrucciones SQL básicas que serán aprendidas a medida que explicamos las diferentes formas de actuar sobre una base de datos dejando para el **tutorial de SQL** los aspectos más avanzados.

Como base ejemplo de estos capítulos hemos elegido MySQL, sin duda la base de datos más extendida en combinación con PHP. Su gratuidad, eficiencia y simplicidad la han hecho una buena candidata.

Ya hemos explicado en capítulos anteriores su **instalación** a la vez que hemos presentado los comandos de base que nos pueden permitir abordarla con una relativa facilidad.

En caso de utilizar cualquier otra base compatible, las correcciones a llevar a cabo con respecto a nuestros ejemplos no son excesivamente grandes y la lectura de esos capítulos sigue siendo de gran utilidad.

Una vez instalado MySQL y antes de poder comenzar con nuestros ejemplos, será necesario llevar a cabo las siguientes operaciones:

-Introducidos dentro de MySQL, crearemos la base de datos ejemplo con la siguiente sentencia:

```
create database ejemplo;
```

-Seleccionaremos la base ejemplo como la base a utilizar:

```
use ejemplo
```

-Crearemos a continuación la tabla clientes a partir de la siguiente sentencia:

```
create table clientes (
nombre varchar(100),
telefono varchar(100)
);
```

Ahora ya disponemos de nuestra tabla vacía. Sólo queda comenzar a llenarla con los datos que iremos insertando.

Introducción de nuevos registros

Una vez creada la tabla *clientes* en nuestra base de datos *ejemplo*, el paso siguiente sea llenarla con registros.

Los datos del registro pueden ser recogidos, por ejemplo, a partir de un formulario. Aquí os proponemos un simple documento HTML que recoge los datos y los envía a una página PHP que se encarga de procesarlos:

```
<HTML>
<HEAD>
<TITLE>Insertar.html</TITLE>
</HEAD>
<BODY>
<div align="center">
<h1>Insertar un registro</h1>
<br>
<FORM METHOD="POST" ACTION="insertar.php">
Nombre<br>
<INPUT TYPE="TEXT" NAME="nombre"><br>
Teléfono<br>
<INPUT TYPE="TEXT" NAME="telefono"><br>
<INPUT TYPE="SUBMIT" value="Insertar">
</FORM>
</div>
</BODY>
</HTML>
```

Llegados a la página destino del formulario (*insertar.php*), lo primero que habrá que hacer es establecer un vínculo entre el programa y la base de datos. Esta conexión se lleva a cabo con la función *mysql_connect*. A continuación, deberemos generar una orden de inserción del registro en lenguaje SQL. Esta orden será ejecutada por medio de la función *mysql_db_query*. En esta función especificaremos primeramente la base de datos sobre la que queremos actuar y a continuación introduciremos la sentencia SQL:

```
<HTML>
<HEAD>
<TITLE>Insertar.php</TITLE>
</HEAD>
<BODY>
<?
//Conexion con la base
mysql_connect("localhost","tu_user","tu_password");
//Ejecucion de la sentencia SQL
mysql_db_query("ejemplo","insert into clientes (nombre,telefono) values ('$nombre','$telefono')");
?>
<h1><div align="center">Registro Insertado</div></h1>
<div align="center"><a href="lectura.php">Visualizar el contenido de la base</a></div>
</BODY>
</HTML>
```

Los parametros user y password son definidos por el creador de la base. Es conveniente en un principio, al crear nuestras bases, trabajar sin ellos con lo cual dejaremos las cadenas correspondientes vacias: "".

Además de la propia inserción, el programa avisa de la introducción del registro y ofrece un enlace hacia una página de lectura la cual será comentada a continuación.

No entraremos en la descripción de la orden SQL, para comprender más acerca de cómo introducir registros, refererirse al **tutorial de SQL**.

Selección y lectura de registros

Dentro de una base de datos, organizada por tablas, la selección de una tabla entera o de un cierto numero de registros resulta una operación rutinaria.

Aquí os mostramos una forma bastante clásica de mostrar en pantalla a partir de un bucle los registros seleccionados por una sentencia SQL:

```
<HTML>
<HEAD>
<TITLE>lectura.php</TITLE>
</HEAD>
<BODY>
<h1><div align="center">Lectura de la tabla</div></h1>
<br>
<br>
<?
//Conexion con la base
mysql_connect("localhost","tu_user","tu_password");

//Ejecutamos la sentencia SQL
$result=mysql_db_query("ejemplo","select * from clientes");
?>
<table align="center">
<tr>
<th>Nombre</th>
<th>Teléfono</th>
</tr>
<?
//Mostramos los registros
while ($row=mysql_fetch_array($result))
{
echo '<tr><td>'.$row["nombre"].'</td>';
echo '<td>'.$row["telefono"].'</td></tr>';
}
mysql_free_result($result)
?>
</table>

<div align="center">
<a href="insertar.html">Añadir un nuevo registro</a><br>
<a href="actualizar1.php">Actualizar un registro existente</a><br>
<a href="borrar1.php">Borrar un registro</a><br>
</div>

</BODY>
</HTML>
```

Los pasos a realizar son, en un principio, los vistos para la inserción de un registro: Conexión a la base y ejecución de la sentencia. Esta vez, la información de dicha ejecución será almacenada en una variable (*$result*).

El siguiente paso será plasmar en pantalla la información recogida en $result. Esto lo haremos mediante la función *mysql_fetch_array* que devuelve una variable array con los contenidos de un registro a la vez que se posiciona sobre el siguiente. El bucle *while* nos permite leer e imprimir secuencialmente cada uno de los registros.

La función *mysql_free_result* se encarga de liberar la memoria utilizada para llevar a cabo la consulta. Aunque no es necesaria su utilización, resulta altamente aconsejable.

Actualizacion de un registro

Para mostrar cómo se actualiza un registro presente en nuestra base de datos, vamos a hacerlo a partir de un caso un poco más complejo para que empecemos a familiarizarnos con estas operaciones. Realizaremos un par de scripts que permitan cambiar el numero de teléfono de las distintas personas presentes en nuestra base. El nombre de estas personas, así como el nuevo numero de teléfono, serán recogidos por medio de un formulario.

El archivo del formulario va a ser esta vez un script PHP en el que efectuaremos una llamada a nuestra base de datos para construir un menú desplegable donde aparezcan todos los nombres. La cosa quedaría así:

```
<HTML>
<HEAD>
<TITLE>Actualizar1.php</TITLE>
</HEAD>
<BODY>
<div align="center">
<h1>Actualizar un registro</h1>
<br>
<?
//Conexion con la base
mysql_connect("localhost","tu_user","tu_password");

echo '<FORM METHOD="POST" ACTION="actualizar2.php">Nombre<br>';

//Creamos la sentencia SQL y la ejecutamos
$sSQL="Select nombre From clientes Order By nombre";
$result=mysql_db_query("ejemplo",$sSQL);

echo '<select name="nombre">';

//Generamos el menu desplegable
while ($row=mysql_fetch_array($result))
{echo '<option>'.$row["nombre"];}
?>
</select>
<br>
Teléfono<br>
<INPUT TYPE="TEXT" NAME="telefono"><br>
<INPUT TYPE="SUBMIT" value="Actualizar">
</FORM>
</div>

</BODY>
</HTML>
```

La manera de operar para construir el menú desplegable es la misma que para visualizar la tabla. De nuevo empleamos un bucle *while* en combinación con la función *mysql_fetch_array* lo que nos permite mostrar cada una de las opciones.

El script de actualización será muy parecido al de inserción:

```
<HTML>
<HEAD>
<TITLE>Actualizar2.php</TITLE>
</HEAD>
<BODY>
<?
//Conexion con la base
mysql_connect("localhost","tu_user","tu_password");

//Creamos la sentencia SQL y la ejecutamos
$sSQL="Update Clientes Set telefono='$telefono' Where nombre='$nombre'";
mysql_db_query("ejemplo",$sSQL);
?>

<h1><div align="center">Registro Actualizado</div></h1>
<div align="center"><a href="lectura.php">Visualizar el contenido de la base</a></div>
```

```
</BODY>
</HTML>
```

Borrado de un registro con PHP

Otra de las operaciones elementales que se pueden realizar sobre una base de datos es borrar un registro. Para hacerlo, SQL nos propone sentencias del tipo *Delete*. Veámoslo con un ejemplo aplicado a nuestra agenda. Primero, crearemos un menú desplegable dinámico como para el caso de las actualizaciones:

```
<HTML>
<HEAD>
<TITLE>Borrar1.php</TITLE>
</HEAD>
<BODY>
<div align="center">
<h1>Borrar un registro</h1>
<br>

<?
//Conexion con la base
mysql_connect("localhost","tu_user","tu_password");

echo '<FORM METHOD="POST" ACTION="borrar2.php">Nombre<br>';

//Creamos la sentencia SQL y la ejecutamos
$sSQL="Select nombre From clientes Order By nombre";
$result=mysql_db_query("ejemplo",$sSQL);

echo '<select name="nombre">';

//Mostramos los registros en forma de menú desplegable
while ($row=mysql_fetch_array($result))
{echo '<option>'.$row["nombre"];}
mysql_free_result($result)
?>

</select>
<br>
<INPUT TYPE="SUBMIT" value="Borrar">
</FORM>
</div>

</BODY>
</HTML>
```

El siguiente paso es hacer efectiva la operación a partir de la ejecución de la sentencia SQL que construimos a partir de los datos del formulario:

```
<HTML>
<HEAD>
<TITLE>Borrar2.php</TITLE>
</HEAD>
<BODY>
<?
//Conexion con la base
mysql_connect("localhost","tu_user","tu_password");

//Creamos la sentencia SQL y la ejecutamos
$sSQL="Delete From Clientes Where nombre='$nombre'";
mysql_db_query("ejemplo",$sSQL);
?>

<h1><div align="center">Registro Borrado</div></h1>
<div align="center"><a href="lectura.php">Visualizar el contenido de la base</a></div>

</BODY>
</HTML>
```

Subir una aplicación PHP al servidor

En el pasado me solicitaron que escribiese sobre un tema que hasta ahora no habíamos tocado más que de refilón, que consiste en la puesta en marcha de una aplicación, programada en local, a nuestro servidor de hosting, es decir, en el paso de subir todos los archivos PHP y la base de datos a nuestro espacio en el servidor web contratado en un proveedor de alojamiento.

El tema espero que resulte familiar a muchas de las personas que leen nuestros artículos, ya que probablemente hayan tenido que pasar por esa etapa en alguna ocasión, aunque pretendo dar algunas claves y truquillos que pueden ayudar a todos, tengan o no experiencia en este asunto.

Subir los archivos

Nuestro servidor web debe tener un directorio para la publicación de las páginas web. Ese sería el lugar donde hay que subir los archivos .php.

Dependiendo del proveedor con el que trabajemos, el directorio de publicación puede variar. Generalmente, cuando contratamos un alojamiento, nos proporcionan una cuenta de FTP con la que conectarnos al servidor web y transferir los archivos de nuestro sitio, además de unos datos para la conexión, que serán el nombre del servidor y el usuario y contraseña para el acceso al FTP.

> **Referencia:** por si alguien no sabe lo que es el FTP, hablamos más sobre ello en el manual de Publicar en Internet, concretamente en el artículo Subir los archivos al servidor.

Al conectarnos al servidor con los datos del FTP, que deben ser proporcionados por nuestro proveedor, accederemos a un directorio. Este directorio podría ser el de publicación, aunque generalmente no es así, sino que suele ser un subdirectorio llamado "HTML" o "docs" o algo similar, que cuelga del directorio de inicio en nuestra conexión FTP. Como decía, este directorio puede tener nombres distintos en proveedores distintos, aunque, en cualquier caso, con una simple pregunta a nuestro proveedor resolveremos esa duda.

Los archivos se deben subir al directorio de publicación, o a cualquier subdirectorio de este. En definitiva, los tendremos que alojar por ahí dentro y para acceder a ellos bastaría con escribir el nombre del dominio o URL de nuestro alojamiento, seguido del nombre del archivo. Si tuviésemos un archivo llamado hola.php y nuestro alojamiento se ha contratado para el dominio www.midominio.com, deberíamos subir ese archivo al directorio de publicación y accederíamos al archivo escribiendo:

http://www.midominio.com/hola.php

Si creamos subdirectorios dentro del directorio de publicación podremos acceder a ellos escribiendo el nombre del dominio o URL de nuestro alojamiento, seguido del nombre del directorio y el nombre del archivo. Por ejemplo, si creamos un subdirectorio llamado paginas y tenemos dentro un archivo llamado pag1.php, podríamos acceder a él de la siguiente manera.

http://www.midominio.com/paginas/pag1.php

> **Referencia:** hay otro concepto interesante que deberíamos conocer llegados a este punto, que es el "documento por defecto". Éste no es más que el archivo que se envía al navegador si en la URL accedida no se especificaba ningún archivo. Suele llamarse index.html o index.php (o index.asp si nuestro servidor soporta programación en ASP), aunque puede variar de un proveedor a otro. Hablamos más sobre el documento por defecto en nuestro manual de Publicar en Internet.

Colocar los archivos PHP fuera del directorio de publicación

Por decir algo más sobre el tema de colocar los archivos, quería señalar que cualquier cosa que pongamos fuera del directorio de publicación no podrá ser accedida a través del navegador. Es decir, si creamos un directorio que se llame funciones_php en el mismo nivel que el directorio de publicación (fuera del directorio de publicación) no podremos acceder con el explorador a los archivos que coloquemos dentro de ninguna de las maneras. Esto es así porque la URL de inicio de nuestro alojamiento corresponde con ese directorio y no podemos movernos hacia debajo de ese directorio con las URLs, que son la manera de especificar al navegador los recursos a los que se quiere acceder.

> **Referencia:** Ya se explicó lo que era el directorio de publicación en el capítulo anterior sobre Subir archivos PHP al servidor.

No sería posible salir del directorio de publicación con una URL como esta, por mucho que utilicemos el operador .. (que sirve para acceder al directorio padre).

http://www.midominio.com/../funciones_php/archivo_inalcanzable.php

Sin embargo, colocar algunos contenidos fuera del directorio de publicación puede ser muy útil. Por ejemplo, podríamos colocar allí copias de seguridad de algunos archivos o documentos que simplemente queremos guardar en el servidor para acceder a ellos desde cualquier parte y con nuestro programa de FTP.

Hay otra utilidad más interesante sobre colocar archivos fuera del directorio de publicación. Se trata de que muchas veces utilizamos en nuestros programas trozos de código repetidamente, por ejemplo, para abrir y cerrar bases de datos, para mostrar la cabecera de nuestro portal, para comprobar que un email escrito en un formulario es correcto, etc. Es muy útil separar estos trozos de código en un archivo a parte y llamar a este archivo con las funciones PHP include() o require(). Así, si un día modificamos la cabecera de nuestro portal, sólo lo tendremos que modificar en un archivo, o, si cambia la base de datos que utilizamos sólo tendríamos que modificar el archivo que hace la conexión a la base de datos una vez, en lugar de ir cambiándolo en todas las páginas PHP que abrían las bases de datos.

Estos archivos no son páginas independientes, sino trozos. Seguramente, si los ejecutamos por separado no mostrarían ningún resultado válido, incluso podrían dar mensajes de error. Por esta razón merece la pena colocarlos en un lugar donde nadie pueda tener acceso: fuera del directorio de publicación. Con PHP si que podremos acceder a ese directorio para incluir esos archivos. Solamente deberíamos utilizar las funciones PHP include() o require() indicando la ruta para acceder a los archivos.

En el caso de que tengamos una página llamada hola.php en el directorio de publicación y un archivo, que se llama abre_base_datos.php, en el directorio funciones_php, que está fuera del directorio de publicación, si quisiéramos acceder (desde hola.php) al archivo que abre la base de datos lo haríamos así.

```
include("../funciones_php/abre_base_datos.php")
```

Desde PHP sí que podemos acceder a los archivos que se encuentran fuera del directorio de publicación. Para ello especificamos la ruta adecuada, en la que utilizamos el operador .. para bajar al directorio padre.

Nada más que decir sobre la colocación de los archivos: una vez situados en el directorio de publicación se podrá acceder a ellos con nuestro navegador y se deberían ejecutar perfectamente. Aunque cabe señalar que, tanto PHP como el servidor donde trabajemos, pueden tener configuraciones distintas y puede que algún detalle de la programación de nuestras páginas no funcione correctamente.

Por ejemplo, nuestro PHP puede declarar o no automáticamente las variables que llegan a través de un formulario. Si en local si que estaba configurado para hacer esto y en remoto no, deberíamos localizar los lugares donde recogemos las variables y utilizar las variables de entorno correctas (mirar artículo sobre Procesar variables de formularios y los comentarios al pie para saber más de esta posible fuente de errores).

Aunque este no es un caso habitual, podemos ponernos en contacto con nuestro proveedor de alojamiento para ver si pueden ayudarnos configurando el sistema o indicando los pasos a seguir para solventar en nuestros scripts el asunto.

Subir una base de datos al sevidor de Internet

Aparte de los archivos de la página, debemos subir la base de datos con la que tenemos que trabajar. Las bases de datos con las que trabaja PHP son muy variadas y en distintos casos podemos utilizar una u otra, por lo que los modos de subir la base de datos también pueden variar.

> **Referencia:** Este artículo y los sucesivos, que tratan sobre subir una base de datos MySQL al servidor, se engloban tanto dentro del Manual de PHP como del Taller de MySQL. Por ello, será importante disponer de conocimientos de ambas tecnologías para entender y aprovechar estas explicaciones.

Es muy corriente que nuestro proveedor de hosting ofrezca junto con PHP la base de datos MySQL, así que las notas para subir esa base de datos al servidor de este artículo van encaminadas a ofrecer soluciones para esa base de datos.

La base de datos MySQL no se puede subir por FTP, como que se hacía con los archivos del código PHP.

Para subirla tendremos que utilizar otros mecanismos. Voy a distinguir entre tres casos distintos en los que nos podríamos encontrar en este momento:

1. **La base de datos que pretendemos subir está vacía**. Tan sólo hemos creado las tablas, pero no hemos introducido datos en ellas o, a lo sumo, tienen algún dato que hemos introducido de pruebas.
2. **La base de datos que queremos subir está completa y es una base de datos MySQL**. En este caso tenemos creada la base de datos en local y con toda la información dentro y, por supuesto, queremos que esa información quede también en la base de datos remota.
3. La base de datos está **completa** (como el caso anterior), pero **no es una base de datos MySQL**. En este caso estaríamos haciendo una migración de la base de datos de un sistema gestor a otro.

Veremos los tres casos por separado en adelante, aunque, antes de ello, vamos a mostrar unas herramientas que nos servirán de mucha ayuda para la administración de cualquier base de datos remota.

Las herramientas en concreto se relatan en el manual Taller de MySQL, son las siguientes:

- **PhpMyAdmin**. Una aplicación creada en PHP que podemos instalar en nuestro espacio de alojamiento para administrar la base de datos.
- **Mysql Control Center** (en adelante MyCC). Una aplicación Windows que permite contectarse a múltiples bases de datos MySQL, que se encuentren en local o en remoto.
- **Access**. También permite administrar una base de datos MySQL conectada en local o en remoto. En este caso se utiliza una interfaz que muchos ya conocen, como es Access, para administrar una base de datos que nada tiene que ver con dicho programa.

En los tres casos lo que nos permite realizar el software de administración son tareas sobre la base de datos de todo tipo, como pueden ser crear tablas, modificarlas, insertar datos, borrarlos, editarlos. Modificar o borrar tablas o campos de las mismas, etc.

La elección de una herramieta o de otra pasa por los recursos que nos permitan utilizar en nuestro proveedor. Básicamente, lo que nos puede decantar a una opción u otra, es si permiten o no conectar de manera remota la base de datos MySQL. Conozco alojamientos donde se permite esa conexión remota y donde no.

Si no permiten conectarnos remotamente nos decantaremos por PhpMyAdmin, pues es una aplicación PHP que se conecta en local y a la que se accede desde una página web y eso lo permiten todos los proveedores, incluso hay muchos que tienen instalado ya este software para administrar las bases de datos.

En caso de que sí nos permitan conectarnos remotamente con la base de datos, eligiremos MyCC o Access, que son aplicaciones Windows mucho más potentes y rápidas que las que utilizan interfaz web, como PhpMyAdmin. Es preferible utilizar MyCC porque está especialmente desarrollado para conectar y operar con bases de datos MySQL.

Subir base de datos MySQL vacía al servidor

Es muy normal que hayamos diseñado una base de datos para nuestro proyecto desde 0, definiendo las distintas entidades de nuestro modelo de datos, junto con sus campos y sus tipos.

En estos casos lo más probable es que la base de datos esté vacía, o bien contenga datos que hayamos introducido a modo de prueba y que no queramos conservar cuando subamos la aplicación a Internet.

La opción más interesante entonces podría ser crear otra vez las tablas que tenemos en local en la base de datos remota. Para ello tenemos dos posibilidades.

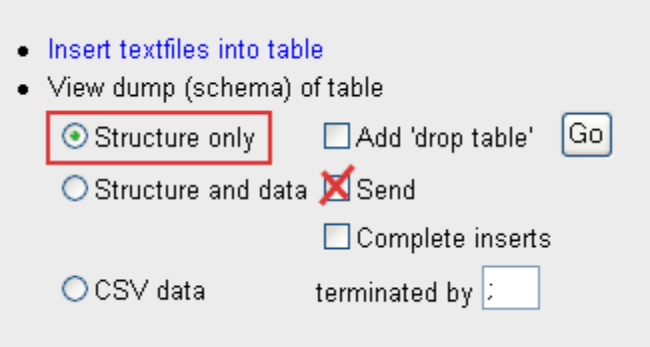
a) Si tenemos pocas tablas y bastante sencillas

Las podemos crear en remoto con alguna herramienta como PhpMyAdmin o MyCC.

b) Si tiene muchas tablas y/o muy complicadas

La recomendación sería hacer un backup de la estructura en local y restaurarla en remoto. Esto nos evitará tener que volver a crear todas las tablas y definir todos sus campos y sus tipos. Puede ser un poco más complicado pero sin duda nos ahorrará tiempo.

Para hacer el backup de la estructura en local podemos utilizar alguna herramienta como PhpMyAdmin, o bien utilizar el comando mysqldump desde línea de comandos de MS-DOS.



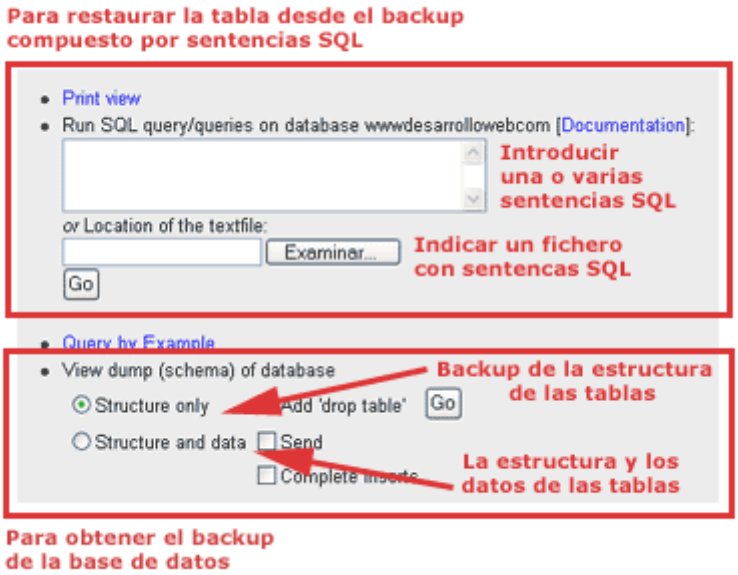
Herramienta de backup de PhpMyAdmin. Está marcada la opción de extraer solamente la estructura de las tablas. Si marcamos además la casilla "Send", nuestro navegador se descargará el backup en un fichero de texto. Si no lo pulsamos simplemente se visualizará.

Lo que tenemos que hacer en este caso es un backup de la estructura de la base de datos, es decir, los "create tables" o sentencias SQL para crear las tablas. Sería un montón de sentencias con esta forma:

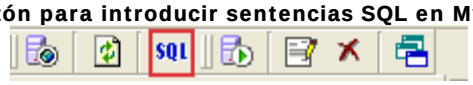
```
# --------------------------------------------------------
#
# Table structure for table 'comentario'
#

CREATE TABLE comentario (
   id_comentario int(5) unsigned NOT NULL auto_increment,
   id_articulo int(4) DEFAULT '0' NOT NULL,
   comentario text NOT NULL,
   fecha int(14) unsigned DEFAULT '0' NOT NULL,
   revisado tinyint(1) DEFAULT '0' NOT NULL,
   nombre_comentario varchar(100) DEFAULT 'Nombre no especificado' NOT NULL,
   email_comentario varchar(100) DEFAULT 'Email sin especificar' NOT NULL,
   tipo tinyint(1) unsigned DEFAULT '1' NOT NULL,
   PRIMARY KEY (id_comentario)
);
```

Para restaurar estas sentencias tenemos opciones tanto dentro de PhpMyAdmin como de MyCC. En ambos casos lo que tenemos que hacer es ejecutar estas sentencias en el servidor MySQL remoto. En PhpMyAdmin tenemos un campo para introducir sentencias SQL y también otro campo para seleccionar un archivo de texto con todas las sentencias SQL, para ejecutarlas una detrás de otra. En MyCC tenemos un botón que nos permite abrir una consola donde introducir una o varias sentencias SQL y ejecutarlas.

Herramienta de backup y restauración de PhpMyAdmin



Botón para introducir sentencias SQL en MyCC



Repetimos, esto sólo nos servirá para subir la estructura de la base de datos y no los datos que contenga. Si deseamos subir también la información de la base de datos entonces debemos utilizar otras estrategias, relatadas próximamente.

Subir una base de datos MySQL con la estructura y los datos

Si la base de datos que deseamos subir está llena de información y deseamos que se conserve una vez subida la base de datos a remoto, tenemos que realizar un backup de la base de datos y restaurarlo en remoto.

> **Nota:** Estas recomendaciones están pensadas para subir una base de datos MySQL que podamos tener en local a una base de datos MySQL que hayamos contratado en remoto. Si la base origen no es MySQL estaríamos hablando de una migración de bases de datos, pero esto lo veremos en un artículo más adelante.

En este caso el procedimiento sería muy parecido al de subir una base de datos vacía, relatado anteriormente, con la salvedad de que ahora debemos extraer no solo la estructura de la base de datos, sino también los registros que contiene.

Para ello podemos utilizar mysqldump, según se relata en este artículo, o bien PhpMyAdmin, seleccionando la opción que indica que el backup contenga la estructura y los datos (Structure and data en versiones en inglés).

La estructura y los datos vendrán en un fichero de texto con una serie de sentencias SQL para crear las tablas y los insert necesarios para introducir cada uno de los datos.

Para restaurar la base de datos lo haremos tal como se ha relatado para el caso de que la base de datos estuviera vacía, con la ayuda de una instalación de PhpMyAdmin en remoto o un MyCC que se conecte a la base de datos contratada en el servidor de Internet.

Si tenemos problemas para subir el fichero de backup de la base de datos es posible que en nuestro proveedor de alojamiento nos pueda ayudar a subir el fichero y restaurarlo. Como el proveedor dispone de los servidores en sus propias instalaciones, tiene muchas más posibilidades que nosotros para trabajar con las bases de datos, sin temor a que las lentas comunicaciones por Internet arrojen errores en la restauración de los datos.

Si nuestro proveedor no puede ayudarnos, seguramente disponga y nos indique algún mecanismo para realizar la tarea sin lugar a errores. Puede ocurrinos con algún proveedor que nos diga que se encarga de todo pero nos exija el pago de las horas de trabajo del informático que va a restaurar el backup de la

base de datos. Si no pone facilidades ni siquiera en esto posiblemente sea mejor ir pidiéndoles que nos devuelvan el dinero invertido porque su servicio no sería muy bueno.

Migrar una base de datos a MySQL

El último caso en el que nos podemos encontrar a la hora de subir una base de datos a nuestro proveedor de alojamiento es que la base de datos la tengamos creada en local, pero en un sistema gestor distinto del que vamos a utilizar en remoto. En remoto suponemos siempre que vamos a utilizar la base de datos MySQL. En local podríamos disponer de una base de datos Access, SQL Server o de otro sistema de base de datos.

El proceso de la migración puede ser bastante complejo y, como hay tantas bases de datos distintas, difícil de dar una receta que funcione en todos los casos. Además, aparte de la dificultad de transferir la información entre los dos sistemas gestores de base de datos, también nos influirá mucho en la complejidad del problema el tipo de los datos de las tablas que estamos utilizando. Por ejemplo, las fechas, los campos numéricos con decimales o los boleanos pueden dar problemas al pasar de un sistema a otro porque pueden almacenarse de maneras distintas o, en el caso de los números, con una precisión distinta.

Recomendaciones para migrar de Access a MySQL

Si nuestra base de datos anterior estaba construida en Access lo tenemos bastante fácil, gracias a que MySQL dispone de un driver ODBC para sistemas Windows, que nos permite conectar Access con el propio MySQL y pasar información fácilmente.

Este tema está relatado en el artículo Exportar datos de MySQL a Access, aunque hay que indicar que si deseamos hacer una exportación desde Access en local a MySQL en remoto puede haber problemas porque no todos los alojadores permiten las conexiones en remoto con la base de datos. Si no tenemos disponible una conexión en remoto con nuestro servidor de bases de datos vamos a tener que cambiar la estrategia un poco.

La idea en este último caso es instalar MySQL en local y realizar la migración desde Access en local a MySQL en local y luego podríamos hacer un backup de la base de datos local y subirla a remoto, tal y como se ha relatado antes.

Recomendaciones para migrar desde SQL Server a MySQL

La verdad es que no he tenido este caso nunca, pero hay que decir que Access también nos puede ayudar en este caso. Access permite seleccionar una base de datos SQL Server y trabajar desde la propia interfaz de Access. La idea es que Access también permite trabajar con MySQL y posiblemente haciendo un puente entre estos dos sistemas gestores podemos exportar datos de SQL Server a MySQL.

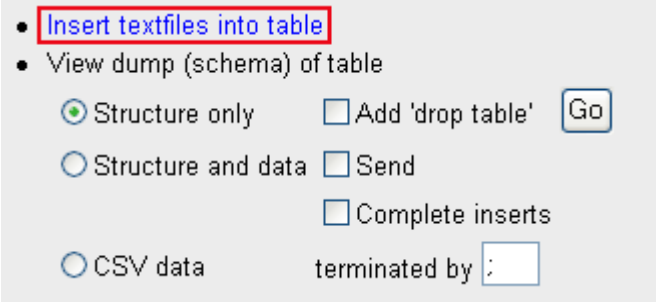
Lo que es seguro que utilizando el propio Access de puente podríamos realizar el trabajo. Primero exportando de SQL Server a Acess y luego desde Access a MySQL.

Otras bases de datos u otras técnicas

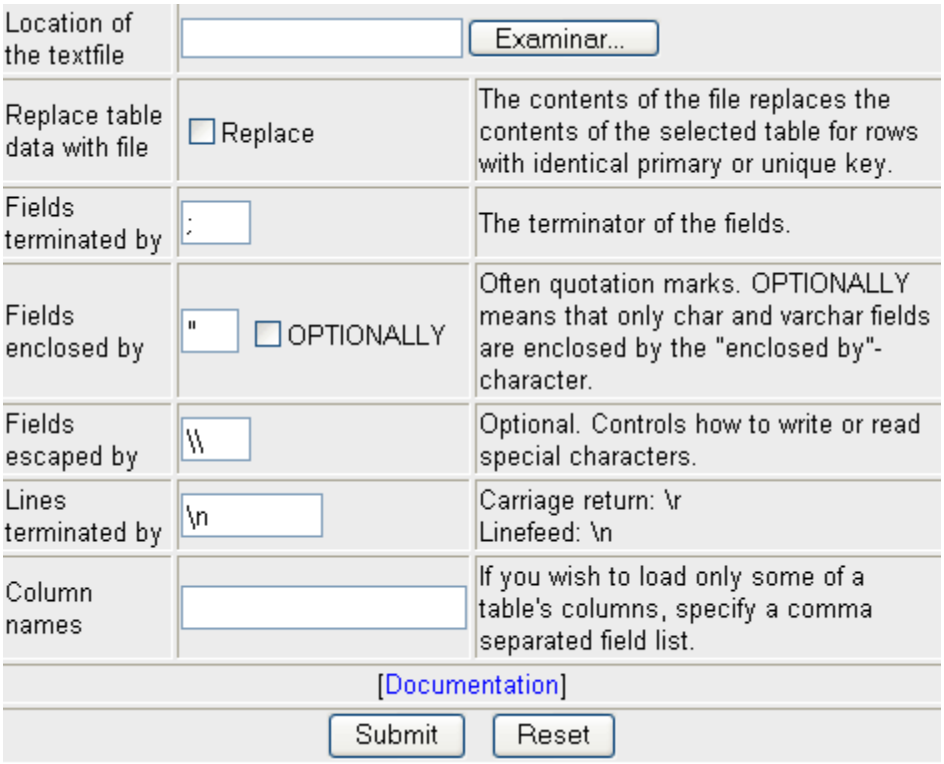
Si la base de datos origen dispone de un driver ODBC no habrá (en teoría) problema para conectarla con Access, de manera similar a como se conecta con MySQL. Entonces podríamos utilizar Access para exportar los datos, porque desde allí se podrían acceder a los dos sistemas gestores de bases de datos.

Si no tenemos Access, o la base de datos original no tiene driver ODBC, o bien no nos funciona correctamente el proceso y no sabemos cómo arreglarlo, otra posibilidad es exportar los datos a ficheros de texto, separados por comas o algo parecido. Muchas bases de datos tienen herramientas para exportar los datos de las tablas a ficheros de texto, los cuales se pueden luego introducir en nuestro sistema gestor destino (MySQL) con la ayuda de alguna herramienta como PhpMyAdmin.

Para ello, en la página de propiedades de la tabla encontraremos una opción para hacer el backup de la tabla y para introducir ficheros de texto dentro de una tabla (Insert textfiles into table en inglés).



Accediendo a ese enlace podremos ver un formulario donde introducir las características del fichero de texto, como el carácter utilizado como separador de campos, o el terminador de líneas, etc, junto con el propio archivo con los datos, y PhpMyAdmin se encargará de todo el trabajo de incluir esos datos en la tabla.



Como se habrá supuesto, es necesario tener creada la tabla en remoto para que podamos introducirle los datos del fichero de texto.

Cambios de un formato de datos a otro

Toda la migración tiene que tener en cuenta muy especialmente, como ya se señaló, las maneras que tenga cada base de datos de guardar la información, es decir, del formato de sus tipos de datos. Tenemos que contar siempre con la posible necesidad de transformar algunos datos como pueden ser los campos boleanos, fechas, campos memo (texto con longitud indeterminada), etc, que pueden almacenarse de maneras distintas en cada uno de los sistemas gestores, origen y destino.

En algunos casos posiblemente tengamos que realizar algún script que realice los cambios necesarios en los datos. Por ejemplo puede ser para localizar los valores boleanos guardados como true / false a valores enteros 0 / 1, que es como se guarda en MySQL. También las fechas pueden sufrir cambios de formato, mientras que en Access aparecen en castellano (dd/mm/aaaa) en MySQL aparecen en el formato aaaa-mm-dd. PHP puede ayudarnos en la tarea de hacer este script, también Visual Basic Script para Access puede hacer estas tareas complejas y el propio lenguaje SQL, a base de sentencias dirigidas contra la base de datos, puede servir para algunas acciones sencillas.

Capitulo 11
Programación en JavaScript

Descubre el lenguaje dinámico de lado cliente por excelencia. Aprende a crear páginas webs con vida propia con nuestro manual de Javascript.

Introducción a Javascript

Javascript es un lenguaje de programación utilizado para crear pequeños programitas encargados de realizar acciones dentro del ámbito de una página web. Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Javascript es el siguiente paso, después del HTML, que puede dar un programador de la web que decida mejorar sus páginas y la potencia de sus proyectos. Es un lenguaje de programación bastante sencillo y pensado para hacer las cosas con rapidez, a veces con ligereza. Incluso las personas que no tengan una experiencia previa en la programación podrán aprender este lenguaje con facilidad y utilizarlo en toda su potencia con sólo un poco de práctica.

Entre las acciones típicas que se pueden realizar en Javascript tenemos dos vertientes. Por un lado los efectos especiales sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo. Por el otro, javascript nos permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que podemos crear páginas interactivas con programas como calculadoras, agendas, o tablas de cálculo.

Javascript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Toda esta potencia de Javascript se pone a disposición del programador, que se convierte en el verdadero dueño y controlador de cada cosa que ocurre en la página.

En este libro vamos a tratar de acercarnos a este lenguaje en profundidad y conocer todos sus secretos y métodos de trabajo. Al final del libro seremos capaces de controlar la página web y discernir el mejor método para atacar los problemas u objetivos que nos hayamos planeado.

Algo de historia

En Internet se han creado multitud de servicios para realizar muchos tipos de comunicaciones, como correo, charlas, búsquedas de información, etc. Pero ninguno de estos servicios se ha desarrollado tanto como el Web. Si estamos leyendo estas líneas no vamos a necesitar ninguna explicación de lo que es el web, pero si podemos hablar un poco sobre cómo se ha ido desarrollando con el paso de los años.

El web es un sistema Hipertexto, una cantidad desmesurada de textos que contienen enlaces que relacionan cada una de las unidades básicas donde podemos encontrar información, las páginas web. En un principio, para diseñar este sistema de páginas con enlaces se pensó en un lenguaje que permitiese presentar cada una de estas informaciones junto con unos pequeños estilos, este lenguaje fue el HTML, que luego se vería que no cumplió todos los objetivos para los que fue diseñado, pero eso es otro tema.

El caso es que HTML no es suficiente para realizar todas las acciones que se pueden llegar a necesitar en una página web. Esto es debido a que conforme fue creciendo el web y sus distintos usos se fueron complicando las páginas y las acciones que se querían realizar a través de ellas. El HTLM se había quedado corto para definir todas estas nuevas funcionalidades, ya que sólo sirve para presentar el texto en un página, definir su estilo y poco más.

El primer ayudante para cubrir las necesidades que estaban surgiendo fue Java, a través de la tecnología de los Applets, que son pequeños programas que se incrustan en las páginas web y que pueden realizar las acciones asociadas a los programas de propósito general. La programación de Applets fue un gran avance y Netscape, por aquel entonces el navegador más popular, había roto la primera barrera del HTML al hacer posible la programación dentro de las páginas web. No cabe duda que la aparición de los Applets supuso un gran avance en la historia del web, pero no ha sido una tecnología definitiva y muchas otras han seguido implementando el camino que comenzó con ellos.

Llega Javascript:

Netscape, después de hacer sus navegadores compatibles con los applets, comenzó a desarrollar un lenguaje de programación al que llamó LiveScript que permitiese crear pequeños programas en las

páginas y que fuese mucho más sencillo de utilizar que Java. De modo que el primer Javascript se llamo LiveScript, pero no duró mucho ese nombre, pues antes de lanzar la primera versión del producto se forjó una alianza con Sun Microsystems, creador de Java, para desarrollar en conjunto ese nuevo lenguaje.

La alianza hizo que Javascript se diseñara como un hermano pequeño de Java, solamente útil dentro de las páginas web y mucho más fácil de utilizar, de modo que cualquier persona, sin conocimientos de programación pudiese adentrase en el lenguaje y utilizarlo a sus anchas. Además, para programar Javascript no es necesario un kit de desarrollo, ni compilar los scripts, ni realizarlos en ficheros externos al código HTML, como ocurría con los applets.

Netscape 2.0 fue el primer navegador que entendía Javascript y su estela fue seguida por los navegadores de la compañía Microsoft a partir de la versión 3.0.

Diferencias entre Java y Javascript

Queremos que quede claro que **Javascript no tiene nada que ver con Java**, salvo en sus orígenes, como se ha podido leer hace unas líneas. Actualmente son productos totalmente distintos y no guardan entre si más relación que la sintaxis idéntica y poco más. Algunas diferencias entre estos dos lenguajes son las siguientes:

- **Compilador**. Para programar en Java necesitamos un Kit de desarrollo y un compilador. Sin embargo, Javascript no es un lenguaje que necesite que sus programas se compilen, sino que éstos se interpretan por parte del navegador cuando éste lee la página.

- **Orientado a objetos**. Java es un lenguaje de programación orientado a objetos. (Más tarde veremos que quiere decir orientado a objetos, para el que no lo sepa todavía) Javascript no es orientado a objetos, esto quiere decir que podremos programar sin necesidad de crear clases, tal como se realiza en los lenguajes de programación estructurada como C o Pascal.

- **Propósito**. Java es mucho más potente que Javascript, esto es debido a que Java es un lenguaje de propósito general, con el que se pueden hacer aplicaciones de lo más variado, sin embargo, con Javascript sólo podemos escribir programas para que se ejecuten en páginas web.

- **Estructuras fuertes**. Java es un lenguaje de programación fuertemente tipado, esto quiere decir que al declarar una variable tendremos que indicar su tipo y no podrá cambiar de un tipo a otro automáticamente. Por su parte Javascript no tiene esta característica, y podemos meter en una variable la información que deseemos, independientemente del tipo de ésta. Además, podremos cambiar el tipo de información de una varible cuando queramos.

- **Otras características**. Como vemos Java es mucho más complejo, aunque también más potente, robusto y seguro. Tiene más funcionalidades que Javascript y las diferencias que los separan son lo suficientemente importantes como para distinguirlos fácilmente.

Antes de empezar

Previamente a comenzar a utilizar Javascript podemos hacernos una idea más concreta de las posibles aplicaciones de este lenguaje así como las herramientas que necesitamos para ponernos manos a la obra.

Usos de Javascript

Veamos brevemente algunos usos de este lenguaje que podemos encontrar en el web para hacernos una idea de las posibilidades que tiene.

Para empezar, podemos ver páginas como la página de SEAT, llena de efectos super interesantes sobre Javascript, que llegan a asemejarse a la tecnología Flash. En esta misma vertiente de uso de Javascript podemos encontrar muchas páginas, por ejemplo la página Guiarte Multimedia, que realiza una animación Javascript para presentar a la empresa dueña de la página web.

Por otro lado, podemos encontrar dentro de Internet muchas aplicaciones de Javascript mucho más serias, que hacen que una página web se convierta en un verdadero programa interactivo de gestión de cualquier recurso. Por ejemplo podemos ver una página que consiste en un test de preguntas, que recoge los resultados y los envía a su evaluador. Esta página fue diseñada en su día por el escritor de este libro. La direccion es http://www.geocities.com/Athens/Oracle/3391/.También se pueden ver más

ejemplos de estos dentro de cualquier página un poco compleja, si nos pasamos por un sitio que tenga una calculadora o un convertidor de divisas, veremos que en muchos casos se han realizado con Javascript.

En realidad es mucho más habitual encontrar Javascript para realizar efectos simples sobre páginas web, o no tan simples, como pueden ser rollovers (que cambie una imagen al pasar el ratón por encima), navegadores desplegables, apertura de ventanas secundarias, etc. Nos atrevemos a decir que este lenguaje es realmente útil para estos casos, pues estos típicos efectos tienen la complejidad justa para ser implementados en cuestión de minutos sin posibilidad de errores. Las páginas de DesarrolloWeb son un ejemplo de páginas que utilizan Javascript para realizar multitud de acciones sin que estas sean demasiado complicadas, que carguen la página o que den lugar a errores en distintas plataformas.

Qué necesitas

Para programar en Javascript necesitamos básicamente lo mismo que para programar páginas web con HTML. Un editor de textos y un navegador compatible con Javascript. Un usuario de Windows posee de salida todo lo necesario para poder programar en Javascript, puesto que dispone dentro de su instalación típica de sistema operativo, de un editor de textos, el Bloc de notas, y de un navegador: Internet Explorer.

Usuarios de otros sistemas pueden encontrar en Internet fácilmente las herramientas necesarias para comenzar en páginas de descarga de software como Tucows.

Permitidme una recomendación con respecto al editor de textos. Se trata de que, aunque el Bloc de Notas es suficiente para empezar, tal vez sea muy útil contar con otros programas que nos ofrecen mejores prestaciones a la hora de escribir las líneas de código. Estos editores avanzados tienen algunas ventajas como que colorean los códigos de nuestros scripts, nos permiten trabajar con varios documentos simultáneamente, tienen ayudas, etc. Entre otros queremos destacar el Home Site o UltraEdit.

Versiones de navegadores y de Javascript

También resulta apropiado introducir las distintas versiones de Javascript que existen y que han evolucionado en conjunto con las versiones de navegadores. El lenguaje ha ido avanzando durante sus años de vida e incrementando sus capacidades. En un principio podía realizar muchas cosas en la página web, pero tenía pocas instrucciones para crear efectos especiales. Con el tiempo también el HTML ha avanzado y se han creado nuevas características como las capas, que permiten tratar y maquetar los documentos de manera distinta. Javascript ha avanzado también y para manejar todas estas nuevas características se han creado nuevas instrucciones y recursos. Para resumir vamos a comentar las distintas versiones de Javascript:

- **Javascript 1**: nació con el Netscape 2.0 y soportaba gran cantidad de instrucciones y funciones, casi todas las que existen ahora ya se introdujeron en el primer estandar.
- **Javascript 1.1**: Es la versión de Javascript que se diseñó con la llegada de los navegadores 3.0. Implementaba poco más que su anterior versión, como por ejemplo el tratamiento de imágenes dinámicamente y la creación de arrays.
- **Javascript 1.2**: La versión de los navegadores 4.0. Esta tiene como desventaja que es un poco distinta en plataformas Microsoft y Netscape, ya que ambos navegadores crecieron de distinto modo y estaban en plena lucha por el mercado.
- **Javascript 1.3**: Versión que implementan los navegadores 5.0. En esta versión se han limado algunas diferencias y asperezas entre los dos navegadores.
- **Javascript 1.5**: Versión actual, en el momento de escribir estas líneas, que implementa Netscape 6.
- Por su parte, **Microsoft** también ha evolucionado hasta presentar su **versión 5.5 de JScript** (así llaman al javascript utilizado por los navegadores de Microsoft).

Efectos rápidos con Javascript

Antes de meternos en materia podemos ver una serie de efectos rápidos que se pueden programar con Javascript. Esto nos puede hacer una idea más clara de las capacidades y potencia del lenguaje que nos vendrán bien para tener una idea más exacta de lo que es Javascript a la hora de recorrer los siguientes capítulos.

Abrir una ventana secundaria

Primero vamos a ver que con una línea de Javascript podemos hacer cosas bastante atractivas. Por ejemplo podemos ver cómo abrir una ventana secundaria sin barras de menús que muestre el buscador Google. El código sería el siguiente.

```
<script>
window.open("http://www.google.com","","width=550,height=420,menubar=no")
</script>
```

Un mensaje de bienvenida

Podemos mostrar una caja de texto emergente al terminarse de cargar la portada de nuestro sitio web, que podría dar la bienvenida a los visitantes.

```
<script>
window.alert("Bienvenido a mi sitio web. Gracias...")
</script>
```

Fecha actual

Veamos ahora un sencillo script para mostrar la fecha de hoy. A veces es muy interesante mostrarla en las webs para dar un efecto de que la página está al "al día", es decir, está actualizada.

```
<script> document.write(new Date()) </script>
```

Estas líneas deberían introducirse dentro del cuerpo de la página en el lugar donde queramos que aparezca la fecha de última actualización. Podemos ver el ejemplo en marcha aquí.

> **Nota:** Un detalle a destacar es que la fecha aparece en un formato un poco raro, indicando también la hora y otros atributos de la misma, pero ya aprenderemos a obtener exactamente lo que deseemos en el formato correcto.

Botón de volver

Otro ejemplo rápido se puede ver a continuación. Se trata de un botón para volver hacia atrás, como el que tenemos en la barra de herramientas del navegador. Ahora veremos una línea de código que mezcla HTML y Javascript para crear este botón que muestra la página anterior en el historial, si es que la hubiera.

```
<input type=button value=Atrás onclick="history.go(-1)">
```

El botón sería parecido al siguiente. Podemos pulsarlo para ver su funcionamiento (debería llevarnos a la página anterior).

Como diferencia con los ejemplos anteriores, hay que destacar que en este caso la instrucción Javascript se encuentra dentro de un atributo de HTML, onclick, que indica que esa instrucción se tiene que ejecutar como respuesta a la pulsación del botón.

Se ha podido comprobar la facilidad con la que se pueden realizar algunas acciones interesantes, existirían muchas otras muestras que nos reservamos para capítulos posteriores.

El lenguaje Javascript

En esta parte del libro vamos a conocer la manera de trabajar con Javascript, como incluir scripts y ser compatible con todos los navegadores. Muchas ideas del funcionamiento de Javascript ya se han descrito en capítulos anteriores, pero con el objetivo de no dejarnos nada en el tintero vamos a tratar de acaparar a partir de aquí todos los datos importantes de este lenguaje.

Javascript se escribe en el documento HTML

Lo más importante y básico que podemos destacar en este momento es que la programación de Javascript se realiza dentro del propio documento HTML. Esto quiere decir que en la página se mezclan los dos lenguajes de programación, y para que estos dos lenguajes se puedan mezclar sin problemas se han de incluir unos delimitadores que separan las etiquetas HTML de las instrucciones Javascript. Estos delimitadores son las etiquetas <SCRIPT> y </SCRIPT>. Todo el código Javascript que pongamos en la página ha de ser introducido entre estas dos etiquetas.

En una misma página podemos introducir varios scripts, cada uno que podría introducirse dentro de

unas etiquetas <SCRIPT> distintas. La colocación de estos scripts es indiferente, en un principio nos da igual donde colocarlos, pero en determinados casos esta colocación si que será muy importante. En cada caso, y llegado el momento se informará de ello convenientemente.

También se puede escribir Javascript dentro de determinados atributos de la página, como el atributo *onclick*. Estos atributos están relacionados con las acciones del usuario y se llaman manejadores de eventos.

Vamos a ver en el siguiente capítulo com más detenidamente estas dos maneras de escribir scripts, que tienen como diferencia principal el momento en que se ejecutan las sentencias.

Maneras de ejecutar scripts

Existen **dos maneras de ejecutar scripts** en la página. La primera de estas maneras se trata de ejecución directa de scripts, la segunda es una ejecución como respuesta a la acción de un usuario. Veremos ahora cada una de ellas.

Ejecución directa

Es el método de ejecutar scripts más básico. En este caso se incluyen las instrucciones dentro de la etiqueta <SCRIPT>, tal como hemos comentado anteriormente. Cuando el navegador lee la página y encuentra un script va interpretando las líneas de código y las va ejecutando una después de otra. Llamamos a esta manera ejecución directa pues cuando se lee la página se ejecutan directamente los scripts.

Este método será el que utilicemos preferentemente en la mayoría de los ejemplos de este libro.

Respuesta a un evento

Es la otra manera de ejecutar scripts, pero antes de verla debemos hablar sobre los eventos. Los eventos son acciones que realiza el usuario. Los programas como Javascript están preparados para atrapar determinadas acciones realizadas, en este caso sobre la página, y realizar acciones como respuesta. De este modo se pueden realizar programas interactivos, ya que controlamos los movimientos del usuario y respondemos a ellos. Existen muchos tipos de eventos distintos, por ejemplo la pulsación de un botón, el movimiento del ratón o la selección de texto de la página.

Las acciones que queremos realizar como respuesta a un evento se han de indicar dentro del mismo código HTML, pero en este caso se indican en atributos HTML que se colocan dentro de la etiqueta que queremos que responda a las acciones del usuario. En el capítulo donde vimos algún ejemplo rápido ya comprobamos que si queríamos que un botón realizase acciones cuando se pulsase sobre el, debíamos indicarlas dentro del atributo onclick del botón.

Comprobamos pues que se puede introducir código Javascript dentro de determinados atributos de las etiquetas HTML. Veremos más adelante este tipo de ejecución en profundidad y los tipos de eventos que existen.

Ocultar scripts en navegadores antiguos

Ya hemos visto que Javascript se implementó a partir de Netscape 2.0 e Internet Explorer 3.0, incluso hay navegadores que funcionan en sistemas donde sólo se puede visualizar texto y por lo tanto determinadas tecnologías, como este lenguaje, están fuera de su alcance. Así pues, no todos los navegadores del web comprenden Javascript. En los casos en los que no se interpretan los scripts, los navegadores asumen que el código de éstos es texto de la propia página web y como consecuencia, presentan los scripts en la página web como si de texto normal se tratara. Para evitar que el texto de los scripts se escriba en la página cuando los navegadores no los entienden se tienen que ocultar los con comentarios HTML (<!-comentario HTML -->). Veamos con un ejemplo cómo se han de ocultar los scripts.

```
<SCRIPT>
<!--
Código Javascript
//-->
</SCRIPT>
```

Vemos que el inicio del comentario HTML es idéntico a cómo lo conocemos en el HTML, pero el cierre del comentario presenta una particularidad, que empieza por doble barra inclinada. Esto es debido a que el final del comentario contiene varios caracteres que Javascript reconoce como operadores y al tratar de analizarlos lanza un mensaje de error de sintaxis. Para que Javascript no lance un mensaje de error se coloca antes del comentario HTML esa doble barra, que no es más que un comentario Javascript, que conoceremos más adelante cuando hablemos de sintaxis.

El inicio del comentario HTML no es necesario comentarlo con la doble barra, dado que Javascript entiende bien que simplemente se pretende ocultar el código. Una aclaración a este punto: si pusiesemos las dos barras en esta línea, se verían en navegadores antiguos por estar fuera de los comentarios HTML. Las etiquetas <SCRIPT> no las entienden los navegadores antiguos, por lo tanto no las interpretan, tal como hacen con cualquier etiqueta que desconocen.

<NOSCRIPT>

Existe la posibilidad de indicar un texto alternativo para los navegadores que no entienden Javascript, para informarles de que en ese lugar debería ejecutarse un script y que la página no está funcionando al 100% de sus capacidades. También podemos sugerir a los visitantes que actualicen su navegador a una versión compatible con el lenguaje. Para ello utilizamos la etiqueta <NOSCRIPT> y entre esta etiqueta y su correspondiente de cierre podemos colocar el texto alternativo al script.

```
<SCRIPT>
código javascript
</SCRIPT>
<NOSCRIPT>
Este navegador no comprende los scripts que se están ejecutando, debes actualizar tu versión de navegador a una más reciente.
<br><br>
<a href=http://netscape.com>Netscape</a>.<br>
<a href=http://microsoft.com>Microsoft</a>.
</NOSCRIPT>
```

Más sobre colocar scripts

Un par de notas adicionales sobre cómo colocar scripts en páginas web.

Lenguaje que estamos utilizando

La etiqueta <SCRIPT> tiene un atributo que sirve para indicar el lenguaje que estamos utilizando, así como la versión de este. Por ejemplo, podemos indicar que estamos programando en Javascript 1.2 o Visual Basic Script, que es otro lenguaje para programar scripts en el navegador cliente que sólo es compatible con Internet Explorer.

El atributo en cuestión es language y lo más habitual es indicar simplemente el lenguaje con el que se han programado los scripts. El lenguaje por defecto es Javascript, por lo que si no utilizamos este atributo, el navegador entenderá que el lenguaje con el que se está programando es Javascript. Un detalle donde se suele equivocar la gente sin darse cuenta es que lenguage se escribe con dos -g- y no con -g- y con -j- como en castellano.

```
<SCRIPT LANGUAGE=javascript>
```

Ficheros externos de Javascript

Otra manera de incluir scripts en páginas web, implementada a partir de Javascript 1.1, es incluir archivos externos donde se pueden colocar muchas funciones que se utilicen en la página. Los ficheros suelen tener extendión .js y se incluyen de esta manera.

```
<SCRIPT language=javascript src="archivo_externo.js">
//estoy incluyendo el fichero "archivo_externo.js"
</SCRIPT>
```

Dentro de las etiquetas <SCRIPT> se puede escribir cualquier texto y será ignorado por el navegador, sin embargo, los navegadores que no entienden el atributo SRC tendrán a este texto por instrucciones, por lo que es aconsejable poner un comentario Javascript antes de cada línea con el objetivo de que no respondan con un error.

El archivo que incluimos (en este caso archivo_externo.js) debe contener tan solo sentencias Javascript. No debemos incluir código HTML de ningún tipo, ni tan siquiera las etiquetas </SCRIPT> y </SCRIPT>.

Sintaxis Javascript

El lenguaje **Javascript tiene una sintaxis muy parecida a la de Java** por estar basado en él. También es muy parecida a la del lenguaje C, de modo que si el lector conoce alguno de estos dos lenguajes se podrá manejar con facilidad con el código. De todos modos, en los siguientes capítulos vamos a describir toda la sintaxis con detenimiento, por lo que los novatos no tendrán ningún problema con ella.

Comentarios

Un comentario es una parte de código que no es interpretada por el navegador y cuya utilidad radica en facilitar la lectura al programador. El programador, a medida que desarrolla el script, va dejando frases o palabras sueltas, llamadas comentarios, que le ayudan a él o a cualquier otro a leer mas fácilmente el script a la hora de modificarlo o depurarlo.

Ya se vio anteriormente algún comentario Javascript, pero ahora vamos a contarlos de nuevo. Existen dos tipos de comentarios en el lenguaje. Uno de ellos, la doble barra, sirve para comentar una línea de código. El otro comentario lo podemos utilizar para comentar varias líneas y se indica con los signos /* para empezar el comentario y */ para terminarlo. Veamos unos ejemplos.

```
<SCRIPT>
//Este es un comentario de una línea
/*Este comentario se puede extender
por varias líneas.
Las que quieras*/
</SCRIPT>
```

Mayúsculas y minúsculas

En javascript se han de respetar las mayúsculas y las minúsculas. Si nos equivocamos al utilizarlas el navegador responderá con un mensaje de error de sintaxis. Por convención los nombres de las cosas se escriben en minúsculas, salvo que se utilice un nombre con más de una palabra, pues en ese caso se escribirán con mayúsculas las iniciales de las palabras siguientes a la primera. También se puede utilizar mayúsculas en las iniciales de las primeras palabras en algunos casos, como los nombres de las clases, aunque ya veremos más adelante cuáles son estos casos y qué son las clases.

Separación de instrucciones

Las distintas instrucciones que contienen nuestros scripts se han de separar convenientemente para que el navegador no indique los correspondientes errores de sintaxis. Javascript tiene dos maneras de separar instrucciones. La primera es a través del carácter punto y coma (;) y la segunda es a través de un salto de línea.

Por esta razón Las sentencias Javascript no necesitan acabar en punto y coma a no ser que coloquemos dos instrucciones en la misma línea.

No es una mala idea, de todos modos, acostumbrarse a utilizar el punto y coma después de cada instrucción pues otros lenguajes como Java o C obligan a utilizarlas y nos estaremos acostumbrando a realizar una sintaxis más parecida a la habitual en entornos de programación avanzados.

Variables Javascript

Una variable es un espacio en memoria donde se almacena un dato, un espacio donde podemos guardar cualquier tipo de información que necesitemos para realizar las acciones de nuestros programas. Por ejemplo, si nuestro programa realiza sumas, será muy normal que guardemos en variables los distintos sumandos que participan en la operación y el resultado de la suma. El efecto sería algo parecido a esto.

```
sumando1 = 23
sumando2 = 33
suma = sumando1 + sumando2
```

En este ejemplo tenemos tres variables, sumando1, sumando2 y suma, donde guardamos el resultado. Vemos que su uso para nosotros es como si tuviésemos un apartado donde guardar un dato y que se pueden acceder a ellos con sólo poner su nombre.

Los nombres de las variables han de construirse con caracteres alfanuméricos y el carácter subrayado (_). Aparte de esta, hay una serie de reglas adicionales para construir nombres para variables. La más importante es que tienen que comenzar por un carácter alfabético o el subrayado. No podemos utilizar caracteres raros como el signo +, un espacio o un $. Nombres admitidos para las variables podrían ser

```
Edad
paisDeNacimiento
_nombre
```

También hay que evitar utilizar nombres reservados como variables, por ejemplo no podremos llamar a nuestra variable palabras como return o for, que ya veremos que son utilizadas para estructuras del propio lenguaje. Veamos ahora algunos nombres de variables que no está permitido utilizar

12meses
tu nombre
return
pe%pe

Declaración de variables

Declarar variables consiste en definir y de paso informar al sistema de que vas a utilizar una variable. Es una costumbre habitual en los lenguajes de programación el definir las variables que se van a usar en los programas y para ello, se siguen unas reglas estrictas. Pero javascript se salta muchas reglas por ser un lenguaje un tanto libre a la hora de programar y uno de los casos en los que otorga un poco de libertad es a la hora de declarar las variables, ya que no estamos obligados a hacerlo, al contrario de lo que pasa en la mayoría de los lenguajes de programación.

De todos modos, es aconsejable declarar las variables, además de una buena costumbre y para ello Javascript cuenta con la palabra var. Como es lógico, se utiliza esa palabra para definir la variable antes de utilizarla.

```
var operando1
var operando2
```

También se puede asignar un valor a la variable cuando se está declarando

```
var operando1 = 23
var operando2 = 33
```

También se permite declarar varias variables en la misma línea, siempre que se separen por comas.

```
var operando1,operando2
```

Ambito de las variables

Se le llama ámbito de las variables al lugar donde estas están disponibles. Por lo general, cuando declaramos una variable hacemos que esté disponible en el lugar donde se ha declarado, esto ocurre en todos los lenguajes de programación y como javascript se define dentro de una página web, **las variables que declaremos en la página estarán accesibles dentro de ella**. De este modo, no podremos acceder a variables que hayan sido definidas en otra página. Este es el ámbito más habitual de una variable y le llamaremos a este tipo de variables globales a la página, aunque no será el único, ya que también podremos declarar variables en lugares más acotados.

Variables globales

Como hemos dicho, las variables globales son las que están declaradas en el ámbito más amplio posible, que en Javascript es una página web. Para declarar una variable global a la página simplemente lo haremos en un script, con la palabra *var*.

```
<SCRIPT>
var variableGlobal
</SCRIPT>
```

Las variables globales son accesibles desde cualquier lugar de la página, es decir, desde el script donde se han declarado y todos los demás scripts de la página, incluidos los manejadores de eventos, como el onclick, que ya vimos que se podía incluir dentro de determinadas etiquetas HTML.

Variables locales

También podremos declarar variables en lugares más acotados, como por ejemplo una función. A estas variables les llamaremos locales. Cuando se declaren variables locales sólo podremos acceder a ellas dentro del lugar donde se ha declarado, es decir, si la habíamos declarado en una función solo podremos acceder a ella cuando estemos en esa función.

Las variables pueden ser locales a una función, pero también pueden ser locales a otros ámbitos, como por ejemplo un bucle. En general, son ámbitos locales cualquier lugar acotado por llaves.

```
<SCRIPT>
function miFuncion (){
   var variableLocal
}
</SCRIPT>
```

En el script anterior hemos declarado una variable dentro de una función, por lo que esa variable sólo tendrá validez dentro de la función. Se pueden ver cómo se utilizan las llaves para acotar el lugar donde está definida esa función o su ámbito.

No hay problema en declarar una variable local con el mismo nombre que una global, en este caso la variable global será visible desde toda la página, excepto en el ámbito donde está declarada la variable local ya que en este sitio ese nombre de variable está ocupado por la local y es ella quien tiene validez.

```
<SCRIPT>
var numero = 2
function miFuncion (){
   var numero = 19
   document.write(numero) //imprime 19
}
document.write(numero) //imprime 2
</SCRIPT>
```

Un consejo para los principiantes podría ser no declarar variables con los mismos nombres, para que nunca haya lugar a confusión sobre qué variable es la que tiene validez en cada momento.

Diferencias entre utilizar var o no

Como hemos dicho, en Javascript tenemos libertad para declarar o no las variables con la palabra var, pero los efectos que conseguiremos en cada caso serán distintos. En concreto, cuando utilizamos var estamos haciendo que la varible que estamos declarando sea local al ámbito donde se declara. Por otro lado, si no utilizamos la palabra var para declarar una variable, ésta será global a toda la página, sea cual sea el ámbito en el que haya sido declarada.

En el caso de una variable declarada en la página web, fuera de una función o cualquier otro ámbito más reducido, nos es indiferente si se declara o no con var, desde un punto de vista funcional. Esto es debido a que cualquier variable declarada fuera de un ámbito es global a toda la página. La diferencia se puede apreciar en una función por ejemplo, ya que si utilizamos var la variable será local a la función y si no lo utilizamos, la variable será global a la página. Esta diferencia es fundamental a la hora de controlar correctamente el uso de las variables en la página, ya que si no lo hacemos en una función podríamos sobreescribir el valor de una variable, perdiendo el dato que pudiera contener previamente.

```
<SCRIPT>
var numero = 2
function miFuncion (){
   numero = 19
   document.write(numero) //imprime 19
}
document.write(numero) //imprime 2
//llamamos a la función
miFuncion()
document.write(numero) //imprime 19
</SCRIPT>
```

En este ejemplo, tenemos una variable global a la página llamada numero, que contiene un 2. También tenemos una función que utiliza la variable numero sin haberla declarado con var, por lo que la variable numero de la funcion será la misma variable global numero declarada fuera de la función. En una situación como esta, al ejecutar la función se sobreescribirá la variable numero y el dato que había antes de ejecutar la función se perderá.

Qué podemos guardar en variables

En una variable podemos introducir varios tipos de información, por ejemplo texto, números enteros o reales, etc. A estas distintas clases de información se les conoce como tipos de datos. Cada uno tiene características y usos distintos, veamos cuáles son los tipos de datos de Javascript.

Números
Para empezar tenemos el tipo numérico, para guardar números como 9 o 23.6

Cadenas
El tipo cadena de carácter guarda un texto. Siempre que escribamos una cadena de caracteres debemos utilizar las comillas (").

Boleanos
También contamos con el tipo boleano, que guarda una información que puede valer si (true) o no (false).

Por último sería relevante señalar aquí que nuestras variables pueden contener cosas más complicadas, como podría ser un objeto, una función, o vacío (null) pero ya lo veremos más adelante.

En realidad nuestras variables no están forzadas a guardar un tipo de datos en concreto y por lo tanto no especificamos ningún tipo de datos para una variable cuando la estamos declarando. Podemos introducir cualquier información en una variable de cualquier tipo, incluso podemos ir cambiando el contenido de una variable de un tipo a otro sin ningún problema. Vamos a ver esto con un ejemplo.

```
var nombre_ciudad = "Valencia"
var revisado = true
nombre_ciudad = 32
revisado = "no"
```

Esta ligereza a la hora de asignar tipos a las variables puede ser una ventaja en un principio, sobretodo para personas inexpertas, pero a la larga puede ser fuente de errores ya que dependiendo del tipo que son las variables se comportarán de un modo u otro y si no controlamos con exactitud el tipo de las variables podemos encontrarnos sumando un texto a un número. Javascript operará perfectamente, y devolverá un dato, pero en algunos casos puede que no sea lo que estábamos esperando. Así pues, aunque tenemos libertad con los tipos, esta misma libertad nos hace estar más atentos a posibles desajustes difíciles de detectar a lo largo de los programas. Veamos lo que ocurriría en caso de sumar letras y números.

```
var sumando1 = 23
var sumando2 = "33"
var suma = sumando1 + sumando2
document.write(suma)
```

Este script nos mostraría en la página el texto 2333, que no se corresponde con la suma de los dos números, sino con su concatenación, uno detrás del otro.

Veremos algunas cosas más referentes a los tipos de datos más adelante.

Tipos de datos en Javascript

En nuestros scripts vamos a manejar variables diversas clases de información, como textos o números. Cada una de estas clases de información son los tipos de datos. Javascipt distingue entre tres tipos de datos y todas las informaciones que se puedan guardar en variables van a estar encajadas en uno de estos tipos de datos. Veamos detenidamente cuáles son estos tres tipos de datos.

Tipo de datos numérico

En este lenguaje sólo existe un tipo de datos numérico, al contrario que ocurre en la mayoría de los lenguajes más conocidos. Todos los números son por tanto del tipo numérico, independientemente de la precisión que tengan o si son números reales o enteros. Los números enteros son números que no tienen coma, como 3 o 339. Los números reales son números fraccionarios, como 2.69 o 0.25, que también se pueden escribir en notación científica, por ejemplo 2.482e12.

Con Javascript también podemos escribir números en otras bases, como la hexadecimal. Las bases son sistemas de numeración que utilizan más o menos dígitos para escribir los números. Existen tres bases con las que podemos trabajar

- Base 10, es el sistema que utilizamos habitualmente, el sistema decimal. Cualquier número, por defecto, se entiende que está escrito en base 10.
- Base 8, también llamado sistema octal, que utiliza dígitos del 0 al 7. Para escribir un número en octal basta con escribir ese número precedido de un 0, por ejemplo 045.
- Base 16 o sistema hexadecimal, es el sistema de numeración que utiliza 16 dígitos, los comprendidos entre el 0 y el 9 y las letras de la A a la F, para los dígitos que faltan. Para escribir un número en hexadecimal debemos escribirlo precedido de un cero y una equis, por ejemplo 0x3EF.

Tipo boleano

El tipo bolean, boolean en inglés, sirve para guardar un si o un no o dicho de otro modo, un verdadero o un falso. Se utiliza para realizar operaciones lógicas, generalmente para realizar acciones si el contenido de una variable es verdadero o falso.

Si una variable es verdadero entonces Ejecuto unas instrucciones Si no Ejecuto otras

Los dos valores que pueden tener las variables boleanas son true o false.

```
miBoleana = true
miBoleana = false
```

Tipo de datos cadena de caracteres

El último tipo de datos es el que sirve para guardar un texto. Javascript sólo tiene un tipo de datos para guardar texto y en el se pueden introducir cualquier número de caracteres. Un texto puede estar compuesto de números, letras y cualquier otro tipo de caracteres y signos. Los textos se escriben entre comillas, dobles o simples.

```
miTexto = "Pepe se va a pescar"
miTexto = '23%%$ Letras & *--*'
```

Todo lo que se coloca entre comillas, como en los ejemplos anteriores es tratado como una cadena de caracteres independientemente de lo que coloquemos en el interior de las comillas. Por ejemplo, en una variable de texto podemos guardar números y en ese caso tenemos que tener en cuenta que las variables de tipo texto y las numéricas no son la misma cosa y mientras que las de numéricas nos sirven para hacer cálculos matemáticos las de texto no.

Caracteres de escape en cadenas de texto.

Hay una serie de caracteres especiales que sirven para expresar en una cadena de texto determinados controles como puede ser un salto de línea o un tabulador. Estos son los caracteres de escape y se escriben con una notación especial que comienza por una contra barra (una barra inclinada al revés de la normal '\') y luego se coloca el código del carácter a mostrar.

Un carácter muy común es el salto de línea, que se consigue escribiendo \n. Otro carácter muy habitual es colocar unas comillas, pues si colocamos unas comillas sin su carácter especial nos cerrarían las comillas que colocamos para iniciar la cadena de caracteres. Las comillas las tenemos que introducir entonces con \" o \' (comillas dobles o simples). Existen otros caracteres de escape, que veremos en la tabla de abajo más resumidos, aunque también hay que destacar como carácter habitual el que se utiliza para escribir una contrabarra, para no confundirla con el inicio de un carácter de escape, que es la doble contrabarra \\.

Tabla con todos los caracteres de escape

```
Salto de línea: \n
Comilla simple: \'
Comilla doble: \"
Tabulador: \t
Retorno de carro: \r
Avance de página: \f
Retroceder espacio: \b
Contrabarra: \\
```

Algunos de estos caracteres probablemente no los llegarás a utilizar nunca, pues su función es un poco rara y a veces poco clara.

Operadores Javascript I

Al desarrollar programas en cualquier lenguaje se utilizan los operadores. Éstos sirven para hacer los cálculos y operaciones necesarios para llevar a cabo sus objetivos. Un programa que no realiza operaciones solo se puede limitar a hacer siempre lo mismo, es el resultado de estas operaciones lo que hace que un programa varíe su comportamiento según los datos que obtenga. Existen operaciones más sencillas o complejas, que se pueden realizar con operandos de distintos tipos de datos, como números o textos, veremos en este capítulo de manera detallada todos estos operadores.

Ejemplos de uso de operadores

Antes de entrar a enumerar los distintos tipos de operadores vamos a ver un par de ejemplos de éstos para que nos ayuden a hacernos una idea más exacta de lo que son. En el primer ejemplo vamos a realizar una suma utilizando el operador suma.

```
3 + 5
```

Esta es una expresión muy básica que no tiene mucho sentido ella sola. Hace la suma entre los dos operandos número 3 y 5, pero no sirve de mucho porque no se hace nada con el resultado. Normalmente se combinan más de un operador para crear expresiones más útiles. La expresión siguiente es una combinación entre dos operadores, uno realiza una operación matemática y el otro sirve para guardar el resultado.

```
miVariable = 23 * 5
```

En el ejemplo anterior, el operador * se utiliza para realizar una multiplicación y el operador = se utiliza para asignar el resultado en una variable, de modo que guardemos el valor para su posterior uso.

Los operadores se pueden clasificar según el tipo de acciones que realizan. A continuación vamos a ver cada uno de estos grupos de operadores y describiremos la función de cada uno.

Operadores aritméticos

Son los utilizados para la realización de operaciones matemáticas simples como la suma, resta o multiplicación. En javascript son los siguientes:

+ Suma de dos valores
- Resta de dos valores, también puede utilizarse para cambiar el signo de un número si lo utilizamos con un solo operando -23
* Multiplicación de dos valores
/ División de dos valores
% El resto de la división de dos números (3%2 devolvería 1, el resto de dividir 3 entre 2)
++ Incremento en una unidad, se utiliza con un solo operando
-- Decremento en una unidad, utilizado con un solo operando

Ejemplos

```
precio = 128 //introduzco un 128 en la variable precio
unidades = 10 //otra asignación, luego veremos operadores de asignación
factura = precio * unidades //multiplico precio por unidades, obtengo el valor factura
resto = factura % 3 //obtengo el resto de dividir la variable factura por 3
precio++ //incrementa en una unidad el precio (ahora vale 129)
```

Operadores de asignación

Sirven para asignar valores a las variables, ya hemos utilizado en ejemplos anteriores el operador de asignación =, pero hay otros operadores de este tipo, que provienen del lenguaje C y que muchos de los lectores ya conocerán.

= Asignación. Asigna la parte de la derecha del igual a la parte de la izquierda. A al derecha se colocan los valores finales y a la izquierda generalmente se coloca una variable donde queremos guardar el dato.
+= Asignación con suma. Realiza la suma de la parte de la derecha con la de la izquierda y guarda el resultado en la parte de la izquierda.
-= Asignación con resta
*= Asignación de la multiplicación
/= Asignación de la división
%= Se obtiene el resto y se asigna

Ejemplos

```
ahorros = 7000 //asigna un 7000 a la variable ahorros
ahorros += 3500 //incrementa en 3500 la variable ahorros, ahora vale 10500
ahorros /= 2 //divide entre 2 mis ahorros, ahora quedan 5250
```

Operadores Javascript II

Operadores de cadenas

Las cadenas de caracteres, o variables de texto, también tienen sus propios operadores para realizar acciones típicas sobre cadenas. Aunque javascript sólo tiene un operador para cadenas se pueden realizar otras acciones con una serie de funciones predefinidas en el lenguaje que veremos más adelante.

+ Concatena dos cadenas, pega la segunda cadena a continuación de la primera.

Ejemplo

```
cadena1 = "hola"
cadena2 = "mundo"
cadenaConcatenada = cadena1 + cadena2 //cadena concatenada vale "holamundo"
```

Un detalle importante que se puede ver en este caso es que el operador + sirve para dos usos distintos, si sus operandos son números los suma, pero si se trata de cadenas las concatena. Esto pasa en general con todos los operadores que se repiten en el lenguaje, javascript es suficientemente listo para entender que tipo de operación realizar mediante una comprobación de los tipos que están implicados en élla.

Un caso que resultaría confuso es el uso del operador + cuando se realiza la operación con operadores texto y numéricos entremezclados. En este caso javascript asume que se desea realizar una concatenación y trata a los dos operandos como si de cadenas de caracteres se trataran, incluso si la cadena de texto que tenemos fuese un número. Esto lo veremos más fácilmente con el siguiente ejemplo.

```
miNumero = 23
miCadena1 = "pepe"
miCadena2 = "456"
resultado1 = miNumero + miCadena1 //resultado1 vale "23pepe"
resultado2 = miNumero + miCadena2 //resultado2 vale "23456"
miCadena2 += miNumero //miCadena2 ahora vale "45623"
```

Como hemos podido ver, también en el caso del operador +=, si estamos tratando con cadenas de texto y números entremezclados, tratará a los dos operadores como si fuesen cadenas.

Operadores lógicos

Estos operadores sirven para realizar operaciones lógicas, que son aquellas que dan como resultado un verdadero o un falso, y se utilizan para tomar decisiones en nuestros scripts. En vez de trabajar con números, para realizar este tipo de operaciones se utilizan operandos boleanos, que conocimos anteriormente, que son el verdadero (true) y el falso (false). Los operadores lógicos relacionan los operandos boleanos para dar como resultado otro operando boleano, tal como podemos ver en el siguiente ejemplo.

Si tengo hambre y tengo comida entonces me pongo a comer

Nuestro programa javascript utilizaría en este ejemplo un operando boleano para tomar una decisión. Primero mirará si tengo hambre, si es cierto (true) mirará si dispongo de comida. Si son los dos ciertos, se puede poner a comer. En caso de que no tenga comida o que no tenga hambre no comería, al igual que si no tengo hambre ni comida. El operando en cuestión es el operando Y, que valdrá verdadero (true) en caso de que los dos operandos valgan verdadero.

! Operador NO o negación. Si era true pasa a false y viceversa.
&& Operador Y, si son los dos verdaderos vale verdadero.
|| Operador O, vale verdadero si por lo menos uno de ellos es verdadero.

Ejemplo

```
miBoleano = true
miBoleano = !miBoleano //miBoleano ahora vale false
tengoHambre = true
tengoComida = true
comoComida = tengoHambre && tengoComida
```

Operadores condicionales

Sirven para realizar expresiones condicionales todo lo complejas que deseemos. Estas expresiones se utilizan para tomar decisiones en función de la comparación de varios elementos, por ejemplo si un numero es mayor que otro o si son iguales. Los operadores condicionales se utilizan en las expresiones condicionales para tomas de decisiones. Como estas expresiones condicionales serán objeto de estudio más adelante será mejor describir los operadores condicionales más adelante. De todos modos aquí podemos ver la tabla de operadores condicionales.

== Comprueba si dos números son iguales
!= Comprueba si dos números son distintos
> Mayor que, devuelve true si el primer operador es mayor que el segundo
< Menor que, es true cuando el elemento de la izquierda es menor que el de la derecha
>= Mayor igual.
<= Menor igual

Operadores Javascript III

Operadores a nivel de bit

Estos son muy poco corrientes y es posible que nunca los llegues a utilizar. Su uso se realiza para efectuar operaciones con ceros y unos. Todo lo que maneja un ordenador son ceros y unos, aunque nosotros utilicemos números y letras para nuestras variables en realidad estos valores están escritos internamente en forma de ceros y unos. En algunos caso podremos necesitar realizar operaciones tratando las variables como ceros y unos y para ello utilizaremos estos operandos. En este manual se nos queda un poco grande realizar una discusión sobre este tipo de operadores, pero aquí podréis ver estos operadores por si algún día os hacen falta.

& Y de bits
^ Xor de bits
| O de bits
<< >> >>> >>>= >>= <<= Varias clases de cambios

Precedencia de los operadores

La evaluación de una sentencia de las que hemos visto en los ejemplos anteriores es bastante sencilla y fácil de interpretar, pero cuando en una sentencia entran en juego multitud de operadores distintos puede haber una confusión a la hora de interpretarla y dilucidar qué operadores son los que se ejecutan antes que otros. Para marcar unas pautas en la evaluación de las sentencias y que estas se ejecuten siempre igual y con sentido común existe la precedencia de operadores, que no es más que el orden por el que se irán ejecutando las operaciones que ellos representan. En un principio todos los operadores se evalúan de izquierda a derecha, pero existen unas normas adicionales, por las que determinados operadores se evalúan antes que otros. Muchas de estas reglas de precedencia están sacadas de las matemáticas y son comunes a otros lenguajes, las podemos ver a continuación.

() [] . Paréntesis, corchetes y el operador punto que sirve para los objetos
! - ++ -- negación, negativo e incrementos
* / % Multiplicación división y módulo
+- Suma y resta
<< >> >>> Cambios a nivel de bit
< <= > >= Operadores condicionales
== != Operadores condicionales de igualdad y desigualdad
& ^ | Lógicos a nivel de bit
&& || Lógicos boleanos
= += -= *= /= %= <<= >>= >>>= &= ^= != Asignación

En los siguientes ejemplos podemos ver cómo las expresiones podrían llegar a ser confusas, pero con la tabla de precedencia de operadores podremos entender sin errores cuál es el orden por el que se ejecutan.

12 * 3 + 4 - 8 / 2 % 3

En este caso primero se ejecutan los operadores * / y %, de izquierda a derecha, con lo que se realizarían estas operaciones. Primero la multiplicación y luego la división por estar más a la izquierda del módulo.

36 + 4 - 4 % 3

Ahora el módulo.

36 + 4 - 1

Por último las sumas y las restas de izquierda a derecha.

40 - 1

39

De todos modos, es importante darse cuenta que el uso de los paréntesis puede ahorrarnos muchos quebraderos de cabeza y sobretodo la necesidad de sabernos de memoria la tabla de precedencia de los operadores. Cuando veamos poco claro el orden con el que se ejecutarán las sentencias podemos utilizarlos y así forzar que se evalúe antes el trozo de expresión que se encuentra dentro de los paréntesis.

Control de tipos

Hemos visto para determinados operadores que es importante el tipo de datos que están manejando, puesto que si los datos son de un tipo se realizarán operaciones distintas que si son de otro.

Así, cuando utilizábamos el operador +, si se trataba de números los sumaba, pero si se trataba de cadenas de caracteres los concatenaba. Vemos pues que el tipo de los datos que estamos utilizando sí que importa y que tendremos que estar pendientes este detalle si queremos que nuestras operaciones se realicen tal como esperábamos.

Para comprobar el tipo de un dato se puede utilizar otro operador que está disponible a partir de javascript 1.1, el operador typeof, que devuelve una cadena de texto que describe el tipo del operador que estamos comprobando.

```
var boleano = true
var numerico = 22
var numerico_flotante = 13.56
var texto = "mi texto"
var fecha = new Date()
document.write("<br>El tipo de boleano es: " + typeof boleano)
document.write("<br>El tipo de numerico es: " + typeof numerico)
document.write("<br>El tipo de numerico_flotante es: " + typeof numerico_flotante)
document.write("<br>El tipo de texto es: " + typeof texto)
document.write("<br>El tipo de fecha es: " + typeof fecha)
```

Este script dará como resultado lo siguiente:

```
El tipo de boleano es: boolean
El tipo de numerico es: number
El tipo de numerico_flotante es: number
El tipo de texto es: string
El tipo de fecha es: object
```

En este ejemplo podemos ver que ser imprimen en la página los distintos tipos de las variables. Estos pueden ser los siguientes:

- boolean, para los datos boleanos. (True o false)
- number, para los numéricos.
- string, para las cadenas de caracteres.
- object, para los objetos.

Queremos destacar tan sólo dos detalles más:

1) Los números, ya tengan o no parte decimal, son siempre del tipo de datos numérico.

2) Una de las variables es un poco más compleja, es la variable fecha que es un objeto de la clase Date(), que se utiliza para el manejo de fechas en los scripts. La veremos más adelante, así como los objetos.

Estructuras de control

Los scripts vistos hasta ahora han sido tremendamente sencillos y lineales: se iban ejecutando las sentencias simples una detrás de la otra desde el principio hasta el fin. Sin embargo, esto no tiene porque ser siempre así, en los programas generalmente necesitaremos hacer cosas distintas dependiendo del estado de nuestras variables o realizar un mismo proceso muchas veces sin escribir la misma línea de código una y otra vez.

Para realizar cosas más complejas en nuestros scripts se utilizan las estructuras de control. Utilizándolas podemos realizar tomas de decisiones y bucles. En los siguientes capítulos vamos a conocer las distintas estructuras de control que existen en Javascript.

Toma de decisiones

Nos sirven para realizar unas acciones u otras en función del estado de las variables. Es decir, tomar decisiones para ejecutar unas instrucciones u otras dependiendo de lo que esté ocurriendo en ese instante en nuestros programas.

Por ejemplo, dependiendo si el usuario que entra en nuestra página es mayor de edad o no lo es, podemos permitirle o no ver los contenidos de nuestra página.

```
Si edad es mayor que 18 entonces
   Te dejo ver el contenido para adultos
Si no
   Te mando fuera de la página
```

En javascript podemos tomar decisiones utilizando dos enunciados distintos.

- IF
- SWITCH

Bucles

Los bucles se utilizan para realizar ciertas acciones repetidamente. Son muy utilizados a todos los niveles en la programación. Con un bucle podemos por ejemplo imprimir en una página los números del 1 al 100 sin necesidad de escribir cien veces el la instrucción imprimir.

```
Desde el 1 hasta el 100
  Imprimir el número actual
```

En javascript existen varios tipos de bucles, cada uno está indicado para un tipo de iteración distinto y son los siguientes:

- FOR
- WHILE
- DO WHILE

Como hemos señalado ya, las estructuras de control son muy importantes en Javascript y en cualquier lenguaje de programación. Es por ello que en los siguientes capítulos veremos cada una de estas estructuras detenidamente, describiendo su uso y ofreciendo algunos ejemplos.

Estructura IF

IF es una estructura de control utilizada para tomar decisiones. Es un condicional que realiza unas u otras operaciones en función de una expresión. Funciona de la siguiente manera, primero se evalúa una expresión, si da resultado positivo se realizan las acciones relacionadas con el caso positivo.

La sintaxis de la estructura IF es la siguiente.

```
if (expresión) {
  acciones a realizar en caso positivo
  ...
}
```

Opcionalmente se pueden indicar acciones a realizar en caso de que la evaluación de la sentencia de resultados negativos.

```
if (expresión) {
  acciones a realizar en caso positivo
  ...
} else {
  acciones a realizar en caso negativo
  ...
}
```

Fijémonos en varias cosas. Para empezar vemos como con unas llaves engloban las acciones que queremos realizar en caso de que se cumplan o no las expresiones. Estas llaves han de colocarse siempre, excepto en el caso de que sólo haya una instrucción como acciones a realizar, que son opcionales.

Otro detalle que salta a la vista es el sangrado (margen) que hemos colocado en cada uno de los bloques de instrucciones a ejecutar en los casos positivos y negativos. Este sangrado es totalmente opcional, sólo lo hemos hecho así para que la estructura IF se comprenda de una manera más visual. Los saltos de línea tampoco son necesarios y se han colocado también para que se vea mejor la estructura. Perfectamente podríamos colocar toda la instrucción IF en la misma línea de código, pero eso no ayudará a que las cosas estén claras. Nosotros, y cualquier programador, te aconsejamos que utilices los sangrados y saltos de línea necesarios para que las instrucciones se puedan entender mejor, hoy y dentro de un mes, cuando te acuerdes menos de lo que hiciste en tus scripts.

Veamos algún ejemplo de condicionales IF.

```
if (dia == "lunes")
  document.write ("Que tengas un feliz comienzo de semana")
```

Si es lunes nos deseará una feliz semana. No hará nada en caso contrario. Como en este ejemplo sólo indicamos una instrucción para el caso positivo, no hará falta utilizar las llaves. Fíjate también en el operador condicional que consta de dos signos igual.

Vamos a ver ahora otro ejemplo, un poco más largo.

```
if (credito >= precio) {
  document.write("has comprado el artículo " + nuevoArtículo) //enseño compra
```

```
  carrito += nuevoArticulo //introduzco el artículo en el carrito de la compra
  credito -= precio //disminuyo el crédito según el precio del artículo
} else {
  document.write("se te ha acabado el crédito") //informo que te falta dinero
  window.location = "carritodelacompra.html" //voy a la página del carrito
}
```

Este ejemplo es un poco más complejo, y también un poco ficticio. Lo que hago es comprobar si tengo crédito para realizar una supuesta compra. Para ello miro si el crédito es mayor o igual que el precio del artículo, si es así informo de la compra, introduzco el artículo en el carrito y resto el precio al crédito acumulado. Si el precio del artículo es superior al dinero disponible informo de la situación y mando al navegador a la página donde se muestra su carrito de la compra.

Expresiones condicionales

La expresión a evaluar se coloca siempre entre paréntesis y está compuesta por variables que se combinan entre si mediante operadores condicionales. Recordamos que los operadores condicionales relacionaban dos variables y devolvían siempre un resultado boleano. Por ejemplo un operador condicional es el operador "es igual" (==), que devuelve true en caso de que los dos operandos sean iguales o false en caso de que sean distintos.

```
if (edad > 18)
  document.write("puedes ver esta página para adultos")
```

En este ejemplo utilizamos en operador condicional "es mayor" (>). En este caso, devuelve true si la variable edad es mayor que 18, con lo que se ejecutaría la línea siguiente que nos informa de que se puede ver el contenido para adultos.

Las expresiones condicionales se pueden combinar con las expresiones lógicas para crear expresiones más complejas. Recordamos que las expresiones lógicas son las que tienen como operandos a los boleanos y que devuelvan otro valor boleano. Son los operadores negación lógica, Y lógico y O lógico.

```
if (bateria == 0 && redElectrica = 0)
  document.write("tu ordenador portatil se va a apagar en segundos")
```

Lo que hacemos es comprobar si la batería de nuestro supuesto ordenador está a cero (acabada) y también comprobamos si el ordenador no tiene red eléctrica (está desenchufado). Luego, el operador lógico los relaciona con un Y, de modo que si está sin batería Y sin red eléctrica, informo que el ordenador se va a apagar.

La lista de operadores que se pueden utilizar con las estructuras IF se pueden ver en el capítulo de operadores condicionales y operadores lógicos.

Estructura IF (parte II)

Sentencias IF anidadas

Para hacer estructuras condicionales más complejas podemos anidar sentencias IF, es decir, colocar estructuras IF dentro de otras estructuras IF. Con un solo IF podemos evaluar y realizar una acción u otra según dos posibilidades, pero si tenemos más posibilidades que evaluar debemos anidar Ifs para crear el flujo de código necesario para decidir correctamente.

Por ejemplo, si deseo comprobar si un número es mayor menor o igual que otro, tengo que evaluar tres posibilidades distintas. Primero puedo comprobar si los dos números son iguales, si lo son, ya he resuelto el problema, pero si no son iguales todavía tendré que ver cuál de los dos es mayor. Veamos este ejemplo en código Javascript.

```
var numero1=23
var numero2=63
if (numero1 == numero2){
  document.write("Los dos números son iguales")
else {
  if (numero1 > numero2) {
    document.write("El primer número es mayor que el segundo")
  }else{
    document.write("El primer número es menor que el segundo")
  }
}
```

El flujo del programa es como comentábamos antes, primero se evalúa si los dos números son iguales. En caso positivo se muestra un mensaje informándolo. En caso contrario ya sabemos que son distintos,

pero aun debemos averiguar cuál de los dos en mayor. Para eso se hace otra comparación para saber si el primero es mayor que el segundo. Si esta comparación da resultados positivos mostramos un mensaje diciendo que el primero es mayor que el segundo, en caso contrario indicaremos que el primero es menor que el segundo.

Volvemos a remarcar que las llaves son en este caso opcionales, pues sólo se ejecuta una sentencia para cada caso. Además, los saltos de línea y los sangrados también opcionales en todo caso y nos sirven sólo para ver el código de una manera más ordenada. Mantener el código bien estructurado y escrito de una manera comprensible es muy importante, ya que nos hará la vida más agradable a la hora de programar y más adelante cuando tengamos que revisar los programas. En este manual utilizaré una notación como la que has podido ver en las líneas anteriores, y verás en adelante, además mantendré esa notación en todo momento. Esto sin lugar a dudas hará que los códigos con ejemplos sean comprensibles más rápidamente, si no lo hiciéramos así sería un verdadero incordio leerlos. Esta misma receta es aplicable a los códigos que has de crear tú y el principal beneficiado serás tú mismo y los compañeros que lleguen a leer tu código.

Operador IF

Hay un operador que no hemos visto todavía y es una forma más esquemática de realizar algunos IF sencillos. Proviene del lenguaje C, donde se escriben muy pocas líneas de código que resulta muy elegante. Este operador es un claro ejemplo de ahorro de líneas y caracteres al escribir los scripts. Lo veremos rápidamente, pues la única razón por la que lo incluyo es para que sepas que existe y si lo encuentras en alguna ocasión por ahí sepas identificarlo y cómo funciona.

Un ejemplo de uso del operador IF se puede ver a continuación.

```
Variable = (condición) ? valor1 : valor2
```

Este ejemplo no sólo realiza una comparación de valores, además asigna un valor a una variable. Lo que hace es evaluar la condición (colocada entre paréntesis) y si es positiva asigna el valor1 a la variable y en caso contrario le asigna el valor2. Veamos un ejemplo:

```
momento = (hora_actual < 12) ? "Antes del mediodía" : "Después del mediodía"
```

Este ejemplo mira si la hora actual es mayor que 12. Si es así, es que ahora es antes del mediodía, así que asigna "Antes del mediodía" a la variable momento. Si la hora es mayo o igual a 12 es que ya es después de mediodía, con lo que se asigna el texto "Después del mediodía" a la variable momento.

Estructura SWITCH

Es la otra expresión disponible en Javascript para tomar decisiones en función de distintos estados de las variables. Esta expresión se utiliza cuando tenemos múltiples posibilidades como resultado de la evaluación de una sentencia.

La estructura SWITCH se incorporó a partir de la versión 1.2 de Javascript (Netscape 4 e Internet Explorer 4). Su sintaxis es la siguiente.

```
switch (expersión) {
   case valor1:
      Sentencias a ejecutar si la expresión tiene como valor a valor1
      break
   case valor2:
      Sentencias a ejecutar si la expresión tiene como valor a valor2
      break
   case valor3:
      Sentencias a ejecutar si la expresión tiene como valor a valor3
      break
   default:
      Sentencias a ejecutar si el valor no es ninguno de los anteriores
}
```

La expresión se evalúa, si vale valor1 se ejecutan las sentencias relacionadas con ese caso. Si la expresión vale valor2 se ejecutan las instrucciones relacionadas con ese valor y así sucesivamente, por tantas opciones como deseemos. Finalmente, para todos los casos no contemplados anteriormente se ejecuta el caso por defecto.

La palabra break es opcional, pero si no la ponemos a partir de que se encuentre coincidencia con un valor se ejecutarán todas las sentencias relacionadas con este y todas las siguientes. Es decir, si en nuestro esquema anterior no hubiese ningún break y la expresión valiese valor1, se ejecutarían las sentencias relacionadas con valor1 y también las relacionadas con valor2, valor3 y default.

También es opcional la opción default u opción por defecto.

Veamos un ejemplo de uso de esta estructura. Supongamos que queremos indicar que día de la semana es. Si el día es 1 (lunes) sacar un mensaje indicándolo, si el día es 2 (martes) debemos sacar un mensaje distinto y así sucesivamente para cada día de la semana, menos en el 6 (sábado) y 7 (domingo) que queremos mostrar el mensaje "es fin de semana". Para días mayores que 7 indicaremos que ese día no existe.

```
Switch (dia_de_la_semana) {
   case 1:
      document.write("Es Lunes")
      break
   case 2:
      document.write("Es Martes")
      break
   case 3:
      document.write("Es Miércoles")
      break
   case 4:
      document.write("Es Jueves")
      break
   case 5:
      document.write("Es viernes")
      break
   case 6:
   case 7:
      document.write("Es fin de semana")
      break
   default:
      document.write("Ese día no existe")
}
```

El ejemplo es relativamente sencillo, solamente puede tener una pequeña dificultad, consistente en interpretar lo que pasa en el caso 6 y 7, que habíamos dicho que teníamos que mostrar el mismo mensaje. En el caso 6 en realidad no indicamos ninguna instrucción, pero como tampoco colocamos un break se ejecutará la sentencia o sentencias del caso siguiente, que corresponden con la sentencia indicada en el caso 7 que es el mensaje que informa que es fin de semana. Si el caso es 7 simplemente se indica que es fin de semana, tal como se pretendía.

Bucle FOR

El bucle FOR se utiliza para repetir mas instrucciones un determinado número de veces. De entre todos los bucles, el FOR se **suele utilizar cuando sabemos seguro el número de veces que queremos que se ejecute la sentencia**. La sintaxis del bucle se muestra a continuación.

```
for (inicialización;condición;actualización) {
   sentencias a ejecutar en cada iteración
}
```

El bucle FOR tiene tres partes incluidas entre los paréntesis. La primera es la inicialización, que se ejecuta solamente al comenzar la primera iteración del bucle. En esta parte se suele colocar la variable que utilizaremos para llevar la cuenta de las veces que se ejecuta el bucle.

La segunda parte es la condición, que se evaluará cada vez que comience la iteración del bucle. Contiene una expresión para comprobar cuándo se ha de detener el bucle, o mejor dicho, la condición que se debe cumplir para que continúe la ejecución del bucle.

Por último tenemos la actualización, que sirve para indicar los cambios que queramos ejecutar en las variables cada vez que termina la iteración del bucle, antes de comprobar si se debe seguir ejecutando.

Después del for se colocan las sentencias que queremos que se ejecuten en cada iteración, acotadas entre llaves.

Un ejemplo de utilización de este bucle lo podemos ver a continuación, donde se imprimirán los números del 0 al 10.

```
var i
for (i=0;i<=10;i++) {
   document.write(i)
}
```

En este caso se inicializa la variable i a 0. Como condición para realizar una iteración, se tiene que cumplir que la variable i sea menor o igual que 10. Como actualización se incremetará en 1 la variable i.

Como se puede comprobar, **este bucle es muy potente, ya que en una sola línea podemos indicar muchas cosas distintas y muy variadas**.

Por ejemplo si queremos escribir los número del 1 al 1.000 de dos en dos se escribirá el siguiente bucle.

```
for (i=1;i<=1000;i+=2)
   document.write(i)
```

Si nos fijamos, en cada iteración actualizamos el valor de i incrementándolo en 2 unidades.

Otro detalle, no utilizamos las llaves englobando las instrucciones del bucle FOR porque sólo tiene una sentencia y en este caso no es obligado, tal como pasaba con las instrucciones del IF.

Si queremos contar descendentemente del 343 al 10 utilizaríamos este bucle.

```
for (i=343;i>=10;i--)
   document.write(i)
}
```

En este caso decrementamos en una unidad la variable i en cada iteración.

Ejemplo

Vamos a hacer una pausa para asimilar el bucle for con un ejercicio que no encierra ninguna dificultad si hemos entendido el funcionamiento del bucle.

Se trata de hacer un bucle que escriba en una página web los encabezamientos desde <H1> hasta <H6> con un texto que ponga "Encabezado de nivel x".

Lo que deseamos escribir en una página web mediante Javascript es lo siguiente:

```
<H1>Encabezado de nivel 1</H1>
<H2>Encabezado de nivel 2</H2>
<H3>Encabezado de nivel 3</H3>
<H4>Encabezado de nivel 4</H4>
<H5>Encabezado de nivel 5</H5>
<H6>Encabezado de nivel 6</H6>
```

Para ello tenemos que hacer un bucle que empiece en 1 y termine en 6 y en cada iteración escribiremos el encabezado que toca.

```
for (i=1;i<=6;i++) {
   document.write("<H" + i + ">Encabezado de nivel " + i + "</H" + i + ">")
}
```

Bucles WHILE y DO WHILE

Veamos ahora los dos tipos de bucles WHILE que podemos utilizar en Javascript y los usos de cada uno.

Bucle WHILE

Estos bucles se utilizan cuando queremos repetir la ejecución de unas sentencias un número indefinido de veces, siempre que se cumpla una condición. Se más sencillo de comprender que el bucle FOR, pues no incorpora en la misma línea la inicialización de las variables su condición para seguir ejecutándose y su actualización. Sólo se indica, como veremos a continuación, la condición que se tiene que cumplir para que se realice una iteración.

```
while (condición){
   sentencias a ejecutar
}
```

Un ejemplo de código donde se utiliza este bucle se puede ver a continuación.

```
var color = ""
while (color != "rojo")
   color = dame un color
}
```

Este es un ejemplo de lo más sencillo que se puede hacer con un bucle while. Lo que hace es pedir que el usuario introduzca un color mientras que el color no sea rojo. Para ejecutar un bucle como este primero tenemos que inicializar la variable que vamos utilizar en la condición de iteración del bucle. Con la variable inicializada podemos escribir el bucle, que comprobará para ejecutarse que el la variable color sea distinto de "rojo". En cada iteración del bucle se pide un nuevo color al usuario para actualizar la variable color y se termina la iteración, con lo que retornamos al principio del bucle, donde tenemos que volver a evaluar si lo que hay en la variable color es "rojo" y así sucesivamente mientras que no se haya introducido como color el texto "rojo". Obviamente la expresión dame un color no es Javascript, pero como no sabemos todavía cómo escribir eso en Javascript es mejor verlo más adelante.

Bucle DO...WHILE

Es el último de los bucles que hay en Javascript. Se utiliza generalmente cuando no sabemos cuantas veces se habrá de ejecutar el bucle, igual que el bucle WHILE, con la diferencia de que sabemos seguro que el bucle por lo menos se ejecutará una vez.

Este tipo de bucle se introdujo en Javascript 1.2, por lo que no todos los navegadores los soportan, sólo los de versión 4 o superior. En cualquiuer caso, cualquier código que quieras escribir con DO...WHILE se puede escribir también utilizando un bucle WHILE, con lo que en navegadores antiguos deberás traducir tu bucle DO...WHILE por un bucle WHILE.

La sintaxis es la siguiente.

```
do {
   sentencias del bucle
} while (condición)
```

El bucle se ejecuta siempre una vez y al final se evalúa la condición para decir si se ejecuta otra vez el bucle o se termina su ejecución.

Veamos el ejemplo que escribimos para un bucle WHILE en este otro tipo de bucle.

```
var color
do {
   color = dame un color
} while (color != "rojo")
```

Este ejemplo funciona exactamente igual que el anterior, excepto que no tuvimos que inicializar la variable color antes de introducirnos en el bucle. Pide un color mientras que el color introducido es distinto que "rojo".

Ejemplo

Vamos a ver a continuación un ejemplo más práctico sobre cómo trabajar con un bucle WHILE. Como resulta muy dificil hacer ejemplos prácticos con lo poco que sabemos sobre Javascript, vamos a adelantar una instrucción que aun no conocemos.

En este ejemplo vamos a declarar una variable e inicializarla a 0. Luego iremos sumando a esa variable un número aleatorio del 1 al 100 hasta que sumemos 1.000 o más, imprimiendo el valor de la variable suma después de cada operación. Será necesario utilizar el bucle WHILE porque no sabemos exactamente el número de iteraciones que tendremos que realizar.

```
var suma = 0
while (suma < 1000){
   suma += parseInt(Math.random() * 100)
   document.write (suma + "<br>")
}
```

Suponemos que por lo que respecta al bucle WHILE no habrá problemas, pero donde si que puede haberlos es en la sentencia utilizada para tomar un número aleatorio. Sin embargo, no es necesario explicar aquí la sentencia porque lo tenemos planeado hacer más adelante.

Break y continue

De manera adicional al uso de las distintas estructuras de bucle se pueden utilizar dos instrucciones para

- Detener la ejecución de un bucle y salirse de él
- Detener la iteración actual y volver al principio del bucle.

Son las instrucciones break y continue.

Break

Se detiene un bucle utilizando la palabra break. Detener un bucle significa salirse de él y dejarlo todo como está para continuar con el flujo del programa inmediatamente después del bucle.

```
for (i=0;i<10;i++){
   document.write (i)
   escribe = dime si continúo
   if (escribe == "no")
      break
}
```

Este ejemplo escribe los números del 0 al 9 y en cada iteración del bucle pregunta al usuario si desea continuar. Si el usuario dice cualquier cosa continua excepto cuando dice "no" que entonces se sale del bucle y deja la cuenta por donde se había quedado.

Continue

Sirve para volver al principio del bucle en cualquier momento, sin ejecutar las líneas que haya por debajo de la palabra continue.

```
var i=0
while (i<7){
   incrementar = dime si incremento
   if (incrementar == "no")
      continue
   i++
}
```

Este ejemplo, en condiciones normales contaría hasta desde i=0 hasta i=7, pero cada vez que se ejecuta el bucle pregunta al usuario si desea incrementar la variable o no. Si introduce "no" se ejecuta la sentencia continue, con lo que se vuelve al principio del bucle sin llegar a incrementar en 1 la variable i, ya que se ignoran las sentencia que hayan por debajo del continue.

Ejemplo

Un ejemplo más práctico sobre estas instrucciones se puede ver a continuación. Se trata de un bucle FOR planeado para llegar hasta 1.000 pero que lo vamos a parar con break cuando lleguemos a 333.

```
for (i=0;i<=1000;i++){
   document.write(i + "<br>")
   if (i==333)
      break;
}
```

Bucles anidados en Javascript

Anidar un bucle consiste en meter ese bucle dentro de otro. La anidación de bucles es necesaria para hacer determinados procesamientos un poco más complejos que los que hemos visto en los ejemplos anteriores y seguro que en vuestra experiencia como programadores los habréis utilizado ya o los utilizareis en un futuro.

Un bucle anidado tiene una estructura como la que sigue. Vamos a tratar de explicarlo a la vista de estas líneas:

```
for (i=0;i<10;i++){
   for (j=0;j<10;j++) {
   document.write(i + "-" + j)
   }
}
```

La ejecución funcionará de la siguiente manera. Para empezar se inicializa el primer bucle, con lo que la variable i valdrá 0 y a continuación se inicializa el segundo bucle, con lo que la variable j valdrá también 0. En cada iteración se imprime el valor de la variable i, un guión ("-") y el valor de la variable j, como las dos variables valen 0, se imprimirá el texto "0-0" en la página web.

El bucle que está anidado (más hacia dentro) es el que más veces se ejecuta, en este ejemplo, para cada iteración del bucle más externo el bucle anidado se ejecutará por completo una vez, es decir, hará sus 10 iteraciones. En la página web se escribirían estos valores, en la primera iteración del bucle

externo y desde el principio:

```
0-0
0-1
0-2
0-3
0-4
0-5
0-6
0-7
0-8
0-9
```

Para cada iteración del bucle externo se ejecutarán las 10 iteraciones del bucle interno o anidado. Hemos visto la primera iteración, ahora vamos a ver las siguientes iteraciones del bucle externo. En cada una acumula una unidad en la variable i, con lo que saldrían estos valores.

```
1-0
1-1
1-2
1-3
1-4
1-5
1-6
1-7
1-8
1-9
```

Y luego estos.

```
2-0
2-1
2-2
2-3
2-4
2-5
2-6
2-7
2-8
2-9
```

Así hasta que se terminen los dos bucles, que sería cuando se alcanzase el valor 9-9.

Veamos un ejemplo muy parecido al anterior, aunque un poco más útil. Se trata de imprimir en la página las todas las tablas de multiplicar. Del 1 al 9, es decir, la tabla del 1, la del 2, del 3...

```
for (i=1;i<10;i++){
   document.write("<br><b>La tabla del " + i + ":</b><br>")
   for (j=1;j<10;j++) {
      document.write(i + " x " + j + ": ")
      document.write(i*j)
      document.write("<br>")
   }
}
```

Con el primer bucle controlamos la tabla actual y con el segundo bucle la desarrollamos. En el primer bucle escribimos una cabecera, en negrita, indicando la tabla que estamos escribiendo, primero la del 1 y luego las demás en orden ascendente hasta el 9. Con el segundo bucle escribo cada uno de los valores de cada tabla.

Funciones en Javascript

Ahora vamos a ver un tema muy importante, sobretodo para los que no han programado nunca y con Javascript están dando sus primeros pasos en el mundo de la programación ya que veremos un concepto nuevo, el de función, y los usos que tiene. Para los que ya conozcan el concepto de función también será un capítulo util, pues también veremos la sintaxis y funcionamiento de las funciones en Javascript.

Qué es una función

A la hora de hacer un programa ligeramente grande existen determinados procesos que se pueden

concebir de forma independiente, y que son más sencillos de resolver que el problema entero. Además, estos suelen ser realizados repetidas veces a lo largo de la ejecución del programa. Estos procesos se pueden agrupar en una función, definida para que no tengamos que repetir una y otra vez ese código en nuestros scripts, sino que simplemente llamamos a la función y ella se encarga de hacer todo lo que debe.

Así que podemos ver una función como una serie de instrucciones que englobamos dentro de un mismo proceso. Este proceso se podrá luego ejecutar desde cualquier otro sitio con solo llamarlo. Por ejemplo, en una página web puede haber una función para cambiar el color del fondo y desde cualquier punto de la página podríamos llamarla para que nos cambie el color cuando lo deseemos.

Las funciones se utilizan constantemente, no sólo las que escribes tu, sino también las que ya están definidas en el sistema, pues todos los lenguajes de programación tienen un montón de funciones para realizar procesos habituales como por ejemplo obtener la hora, imprimir un mensaje en la pantalla o convertir variables de un tipo a otro. Ya hemos visto alguna función en nuestros sencillos ejemplos anteriores cuando hacíamos un document.write() en realidad estabamos llamando a la función write() asociada al documento de la página que escribe un texto en la página. En los capítulos de funciones vamos primero a ver cómo realizar nuestras propias funciones y cómo llamarlas luego. A lo largo del libro veremos muchas de las funciones definidas en Javascript que debemos utilizar para realizar distintos tipos de acciones habituales.

Cómo se escribe una función

Una función se debe definir con una sintaxis especial que vamos a conocer a continuación.

```
function nombrefuncion (){
  instrucciones de la función
  ...
}
```

Primero se escribe la palabra function, reservada para este uso. Seguidamente se escribe el nombre de la función, que como los nombres de variables puede tener números, letras y algún carácter adicional como en guión bajo. A continuación se colocan entre llaves las distintas instrucciones de la función. Las llaves en el caso de las funciones no son opcionales, además es útil colocarlas siempre como se ve en el ejemplo, para que se vea fácilmente la estructura de instrucciones que engloba la función.

Veamos un ejemplo de función para escribir en la página un mensaje de bienvenida dentro de etiquetas <H1> para que quede más resaltado.

```
function escribirBienvenida(){
  document.write("<H1>Hola a todos</H1>")
}
```

Simplemente escribe en la página un texto, es una función tan sencilla que el ejemplo no expresa suficientemente el concepto de función, pero ya veremos otras más complejas. Las etiquetas H1 no se escriben en la página, sino que son interpretadas como el significado de la misma, en este caso que escribimos un encabezado de nivel 1. Como estamos escribiendo en una página web, al poner etiquetas HTML se interpretan como lo que son.

Cómo llamar a una función

Cuando se llaman a las funciones Para ejecutar una función la tenemos que llamar en cualquier parte de la página, con eso conseguiremos que se ejecuten todas las instrucciones que tiene la función entre las dos llaves. Para ejecutar la función utilizamos su nombre seguido de los paréntesis.

```
NombreDeLaFuncion()
```

Dónde colocamos las funciones

En principio, podemos colocar las funciones en cualquier parte de la página, siempre entre etiquetas <SCRIPT>, claro está. No obstante existe una limitación a la hora de colocarla con relación a los lugares desde donde se la llame. Lo más normal es colocar la función antes de cualquier llamada a la misma y así seguro que nunca nos equivocaremos.

En concreto, la función se debe definir en el bloque <SCRIPT> donde esté la llamada a la función, aunque es indiferente si la llamada se encuentra antes o después la función, dentro del mismo bloque <SCRIPT>.

```
<SCRIPT>
miFuncion()
function miFuncion(){
```

```
   //hago algo...
   document.write("Esto va bien")
}
</SCRIPT>
```

Este ejemplo funciona correctamente porque la función está declarada en el mismo bloque que su llamada.

También es válido que la función se encuentre en un bloque <SCRIPT> anterior al bloque donde está la llamada.

```
<HTML>
<HEAD>
  <TITLE>MI PÁGINA</TITLE>
<SCRIPT>
function miFuncion(){
   //hago algo...
   document.write("Esto va bien")
}
</SCRIPT>
</HEAD>
<BODY>

<SCRIPT>
miFuncion()
</SCRIPT>

</BODY>
</HTML>
```

Vemos un código completo sobre cómo podría ser una página web donde las funciones están en la cabecera. Un lugar muy bueno donde colocarlas, porque se supone que en la cabecera no se van a utilizar todavía y siempre podremos disfrutar de ellas en el cuerpo porque ya han sido declaradas seguro.

Esto último en cambio sería un error.

Lo que será un error es una llamada a una función que se encuentra declarada en un bloque <SCRIPT> posterior.

```
<SCRIPT>
miFuncion()
</SCRIPT>

<SCRIPT>
function miFuncion(){
   //hago algo...
   document.write("Esto va bien")
}
</SCRIPT>
```

Parámetros de las funciones

Las estructuras que hemos visto anteriormente sobre funciones no son las únicas que debemos aprender para manejarlas en toda su potencia. Las funciones también tienen una entrada y una salida, que se pueden utilizar para recibir y devolver datos.

Parámetros

Los parámetros se usan para mandar valores a la función, con los que ella trabajará para realizar las acciones. Son los valores de entrada que recibe una función. Por ejemplo, una función que realizase una suma de dos números tendría como parámetros a esos dos números. Los dos números son la entrada, así como la salida sería el resultado, pero eso lo veremos más tarde.

Veamos un ejemplo anterior en el que creábamos una función para mostrar un mensaje de bienvenida en la página web, pero al que ahora le vamos a pasar un parámetro que contendrá el nombre de la persona a la que hay que saludar.

```
function escribirBienvenida(nombre){
   document.write("<H1>Hola " + nombre + "</H1>")
}
```

Como podemos ver en el ejemplo, para definir en la función un parámetro tenemos que poner el nombre de la variable que va a almacenar el dato que le pasemos. Esa variable, que en este caso se llama nombre, tendrá como valor el dato que le pasemos a la función cuando la llamemos, además, la variable tendrá vida durante la ejecución de la función y dejará de existir cuando la función termine su ejecución.

Para llamar a una función que tiene parámetros se coloca entre paréntesis el valor del parámetro. Para llamar a la función del ejemplo habría que escribir:

```
escribirBienvenida("Alberto García")
```

Al llamar a la función así, el parámetro nombre toma como valor "Alberto García" y al escribir el saludo por pantalla escribirá "Hola Alberto García" entre etiquetas <H1>.

Los parámetros pueden recibir cualquier tipo de datos, numérico, textual, boleano o un objeto. Realmente no especificamos el tipo del parámetro, por eso debemos tener un cuidado especial al definir las acciones que realizamos dentro de la función y al pasarle valores a la función para asegurarnos que todo es consecuente con los tipos de nuestras variables o parámetros.

Múltiples parámetros

Una función puede recibir tantos parámetros como queramos y para expresarlo se colocan los parámetros separados por comas dentro de los paréntesis. Veamos rápidamente la sintaxis para que la función de antes reciba dos parámetros, el primero el nombre al que saludar y el segundo el color del texto.

```
function escribirBienvenida(nombre,colorTexto){
   document.write("<FONT color=" + colorTexto + ">)
   document.write("<H1>Hola " + nombre + "</H1>")
   document.write("</FONT>")
}
```

Llamaríamos a la función con esta sintaxis. Entre los paréntesis colocaremos los valores de los parámetros.

```
var miNombre = "Pepe"
var miColor = "red"
escribirBienvenida(miNombre,miColor)
```

He colocado entre los paréntesis dos variables en lugar de dos textos entrecomillados. Cuando colocamos variables entre los parámetros en realidad lo que estamos pasando a la función son los valores que contienen las variables y no las mismas variables.

Parámetros se pasan por valor

Al hilo del uso de parámetros en nuestros programas Javascript tenemos que indicar que los parámetros de las funciones se pasan por valor. Esto quiere decir que aunque modifiquemos un parámetro en una función la variable original que habíamos pasado no cambiará su valor. Se puede ver fácilmente con un ejemplo.

```
function pasoPorValor(miParametro){
   miParametro = 32
   document.write("he cambiado el valor a 32")
}
var miVariable = 5
pasoPorValor(miVariable)
document.write ("el valor de la variable es: " + miVariable)
```

En el ejemplo tenemos una función que recibe un parámetro y que modifica el valor del parámetro asignándole el valor 32. También tenemos una variable, que inicializamos a 5 y posteriormente llamamos a la función pasándole esta variable como parámetro. Como dentro de la función modificamos el valor del parámetro podría pasar que la variable original cambiase de valor, pero como los parámetros no modifican el valor original de las variables esta no cambia de valor. De este modo, al imprimir en pantalla el valor de miVariable se imprimirá el número 5, que es el valor original de la variable, en lugar de 32 que era el valor col el que habíamos actualizado el parámetro.

En javascript sólo se pueden pasar las variables por valor.

Valores de retorno

Las funciones también pueden retornar valores, de modo que al ejecutar la función se podrá realizar

acciones y dar un valor como salida. Por ejemplo, una función que calcula el cuadrado de un número tendrá como entrada -tal como vimos- a ese número y como salida tendrá el valor resultante de hallar el cuadrado de ese número. Una función que devuelva el día de la semana tendría como salida en día de la semana.

Veamos un ejemplo de función que calcula la media de dos números. La función recibirá los dos números y retornará el valor de la media.

```
function media(valor1,valor2){
   var resultado
   resultado = (valor1 + valor2) / 2
   return resultado
}
```

Para especificar el valor que retornará la función se utiliza la palabra return seguida de el valor que se desea devolver. En este caso se devuelve el contenido de la variable resultado, que contiene la media de los dos números.

Para recibir los valores que devuelve una función se coloca el operador de asignación =. Para ilustrar esto se puede ver este ejemplo, que llamará a la función media() y guardará el resultado de la media en una variable para luego imprimirla en la página.

```
var miMedia
miMedia = media(12,8)
document.write (miMedia)
```

Múltiples return

En una misma función podemos colocar más de un return. Lógicamente sólo vamos a poder retornar una cosa, pero dependiendo de lo que haya sucedido en la función podrá ser de un tipo u otro, con unos datos u otros.

En esta función podemos ver un ejemplo de utilización de múltiples return. Se trata de una función que devuelve un 0 si el parámetro recibido era par y el valor del parámetro si este era impar.

```
function multipleReturn(numero){
   var resto = numero % 2
   if (resto == 0)
      return 0
   else
      return numero
}
```

Para averiguar si un número es par hallamos el resto de la división al dividirlo entre 2. Si el resto es cero es que era par y devolvemos un 0, en caso contrario -el número es impar- devolvemos el parámetro recibido.

Ámbito de las variables en funciones

Dentro de las funciones podemos declarar variables, incluso los parámetros son como variables que se declaran en la cabecera de la función y que se inicializan al llamar a la función. Todas las variables declaradas en una función son locales a esa función, es decir, solo tendrán validez durante la ejecución de la función.

Podemos declarar variables en funciones que tengan el mismo nombre que una variable global a la página. Entonces, dentro de la función la variable que tendrá validez es la variable local y fuera de la función tendrá validez la variable global a la página.

En cambio, si no declaramos las variables en las funciones se entenderá por javascript que estamos haciendo referencia a una variable global a la página, de modo que si no está creada la variable la crea, pero siempre global a la página en lugar de local a la función.

Arrays en Javascript

En los lenguajes de programación existen estructuras de datos especiales que nos sirven para guardar información más compleja que simples variables. Una estructura típica en todos los lenguajes es el Array, que es como una variable donde podemos introducir varios valores, en lugar de solamente uno como ocurre con la variables normales.

Los arrays nos permiten guardar varias variables y acceder a ellas de manera independiente, es como tener una variable con distintos compartimentos donde podemos introducir datos distintos. Para ello

utilizamos un índice que nos permite especificar el compartimiento o posición a la que nos estamos refiriendo.

Los arrays se introdujeron en versiones Javascript 1.1 o superiores, es decir, solo los podemos utilizar a partir de los navegadores 3.0. Para navegadores antiguos se puede simular el array utilizando sintaxis de programación orientada a objetos, pero dada la complejidad de esta tarea, por lo menos en el momento en que nos encontramos y las pocas ocasiones que los deberemos utilizar vamos a ver cómo utilizar el auténtico array de Javascript.

Creación de Arrays

El primer paso para utilizar un array es crearlo. Para ello utilizamos un objeto Javascript ya implementado en el navegador. Veremos en adelante un tema para explicar lo que es la orientación a objetos, aunque no será necesario para poder entender el uso de los arrays. Esta es la sentencia para crear un objeto array:

```
var miArray = new Array()
```

Esto crea un array en la página que esta ejecutándose. El array se crea sin ningún contenido, es decri, no tendrá ninguna casilla o compartimiento creado. También podemos crear el array especificando el número de compartimentos que va a tener.

```
var miArray = new Array(10)
```

En este caso indicamos que el array va a tener 10 posiciones, es decir, 10 casillas donde guardar datos.

Es importante que nos fijemos que la palabra Array en código Javascript se escribe con la primera letra en mayúscula. Como en Javascript las mayúsculas y minúsculas si que importan, si lo escribimos en minúscula no funcionará.

Tanto se indique o no el número de casillas del array, podemos introducir en el array cualquier dato. Si la casilla está creada se introduce simplemente y si la casilla no estaba creada se crea y luego se introduce el dato, con lo que el resultado final es el mismo. Esta creación de casillas es dinámica y se produce al mismo tiempo que los scripts se ejecutan. Veamos a continuación cómo introducir valores en nuestros arrays.

```
miArray[0] = 290
miArray[1] = 97
miArray[2] = 127
```

Se introducen indicando entre corchetes el índice de la posición donde queríamos guardar el dato. En este caso introducimos 290 en la posición 0, 97 en la posición 1 y 127 en la 2. Los arrays empiezan siempre en la posición 0, así que un array que tenga por ejemplo 10 posiciones, tendrá casillas de la 0 a la 9. Para recoger datos de un array lo hacemos igual: poniendo entre corchetes el índice de la posición a la que queremos acceder. Veamos cómo se imprimiría en la pantalla el contenido de un array.

```
var miArray = new Array(3)

miArray[0] = 155
miArray[1] = 4
miArray[2] = 499

for (i=0;i<3;i++){
   document.write("Posición " + i + " del array: " + miArray[i])
   document.write("<br>")
}
```

Hemos creado un array con tres posiciones, luego hemos introducido un valor en cada una de las posiciones del array y finalmente las hemos impreso. En general, el recorrido por arrays para imprimir sus posiciones o cualquier otra cosa se hace utilizando bucles. En este caso utilizamos un bucle FOR que va desde el 0 hasta el 2.

Tipos de datos en los arrays

En las casillas de los arrays podemos guardar datos de cualquier tipo. Podemos ver un array donde introducimos datos de tipo caracter.

```
miArray[0] = "Hola"
miArray[1] = "a"
miArray[2] = "todos"
```

Incluso, en Javascript podemos guardar distintos tipos de datos en las casillas de un mismo array. Es

decir, podemos introducir números en unas casillas, textos en otras, boleanos o cualquier otra cosa que deseemos.

```
miArray[0] = "desarrolloweb.com"
miArray[1] = 1275
miArray[1] = 0.78
miArray[2] = true
```

Longitud de los arrays

Todos los arrays en javascript, aparte de almacenar el valor de cada una de sus posiciones también almacenan el número de posiciones que tienen. Para ello utilizan una propiedad del objeto array, la propiedad length. Ya veremos en objetos qué es una propiedad, pero para nuestro caso podemos imaginarnos que es como una variable, adicional a las posiciones, que almacena un número igual al número de casillas del array.

Para acceder a una propiedad de un objeto se ha de utilizar el operador punto. Se escribe el nombre del array que queremos acceder al número de posiciones que tiene, sin corchetes ni paréntesis, seguido de un punto y la palabra length.

```
var miArray = new Array()

miArray[0] = 155
miArray[1] = 499
miArray[2] = 65

document.write("Longitud del array: " + miArray.length)
```

Este código imprimiría en pantalla el número de posiciones del array, que en este caso es 3. Recordamos que un array con 3 posiciones abarca desde la posición 0 a la 2.

Es muy habitual que se utilice la propiedad length para poder recorrer un array por todas sus posiciones. Para ilustrarlo vamos a ver un ejemplo de recorrido por este array para mostrar sus valores.

```
for (i=0;i<miArray.length;i++){
   document.write(miArray[i])
}
```

Hay que fijarse que el bucle for se ejecuta siempre que i valga menos que la longitud del array, extraída de su propiedad length.

El siguiente ejemplo nos servirá para conocer mejor los recorridos por los arrays, el funcionamiento de la propiedad length y la creación dinámica de nuevas posiciones. Vamos a crear un array con 2 posiciones y rellenar su valor. Posteriormente introduciremos un valor en la posición 5 del array. Finalmente imprimiremos todas las posiciones del array para ver lo que pasa.

```
var miArray = new Array(2)

miArray[0] = "Colombia"
miArray[1] = "Estados Unidos"

miArray[5] = "Brasil"

for (i=0;i<miArray.length;i++){
   document.write("Posición " + i + " del array: " + miArray[i])
   document.write("<br>")
}
```

El ejemplo es sencillo. Se puede apreciar que hacemos un recorrido por el array desde 0 hasta el número de posiciones del array (indicado por la propiedad length). En el recorrido vamos imprimiendo el número de la posición seguido del contenido del array en esa posición. Pero podemos tener una duda al preguntarnos cuál será el número de elementos de este array, ya que lo habíamos declarado con 2 y luego le hemos introducido un tercero en la posición 5. Al ver la salida del programa podremos contestar nuestras preguntas. Será algo parecido a esto:

```
Posición 0 del array: Colombia
Posición 1 del array: Estados Unidos
Posición 2 del array: null
Posición 3 del array: null
Posición 4 del array: null
Posición 5 del array: Brasil
```

Se puede ver claramente que el número de posiciones es 6, de la 0 a la 5. Lo que ha ocurrido es que al introducir un dato en la posición 5, todas las casillas que no estaban creadas hasta la quinta se crean también.

Las posiciones de la 2 a la 4 están sin inicializar. En este caso nuestro navegador ha escrito la palabra null para expresar esto, pero otros navegadores podrán utilizar la palabra undefined. Ya veremos más adelante qué es este null y dónde lo podemos utilizar, lo importante ahora es que comprendas cómo trabajan los arrays y los utilices correctamente.

Arrays multidimensionales

Los arrays multidimensionales son un estructuras de datos que almacenan los valores en más de una dimensión. Los arrays que hemos visto hasta ahora almacenan valores en una dimensión, por eso para acceder a las posiciones utilizamos tan solo un índice. Los arrays de 2 dimensiones guardan sus valores, por decirlo de alguna manera, en filas y columnas y por ello necesitaremos dos índices para acceder a cada una de sus posiciones.

Dicho de otro modo, un array multidimensional es como un contenedor que guardara más valores para cada posición, es decir, como si los elementos del array fueran a su vez otros arrays.

En Javascript no existe un auténtico objeto array-multidimensinal. Para utilizar estas estructuras podremos definir arrays que donde en cada una de sus posiciones habrá otro array. En nuestros programas podremos utilizar arrays de cualquier dimensión, veremos a continuación cómo trabajar con arrays de dos dimensiones, que serán los más comunes.

En este ejemplo vamos a crear un array de dos dimensiones donde tendremos por un lado ciudades y por el otro la temperatura media que hace en cada una durante de los meses de invierno.

```
var temperaturas_medias_ciudad0 = new Array(3)
temperaturas_medias_ciudad0[0] = 12
temperaturas_medias_ciudad0[1] = 10
temperaturas_medias_ciudad0[2] = 11

var temperaturas_medias_ciudad1 = new Array (3)
temperaturas_medias_ciudad1[0] = 5
temperaturas_medias_ciudad1[1] = 0
temperaturas_medias_ciudad1[2] = 2

var temperaturas_medias_ciudad2 = new Array (3)
temperaturas_medias_ciudad2[0] = 10
temperaturas_medias_ciudad2[1] = 8
temperaturas_medias_ciudad2[2] = 10
```

Con las anteriores líneas hemos creado tres arrays de 1 dimensión y tres elementos, como los que ya conocíamos. Ahora crearemos un nuevo array de tres elementos e introduciremos dentro de cada una de sus casillas los arrays creados anteriormente, con lo que tendremos un array de arrays, es decir, un array de 2 dimensiones.

```
var temperaturas_cuidades = new Array (3)
temperaturas_cuidades[0] = temperaturas_medias_ciudad0
temperaturas_cuidades[1] = temperaturas_medias_ciudad1
temperaturas_cuidades[2] = temperaturas_medias_ciudad2
```

Vemos que para introducir el array entero hacemos referencia al mismo sin paréntesis ni corchetes, sino sólo con su nombre. El array temperaturas_cuidades es nuestro array bidimensinal.

También es interesante ver cómo se realiza un recorrido por un array de dos dimensiones. Para ello tenemos que hacer un recorrido por cada una de las casillas del array bidimensional y dentro de estas hacer un nuevo recorrido para cada una de sus casillas internas. Es decir, un recorrido por un array dentro de otro.

El método para hacer un recorrido dentro de otro es colocar un bucle dentro de otro, lo que se llama un bucle anidado. Puede resultar complicado el hacer un bucle anidado, pero nosotros ya hemos tenido ocasión de practicar en un capítulo anterior. Así que en este ejemplo vamos a meter un bucle FOR dentro de otro. Además, vamos a escribir los resultados en una tabla, lo que complicará un poco el script, pero podremos ver cómo construir una tabla desde javascript a medida que realizamos el recorrido anidado al bucle.

```
document.write("<table width=200 border=1 cellpadding=1 cellspacing=1>");
for (i=0;i<temperaturas_cuidades.length;i++){
```

```
    document.write("<tr>")
    document.write("<td><b>Ciudad " + i + "</b></td>")
    for (j=0;j<temperaturas_cuidades[i].length;j++){
      document.write("<td>" + temperaturas_cuidades[i][j] + "</td>")
    }
    document.write("</tr>")
}
document.write("</table>")
```

Este script resulta un poco más complejo que los vistos anteriormente. La primera acción consiste en escribir la cabecera de la tabla, es decir, la etiqueta <TABLE> junto con sus atributos. Con el primer bucle realizamos un recorrido a la primera dimensión del array y utilizamos la variable i para llevar la cuenta de la posición actual. Por cada iteración de este bucle escribimos una fila y para empezar la fila abrimos la etiqueta <TR>. Además, escribimos en una casilla el numero de la ciudad que estamos recorriendo en ese momento. Posteriormente ponemos otro bucle que va recorriendo cada una de las casillas del array en su segunda dimensión y escribimos la temperatura de la ciudad actual en cada uno de los meses, dentro de su etiqueta <TD>. Una vez que acaba el segundo bucle se han impreso las tres temperaturas y por lo tanto la fila está terminada. El primer bucle continúa repitiéndose hasta que todas las ciudades están impresas y una vez terminado cerramos la tabla.

Inicialización de arrays

Para terminar con el tema de los arrays vamos a ver una manera de inicializar sus valores a la vez que lo declaramos, así podemos realizar de una manera más rápida el proceso de introducir valores en cada una de las posiciones del array.

El método normal de crear un array vimos que era a través del objeto Array, poniendo entre paréntesis el número de casillas del array o no poniendo nada, de modo que el array se crea sin ninguna posición. Para introducir valores a un array se hace igual, pero poniendo entre los paréntesis los valores con los que deseamos rellenar las casillas separados por coma. Veámoslo con un ejemplo que crea un array con los nombres de los días de la semana.

```
var diasSemana = new Array("Lunes","Martes","Miércoles,","Jueves","Viernes","Sábado","Domingo")
```

El array se crea con 7 casillas, de la 0 a la 6 y en cada casilla se escribe el dia de la semana correspondiente (Entre comillas porque es un texto).

Ahora vamos a ver algo más complicado, se trata de declarar el array bidimensional que utilizamos antes para las temperaturas de las ciudades en los meses en una sola línea, introduciendo los valores a la vez.

```
var temperaturas_cuidades = new Array(new Array (12,10,11), new Array(5,0,2),new Array(10,8,10))
```

En el ejemplo introducimos en cada casilla del array otro array que tiene como valores las temperaturas de una ciudad en cada mes.

Pausa y consejos Javascript

Hasta aquí hemos visto la mayor parte de la sintaxis y forma de funcionar de el lenguaje Javascript. Ahora podemos escribir scripts simples que hagan uso de variables, funciones, arrays, estructuras de control y toda clase de operadores. Con todo esto conocemos el lenguaje, pero aun queda mucho camino por delante para dominar Javascript y saber hacer todos los efectos posibles en páginas web, que en definitiva es lo que interesa.

De todos modos, este manual nos lo hemos tomado con mucha calma, con intención de que las personas que no estén familiarizadas con el mundo de la programación puedan también tener acceso al lenguaje y aprendan las técnicas básicas que permitirán afrontar futuros retos en la programación. Esperamos entonces que la marcha del manual haya sido provechosa para los más inexpertos y que ahora puedan entender con facilidad las siguientes lecciones o ejemplos, ya que conocen las bases sobre las que están implementados.

Antes de acabar, vamos a dar una serie de consejos a seguir a la hora de programar nuestros scripts que nos pueden ayudar a hacernos la vida más fácil. Algunos consejos son nuevos e importantes, otros se han señalado con anterioridad, pero sin duda viene bien recordar.

Distintos navegadores

Lo primero importante en señalar es que Javascript es un lenguaje muy dinámico y que ha sido implementado poco a poco, a medida que salían nuevos navegadores. Si pensamos en el Netscape 2, el

primer navegador que incluía Javascript, y el Netscape 6 o Internet Explorer 6 nos daremos cuenta que han pasado un mundo de novedades entre ellos. Javascript ha cambiado por lo menos tanto como los navegadores y eso representa un problema para los programadores, porque tienen que estar al tanto de las distintas versiones y la manera de funcionar que tienen.

Las bases de javascript, sobre las que hemos hablado hasta ahora, no han cambiado prácticamente nada. Sólo en algunas ocasiones, que hemos señalado según las conocíamos -como los arrays por ejemplo-, el lenguaje ha evolucionado un poco. Sin embargo, a medida que nuevas tecnologías como el DHTML se desarrollaban, los navegadores han variado su manera de manejarlas.

Nuestro consejo es que estéis al tanto de las cosas que funcionan en unos u otros sistemas y que programéis para que vuestras páginas sean compatibles en el mayor número de navegadores. Para procurar esto último es muy aconsejable probar las páginas en varias plataformas distintas. También es muy útil dejar de lado los últimos avances, es decir, no ir a la última, sino ser un poco conservadores, para que las personas que han actualizado menos el browser puedan también visualizar los contenidos.

Consejos para hacer un código sencillo y fácil de mantener

Ahora vamos a dar una serie de consejos para que el código de nuestros scripts sea más claro y libre de errores. Muchos de ellos ya los hemos señalado y somos libres de aplicarlos o no, pero si lo hacemos nuestra tarea como programadores puede ser mucho más agradable, no sólo hoy, sino también el día que tengamos que revisar los scripts en su mantenimiento.

- Utiliza comentarios habitualmente para que lo que estás haciendo en los scripts pueda ser recordado por ti y cualquier persona que tenga que leerlos más adelante.

- Ten cuidado con el ámbito de las variables, recuerda que existen variables globales y locales, que incluso pueden tener los mismos nombres y conoce en cada momento la variable que tiene validez.

- Escribe un código con suficiente claridad, que se consigue con saltos de línea después de cada instrucción, un sangrado adecuado (poner márgenes a cada línea para indicar en qué bloque están incluidas), utilizar las llaves {} siempre, aunque no sean obligatorias y en general mantener siempre el mismo estilo a la hora de programar.

- Aplica un poco de consistencia al manejo de variables. Declara las variables con var. No cambies el tipo del dato que contiene (si era numérica no pongas luego texto, por ejemplo). Dales nombres comprensibles para saber en todo momento qué contienen. Incluso a la hora de darles nombre puedes aplicar una norma, que se trata de que indiquen en sus nombres el tipo de dato que contienen. Por ejemplo, las variables de texto pueden empezar por una s (de String), las variables numéricas pueden empezar por una n o las boleanas por una b.

- Prueba tus scripts cada poco a medida que los vas programando. Puedes escribir un trozo de código y probarlo antes de continuar para ver que todo funciona correctamente. Es más fácil encontrar los errores de código en bloques pequeños que en bloques grandes.

Tratamiento de errores en javascript

Para acabar la primera parte del manual de javascript vamos a explicar los errores comunes que podemos cometer y cómo evitarlos y depurarlos. Además veremos una pequeña conclusión a esta parte del manual.

Errores comunes

Cuando ejecutamos los scripts pueden ocurrir dos tipos de errores de sintaxis o de ejecución, los vemos a continuación.

Errores de sintaxis ocurren por escribir de manera errónea las líneas de código, equivocarse a la hora de escribir el nombre de una estructura, utilizar incorrectamente las llaves o los paréntesis o cualquier cosa similar. Estos errores los indica javascript a medida que está cargando los scripts en memoria, lo que hace siempre antes de ejecutarlos, como se indicó en los primeros capítulos. Cuando el analizador sintáctico de javascript detecta un error de estos se presenta el mensaje de error.

Errores de ejecución ocurren cuando se están ejecutando los scripts. Por ejemplo pueden ocurrir cuando llamamos a una función que no ha sido definida. javascript no indica estos errores hasta que no se realiza la llamada a la función.

La manera que tiene javascript de mostrar un error puede variar de un navegador a otro. En versiones antiguas se mostraba una ventanita con el error y un botón de aceptar, tanto en Internet Explorer como en Netscape. En la actualidad los errores de javascript permanecen un poco más ocultos al usuario. Esto viene bien, porque si nuestras páginas tienen algún error en alguna plataforma no será muy molesto para el usuario que en muchas ocasiones ni se dará cuenta. Sin embargo para el programador puede ser un poco más molesto de revisar y se necesitará conocer la manera de que se muestren los errores para que puedan ser reparados.

En versiones de Internet Explorer mayores que la 4 se muestra el error en la barra de estado del navegador y se puede ver una descripción más grande del error si le damos un doble click al icono de alerta amarillo que aparece en la barra de estado. En Netscape aparece también un mensaje en la barra de estado que además nos indica que para mostrar más información debemos teclear "javascript:" en la barra de direcciones (donde escribimos las URL para acceder a las páginas). Con eso conseguimos que aparezca la Consola javascript, que nos muestra todos los errores que se encuentran en las páginas.

También podemos cometer fallos en la programación que hagan que los scripts no funcionen tal y como deseábamos y que javascript no detecta como errores y por lo tanto no muestra ningún mensaje de error.

Por dejarlo claro vamos a ver un ejemplo en el que nuestro programa puede no funcionar como deseamos sin que javascript ofrezca ningún mensaje de error. En este ejemplo trataríamos de sumar dos cifras pero si una de las variables es de tipo texto podríamos encontrarnos con un error.

```
var numero1 = 23
var numero2 = "42"
var suma = numero1 + numero2
```

¿Cuánto vale la variable suma? Como muchos ya sabéis, la variable suma vale "2342" porque al intentar sumar una variable numérica y otra textual, se tratan a las dos como si fueran de tipo texto y por lo tanto, el operador + se aplica como una concatenación de cadenas de caracteres. Si no estamos al tanto de esta cuestión podríamos tener un error en nuestro script ya que el resultado no es el esperado y además el tipo de la variable suma no es numérico sino cadena de caracteres.

Evitar errores comunes

Vamos a ver ahora una lista de los errores típicos cometidos por inexpertos en la programación en general y en javascript en particular, y por no tan inexpertos.

Utilizar = en expresiones condicionales en lugar de == es un error difícil de detectar cuando se comete, si utilizamos un solo igual lo que estamos haciendo es una asignación y no una comparación para ver si dos valores son iguales.

No conocerse la precedencia de operadores y no utilizar paréntesis para agrupar las operaciones que se desea realizar. En este caso nuestras operaciones pueden dar resultados no deseados.

Usar comillas dobles y simples erróneamente. Recuerda que se pueden utilizar comillas dobles o simples indistintamente, con la norma siguiente: dentro de comillas dobles se deben utilizar comillas simples y viceversa.

Olvidarse de cerrar una llave o cerrar una llave de más.

Colocar varias sentencias en la misma línea sin separarlas de punto y coma. Esto suele ocurrir en los manejadores de eventos, como onclick, que veremos más adelante.

Utilizar una variable antes de inicializarla o no declarar las variables con var antes de utilizarlas también son errores comunes. Recuerda que si no la declaras puedes estar haciendo referencia a una variable global en lugar de una local.

Contar las posiciones de los arrays a partir de 1. Recuerda que los arrays empiezan por la posición 0.

Depurar errores javascript

Cualquier programa es susceptible de contener errores. javascript nos informará de muchos de los errores de la página: los que tienen relación con la sintaxis y los que tienen lugar en el momento de la ejecución de los scripts a causa de equivocarnos al escribir el nombre de una función o una variable. Sin embargo, no son los únicos errores que nos podemos encontrar, también están los errores que ocurren sin que javascript muestre el correspondiente mensaje de error, como vimos anteriormente, pero que hacen que los programas no funcionen como esperábamos.

Para todo tipo de errores, unos más fáciles de detectar que otros, debemos utilizar alguna técnica de depuración que nos ayude a encontrarlos. Lenguajes de programación más potentes que el que tratamos

ahora incluyen importantes herramientas de depuración, pero en javascript debemos contentarnos con una rudimentaria técnica. Se trata de utilizar una función predefinida, la función alert() que recibe entre paréntesis un texto y lo muestra en una pequeña ventana que tiene un botón de aceptar.

Con la función alert() podemos mostrar en cualquier momento el contenido de las variables que estamos utilizando en nuestros scripts. Para ello ponemos entre paréntesis la variable que deseamos ver su contenido. Cuando se muestra el contenido de la variable el navegador espera a que apretemos el botón de aceptar y cuando lo hacemos continúa con las siguientes instrucciones del script.

Este es un sencillo ejemplo sobre cómo se puede utilizar la función alert() para mostrar el contenido de las variables.

```
var n_actual = 0
var suma = 0
while (suma<300){
   n_actual ++
   suma += suma + n_actual
   alert("n_actual vale " + n_actual + " y suma vale " + suma)
}
```

Con la función alert() se muestra el contenido de las dos variables que utilizamos en el script. Algo similar a esto es lo que tendremos que hacer para mostrar el contenido de las variables y saber cómo están funcionando nuestros scripts, si todo va bien o hay algún error.

Conclusión

Hasta aquí hemos conocido la sintaxis javascript en profundidad. Aunque aun nos quedan cosas importantes de sintaxis, la visión que has podido tener del lenguaje será suficiente para enfrentarte a los problemas más fundamentales. En adelante presentaremos otros reportajes para aprender a utilizar los recursos con los que contamos a la hora de hacer efectos en páginas web.

Veremos la jerarquía de objetos del navegador, cómo construir nuestros propios objetos, las funciones predefinidas de javascript, características del HTML Dinámico, trabajo con formularios y otras cosas importantes para dominar todas las posibilidades del lenguaje.

Todo ello en nuestro manual de Javascript II.

Capitulo 12
Programación en Javascript II

En este manual explicamos todos los recursos con los que cuenta un programador de Javascript para crear todo tipo de efectos y aplicaciones.

Introducción al manual II de Javascript

En esta segunda parte del manual de Javascript vamos a tratar de explicar todos los recursos con los que cuenta un programador de Javascript y con los que puede crear todo tipo de efectos y aplicaciones.

Para leer y entender bien lo que viene en los siguientes capítulos es necesario haber leído antes la primera parte de este manual: Programación en Javascript I, donde se explican las bases sobre las que tenemos que asentar los siguientes conocimientos. En la primera parte de este manual conocimos los orígenes y las aplicaciones de Javascript, pero sobretodo hicimos hincapié en su sintaxis, muy importante para entender los scripts que haremos en los siguientes capítulos.

Los objetivos de los siguientes capítulos cubrirán aspectos diversos de Javascript como:

- Funciones incorporadas en el lenguaje Javascript
- Los objetos en Javascript
- Jerarquía de objetos del navegador
- Trabajo con formularios
- Control de ventanas secundarias y frames
- Eventos

Como se puede ver, todos los temas tienen un fuerte carácter práctico y cubren aspectos varios con los que formarnos a nivel avanzado en Javascript. Esperamos que sirvan para iluminar un área tan amplia del desarrollo de páginas web como es el scripting del lado del cliente.

Vamos sin más pausa con esta segunda parte del manual, que resultará mucho más entretenida y práctica que la primera.

Libreria de funciones Javascript

En todos los lenguajes de programación existen librerías de funciones que sirven para hacer cosas diversas y muy repetitivas a la hora de programar. Las librerías de los lenguajes de programación ahorran la tarea de escribir las funciones comunes que por lo general pueden necesitar los programadores. Un lenguaje de programación bien desarrollado tendrá una buena cantidad de ellas. En ocasiones es más complicado conocer bien todas las librerías que aprender a programar en el lenguaje.

Javascript contiene una buena cantidad de funciones en sus librerías. Como se trata de un lenguaje que trabaja con objetos muchas de las librerías se implementan a través de objetos. Por ejemplo, las funciones matemáticas o las de manejo de strings se implementan mediante los objetos Math y String. Sin embargo, existen algunas funciones que no están asociadas a ningún objeto y son las que veremos en este capítulo, ya que todavía no conocemos los objetos y no los necesitaremos para estudiarlas.

Estas son las funciones que Javascript pone a disposición de los programadores.

eval(string)
Esta función recibe una cadena de caracteres y la ejecuta como si fuera una sentencia de Javascript.

parseInt(cadena,base)
Recibe una cadena y una base. Devuelve un valor numérico resultante de convertir la cadena en un número en la base indicada.

parseFloat(cadena)
Convierte la cadena en un número y lo devuelve.

escape(carácter)
Devuelve un el carácter que recibe por parámetro en una codificación ISO Latin 1.

unescape(carácter)
Hace exatamente lo opuesto a la función escape.

isNaN(número)
Devuelve un boleano dependiendo de lo que recibe por parámetro. Si no es un número devuelve un true, si es un numero devuelve false.

Las librerías que se implementan mediante objetos y las del manejo del explorador, que también se manejan con objetos, las veremos más adelante.

Vamos a ver algún ejemplo con las funciones más importantes de esta lista.

Ejemplos de funciones de la librería Javascript

Ahora podemos ver varios ejemplos de utilización de funciones de la librería que proporciona Javascript

Función eval

Esta función es muy importante, tanto que hay algunas aplicaciones de Javascript que no se podrían realizar si no la utilizamos. Su utilización es muy simple, pero puede que resulte un poco más complejo entender en qué casos utilizarla porque a veces resulta un poco sutil su aplicación.

Con los conocimientos actuales no podemos hacer un ejemplo muy complicado, pero por lo menos podemos ver en marcha la función. Vamos a utilizarla en una sentencia un poco rara y bastante inservible, pero si la conseguimos entender conseguiremos entender también la función eval.

```
var miTexto = "3 + 5"
eval("document.write(" + miTexto +")")
```

Primero creamos una variable con un texto, en la siguiente línea utilizamos la función eval y como parámetro le pasamos una instrucción javascript para escribir en pantalla. Si concatenamos los strings que hay dentro de los paréntesis de la función eval nos queda esto.

```
document.write(3 + 5)
```

La función eval ejecuta la instrucción que se le pasa por parámetro, así que ejecutará esta sentencia, lo que dará como resultado que se escriba un 8 en la página web. Primero se resuelve la suma que hay entre paréntesis, con lo que obtenemos el 8 y luego se ejecuta la instrucción de escribir en pantalla.

Función parseInt

Esta función recibe un número, escrito como una cadena de caracteres, y un número que indica una base. En realidad puede recibir otros tipos de variables, dado que las variables no tienen tipo en Javascript, pero se suele utilizar pasándole un string para convertir la variable de texto en un número.

Las distintas bases que puede recibir la función son 2, 8, 10 y 16. Si no le pasamos ningún valor como base la función interpreta que la base es decimal. El valor que devuelve la función siempre tiene base 10, de modo que si la base no es 10 convierte el número a esa base antes de devolverlo.

Veamos una serie de llamadas a la función parseInt para ver lo que devuelve y entender un poco más la función.

```
document.write (parseInt("34"))
Devuelve el numero 34

document.write (parseInt("101011",2))
Devuelve el numero 43

document.write (parseInt("34",8))
Devuelve el numero 28
```

```
document.write (parseInt("3F",16))
```
Devuelve el numero 63

Esta función se utiliza en la práctica para un montón de cosas distintas en el manejo con números, por ejemplo obtener la parte entera de un decimal.

```
document.write (parseInt("3.38"))
```
Devuelve el numero 3

También es muy habitual su uso para saber si una variable es numérica, pues si le pasamos un texto a la función que no sea numérico nos devolverá NaN (Not a Number) lo que quiere decir que No es un Número.

```
document.write (parseInt("desarrolloweb.com"))
```
Devuelve el numero NaN

Este mismo ejemplo es interesante con una modificación, pues si le pasamos una combinación de letras y números nos dará lo siguiente.

```
document.write (parseInt("16XX3U"))
```
Devuelve el numero 16

```
document.write (parseInt("TG45"))
```
Devuelve el numero NaN

Como se puede ver, la función intenta convertir el string en número y si no puede devuelve NaN.

Todos estos ejemplos, un tanto inconexos, sobre cómo trabaja parseInt los revisaremos más adelante en ejemplos más prácticos cuando tratemos el trabajo con formularios.

Función isNaN

Esta función devuelve un boleano dependiendo de si lo que recibe es un número o no. Lo único que puede recibir es un número o la expresión NaN. Si recibe un NaN devuelve true y si recibe un número devuelve false. Es una función muy sencilla de entender y de utilizar.

La función suele trabajar en combinación con la función parseInt o parseFloat, para saber si lo que devuelven estas dos funciones es un número o no.

```
miInteger = parseInt("A3.6")
isNaN(miInteger)
```

En la primera línea asignamos a la variable miInteger el resultado de intentar convertir a entero el texto A3.6. Como este texto no se puede convertir a número la función parseInt devuelve NaN. La segunda línea comprueba si la variable anterior es NaN y como si que lo es devuelve un true.

```
miFloat = parseFloat("4.7")
isNaN(miFloat)
```

En este ejemplo convertimos un texto a número con decimales. El texto se convierte perfectamente porque corresponde con un número. Al recibir un número la función isNaN devuelve un false.

Objetos en Javascript

Vamos a introducirnos en un tema muy importante de Javascript como son los objetos. Es un tema que aun no hemos visto y sobre el que en adelante vamos a tratar constantemente pues todas las cosas en Javascript, incluso las más sencillas, las vamos a realizar a través del manejo de objetos. De hecho, en los ejemplos realizados hasta ahora hemos hecho grandes esfuerzos para no utilizar objetos y aun así los hemos utilizado en alguna ocasión, pues es muy difícil encontrar ejemplos en Javascript que, aunque sean simples, no hagan uso de ellos.

La programación orientada a objetos representa una nueva manera de pensar a la hora de hacer un programa. Javascript no es un lenguaje de programación orientado a objetos, aunque los utiliza en muchas ocasiones: podemos crear nuevos objetos y utilizar muchos que están creados desde un

principio. Sin embargo la manera de programar no va a cambiar mucho y lo que hemos visto hasta aquí relativo a sintaxis, funciones, etc. puede ser utilizado igual que se ha indicado. Solo vamos a aprender una especie de estructura nueva.

Para empezar a empaparnos de programación orientada a objetos es imprescindible que nos leamos un pequeño artículo publicado en DesarrolloWeb sobre este tema. Después de su lectura puedes continuar con estas líneas y si conoces ya la programación orientada a objetos continúa leyendo sin pausa.

Cómo instanciar objetos

Instanciar un objeto es la acción de crear un ejemplar de una clase para poder trabajar con él luego. Recordamos que un objeto se crea a partir de una clase y la clase es la definición de las características y funcionalidades de un objeto. Con las clases no se trabaja, estas sólo son definiciones, para trabajar con una clase debemos tener un objeto instanciado de esa clase.

En javascript para crear un objeto a partir de una clase se utiliza la instrucción new, de esta manera.

```
var miObjeto = new miClase()
```

En una variable que llamamos miObjeto asigno un nuevo (new) ejemplar de la clase miClase. Los paréntesis se rellenan con los datos que necesite la clase para inicializar el objeto, si no hay que meter ningún parámetro los paréntesis se colocan vacíos. En realidad lo que se hace cuando se crea un objeto es llamar al constructor de esa clase y el constructor es el encargado de crearlo e inicializarlo. Hablaremos sobre esto más adelante.

Cómo acceder a propiedades y métodos de los objetos

En Javascript podemos acceder a las propiedades y métodos de objetos de forma similar a como se hace en otros lenguajes de programación, con el operador punto (".").

Las propiedades se acceden colocando el nombre del objeto seguido de un punto y el nombre de la propiedad que se desea acceder. De esta manera:

```
miObjeto.miPropiedad
```

Para llamar a los métodos utilizamos una sintaxis similar pero poniendo al final entre paréntesis los parámetros que pasamos a los métodos. Del siguiente modo:

```
miObjeto.miMetodo(parametro1,parametro2)
```

Si el método no recibe parámetros colocamos los paréntesis también, pero sin nada dentro.

```
miObjeto.miMetodo()
```

Objetos incorporados en Javascript

Sabiendo ya lo que es la programación orientada a objetos vamos a introducirnos en el manejo de objetos en Javascript y para ello vamos a empezar con el estudio de las clases predefinidas que implementan las librerías para el trabajo con stings, matemáticas, fechas etc. Las clases que vamos a ver a continuación son las siguientes:

- **String**, para el trabajo con cadenas de caracteres.
- **Date**, para el trabajo con fechas.
- **Math**, para realizar funciones matemáticas.
- **Number**, para realizar algunas cosas con números
- **Boolean**, trabajo con boleanos.

Nota: Las clases se escriben con la primera letra en mayúsculas. Tiene que quedar claro que una clase es una especie de "declaración de características y funcionalidades" de los objetos. Dicho de otro modo, las clases son descripciones de la forma y funcionamiento de los objetos. Con las clases generalmente no se realiza ningún trabajo, sino que se hace con los objetos, que creamos a partir de las clases.

Una vez comprendida la diferencia entre objetos y clases, cabe señalar que String es una clase, lo mismo que Date. Si queremos trabajar con cadenas de caracteres o fechas necesitamos crear objetos de las clases String y Date respectivamente. Como sabemos, Javascript es un lenguaje sensible a las mayúsculas y las minúsculas y en este caso, **las clases, por convención, siempre se escriben con la primera letra en mayúscula y**

también la primera letra de las siguientes palabras, si es que el nombre de la clase está compuesto de varias palabras. Por ejemplo si tuvieramos la clase de "Alumnos universitarios" se escribiría con la -A- de alumnos y la -U- de universitarios en mayúscula. AlumnosUniversitarios sería el nombre de nuestra supuesta clase.

Hay otros que pertenecen a este mismo conjunto, los enumeramos aquí para que quede constancia de ellos, pero no los vamos a tocar en capítulos siguientes.

- **Array**, ya los hemos visto en capítulos correspondientes al primer manual de Javascript.
- También la clase **Function**, lo hemos visto parcialmente al estudiar las funciones.
- Hay otra clase **RegExp** que sirve para construir patrones que veremos tal vez junto con Function cuando tratemos temas más avanzados todavía.

Nota: Uso de mayúsculas y minúsculas. Ya que nos hemos parado anteriormente a hablar sobre el uso de mayúsculas en ciertas letras de los nombre de las clases, vamos a terminar de explicar la convención que se lleva a cabo en Javascript para nombrar a los elementos.

Para empezar, cualquier variable se suele escribir en minúsculas, aunque si este nombre de variable se compone de varias palabras, se suele escribir en mayúscula la primera letra de las siguientes palabras a la primera. Esto se hace así en cualquier tipo de variable o nombre menos en los nombres de las clases, donde se escribe también en mayúscula el primer caracter de la primera palabra.

Ejemplos:

Number, que es una clase se escribe con la primera en mayúscula.
RegExp, que es el nombre de otra clase compuesto por dos palabras, tiene la primera letras de las dos palabras en mayúscula.
miCadena, que supongamos que es un objeto de la clase String, se escribe con la primera letra en minúscula y la primera letra de la segunda palabra en mayúscula.
fecha, que supongamos que es un objeto de la clase Date, se escribe en minúsculas por ser un objeto.
miFunción(), que es una función definida por nosotros, se acostumbra a escribir con minúscula la primera.

Como decimos, esta es la norma general para dar nombres a las variables, clases, objetos, funciones, etc. en Javascript. Si la seguimos así, no tendremos problemas a la hora de saber si un nombre en Javascript tiene o no alguna mayúscula y además todo nuestro trabajo será más claro y fácil de seguir por otros programadores o nosotros mismos en caso de retomar un código una vez pasado un tiempo.

Clase String en Javascript

En javascript las variables de tipo texto son objetos de la clase String. Esto quiere decir que cada una de las variables que creamos de tipo texto tienen una serie de propiedades y métodos. Recordamos que las propiedades son las características, como por ejemplo longitud en caracteres del string y los métodos son funcionalidades, como pueden ser extraer un substring o poner el texto en mayúsculas.

Para crear un objeto de la clase String lo único que hay que hacer es asignar un texto a una variable. El texto va entre comillas, como ya hemos visto en los capítulos de sintaxis. También se puede crear un objeto string con el operador new, que veremos más adelante. La única diferencia es que en versiones de Javascript 1.0 no funcionará new para crear los Strings.

Propiedades de String

Length
La clase String sólo tiene una propiedad: length, que guarda el número de caracteres del String.

Métodos de String

Los objetos de la clase String tienen una buena cantidad de métodos para realizar muchas cosas interesantes. Primero vamos a ver una lista de los métodos más interesantes y luego vamos a ver otra lista de métodos menos útiles.

charAt(indice)
Devuelve el carácter que hay en la posición indicada como índice. Las posiciones de un string empiezan en 0.

indexOf(carácter,desde)
Devuelve la posición de la primera vez que aparece el carácter indicado por parámetro en un string. Si no encuentra el carácter en el string devuelve -1. El segundo parámetro es opcional y sirve para indicar a partir de que posición se desea que empiece la búsqueda.

lastIndexOf(carácter,desde)
Busca la posición de un carácter exáctamente igual a como lo hace la función indexOf pero desde el final en lugar del principio. El segundo parámetro indica el número de caracteres desde donde se busca, igual que en indexOf.

replace(substring_a_buscar,nuevoStr)
Implementado en Javascript 1.2, sirve para reemplazar porciones del texto de un string por otro texto, por ejemplo, podríamos uilizarlo para reemplazar todas las apariciones del substring "xxx" por "yyy". El método no reemplaza en el string, sino que devuelve un resultante de hacer ese reemplazo. Acepta expresiones regulares como substring a buscar.

split(separador)
Este método sólo es compatible con javascript 1.1 en adelante. Sirve para crear un vector a partir de un String en el que cada elemento es la parte del String que está separada por el separador indicado por parámetro.

substring(inicio,fin)
Devuelve el substring que empieza en el carácter de inicio y termina en el carácter de fin. Si intercambiamos los parámetros de inicio y fin también funciona. Simplemente nos da el substring que hay entre el carácter menor y el mayor.

toLowerCase()
Pone todas los caracteres de un string en minúsculas.

toUpperCase()
Pone todas los caracteres de un string en mayúsculas.

toString()
Este método lo tienen todos los objetos y se usa para convertirlos en cadenas.

Hasta aquí hemos visto los métodos que nos ayudarán a tratar cadenas. Ahora vamos a ver otros métodos que son menos útiles, pero hay que indicarlos para que quede constancia de ellos. Todos sirven para aplicar estilos a un texto y es como si utilizásemos etiquetas HTML. Veamos cómo.

anchor(name)
Convierte en un ancla (sitio a donde dirigir un enlace) una cadena de caracteres usando como el atributo name de la etiqueta <A> lo que recibe por parámetro.

big()
Aumenta el tamaño de letra del string. Es como si colocásemos en un string al principio la etiqueta <BIG> y al final </BIG>.

blink()
Para que parpadee el texto del string, es como utilizar la etiqueta <BLINK>. Solo vale para Netscape.

bold()
Como si utilizásemos la etiqueta <B>.

fixed()
Para utilizar una fuente monoespaciada, etiqueta <TT>.

fontColor(color)
Pone la fuente a ese color. Como utilizar la etiqueta <FONT color=el_color_indicado>.

fontSize(tamaño)
Pone la fuente al tamaño indicado. Como si utilizásemos la etiqueta <FONT> con el atributo size.

italics()
Pone la fuente en cursiva. Etiqueta <I>.

link(url)
Pone el texto como un enlace a la URL indicada. Es como si utilizásemos la etiqueta <A> con el atributo href indicado como parámetro.

small()
Es como utilizar la etiqueta <SMALL>

strike()
Como utilizar la etiqueta <STRIKE>, que sirve para que el texto aparezca tachado.

sub()
Actualiza el texto como si se estuviera utilizando la etiqueta <SUB>, de subíndice.

sup()
Como si utilizásemos la etiqueta <SUP>, de superíndice.

Informe de **Miguel Angel Alvarez**
Director DesarrolloWeb.com
Mail: **miguel@desarrolloweb.com**

Ejemplos de funcionamiento de la clase String

Ahora vamos a ver unos ejemplos sobre cómo se utilizan los métodos y propiedades del objeto String.

Ejemplo de strings 1

Vamos a escribir el contenido de un string con un carácter separador ("-") entre cada uno de los caracteres del string.

```
var miString = "Hola Amigos"
var result = ""

for (i=0;i<miString.length-1;i++) {
   result += miString.charAt(i)
   result += "-"
}
result += miString.charAt(miString.length - 1)

document.write(result)
```

Primero creamos dos variables, una con el string a recorrer y otra con un string vacío, donde guardaremos el resultado. Luego hacemos un bucle que recorre desde el primer hasta el penúltimo carácter del string -utilzamos la propiedad length para conocer el número de caracteres del string- y en cada iteración colocamos un carácter del string seguido de un carácter separador "-". Como aun nos queda el último carácter por colocar lo ponemos en la siguiente línea después del bucle. Utilizamos la función charAt para acceder a las posiciones del string. Finalmente imprimimos en la página el resultado.

Podemos ver el ejemplo en funcionamiento en una página a parte.

Ejemplo de strings 2

Vamos a hacer un script que rompa un string en dos mitades y las imprima por pantalla. Las mitades serán iguales, siempre que el string tenga un número de caracteres par. En caso de que el número de caracteres sea impar no se podrá hacer la mitad exacta, pero partiremos el string lo más aproximado a la mitad.

```
var miString = "0123456789"
var mitad1,mitad2

posicion_mitad = miString.length / 2

mitad1 = miString.substring(0,posicion_mitad)
mitad2 = miString.substring(posicion_mitad,miString.length)

document.write(mitad1 + "<br>" + mitad2)
```

Las dos primeras líneas sirven para declarar las variables que vamos a utilizar e inicializar el string a partir. En la siguiente línea hallamos la posición de la mitad del string.

En las dos siguientes líneas es donde realizamos el trabajo de poner en una variable la primera mitad del string y en la otra la segunda. Para ello utilizamos el método substring pasándole como inicio y fin en el primer caso desde 0 hasta la mitad y en el segundo desde la mitad hasta el final. Para finalizar imprimimos las dos mitades con un salto de línea entre ellas.

Clase Date en Javascript

Sobre este objeto recae todo el trabajo con fechas en Javascript, como obtener una fecha, el día la hora y otras cosas. Para trabajar con fechas necesitamos instanciar un objeto de la clase Date y con él ya podemos realizar las operaciones que necesitemos.

Un objeto de la clase Date se puede crear de dos maneras distintas. Por un lado podemos crear el objeto con el día y hora actuales y por otro podemos crearlo con un día y hora distintos a los actuales. Esto depende de los parámetros que pasemos al construir los objetos.

Para crear un objeto fecha con el día y hora actuales colocamos los paréntesis vacíos al llamar al constructor de la clase Date.

```
miFecha = new Date()
```

Para crear un objeto fecha con un día y hora distintos de los actuales tenemos que indicar entre paréntesis el momento con que inicializar el objeto. Hay varias maneras de expresar un día y hora válidas, por eso podemos construir una fecha guiándonos por varios esquemas. Estos son dos de ellos, suficientes para crear todo tipo de fechas y horas.

```
miFecha = new Date(año,mes,dia,hora,minutos,segundos)
miFecha = new Date(año,mes,dia)
```

Los valores que debe recibir el constructor son siempre numéricos. Un detalle, el mes comienza por 0, es decir, enero es el mes 0. Si no indicamos la hora, el objeto fecha se crea con hora 00:00:00.

Los objetos de la clase Date no tienen propiedades pero si un montón de métodos, vamos a verlos ahora.

getDate()
Devuelve el día del mes.

getDay()
Devuelve el día de la semana.

getHours()
Retorna la hora.

getMinutes()
Devuelve los minutos.

getMonth()
Devuelve el mes (atención al mes que empieza por 0).

getSeconds()
Devuelve los segundos.

getTime()
Devuelve los milisegundos transcurridos entre el día 1 de enero de 1970 y la fecha correspondiente al objeto al que se le pasa el mensaje.

getYear()
Retorna el año, al que se le ha restado 1900. Por ejemplo, para el 1995 retorna 95, para el 2005 retorna 105. Este método está obsoleto en Netscape a partir de la versión 1.3 de Javascript y ahora se utiliza getFullYear().

getFullYear()
Retorna el año con todos los dígitos. Usar este método para estar seguros de que funcionará todo bien en fechas posteriores al año 2000.

setDate()
Actualiza el día del mes.

setHours()
Actualiza la hora.

setMinutes()
Cambia los minutos.

setMonth()
Cambia el mes (atención al mes que empieza por 0).

setSeconds()
Cambia los segundos.

setTime()
Actualiza la fecha completa. Recibe un número de milisegundos desde el 1 de enero de 1970.

setYear()
Cambia el año recibe un número, al que le suma 1900 antes de colocarlo como año de la fecha. Por ejemplo, si recibe 95 colocará el año 1995. Este método está obsoleto a partir de Javascript 1.3 en Netscape. Ahora se utiliza setFullYear(), indicando el año con todos los dígitos.

setFullYear()
Cambia el año de la fecha al número que recibe por parámetro. El número se indica completo ej: 2005 o 1995. Utilizar este método para estar seguros que todo funciona para fechas posteriores a 2000.

Como habréis podido apreciar, hay algún método obsoleto por cuestiones relativas al año 2000: setYear() y getYear(). Estos métodos se comportan bien en Internet Explorer y no nos dará ningún problema el utilizarlos. Sin embrago, no funcionarán correctamente en Netscape, por lo que es interesante que utilicemos en su lugar los métodos getFullYear() y setFullYear(), que funcionan bien en Netscape e Internet Explorer.

Ejemplo de funcionamiento de Date

En este ejemplo vamos a crear dos fechas, una con el instante actual y otra con fecha del pasado. Luego las imprimiremos las dos y extraeremos su año para imprimirlo también. Luego actualizaremos el año de una de las fechas y la volveremos a escribir con un formato más legible.

```
//en estas líneas creamos las fechas
miFechaActual = new Date()
miFechaPasada = new Date(1998,4,23)

//en estas líneas imprimimos las fechas.
document.write (miFechaActual)
document.write ("<br>")
document.write (miFechaPasada)

//extraemos el año de las dos fechas
anoActual = miFechaActual.getFullYear()
anoPasado = miFechaPasada.getFullYear()

//Escribimos en año en la página
document.write("<br>El año actual es: " + anoActual)
document.write("<br>El año pasado es: " + anoPasado)

//cambiamos el año en la fecha actual
miFechaActual.setFullYear(2005)

//extraemos el día mes y año
dia = miFechaActual.getDate()
mes = parseInt(miFechaActual.getMonth()) + 1
ano = miFechaActual.getFullYear()

//escribimos la fecha en un formato legible
document.write ("<br>")
document.write (dia + "/" + mes + "/" + ano)
```

Hay que destacar un detalle antes de acabar y es que el número del mes puede empezar desde 0. Por lo menos en el Netscape con el que realizamos las pruebas empezaba en 0 el mes. Por esta razón sumamos uno al mes que devuelve el método getMonth.

Hay más detalles a destacar, pues resulta que en Netscape el método getFullYear() devuelve los años trascurridos desde 1900, con lo que al obtener el año de una fecha de, por ejemplo, 2005, indica que es el año 105. Para obtener el año completo tenemos a nuestra disposición el método getFullYear() que devolvería 2005 del mismo modo en Netscape e Internet Explorer.

Mucha atención, pues, al trabajo con fechas en distintas plataformas, puesto que podría ser problemático el hecho de que nos ofrezcan distintas salidas los métodos de manejo de fechas, com siempre dependiendo de la marca y versión de nuestro navegador.

Clase Math en Javascript

La clase Math proporciona los mecanismos para realizar operaciones matemáticas en Javascript. Algunas operaciones se resuelven rápidamente con los operadores aritméticos que ya conocemos, como la multiplicación o la suma, pero hay una serie de operaciones matemáticas adicionales que se tienen que realizar usando la clase Math como pueden ser calcular un seno o hacer una raiz cuadrada.

De modo que para cualquier cálculo matemático complejo utilizaremos la clase Math, con una particularidad. Hasta ahora cada vez que queríamos hacer algo con una clase debíamos instanciar un objeto de esa clase y trabajar con el objeto y en el caso de la clase Math se trabaja directamente con la clase. Esto se permite por que las propiedades y métodos de la clase Math son lo que se llama propiedades y métodos de clase y para utilizarlos se opera a través de la clase en lugar de los objetos. Dicho de otra forma, para trabajar con la clase Math no deberemos utilizar la instrucción new y utilizaremos el nombre de la clase para acceder a sus propiedades y métodos.

Propiedades de Math

Las propiedades guardan valores que probablemente necesitemos en algún momento si estamos haciendo cálculos matemáticos. Es probable que estas propiedades resulten un poco raras a las personas que desconocen las matemáticas avanzadas, pero los que las conozcan sabrán de su utilidad.

E
Número E o constante de Euler, la base de los logaritmos neperianos.

LN2
Logaritmo neperiano de 2.

LN10
Logaritmo neperiano de 10.

LOG2E
Logaritmo en base 2 de E.

LOG10E
Logaritmo en base 10 de E.

PI
Conocido número para cálculo con círculos.

SQRT1_2
Raiz cuadrada de un medio.

SQRT2
Raiz cuadrada de 2.

Métodos de Math

Así mismo, tenemos una serie de métodos para realizar operaciones matemáticas típicas, aunque un poco complejas. Todos los que conozcan las matemáticas a un buen nivel conocerán el significado de estas operaciones.

abs()
Devuelve el valor absoluto de un número. El valor después de quitarle el signo.

acos()
Devuelve el arcocoseno de un número en radianes.

asin()
Devuelve el arcoseno de un numero en radianes.

atan()
Devuelve un arcotangente de un numero.

ceil()
Devuelve el entero igual o inmediatamente siguiente de un número. Por ejemplo, ceil(3) vale 3, ceil(3.4) es 4.

cos()
Retorna el coseno de un número.

exp()

Retorna el resultado de elevar el número E por un número.

floor()
Lo contrario de ceil(), pues devuelve un número igual o inmediatamente inferior.

log()
Devuelve el logaritmo neperiano de un número.

max()
Retorna el mayor de 2 números.

min()
Retorna el menor de 2 números.

pow()
Recibe dos números como parámetros y devuelve el primer número elevado al segundo número.

random()
Devuelve un número aleatorio entre 0 y 1. Método creado a partir de Javascript 1.1.

round()
Redondea al entero más próximo.

sin()
Devuelve el seno de un número con un ángulo en radianes.

sqrt()
Retorna la raiz cuadrada de un número.

tan()
Calcula y devuelve la tangente de un número en radianes.

Ejemplo de utilización de la clase Math

Vamos a ver un sencillo ejemplo sobre cómo utilizar métodos y propiedaes de la clase Math para calcular el seno y el coseno de 2 PI radianes (una vuelta completa). Como algunos de vosotros sabréis, el coseno de 2 PI radianes debe dar como resultado 1 y el seno 0.

```
document.write (Math.cos(2 * Math.PI))

document.write ("<br>")

document.write (Math.sin(2 * Math.PI))
```

2 PI radianes es el resultado de multiplicar 2 por el número PI. Ese resultado es lo que recibe el método cos, y da como resultado 1. En el caso del seno el resultado no es exactamente 0 pero está aproximado con una exactitud de más de una millonésima de fracción. Se escriben los el seno y coseno con un salto de línea en medio para que quede más claro.

Clase Number en Javascript

La clase Number modeliza el tipo de datos numérico. Fue añadido en la versión 1.1 de Javascript (con Netscape Navigator 3). Nos sirve para crear objetos que tienen datos numéricos como valor. Es muy probable que no lo llegues a utilizar en ninguna ocasión. Por lo menos en todos los scripts que sirven para hacer las cosas más dispares y útiles.

> Nota: conocimos el tipo de datos numérico en el primer manual de javascript. Este nos servía para guardar un valores numéricos sin más. Este objeto modeliza este tipo de datos y la clase en si, ofrece algún método que puede ser útil. Para los cálculos matemáticos y el uso de números en general vamos a utilizar siempre las variables numéricas vistas anteriormente.

El valor del objeto Number que se crea depende de lo que reciba el constructor de la clase Number. Con estas reglas:

- Si el constructor recibe un número, entonces inicializa el objeto con el número que recibe. Si recibe un número entrecomillado lo convierte a valor numérico, devolviendo también dicho número.

- Devuelve 0 en caso de que no reciba nada.

- En caso de que reciba un valor no numérico devuelve NaN, que significa "Not a Number" (No es un número)

- Si recibe false se inicializa a 0 y si recibe true se inicializa a 1.

Su funcionamiento se puede resumir en estos ejemplos.

```
var n1 = new Number()
document.write(n1 + "<br>")
//muestra un 0

var n2 = new Number("hola")
document.write(n2 + "<br>")
//muestra NaN

var n3 = new Number("123")
document.write(n3 + "<br>")
//muestra 123

var n4 = new Number("123asdfQWERTY")
document.write(n4 + "<br>")
//muestra NaN

var n5 = new Number(123456)
document.write(n5 + "<br>")
//muestra 123456

var n6 = new Number(false)
document.write(n6 + "<br>")
//muestra 0

var n7 = new Number(true)
document.write(n7 + "<br>")
//muestra 1
```

Este ejemplo y el siguiente, se pueden ver en una página a parte.

Propiedades de la clase Number

Esta clase también nos ofrece varias propiedades que contienen los siguientes valores:

NaN
Como hemos visto, significa Not a Number, o en español, no es un número.

MAX_VALUE y MIN_VALUE
Guardan el valor del máximo y el mínimo valor que se puede representar en Javascript

NEGATIVE_INFINITY y POSITIVE_INFINITY
Representan los valores, negativos y positivos respectivamente, a partir de los cuales hay desbordamiento.

Estas propiedades son de clase, así que accederemos a ellas a partir del nombre de la clase, tal como podemos ver en este ejemplo en el que se muestra cada uno de sus valores.

```
document.write("Propiedad NaN: " + Number.NaN)
document.write("<br>")
document.write("Propiedad MAX_VALUE: " + Number.MAX_VALUE)
document.write("<br>")
document.write("Propiedad MIN_VALUE: " + Number.MIN_VALUE)
document.write("<br>")
document.write("Propiedad NEGATIVE_INFINITY: " + Number.NEGATIVE_INFINITY)
document.write("<br>")
document.write("Propiedad POSITIVE_INFINITY: " + Number.POSITIVE_INFINITY)
```

Clase Boolean en Javascript

Esta clase nos sirve para crear valores boleanos. Fue añadido en la versión 1.1 de Javascript (con Netscape Navigator 3). Una de sus posibles utilidades es la de conseguir valores boleanos a partir de datos de cualquier otro tipo.

Nota: conocimos el tipo de datos boolean en el primer manual de Javascript. Este nos servía para guardar un valor verdadero (true) o falso (false). Esta clase modeliza ese tipo de datos para crear objetos boleanos.

Dependiendo de lo que reciba el constructor de la clase Bolean el valor del objeto boleano que se crea será verdadero o falso, de la siguiente manera

- **Se inicializa a false** cuando no pasas ningún valor al constructor, o si pasas una cadena vacía, el número 0 o la palabra false sin comillas.

- **Se inicializa a true** cuando recibe cualquier valor entrecomillado o cualquier número distinto de 0.

Se puede comprender el funcionamiento de este objeto fácilmente si examinamos unos ejemplos.

```
var b1 = new Boolean()
document.write(b1 + "<br>")
//muestra false

var b2 = new Boolean("")
document.write(b2 + "<br>")
//muestra false

var b25 = new Boolean(false)
document.write(b25 + "<br>")
//muestra false

var b3 = new Boolean(0)
document.write(b3 + "<br>")
//muestra false

var b35 = new Boolean("0")
document.write(b35 + "<br>")
//muestra true

var b4 = new Boolean(3)
document.write(b4 + "<br>")
//muestra true

var b5 = new Boolean("Hola")
document.write(b5 + "<br>")
//muestra true
```

Creación de clases en Javascript

Ahora que ya hemos conocido un poco los objetos y hemos aprendido a manejarlos podemos pasar a un tema más avanzado, como es el de construir nuestros propios objetos, que puede sernos útil en ciertas ocasiones para temas avanzados.

Así que vamos a ver cómo podemos definir nuestros propios objetos, con sus propiedades y métodos, de manera que aprendamos el mecanismo, pero sin entrar demasiado en aspectos prácticos que los dejamos para ejemplos del futuro.

Para crear nuestros propios objetos debemos crear una clase, que recordamos que es algo así como la definición de un objeto con sus propiedades y métodos. Para crear la clase en Javascript debemos escribir una función especial, que se encargará de construir el objeto en memoria e inicializarlo. Esta función se le llama constructor en la terminología de la programación orientada a objetos.

```
function MiClase (valor_inicializacion){
   //Inicializo las propiedades y métodos
   this.miPropiedad = valor_inicializacion
   this.miMetodo = nombre_de_una_funcion_definida
}
```

Eso era un constructor. Utiliza la palabra this para declarar las propiedades y métodos del objeto que se está construyendo. This hace referencia al objeto que se está construyendo, pues recordemos que a esta función la llamaremos para construir un objeto. A ese objeto que se está construyendo le vamos asignando valores en sus propiedades y también le vamos asignando nombres de funciones definidas para sus métodos. Al construir un objeto técnicamente es lo mismo declarar una propiedad o un método, solo difiere en que a una propiedad le asignamos un valor y a un método le asignamos una función.

La clase AlumnoUniversitario

Lo veremos todo más detenidamente si hacemos un ejemplo. En este ejemplo vamos a crear un objeto estudiante universitario. Como estudiante tendrá unas características como el nombre, la edad o el número de matrícula. Además podrá tener algún método como por ejemplo matricular al alumno.

Constructor: Colocamos propiedades

Veamos cómo definir el constructor de la clase Alumnouniversitario, pero solamente vamos a colocar por ahora las propiedades de la clase.

```
function AlumnoUniversitario(nombre, edad){
   this.nombre = nombre
   this.edad = edad
   this.numMatricula = null
}
```

Los valores de inicialización los recibe el constructor como parámetros, en este caso es sólo el nombre y la edad, porque el número de matrícula no lo recibe un alumno hasta que es matriculado. Es por ello que asignamos a null la propiedad numMatrícula.

Creación de clases en Javascript II

Para construir un método debemos crear una función. Una función que se construye con intención de que sea un método para una clase puede utilizar también la variable this, que hace referencia al objeto sobre el que invocamos el método. Pues debemos recordar que para llamar a un método debemos tener un objeto y this hace referencia a ese objeto.

```
function matriculate(num_matricula){
   this.numMatricula = num_matricula
}
```

La función matricular recibe un número de matrícula por parámetro y lo asigna a la propiedad numMatricula del objeto que recibe este método. Así rellenamos el la propiedad del objeto que nos faltaba.

Vamos a construir otro método que imprime los datos del alumno.

```
function imprimete(){
   document.write("Nombre: " + this.nombre)
   document.write("<br>Edad: " + this.edad)
   document.write("<br>Número de matrícula: " + this.numMatricula)
}
```

Esta función va imprimiendo todas las propiedades del objeto que recibe el método.

Constructor: Colocamos métodos

Para colocar un método en una clase debemos asignar la función que queremos que sea el método al objeto que se está creando. Veamos cómo quedaría el constructor de la clase AlumnoUniversitario con el método matricular.

```
function AlumnoUniversitario(nombre, edad){
   this.nombre = nombre
   this.edad = edad
   this.numMatricula = null
   this.matriculate = matriculate
   this.imprimete = imprimete
}
```

Vemos que en las últimas líneas asignamos a los métodos los nombres de las funciones que contienen su código.

Para instanciar un objeto

Para instanciar objetos de la clase AlumnoUniversitario utilizamos la sentencia new, que ya hemos tenido ocasión de ver en otras ocasiones.

```
miAlumno = new AlumnoUniversitario("José Díaz",23)
```

Creación de clases en Javascript III

Para ilustrar el trabajo con objetos y terminar con el ejemplo del AlunnoUniversitario, vamos a ver todo este proceso en un solo script en el que definiremos la clase y luego la utilizaremos un poquito.

```
//definimos el método matricularte para la clase AlumnoUniversitario
function matriculate(num_matricula){
   this.numMatricula = num_matricula
}

//definimos el método imprimete para la clase AlumnoUniversitario
function imprimete(){
   document.write("<br>Nombre: " + this.nombre)
   document.write("<br>Edad: " + this.edad)
   document.write("<br>Número de matrícula: " + this.numMatricula)
}

//definimos el constructor para la clase
function AlumnoUniversitario(nombre, edad){
   this.nombre = nombre
   this.edad = edad
   this.numMatricula = null
   this.matriculate = matriculate
   this.imprimete = imprimete
}

//creamos un alumno
miAlumno = new AlumnoUniversitario("José Díaz",23)

//le pedimos que se imprima
miAlumno.imprimete()

//le pedimos que se matricule
miAlumno.matriculate(305)

//le pedimos que se imprima de nuevo (con el número de matricula relleno)
miAlumno.imprimete()
```

Jerarquía de objetos del navegador

Llegamos al tema más importante para aprender a manejar Javascript con toda su potencia, el tema en el que aprenderemos a controlar al navegador y los distintos elementos de la página.

Sin duda, este tema le va a dar mucha vida a nuestros ejemplos, ya que hasta ahora no tenían mucho carácter práctico porque no trabajaban con el navegador y las páginas, que es realmente para lo que está hecho Javascript. De modo que esperamos que a partir de aquí el manual sea más entretenido para todos, porque va a cubrir los aspectos más prácticos.

Cuando se carga una página, el navegador crea una jerarquía de objetos en memoria que sirven para controlar los distintos elementos de dicha página. Con Javascript y la nomenclatura de objetos que hemos aprendido, podemos trabajar con esa jerarquía de objetos, acceder a sus propiedades e invocar sus métodos.

Cualquier elemento de la página se puede controlar de una manera u otra accediendo a esa jerarquía. Es crucial conocerla bien para poder controlar perfectamente las páginas web con Javascript o cualquier otro lenguaje de programación del lado del cliente.

Ejemplo de acceso a la jerarquía

Antes de empezar a ver rigurosamente la jerarquía de objetos del navegador, vamos a ver el ejemplo típico de acceso a una propiedad de esta jerarquía para cambiar el aspecto de la página. Se trata de una propiedad de la página que almacena el color de fondo de la página web: la propiedad bgColor del objeto document.

```
document.bgColor = "red"
```

Si ejecutamos esta sentencia en cualquier momento cambiamos el color de fondo de la página a rojo. Hay que fijarse en que la propiedad bgColor tiene la "C" en mayúscula. Es un error típico equivocarse con las mayúsculas y minúsculas en la jerarquía. Si no lo escribimos bien no funcionará y en algunos casos ni siquiera dará un mensaje de error.

En esta página definida con color de fondo blanco hemos cambiado esa propiedad luego con Javascript, por lo que saldrá con color de fondo rojo.

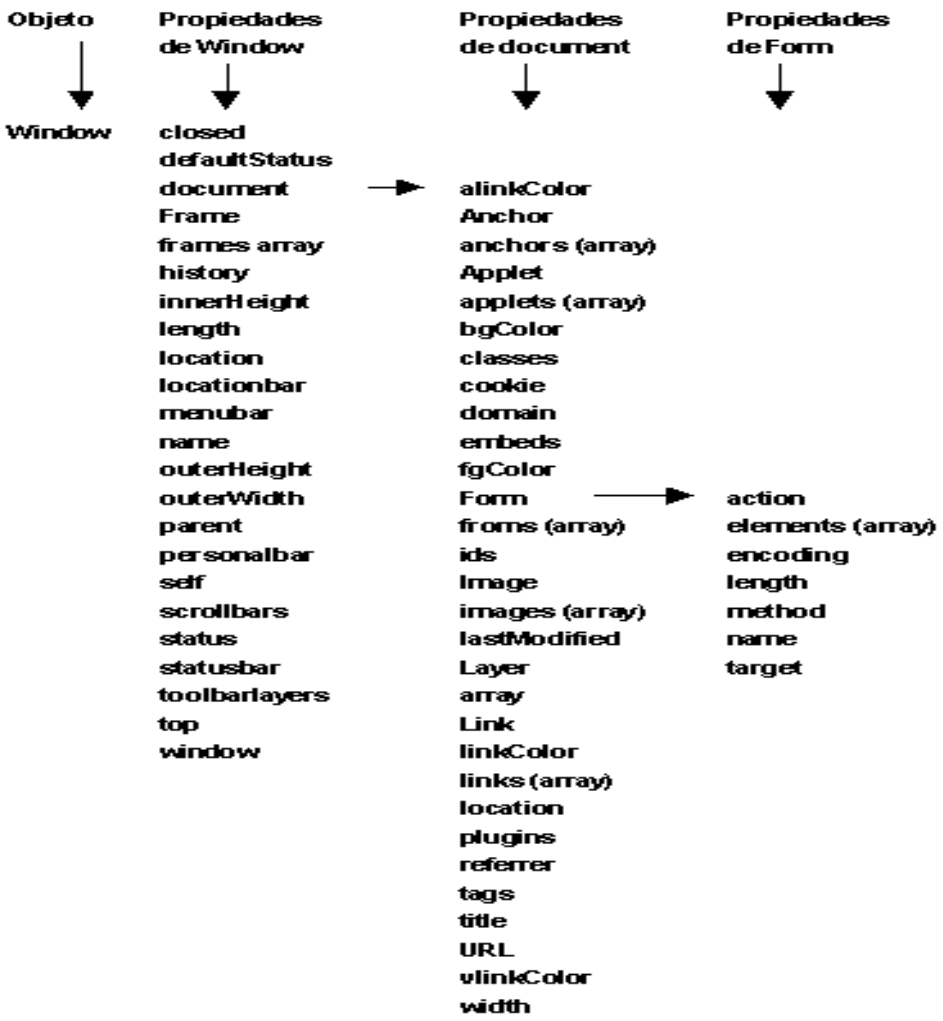
```
<HTML>
<HEAD>
   <TITLE>Prueba bgColor</TITLE>
</HEAD>
<BODY bgcolor=white>

<script>
  document.bgColor = "red"
</script>
</BODY>
</HTML>
```

En los ejemplos que hemos visto hasta ahora también hemos hecho uso de los objetos de la jerarquía del navegador. En concreto hemos utilizado mucho el método write() del objeto document para escribir un texto en la página.

```
document.write("El texto a escribir")
```

Trabajando con la Jerarquía en Javascript

Vamos a ver ahora como está compuesta esta jerarquía. Los objetos que forman parte de ella están representados en el gráfico siguiente.



Jerarquía de objetos del navegador en Javascript 1.2.
Podría faltar por recoger algún objeto, pero sirve perfectamente para hacerse una idea de cómo se organizan los objetos en la jerarquía.

Como se puede apreciar, todos los objetos comienzan en un objeto que se llama window. Este objeto ofrece una serie de métodos y propiedades para controlar la ventana del navegador. Con ellos podemos controlar el aspecto de la ventana, la barra de estado, abrir ventanas secundarias y otras cosas que veremos más adelante cuando expliquemos con detalle el objeto.

Además de ofrecer control, el objeto window da acceso a otros objetos como el documento (La página web que se está visualizando), el historial de páginas visitadas o los distintos frames de la ventana. De modo que para acceder a cualquier otro objeto de la jerarquía deberíamos empezar por el objeto window. Tanto es así que javascript entiende perfectamente que la jerarquía empieza en window aunque no lo señalemos.

En los ejemplos incluidos en el capítulo anterior podíamos haber escrito también las sentencias de acceso a la jerarquía empezando por el objeto window, de esta manera.

```
window.document.bgColor = "red"
window.document.write("El texto a escribir")
```

No lo hicimos por que quedase más claro el código y ahorrar algo de texto, pero ahora ya sabemos que toda la jerarquía empieza en el objeto window.

Las propiedades de un objeto pueden ser a su vez otros objetos

Muchas de las propiedades del objeto window son a su vez otros objetos. Es el caso de objetos como el historial de páginas web o el objeto documento, que tienen a su vez otras propiedades y métodos.

Entre ellos destaca el objeto document, que contiene todas las propiedades y métodos necesarios para controlar muchos aspectos de la página. Ya hemos visto alguna propiedad como bgColor, pero tiene muchas otras como el título de la página, las imágenes que hay incluidas, los formularios, etc. Muchas propiedades de este objeto son a su vez otros objetos, como los formularios. Los veremos todos cuando tratemos cada uno de los objetos por separado. Además, el objeto document tiene métodos para escribir en la página web y para manejar eventos de la página.

Navegación a través de la jerarquía

El objetivo de este capítulo sobre la jerarquía de objetos es aprender a navegar por ella para acceder a cualquier elemento de la página. Esta no es una tarea difícil, pero puede haber algún caso especial en el que acceder a los elementos de la página se haga de una manera que aun no hemos comentado.

Como ya dijimos, toda la jeraquía empieza en el objeto window, aunque no era necesario hacer referencia a window para acceder a cualquier objeto de la jerarquía. Luego en importancia está el objeto document, donde podemos encontrar alguna propiedad especial que merece la pena comentar por separado, porque su acceso es un poco diferente. Se trata de las propiedades que son arrays, por ejemplo la propiedad images es un array con todas las imágenes de la página web. También encontramos arrays para guardar los enlaces de la página, los applets, los formularios y las anclas.

Cuando una página se carga, el navegador construye en memoria la jerarquía de objetos. De manera adicional, construye también estos arrays de objetos. Por ejemplo, en el caso de las imágenes, va creando el array colocando en la posición 0 la primera imagen, en la posición 1 la segunda imagen y así hasta que las introduce todas. Vamos a ver un bucle que recorre todas las imágenes de la página e imprime su propiedad src, que contiene la URL donde está situada la imagen.

```
for (i=0;i<document.images.length;i++){
   document.write(document.images[i].src)
   document.write("<br>")
}
```

Utilizamos la propiedad length del array images para limitar el número de iteraciones del bucle. Luego utilizamos el método write() del objeto document pasándole el valor de cada una de las propiedades src de cada imagen.

Ahora vamos a ver el uso de otro array de objetos. En este caso vamos a acceder un poco más dentro de la jerarquía para llegar a la matriz elements de los objetos formulario, que contiene cada uno de los elementos que componen el formulario. Para ello tendremos que acceder a un formulario de la página, al que podremos acceder por el array de formularios, y dentro de él a la propiedad elements, que es otro array de objetos. Para cada elemento del formulario vamos a escribir su propiedad value, que corresponde con la propiedad value que colocamos en HTML.

```
for (i=0;i<document.forms[0].elements.length;i++){
   document.write(document. forms[0].elements[i].value)
   document.write("<br>")
}
```

Es un bucle muy parecido al que teníamos para recorrer las imágenes, con la diferencia que ahora recorremos el vector de elements, al que accedemos por la jerarquía de objetos pasando por el array de formularios en su posición 0, que corresponde con el primer formulario de la página.

Con esto hemos aprendido a movernos por la jerarquía de objetos, con lo que podremos acceder a cualquier elemento del navegador o la página. En adelante conoceremos con detalle cada uno de los objetos de la jerarquía, empezando por el objeto window y bajando por la jerarquía hasta verlos todos.

Objeto window de Javascript

Es el objeto principal en la jerarquía y contiene las propiedades y métodos para controlar la ventana del navegador. De él dependen todos los demás objetos de la jerarquía. Vamos a ver la lista de sus propiedades y métodos.

Propiedades del objeto window

A continuación podemos ver las propiedades del objeto window. Hay algunas muy útiles y otras que lo son menos.

closed
Indica la posibilidad de que se haya cerrado la ventana. (Javascript 1.1)

defaultStatus
Texto que se escribe por defecto en la barra de estado del navegador.

document
Objeto que contiene el la página web que se está mostrando.

Frame
Un objeto frame de una página web. Se accede por su nombre.

frames array
El vector que contiene todos los frames de la página. Se accede por su índice a partir de 0.

history
Objeto historial de páginas visitadas.

innerHeight
Tamaño en pixels del espacio donde se visualiza la página, en vertical. (Javascript 1.2)

innerWidth
Tamaño en pixels del espacio donde se visualiza la página, en horizontal. (Javascript 1.2)

length
Numero de frames de la ventana.

location
La URL del documento que se está visualizando. Podemos cambiar el valor de esta propiedad para movernos a otra página. Ver también la propiedad location del objeto document.

locationbar
Objeto barra de direcciones de la ventana. (Javascript 1.2)

menubar
Objeto barra de menús de la ventana. (Javascript 1.2)

name
Nombre de la ventana. Lo asignamos cuando abrimos una nueva ventana.

opener
Hace referencia a la ventana de navegador que abrió la ventana donde estamos trabajando. Se verá con detenimiento en el tratamiento de ventanas con Javascript.

outherHeight
Tamaño en pixels del espacio de toda la ventana, en vertical. Esto incluye las barras de desplazamiento, botones, etc. (Javascript 1.2)

outherWidth
Tamaño en pixels del espacio de toda la ventana, en horizontal. Esto incluye las barras de desplazamiento. (Javascript 1.2)

parent
Hace referencia a la ventana donde está situada el frame donde estamos trabajando. La veremos con detenimiento al estudiar el control de frames con Javascript.

personalbar
Objeto barra personal del navegador. (Javascript 1.2)

self
Ventana o frame actual.

scrollbars
Objeto de las barras de desplazamiento de la ventana.

status
Texto de la barra de estado.

statusbar
Objeto barra de estado del navegador. (Javascript 1.2)

toolbar
Objeto barra de herramientas. (Javascript 1.2)

top
Hace referencia a la ventana donde está situada el frame donde estamos trabajando. Como la propiedad parent.

window
Hace referencia a la ventana actual, igual que la propiedad self.

Vamos a ver un ejemplo de utilización de la propiedad status del objeto window. Esta propiedad sirve para escribir un texto en la barra de estado del navegador (la barra de debajo de la ventana). En este ejemplo hemos tenido que adelantarnos un poco en la marcha del manual, pues utilizamos un manejador de eventos y no hemos visto todavía lo que son. En concreto utilizamos el manejador de eventos onclick, que sirve para ejecutar sentencias Javascript cuando el usuario pulsa un elemento de la página.

Los manejadores de eventos se colocan en etiquetas HTML, en nuestro caso lo colocamos en un botón de formulario. Las sentencias Javascript asociadas al evento onclick del botón se ejecutarán cuando pulsemos el botón.

Veamos ya el código que hace que se cambie el texto de la barra de estado cuando pulsemos un botón.

```
<form>
<input type="Button" value="Pulsame!" onclick="window.status='Hola a todo el mundo!'">
</form>
```

Simplemente asignamos un texto a la propiedad status del objeto window. El texto que colocamos en la barra de estado está escrito entre comillas simples porque estamos escribiendo dentro de unas comillas dobles.

Métodos de window en Javascript

Vamos a ver ahora los distintos métodos que tiene el objeto window. Muchos de estos métodos habrá que verlos por separado porque son muy útiles y aun no los hemos utilizado, ahora vamos a listarlos y ya veremos algunos ejemplos.

alert(texto)
Presenta una ventana de alerta donde se puede leer el texto que recibe por parámetro

back()
Ir una página atrás en el historial de páginas visitadas. Funciona como el botón de volver de la barra de herramientas. (Javascript 1.2)

blur()
Quitar el foco de la ventana actual. (Javascript 1.1)

captureEvents(eventos)
Captura los eventos que se indiquen por parámetro (Javascript 1.2).

clearInterval()

Elimina la ejecución de sentencias asociadas a un intervalo indicadas con el método setInterval().(Javascript 1.2)

clearTimeout()
Elimina la ejecución de sentencias asociadas a un tiempo de espera indicadas con el método setTimeout().

close()
Cierra la ventana. (Javascript 1.1)

confirm(texto)
Muestra una ventana de confirmación y permite aceptar o rechazar.

find()
Muestra una ventanita de búsqueda. (Javascript 1.2 para Netscape)

focus()
Coloca el foco de la aplicación en la ventana. (Javascript 1.1)

forward()
Ir una página adelante en el historial de páginas visitadas. Como si pulsásemos el botón de adelante del navegador. (Javascript 1.2)

home()
Ir a la página de inicio que haya configurada en el explorador. (Javascript 1.2)

moveBy(pixelsX, pixelsY)
Mueve la ventana del navegador los pixels que se indican por parámetro hacia la derecha y abajo. (Javascript 1.2)

moveTo(pixelsX, pixelsY)
Mueve la ventana del navegador a la posición indicada en las coordenadas que recibe por parámetro. (Javascript 1.2)

open()
Abre una ventana secundaria del navegador. Se puede aprender a utilizarla en el reportaje de cómo abrir ventanas secundarias.

print()
Como si pulsásemos el botón de imprimir del navegador. (Javascript 1.2)

prompt(pregunta,inicializacion_de_la_respuesta)
Muestra una caja de diálogo para pedir un dato. Devuelve el dato que se ha escrito.

releaseEvents(eventos)
Deja de capturar eventos del tipo que se indique por parámetro. (Javascript 1.2)

resizeBy(pixelsAncho,pixelsAlto)
Redimensiona el tamaño de la ventana, añadiendo a su tamaño actual los valores indicados en los parámetros. El primero para la altura y el segundo para la anchura. Admite valores negativos si se desea reducir la ventana. (Javascript 1.2)

resizeTo(pixelsAncho,pixelsAlto)
Redimensiona la ventana del navegador para que ocupe el espacio en pixels que se indica por parámetro (Javascript 1.2)

routeEvent()
Enruta un evento por la jerarquía de eventos. (Javascript 1.2)

scroll(pixelsX,pixelsY)
Hace un scroll de la ventana hacia la coordenada indicada por parámetro. Este método está desaconsejado, pues ahora se debería utilizar scrollTo()(Javascript 1.1)

scrollBy(pixelsX,pixelsY)
Hace un scroll del contenido de la ventana relativo a la posición actual. (Javascript 1.2)

scrollTo(pixelsX,pixelsY)
Hace un scroll de la ventana a la posición indicada por el parámetro. Este método se tiene que utilizar en lugar de scroll. (Javascript 1.2)

setInterval()
Define un script para que sea ejecutado indefinidamente en cada intervalo de tiempo. (Javascript 1.2)

setTimeout(sentencia,milisegundos)
Define un script para que sea ejecutado una vez después de un tiempo de espera determinado.

stop()
Como pulsar el botón de stop de la ventana del navegador. (Javascript 1.2)

Para ilustrar un poco mejor el funcionamiento de alguno de estos métodos -los más extraños-, hemos creado una página web que los utiliza. El código de la página se muestra a continuación:

```
<form>
<input type="button" value="Ventana de búsqueda (Solo Netscape)" onClick="window.find()">
<br>
<br>
<input type="button" value="Mover la ventana 10 derecha,10 abajo" onClick="moveBy(10, 10)">
<br>
<br>
<input type="button" value="Mover la ventana al punto (100,10)" onClick="moveTo(100, 10)">
<br>
<br>
<input type="button" value="Imprimir esta página" onClick="print()">
<br>
<br>
<input type="button" value="Aumenta la ventana 10 ancho,10 largo" onClick="resizeBy(10, 10)">
<br>
<br>
<input type="button" value="Fija el tamaño de la ventana en 400 x 200" onClick="resizeTo(400,
200)">
<br>
<br>
<input type="button" value="Scroll arriba del todo" onClick="scroll(0,0)">
<br>
<br>
<input type="button" value="Scroll arriba 10 pixels" onClick="scrollBy(0,-10)">
</form>
```

Estos ejemplos son muy simples, aunque poco prácticos. Únicamente se trata de una serie de botones que, al pulsarlos, llaman a otros tantos métodos del objeto window. En el atributo onclick de la etiqueta del botón se indican las sentencias Javascript que queremos que se ejecuten cuando se pulsa sobre dicho botón.

En el capítulo siguiente veremos otros ejemplos realizados con métodos del objeto window de Javascript, un poco más detallados.

Ejemplos de métodos de window

Ahora vamos a realizar algún ejemplo de utilización de los métodos de la ventana. Nos vamos a centrar en los ejemplos que sirven para sacar cajas de diálogo, que son muy útiles.

Caja de alerta

Para sacar un texto en una ventanita con un botón de aceptar. Recibe el texto por parámetro.

```
window.alert("Este es el texto que sale")
```

Como el objeto window se sobreentiende podemos escribirlo así.

```
alert("Este es el texto que sale")
```

Saca una ventana de alerta.

Caja de confirmación

Muestra una ventana con un texto indicado por parámetro y un botón de aceptar y otro de rechazar. Dependiendo del botón que se pulsa devuelve un true (si se pulsa aceptar) o un false (si se pulsa rechazar).

```
<script>
var respuesta = confirm("Aceptame o rechazame")
alert ("Has pulsado: " + respuesta)
</script>
```

Este script muestra una caja de diálogo confirm y luego muestra en otra caja de diálogo alert el contenido de la variable que devolvió la caja de diálogo.

Caja de introducción de un dato

Muestra una caja de diálogo donde se formula una pregunta y hay un campo de texto para escribir una respuesta. El campo de texto aparece relleno con lo que se escriba en el segundo parámetro del método. También hay un botón de aceptar y otro de cancelar. En caso de pulsar aceptar, el método devuelve el texto que se haya escrito. Si se pulsó cancelar devuelve null.

El ejemplo siguiente sirve para pedir el nombre de la persona que está visitando la página y luego mostrar en la página un saludo personalizado. Utiliza un bucle para repetir la toma de datos siempre que el nombre de la persona sea null (porque pulsó el botón de cancelar) o sea un string vacío (porque no escribió nada).

```
<script>
nombre = null
while (nombre == null || nombre == ""){
   nombre = prompt("Dime tu nombre:","")
}
document.write("<h1>Hola " + nombre + "</h1>")
</script>
```

Si nos fijamos en la caja prompt veremos que recibe dos parámetros. El segundo era el texto por defecto que sale en la caja como respuesta. Lo hemos dejado como un string vacío para que no salga nada como texto por defecto.

Hasta aquí los ejemplos de los métodos del objeto window. De todos modos, en el resto del manual tendremos ocasión de ver cómo trabajar con muchas propiedades y métodos de este objeto.

Objeto document en Javascript

Con el objeto document se controla la página web y todos los elementos que contiene. El objeto document es la página actual que se está visualizando en ese momento. Depende del objeto window, pero también puede depender del objeto frame en caso de que la página se esté mostrando en un frame.

Propiedades del objeto document

Veamos una lista de las propiedades del objeto document y luego veremos algún ejemplo.

alinkColor
Color de los enlaces activos

Anchor
Un ancla de la página. Se consigue con la etiqueta <A name="nombre_del_ancla">. Se accede por su nombre.

anchors array
Un array de las anclas del documento.

Applet
Un applet de la página. Se accede por su nombre. (Javascript 1.1)

applets array
Un array con todos los applets de la página. (Javascript 1.1)

Area
Una etiqueta <AREA>, de las que están vinculadas a los mapas de imágenes (Etiqueta). Se accede por su nombre. (Javascript 1.1)

bgColor
El color de fondo del documento.

classes
Las clases definidas en la declaración de estilos CSS. (Javascript 1.2)

cookie
Una cookie

domain

Nombre del dominio del servidor de la página.

Embed
Un elemento de la pagina incrustado con la etiqueta <EMBED>. Se accede por su nombre. (Javascript 1.1)

embeds array
Todos los elementos de la página incrustados con <EMBED>. (Javascript 1.1)

fgColor
El color del texto. Para ver los cambios hay que reescribir la página.

From
Un formulario de la página. Se accede por su nombre.

forms array
Un array con todos los formularios de la página.

ids
Para acceder a estilos CSS. (Javascript 1.2)

Image
Una imagen de la página web. Se accede por su nombre. (Javascript 1.1)

images array
Cada una de las imágenes de la página introducidas en un array. (Javascript 1.1)

lastModified
La fecha de última modificación del documento.

linkColor
El color de los enlaces.

Link
Un enlace de los de la página. Se accede por su nombre.

links array
Un array con cada uno de los enlaces de la página.

location
La URL del documento que se está visualizando. Es de solo lectura.

referrer
La página de la que viene el usuario.

tags
Estilos definidos a las etiquetas de HTML en la página web. (Javascript 1.2)

title
El titulo de la página.

URL
Lo mismo que location, pero es aconsejable utilizar location ya que URL no existe en todos los navegadores.

vlinkColor
El color de los enlaces visitados.

Ejemplos de propiedades de document

Después de estudiar las propiedades del objeto document, vamos a ver algún ejemplo para ilustrar el modo de acceso y utilización de las mismas.

Ejemplo con la propiedad bgColor

Vamos a utilizar el evento onclick para colocar tres botones en la página que al pulsarlos nos cambie el color del fondo de la página.

```
<script>
function cambiaColor(colorin){
   document.bgColor = colorin
```

```
}

</script>
<form>
<input type="Button" value="Rojo" onclick="cambiaColor('ff0000')">
<input type="Button" value="Verde" onclick="cambiaColor('00ff00')">
<input type="Button" value="Azul" onclick="cambiaColor('0000ff')">
</form>
```

Primero definimos una función que será la encargada de cambiar el color y luego tres botones que llamarán a la función cuando sean pulsados pasándole el color como parámetro.

Propiedad title

La propiedad title guarda la cadena que conforma el título de nuestra página web. Si accedemos a dicha propiedad obtendremos el título y si la cambiamos, cambiará el título de la página web.

> **Nota:** recordamos que el título se puede ver en la barra de arriba del todo de la ventana del navegador.

Vamos a mostrar e título de la página en una caja de alerta.

```
alert (document.title)
```

Ahora vamos a hacer una función que modifica el título de la página , asignándole el texto que le llegue por parámetro.

```
function cambiaTitulo(texto){
   document.title = texto
}
```

Como en el ejemplo anterior, vamos a crear varios botones que llamen a la función pasándole distintos textos, que se colocarán en el título de la página.

```
<form>
<input type="Button" value="Titulo = Hola a todos" onclick="cambiaTitulo('Hola a todos')">
<input type="Button" value="Titulo = Bienvenidos a mi página web" onclick="cambiaTitulo('Bienvenidos a mi página web')">
<input type="Button" value="Titulo = Más dias que longanizas" onclick="cambiaTitulo('Más dias que longanizas')">
</form>
```

Métodos de document

El objeto document, localizado debajo del objeto window en la jerarquía de objetos de Javascript, también tiene una lista de métodos interesantes. La vemos a continuación.

captureEvents()
Para capturar los eventos que ocurran en la página web. Recibe como parámetro el evento que se desea capturar.

close()
Cierra el flujo del documento. (Se verá más adelante en este manual un artículo sobre el flujo del documento)

contextual()
Ofrece una línea de control de los estilos de la página. En el caso que deseemos especificarlos con Javascript.

getSelection()
Devuelve un string que contiene el texto que se ha seleccionado. Sólo funciona en Netscape.

handleEvent()
Invocas el manejador de eventos del elemento especificado.

open()
Abre el flujo del documento.

releaseEvents()

Liberar los eventos capturados del tipo que se especifique, enviándolos a los objetos siguientes en la jerarquía.

routeEvent()
Pasa un evento capturado a través de la jerarquía de eventos habitual.

write()
Escribe dentro de la página web. Podemos escribir etiquetas HTML y texto normal.

writeln()
Escribe igual que el método write(), aunque coloca un salto de línea al final.

Los eventos de document sirven principalmente para controlar dos cosas. Un grupo nos ofrece una serie de funciones para el control de los documentos, como escribir, abrirlos y cerrarlos. Los veremos en el capítulo siguiente que habla sobre el control del flujo de escritura del documento. El otro grupo ofrece herramientas para el control de los eventos en el documento y lo veremos más adelante cuando tratemos con detenimiento el tema de eventos.

Se nos queda un poco suelto el método getSelection(), que sólo funciona en los navegadores de Netscape y, por tanto, no resulta muy útil para aplicaciones que deseemos que sean compatibles en todos los sistemas. Aun así, haremos el ejemplo sobre este método, ya que los otros los vamos a ver en siguientes capítulos.

El ejemplo consiste en una página que tiene un poco de texto y un botón. En la página podremos seleccionar algo de texto y luego apretar el botón, que llama a una función que muestra en una caja alert el texto que se ha seleccionado. El código es el siguiente:

```
<html>
<head>
<title>Rescatar lo seleccionado</title>
<script language="JavaScript">
function mostrarSeleccionado(){
   alert("Has seleccionado:\n" + document.getSelection())
}
</script>
</head>

<body>
<h1>Rescatar lo seleccionado</h1>
<br>

<form>
<input type="button" value="pulsame!" onClick="mostrarSeleccionado()">
</form>

Selecciona cualquier texto de la página y pulsa sobre el botón.

</body>
</html>
```

Flujo de escritura del documento

Acerca del objeto document también es interesante hablar un poco sobre el flujo de escritura del documento o página web.

Cuando el navegador lee una página web la va interpretando y escribiendo en su ventana. El proceso en el que el navegador está escribiendo la página le llamamos flujo de escritura del documento. El flujo comienza cuando se empieza a escribir la página y dura hasta que ésta ha terminado de escribirse. Una vez terminada la escritura de la página el flujo de escritura del documento se cierra automáticamente. Con esto termina la carga de la página.

Una vez cerrado el flujo no se puede escribir en la página web, ni texto ni imágenes ni otros elementos.

En javascript tenemos que respetar el flujo de escritura del documento forzosamente. Es por ello que sólo podemos ejecutar sentencias document.write() (método write() del objeto document) cuando está abierto el flujo de escritura del documento, o lo que es lo mismo, mientras se está cargando la página.

Si recordamos las dos formas de ejecutar un script en Javascript:

- Ejecución de los scripts mientras que carga la página. Aquí podremos ejecutar document.write() y lo hemos hecho habitualmente en los ejemplos anteriores.
- Ejecución de los scripts cuando la página ha sido cargada, como respuesta a un evento del usuario. Aquí no podemos hacerlo porque la página ha terminado de cargarse, de hecho, no lo hemos hecho nunca hasta ahora.

Hay un matiz que dar a que no se puede escribir en la página cuando ya está cerrado el flujo. En realidad si que se puede, pero necesitamos abrir el flujo otra vez para escribir en la página, tanto es así que aunque nosotros no lo abramos explícitamente Javascript se encargará de ello. Lo que tiene que quedar claro es que si hacemos un document.write() el flujo tiene que estar abierto y si no lo está se abrirá. El problema de abrir el flujo de escritura del documento una vez ya está escrita la página es que se borra todo su contenido.

Para que quede claro vamos a hacer un script para escribir en la página una vez ésta ha sido cargada. Simplemente necesitamos un botón y al pulsarlo ejecutar el método write() del objeto document.

```
<form>
<INPUT type=button value=escribir onclick="window.document.write('hola')">
</form>
```

Si nos fijamos, en el manejador de eventos onclick hemos colocado la jerarquía de objetos desde el principio, es decir, empezando por el objeto window. Esto es porque hay algunos navegadores que no sobreentienden el objeto window en las sentencias escritas en los manejadores de eventos.

El resultado de la ejecución puede variar de un navegador a otro, pero en cualquier caso se borrará la página que se está ejecutando.

Métodos open() y close() de document

Los métodos open() y close() del objeto document sirven para controlar el flujo del documento desde Javascript. Nos permiten abrir y cerrar el documento explícitamente.

El ejemplo de escritura en la página anterior se debería haber escrito con su correspondiente apertura y cierre del documento y hubiese quedado algo parecido a esto.

```
<script>
function escribe(){
   document.open()
   window.document.write('Hola')
   document.close()
}
</script>
<form>
<INPUT type=button value=escribir onclick="escribe()">
</form>
```

Vemos que ahora no escribimos las sentencias dentro del manejador, porque, cuando hay más de una sentencia, queda más claro poner una llamada a una función y en la función colocamos las sentencias que queramos.

Trabajo con formularios en Javascript

Para continuar vamos a ver una serie de capítulos enfocados a aprender a trabajar con los formularios, acceder a sus elementos para controlar sus valores y actualizarlos.

El objeto form depende en la jerarquía de objetos del objeto document. En un objeto form podemos encontrar algunos métodos y propiedades, pero lo más destacado que podremos encontrar son cada uno de los elementos del formulario. Es decir, de un formulario dependen todos los elementos que hay dentro, como pueden ser campos de texto, cajas de selección, áreas de texto, botones de radio, etc.

Para acceder a un formulario desde el objeto document podemos hacerlo de dos formas.

1. A partir de su nombre, asignado con el atributo NAME de HTML.
2. Mediante la matriz de formularios del objeto document, con el índice del formulario al que queremos acceder.

Para este formulario

```
<FORM name="f1">
```

```
<INPUT type=text name=campo1>
<INPUT type=text name=campo2>
</FORM>
```

Podremos acceder con su nombre de esta manera.

```
document.f1
```

O con su índice, si suponemos que es el primero de la página.

```
document.forms[0]
```

De similar manera accedemos a los elementos de un formulario, que dependen del objeto form.

1. A partir del nombre del objeto asignado con el atributo NAME de HTML.
2. Mediante la matriz de elementos del objeto form, con el índice del elemento al que queremos acceder.

Podríamos acceder al campo 1 del anterior formulario de dos maneras. Con su nombre lo haríamos así.

```
document.f1.campo1
```

o a partir del array de elementos.

document.f1.elements[0] (utilizamos el índice 0 porque es el primer elemento y en Javascript los arrays empiezan por 0.)

Si deseamos acceder al segundo campo del formulario escribiríamos una de estas dos cosas:

```
document.f1.campo2
document.f1.elements[1]
```

recordamos que también podemos acceder a un fomulario por el array de forms, indicando el índice del formulario al que acceder. De este modo, el acceso al campo2 sería el siguiente:

```
document.forms[0].campo2
document.forms[0].elements[1]
```

En estos casos hemos supuesto que este formulario es el primero que hay escrito en el código HTML, por eso accedemos a él con el índice 0.

Esperamos que haya quedado claro el acceso a formularios y sus campos. Pasaremos entonces, sin más, a un ejemplo para practicar todo esto.

Ej. trabajo con formularios. Calculadora sencilla

Para ilustrar un poco el trabajo con formularios, vamos a realizar un ejemplo práctico. Puede que algunas cosas que vamos a tratar queden un poco en el aire porque no se hayan explicado con detenimiento antes, pero seguro que nos sirve para enterarnos de cómo se trabaja con formularios y las posibilidadedes que tenemos.

Ejemplo calculadora sencilla

En este ejemplo vamos a construir una calculadora, aunque bastante sencilla, que permita realizar las operaciones básicas. Para hacer la calculadora vamos a realizar un formulario en el que vamos a colocar tres campos de texto, los dos primeros para los operandos y un tercero para el resultado. Además habrán unos botones para hacer las operaciones básicas.

El formulario de la calculadora se puede ver aquí.

```
<form name="calc">
<input type="Text" name="operando1" value="0" size="12">
<br>
<input type="Text" name="operando2" value="0" size="12">
<br>
<input type="Button" name="" value=" + " onclick="calcula('+')">
<input type="Button" name="" value=" - " onclick="calcula('-')">
<input type="Button" name="" value=" X " onclick="calcula('*')">
<input type="Button" name="" value=" / " onclick="calcula('/')">
<br>
```

```
<input type="Text" name="resultado" value="0" size="12">
</form>
```

Mediante una función vamos a acceder a los campos del formulario para recoger los operandos en dos variables. Los campos de texto tienen una propiedad llamada value que es donde podemos obtener lo que tienen escrito en ese momento. Mas tarde nos ayudaremos de la función eval() para realizar la operación. Pondremos por último el resultado en el campo de texto creado en tercer lugar, utilizando también la propiedad value del campo de texto.

A la función que realiza el cálculo (que podemos ver a continuación) la llamamos apretando los botones de cada una de las operaciones. Dichos botones se pueden ver en el formulario y contienen un atributo onclick que sirve para especificar las sentencias Javascript que deseamos que se ejecuten cuando el usuario pulse sobre él. En este caso, la sentencia a ejecutar es una llamada a la función calcula() pasando como parámetro el símbolo u operador de la operación que deseamos realizar.

El script con la función calcula()

```
<script>
function calcula(operacion){
   var operando1 = document.calc.operando1.value
   var operando2 = document.calc.operando2.value
   var result = eval(operando1 + operacion + operando2)
   document.calc.resultado.value = result
}
</script>
```

La función eval(), recordamos, que recibía un string y lo ejecutaba como una sentencia Javascript. En este caso va a recibir un número que concatenado a una operación y otro número será siempre una expresión aritmética que eval() solucionará perfectamente.

El acceso a otros elementos de los formularios se hace de manera parecida en cuanto respecta a la jerarquía de objetos, aunque como cada elemento tiene sus particularidades las cosas que podremos hacer con ellos diferirán un poco. Lo veremos un poco más adelante.

Propiedades y métodos del objeto form

Vamos a ver ahora el objeto form por si solo, para destacar sus propiedades y métodos.

Propiedades del objeto form

Tiene unas cuantas propiedades para ajustar sus atributos mediante Javascript.

action
Es la acción que queremos realizar cuand se submite un formulario. Se coloca generalmente una dirección de correo o la URL a la que le mandaremos los datos. Corresponde con el atributo ACTION del formulario.

elements array
La matriz de elementos contiene cada uno de los campos del formulario.

encoding
El tipo de codificación del formulario

length
El número de campos del formulario.

method
El método por el que mandamos la información. Corresponde con el atributo METHOD del formulario.

name
El nombre del formulario, que corresponde con el atributo NAME del formulario.

target
La ventana o frame en la que está dirigido el formulario. Cuando se submita se actualizará la ventana o frame indicado. Corresponde con el atributo target del formulario.

Ejemplos de trabajo con las propiedades

Con estas propiedades podemos cambiar dinámicamente con Javascript los valores de los atributos del formulario para hacer con él lo que se desee dependiendo de las exigencias del momento.

Por ejemplo podríamos cambiar la URL que recibiría la información del formulario con la instrucción.

```
document.miFormulario.action = "miPágina.asp"
```

O cambiar el target para submitir un formulario en una posible ventana secundaria llamada mi_ventana.

```
document.miFormulario.target = "mi_ventana"
```

Métodos del objeto form

Estos son los métodos que podemos invocar con un formulario.

submit()
Para hacer que el formulario se submita, aunque no se haya pulsado el botón de submit.

reset()
Para reinicializar todos los campos del formulario, como si se hubiese pulsado el botón de reset. (Javascript 1.1)

Ejemplo de trabajo con los métodos

Vamos a ver un ejemplo muy sencillo sobre cómo validar un formulario para submitirlo en caso de que esté relleno. Para ello vamos a utilizar el método submit() del formulario.

El mecanismo es el siguiente: en vez de colocar un botón de submit colocamos un botón normal (<INPUT type="button">) y hacemos que al pulsar ese botón se llame a una función que es la encargada de validar el formulario y, en caso de que esté correcto, submitirlo.

El formulario quedaría así.

```
<form name="miFormulario" action="mailto:promocion@guiarte.com" enctype="text/plain">
<input type="Text" name="campo1" value="" size="12">
<input type="button" value="Enviar" onclick="validaSubmite()">
</form>
```

Nos fijamos en que no hay botón de submit, sino un botón normal con una llamada a una función que podemos ver a continuación.

```
function validaSubmite(){
   if (document.miFormulario.campo1.value == "")
      alert("Debe rellenar el formulario")
   else
      document.miFormulario.submit()
}
```

En la función se comprueba si lo que hay escrito en el formulario es un string vacío. Si es así se muestra un mensaje de alerta que informa que se debe rellenar el formulario. En caso de haya algo en el campo de texto submite el formulario utilizando el método submit del objeto form.

Control de campos de texto con Javascript

Vamos a ver ahora los campos donde podemos guardar cadenas de texto, es decir, los campos de texto, password y hidden. Hay otro campo relacionado con la escritura de texto, el campo TextArea, que veremos más adelante.

Campo Text

Es el campo que resulta de escribir la etiqueta <INPUT type="text">. Lo hemos utilizado hasta el momento en varios ejemplos, pero vamos a parar un momento en él para describir sus propuedades y métodos.

Propiedades del campo text

Vemos la lista de propiedades de estos tipos de campo.

defaultValue
Es el valor por defecto que tiene un campo. Lo que contiene el atributo VALUE de la etiqueta <INPUT>.

form
Hace referencia al formulario.

name
Contiene el nombre de este campo de formulario

type
Contiene el tipo de campo de tormulario que es.

value
El texto que hay escrito en el campo.

Vamos a ver un ejemplo sobre lo que puede hacer la propiedad defaultValue. En este ejemplo tenemos un formulario y un botón de reset. Si pulsamos el botón de reset el campo de texto se vacía porque su value de HTML era un string vacío. Pero si pulsamos el botón siguiente llamamos a una función que cambia el valor por defecto de ese campo de texto, de modo que al pulsar el botón de reset mostrará el nuevo valor por defecto.

Este es el código de la página completa.

```
<html>
<head>
  <title>Cambiar el valor por defecto</title>
  <script>
   function cambiaDefecto(){
     document.miFormulario.campo1.defaultValue = "Hola!!"
   }
  </script>
</head>

<body>
<form name="miFormulario" action="mailto:promocion@guiarte.com" enctype="text/plain">
<input type="Text" name="campo1" value="" size="12">
<input type="Reset">
<br>
<br>
<input type="button" value="Cambia valor por defecto" onclick="cambiaDefecto()">
</form>
</body>
</html>
```

Métodos del objeto Text

Se pueden invocar los siguientes métodos sobre los objetos tipo Text.

blur()
Retira el foco de la aplicación del campo de texto.

focus()
Pone el foco de la aplicación en el campo de texto.

select()
Selecciona el texto del campo.

Como ejemplo vamos a mostrar una función que selecciona el texto de un campo de texto de un formulario como el de la página del ejemplo anterior. Para hacerlo hemos utilizado dos métodos, el primero para pasar el foco de la aplicación al campo de texto y el segundo para seleccionar el texto.

```
function seleccionaFoco(){
   document.miFormulario.campo1.focus()
   document.miFormulario.campo1.select()
}
```

Campos Password

Estos funcionan igual que los hidden, con la peculiaridad que el contenido del campo no puede verse escrito en el campo, por lo que salen asteríscos en lugar del texto.

Campos Hidden

Los campos hidden son como campos de texto que están ocultos para el usuario, es decir, que no se ven en la página. Son muy útiles en el desarrollo de webs para pasar variables en los formularios a las que no debe tener acceso el usuario.

Se colocan en con HTML con la etiqueta <INPUT type=hidden> y se rellenan de datos con su atributo value. Mas tarde podremos cambiar el dato que figura en el campo accediendo a la propiedad value del campo de texto igual que lo hemos hecho antes.

document.miFormulario.CampoHidden.value = "nuevo texto"

El campo hidden sólo tiene algunas de las propiedades de los campos de texto. En concreto tiene la propiedad value y las propiedades que son comunes de todos los campos de formulario: name, from y type, que ya se describieron para los campos de texto.

Control de Checkbox en Javascript

Los checkbox son las unas cajas que permiten marcarlas o no para verificar alguna cosa en un formulario. Podemos ver una caja checkbok a continuación.

Los checkbox se consiguen con la etiqueta <INPUT type=checkbox>. Con el atributo NAME de la etiqueta <INPUT> le podemos dar un nombre para luego acceder a ella con javascript. Con el atributo CHECKED indicamos que el campo debe aparecer chequeado por defecto.

Con Javascript, a partir de la jerarquía de objetos del navegador, tenemos acceso al checkbox, que depende del objeto form del formulario donde está incluido.

Propiedades de un checkbox

Las propiedades que tiene un checkbox son las siguientes.

checked
Informa sobre el estado del checkbox. Puede ser true o false.

defaultChecked
Si está chequeada por defecto o no.

value
El valor actual del checkbox.

También tiene las propiedades form, name, type como cualquier otro elemento de formulario.

Métodos del checkbox

Veamos la lista de los métodos de un checkbox.

click()
Es como si hiciésemos un click sobre el checkbox, es decir, cambia el estado del checkbox.

blur()
Retira el foco de la aplicación del checkbox.

focus()
Coloca el foco de la aplicación en el checkbox.

Para ilustrar el funcionamiento de las checkbox vamos a ver una página que tiene un checkbox y tres botones. Los dos primeros para mostrar las propiedades checked y value y el tercero para invocar a su método click() con objetivo de simular un click sobre el checkbox.

```
<html>
<head>
   <title>Ejemplo Checkbox</title>
<script>
function alertaChecked(){
   alert(document.miFormulario.miCheck.checked)
}
function alertaValue(){
   alert(document.miFormulario.miCheck.checked)
}
function metodoClick(){
   document.miFormulario.miCheck.click()
}
</script>
```

```
</head>

<body>
<form name="miFormulario" action="mailto:promocion@guiarte.com" enctype="text/plain">
<input type="checkbox" name="miCheck">
<br>
<br>
<input type="button" value="informa de su propiedad checked" onclick="alertaChecked()">
<input type="button" value="informa de su propiedad value" onclick="alertaValue()">
<br>
<br>
<input type="button" value="Simula un click" onclick="metodoClick()">
</form>
</body>
</html>
```

Control de botones de radio en Javascript

El botón de radio (o radio button en inglés) es un elemento de formulario que permite seleccionar una opción y sólo una, sobre un conjunto de posibilidades. Se puede ver a continuación.

 Blanco

 Rojo

 Verde

---

> **Nota:** En la parte de arriba podemos ver tres radio buttons en lugar de uno solo. Se colocan tres botones porque así podemos examinar su funcionamiento al formar parte de un grupo. Veamos que al seleccionar una opción se deselecciona la opción que estuviera marcada antes.

Se consiguen con la etiqueta <INPUT type=radio>. Con el atributo NAME de la etiqueta <INPUT> les damos un nombre para agrupar los radio button y que sólo se pueda elegir una opción entre varias. Con el atributo value indicamos el valor de cada uno de los radio buttons. El atributo checked nos sirve para indicar cuál de los radio butons tiene que estar seleccionado por defecto.

> **Referencia:** Explicamos en detalle la creación de botones de radio en nuestro manual de HTML, en el capítulo Otros elementos de formularios.

Cuando en una página tenemos un conjunto de botones de radio se crea un objeto radio por cada uno de los botones. Los objetos radio dependen del formulario y se puede acceder a ellos por el array de elements, sin embargo también se crea un array con los botones de radio. Este array depende del formulario y tiene el mismo nombre que los botones de radio.

Propiedades del objeto radio

Veamos una lista de las propiedades de este elemento.

checked
Indica si está chekeado o no un botón de radio.

defaultChecked
Su estado por defecto.

value
El valor del campo de radio, asignado por la propiedad value del radio.

Length (como propiedad del array de radios)
El número de botones de radio que forman parte en el grupo. Accesible en el vector de radios.

Métodos del objeto radio

Son los mismos que los que tenía el objeto checkbox.

Ejemplo de utilización

Veamos con un ejemplo el método de trabajo con los radio buttons en el que vamos a colocar un montón de ellos y cada uno tendrá asociado un color. También habrá un botón y al pulsarlo cambiaremos el color de fondo de la pantalla al color que esté seleccionado en el conjunto de botones de radio.

Vamos a ver la página entera y luego la comentamos.

```
<html>
<head>
   <title>Ejemplo Radio Button</title>
<script>
function cambiaColor(){
   var i
   for (i=0;i<document.fcolores.colorin.length;i++){
      if (document.fcolores.colorin[i].checked)
         break;
   }
   document.bgColor = document.fcolores.colorin[i].value
}
</script>
</head>

<body>
<form name=fcolores>
<input type="Radio" name="colorin" value="ffffff" checked> Blanco
<br>
<input type="Radio" name="colorin" value="ff0000"> Rojo
<br>
<input type="Radio" name="colorin" value="00ff00"> Verde
<br>
<input type="Radio" name="colorin" value="0000ff"> Azul
<br>
<input type="Radio" name="colorin" value="ffff00"> Amarillo
<br>
<input type="Radio" name="colorin" value="00ff00"> Turquesa
<br>
<input type="Radio" name="colorin" value="ff00ff"> Morado
<br>
<input type="Radio" name="colorin" value="000000"> Negro
<br>
<br>
<input type="Button" name="" value="Cambia Color" onclick="cambiaColor()">
</form>
</body>
</html>
```

Primero podemos fijarnos en el formulario y en la lista de botones de radio. Todos se llaman "colorin", así que están asociados en un mismo grupo. Además vemos que el atributo value de cada botón cambia. También vemos un botón abajo del todo.

Con esta estructura de formulario tendremos un array de elements de 9 elementos, los 8 botones de radio y el botón de abajo.

Además tendremos un array de botones de radio que se llamará colorín y depende del formulario, accesible de esta manera.

document.form.colorin

Este array tiene en cada posición uno de los botones de radio. Así en la posición 0 está el botón del color blanco, en la posición 1 el del color rojo... Para acceder a esos botones de radio lo hacemos con su índice.

document.fcolores.colorin[0]

Si queremos acceder por ejemplo a la propiedad value del último botón de radio escribimos lo siguiente.

document.fcolores.colorin[7].value

La propiedad length del array de radios nos indica el número de botones de radio que forman parte del grupo.

document.fcolores.colorin.length

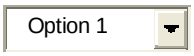
En este caso la propiedad length valdrá 8.

Con estas notas podremos entender más o menos bien la función que se encarga de encontrar el radio button seleccionado y cambiar el color de fondo de la página.

Se define una variable en la que introduciremos el índice del radio button que tenemos seleccionado. Para ello vamos recorriendo el array de botones de radio hasta que encontramos el que tiene su propiedad checked a true. En ese momento salimos del bucle, con lo que la variable i almacena el índice del botón de radio seleccionado. En la última línea cambiamos el color de fondo a lo que hay en el atributo value del radio button seleccionado.

Control de campos select con Javascript

El objeto select de un formulario es una de esas listas desplegables que nos permiten seleccionar un elemento. Se despliegan apretando sobre una flecha, a continuación se puede escoger un elemento y para acabar se vuelven a plegar. Se puede ver un elemento select de un formulario a continuación.



Uno de estos elementos se puede obtener utilizando la etiqueta <SELECT> dentro de un formulario. A la etiqueta le podemos añadir un atributo para darle el nombre, NAME, para luego acceder a ese campo mediante Javascript. Para expresar cada una de las posibles opciones del campo select utilizamos una etiqueta <OPTION> a la que le introducimos un atributo VALUE para expresar su valor. El texto que colocamos después de la etiqueta <OPTION> sirve como etiqueta de esa opción, es el texto que ve el usuario asociado a esa opción.

Propiedades del objeto select

Vamos a ver una lista de las propiedades de este elemento de formulario.

length
Guarda la cantidad de opciones del campo select. Cantidad de etiquetas <OPTION>

Option
Hace referencia a cada una de sus opciones. Son por si mismas objetos.

options
Un array con cada una de las opciones del select.

selectedIndex
Es el índice de la opción que se encuentra seleccionada.

Aparte de las conocidas propiedades comunes a todos los elementos de formulario: form y name y type.

Métodos del objeto select

Los métodos son solamente 2 y ya conocemos su significado.

blur()
Para retirar el foco de la aplicación de ese elemento de formulario.

focus()
Para poner el foco de la aplicación.

Objeto option

Tenemos que pararnos a ver también este objeto para entender bien el campo select. Recordamos que las option son las distintas opciones que tiene un select, expresadas con la etiqueta <OPTION>.

Propiedades de option

Estos objetos sólo tienen propiedades, no tienen métodos. Vamos a verlas.

defaultSelected
Indica con un true o un false si esa opción es la opción por defecto. La opción por defecto se consigue con HTML colocando el atributo selected a un option.

index

El índice de esa opción dentro del select.

selected
Indica si esa opción se encuentra seleccionada o no.

text
Es el texto de la opción. Lo que puede ver el usuario en el select, que se escribe después de la etiqueta <OPTION>.

value
Indica el valor de la opción, que se introduce con el atributo VALUE de la etiqueta <OPTION>.

Ejemplo de acceso a un select

Vamos a ver un ejemplo sobre cómo se accede a un select con Javascript, como podemos acceder a sus distintas propiedades y a la opción seleccionada.

Vamos a empezar viendo el formulario que tiene el select con el que vamos a trabajar. Es un select que sirve para valorar el web que estamos visitando.

```
<form name="fomul">
Valoración sobre este web:
<select name="miSelect">
<option value="10">Muy bien
<option value="5" selected>Regular
<option value="0">Muy mal
</select>
<br>
<br>
<input type=button value="Dime propiedades" onclick="dimePropiedades()">
</form>
```

Ahora vamos a ver una función que recoge las propiedades más significativas del campo select y las presenta en una caja alert.

```
function dimePropiedades(){
   var texto
   texto = "El numero de opciones del select: " + document.formul.miSelect.length
   var indice = document.formul.miSelect.selectedIndex
   texto += "\nIndice de la opcion escogida: " + indice
   var valor = document.formul.miSelect.options[indice].value
   texto += "\nValor de la opcion escogida: " + valor
   var textoEscogido = document.formul.miSelect.options[indice].text
   texto += "\nTexto de la opcion escogida: " + textoEscogido
   alert(texto)
}
```

Esta función crea una variable de texto donde va introduciendo cada una de las propiedades del select. La primera contiene el valor de la propiedad length del select, la segunda el índice de la opción seleccionada y las dos siguientes contienen el valor y el texto de la opción seleccionada. Para acceder a la opción seleccionada utilizamos el array options con el índice recogido en las segunda variable.

Propiedad value de un objeto select

Para acceder al valor seleccionado de un campo select podemos utilizar la propiedad value del campo select en lugar de acceder a partir del vector de options.

Para el anterior formulario sería algo parecido a esto.

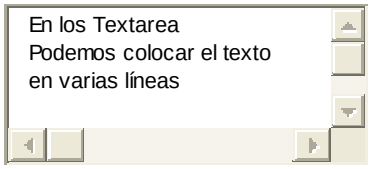
```
formul.miSelect.value
```

Sin embargo, esta propiedad sólo está presente en navegadores Internet Explorer, por lo que es recomendable acceder utilizando el vector de options con el índice de la posición seleccionada si deseamos que la página sea compatible con todos los navegadores. Hemos querido añadir este punto para que, si alguna vez utilizamos o vemos utilizar este método de acceso, sepamos su pega y porque es mejor utilizar el vector de options.

Control de elementos Textarea en Javascript

Para acabar de describir todos los elementos de formularios vamos a ver el objeto textarea que es un elemento que presenta un lugar para escribir texto, igual que los campos text, pero con la particularidad que podemos escribir varias líneas a la vez.

El campo textarea se puede ver a continuación.



Un campo textarea se consigue con la etiqueta <TEXTAREA>. Con el atributo name le podemos dar un nombre para acceder al campo textarea mediante Javascript. Otros atributos interesantes son cols y rows que sirven para indicar la anchura y altura del campo textarea en caracteres, cols indica el número de columnas y rows el de filas. aunque no se puede acceder a ellos con Javascript. El valor por defecto de un campo textarea se coloca entre las etiquteta <TEXTAREA> y su correspondiente cierre.

Propiedades de textarea

Se puede ver una lista de las propiedades disponibles en un textarea a continuación, que son los mismos que un campo de texto.

defaultValue
Que contiene el valor por defecto del textarea.

value
Que contiene el texto que hay escrito en el textarea.

Además tiene las conocidas propiedades de elementos de formulario form, name y type.

Métodos de textarea

Veamos una lista de los métodos, que son los mismos que en un campo de texto.

blur()
Para quitar el foco de la aplicación del textarea.

focus()
Para poner el foco de la aplicación en el textarea.

select()
Selecciona el texto del textarea.

Vamos a ver un ejemplo a continuación que presenta un formulario que tiene un textarea y un botón. Al apretar el botón se cuenta el número de caracteres y se coloca en un campo de texto.

Para acceder al número de caracteres lo hacemos a partir del la propiedad value del objeto textarea. Como value contiene un string podemos acceder a la propiedad length que tienen todos los strings para averiguar su longitud.

El código de la página se puede ver aquí.

```
<html>
<head>
   <title>Ejemplo textarea</title>
<script>
function cuenta(){
   numCaracteres = document.formul.textito.value.length
   document.formul.numCaracteres.value = numCaracteres
}
</script>
</head>
<body>
<form name="formul">
<textarea name=textito cols=40 rows=3>
Este es el texto por defecto
```

```
</textarea>
<br>
<br>
Número de caracteres <input type="Text" name="numCaracteres" size="4">
<br>
<br>
<input type=button value="Cuenta caracteres" onclick="cuenta()">
</form>
</body>
</html>
```

Los eventos en Javascript

Los eventos son la manera que tenemos en Javascript de controlar las acciones de los visitantes y definir un comportamiento de la página cuando se produzcan. Cuando un usuario visita una página web e interactúa con ella se producen los eventos y con Javascript podemos definir qué queremos que ocurra cuando se produzcan.

Con javascript podemos definir qué es lo que pasa cuando se produce un evento como podría ser que un usuario pulse sobre un botón, edite un campo de texto o abandone la página.

El manejo de eventos es el caballo de batalla para hacer páginas interactivas, porque con ellos podemos responder a las acciones de los usuarios. Hasta ahora en este manual hemos podido ver muchos ejemplos de manejo de uno de los eventos de Javascript, el evento onclick, que se produce al pulsar un elemento de la página. Hasta ahora siempre hemos aplicado el evento a un botón, pero podríamos aplicarlo a otros elementos de la página.

Cómo se define un evento

Para definir las acciones que queremos realizar al producirse un evento utilizamos los manejadores de eventos. Existen muchos tipos de manejadores de eventos, para muchos tipos de acciones del usuario. El maneador de eventos se coloca en la etiqueta HTML del elemento de la página que queremos que responda a las acciones del usuario.

Por ejemplo tenemos el manejador de eventos onclick, que sirve para describir acciones que queremos que se ejecuten cuando se hace un click. Si queremos que al hacer click sobre un botón pase alguna cosa, escribimos el manejador onclick en la etiqueta <INPUT type=button> de ese botón. Algo parecido a esto.

```
<INPUT type=button value="pulsame" onclick="sentencias_javascript...">
```

Se coloca un atributo nuevo en la etiqueta que tiene el mismo nombre que el evento, en este caso onclick. El atributo se iguala a las sentencias Javascript que queremos que se ejecuten al producirse el evento.

Cada elemento de la página tiene su propia lista de eventos soportados, vamos a ver otro ejemplo de manejo de eventos, esta vez sobre un menú desplegable, en el que definimos un comportamiento cuando cambiamos el valor seleccionado.

```
<SELECT onchange="window.alert('Cambiaste la selección')">
<OPTION value="opcion1">Opcion 1
<OPTION value="opcion2">Opcion 2
</SELECT>
```

En este ejemplo cada vez que se cambia la opción muestra una caja de alerta.

Dentro de los manejadores de eventos podemos colocar tantas instrucciones como deseemos, pero siempre separadas por punto y coma. Lo habitual es colocar una sola instrucción, y si se desean colocar más de una se suele crear una función con todas las instrucciones y dentro del manejador se coloca una sola instrucción que es la llamada a la función.

Vamos a ver cómo se colocarían en un manejador varias instrucciones.

```
<input type=button value=Pulsame
  onclick="x=30; window.alert(x); window.document.bgColor = 'red'">
```

Son instrucciones muy simples como asignar a x el valor 30, hacer una ventana de alerta con el valor de x y cambiar el color del fondo a rojo.

Sin embargo, tantas instrucciones puestas en un manejador quedan un poco confusas, habría sido mejor crear una función así.

```
<script>
function ejecutaEventoOnclick(){
   x = 30
   window.alert(x)
   window.document.bgColor = 'red'
}
</script>

<FORM>
<input type=button value=Pulsame onclick="ejecutaEventoOnclick()">
</FORM>
```

Ahora utilizamos más texto para hacer lo mismo, pero seguro que a la mayoría les parece más claro este segundo ejemplo.

Jerarquía desde el objeto window

En los manejadores de eventos se tiene que especificar la jerarquía entera de objetos del navegador, empezando siempre por el objeto window. Esto es necesario porque hay algún browser antiguo que no sobreentiende el objeto window cuando se escriben sentencias Javascript vinculadas a manejadores de eventos.

Los manejadores de eventos en Javascript

Ahora vamos a ver una lista de los manejadores de eventos que hay disponibles en Javascript, ofreciendo una pequeña descripción de cada uno.

> **Nota:** Estos manejadores de eventos son los más comunes, presentes en Javascript 1.2 de Netscape Navigator. Existen otros manejadores que también son muy interesantes y veremos más adelante en capítulos de temas avanzados de eventos.
>
> La lista de manejadores de eventos contiene el nombre del manejador en negrita, su descripción y finalmente la versión de Javascript que incorporó dicho manejador.

onabort
Este evento se produce cuando un usuario detiene la carga de una imagen, ya sea porque detiene la carga de la página o porque realiza una acción que la detiene, como por ejemplo irse de la página.
Javascript 1.1

onblur
Se desata un evento onblur cuando un elemento pierde el foco de la aplicación. El foco de la aplicación es el lugar donde está situado el cursor, por ejemplo puede estar situado sobre un campo de texto, una página, un botón o cualquier otro elemento.
Javascript 1.0

onchange
Se desata este evento cuando cambia el estado de un elemento de formulario, en ocasiones no se produce hasta que el usuario retira el foco de la aplicación del elemento. Javascript 1.0
Javascript 1.0

onclick
Se produce cuando se da una pulsación o clic al botón del ratón sobre un elemento de la página, generalmente un botón o un enlace.
Javascript 1.0

ondragdrop
Se produce cuando un usuario suelta algo que había arrastrado sobre la página web.
Javascript 1.2

onerror
Se produce cuando no se puede cargar un documento o una imagen y esta queda rota.
Javascript 1.1

onfocus
El evento onfocus es lo contrario de onblur. Se produce cuando un elemento de la página o la ventana ganan el foco de la aplicación.
Javascript 1.0

onkeydown

Este evento se produce en el instante que un usuario presiona una tecla, independientemente que la suelte o no. Se produce en el momento de la pulsación.
Javascript 1.2

onkeypress
Ocurre un evento onkeypress cuando el usuario deja pulsada una tecla un tiempo determinado. Antes de este evento se produce un onkeydown en el momento que se pulsa la tecla..
Javascript 1.2

onkeyup
Se produce cuando el usuario deja de apretar una tecla. Se produce en el momento que se libera la tecla.
Javascript 1.2

onload
Este evento se desata cuando la página, o en Javascript 1.1 las imágenes, ha terminado de cargarse.
Javascript 1.0

onmousedown
Se produce el evento onmousedown cuando el uuario pulsa sobre un elemento de la página. onmousedown se produce en el momento de pulsar el botón, se suelte o no.
Javascript 1.2

onmousemove
Se produce cuando el ratón se mueve por la página.
Javascript 1.2

onmouseout
Se desata un evento onmuoseout cuando el puntero del ratón sale del área ocupada por un elemento de la página.
Javascript 1.1

onmouseover
Este evento se desata cuando el puntero del ratón entra en el área ocupada por un eolemento de la página.
Javascript 1.0

onmouseup
Este evento se produce en el momento que el usuario suelta el botón del ratón, que previamente había pulsado.
Javascript 1.2

onmove
Evento que se ejecuta cuando se mueve la ventana del navegador, o un frame.
Javascript 1.2

onresize
Evento que se produce cuando se redimensiona la ventana del navegador, o el frame, en caso de que la página los tenga.
Javascript 1.2

onreset
Este evento está asociado a los formularios y se desata en el momento que un usuario hace clic en el botón de reset de un formulario.
Javascript 1.1

onselect
Se ejecuta cuando un usuario realiza una selección de un elemento de un formulario.
Javascript 1.0

onsubmit
Ocurre cuando el visitante apreta sobre el botón de enviar el formulario. Se ejecuta antes del envío propiamente dicho.
Javascript 1.0

onunload
Al abandonar una página, ya sea porque se pulse sobre un enlace que nos lleve a otra página o porque se cierre la ventana del navegador, se ejecuta el evento onunload.
Javascript 1.0

Ejemplos de eventos en Javascript. Onabort

A lo largo de los manuales I y II de Javascript, así como del Taller, hemos mostrado muchos ejemplos de utilización de los manejadotes de eventos. Aquí veremos ejemplos sencillos que se nos ocurren para utilizar otros manejadotes que no hemos visto todavía, aunque antes podemos hacer una lista de algunos ejemplos publicados anteriormente que deberían servir de ayuda para ir captando la práctica de el manejo de eventos.

- Acceso por clave con Javascript (Evento onclick)
- Rollover con Javascript (Eventos onmuoseover y onmouseout)
- Navegador desplegable (Evento onchange)
- Calculadora sencilla (Evento onclick)
- Confirmación del envío de formulario (Evento onclick)
- Posicionarse en un select (Evento onkeypress)
- Inhibir campo de formulario (Evento onfocus)

Evento onabort

Veamos un primer ejemplo, en este caso sobre el evento onabort. Este evento se activa al cancelarse la carga de una página, ya sea porque se pulsa el botón de cancelar o porque el usuario se marcha de la página por otro enlace.

Este ejemplo contiene una imagen que tiene el evento onabort asignado para que se ejecute una función en caso de que la imagen no llegue a cargarse. La función informa al usuario de que la imagen no se ha llegado a cargar y le pregunta si desea cargarla otra vez. Si el usuario contesta que sí, entonces se pone a descargar la imagen otra vez. Si dice que no, no hace nada. La pregunta se hace con una caja confirm de Javascript.

```
<html> <head>
   <title>Evento onabort</title>

<script>
function preguntarSeguir(){
   respuesta = confirm ("Has detenido la carga de la página y hay una imagen que no estás
viendo.\n¿Deseas cargar la imagen?")
   if (respuesta)
      document.img1.src = "http://ipaginate.iespana.es/ipaginate/desarrollogrande.gif"
}
</script>

</head>
<body>
<img name=img1 src="http://ipaginate.iespana.es/ipaginate/desarrollogrande.gif" width=500
height=458 alt="Imagen que pesa 115K" border="0" onabort="preguntarSeguir()">
<br>
Pulsa el botón de parar la carga de la página y se pondrá en marcha el evento onerror

</body>
</html>
```

Este ejemplo estaría bien si siempre se detuviese la carga por pulsar el botón de cancelar, pero si lo que pasa es que el usuario ha cancelado por irse a otra página a través de un enlace, saldrá la caja de confirmación pero no ocurrirá nada independientemente de lo que se responda y el navegante se marchará irremediablemente a la nueva página.

Ejemplo del evento onblur en Javascript

Onblur se activa cuando el usuario retira el foco de la aplicación de un elemento de la página. El foco de la aplicación es el lugar donde está el cursor.

Si por ejemplo, estamos situados en un campo de texto y nos vamos de dicho campo, ya sea porque pulsamos con el ratón en otro campo del formulario o en un área vacía, ya sea porque el usuario a apretado el botón tabulador (Tab) que mueve el foco hasta el siguiente elemento de la página.

Si yo deseo que, al situarse fuera de un campo de texto, se compruebe que el valor introducido es correcto puedo utilizar onblur y llamar a una función que compruebe si el dato es correcto. Si no es correcto puedo obligar al foco de la aplicación a situarse nuevamente sobre el campo de texto y avisar al usuario para que cambie dicho valor.

Puede ser una manera interesante de asegurarnos que en un campo de texto se encuentra un valor

válido. Aunque tiene alguna pega, como veremos más adelante.

Vamos a empezar por un caso sencillo, en el que solamente deseamos comprobar un campo de texto para asegurarnos que es un número entero.

Referencia: La función que valida un entero la hemos explicado en un taller anterior de Javascript: Validar entero en campo de formulario.

```
<html>
<head>
   <title>Evento onblur</title>

<script>
function validarEntero(valor){
   //intento convertir a entero.
   //si era un entero no le afecta, si no lo era lo intenta convertir
   valor = parseInt(valor)

   //Compruebo si es un valor numérico
   if (isNaN(valor)) {
      //entonces (no es numero) devuelvo el valor cadena vacia
      return ""
   }else{
      //En caso contrario (Si era un número) devuelvo el valor
      return valor
   }
}

function compruebaValidoEntero(){
   enteroValidado = validarEntero(document.f1.numero.value)
   if (enteroValidado == ""){
      //si era la cadena vacía es que no era válido. Lo aviso
      alert ("Debe escribir un entero!")
      //selecciono el texto
      document.f1.numero.select()
      //coloco otra vez el foco
      document.f1.numero.focus()
   }else
      document.f1.numero.value = enteroValidado
}
</script>
</head>
<body>
<form name=f1>
Escriba un número entero: <input type=text name=numero size=8 value=""
onblur="compruebaValidoEntero()">
</form>

</body>
</html>
```

Al salirse del campo de texto (onblur) se ejecuta compruebaValidoEntero(), que utiliza la función validarEntero, explicada en un taller de Javascript. Si el valor devuelto por la función no es el de un entero, en este caso se recibiría una cadena vacía, muestra un mensaje en una caja alert, selecciona el texto escrito en la caja y coloca el foco de la aplicación en la caja de texto, para que el usuario coloque otro valor.

Hasta que el visitante no escriba un número entero en el campo de texto el foco de la aplicación permanecerá en el campo y continuará recibiendo mensajes de error.

Continuación del ejemplo de onblur, para validar varios campos de texto.

Hemos visto en el ejemplo del método onblur relatado anteriormente una posible técnica para comprobar los datos de un campo de formulario. Ahora vamos a ver cómo evolucionar esta técnica si tenemos más de un campo a validar, dado que se puede complicar bastante el problema.

De hecho, antes de leer nuestra solución propuesta, creo que sería un buen ejercicio a realizar por el lector la práctica de hacer ese mismo ejemplo para dos campos y trabajar un poco con la página a ver si

encontramos algún problema.

Muy probablemente nos encontraremos con un curioso bucle infinito que nos va a dar más de un quebradero de cabeza para solucionarlo.

En la práctica, el lector puede intentar validar un número entero y un código postal. Para validar un código postal necesitamos comprobar que es una cadena de texto compuesta por 5 caracteres y todos son enteros, por lo menos para los códigos en España.

Por si alguien lo quiere intentar, la función para validar un código postal sería algo parecido a esto:

```
function ValidoCP(){
   CPValido=true
   //si no tiene 5 caracteres no es válido
  if (document.f1.codigo.value.length != 5)
      CPValido=false
    else{
      for (i=0;i<5;i++){
        CActual = document.f1.codigo.value.charAt(i)
        if (validarEntero(CActual)==""){
          CPValido=false
          break;
        }
      }
    }
  return CPValido
}
```

Simplemente se encarga de comprobar que el campo de texto contiene 5 caracteres y hacer un recorrido por cada uno de los caracteres para comprobar que todos son enteros. Al principio se supone que el código postal es correcto, colocando la variable CPValido a true, y si alguna comprobación falla se cambia el estado correcto a incorrecto, pasando dicha variable a false.

Se puede probar a montar el ejemplo con dos campos... nosotros ahora vamos a dar una solución al problema bastante complicadilla, ya que incluimos instrucciones para evitar el efecto del bucle infinito. No vamos a ver el ejemplo que daría el error, lo dejamos para el que desee intentarlo por si mismo y recomendamos hacerlo porque así comprenderemos mejor el siguiente código.

```
<html>
<head>
   <title>Evento onblur</title>

<script>
avisado=false
function validarEntero(valor){
     //intento convertir a entero.
     //si era un entero no le afecta, si no lo era lo intenta convertir
      valor = parseInt(valor)

     //Compruebo si es un valor numérico
     if (isNaN(valor)) {
       //entonces (no es numero) devuelvo el valor cadena vacia
       return ""
     }else{
       //En caso contrario (Si era un número) devuelvo el valor
       return valor
     }
}

function compruebaValidoEntero(){
  enteroValidado = validarEntero(document.f1.numero.value)
  if (enteroValidado == ""){
     //si era la cadena vacía es que no era válido. Lo aviso
     if (!avisado){
       alert ("Debe escribir un entero!")
       //selecciono el texto
       document.f1.numero.select()
       //coloco otra vez el foco
       document.f1.numero.focus()
       avisado=true
       setTimeout('avisado=false',50)
     }
```

```
  }else
    document.f1.numero.value = enteroValidado
}

function compruebaValidoCP(){
  CPValido=true
  //si no tiene 5 caracteres no es válido
  if (document.f1.codigo.value.length != 5)
    CPValido=false
  else{
    for (i=0;i<5;i++){
      CActual = document.f1.codigo.value.charAt(i)
      if (validarEntero(CActual)==""){
        CPValido=false
        break;
      }
    }
  }
  if (!CPValido){
    if (!avisado){
      //si no es valido, Lo aviso
      alert ("Debe escribir un código postal válido")
      //selecciono el texto
      document.f1.codigo.select()
      //coloco otra vez el foco
      //document.f1.codigo.focus()
      avisado=true
      setTimeout('avisado=false',50)
    }
  }
}
</script>

</head>
<body>

<form name=f1>
Escriba un número entero: <input type=text name=numero size=8 value=""
onblur="compruebaValidoEntero()">
<br>
Escriba un código postal: <input type=text name=codigo size=8 value=""
onblur="compruebaValidoCP()"> *espera una cadena con 5 carácteres numéricos

</form>

</body>
</html>
```

Este ejemplo sigue la guía del primer ejemplo de onblur de este artículo, incluyendo un nuevo campo a validar.

Para solucionar el tema del bucle infinito, que habréis podido investigar por vosotros mismos y en el que se mostraban una caja de alerta tras otra indefinidamente, hemos utilizado una variable llamada avisado que contiene un true si ya se ha avisado de que el campo estaba mal y un false si no se ha avisado todavía.

Cuando se va a mostrar la caja de alerta se comprueba si se ha avisado o no al usuario. Si ya se avisó no se hace nada, evitando que se muestren más cajas de alerta. Si no se había avisado todavía se muestra la caja de alerta y se coloca el foco en el campo que era incorrecto.

Para restituir la variable avisado a false, de modo que la próxima vez que se escriba mal el valor se muestre el mensaje correspondiente, se utiliza el método setTimeout, que ejecuta la instrucción con un retardo, en este caso de 50 milisegundos. Lo suficiente para que no se meta en un bucle infinito.

Nota: Después de todos los parches que hemos tenido que colocar para que este evento se comporte correctamente para cumplir el cometido deseado, es posible que no merezca la pena utilizarlo para este cometido. Podemos hacer uso del evento onchange, o comprobar todos los campos de una sola vez cuando el usuario ha decidido enviarlo.

Elementos de formulario select asociados

Vamos a conocer uno de los trucos más solicitados de Javascript, que tiene mucha relación con el tema de formularios y donde se utiliza el evento onchange de Javascript. Es un ejemplo sobre cómo realizar una página con un par de selects donde, según el valor escogido en uno de ellos, cambien las opciones posibles del otro select.

Lo mejor para ver lo que vamos a hacer es ver una página web donde se muestra en funcionamiento el script. Para ver su funcionamiento debemos cambiar la selección del primer select y comprobaremos como las opciones del segundo select cambian automáticamente.

El ejemplo que hemos ilustrado utiliza provincias y países. Al escoger en el primer select un país, en el segundo debe mostrarnos las provincias de ese país para que escojamos una provincia, pero sólo una que tenga que esté en el país seleccionado en primer término.

Conocer el objeto select y los option

Es importante conocer los objetos de formulario select y los option. Los select corresponden con las cajas de selección desplegables y los option con cada una de las opciones de la caja desplegable. Podemos ver un artículo que habla de ello.

En concreto nos interesa hacer varias cosas que tienen que ver con extraer el valor de un select en cualquier momento, fijar su número de opciones y, para cada opción, colocar su valor y su texto asociado. Todo esto aprenderemos a hacerlo en este ejemplo.

> **Referencia:** Para conocer el trabajo con formularios y la jerarquía de objetos javascript (Todo eso es la base del trabajo con los elementos de las páginas en Javascript) debemos haber leer el manual de Javascript II.

Modo de llevar a cabo el problema

Para empezar, vamos a utilizar un formulario con dos selects, uno para el país y otro para la provincia.

```
<form name="f1">
<select name=pais onchange="cambia_provincia()">
<option value="0" selected>Seleccione...
<option value="1">España
<option value="2">Argentina
<option value="3">Colombia
<option value="4">Francia
</select>

<select name=provincia>
<option value="-">-
</select>
</form>
```

Nos fijamos en el select asociado al país de este formulario que, cuando se cambia la opción de país, se debe llamar a la función cambia_provincia(). Veremos más adelante esta función, ahora es importante fijarse que está asociada al evento onchange que se activa cuando cambia la opción en el select.

Todo lo demás será código Javascript. Empezamos definiendo un montón de arrays con las provincias de cada país. En este caso tenemos sólo 4 países, entonces necesitaré 4 arrays. En cada array tengo la lista de provincias de cada país, colocada en cada uno de los elementos del array. Además, dejaré una primera casilla con un valor "-" que indica que no se ha seleccionado ninguna provincia.

```
var provincias_1=new Array("-","Andalucía","Asturias","Baleares","Canarias","Castilla y León","Castilla-La Mancha","...")
var provincias_2=new Array("-","Salta","San Juan","San Luis","La Rioja","La Pampa","...")
var provincias_3=new Array("-","Cali","Santamarta","Medellin","Cartagena","...")
var provincias_4=new Array("-","Aisne","Creuse","Dordogne","Essonne","Gironde ","...")
```

Hay que fijarse que los índices del array de cada país se corresponden con los del select del país. Por ejemplo, la opción España, tiene el valor asociado 1 y el array con las provincias de España se llama provincias_1.

El script se completa con una función que realiza la carga de las provincias en el segundo select. El mecanismo realiza básicamente estas acciones:

- Detecto el país que se ha seleccionado
- Si el valor del país no es 0 (el valor 0 es cuando no se ha seleccionado país)
    - Tomo el array de provincias adecuado, utilizando el índice del país.
    - Marco el número de opciones que debe tener el select de provincias
    - Para cada opcion del select, coloco su valor y texto asociado, que se hace corresponder con lo indicado en el array de provincias.
- SI NO (El valor de país es 0, no se ha seleccionado país)
    - Coloco en el select de provincia un único option con el valor "-", que significaba que no había provincia.
- Coloco la opción primera del select de provincia como la seleccionada.

La función tiene el siguiente código. Está comentado para que se pueda entender mejor.

```
function cambia_provincia(){
   //tomo el valor del select del pais elegido
   var pais
   pais = document.f1.pais[document.f1.pais.selectedIndex].value
   //miro a ver si el pais está definido
   if (pais != 0) {
      //si estaba definido, entonces coloco las opciones de la provincia correspondiente.
      //selecciono el array de provincia adecuado
      mis_provincias=eval("provincias_" + pais)
      //calculo el numero de provincias
      num_provincias = mis_provincias.length
      //marco el número de provincias en el select
      document.f1.provincia.length = num_provincias
      //para cada provincia del array, la introduzco en el select
      for(i=0;i<num_provincias;i++){
         document.f1.provincia.options[i].value=mis_provincias[i]
         document.f1.provincia.options[i].text=mis_provincias[i]
      }
   }else{
      //si no había provincia seleccionada, elimino las provincias del select
      document.f1.provincia.length = 1
      //coloco un guión en la única opción que he dejado
      document.f1.provincia.options[0].value = "-"
      document.f1.provincia.options[0].text = "-"
   }
   //marco como seleccionada la opción primera de provincia
   document.f1.provincia.options[0].selected = true
}
```

Evento onunload de Javascript

Veamos un ejemplo del evento onunload, que, recordamos, se activa cuando el usuario ha abandona la página web. Por tanto, onunload sirve para ejecutar una acción cuando el usuario se marcha de la página, ya sea porque pulsa un enlace que le lleva fuera de la página o porque cierra la ventana del navegador.

El ejemplo que deseamos mostrar sirve para abrir una página web en otra ventana cuando el usuario abandona la página. De este modo actúan muchos de los molestos popups de las páginas web, abriéndose justo cuando abandonamos el sitio que estábamos visitando.

```
<html>
<head>
   <title>Abre al salir</title>
   <script>
   function abreventana(){
      window.open("http://www.google.es","venta","")
   }
   </script>
</head>

<body onunload="abreventana()">


<a href="http://www.desarrolloweb.com">DW!!</a>
</body>
```

```
</html>
```

El ejemplo es tan sencillo que casi sobran las explicaciones. Simplemente creamos una función que abre una ventana secundaria y la asociamos con el evento onunload, que se coloca en la etiqueta <body>.

Capitulo 13
Tutorial de Visual Basic Script

Manual del lenguaje de scripting de Microsoft para páginas web con el que podrás aprender a realizar efectos para el Internet Explorer.
Explora las características del HTML Dinámico con el lenguaje del navegador más habitual.

Introducción a Visual Basic Script

El lenguaje para describir páginas, HTML, queda limitado a la hora de definir cualquier tipo de interactividad. Una vez que hemos explotado su potencia, estamos en necesidad de aprender algún lenguaje nuevo para hacer pequeños efectos o interactividades.

Scripts
Son los pequeños programitas que, incrustados en las páginas, nos permiten definir aquellos efectos o interactividades.

Visual Basic Script
En este manual nos vamos a ocupar de Visual Basic Script, un lenguaje compatible con Internet Explorer y otros sistemas Microsoft, por lo que en principio es una ventaja para programadores experimentados en estos sitemas.

Otros lenguajes
Existen dos tipos principales de lenguajes de scripting, y multitud de utilidades distintas, pero cabria destacar el lenguaje **Javascript** por ser parecido en utilidad a VBScript pero compatible con los dos navegadores más habituales.

Cómo poner scripts

Para poner un script en una página web utilizamos la etiqueta de HTML **<SCRIPT>**. Todo lo que pongamos entre esa etigueta y la de cierre, **</SCRITP>**, tiene que ser código del lenguaje de scripting que estemos utilizando.
También debemos indicar el lenguaje con el que estamos programando. En nuestro caso pondremos:

```
<script language="VBScript" >
      ---Aqúi pondremos nuestros scripts---
</script>
```

Parece una tonteria, pero fijaros que la palabra **language** en inglés se escribe con dos "G": **language**. Si os equivocais en este punto, cosa bastante probable si escribis rápido y despistados, no funcionarán vuestros scripts pues el navegador pensará que están escritos en JavaScript.

Primer Script sencillo

Para terminar este capítulo vamos a ver un primer ejemplo de script en una página web. El objetivo de este script es mostrar la fecha de la última modificación del documento

```
<html>
<head>
  <title> La &uacute;ltima modificaci&oacute;n del documento</title>
</head>

<body>
<h1>Script de la &uacute;ltima modificaci&oacute;n de un documento</h1>
<script language="VBScript">
  document.write "Este documento fue actualizado por &uacute;ltima vez en: "
  document.write document.lastmodified
</script>
</body>
</html>
```

La sentencia **document.write** es un procedimiento que escribe en la página web el texto que recibe por parámetro, el texto que esta después de la sentencia.
La variable **document.lastmodified** almacena la fecha y la hora de la última actualización.

Este script dará como resultado que el documento te informe de su última actualización, de una manera parecida a esta:

Este documento fue actualizado por última vez en: 04/19/2002 03:32:16

Primeros pasos con el lenguaje

Los lenguajes de scripting tienen una serie de características comunes, estas suelen hacer la programación más fácil para personas inexpertas, pero a la larga pueden convertirse en una fuente de errores. Veamos cuáles son estas carcaterísticas, en concreto para VBScript.

Mayúsculas y minúsculas
En VBScript no importa si utilizamos mayúsculas o minúsculas a la hora de escribir nuestro código.

Variables
Las variables son espacios donde se almacenan los datos que utilizan los programas o scripts.
No se declaran: en VBScript las variables no se han de declarar, es decir, cuando necesitemos una variable, simplemente la utilizamos y ya está. Aín así, si deseamos declarar una variable utilizamos la palabra **DIM**
No hay tipos: las variables no están tipadas, esto quiere decir que podemos guardar en ellas igualmente números que letras que otras cosas.

Saltos de línea
Son importantes los saltos de línea. Expresan el final de una instrucción y el principio de la siguiente. No se pueden poner dos instrucciones en una misma línea.

Comentarios
En VBScript los comentarios se colocan con una comilla simple **'**. Esto sirve para que todo lo que se encuentre en esa línea después de la comilla simple sea ignorado por el explorador.

Ejemplo de todo esto

Vamos a ver a continuación un sencillo script que sirve de ejemplo para todo lo dicho anteriormente.
El ejemplo siguiente despliega unas ventanitas con mensajes (sentencia **msgbox**) siendo los mensajes el contenido de la variable **pepe**. Durante el ejemplo se cambia el valor de la variable y se vuelve a mostrar.
El ejemplo demuestra que no importan las mayúsculas y minúsculas y que es indiferente el tipo del contenido de la variable, texto o números.

```
<HTML>
<HEAD>
   <TITLE>Ejemplo2 Comentario, caja alert y variables</TITLE>
</HEAD>
<BODY>
<script language=VBScript>
      'Esto es un comentario
      PEPE="HOLA"
      msgbox(pepe)
      pepe=3456
      'NO importan las mayusculas-minusculas
      msgbox(PEPE)
</script>
</BODY>
</HTML>
```

Distintas formas de ejecutar scripts

Ahora que ya sabes cómo incluir scripts en tus páginas y unos cuantos fundamentos del lenguaje, vamos a ver los dos casos en los que Internet Explorer puede ejecutar tus scripts, de paso que le damos un primer vistazo a el concepto de evento.

Las formas de ejecución de VBScript son las siguientes:

- Scripts que se ejecutan mientras que el navegador abre la página.
- Scripts que se ejecutan como respuesta a la acción de un usuario.

El primero de los casos se utiliza cuando quieres hacer algo cuando el navegador carga la página. Por ejemplo, podrías mostrar un mensaje de bienvenida que aparezca cuando el usuario entra en tu página,

o que el navegador te informe de la última actualización del documento (Tal como se vió en el capitulo 1).

El segundo caso es útil cuando deseas realizar acciones como respuesta a eventos del usuario.

Los **eventos** son acciones que ocurren cuando un usuario hace alguna cosa sobre la página web, es decir, un evento podría ser que el usuario escriba algo en una caja de texto, o que se coloque con el ratón encima de un enlace, y así un montón de cosas. Casi cualquier cosa que puede realizar el usuario dentro de la página tiene un evento relacionado.

Utilizando los eventos podemos preparar algún efecto que sea solo visible cuando el usuario realice alguna acción dentro de la página web.

Ejemplo de todo esto

Veamos ahora un ejemplo para acabar de comprender las dos formas de ejecución de los scripts

Vamos hacer que el navegador nos diga su número de versión y otros datos en un cuadro de diálogo. Lo vamos a hacer de las dos maneras, según se carga la página y cuando el usuario aprete un botón.

Ejemplo de ejecución al cargar la página
Comencemos por la ejecución de scripts cuando el usuario carga la página. Esta es la forma más sencilla, y realmente ya conoces varios ejemplos de esto que viste en los anteriores capítulos.

```
<html>
<head>
  <title>Escript de ejecución directa</title>
</head>
<body>
   Según se carga la página vamos a ver
   la versión del navegado en una caja de diálogo.
   <script language=vbscript>
      msgbox(navigator.appVersion)
   </script>
</body>
</html>
```

Este ejemplo no tiene ningún misterio, pues es muy parecido a los ejemplos que hemos realizado. la única novedad es la variable **navigator.appVersion**. Esta almacena lo que queremos que se vea en la caja de diálogo: la versión del navegador.

Ejemplo de ejecución como respuesta a la acción del usuario
Ahora veamos lo que hay que hacer cuando deseamos que esta caja de diálogo no aparezca hasta que el usuario pulse en un botón.

```
<html>
<head>
   <title>Script de ejecución como respuesta a un evento</title>
</head>
<body>
 Pulse el botón para ver la versión del navegador
 <input type=button value=Pulsame onclick="msgbox(navigator.appVersion)"
language="vbscript">
</body>
</html>
```

Este ejemplo tiene cosas nuevas que habría que destacar:

1. Se crea un botón con la etiqueta <INPUT>
2. Se le añade el atributo **onclick**. Este sirve para indicar (en lenguaje de script) las acciones que queremos realizar como respuesta al evento "click sobre el botón".
3. Se le añade el atributo **language** para especificar el lenguaje en el que está escrito el código script asociado al evento.

Ahora tenemos un botón que, cuando se pulse, ejecutará el código que despliega una caja de diálogo con la versión del navegador.

Declarar variables

Antes de pasar a temas más interesantes queda explicar la forma de declarar variables en VBScript. Hace poco se dijo que no era necesario declarar estas variables, pero puede ser una buena costumbre hacerlo y nos puede evitar errores.

Una variable se declara utilizando la palabra **DIM**, veamos cómo:

```
<script language="vbscript">
  dim mi_nueva_variable
  'Ahora ya existe la variable
  'Seguidamente voy a hacer uso de ella
  mi_nueva_variable = "Valor de la variable"
</script>
```

Como se ha de recordar no importa que tipo de información va a contener la variable, siempre se declaran igual.

Option explicit
Se puede utilizar la clausula **Option explicit** para forzar la declaración de variables en nuestros scripts. Si deseas evitar la posible fuente de errores que supone la libertad de no declarar las variables puedes utilizar esta clausula y hará que tus scripts respondan con mensajes de error si utilizas una variable que no has declarado previamente. Veamos su uso con un ejemplo:

```
<HTML>
<HEAD>
 <TITLE>Option explicit</TITLE>
</HEAD>
<BODY>
 <script language=vbscript>
   option explicit

   dim Pepe
   pepe = 3
   tomas = 87
 </script>
</BODY>
</HTML>
```

Este script responderá con un mensaje de error cuando se ejecute, pues la variable tomas no se ha declarado antes de su uso.

Tipos de datos

Visual Basic Script posee varios tipos de datos pero en la práctica sólo posee un tipo de variable, que va cambiando de un estado a otro según la información que introducimos dentro. Este tipo *principal* de datos es el tipo **Variant**, en él podemos introducir varios *subtipos* de datos con total libertad.

Para cambiar el subtipo de un variant, sólo tenemos que introducir un dato en la variable. La variable variant cambia automáticamente de un subtipo a otro, sin que tengamos que hacer ninguna operación adicional. Los distintos subtipos de datos que tenemos son los siguientes:

Booleano	Es un tipo de datos que contiene un si o un no. se corresponden: TRUE equivale a (-1) FALSE equivale a (0)
Byte	Numérico, entero sin signo hasta 65.000
Currency	Tipo de **moneda**, se utiliza para manipular de manera exacta valores monetarios, y en general cualquier cálculo que requiera una precisión de hasta 15 dígitos decimales
Fecha	Es un tipo de 64 bits de tamaño que almacena fechas. Se utiliza el formato americano: mes, día, hora.
Double	Coma flotante con doble precisión (64 bits)

Entero	Número entero, con signo. Desde -32.768 hasta 32.767
Entero largo	Este tipo es un valor entero con signo de doble precisión. Como los nuevos ordenadores trabajan con palabras de 32 bits, y no menos, se recomienda usar este tipo antes de el tipo entero normal.
Objeto	El subtipo de objeto es una referencia de puntero de 32 bits a una instancia de de objeto de automatización OLE. Los controles Active-X y java. Utilizan esta sintaxis: Set miobjeto = new oleObjeto
Single	Coma flotante de precisión simple
Cadena	Conjunto continuo de valores de caracteres, de longitud variable.

Como saber de qué subtipo es una variable

Para averigüar el subtipo de una variable podemos utilizar la función VarType, de esta manera:

```
v1 = 3

document.write VarType (v1)
```

En este script declaramos una variable y le metemos un número y a continuación imprimimos en la página el valor que devuelve la función VarType.

Al ejecutar este script podremos ver un "2" escrito en la página.

Según el tipo de datos que halla en la variable, VarType devolverá un valor distinto, como indica esta tabla:

Constant	Value	Description
vbEmpty	0	Empty (uninitialized)
vbNull	1	Null (no valid data)
vbInteger	2	Integer
vbLong	3	Long integer
vbSingle	4	Single-precision floating-point number
vbDouble	5	Double-precision floating-point number
vbCurrency	6	Currency
vbDate	7	Date
vbString	8	String
vbObject	9	Automation object
vbError	10	Error
vbBoolean	11	Boolean
vbVariant	12	Variant (used only with arrays of Variants)
vbDataObject	13	A data-access object
vbByte	17	Byte
vbArray	8192	Array

Operadores I - Aritméticos

Visual Basic Script, como cualquier lenguaje de programación, tiene un conjunto de operadores, divididos en varias secciones:

Operadores Aritméticos

Que soportan las operaciones matemáticas más sencillas.

+	Suma
-	Resta
*	Multiplicacion

/	División en coma flotante. Es la división normal. Devuelve un numero real si es el resultado
\	División de enteros Devuelve un numero entero, resultado de la división.
^	Potencia
Mod	Resto de la división

Veamos a continuación un ejemplo de script que realiza acciones con estos operadores:

```
dim v1

dim v2

v1 = 34

v2 = 43

suma = v1 + v2

resta = v1- v2

potencia = v1 ^ v2

divisionEnteros = v1 \ v2

msgbox(divisionEnteros)

DivisionReal = v1 /v2

msgbox(divisionReal)
```

La función msgbox sirve para mostrar un valor en una ventanita de alerta típica de Windows.

Operadores II - Comparación

Para realizar comparaciones, Visual Basic Script posee los siguientes operadores:

= <>	Igual y distinto
> <	Mayor que y menor que
>= <=	Mayor o igual que y menor o igual que

Veamos a continuación un ejemplo de script que realiza operaciones de comparación, aunque antes que verlo deberíamos puntualizar que los operadores de comparación se suelen utilizar dentro de una estructura condicional, que evalúa una expersión con estos comparadores y realiza acciones dependiendo del resultado de esas comparaciones. Por este motivo hemos incluido en el script la estructura condicional IF que veremos con profundidad más adelante.

```
precio = 20000

dineroActual = 3500

if (dineroActual  = precio) then

        msgbox  ("Lo tienes justo")

end if

if (dineroActual < precio) then

        msgbox ("te falta dinero")

end if
```

Operadores III lógicos y cadenas

Operadores lógicos

Los operadores lógicos se utilizan sobre expresiones booleanas y nos devuelven un valor booleano (verdadero o falso) resultado de esa operación. Un matiz sería que no es necesario que los operandos relacionados en el cálculo sean variables booleanas, pudienso ser de cualquier tipo.

Como operadores lógicos en Visual Basic Script tenemos:

AND	Y lógico
OR	O lógico
Xor	Xor
Not	NO lógico

Operadores de cadenas

Como operador de cadenas de caracteres en Visual Basic Script tenemos un único ejemplo: la concatenación. El operador para concatenar cadenas es el **&**. Veamos un ejemplo de utilización de este operador:

```
cadena1 = "Hola "

cadena2 = "Pepe"

concatenar = cadena1 & cadena2

msgbox  (concatenar)
```

Estructuras de control

Las estructuras de control nos permiten realizar acciones típicas en nuestros scripts como lo pueden ser los bucles o la toma de decisiones.

VBScript tiene las estructuras de control típicas de los lenguajes de programación. Vamos a ver la sintaxis y la manera de trabajar de estas estructuras detenidamente:

Condicionales

- **IF**, condicional que decide entre si/no
- **CASE**, otro condicional con varias posibilidades

Bucles

- **FOR**, repetición un determinado número de veces
- **FOR EACH**, repetición para un conjunto de elementos
- **WHILE...WEND**, repetición mientras ocurra alguna cosa
- **DO...LOOP**, repetición un determinado número de veces

Estructura IF

La estrutura de control IF permite decidr entre dos opciones resultantes de la evaluación de una sentencia. Si la evaluación es positiva hace una cosa, también podemos especificar accioens para realizar en caso de que la evaluación sea negativa. Veamos cómo funciona en VBscript.

```
IF (expresion) then
    Sentencias
    ....
END IF
```

Vemos que en primer lugar tenemos la sentencia IF, luego una expresión, que puede o no ir entre paréntesis, y más tarde la parabra THEN. Vemos que luego hay un salto de línea antes de colocar las sentencias asociadas a la evaluación positiva de la sentencia. En VBScript las líneas si que importan.

Después de poner las sentencias del asociadas a la evaluación positiva colocamos un END IF, para acabar la estructura del IF.

Enunciado ELSE

Opcionalmente se puede colocar una serie de sentencias asociadas a la evaluación negativa de la expresión. Estas sentencias se deben colocar después de la orden ELSE y antes del END IF.

```
IF (expresion) then
    Sentencias
    ....
ELSE
    Sentencias
    ....
END IF
```

Enunciado ELSEIF

En Visual Basic Script existe la posibilidad de utilizar un enunciado especial en el lugar donde utilizaríamos un ELSE. Sirve para encadenar sentencias IF de modo que en un resultado negativo de un IF se pueda evaluar otra expresión, que tendría a su vez otros enunciados THEN y probablemente ELSE u otro ELSEIF. Se vería en un ejemplo com más facilidad:

```
IF (expresion1) then
    Sentencias1
    ....
ELSEIF (expresion2) then
    Sentencias2
    ....
ELSEIF (expresion3) then
    Sentencias3
    ....
ELSE
    Sentencias4
    ....
END IF
```

Se evalúa la primera expresión, en caso positivo se ejecutan las sentencias1, en caso negativo se evalua la expresion 2. Si la expresión 2 es positiva se ejecutan las sentencias 2 en caso negativo evaluamos la expresión 3 con el siguiente ELSEIF. Todo acaba en un ELSE en este ejemplo, pero el ELSE final no es obligatorio.

Estructura CASE

Con la estructurad de control CASE podemos evaluar una variable y realizar acciones dependiendo del valor de esta. La diferencia con el IF consiste en que el número de posibilidades de la evaluación de esta variable no tiene por que ser si o no, pudiendo hacer cosas para un número indeterminado de valores.

La sintaxis es la siguiente:

```
SELECT CASE (variable)
   CASE (valor1):
            (acción para caso valor1)
   CASE (valor3):
            (acción para caso valor2)
   CASE (valor3):
            (acción para caso valor3)
   CASE ELSE:
            (accion en caso de que no se cumpla ningun anterior caso)
END SELECT
```

Funciona asi, primero se evalua la variable, si esa variable tiene como valor el valor1 realizamos las acciones asociadas al valor1. Si tiene el valor2, ejecutamos las acciones relacionadas con este valor3. Así con cuantos valores deseemos. Por último tenemos un ELSE para realizar acciones en caso de que no hubiesen sido ninguno de los valores anteriores. Este ELSE es opcional.

Veamos con un ejemplo esta sentencia muy sencillito. Lo primero que hace es solicitar un número y luego informa del día de la semana con el que corresponde. Si el número no es del uno al siete informa de ello también.

```
dim dia
dia = inputbox ("dime un dia de la semana")
SELECT CASE dia
CASE 1:
  msgbox("El dia es LUNES")
CASE 2:
  msgbox("El dia es MARTES")
CASE 3:
  msgbox("El dia es MIERCOLES")
CASE 4:
  msgbox("El dia es JUEVES")
CASE 5:
  msgbox("El dia es VIERNES")
CASE 6:
  msgbox("El dia es SABADO")
CASE 7:
  msgbox("El dia es DOMINGO")
CASE ELSE:
  msgbox("Tiene que ser un dia de la semana en número, del 1 al 7")
END SELECT
```

Bucle FOR

La sentencia FOR se utiliza para los bucles, cuando sabemos el número de veces que debemos ejecutar el bucle. Veamos su sencilla sintaxis:

```
FOR (inicializacion) TO (termino del bucle) STEP (paso)
      sentencias
      ........
NEXT
```

La sentecia realiza una repetición desde la *inicialización* hasta el *término del bucle*. Para llevar la cuenta se utiliza una variable, ya veremos en el ejemplo cómo se utiliza esta variable. Con cada ejecución del bucle se ejecutan unas sentencias. NEXT sirve para delimitar el final del bucle, cuando se encuentra con el NEXT se vuelve otra vez al principio del FOR, así hasta realizar el número de ejecuciones determinado.

Existe un valor que sirve para indicar lo grandes que se desean realizar los saltos entre ejecución y ejecución, es el valor STEP. Un STEP 2 determinaría que entre ejecución y ejecución la variable se ha de incremetar en 2 unidades. En el caso de no indicar nada se realizan pasos de 1 en 1. También podemos realizar pasos en valores negativos.

Un ejemplo de estos datos sería el siguiente:

```
for i=0 to 6 step 2
     msgbox(i)
next
```

Este ejemplo presentaría un mensaje con un numerito de la variable i, utilizada para llevar la cuenta de las ejecuciones del bucle.

Bucle FOR EACH

La estructura de control FOR EACH sirve para moverse por los elementos de una estructra de datos, como podría ser un vector, y realizar acciones para cada una de los elementos.

Veamos con un ejemplo esta estructura de control: En el ejemplo primero creamos un vector y rellenamos con números cada una de sus casillas, con un bucle FOR normalito. Más tarde utilizamos el bucle FOR EACH para acceder a cada una de las posiciones de este vector de números y escribir en la página cada una de estos números.

```
dim tor(20)
```

```
for i=0 to 20
   tor(i) = i
next
for each i in tor
   document.write (tor(i))
next
```

Fijemonos en el segundo bucle, se indica que para cada i (i es el índice con el que podemos movernos en el bucle y en la estructura) dentro de tor (que es la estructura, en este caso un vector) haga un document.write(tor(i)). Con tor(i) accedemos a la casilla actual y document.write() sirve para escribir algo en la página web. Conbinadas lo que se escribe es lo que hay en la posición actual del vector.

Bucle WHILE WEND

El bucle WHILE...WEND sirve para realizar un tipo de bucle mu utilizado en programación que es el bucle Mientras, que se ejecuta mientras que se cumpla una condición. A diferencia del bucle FOR, éste se utiliza cuando no conocemos el número de iteraciones que tenemos que realizar.

El bucle funciona de la siguiente manera. Cuando se va a ejecutar, evalúa una expresión y compreba que esta da resultados positivos. Si es así, ejecuta el cuerpo del bucle (las sentencias que siguen hasta el WEND), en caso contrario se sale. Pdemos ver la sintaxis a continuación.

```
WHILE (condicion)
        sentencias
        ....
WEND
```

Ahora vamos a ver un ejemplito sobre este bucle, que realiza una cuenta número a número hasta llegar al 13. En cada iteración del bucle muestra en una ventanita el número actual y ofrece la posibilidad de cambiarlo, ya que la ventanita es una ventana Input, que ofrece la oportunidad de cambiar el valor y devuelve ese valor, cambiado o no. Como decíamos, si dejamos el ejemplo sin tocar nada, cuenta hasta 13, pero si introducimos un número en el inputbox continúa la cuenta por el número introducido. Si el número introducido es mayor que 13 también se sale del bucle.

```
option explicit
dim a
a = 0
WHILE (a < 13)
  a = a + 1
  a = inputbox("Dame un valor entero, please","Petición de número",a,200,100)
WEND
```

Bucle DO LOOP

El bucle DO...LOOP es muy versatil. Con el se pueden crear gran variedad de bucles distintos, bucles que comprueben una condición antes de ejecutar el bucle una vez, después de la primera ejecución y con conbinaciones con mientras (WHILE) que se cumple una condición o hasta (UNTIL) que esa condicion se cumpla. la sintaxis de esta estructura es la siguiente:

```
DO [WHILE | UNTIL (condicion)]
    Sentencias
    .....
LOOP [WHILE | UNTIL (condicion)]
```

Vamos a tratar de explicar esta sentencia de manera pausada para que sea más fácil de entender. Lo que siempre tendremos en estos bucles es el DO y el LOOP, entre estos dos colocaremos las sentencias que queremos ejecutar en cada iteración del bucle. Los bucles tienen que evaluar entre cada iteración si se siguen ejecutando o no, para ello evaluan una condición. Lo versatil de este bucle es que la condición se puede expresar de muchas maneras distintas.

Condición expresada al lado del DO: en este caso la condición se evalua antes de empezar a ejecutarse el bucle.
Condición expresada al lado del LOOP: en este caso la condición se evalua después de ejecutarse el bucle. Tiene como diferencia principal frenta al otro método que en este caso el bucle se ejecutará por lo menos una vez.

Además de poder expresar la condición en estos dos sitios también se puede construir la condición con un enunciado mientras (WHILE) o un enunciado hasta (UNTIL). Las diferencias semanticas de estas dos posibilidades se trasladan también a su manera de funcionar.

Vamos a ver un par de ejemplos de este bucle para comprender su funcionamiento. El ejemplo pide cunstantemente el nombre del autor de la página y no para hasta que el nombre sea "migue". También tiene el usuario la posibilidad de escribir "out", en ese caso, comprobado con un enunciado IF, se sale del bucle rompiéndolo con la sentencia EXIT DO, utilizada para romper bucles.

```
Dim entrada
entrada = ""
DO WHILE (entrada <> "migue")
  entrada = inputbox ("Dime el nombre del autor","seguridad","migue",2,3)
  if (entrada = "out") then
    msgbox "salgo por la puerta de atras"
     exit do
  end if
LOOP
```

El siguente ejemplo realiza una cuenta y entre cuenta y cuenta se muestra el valor de la cuenta actual en una ventanita donde sale un botón de Reintentar y otro de Cancelar. Si se pulsa reintentar se sigue ejecutando el bucle y si se pulsa Cancelar se sale por la puerta de atrás, de manera similar a como se salía en el ejemplo anterior, con EXIT DO.

```
option explicit
dim cont
dim respuesta
cont = 0
DO
  cont = cont +1
  respuesta = msgbox (cont,69,"Variable del bucle, con valor 6 se sale")
  if (respuesta = 2) then
     msgbox "Cuenta Cancelada",16,"Cancelaste!"
     exit do
  end if
LOOP UNTIL (cont = 6)
```

El ejemplo es un poco rarito, pero servirá para comprender estos bucles.

Arrays en Visual Basic Script

Los Arrays o matrices son unas estructuras de datos muy utilizadas en cualquier lenguaje. Se tratan de variables, pero que están preparadas para guardar una cantidad mayor de elementos. Es como una variable que tiene varios compartimentos para guardar la información y a cada uno de esos compartimentos hay que acceder con un índice.

Antes de utilizar un array debemos declararlo de manera obligatoria, para ello utilizamos la palabra clave DIM, de este modo.

```
dim miArray(20)
```

Después de la palabra DIM debemos indicar el nombre del array y acontinuación, entre paréntesis, se coloca el número de posición máxima del array, en este caso 20.

Los arrays en ASP comienzan desde la posición 0, es decir, el primer elemento de un array está en la posición 0. Por tanto, si el array ha sido definido con 20 casillas, como en el ejemplo, tendrá 21 elementos, primera posición será la 0 y la última posición sería la 20.

Para asignar un valor a un array se realiza igual que una variable, pero accediendo con el índice de la posición que queremos escribir.

```
miArray(0) = 234
```

Para utilizar el contenido de un array debemos hacerlo indiando el índice al que se desea acceder. Por ejemplo, si quisiesemos imprimir en la página la primera posición de nuestro Array lo haríamos de esta manera.

```
document.write(miArray(0))
```

Ahora vamos a ver un ejemplo sobre cómo utilizar los arrays, donde vamos a realizar dos recorridos, uno para escribir en él y el otro para leer la información y escribirla en la página.

```
dim matriz (10)
for i=0 to 10
  matriz(i)=100 * i
next

for i=0 to 10
  document.writeln("Posicion " & i & ": " & matriz(i) & "<br>")
next
```

Este ejemplo escribiría en la página las posiciones del array, que contienen variables numéricas que corresponden de multiplicar su índice por 100.

Se pueden construir matrices multidimensionales, es decir, que nos permitan crear matrices de varias coordenadas. Para trabajar con ellos se utiliza una coma que separa los dos índices. Por ejemplo podemos definir una matriz de 8x8 de esta manera.

```
dim miArray2Dimensiones (7,7)
```

Como el array es de 8 casillas, utilizamos un 7 y sus posiciones serán las 8 que van desde el 0 al 7. Para escribir y leer del Array podemos utilizar la coma de manera similara a como se declara. Por ejemplo, para meter datos en la posición 0,2 haríamos lo siguiente:

```
miArray2Dimensiones (0,2) = "texto posicion 0,2"
```

Procedimientos y funciones

Los procedimientos o funciones son muy interesantes y útilies en la programación. Nos sirven para realizar una tarea concreta que probablemente se vaya a ejecutar varias veces a lo largo de la vida de la página. Esta tarea se especifica en un bloque de código de manera independiente y cuando se desean realizar las acciones del procedimiento se llama al porcedimiento o función. Una vez realizadas las acciones pertinentes se devuelve el flujo del programa al lugar desde donde se imbocó ese procedimiento o función.

Lo primero que debemos hacer al crear un procedimiento es pensar las cosas que se desean hacer dentro de la función, la información que necesitaremos (y que tendremos que recibir como parámetros) y la información que devolverá. Con estas ideas claras se pueden construir los procedimientos y funciones sin mucha dificultad, siguiendo estas estructuras.

Para un procedimiento

```
Sub nombre (parametro1, parametro2...)
  ... Código del procedimiento
end Sub
```

Para una función

```
Function nombre (parametro1, parametro2...)
  ... Código de la función
end Function
```

Procedimientos. SUB

Decíamos que un procedimiento era una subrutina que se llamaba y realizaba acciones, pero que no devolvía ningún valor y por lo tanto, no era posible utilizarla dentro de una expresión.

Veamos algún ejemplo de procedimiento. Es una subrutina que escribe en la barra de estado un mensaje. No es muy complicada, pero tal como la presentamos aquí no se debería hacer, puesto que utilizamos un bucle vacío para que el navegador esté un poco más lento y el texto salga poco a poco. En lugar de ese bucle deberíamos utilizar una función llamada setTimeout, pero no deseamos introducirla ahora.

```
sub muestraAbajo(texto)
  dim i
  for i=0 to len(texto)
    dim actual
    actual = left(texto,i)
    window.status = actual
    dim j
    'bucle para ralentizar al navegador debería utilizarse la función setTimeOut
    for j=0 to 20000
      j = j
    next
  next
end sub
```

Este ejemplo utiliza además varias funciones de cadenas de caracteres, esperamos que no represente mucho problema para entenderlo. Básicamente es un bucle que va recorriendo toda la cadena de caracteres que recibe por parámetro. A medida que se realiza el bucle se va creando una subcadena de caracteres de la parte izquierda de la cadena original, que cada vez es más larga. Luego se imprime esa cadena en la barra de estado del navegador. Entre ejecución y ejecución del bucle se realiza un retardo, en el segundo bucle for que se debería realizarse con un setTimeout.

Podemos ver a continuación cómo se colocaría un botón en la página que llamase a este procedimiento.

```
<HTML>
<HEAD>
<TITLE>Procedimientos en VBS</TITLE>
<script language=vbscript>
option explicit
sub muestraAbajo(texto)
 dim i
 for i=0 to len(texto)
  dim actual
  actual = left(texto,i)
  window.status = actual
  dim j
  for j=0 to 20000
   j = j
  next
 next
end sub
</script>
</HEAD>
<BODY>
<h1>Procedimientos en VBS</h1>

<P>
<form>
<input type="button" name=b value=ponerAbajo!
      onclick="muestraAbajo('Saludos de Miguel')" language=vbscript>
</form>
</P>
```

```
</BODY>
</HTML>
```

Funciones. Function

Ya vimos lo que consistía una función, que no es más que un trozo de código que opera para devolver un valor. Ahora vamos a ver con detenimiento un ejemplo de su uso.

Vamos a definir una función que realice un cálculo matemático y devuelva el resultado del mismo. Los operandos los vamos a extraer de un formulario. El ejemplo puede ser ahora mismo un poco complejo, por tratar con formularios -que no hemos visto todavía-, pero podemos ver el código de la función y hacernos una idea exacta de su uso, que al fin y al cabo es lo que nos importa.

El código de la función será el siguiente:

```
function operar (operador,op1,op2)
  select case operador
    case "+":
      operar = op1 + op2
    case "-":
      operar = op1 - op2
    case "*":
      operar = op1 * op2
    case else:
      operar = op1 / op2
  end select
end function
```

Vemos que la función recibe tres parámetros, el primero es un operador, que no es más que un texto con el signo de la operación a realizar. Los dos siguientes parámetros son los operadores que hay que tratar.

La función realiza una operación matemática dependiendo de del operador y devuelve en cada caso el resultado conveniente. Fijémonos que para devolver un valor se debe realizar una asignación del nombre de la función al valor que se desea devolver.

No creemos que revista ninguna complicación. Vamos a ver ahora el código que podríamos utilizar para hacer la llamada a la función.

```
miOperador="+"
miOperando1=221
miOperando2=32
resultado = operar(miOperador,miOperando1,miOperando2)
```

Al final de todas estas sentencias la variable resultado tendrá como valor 253.

Veamos el ejemplo completo, que consistía en una calculadora hecha con un formulario, que usa esta función para obtener los resultados.

```
<HTML>
<HEAD>
<link rel=stylesheet type=text/css href=estiloglobal.css>
<TITLE>Funciones en VBS</TITLE>
</HEAD>
<h1>Funciones en VBS</h1>
<script language=vbscript>
function operar (operador,op1,op2)
 select case operador
  case "+":
   operar = op1 + op2
  case "-":
   operar = op1 - op2
  case "*":
   operar = op1 * op2
  case else:
```

```
   operar = op1 / op2
 end select
end function


sub opera ()
 dim res
 operador = document.forms(0).operacion.value
 operando1 = cint(document.forms(0).op1.value)
 operando2 = cint(document.forms(0).op2.value)
  res = operar (operador,operando1,operando2)
 document.forms(0).result.value = res
end sub
</script>
<BODY>
<form>
Operando 1
<input name=op1 >
<br>
Operando 2
<input name=op2 >
<br>
operacion:
<select name=operacion>
<option value="+" selected>+
<option value="-">-
<option value="*">*
<option value="/">/
</select>
<input type=button name=b value="realizar operacion"
        onclick=opera language=vbscript>
<br>
Resultado:
<input name=result >
</BODY>
</HTML>
```

Hemos tenido que utilizar un procedimiento de apoyo para hacer el ejercicio, ya que, en caso de no utilizarlo, haría un poco más compleja a la funció. Podremos enentenderlo todo ya que no reviste mucha complicación y los procedimientos los pudimos ver en el capítulo anterior.

Tenemos un formulario donde podemos ver campos para los operadores, una caja de selección para el operando y un último campo para el resultado. Es interesante también el botón de realizar operación, que es el que lo pone todo en marcha gracias a su manejador de evento onclick, que quiere decir que cuando se pulse sobre el botón se realice una acción. En este caso es una llamada al prodedimiento opera.

En el procedimiento opera podemos ver varias sentencias para extraer la información del formulario y también la llamada a la función que realiza los cálculos. Por último, se introduce en el campo resultado lo que devolbío la función como resultado de realizar las operaciones.

Más sobre procedimientos y funciones

Ahora vamos a ver algunas cosas más sobre subrutinas que nos han quedado en el tintero. Un poco en cajón de sastre.

Llamadas a subrutinas

En Visual Basic Script las funciones se utilizan como partes de expresiones y los procedimientos como si fuera una sentencia independiente.

La llamada a una función, si se utiliza como parte de una expresión se debe llamar utilizando paréntesis.

```
miResultado = suma(1,2)
```

Si no se utiliza como parte de una expresión, no tienen por que utilizarse los paréntesis, pero el resultado de la funcuión (lo que devuelve) se perderá.

```
suma 1,2
```

Call

Es una llamada a una subrutina, utilizada para transferir el flujo de la aplicación hacia una subrutina. Es necesario utilizar paréntesis cuando se utiliza. Además, si se utiliza con una función se perderá el resultado que devuelva.

```
call suma(1,2)
```

Salida de una subrutina

Podemos salirnos de un procedimiento o función en cualquier momento, independientemente de que la función haya termiando o no. El enunciado para escaparse de una función es EXIT, que se puede utilizar en cualquier lugar del procedimiento o función. La palabra exit debe ir acompañada del tipo de subrutina de la que se desea salir, así pues se deberá utilizar o bien **exit function** o **exit sub**.

Capitulo 14
CSS, hojas de estilos

Manual completo y práctico sobre hojas de estilo en cascada (CSS). Aprende a utilizar esta tecnología que te ayudará a crear páginas más atractivas y precisas. El curso contiene la descrición, uso, sintaxis, y lista de atributos para crear estilos.

Introducción a las CSS

El lenguaje HTML está limitado a la hora de aplicarle forma a un documento. Esto es así porque fué concebido para otros usos (científicos sobretodo), distinto a los actuales, mucho más amplios.

Para solucionar estos problemas los diseñadores han utilizado técnicas tales como la utilización de tablas imagenes transparentes para ajustarlas, utilización de etiquetas que no son estádares del HTML y otras. Estas "trampas" han causado a menudo problemas en las páginas a la hora de su visualizaciónen distintas plataformas.

Además, los diseñadores se han visto frustrados por la dificultad con la que, aun utilizando estos trucos, se encontraban a la hora de maquetar las páginas, ya que muchos de ellos venian maquetando páginas sobre el papel, donde el control sobre la forma del documento es absoluto.

Finalmente, otro antecedente que ha hecho necesario el desarrollo de esta tecnología consiste en que las páginas web tienen mezclado en su código HTML el contenido del documento con las etiquetas necesarias para darle forma. Esto tiene sus inconvenientes ya que la lectura del código HTML se hece pesada y difícil a la hora de buscar errores o depurar las páginas. Aunque, desde el punto de vista de la riqueza de la información y la utilidad de las páginas a la hora de almacenar su contenido, es un gran problema que estos textos estáen mezclados con etiquetas incrustadas para dar forma a estos: se degrada su utilidad.

En estas páginas de CSS pretendemos dar a conocer la tecnología con un enfoque práctico para que en pocos capítulos podáis usar las CSS de una manera depurada, reflejando toda nuestra experiencia en su uso. No pretendendemos explorar todos los aspectos de la tecnología ya que para realizar esto necesitariamos un la extensión de un libro entero.

Características y ventajas de las CSS

El modo de fucionamiento de las CSS consiste en definir, mediante una sintaxis especial, la forma de presentación que le aplicaremos a:

- Un web entero, de modo que se puede definir la forma de todo el web de una sola vez.
- Un documento HTML o página, se puede definir la forma, en un pequeño trozo de código en la cabecera, a toda la página.
- Una porción del documento, aplicando estilos visibles en un trozo de la página.
- Una etiqueta en concreto, llegando incluso a poder definir varios estilos diferentes para una sola etiqueta. Esto es muy importante ya que ofrece potencia en nuestra programación. Podemos definir, por ejemplo, varios tipos de párrafos: en rojo, en azul, con margenes, sin ellos...

La potencia de la tecnología salta a la vista. Pero no solo se queda aquí, ya que además esta sintaxis CSS permite aplicar al documento formato de modo mucho más exacto. Si antes el HTML se nos quedaba corto para maquetar las páginas y teníamos que utilizar trucos para conseguir nuestros efectos, ahora tenemos muchas más herramientas que nos permiten definir esta forma:

- Podemos definir la distancia entre líneas del documento.
- Se puede aplicar identado a las primeras líneas del párrafo.
- Podemos colocar elementos en la página con mayor precisión, y sin lugar a errores.
- Y mucho más, como definr la visivilidad de los elementos, margenes, subrallados, tachados...

Y seguimos mostrandoos ventajas, ya que si con el HTML tan sólo podíamos definir atributos en las páginas con pixeles y porcentajes, ahora podemos definir utilizando muchas más unidades como:

- Pixels (px) y porcentaje (%), como antes.
- Pulgadas (in)

- Puntos (pt)
- Centímetros (cm)

Navegadores que lo soportan

Esta tecnología es bastante nueva, por lo que no todos los navegadores la soportan. En concreto, sólo los navegadores de Netscape versiones de la 4 en adelante y de Microsoft a partir de la versión 3 son capaces de comprender los estilos en sintaxis CSS. Además cabe destacar que no todos los navegadores implementan las mismas funciones de hojas de estilos, por ejemplo, Microsoft Internet Explorer 3 no soporta todo lo relativo a capas.

Esto quiere decir que debemos de usar esta tecnología con cuidado, ya que muchos usuarios no podrán ver los formatos que apliquemos a las páginas con CSS. Así pues, utilizad las hojas de estilos cuando estas no vayan a suponer un problema.

Usos de las CSS I

Vamos ahora a describir los diferentes usos de las CSS introducidos en el anterior capítulo. Vamos por orden, describiendo los puntos según su dificultad e importancia.

CSS tiene una sintaxis propia, la veremos a través de ejemplos.

Luego se verá con detalle

Hemos partido este capítulo en dos partes por su extensión y por haber varias formas distintas de aplicar estilos, aquí veremos las más sencillas y en el capítulo siguiente otras más complicadas pero más potentes.

Pequeñas partes de la página

Para definir estilos en secciones reducidas de una página se utiliza la etiqueta **<SPAN>**. Con su atributo **style** indicamos en sintaxis CSS las características de estilos. Lo vemos con un ejemplo, pondremos un párrafo en el que determinadas palabras las vamos a visualizar en color verde.

```
<p>
Esto es un párrafo en varias palabras <SPAN style="color:green">en color verde</SPAN>.
resulta muy fácil.
</p>
```

Que tiene como resultado:

Esto es un párrafo con varias palabras en color verde. resulta muy fácil.

Estilo definido para una etiqueta

De este modo podemos hacer que toda una etiqueta muestre un estilo determinado. Por ejemplo, podemos definir un párrafo entero en color rojo y otro en color azul. Para ello utilizamos el atributo **style**, que es admitido por todas las etiquetas del HTML (siempre y cuando dispongamos de un navegador compatible con CSS).

```
<p style="color:#990000">
Esto es un párrafo de color rojo.
</p>
<p style="color:#000099">
Esto es un párrafo de color azul.
</p>
```

Que tiene como resultado:

Esto es un párrafo de color rojo.

Esto es un párrafo de color azul.

Estilo definido en una parte de la página

Con la etiqueta **<DIV>** podemos definir secciones de una página y aplicarle estilos con el atributo **style**, es decir, podemos definir estilos de una vez a todo un bloque de la página.

```
<div style="color:#000099; font-weight:bold">
<h3>Estas etiquetas van en <i>azul y negrita</i></h3>
<p>
Seguimos dentro del DIV, luego permanecen los etilos
</p>
</div>
```

Que tiene como resultado:

Estas etiquetas van en *azul y negrita*

Seguimos dentro del DIV, luego permanecen los etilos

Hasta aquí este capítulo, en el siguiente seguiremos viendo formas más avanzadas de usar las CSS.

Usos de las CSS y II

Estilo definido para toda una página

Podemos definir, en la cabecera del documento, estilos para que sean aplicados a toda la página. Es una manera muy cómoda de darle forma al documento y muy potente, ya que estos estilos serán seguidos en toda la página y nos ahorraremos así muchas etiquetas HTML que apliquen forma al documento. Además, si deseamos cambiar los estilos de la página lo haremos de una sola vez.

Este ejemplo es más complicado, puesto que se utiliza la sintaxis CSS de manera más avanzada. Pero no te preocupes puesto que con los ejemplos irás aprendiendo su uso y más tarde comentaremos la sintaxis en profundidad.

En el ejemplo vemos que se utiliza la etiqueta <STYLE> colocada en la cabecera de la página para definir los distintos estilos del documento.

A grandes rasgos, entre de <STYLE> y </STYLE>, se coloca el nombre de la etiqueta que queremos definir los estilos y entre llaves -{}- colocamos en sintaxis CSS las características de estilos.

```
<html>
<head>
 <title>Ejemplo de estilos para toda una p&aacute;gina</title>
 <STYLE type="text/css">
 <!--
 H1 {text-decoration: underline; text-align:center}
 P {font-Family:arial,verdana; color: white; background-color: black}
 BODY {color:black;background-color: #cccccc; text-indent:1cm}
 // -->
 </STYLE>
</head>

<body>
<h1>P&aacute;gina con estilos</h1>
Bienvenidos...
<p>Siento ser tan hortera, pero esto es un ejemplo sin m&aacute;s importancia</p>
</body>
</html>
```

Como se puede apreciar en el código, hemos definido que la etiqueta <H1> se presentará

- Subrallado
- Centrada

También, por ejemplo, hemos definido que el cuerpo entero de la página (etiqueta <BODY>) se le apliquen los estilos siguientes:

- Color del texto negro
- Color del fondo grisaceo

- Margen lateral de 1 centímetro

Caber destacar que si aplicamos estilos a la etiqueta <BODY>, estos serán heredados por el resto de las etiquetas del documento. Esto es así siempre y cuando no se vuelvan a definir esos estilos en las siguientes etiquetas, en cuyo caso el estilo de la etiqueta más concreta será el que mande. Puede verse este detalle en la etiqueta <P>, que tiene definidos estilos que ya fueron definidos para <BODY>. Los estilos que se tienen en cuenta son los de la etiqueta <P>, que es más concreta.

Por último, ha de apreciarse los comentarios HTML que engloban toda la declaración de estilos: <!-- Declaración de estilos-->. Estos comentarios se utilizan para que los navegadores antiguos, que no comprenden la sintaxis CSS, no incluyan ese texto en el cuerpo de la página. Si no se pusiera, los navegadores antiguos (por ejemplo Netscape 3) escribirían ese "feo código" en la página.

Estilo definido para todo un sitio web

Una de las características más potentes de la programación con hojas de estilos consiste en que, de una vez, podemos definir los estilos de todo un sitio web. Esto se consigue creando un archivo donde tan sólo colocamos las declaraciones de estilos de la página y enlazando todas las páginas del sitio con ese archivo. De este modo, todas las páginas comparten una misma declaración de estilos y, por tanto, si la cambiamos, cambiarán todas las páginas. Con las ventajas añadidas de que se ahorra en líneas de código HTML (lo que reduce el peso del documento) y se evita la molestia de definir una y otra vez los estilos con el HTML, tal como se comentó anteriormente.

Veamos ahora cómo el proceso para incluir estilos con un fichero externo.

1- Creamos el fichero con la declaración de estilos
Es un fichero de texto normal, que puede tener cualquier extensión, aunque le podemos asignar la extensión .css para aclararnos qué tipo de archivo es. El texto que debemos incluir debe ser escrito exclusivamente en sintaxis CSS, es decir, sería erroneo incluir código HTML en el: etiquetas y demás. Podemos ver un ejemplo a continuación.

```
P  {
 font-size : 12pt;
 font-family : arial,helvetica;
 font-weight : normal;
}

H1  {
 font-size : 36pt;
 font-family : verdana,arial;
 text-decoration : underline;
 text-align : center;
 background-color : Teal;
}

TD  {
 font-size : 10pt;
 font-family : verdana,arial;
 text-align : center;
 background-color : 666666;
}

BODY  {
 background-color : #006600;
 font-family : arial;
 color : White;
}
```

2- Enlazamos la pána web con la hoja de estilos
Para ello, vamos a colocar la etiqueta <LINK> con los atributos

- **rel="STYLESHEET"** indicando que el enlace es con una hoja de estilos
- **type="text/css"** porque ela archivo es de texto, en sintaxis CSS
- **href="estilos.css"** indica el nombre del fichero fuente de los estilos

Veamos una página web entera que enlaza con la declaración de estilos anterior:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>
<head>
 <link rel="STYLESHEET" type="text/css" href="estilos.css">
 <title>P&aacute;gina que lee estilos</title>
</head>

<body>
<h1>P&aacute;gina que lee estilos</h1>
Esta p&aacute;gina tiene en la cabecera la etiqueta necesaria para enlazar con la hoja de
estilos. Es muy f&aacute;cil.
<br>
<br>
<table width="300" cellspacing="2" cellpadding="2" border="0">
<tr>
   <td>Esto est&aacute; dentro de un TD, luego tiene estilo propio, declarado en el fichero
externo</td>
</tr>
<tr>
   <td>La segunda fila del TD</td>
</tr>
</table>

</body>
</html>
```

Reglas de importancia en los estilos

Los estilos se heredan de una etiqueta a otra, como se indicó anteriormente. Por ejemplo, si tenemos declarado en el <BODY> unos estilos, por lo general, estas declaraciones también afectatarán a etiquetas que estén dentro de esta etiqueta, o lo que es lo mismo, dentro de todo el cuerpo.

En muchas ocasiones más de una declaración de estilos afecta a la misma porción de la página. Siempre se tiene en cuenta la declaración más particular. Pero las declaraciones de estilos se pueden realizar de múltiples modos y con varias etiquetas, también entre estos modos hay una jerarquía de importancia para resolver conflictos entre varias declaracionesde estilos distintas para una misma porción de página. Se puede ver a continuación esta jerarquía, primero ponemos las formas de declaración más generales, y por tanto menos respetadas en caso de conflicto:

- Declaración de estilos con fichero externo. (Para todo un sitio web)
- Declaración de estilos para toda la página. (Con la etiqueta <STYLE> en la cabecera de la página)
- Estilos definidos en una parte de la página. (Con la etiqueta <DIV>)
- Definidos en una etiqueta en concreto. (Utilizando el atributo style en la etiqueta en cuestión)
- Declaración de estilo para una porción pequeña del documento. (Con la etiqueta <SPAN>)

Ya vimos cómo incluir estilos en la página, de todas las maneras posibles e hicimos un repaso con la lista anterior. Ahora estás en condiciones de empezar a usar las hojas de estilo en cascada para mejorar tus páginas y aumentar la productividad de tu trabajo. Pero estate atento a los siguientes capítulos donde aprenderás las lecciones que te faltan para dominar bien la materia: conocer la sintaxis, los distintos atributos de estilos y otras cosas que mejorarán tus páginas.

Otra manera de definir estilos en un archivo externo

Veamos ahora otra manera de importar una declaración externa de estilos CSS: @import url("archivo_a_importar.css"), que se utiliza para importar unas declaraciones de estilos guardadas en la ruta que se indica entre paréntesis. (las comillas son opcionales, pero los paréntesis son obligatorios, por lo menos, en Explorer).

Se debe incluir en la declaración de estilos global a una página, es decir entre las etiquetas <style type="text/css"> y </style>, que se colocan en la cabecera del documento.

Es importante señalar que la sentencia de importación del archivo CSS se debe escribir en la primera línea de la declaración de estilos, algo parecido al código siguiente.

```
<style type="text/css">
@import url ("estilo.css");
body{
  background-color:#ffffcc;
}
</style>
```

El funcionamiento es el mismo que si escribiésemos todo el fichero a importar dentro de las etiquetas de los estilos, con la salvedad de que, si redefinimos dentro del código HTML (entre las etiquetas </style>) estilos que habían quedado definidos en el archivo externo, los que se aplicarán serán los que hayamos redefinido.

Así, en el ejemplo anterior, aunque hubiésemos definido en estilo.css un color de fondo para la página, el color que prevalecería sería el definido a continuación de la importación: #ffffcc

La diferencia entre este tipo de importación del tipo y la que hemos visto anteriormente:

```
<link rel="stylesheel" type="text/css" href="hoja.css">
```

Es que @import url ("estilo.css") se suele utilizar cuando hay unas pautas básicas en el trabajo con los estilos (que se definen en un archivo a importar) y unos estilos específicos para cada página, que se definen a continuación, dentro del código HTML entre las etiquetas </style>, como es el caso del ejemplo visto anteriormente.

Sintaxis CSS

Tal como se vió en los ejemplos la sintaxis es bastante sencilla y repetitiva. Vamos a verla:

- Para definir un estilo se utilizan atributos como font-size,text-decoration... segudos de dos puntos y el valor que le deseemos asignar. Podemos definir un estilo a base de definir muchos atributos separados por punto y coma.
  Ejemplo:
  font-size: 10pt; text-decoration: underline; color: black; (el último punto y coma de la lista de atributos es opcional)
- Para definir el estilo de una etiqueta se escribe la etiqueta seguida de la lista de atributos encerrados entre llaves.
  Ejemplo:
  H1{text-align: center; color:black}
- Los valores que se pueden asignar a los atributos de estilo se pueden ver en una tabla en el siguiente capítulo. Muchos estos valores son unidades de medida, por ejemplo, el valor del tamaño de un margen o el tamaño de la fuente. Las unidades de medida son las siguientes:

Puntos	pt
Pulgadas	in
Centímetros	cm
pixels	px

Hasta aquí, queda dicho todo lo relativo a la sintaxis. En el siguiente capítulo podrás encontrar una lista de los atributos de las hojas de estilo en cascada.

Atributos de las hojas de estilo

Tanto para practicar en tu aprendizaje como para trabajar con las CSS lo mejor es disponer de una tabla donde se vean los distintos atributos y valores de estilos que podemos aplicarle a las páginas web.

Aquí puedes ver la tabla de los atributos CSS, tenla a mano cuando utilices las CSS.

Nombre del atributo	Posibles valores	Ejemplos
FUENTES - FONT		
color	valor RGB o nombre de color	color: #009900; color: red;
Sirve para indicar el color del texto. Lo admiten casi todas las etiqetas de HTML. No		

todos los nombres de colores son admitidos en el estandar, es aconsejable entonces utilizar el valor RGB.

font-size	xx-small | x-small | small | medium | large | x-large | xx-large Unidades de CSS	font-size:12pt; font-size: x-large;

Sirve para indicar el tamaño de las fuentes de manera más rígida y con mayor exactitud.

font-family	serif | sans-serif | cursive | fantasy | monospace Todas las fuentes habituales	font-family:arial,helvetica; font-size: fantasy;

Con este atributo indicamos la familia de tipografia del texto. Los primeros valores son genéricos, es decir, los exploradores las comprenden y utilizan las fuentes que el usuario tenga en su sistema.
También se pueden definir con tipografías normales, como ocurría en html. Si el nombre de una fuente tiene espacios se utilizan comillas para que se entienda bien.

font-weight	normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900	font-weight:bold; font-weight: 200;

Sirve para definir la anchura de los caracteres, o dicho de otra manera, para poner negrillas con total libertad.
Normal y 400 son el mismo valor, así como bold y 700.

font-style	normal | italic | oblique	font-style:normal; font-style: italic;

Es el estilo de la fuente, que puede ser normal, itálica u oblícua. El estilo oblique es similar al italic.

PÁRRAFOS - TEXT		
line-height	normal y unidades CSS	line-height: 12px; line-height: normal;

El alto de una línea,y por tanto, el espaciado entre líneas. Es una de esas características que no se podian mofificar utilizando HTML.

text-decoration	none | [underline || overline || line-through]	text-decoration: none; text-decoration: underline;

Para establecer la decoración de un texto, es decir, si está subrayado, sobrerayado o tachado.

text-align	left | right | center | justify	text-align: right; text-align: center;

Sirve para indicar la alineación del texto. Es interesante destacar que las hojas de estilo permiten el justificado de texto, aunque recuerda que no tiene por que funcionar en todos los sistemas.

text-indent	Unidades CSS	text-indent: 10px; text-indent: 2in;

Un atributo que sirve para hacer sangrado o márgenes en las páginas. Muy útil y novedosa.

text-transform	capitalize | uppercase | lowercase | none	text-transform: none; text-transform: capitalize;

Nos permite transformar el texto, haciendo que tenga la primera letra en mayúsculas de todas las palabrs, todo en mayúsculas o minúsculas.

FONDO - BACKGROUND		
Background-color	Un color, con su nombre o su valor RGB	background-color: green; background-color: #000055;

Sirve para indicar el color de fondo de un elemento de la página.

Background-image	El nombre de la imagen con su camino relativo o absoluto	background-image: url(mármol.gif) ; background-image: url(http://www.x.com/fondo.gif)

Colocamos con este atributo una imagen de fondo en cualquier elemento de la página.

BOX - CAJA		
Margin-left	Unidades CSS	margin-left: 1cm; margin-left: 0,5in;

Indicamos con este atributo el tamaño del margen a la izquierda

Margin-right	Unidades CSS	margin-right: 5%; margin-right: 1in;
Se utiliza para definir el tamaño del margen a la derecha		
Margin-top	Unidades CSS	margin-top: 0px; margin-top: 10px;
Indicamos con este atributo el tamaño del margen arriba de la página		
Margin-bottom	Unidades CSS	margin-bottom: 0pt; margin-top: 1px;
Con el se indica el tamaño del margen en la parte de abajo de la página		
Padding-left	Unidades CSS	padding-left: 0.5in; padding-left: 1px;
Indica el espacio insertado, por la izquierda, entre el borde del elemento-continente y el contenido de este. Es parecido a el atributo cellpadding de las tablas. El espacio insertado tiene el mismo fondo que el fondo del elemento-continente.		
Padding-right	Unidades CSS	padding-right: 0.5cm; padding-right: 1pt;
Indica el espacio insertado, en este caso por la derecha, entre el borde del elemento-continente y el contenido de este. Es parecido a el atributo cellpadding de las tablas. El espacio insertado tiene el mismo fondo que el fondo del elemento-continente.		
Padding-top	Unidades CSS	padding-top: 10pt; padding-top: 5px;
Indica el espacio insertado, por arriba, entre el borde del elemento-continente y el contenido de este.		
Padding-bottom	Unidades CSS	padding-right: 0.5cm; padding-right: 1pt;
Indica el espacio insertado, en este caso por abajo, entre el borde del elemento-continente y el contenido de este.		
Border-color	color RGB y nombre de color	border-color: red; border-color: #ffccff;
Para indicar el color del borde del elemento de la página al que se lo aplicamos. Se puede poner colores por separado con los atributos border-top-color, border-right-color, border-bottom-color, border-left-color.		
Border-style	none | dotted | solid | double | groove | ridge | inset | outset	border-style: solid; border-style: double;
El estilo del borde, los valores significan: none=ningun borde, dotted=punteado (no parece funcionar), solid=solido, double=doble borde, y desde groove hasta outset son bordes con varios efectos 3D.		
border-width	Unidades CSS	border-width: 10px; border-width: 0.5in;
El tamaño del borde del elemento al que lo aplicamos.		

float	none | left | right	float: right;
Sirve para alinear un elemento a la izquierda o la derecha haciendo que el texto se agrupe alrededor de dicho elemento. Igual que el atributo align en imagenes en sus valores right y left.		
clear	none | right | left	clear: right;
Si este elemento tiene a su altura imagenes u otros elementos alineados a la derecha o la izquierda, con el atributo clear hacemos que se coloque en un lugar donde ya no tenga esos elementos a el lado que indiquemos.		

La especificación de estilos CSS es muy amplia, seguro que se queda en el tintero algún atributo de estilo, pero creo que la inmensa mayoría están, y por supuesto la selección de los más importantes.

Trucos avanzados con CSS

Las hojas de estilos son un tema largo del que se han escrito libros enteros. Nosotros nos centramos en los temas prácticos y por ello vamos a acabar ya con este capítulo, en unos cuantos puntos

Definir estilos utilizando clases

Las clases nos sirven para crear definiciones de estilos que se pueden utilizar repetidas veces.

Una clase se puede definir entre las etiquetas <STYLE> (en la cabecera del documento), o en un archivo externo a la página. Para definirlas utilizamos la siguiente sintaxis, un punto seguido del nombre de la clase y entre llaves los atributos de estilos deseados. De esta manera:

.nombredelaclase {atributo: valor;atributo2:valor2; ...}

Una vez tenemos una clase, podemos utilizarla en cualquier etiqueta HTML. Para ello utilizaremos el atributo class, poniéndole como valor el nombre de la clase, de esta forma:

<ETIQUETA class="nombredelaclase">

Ejemplo de la utilización de clases

```
<html>
<head>
 <title>Ejemplo de la utilizaci&oacute;n de clases</title>
 <STYLE type="text/css">
  .fondonegroletrasblancas {background-color:black;color:white;font-size:12;font-family:arial}
  .letrasverdes {color:#009900}
 </STYLE>
</head>

<body>
<h1 class=letrasverdes>Titulo 1</h1>
<h1 class=fondonegroletrasblancas>Titulo 2</h1>

<p class=letrasverdes>
Esto es un p&aacute;rrafo con estilo de letras verdes</p>
<p class=fondonegroletrasblancas>
Esto es un p&aacute;rrafo con estilo de fondo negro y las letras blancas. Es todo!</p>
</body>
</html>
```

Estilo en los enlaces

Una técnica muy habitual, que se puede realizar utilizando las hojas de estilo en cascada y no se podía en HTML, es la definición de estilos en los enlaces, quitandoles el subrayado o hacer enlaces en la misma página con distintos colores.

Para aplicar estilo a los enlaces debemos definirlos para los distintos tipos de enlaces que existen (visitados, activos...). Utilizaremos la siguiente sintaxis, en la declaración de estilos global de la página (<STYLE>) o del sitio (Definición en un archivo externo):

Enlaces normales
A:link {atributos}

Enlaces visitados
A:visited {atributos}

Enlaces activos (Los enlaces están activos en el presiso momento en que se pincha sobre ellos)
A:active {atributos}

Enlaces hover (Cuando el ratón está encima de ellos, solo funciona en ieplorer)
A:hover {atributos}

El atributo para definir enlaces sin subrayado es **text-decoration:none**, y para darles color es color:tu_color.

También podemos definir el estilo de cada enlace en la propia etiqueta <A>, con el atributo style. De esta manera podemos hacer que determinados enlaces de la página se vean de manera distinta

Ejemplo de estilos en enlaces

```
<html>
<head>
 <title>Ejemplos de estilo en enlaces</title>
 <STYLE type="text/css">
```

```
 A:link {text-decoration:none;color:#0000cc;}
 A:visited {text-decoration:none;color:#ffcc33;}
 A:active {text-decoration:none;color:#ff0000;}
 A:hover {text-decoration:underline;color:#999999;font-weight:bold}
 </STYLE>
</head>

<body>

<a href="http://dominioinexistente.nofunciona.com">Enlace normal</a>
<br>
<br>
<a href="enlaces.html">Enlace visitado</a>
Pulsar este enlace para verlo activo,
poner el rat&oacute;n por encima para que cambie.

</body>
</html>
```

URL como valor de un atributo:

Determinados atributos de estilos, como **background-image** necesitan una URL como valor, para indicarlas podemos definir tanto caminos relativos como absolutos. Así pues, podemos indicar la URL de la imagen de fondo de estas dos maneras:

background-image: url(fondo.gif) En caso de que la imagen esté en el mismo directorio que la página. **background-image: url(http://www.desarrolloweb.com/images/fondo.gif)** ç

Ocultar estilos en navegadores antiguos

En caso de definir dentro de la etiqueta <STYLE> unas declaraciones de estilos debemos asegurarnos que estas no se imprimirán en la página web con navegadores antiguos. Pensar en un navegador que no reconozca la etiqueta <STYLE>, pensará que corresponde con algo que no entiende y se olvidará de la etiqueta. Lo siguiente que encuentra es texto normal y hará que este se vea en la página, como con cualquier otro texto.

Para evitarlo debemos ocultar con comentarios HTML (<!-- esto es un comentario -->) todo lo que hay dentro de la etiqueta <STYLE>. Se puede ver un ejemplo de esto a continuación:

De este modo hemos terminado nuestro manual de CSS, que espero pueda ayudar a hacer páginas mejores y más rápidamente.

Quiero recordaros que siempre es útil ver como han hecho sus páginas otros progradores de Internet. Para ver una página definida enteramente con hojas de estilos visitar la dirección www.guiarte.com

Aplicación de estilo avanzada a los enlaces

En este artículo vamos a ver cómo podríamos crear una barra de navegación bastante dinámica utilizando únicamente las Hojas de Estilo en Cascada. En el ejemplo vamos a construir una barra de navegación que contiene enlaces de varios colores que cambian de tonalidad al pasar el ratón por encima.

Cómo poner estilo a los enlaces

Ya lo vimos en capítulos anteriores de nuestro manual de CSS, pero lo repetimos aquí. Se define el estilo de los enlaces asignando su apariencia en sus distintos estados:

- Enlace no visitado. Se define con el atributo link.
- Enlace visitado. Se define con el atributo visited.
- Enlace activo (cuando se está pulsando). Se define con active.
- Enlace con el reton encima. Se define con hover.

Esta definición se realiza en la cabecera de la página, entre las etiquetas <STYLE> Y </STYLE>, y es global a toda la página.

Un ejemplo de esto se puede ver en esta declaración de estilos:

```
<STYLE type="text/css">
   A:link {text-decoration:none;color:#0000cc;}
   A:visited {text-decoration:none;color:#ffcc33;}
   A:active {text-decoration:none;color:#ff0000;}
   A:hover {text-decoration:underline;color:#999999;}
</STYLE>
```

Cómo dar estilo a un enlace en concreto

Si queremos resaltar nuestra barra de navegación probablemente querramos colocarla en una tabla de nuestra página web, con un color que contraste un poco con el fondo. En un caso como este, será necesario que los enlaces de la barra de navegación y los enlaces normales de la página tengan colores distintos, por estar situados sobre dos tipos de fondos distintos.

Es por esto que los enlaces de la barra van a tener un color distinto de los definidos en la cabecera de la página, con los estilos. Esto lo podemos conseguir de esta manera.

```
<a href="#" style="color:#ff0000">Mi enlace</a>
```

Hemos definido el color de un enlace de una manera específica, utilizando el atributo style, de modo que este enlace siempre tendrá el color indicado, independientemente de su estado.

Es un enlace amarillo, que quedaría muy bien resaltado sobre un fondo oscuro, como se puede ver en el ejemplo de barra de navegación siguiente.

Enlace 1
Otro enlace
Enlace 3

En la tabla anterior tenemos enlaces amarillos en una web donde los enlaces son azules por defecto.

Cómo utilizar las clases al aplicar estilo a los enlaces

También vimos en anteriores capítulos que el uso de clases puede ser muy útil a la hora de definir estilos especiales que podemos aplicar a las etiquetas que queramos. A la hora de trabajar con los enlaces también podemos aplicar las clases para definir distintos tipos de enlaces, que tienen distintos tipos de estilos.

```
A.clase1:visited {color:#ff0000;}
A.clase1:active {color:#ff0000;}
A.clase1:link {color:#ff0000}
A.clase1:hover {color:#00ff00}
```

La ventaja al utilizar las clases con los estilos de los enlaces es que podemos especificar un formato distinto al enlace dependiendo de su estado: visitado o no, activo o con el ratón sobre él.

Por si no quedó claro, al especificar el estilo con el atributo style del enlace sólo podíamos decir que el enlace lo queremos en amarillo, y siempre lo tendremos en amarillo (sea visitado o no, activo, o estemos o no con el ratón encima). Con las clases definimos un nuevo tipo de enlace al que podemos dar distintos formatos dependiendo su estado.

Otras ventajas de utilizar las clases consisten en que escribimos una única vez los estilos y que podemos cambiar el color de todos los enlaces de la clase con cambiar la delaración.

A partir de lo que acabamos de aprender podemos crear el ejemplo de barra de navegación dinámica utilizando CSS que habíamos visto al principio del capítulo. El código sería el siguiente.

```
<html>
<head>
   <title>Ejemplo CSS para enlaces</title>
<style type="text/css">
   A:link {color:#0000cc;}
   A:visited {color:#0000cc;}
   A:active {color:#0000cc;}
   A:hover {color:#0000ff;}

   A.clase1:visited {color:#ffff00;}
   A.clase1:active {color:#ffff00;}
   A.clase1:link {color:#ffff00}
   A.clase1:hover {color:#00ff00}
```

```
   A.clase2:visited {font-size:12;color:#ffffff;}
   A.clase2:active {font-size:12;color:#ffffff;}
   A.clase2:link {font-size:12;color:#ffffff;}
   A.clase2:hover {font-size:12;color:#ffff33;}

   body {font-family:arial;font-size:11;font-weight:bold}
   td {font-family:arial;font-size:11;font-weight:bold}
</style>
</head>

<body>
<a href="#">Este enlace es normal</a>
<br>
<br>
<br>
<span style="font-weight:normal;font-size:10">
Los enlaces de esta barra son especiales,
<br>
están definidos por clases
</span>
<br>
<table width="110" cellspacing="1" cellpadding="2" border="0">
<tr>
   <td bgcolor="#aa0000"><a href="#" class="clase2">Opciones 1</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Enlace clase1</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Otro de clase1</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Más enlaces</a></td>
</tr>
<tr>
   <td bgcolor="red"><a href="#" class="clase1">Todavía más</a></td>
</tr>
</table>

</body>
</html>
```

Qué son las capas

Veamos una pequeña introducción a lo que son las capas, la etiqueta HTML <DIV> utilizada para construirla y los atributos CSS con los que podemos aplicar estilos.

Como ya hemos visto en nuestro manual de CSS, <SPAN> sirve para aplicarle estilo a una pequeña parte de una página HTML. Por ejemplo, con ella podríamos hacer que una parte de un párrafo se coloree en rojo. Con <SPAN> no es habitual englobar un trozo muy grande de texto, por ejemplo el que comprenda a varios párrafos.

Con <DIV> también podemos aplicar estilo a partes de la página HTML.

La diferencia entre <SPAN> y <DIV> es que con esta última si que podemos aplicar estilo a una parte más amplia de la página, por ejemplo a tres párrafos. **Además que la etiqueta <DIV> tiene un uso adicional que es el de crear divisiones en la página a las que podremos aplicar una cantidad adicional de atributos para modificar sus comportamientos**. Por ejemplo, con el atributo align de HTML podemos alinear la división al centro, izquierda, derecha o justificado. Pero **su uso más destacado es el de convertir esa división en una capa**.

Una capa es una división, una parte de la página, **que tiene un comportamiento muy independiente** dentro de la ventana del navegador, ya que la podemos colocar en cualquier parte de la misma y la podremos mover por ella independientemente, por poner dos ejemplos. En el uso de capas se basan muchos de los efectos más comunes del DHTML.

Las etiquetas <LAYER> e <ILAYER> tienen como objetivo construir capas, pero estas no son compatibles más que con Netscape, por lo que es recomendable utilizar la etiqueta <DIV> para hacer capas preferentemente a las otras dos.

Los atributos que podemos aplicar a estas etiquetas, pero en concreto a las dos recomendadas <SPAN> y <DIV>, son principalmente de estilos CSS. Estos atributos nos permiten, como hemos podido ver en el manual de hojas de estilo en cascada de desarrolloweb, modificar de una manera muy exhaustiva la presentación de los contenidos en la página. Para aplicar estilos a estas etiquetas se utiliza el atributo de HTML style, de esta manera:

```
<SPAN style="text-decoration:underline;font-weight:bold">...</SPAN>
```

```
<DIV style="color:red;font-size:10px">...</DIV>
```

Como ya pudimos ver muchos ejemplos en el manual de CSS, nos referimos a él para ampliar esta información. Pero no habíamos visto todavía una serie de atributos que nos sirven para posicionar la división en la página como una capa. Estos atributos se pueden aplicar a la etiqueta <DIV> que es la que servía para crear capas compatibles con todos los navegadores.

Los atributos para que la división sea una capa son varios y se pueden ver a continuación.

```
<div id="c1" style="position:absolute; left: 200px; top: 100px;">
Hola!
</div>
```

El primero, **position**, indica que se posicione de manera absoluta en la página y los segundos, **left** y **top**, son la distancia desde el borde izquierdo de la página y el borde superior.

Hay otros atributos especiales para capas como **width** y **height** para indicar la anchura y altura de la capa, **Z-index** que sirve para indicar qué capas se ven encima de qué otras, **clip** que sirve para recortar una capa y hacer que partes de ella no sean visibles, o **visibility** para definir si la capa es visible o no. Estos y otros atributos los veremos en el siguiente capítulo, donde hablaremos del posicionamiento de capas.

Atributos para capas

Hemos visto en el capítulo anterior qué son las capas y algunas pequeñas muestras sobre cómo crearlas y darle algún estilo. Ahora vamos a ver en detenimiento los atributos específicos para aplicar posicionamiento a una capa y otros estilos.

Antes que nada cabe decir que una capa puede tener cualquier atributo de estilos de los que hemos visto en el manual de CSS. Así, el atributo color indica el color del texto de la capa, el atributo font-size indica el tamaño del texto y así con todos los atributos ya vistos.

Ahora bien, existen una serie de atributos que sirven para indicar la forma, el tamaño de las capas, la visibilidad, etc, que no hemos visto en capítulos anteriores y que veremos a continuación.

Atributo position

Indica el tipo de posicionamiento de la capa. Puede tener dos valores, relative o absolute.

- relative indica que la posición de la capa es relativa a el lugar donde se estaba escribiendo en la página en el momento de escribir la capa con su etiqueta

- absolute indica que la posición de la capa se calcula con respecto al punto superior izquierdo de la página.

Atributo top

Indica la distancia en vertical donde se colocará la capa. Si el atributo position es absolute, top indica la distancia del borde superior de la capa con respecto al borde superior de la página. Si el atributo position era relative, top indica la distancia desde donde se estaba escribiendo en ese momento en la página hasta el borde superior de la capa.

Atributo left

Básicamente funciona igual que el atributo top, con la diferencia que el atributo left indica la distancia en horizontal a la que estará situada la capa.

Atributo height

Sirve para indicar el tamaño de la capa en vertical, es decir, su altura.

Atributo width

Indica la anchura de la capa

Atributo visibility

Sirve para indicar si la capa se puede ver en la página o permanece oculta al usuario. Este atributo puede tener tres valores.

- visible sirve para indicar que la capa se podrá ver.
- hidden indicará que la capa está oculta.
- inherit es el valor por defecto, que quiere decir que hereda la visibilidad de la capa donde está metida la capa en cuestión. Si la capa no está metida dentro de ninguna otra se supone que está metida en la capa documento, que es toda la página y que siempre está visible.

Atributo z-index

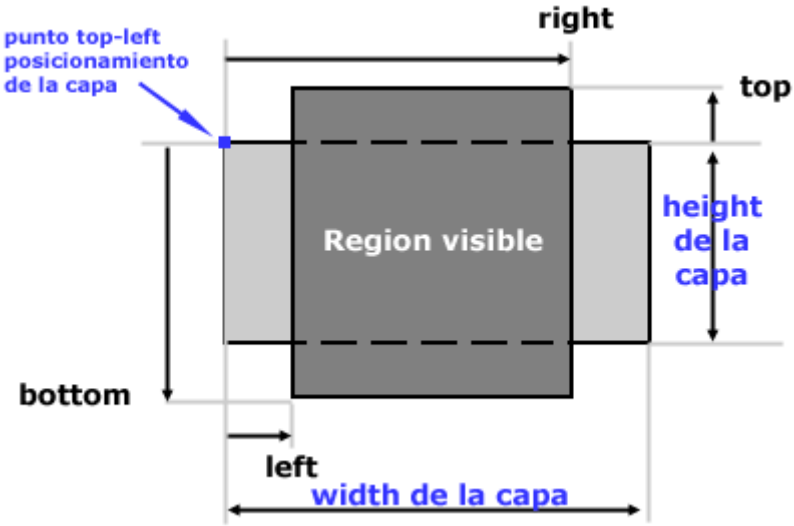
Sirve para indicar la posición sobre el eje z que tendrán las distintas capas de la página. Dicho de otra forma, con z-index podemos decir qué capas se verán encima o debajo de otras, en caso de que estén superpuestas. El atributo z-index toma valores numéricos y a mayor z-index, más arriba se verá la página.

Atributo clip

Es un atributo un poco difícil de explicar. En concreto sirve para recortar determinadas áreas de la capa y que no se puedan ver. Una capa que está visible se puede recortar para que se vea, pero que no se vea algún trozo de esta. El cliping se indica por medio de 4 valores, con esta sintaxis.

rect (<top>, <right>, <bottom>, <left>)

Los valores <top>, <right>, <bottom> y <left> indican distancias que se pueden apreciar en este esquema.



Este es un ejemplo de capa que utiliza todos los atributos que hemos visto en este artículo y alguno más para aplicar estilo a la capa.

<div style="clip: rect(0,158,148,15); height: 250px; width: 170px; left: 10px; top: 220px; position: absolute; visibility: visible; z-index:10; font-size: 14pt; font-family: verdana; text-align: center; background-color: #bbbbbb">

Esta capa tiene un cliping, por eso se ve entrecortada.

Esto es una capa de prueba

Problema con el posicionamiento absoluto de capas

He recibido una consulta en mi correo sobre colocación de capas de manera absoluta, pero en la que no nos importe la definición de la pantalla del usuario y otros ir y venir de los elementos HTML. Nuestro compañero expresó su duda de la siguiente manera:

Si trabajamos con position:absolute dando un left y un top funciona si tienes tu página alineada a la izquierda. Mi página está alineada en el centro, entonces lo que sucede es que dependiendo de la resolución de pantalla que tengas (ancho de 800px,1024px,etc) me baila toda la página y no cuadra nada.

Primero que todo, debemos saber que si trabajamos con el position relative las capas se colocan en el lugar donde aparecen dentro del código HTML. De este modo, si colocamos una capa con position relative dentro de una celda de una tabla, dicha capa aparecería dentro de la celda donde la estamos colocando, independientemente del lugar donde se sitúe la celda al cambiar la definición de la pantalla.

El problema de esta solución es que la capa haría crecer la celda de la tabla donde queremos colocarla (al igual que cualquier otro elemento HTML que colocásemos dentro de la tabla) y es muy probable que nuestro diseño no nos permita este hecho. Seguramente ya habrías notado este problema y si no es así te invito a que crees la capa que intentas colocar con el atributo position a relative para ver si con eso tu problema ya está resuelto.

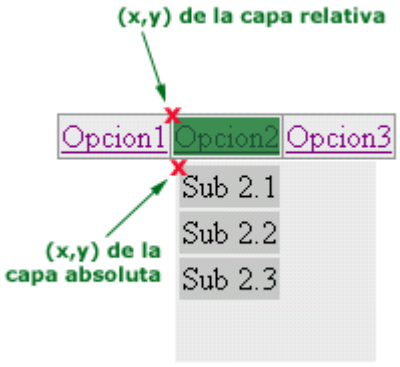
En casi todos los casos, la capa que intentamos colocar va a tener que tener el position absolute, porque con relative no arreglamos totalmente el problema. Entonces volvemos a el problema inicial, que era situar la capa con position absolute en el lugar exacto, independientemente de la definición de pantalla.

La solución final que propongo pasa por aplicar algún truquillo. De hecho, estuve hace unos días preguntándome sobre esa cuestión y al final encontré la solución, aunque no se me ocurrió a mi, sino que la extraje de www.cross-browser.com.

La idea es un poco compleja y para su puesta en marcha debemos realizar una serie de acciones que, sinceramente, considero excesivas para un problema inicialmente sencillo. Así pues, que no asuste lo que voy a soltar a continuación, que luego trataré de explicarlo un poco mejor.

Nuestro esquema de trabajo consistirá en una capa con posición relativa, que nos servirá de "ancla" y otra con la posición absoluta, donde colocaremos el contenido final a mostrar en la capa.

La capa relativa la colocaremos en el lugar aproximado donde queramos que aparezca la capa absoluta. La capa absoluta la posicionaremos, una vez cargada la página, en un lugar próximo a la capa relativa. Por supuesto, estas acciones las vamos a tener que realizar con Javascript, que es el lenguaje que nos permite actualizar las posiciones de las capas dinámicamente.



Detenidamente

Decíamos que habría que colocar una capa relativa cercana al lugar donde tiene que aparecer la capa con position absolute. Insisto en que las capas relativas se colocan en el lugar donde las metemos dentro del código HTML, por lo que será fácil colocar la capa relativa en el lugar exacto y que este lugar sea válido para cualquier definición.

La segunda capa, la que tiene el contenido final, la pondremos inicialmente en una posición cualquiera y escondida, de manera que no se vea que está mal colocada. Una vez terminada de cargar la página, podremos acceder a la posición de la capa relativa, extrayendo sus valores top y left y colocándolos en los correspondientes top y left de la capa con posición absoluta. Una vez marcada la posición de la capa absoluta podemos volverla visible.

A la vista de la imagen siguiente, la capa con posición relativa la hemos colocado en el enlace. En realidad habría tres capas con posición relativa para poder posicionar otras tantas capas con posición absoluta. La parte que vemos sombreada de verde corresponde al espacio que abarcaría la capa relativa.

Su posición sería la que está marcada por el aspa roja que aparece en su esquina superior izquierda. Dicha posición depende del lugar donde aparezcan los enlaces en la página.

Luego, con Javascript deberíamos asignar la posición de la capa absoluta de una manera parecida a

esta.

left de la capa absoluta = left de la capa relativa
top de la capa absoluta = top de la capa relativa + altura de la capa relativa

Podemos sumarle algún píxel más a la posición de la capa, si es que queremos moverla un poco abajo y a la derecha, tal como hemos visto en la imagen.

No pretendo en este artículo, muy a mi pesar y por falta de espacio y tiempo, explicar cómo se hacen esas operaciones de Javascript. Advierto que si no se conoce nada de Javascript va a ser imposible ponerse con una tarea tan tediosa como el manejo de capas. Si por el contrario, ya hemos tenido contacto con Javascript y DHTML anteriormente, no debería ser un problema realizar esas acciones.

Cómo evitar que una página se imprima

Para ello, hay que echar mano de las hojas de estilo. Tanto si el documento tiene una hoja ya asociada como sino, lo que vamos a hacer es asociarle un nueva hoja de estilos. Dicha hoja contendrá un único estilo, con el código necesario para ocultar un elemento:

```
.nover{
visibility:hidden
}
```

A la hora de asociar la hoja de estilos, se le añade un modificador a la etiqueta HTML que enlaza con el fichero .css que permite especificar para qué tipo de medio se aplicará esta hoja. En nuestro caso, se aplica en el ámbito de la impresión, por lo que se utiliza el atributo media="print".

```
<link href="nombre_hoja" rel="stylesheet" type="text/css" media="print">
```

Una vez hecho esto, basta que toda nuestra página este dentro de un elemento div, que pertenezca a la clase nover.

```
<body>
<div class="nover">

-- Contenido --


</div>
</body>
```

Al hacer esto se provoca que en pantalla se visualice la página, pero que si por el contrario se decide imprimir, se le aplicará la hoja de estilos de impresión, en la que el elemento esta puesto como no visible, por lo que no se imprimirá.

Codigo Completo:

Veamos el código íntegro de la página web y la hoja de estilos asociada.

Página HTML

```
<html>
<head>
<link href="estilos.css" rel="stylesheet" type="text/css" media="print">
</head>
<body>
<div class="nover">

... contenido de la pagina

</div>
</body>
</html>
```

Hoja estilos: estilos.css

```
.nover{
visibility:hidden
}
```

Nota: Esta catacterística de las hojas de estilos funciona con éxito en navegadores IExplorer 6, Netscape 7 y Opera 7. No la hemos probado en otras versiones.

Capitulo 15
Introducción a XML

Una breve intruducción al mundo XML que explica qué es este lenguaje y sus tecnologías relacionadas.

Qué es XML

XML es una tecnología en realidad muy sencilla que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Vamos a ver a lo largo de varios capítulos una introducción al mundo XML, es decir, al lenguaje así como a las tecnologías que trabajan con él, sus usos, ventajas y modos de llevar a cabo las tareas.

XML, con todas las tecnologías relacionadas, representa una manera distinta de hacer las cosas, más avanzada, cuya principal novedad consiste en permitir compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. Así pues, el XML juega un papel importantísimo en este mundo actual, que tiende a la globalización y la compatibilidad entre los sistemas, ya que es la tecnología que permitirá compartir la información de una manera segura, fiable, fácil. Además, XML permite al programador y los soportes dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corre a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello.

Vemos que XML no está sólo, sino que hay un mundo de tecnologías alrededor de él, de posibilidades, maneras más fáciles e interesantes de trabajar con los datos y, en definitiva, un avance a la hora de tratar la información, que es en realidad el objetivo de la informática en general. XML, o mejor dicho, el mundo XML no es un lenguaje, sino varios lenguajes, no es una sintaxis, sino varias y no es una manera totalmente nueva de trabajar, sino una manera más refinada que permitirá que todas las anteriores se puedan comunicar entre si sin problemas, ya que los datos cobran sentido. Todo esto lo veremos con calma en la Introducción a XML.

XML es interesante en el mundo de Internet y el e-bussiness, ya que existen muchos sistemas distintos que tienen que comunicarse entre si, pero como se ha podido imaginar, interesa por igual a todas las ramas de la informática y el tratamiento de datos, ya que permite muchos avances a la hora de trabajar con ellos.

En la introducción a XML, a lo largo de los siguientes capítulos, vamos a ver algunas características importantes de la tecnología que nos permitirán comprender mejor el mundo XML y cómo soluciona nuestros problemas a la hora de trabajar con los datos.

Historia del XML

El XML proviene de un lenguaje que inventó IBM allá por los años 70. El lenguaje de IBM se llama GML (General Markup Language) y surgió por la necesidad que tenían en la empresa de almacenar grandes cantidades de información de temas diversos.

Imaginar por un momento la cantidad de documentación que generaría IBM sobre todas las áreas en las que trabajaba e investigaba, y la cantidad de información que habrá generado hasta hoy. Así pues, necesitaban una manera de guardar la información y los expertos de IBM se inventaron GML, un lenguaje con el que poder clasificarlo todo y escribir cualquier documento para que se pueda luego procesar adecuadamente.

Este lenguaje gustó mucho a la gente de ISO, una entidad que se encarga de normalizar cuantas cosas podáis imaginar para los procesos del mundo actual, de modo que allá por el 86 trabajaron para normalizar el lenguaje, creando el SGML, que no era más que el GML pero estándar (Standar en inglés).

SGML es un lenguaje muy trabajado, capaz de adaptarse a un gran abanico de problemas y a partir de él se han creado los siguientes sistemas para almacenar información.

Por el año 89, para el ámbito de la red Internet, un usuario que había conocido el lenguaje de etiquetas (Markup) y los hiperenlaces creo un nuevo lenguaje llamado HTML, que fue utilizado para un nuevo servicio de Internet, la Web. Este lenguaje fue adoptado rápidamente por la comunidad y varias organizaciones comerciales crearon sus propios visores de HTML y riñeron entre ellos para hacer el visor más avanzado, inventándose etiquetas como su propia voluntad les decía. Desde el 96 hasta hoy una entidad llamada W3C ha tratado de poner orden en el HTML y establecer sus reglas y etiquetas para que

sea un estándar. Sin embargo el HTML creció de una manera descontrolada y no cumplió todos los problemas que planteaba la sociedad global de Internet.

El mismo W3C en el 98 empezó y continúa, en el desarrollo de XML (Extended Markup Language). En este lenguaje se ha pensado mucho más y muchas personas con grandes conocimientos en la materia están trabajando todavía en su gestación. Pretendían solucionar los carencias del HTML en lo que se respecta al tratamiento de la información. Problemas del HTML como:

- El contenido se mezcla con los estilos que se le quieren aplicar.
- No permite compartir información con todos los dispositivos, como pueden ser ordenadores o teléfonos móviles.
- La presentación en pantalla depende del visor que se utilice.

Imagínese, una persona que conoce el HTML y lo difícil que puede llegar a ser entender su código, que tuviese que procesarlo para extraer datos que necesite en otras aplicaciones. Sería muy difícil saber dónde está realmente la información que busca, siempre mezclada entre etiquetas <FONT>, <TABLE>, <TD>, etc... Esto es una mala gestión de la información y el XML la soluciona.

Sintaxis del XML

Dicen que el XML es un 10% del SGML y de verdad lo es, porque en realidad las normas que tiene son muy simples. Se escribe en un documento de texto ASCII, igual que el HTML y en la cabecera del documento se tiene que poner el texto

```
<?xml version="1.0"?>
```

En el resto del documento se deben escribir etiquetas como las de HTML, las etiquetas que nosotros queramos, por eso el lenguaje se llama XML, lenguaje de etiquetas extendido. Las etiquetas se escriben anidas, unas dentro de otras.

```
<ETIQ1>...<ETIQ2>...</ETIQ2>...</ETIQ1>
```

Cualquier etiqueta puede tener atributos. Le podemos poner los atributos que queramos.

```
<ETIQ atributo1="valor1" atributo2="valor2"...>
```

Los comentarios de XML se escriben igual que los de HTML.

```
<!-- Comentario -->
```

Y esto es todo lo que es el lenguaje XML en si, aunque tenemos que tener en cuenta que el XML tiene muchos otros lenguajes y tecnologías trabajando alrededor de él. Sin embargo, no cabe duda que la sintaxis XML es realmente reducida y sencilla.

Para definir qué etiquetas y atributos debemos utilizar al escribir en XML tenemos que fijarnos en la manera de guardar la información de una forma estructurada y ordenada. Por ejemplo, si deseamos guardar la información relacionada con una película en un documento XML podríamos utilizar un esquema con las siguientes etiquetas.

```
<?xml version="1.0"?>
<PELICULA nombre="El Padrino" año=1985>
<PERSONAL>
</DIRECTOR nombre="Georgie Lucar">
</INTERPRETE nombre="Marlon Brando" interpreta-a="Don Corleone">
</INTERPRETE nombre="Al Pacino" interpreta-a="Michael Corleone">
</PERSONAL>
</ARGUMENTO descripción="Pelicula de mafias sicilianas en Estados Unidos">
</PELICULA>
```

Como podéis ver, nos hemos inventado las etiquetas que nos venían en gana para poner este ejemplo y las hemos anidado de manera que la etiqueta más grande es la PELICULA y dentro de ella tenemos el PRESONAL y el ARGUMENTO. A su vez, dentro de PERSONAL tenemos tanto al DIRECTOR como a los actores (INTERPRETE).

Diferencias entre HTML y XML

Para los que conozcan también el lenguaje HTML, que espero que seáis muchos, he compilado aquí una serie de diferencias entre HTML y XML que sirven de muestra para ver hasta dónde llegan estos dos lenguajes.

El HTML se preocupa por formatear datos y para ello son las etiquetas que tiene el lenguaje, para formatear la información que se desea mostrar.

El XML se preocupa por estructurar la información que pretende almacenar. La estructura la marca la lógica propia de la información.

El desarrollo del HTML estuvo marcado la competencia entre los distintos visores del mercado. Cada uno quería ser el mejor e inventaba etiquetas nuevas que a la larga entraban a formar parte del estándar del W3C, como la etiqueta <FRAME>.

El desarrollo del XML está siendo llevado a cabo con rigor, siempre ajustado a lo que marca el estándar que desarrolla el W3C, entidad que está desarrollando el XML con más diligencia que las empresas con intereses particulares.

Procesar la información en HTML es inviable, por estar mezclada con los estilos y las etiquetas que formatean la información.

En XML se puede procesar la información con mucha facilidad, porque todo está ordenado de una manera lógica, así mismo el formateo de la información para que se pueda entender bien por el usuario es viable a través de un pequeño procesamiento, a través de hojas de estilos o similares.

Objetivos y usos del XML

Objetivos y usos del XML El XML se creó para que cumpliera varios objetivos.

- Que fuera idéntico a la hora de servir, recibir y procesar la información que el HTML, para aprovechar toda la tecnología implantada para este último.
- Que fuera formal y conciso desde el punto de vista de los datos y la manera de guardarlos.
- Que fuera extensible, para que lo puedan utilizar en todos los campos del conocimiento.
- Que fuese fácil de leer y editar.
- Que fuese fácil de implantar, programar y aplicar a los distintos sistemas.

El XML se puede usar para infinidad de trabajos y aporta muchas ventajas en amplios escenarios. Veamos algunas ventajas del XML en algunos campos prácticos.

- Comunicación de datos. Si la información se transfiere en XML, cualquier aplicación podría escribir un documento de texto plano con los datos que estaba manejando en formato XML y otra aplicación recibir esta información y trabajar con ella.
- Migración de datos. Si tenemos que mover los datos de una base de datos a otra sería muy sencillo si las dos trabajasen en formato XML.
- Aplicaciones web. Hasta ahora cada navegador interpreta la información a su manera y los programadores del web tenemos que hacer unas cosas u otras en función del navegador del usuario. Con XML tenemos una sola aplicación que maneja los datos y para cada navegador o soporte podremos tener una hoja de estilo o similar para aplicarle el estilo adecuado. Si mañana nuestra aplicación debe correr en WAP solo tenemos que crear una nueva hoja de estilo o similar.

Son sólo unos ejemplos que esperamos que comprendas aunque sea por encima ya que todavía hay muchas cosas que no sabes sobre XML y las tecnologías relacionadas.

Tecnologías relacionadas con XML

Hemos visto lo sencillo que es XML y las pocas normas que tenemos para su sintaxis. Simplemente utilizamos las etiquetas que necesitamos, abriendo y cerrando cada epígrafe de manera parecida a como lo hacemos en HTML.

Toda esta sencillez es gracias a que XML tiene muchas otras tecnologías relacionadas que son las encargadas de manejar importantes procesos dentro del ámbito de una aplicación XML. La sintaxis, la manera de aplicar estilos, programar o acceder a bases de datos, va por su parte, es decir, son tecnologías relacionadas con el XML.

En siguientes capítulos vamos a ver un montón de nuevas tecnologías, cada una para llevar a cabo un aspecto de la aplicación XML. En esta introducción a XML no pretendemos entrar mucho en la discusión de cada tecnología sino presentarlas y conocer sus usos.

Veremos tecnologías relacionadas con los procesos de:

- Contenidos: DTD o XML Schema.
- Diseño: CSS o XSL.
- Programación: SAX o DOM.

Contenidos: DTD o XML Schema

Un documento XML puede contener muchos tipos de información. Es decir, pueden haber muchos lenguajes escritos en XML para cualquier colectivo de usuarios. Por ejemplo,

- Si lo utiliza el colectivo de médicos podría crear un lenguaje en XML específico para almacenar diagnósticos de los pacientes. Este lenguaje se podría llama PacientesML.
- Si los distribuidores de películas utilizan XML podrán crear sus propios lenguajes para guardar la información de las películas. Este lenguaje se podría llamar PeliculasML.
- Si estamos escribiendo aplicaciones para móviles podremos utilizar un lenguaje para aplicaciones inalámbricas (Wireless), que se llama WML.

Como vemos, se pueden crear infinitos lenguajes a partir del XML. Para especificar cada uno de los usos de XML, o lo que es lo mismo, para especificar cada uno de los sublenguajes que podemos crear a partir de XML, se utilizan unos lenguajes propios.

Son unos lenguajes que sirven para definir otros lenguajes, es decir, son metalenguajes. Los definen especificando qué etiquetas podemos o debemos encontrarnos en los documentos HTML, en qué orden, dentro de qué otras, además de especificar los atributos que pueden o deben tener cada una de las etiquetas.

Hay dos metalenguajes con los que definir los lenguajes que podemos obtener a partir de XML, el DTD y el XML Schema.

El DTD, Definition Type Document, tiene una sintaxis especial, distinta de la de XML, que es sencilla, aunque un poco rara si nunca hemos visto un documento similar.

Para evitar el DTD, que tiene una sintaxis muy especial, se intentó encontrar una manera de escribir en XML la definición de otro lenguaje XML. Se definió entonces el lenguaje XML Schema y funciona bien, aunque puede llegar a ser un poco más complicado que especificarlo en DTD. Simplemente nos ahorramos de aprender un nuevo lenguaje con su sintaxis particular.

Un detalle importante de señalar a la hora de hablar de los DTD o XML Schema es que estos lenguajes también permiten comprobar la integridad de los datos en cualquier momento. Se calcula que un 70% de las líneas de código que escribe un programador están orientadas a comprobar la integridad de los datos, es decir, comprobar si donde se supone que hay un número efectivamente lo hay, si el número es entero o cualquier otra comprobación. Nuestros metalenguajes de XML nos sirven para tomar un documento XML y comprobar que los datos que él incluye son válidos, comprobando si lo que tenemos en el XML concuerda con lo que tendríamos que tener. Eso lo podemos hacer al leer el documento, si no son validos se saca un mensaje de error y se detiene el proceso del documento. Si son válidos hacemos lo que toque sin tener que preocuparnos por la integridad de los datos.

Diseño: CSS o XSL

Para cada documento XML que se desee presentar en pantalla formateado de la manera que deseemos se tiene que escribir una hoja de estilos o similar. Hemos utilizado esa frase en otras partes de la introducción a XML, veamos ahora que significa.

También tenemos dos posibles lenguajes con los que formatear los textos de un documento XML para poder verlo por pantalla. La primera posibilidad es el CSS, que muchos ya conocerán. La segunda opción es el XSL, bastante más avanzada.

CSS (Cascading Style Sheets o hojas de estilo en cascada)no es nada nuevo, ya se podía utilizar con

HTML y se creó en un intento de separar la forma del contenido en HTML. En XML también podemos utilizar las CSS, y se utilizan de una manera muy similar a cómo se utilizan en HTML, por lo menos los atributos de estilo que podemos aplicar son los mismos y sus posibles valores también.

XSL, que son las siglas de XML Style Language, es el segundo lenguaje con el que trabajar en XML. Este lenguaje no se limita a definir qué estilo aplicar a cada elemento del documento XML. Además se pueden realizar pequeñas instrucciones típicas de los lenguajes de programación y la salida no tiene porque ser un documento HTML, sino que además podría ser de otros tipos, cualquiera que podamos necesitar como un documento escrito en WML (para WAP), un documento de texto plano u otro documento XML.

XSL resulta mucho más potente que CSS y de hecho es mucho más adecuado utilizarlo. Una de sus principales ventajas la vemos a continuación. Si tenemos un documento XML que queremos que se visualice en múltiples dispositivos distintos será imprescindible utilizar XSL. En este esquema tendríamos un solo documento XML y un documento XSL para cada dispositivo que queramos incluir, por ejemplo para un navegador Netscape, otro para Internet Explorer, otro para un móvil Ericson y otro para un móvil Nokia. Si mañana aparece un nuevo dispositivo, por muy particular que sea, sólo necesitaremos crear un documento XSL para que nuestros XML se puedan visualizar en él.

Programación: SAX o DOM

Si queremos realizar acciones con nuestros datos escritos en XML tenemos también mucho camino ya implementado. El W3C ha especificado dos mecanismos para acceder a documentos XML y trabajar con ellos. Se tratan simplemente de unas normas que indican a los desarrolladores la manera de acceder a los documentos. Estas normas incluyen una jerarquía de objetos que tienen unos métodos y atributos con los que tendremos que trabajar y que nos simplificarán las tareas relativas al recorrido y acceso a las partes del documento.

Estos dos mecanismos se denominan **SAX y DOM**. SAX se utiliza para hacer un recorrido secuencial de los elementos del documento XLM y DOM implica la creación de un árbol en memoria que contiene el documento XML, y con él en memoria podemos hacer cualquier tipo de recorrido y acciones con los elementos que queramos.

Se puede programar con el lenguaje de programación que se desee para acceder a un documento XML. Los creadores del lenguaje son los responsables de crear unas API que cumplan las especificaciones de XML para que luego los desarrolladores de cada lenguaje las encuentren y puedan trabajar con ellas. Un lenguaje típico para trabajar con XML es Java y en este caso es SUN Microsystems la encargada de proveer el API que ha especificado el W3C y por lo tanto,.los desarrolladores en Java cuentan con unas clases especiales que ha creado SUN para programar con XML.

Por su parte, los creadores de algunos lenguajes han implementado una tercera manera de programar con XML que se llama XSLT. Empresas como por ejemplo la organización Apache, SUN o Microsoft, ya la están apoyando, aunque en el W3C no han dicho que sea un estándar. Es importante señalar que la W3C es un organismo muy lento y que mucho de lo que se hace en XML actualmente sólo está en la W3C contemplado como una "nota" en la que los gurús están pensando.

El trabajo con bases de datos y XML se está desarrollando con un lenguaje que se llama XQL (XML Query Language), que es uno de los ejemplos de lenguaje que sólo está publicado en el W3C como una "nota".

Bibliografía de XML

Hasta aquí llega nuestra breve introducción a XML. Como se ha podido ver, XML es muy ámplio y para tratar bien el tema será necesario contar con muchas más ayudas.

Nuestro objetivo era ofrecer una buena introducción para que aquellos que no conocían el mundo XML puedan ahora entender un poco en qué consiste y qué ventajas trae con él.

Ahora proponemos un par de libros de XML que pueden servir para que las personas que han encontrado el tema interesante puedan continuar con su estudio.

- "XML al descubierto" *Michael Morrison* Ed. Prentice Hall, 2000
- "Java y XML" *Brett McLaughlin* Ed. Anaya Multimedia, 2001

Estos libros han sido leídos y recomendados a nosotros por personas expertas en la materia. Hay otros muchos libros, pero hay que tener cuidado con lo que compramos acerca de XML pues hay posibilidades de adquirir libros que no nos ayudarán en nuestro aprendizaje, libros incomestibles, hechos por marcianos, o libros que no se ajustan a la filosofía que trae consigo el XML.

También recomendamos aquí pasarse por la categoría XML de nuestro buscador, donde se podrán encontrar enlaces que pueden completar también esta información.

Capitulo 16
Tutorial de SQL

Aprende a utilizar el estándar utilizado para la consulta de bases de datos. Seleccionar, crear, modificar y borrar registros. Todo lo que necesitas para la creación de tus páginas dinámicas.

Qué es SQL

Las aplicaciones en red son cada día más numerosas y versátiles. En muchos casos, el esquema básico de operación es una serie de scripts que rigen el comportamiento de una base de datos.

Debido a la diversidad de lenguajes y de bases de datos existentes, la manera de comunicar entre unos y otras sería realmente complicada a gestionar de no ser por la existencia de estándares que nos permiten el realizar las operaciones básicas de una forma universal.

Es de eso de lo que trata el Structured Query Language que no es mas que un lenguaje estándar de comunicación con bases de datos. Hablamos por tanto de un lenguaje normalizado que nos permite trabajar con cualquier tipo de lenguaje (ASP o PHP) en combinación con cualquier tipo de base de datos (MS Access, SQL Server, MySQL...).

El hecho de que sea estándar no quiere decir que sea idéntico para cada base de datos. En efecto, determinadas bases de datos implementan funciones específicas que no tienen necesariamente que funcionar en otras.

Aparte de esta universalidad, el SQL posee otras dos características muy apreciadas. Por una parte, presenta una potencia y versatilidad notables que contrasta, por otra, con su accesibilidad de aprendizaje.

El manual de SQL de desarrolloweb pretende dar a conocer las operaciones básicas que se pueden realizar con SQL y que tienen una aplicación directa con la creación de aplicaciones en red sin profundizar más de lo estrictamente necesario. Buscamos con ello ofrecer al webmaster un manual de referencia práctico y aplicado.

Tipos de campo

Como sabemos una base de datos esta compuesta de tablas donde almacenamos registros catalogados en función de distintos campos (características).

Un aspecto previo a considerar es la naturaleza de los valores que introducimos en esos campos. Dado que una base de datos trabaja con todo tipo de informaciones, es importante especificarle qué tipo de valor le estamos introduciendo de manera a, por un lado, facilitar la búsqueda posteriormente y por otro, optimizar los recursos de memoria.

Cada base de datos introduce tipos de valores de campo que no necesariamente están presentes en otras. Sin embargo, existe un conjunto de tipos que están representados en la totalidad de estas bases. Estos tipos comunes son los siguientes:

Alfanuméricos	Contienen cifras y letras. Presentan una longitud limitada (255 caracteres)
Numéricos	Existen de varios tipos, principalmente, enteros (sin decimales) y reales (con decimales).
Booleanos	Poseen dos formas: Verdadero y falso (Sí o No)
Fechas	Almacenan fechas facilitando posteriormente su explotación. Almacenar fechas de esta forma posibilita ordenar los registros por fechas o calcular los días entre una fecha y otra...
Memos	Son campos alfanuméricos de longitud ilimitada. Presentan el inconveniente de no poder ser indexados (veremos más adelante lo que esto quiere decir).
Autoincrementables	Son campos numéricos enteros que incrementan en una unidad su valor para cada registro incorporado. Su utilidad resulta más que evidente: Servir de identificador ya que resultan exclusivos de un registro.

Añadir un nuevo registro

Los registros pueden ser introducidos a partir de sentencias que emplean la instrucción Insert.

La sintaxis utilizada es la siguiente:

Insert Into nombre_tabla (nombre_campo1, nombre_campo2,...) Values (valor_campo1, valor_campo2...)

Un ejemplo sencillo a partir de nuestra tabla modelo es la introducción de un nuevo cliente lo cual se haría con una instrucción de este tipo:

Insert Into clientes (nombre, apellidos, direccion, poblacion, codigopostal, email, pedidos) Values ('Perico', 'Palotes', 'Percebe n°13', 'Lepe', '123456', 'perico@desarrolloweb.com', 33)

Como puede verse, los campos no numéricos o booleanos van delimitados por apostrofes: '. También resulta interesante ver que el código postal lo hemos guardado como un campo no numérico. Esto es debido a que en determinados paises (Inglaterra,como no) los codigos postales contienen también letras.

Por supuesto, no es imprescindible rellenar todos los campos del registro. Eso sí, puede ser que determinados campos sean necesarios. Estos campos necesarios pueden ser definidos cuando construimos nuestra tabla mediante la base de datos.

Resulta muy interesante, ya veremos más adelante el por qué, el introducir durante la creación de nuestra tabla un campo autoincrementable que nos permita asignar un único número a cada uno de los registros. De este modo, nuestra tabla clientes presentaría para cada registro un número exclusivo del cliente el cual nos será muy util cuando consultemos varias tablas simultáneamente.

Borrar un registro

Para borrar un registro nos servimos de la instrucción Delete. En este caso debemos especificar cual o cuales son los registros que queremos borrar. Es por ello necesario establecer una selección que se llevara a cabo mediante la cláusula Where.

La forma de seleccionar se verá detalladamente en capítulos posteriores. Por ahora nos contentaremos de mostrar cuál es el tipo de sintaxis utilizado para efectuar estas supresiones:

Delete From nombre_tabla Where condiciones_de_selección

Si queremos por ejemplo borrar todos los registros de los clientes que se llamen Perico lo haríamos del siguiente modo:

Delete From clientes Where nombre='Perico'

Hay que tener cuidado con esta instrucción ya que si no especificamos una condición con Where, lo que estamos haciendo es **borrar toda la tabla**:

Delete From clientes

Actualizar un registro

Update es la instrucción que nos sirve para modificar nuestros registros. Como para el caso de Delete, necesitamos especificar por medio de Where cuáles son los registros en los que queremos hacer efectivas nuestras modificaciones. Además, obviamente, tendremos que especificar cuáles son los nuevos valores de los campos que deseamos actualizar. La sintaxis es de este tipo:

Update nombre_tabla Set nombre_campo1 = valor_campo1, nombre_campo2 = valor_campo2,... Where condiciones_de_selección

Un ejemplo aplicado:

Update clientes Set nombre='José' Where nombre='Pepe'

Mediante esta sentencia cambiamos el nombre Pepe por el de José en todos los registros cuyo nombre sea Pepe.

Aquí también hay que ser cuidadoso de no olvidarse de usar Where, de lo contrario, modificaríamos todos los registros de nuestra tabla.

Selección de tablas I

La selección total o parcial de una tabla se lleva a cabo mediante la instrucción Select. En dicha selección hay que especificar:

-Los campos que queremos seleccionar
-La tabla en la que hacemos la selección

En nuestra tabla modelo de clientes podríamos hacer por ejemplo una selección del nombre y dirección de los clientes con una instrucción de este tipo:

Select nombre, dirección From clientes

Si quisiésemos seleccionar todos los campos, es decir, **toda la tabla**, podríamos utilizar el comodín * del siguiente modo:

Select * From clientes

Resulta también muy útil el filtrar los registros mediante condiciones que vienen expresadas después de la **cláusula Where**. Si quisiésemos mostrar los clientes de una determinada ciudad usaríamos una expresión como esta:

Select * From clientes Where poblacion Like 'Madrid'

Además, podríamos **ordenar los resultados** en función de uno o varios de sus campos. Para este ultimo ejemplo los podríamos ordenar por nombre así:

Select * From clientes Where poblacion Like 'Madrid' **Order** By nombre

Teniendo en cuenta que puede haber más de un cliente con el mismo nombre, podríamos dar un segundo criterio que podría ser el apellido:

Select * From clientes Where poblacion Like 'Madrid' Order By nombre, apellido

Si invirtiésemos el orden « nombre,apellido » por « apellido, nombre », el resultado sería distinto. Tendríamos los clientes ordenados por apellido y aquellos que tuviesen apellidos idénticos se subclasificarían por el nombre.

Es posible también **clasificar por orden inverso**. Si por ejemplo quisiésemos ver nuestros clientes por orden de pedidos realizados teniendo a los mayores en primer lugar escribiríamos algo así:

Select * From clientes Order By pedidos **Desc**

Una opción interesante es la de efectuar **selecciones sin coincidencia**. Si por ejemplo buscásemos el saber en qué ciudades se encuentran nuestros clientes sin necesidad de que para ello aparezca varias veces la misma ciudad usaríamos una sentencia de esta clase:

Select **Distinct** poblacion From clientes Order By poblacion

Así evitaríamos ver repetido Madrid tantas veces como clientes tengamos en esa población.

Selección de tablas II

Hemos querido compilar a modo de tabla ciertos operadores que pueden resultar útiles en determinados casos. Estos operadores serán utilizados después de la cláusula Where y pueden ser **combinados hábilmente mediante paréntesis** para optimizar nuestra selección a muy altos niveles.

Operadores matemáticos:	
>	Mayor que
<	Menor que
>=	Mayor o igual que
<=	Menor o igual que
<>	Distinto
=	Igual

Operadores lógicos
And
Or
Not

Otros operadores	
Like	Selecciona los registros cuyo valor de campo se asemeje, no teniendo en cuenta mayúsculas y minúsculas.
In y Not In	Da un conjunto de valores para un campo para los cuales la condición de selección es (o no) valida
Is Null y Is Not Null	Selecciona aquellos registros donde el campo especificado esta (o no) vacío.
Between...And	Selecciona los registros comprendidos en un intervalo
Distinct	Selecciona los registros no coincidentes
Desc	Clasifica los registros por orden inverso

Comodines	
*	Sustituye a todos los campos
%	Sustituye a cualquier cosa o nada dentro de una cadena
_	Sustituye un solo carácter dentro de una cadena

Veamos a continuación aplicaciones practicas de estos operadores.

En esta sentencia seleccionamos todos los clientes de Madrid cuyo nombre no es Pepe. Como puede verse, empleamos **Like** en lugar de **=** simplemente para evitar inconvenientes debido al empleo o no de mayúsculas.

Select * From clientes Where poblacion **Like** 'madrid' **And Not** nombre **Like** 'Pepe'

Si quisiéramos recoger en una selección a los clientes de nuestra tabla cuyo **apellido comienza por A y cuyo número de pedidos esta comprendido entre 20 y 40**:

Select * From clientes Where apellidos **like 'A%'** And pedidos **Between 20 And 40**

El operador **In**, lo veremos más adelante, es muy práctico para consultas en varias tablas. Para casos en una sola tabla es empleado del siguiente modo:

Select * From clientes Where poblacion **In ('Madrid','Barcelona','Valencia')**

De esta forma **seleccionamos aquellos clientes que vivan en esas tres ciudades**.

Selección de tablas III

Una base de datos puede ser considerada como un conjunto de tablas. Estas tablas en muchos casos están relacionadas entre ellas y se complementan unas con otras.

Refiriéndonos a nuestro clásico ejemplo de una base de datos para una aplicación de e-comercio, la tabla clientes de la que hemos estado hablando puede estar perfectamente coordinada con una tabla donde almacenamos los pedidos realizados por cada cliente. Esta tabla de pedidos puede a su vez estar conectada con una tabla donde almacenamos los datos correspondientes a cada artículo del inventario.

De este modo podríamos fácilmente obtener informaciones contenidas en esas tres tablas como puede ser la designación del artículo más popular en una determinada región donde la designación del artículo sería obtenida de la tabla de artículos, la popularidad (cantidad de veces que ese artículo ha sido vendido) vendría de la tabla de pedidos y la región estaría comprendida obviamente en la tabla clientes.

Este tipo de organización basada en múltiples tablas conectadas nos permite trabajar con tablas mucho más manejables a la vez que nos evita copiar el mismo campo en varios sitios ya que podemos acceder a él a partir de una simple llamada a la tabla que lo contiene.

En este capítulo veremos como, sirviéndonos de lo aprendido hasta ahora, podemos realizar fácilmente selecciones sobre varias tablas. Definamos antes de nada las diferentes tablas y campos que vamos a utilizar en nuestros ejemplos:

Tabla de clientes	
Nombre campo	**Tipo campo**
id_cliente	Numérico entero
nombre	Texto
apellidos	Texto
direccion	Texto
poblacion	Texto
codigopostal	Texto
telefono	Numérico entero
email	Texto

Tabla de pedidos	
Nombre campo	**Tipo campo**
id_pedido	Numérico entero
id_cliente	Numérico entero
id_articulo	Numérico entero
fecha	Fecha
cantidad	Numérico entero

Tabla de artículos	
Nombre campo	**Tipo campo**
id_articulo	Numérico entero
titulo	Alfanumérico
autor	Alfanumérico
editorial	Alfanumérico
precio	Numérico real

Estas tablas pueden ser utilizadas simultáneamente para extraer informaciones de todo tipo. Supongamos que queremos enviar un mailing a todos aquellos que hayan realizado un pedido ese mismo día. Podríamos escribir algo así:

Select clientes.apellidos, clientes.email From clientes,pedidos Where pedidos.fecha like '25/02/00' And pedidos.id_cliente= clientes.id_cliente

Como puede verse esta vez, después de la cláusula From, introducimos el nombre de las dos tablas de donde sacamos las informaciones. Además, el nombre de cada campo va precedido de la tabla de provenencia separados ambos por un punto. En los campos que poseen un nombre que solo aparece en una de las tablas, no es necesario especificar su origen aunque a la hora de leer la sentencia puede resultar más claro el precisarlo. En este caso el campo fecha podría haber sido designado como "fecha" en lugar de "pedidos.fecha".

Veamos otro ejemplo más para consolidar estos nuevos conceptos. Esta vez queremos ver el título del libro correspondiente a cada uno de los pedidos realizados:

Select pedidos.id_pedido, articulos.titulo From pedidos, articulos Where pedidos.id_articulo=articulos.id_articulo

En realidad la filosofía continua siendo la misma que para la consulta de una única tabla.

Selección de tablas IV

Además de los criterios hasta ahora explicados para realizar las consultas en tablas, SQL permite también aplicar un conjunto de funciones predefinidas. Estas funciones, aunque básicas, pueden ayudarnos en algunos momentos a expresar nuestra selección de una manera más simple sin tener que recurrir a operaciones adicionales por parte del script que estemos ejecutando.

Algunas de estas funciones son representadas en la tabla siguiente :

Función	Descripción
Sum(campo)	Calcula la suma de los registros del campo especificado
Avg(Campo)	Calcula la media de los registros del campo especificado
Count(*)	Nos proporciona el valor del numero de registros que han sido seleccionados
Max(Campo)	Nos indica cual es el valor máximo del campo
Min(Campo)	Nos indica cual es el valor mínimo del campo

Dado que el campo de la función no existe en la base de datos, sino que lo estamos generando virtualmente, esto puede crear inconvenientes cuando estamos trabajando con nuestros scripts a la hora de tratar su valor y su nombre de campo. Es por ello que el valor de la **función ha de ser recuperada a partir de un alias** que nosotros especificaremos en la sentencia SQL a partir de la instrucción **AS**. La cosa podría quedar así:

Select Sum(total) As suma_pedidos From pedidos

A partir de esta sentencia calculamos la suma de los valores de todos los pedidos realizados y almacenamos ese valor en un campo virtual llamado suma_pedidos que podrá ser utilizado como cualquier otro campo por nuestras paginas dinámicas.

Por supuesto, todo lo visto hasta ahora puede ser aplicado en este tipo de funciones de modo que, por ejemplo, podemos establecer condiciones con la cláusula Where construyendo sentencias como esta:

Select Sum(cantidad) as suma_articulos From pedidos Where id_articulo=6

Esto nos proporcionaría la cantidad de **ejemplares de un determinado libro que han sido vendidos**.

Otra propiedad interesante de estas funciones es que **permiten realizar operaciones con varios campos dentro de un mismo paréntesis**:

Select Avg(total/cantidad) From pedidos

Esta sentencia da como resultado el **precio medio al que se están vendiendo los libros**. Este resultado no tiene por qué coincidir con el del **precio medio de los libros presentes en el inventario**, ya que, puede ser que la gente tenga tendencia a comprar los libros caros o los baratos:

Select Avg(precio) as precio_venta From articulos

Una cláusula interesante en el uso de funciones es Group By. Esta cláusula nos permite agrupar registros a los cuales vamos a aplicar la función. Podemos por ejemplo calcular el **dinero gastado por cada cliente**:

Select id_cliente, Sum(total) as suma_pedidos From pedidos Group By id_cliente

O saber el **numero de pedidos que han realizado**:

Select id_cliente, Count(*) as numero_pedidos From pedidos Group By id_cliente

Las posibilidades como vemos son numerosas y pueden resultar prácticas. Todo queda ahora a disposición de nuestras ocurrencias e imaginación.

Optimizar prestaciones I

Las bases de datos (BD) cuanto más extensas requieren una mayor atención a la hora de organizar sus contenidos. Cuando se trabaja con tablas de miles o decenas de miles de registros la búsqueda de un determinado dato puede resultar un proceso largo que ralentiza enormemente la creación de nuestra página.

Es por ello importante tener en cuenta una serie de aspectos indispensables para el mejor funcionanmiento de la base.

Gestión y elección de los índices

Los índices son campos elegidos arbitrariamente por el constructor de la BD que permiten la búsqueda a partir de dicho campo a una velocidad notablemente superior. Sin embargo, esta ventaja se ve contrarrestada por el hecho de ocupar mucha más memoria (el doble más o menos) y de requerir para su inserción y actualización un tiempo de proceso superior.

Evidentemente, **no podemos indexar todos los campos** de una tabla extensa ya que doblamos el tamaño de la BD. Igualmente, tampoco sirve de mucho el indexar todos los campos en una tabla pequeña ya que las selecciones pueden efectuarse rápidamente de todos modos.

Un caso en el que los índices pueden resultar muy útiles es cuando realizamos peticiones simultáneas sobre varias tablas. En este caso, el proceso de selección puede acelerarse sensiblemente si **indexamos los campos que sirven de nexo entre las dos tablas**. En el ejemplo de nuestra librería virtual estos campos serían id_cliente e id_articulo.

Los índices pueden resultar contraproducentes si los introducimos sobre campos triviales a partir de los cuales no se realiza ningún tipo de petición ya que, además del problema de memoria ya mencionado, estamos ralentizando otras tareas de la base de datos como son la edición, inserción y borrado. Es por ello que vale la pena pensarselo dos veces antes de indexar un campo que no sirve de criterio para búsquedas de los internautas y que es usado con muy poca frecuencia por razones de mantenimiento.

Gestión de los nexos entre tablas

El enlace entre tablas es uno de los puntos más peliagudos y que puede llevar a la absoluta ralentización de la base de datos a causa "pequeños" detalles que resultan ser fatales.

Imaginemos que trabajamos con una pequeña BD constituida por dos tablas de 1000 registros cada una. Imaginemos ahora una selección simultánea en la que imponemos la condición de que el valor un campo de la primera sea igual a de una segunda, algo que se realiza con mucha frecuencia. En este tipo de casos, la BD leerá y comparará cada valor de campo de una con cada valor de campo de la otra. Esto representaría un millón de lecturas. Este hecho podría agravarse si consultamos una tercera tabla al mismo tiempo y podría llegar a ser catastrófico si tenemos en cuenta que la BD esta siendo consultada por varios internautas al mismo tiempo.

Para evitar situaciones de colapso, es necesario **indexar cada uno de los campos que sirven de enlace entre esas tablas**. En el ejemplo de nuestra librería virtual, ya lo hemos dicho, estos campos serían id_cliente e id_articulo. Además, resulta también de vital importancia el **definir esos campos de una forma estrictamente idéntica en cada una de las tablas**, es decir, el campo ha de ser de la misma naturaleza y características. No vale definirlo como real en una tabla y entero en otra o cambiar la longitud máxima para los alfanuméricos o que en una tabla sea de longitud constante y en otra variable...

El gestionar inteligentemente estos aspectos puede solucionarnos muchos quebraderos de cabeza y permitir a los internautas navegar más agradablemente por nuestro sitio.

Optimizar prestaciones II

Gestion de los campos

Ya hemos comentado por encima los diferentes tipos de campo existentes en una base de datos. La elección del tipo de campo apropiado para cada caso puede ayudarnos también a optimizar el tamaño y rapidez de nuestra base de datos.

La preguntas que hay que hacerse a la hora de elegir la naturaleza y dimensiones del campo son:

-¿Qué tipo de dato voy a almacenar en el campo? Números, texto, fechas...

-¿Cuál es el tamaño máximo que espero que pueda alcanzar alguno de los registros del

campo?

Hay que tener en cuenta que cuanto más margen le demos al valor máximo del campo, más aumentará el tamaño de nuestra base de datos y más tiempo tardara en realizar las consultas. Además, el factor tamaño puede verse agravado si estamos definiendo un campo indexado, para los cuales, el espacio ocupado es aproximadamente del doble.

Un consejo práctico es que las fechas sean almacenadas en formato de fecha ya que ello nos permite reducir el espacio que ocupan en memoria de más del doble y por otro lado, podremos aprovechar las prestaciones que SQL y nuestro lenguaje de servidor nos ofrecen. Podremos calcular la diferencia de días entre dos fechas, ordenar los registros por fecha, mostrar los registros comprendidos en un intervalo de tiempo...

Existe la posibilidad para los campos de texto de fijar una cierta longitud para el campo o dejar que cada registro tenga una longitud variable en función del número de carácteres que posea. Elegir campos de longitud variable nos puede ayudar a optimizar los recursos de memoria de la BD, no obstante, es un arma de doble filo ya que las consultas se realizan más lentamente puesto que obligamos a la tabla a establecer cuál es el tamaño de cada registro que se está comparando en lugar de saberlo de antemano. Es por tanto aconsejable, para los campos indexados de pequeño tamaño, atribuirles una longitud fija.

Algunos trucos prácticos

Eliminar llamadas a bases de datos

En páginas tipo portal en las que a los lados se encuentran enlaces que son impresos a partir de bases de datos (distintas secciones, servicios,...) existe siempre un efecto ralentizador debido a que se trata de páginas altamente visitadas que efectúan múltiples llamadas a BD sistemáticamente en cada una de sus páginas.

Una forma de agilizar la visualización de estas páginas es textualizando estos enlaces a partir de scripts internos. Pongamos el ejemplo de Desarrolloweb:

Como puede verse, a los lados hay secciones como "Vuestras páginas", "Cosecha del 2000", "Manuales" cuyos enlaces están almacenados en bases de datos. Sin embargo, los enlaces que se visualizan en la página no han sido obtenidos por llamadas a bases de datos sino que, cada vez que un nuevo elemento de la sección es añadido, esto actualiza automáticamente, por medio de un script, un archivo texto en el que el nuevo enlace es incluido y él más antiguo es eliminado. Es, de hecho, este archivo texto el que es insertado en el código fuente de la página. De este modo evitamos media docena de llamadas a bases de datos cada vez que una página es vista lo cual permite optimizar recursos de servidor de una manera significativa.

Eliminar palabras cortas y repeticiones

En situaciones en la que nuestra base de datos tiene que almacenar campos de texto extremadamente largos y dichos campos son requeridos para realizar selecciones del tipo LIKE '%algo%', los recursos de la BD pueden verse sensiblemente mermados. Una forma de ayudar a gestionar este tipo búsquedas es incluyendo un campo adicional.

Este campo adicional puede ser creado automáticamente por medio de scripts y en él incluiríamos el texto original, del cual habremos eliminado palabras triviales como artículos, preposiciones o posesivos. Nos encargaremos además de eliminar las palabras que estén repetidas. De esta forma podremos disminuir sensiblemente el tamaño del campo que va a ser realmente consultado.

Hemos comentado en otros capítulos que los campos texto de mas de 255 caracteres denominados memo no pueden ser indexados. Si aún después de esta primera filtración nuestro campo continua siendo demasiado largo para ser indexado, lo que se puede hacer es cortarlo en trozos de 255 caracteres de manera a que lo almacenemos en distintos campos que podrán ser indexados y por tanto consultados con mayor rapidez.

Creación de tablas

En general, la mayoría de las bases de datos poseen potentes editores de bases que permiten la creación rápida y sencilla de cualquier tipo de tabla con cualquier tipo de formato.

Sin embargo, una vez la base de datos está alojada en el servidor, puede darse el caso de que queramos introducir una nueva tabla ya sea con carácter temporal (para gestionar un carrito de compra por ejemplo) o bien permanente por necesidades concretas de nuestra aplicación.

En estos casos, podemos, a partir de una sentencia SQL, crear la tabla con el formato que deseemos lo cual nos puede ahorrar más de un quebradero de cabeza.

Este tipo de sentencias son especialmente útiles para bases de datos como Mysql, las cuales trabajan directamente con comandos SQL y no por medio de editores.

Para crear una tabla debemos especificar diversos datos: El nombre que le queremos asignar, los nombres de los campos y sus características. Además, puede ser necesario especificar cuáles de estos campos van a ser índices y de qué tipo van a serlo.

La sintaxis de creación puede variar ligeramente de una base de datos a otra ya que los tipos de campo aceptados no están completamente estandarizados.

A continuación os explicamos someramente la sintaxis de esta sentencia y os proponemos una serie de ejemplos prácticos:

Sintaxis

```
Create Table nombre_tabla
(
nombre_campo_1 tipo_1
nombre_campo_2 tipo_2
nombre_campo_n tipo_n
Key(campo_x,...)
)
```

Pongamos ahora como ejemplo la creación de la tabla pedidos que hemos empleado en capítulos previos:

```
Create Table pedidos
(
id_pedido INT(4) NOT NULL AUTO_INCREMENT,
id_cliente INT(4) NOT NULL,
id_articulo INT(4)NOT NULL,
fecha DATE,
cantidad INT(4),
total INT(4), KEY(id_pedido,id_cliente,id_articulo)
)
```

En este caso creamos los campos *id* los cuales son considerados de tipo entero de una longitud especificada por el número entre paréntesis. Para *id_pedido* requerimos que dicho campo se incremente automáticamente (AUTO_INCREMENT) de una unidad a cada introducción de un nuevo registro para, de esta forma, automatizar su creación. Por otra parte, para evitar un mensaje de error, es necesario requerir que los campos que van a ser definidos como índices no puedan ser nulos (NOT NULL).

El campo *fecha* es almacenado con formato de fecha (DATE) para permitir su correcta explotación a partir de las funciones previstas a tal efecto.

Finalmente, definimos los índices enumerándolos entre paréntesis precedidos de la palabra KEY o INDEX.

Del mismo modo podríamos crear la tabla de *artículos* con una sentencia como ésta:

```
Create Table articulos
(
id_articulo INT(4) NOT NULL AUTO_INCREMENT,
titulo VARCHAR(50),
autor VARCHAR(25),
editorial VARCHAR(25),
precio REAL,
KEY(id_articulo)
)
```

En este caso puede verse que los campos alfanuméricos son introducidos de la misma forma que los numéricos. Volvemos a recordar que en tablas que tienen campos comunes es de vital importancia definir estos campos de la misma forma para el buen funcionamiento de la base.

Muchas son las opciones que se ofrecen al generar tablas. No vamos a tratarlas detalladamente pues sale de lo estrictamente práctico. Tan sólo mostraremos algunos de los tipos de campos que pueden ser empleados en la creación de tablas con sus características:

Tipo	Bytes	Descripción

INT o INTEGER	4	Números enteros. Existen otros tipos de mayor o menor longitud específicos de cada base de datos.
DOUBLE o REAL	8	Números reales (grandes y con decimales). Permiten almacenar todo tipo de número no entero.
CHAR	1/caracter	Alfanuméricos de longitud fija predefinida
VARCHAR	1/caracter+1	Alfanuméricos de longitud variable
DATE	3	Fechas, existen multiples formatos específicos de cada base de datos
BLOB	1/caracter+2	Grandes textos no indexables
BIT o BOOLEAN	1	Almacenan un bit de información (verdadero o falso)

Apéndice I
Qué es cada tecnología

Este apendice trata en cada uno de sus capítulos de introducir cada una de las tecnologías utilizadas en el desarrollo de páginas web.

Qué es HTML

HTML es el lenguaje con el que se definen las páginas web. Básicamente se trata de un conjunto de etiquetas que sirven para definir la forma en la que presentar el texto y otros elementos de la página.

El HTML se creó en un principio con objetivos divulgativos. No se pensó que la web llegara a ser un área de ocio con carácter multimedia, de modo que, el HTML se creó sin dar respuesta a todos los posibles usos que se le iba a dar y a todos los colectivos de gente que lo utilizarían en un futuro. Sin embargo, pese a esta deficiente planificación, si que se han ido incorporando modificaciones con el tiempo, estos son los estándares del HTML. Numerosos estándares se han presentado ya. El HTML 4.01 es el último estándar a febrero de 2001.

El HTML es un lenguaje de programación muy fácil de aprender, lo que permite que cualquier persona, aunque no haya programado en la vida pueda enfrentarse a la tarea de crear una web. HTML es fácil y pronto podremos dominar el lenguaje. Más adelante se conseguirán los resultados profesionales gracias a nuestras capacidades para el diseño y nuestra vena artista.

Una vez conocemos el concepto de HTML os vamos a adelantar algunas cosas más. Este lenguaje se escribe en un documento de texto, por eso necesitamos un editor de textos para escribir una página web. Así pues, el archivo donde está contenido el código HTML es un archivo de texto, con una peculiaridad, que tiene extensión .html o .htm (es indiferente cuál utilizar). De modo que cuando programemos en HTML lo haremos con un editor de textos, lo más sencillo posible y guardaremos nuestros trabajos con extensión .html, por ejemplo mipagina.html

Por adelantar un poco cómo se utiliza el HTML os diremos que el lenguaje consta de etiquetas que tienen esta forma <B> o <P>. Cada etiqueta significa una cosa, por ejemplo <B> significa que se escriba en negrita (bold) o <P> significa un párrafo, <A> es un enlace, etc. Casi todas las etiquetas tienen su correspondiente etiqueta de cierre, que indica que a partir de ese punto no debe de afectar la etiqueta. Por ejemplo </B> se utiliza para indicar que se deje de escribir en negrita. Así que el HTML no es más que una serie de etiquetas que se utilizan para definir la forma o estilo que queremos aplicar a nuestro documento. <B>**Esto está en negrita**</B>.

Si lo que deseamos es tener una idea global de lo que es la publicación en Internet y los pasos a seguir para colocar nuestras páginas en la web lo más adecuado será consultar el capitulo de Publicar en Internet.

Qué es DHTML

A medida que vamos avanzando en la programación de páginas web nos vamos fijando nuevos objetivos para crear cada día webs más excitantes. Siguiendo este camino, **llega un momento que el lenguaje HTML se nos queda corto y tenemos que servirnos de alguna tecnología superior, que nos permita realizar esos desarrollos más complejos y dinámicos**.

Imaginaros por un momento que tuvieseis entre manos un gran proyecto, un proyecto que supusiese la creación masiva de páginas, como puede ser un periódico, donde cada día hay que cambiar los contenidos por completo, o una enciclopedia online, con miles de páginas y referencias, por poner dos ejemplos. Si utilizásemos únicamente HTML necesitaríamos un regimiento de maquetadores web para poder llevar a cabo el trabajo de crear tantas y tantas páginas y su actualización.

Así mismo, si quisiésemos desarrollar una aplicación en la web donde el usuario tuviese que interaccionar con la página, o una aplicación que ofreciese algún servicio, como un buscador o un gestor de correo a través de la web, también nos veríamos muy limitados con el HTML.

Además, también estamos muy limitados con el HTML a la hora de crear efectos en las páginas, animaciones que llamen un poco la atención del usuario y que permitan hacer que las páginas web sean más divertidas.

DHTML es lo que hace posible crear una páginas web que salven todas las limitaciones del HTML como las comentadas con anterioridad. Como vemos, el DHTML es muy amplio y **engloba**

muchas técnicas que se pueden realizas con multitud de lenguajes de programación y programas distintos .

Vamos a hacer una clasificación de DHTML para acotar un poco sus radios de acción y para que el concepto se acote en áreas de la programación web que podemos ya conocer.

DHTML de cliente

Por un lado tenemos el **DHTML que se desarrolla en el ámbito de una página web, cuando la página se está viendo en la pantalla de los usuarios** , es decir, en los navegadores. En estos casos, para realizar cualquier tipo de efecto o interactividad en la página tenemos como recurso al navegador, por eso se llama de cliente.

La programación en el cliente sirve para muchas cosas, ejemplos de ello son efectos diversos en las páginas, sonidos, videos, menús interactivos, control y respuesta a las acciones de un usuario en la página, control sobre los formularios, etc. Para hacer muchas de estas cosas podemos utilizar diversos lenguajes de programación como **Javascript** y **VBScript**, o incluso podemos meter aquí programas como **Flash**.

No obstante está más cercana a la idea del DHTML el programar scripts dentro de la página con los lenguajes del lado del cliente. Javascript para todos los navegadores y VBScript para Internet Explorer. Estos lenguajes trabajan, como se ha dicho, integrados con el navegador y dependen del modelo y de la versión de éste.

Estos lenguajes no permiten el desarrollo de cualquier proyecto en Internet, ya que al ser ejecutados en el navegador del cliente, no tienen acceso a todos los recursos del sistema del usuario, para evitar agujeros de seguridad, ni a los recursos del servidor donde están alojadas las páginas. Esta limitación, añadida a la ya comentada de su dependencia del navegador, los hacen insuficientes para desarrollos avanzados, siendo más bien un complemento de programación que el núcleo de verdaderas aplicaciones en el web.

DHTML de servidor

Por otro lado, existen una serie de **lenguajes que se basan en el servidor para ejecutar sus scripts**, al igual que la programación del cliente se basa en el navegador. **Cuando una página es solicitada por parte de un cliente, el servidor ejecuta los scripts y genera una página resultado, que envía al cliente. La página resultado contiene únicamente código HTML** , por lo que puede ser interpretada por cualquier navegador sin lugar a errores, independientemente de su versión.

Esta independencia del navegador ya es una ventaja significativa con respecto a la programación en el cliente, pero lo es aun más que contamos con todos los recursos del servidor donde están alojadas las páginas. Estos recursos, como podrían ser gestores de bases de datos, servidores de correo o el propio sistema de archivos del servidor, son los que nos van ha permitir construir todo tipo de aplicaciones.

Como ventajas adicionales se puede destacar que el código de las páginas con los scripts nunca llega al cliente, recordamos que al navegador sólo le llega HTML, y esto implica que nuestros visitantes nunca van a poder acceder al corazón de las aplicaciones que hayamos desarrollado, es decir, a los scripts del lado del servidor.

Lenguajes del lado del servidor son **ASP**, desarrollado por Microsoft, **PHP** de código libre, **JSP** para programar en Java, o alguna otra interfaz como **CGI**, que se desarrolla en lenguajes como **C** o **Perl**.

Para **tratar en extensión el tema del DHTML** tenemos un manual que ofrece mucha más información. Es el **capitulo de páginas dinámicas**.

Qué es CSS

CSS, es una tecnología que nos permite crear páginas web de una manera más exacta. Gracias a las CSS somos mucho más dueños de los resultados finales de la página, pudiendo hacer muchas cosas que no se podía hacer utilizando solamente HTML, como incluir márgenes, tipos de letra, fondos, colores...

CSS son las siglas de Cascading Style Sheets, en español Hojas de estilo en Cascada. En este reportaje

vamos a ver algunos de los efectos que se pueden crear con las CSS sin necesidad de conocer la tecnología entera.
Para empezar

Las Hojas de Estilo en Cascada se escriben dentro del código HTML de la página web, solo en casos avanzados se pueden escribir en un archivo a parte y enlazar la página con ese archivo. En un principio vamos a utilizar la manera más directa de aplicar estilos a los elementos de la página, mas adelante veremos la declaración en archivos externos. Para ello, y esto es la primera lección de este artículo, vamos a conocer un nuevo atributo que se puede utilizar en casi todas las etiquetas HTML: style.

Ejemplo:

```
<p style="color:green;font-weight:bold">El párrafo saldrá con color verde y en negrita</p>
```

Dentro del atributo style se deben indicar los atributos de estilos CSS separados por punto y coma (;). Durante este artículo vamos a conocer muchos atributos de CSS, los dos primeros que hemos visto aquí son:

Color: indica el color del contenido la etiqueta donde estemos utilizándolo, generalmente indica el color del texto.

Font-weight: indica el grosor del texto. Bold sirve para poner en negrita.

Color en los enlaces

Con HTML definimos el color de los enlaces en la etiqueta <body>, con lo atributos link, vlink y alink. Esto nos permite cambiar el color de los enlaces para todo el documento, pero ¿Y si queremos cambiar el color de un enlace en concreto, para que tenga otro color que el definido en la etiqueta <body>?

Para hacer esto utilizaremos el atributo style dentro del enlace:

```
<a href="mienlace.html" style="color:red">
```

Así saldrá el enlace en color rojo, independientemente de lo definido para todo el documento.

Espaciado entre líneas

Con CSS podemos definir el espacio que hay entre cada línea del documento, utilizando el atributo line-height. Por ejemplo, podemos definir que para todo un párrafo el espacio entre cada una de sus líneas sea 25 pixels:

```
<p style="line-height: 25px;">
Un párrafo normal en el que cada una de las líneas está separada 25 pixels de la otra. Hay que poner suficiente texto como para que se vean 2 líneas, así saldrán separadas
</p>
```

Espaciado entre caracteres

Se puede definir también el espacio entre cada carácter. Esto se hace con el atributo de CSS letter-spacing. Veamos un ejemplo:

```
<p style="letter-spacing:12cm">
Este párrafo tiene las letras espaciadas por 1 centímetro.
</p>
```

Este atributo, al igual que ocurre con muchos otros de CSS, no está soportado por todos los navegadores. En concreto Netscape, en su versión 4 todavía no lo incluye.

Enlaces sin subrayado

Uno de los efectos más significativos y fáciles de realizar con CSS es eliminar el subrayado de los enlaces de una página web. Existe un atributo que sirve para definir la decoración de un texto, si está subrayado, tachado, o si no tiene ninguna de estas "decoraciones". Es el atributo text-decoration, en

este caso indicaremos en un enlace que no queremos decoración:

```
<a href="mipagina.html" style="text-decoration:none">
```

Incluir estilos para todo un sitio web

Una de las características más potentes de la programación con hojas de estilo consiste en definir los estilos de todo un sitio web. Esto se consigue creando un archivo donde tan sólo colocamos las declaraciones de estilos de la página y enlazando todas las páginas del sitio con ese archivo. De este modo, todas las páginas comparten una misma declaración de estilos y, por tanto, si la cambiamos, cambiarán todas las páginas.

Veamos ahora todo el proceso para incluir estilos con un fichero externo.

1- Creamos el fichero con la declaración de estilos

Es un fichero de texto normal, que puede tener cualquier extensión, aunque le podemos asignar la extensión .css para aclararnos qué tipo de archivo es. El texto que debemos incluir debe ser escrito exclusivamente en sintaxis CSS, es un poco distinta que la sintaxis que utilizamos dentro del atributo style. Sería erróneo incluir código HTML en este archivo: etiquetas y demás. Podemos ver un ejemplo a continuación.

```
P {
font-size : 12pt;
font-family : arial,helvetica;
font-weight : normal;
}
H1 {
font-size : 36pt;
font-family : verdana,arial;
text-decoration : underline;
text-align : center;
background-color : Teal;
}
BODY {
background-color : #006600;
font-family : arial;
color : White;
}
```

2- Enlazamos la página web con la hoja de estilos

Para ello vamos a colocar la etiqueta <LINK> con los atributos

- rel="STYLESHEET" indicando que el enlace es con una hoja de estilo.
- type="text/css" porque el archivo es de texto, en sintaxis CSS.
- href="estilos.css" indica el nombre del fichero fuente de los estilos.

Veamos una página web entera que enlaza con la declaración de estilos anterior:

```
<html>
<head>
<link rel="STYLESHEET" type="text/css" href="estilos.css">
<title>Página que lee estilos</title>
</head>
<body>
<h1>Página que lee estilos</h1>
<p>
Esta página tiene en la cabecera la etiqueta necesaria para enlazar con la hoja de estilos. Es muy fácil.
</p>
</body>
</html>
```

Las CSS tienen mucho más jugo

Las Hojas de Estilo en Cascada son un estándar muy amplio, con unas especificaciones y posibilidades muy grandes. En este artículo hemos visto unos cuantos efectos interesantes que realizar aunque no

tengamos ningún conocimiento previo. Sin embargo, lo mejor para trabajar con esta tecnología es conocerla bien, gracias a ello, los resultados serán mucho más sorprendentes.

Para ampliar esta información y conocer más sobre CSS se puede encontrar el capitulo dedicado a CSS.

Qué es Javascript

Javascript es un lenguaje de programación utilizado para crear pequeños programitas encargados de realizar acciones dentro del ámbito de una página web.

Se trata de un lenguaje de programación del lado del cliente, porque es el navegador el que soporta la carga de procesamiento. Gracias a su compatibilidad con la mayoría de los navegadores modernos, es el lenguaje de programación del lado del cliente más utilizado.

Con Javascript podemos crear efectos especiales en las páginas y definir interactividades con el usuario. El navegador del cliente es el encargado de interpretar las instrucciones Javascript y ejecutarlas para realizar estos efectos e interactividades, de modo que el mayor recurso, y tal vez el único, con que cuenta este lenguaje es el propio navegador.

Javascript es el siguiente paso, después del HTML, que puede dar un programador de la web que decida mejorar sus páginas y la potencia de sus proyectos. Es un lenguaje de programación bastante **sencillo y pensado para hacer las cosas con rapidez**, a veces con ligereza. Incluso las personas que no tengan una experiencia previa en la programación podrán aprender este lenguaje con facilidad y utilizarlo en toda su potencia con sólo un poco de práctica.

Entre las acciones típicas que se pueden realizar en Javascript tenemos dos vertientes. Por un lado los **efectos especiales** sobre páginas web, para crear contenidos dinámicos y elementos de la página que tengan movimiento, cambien de color o cualquier otro dinamismo. Por el otro, javascript nos permite ejecutar instrucciones como respuesta a las acciones del usuario, con lo que podemos crear **páginas interactivas** con programas como calculadoras, agendas, o tablas de cálculo.

Javascript es un lenguaje con muchas posibilidades, permite la programación de pequeños scripts, pero también de programas más grandes, orientados a objetos, con funciones, estructuras de datos complejas, etc. Además, Javascript pone a disposición del programador todos los elementos que forman la página web, para que éste pueda acceder a ellos y modificarlos dinámicamente.

Con Javascript el programador, que se convierte en el verdadero dueño y controlador de cada cosa que ocurre en la página cuando la está visualizando el cliente.

> **Ver también:** Para que quede claro el lenguaje y alguna aplicación práctica, que se puede hacer y entender rápidamente, se puede acceder al artículo Efectos Rápidos con Javascript. En dicho artículo veremos la implementación de un botón de volver y la muestra de la última modificación de una página web.

Referencias

El capitulo manual de programación en Javascript, donde explicamos toda la sintaxis y metodología de programaión.

Además, podemos acceder al Manual de Javascript II, donde vamos a tratar de acercarnos a este lenguaje en profundidad y conocer todos sus secretos y recursos disponibles.

Qué es Visual Basic Script

Es un lenguaje de programación de scripts del lado del cliente, pero sólo compatible con Internet Explorer. Es por ello que su utilización está desaconsejada a favor de Javascript.

Está basado en Visual Basic, un popular lenguaje para crear aplicaciones Windows. Tanto su sintaxis como la manera de trabajar están muy inspirados en él. Sin embargo, no todo lo que se puede hacer en Visual Basic lo podremos hacer en Visual Basic Script, pues este último es una versión reducida del primero.

El modo de funcionamiento de Visual Basic Script para construir efectos especiales en páginas web es muy similar al utilizado en Javascript y los recursos a los que se puede acceder también son los mismos: el navegador.

Como decimos, no debemos utilizar este lenguaje en la mayoría de las ocasiones, aunque un caso donde tendría sentido utilizar Visual Basic Script sería la construcción de una Intranet donde sepamos con toda seguridad que los navegadores que se van a conectar serán siempre Internet Explorer. En este caso, un programador habitual de Visual Basic tendría más facilidades para realizar los scripts utilizando Visual Basic Script en lugar de Javascript.

> **Nota:** El popular ASP (Active Server Pages) es una tecnología de programación del lado del servidor. Habitualmente, los scripts ASP se escriben con Visual Basic Script también y eso no nos debe liar. Visual Basic Script, por tanto, es un lenguaje que se puede utilizar para la programación en el cliente, pero también para la programación en el servidor.
>
> En este artículo hemos hablado del lenguaje en su faceta del lado del cliente, puesto que en la faceta del servidor tenemos muchos manuales, pero están englobados dentro de la programación en ASP.

Referencias

Ver el capitulo manual de Visual Basic Script. Está orientado a enseñar la sintaxis y la programación del lado del cliente con el lenguaje.

Qué es Java

Java es un lenguaje de programación con el que podemos realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Está desarrollado por la compañía Sun Microsystems con gran dedicación y siempre enfocado a cubrir las necesidades tecnológicas más punteras.

Una de las principales características por las que Java se ha hecho muy famoso es que es un lenguaje independiente de la plataforma. Eso quiere decir que si hacemos un programa en Java podrá funcionar en cualquier ordenador del mercado. Es una ventaja significativa para los desarrolladores de software, pues antes tenían que hacer un programa para cada sistema operativo, por ejemplo Windows, Linux, Apple, etc. Esto lo consigue porque se ha creado una Máquina de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java y posibilita que este último se entienda perfectamente.

La independencia de plataforma es una de las razones por las que Java es interesante para Internet, ya que muchas personas deben tener acceso con ordenadores distintos. Pero no se queda ahí, Java está desarrollándose incluso para distintos tipos de dispositivos además del ordenador como móviles, agendas y en general para cualquier cosa que se le ocurra a la industria.

Pasado y presente

Java fue pensado originalmente para utilizarse en cualquier tipo de electrodoméstico pero la idea fracasó. Uno de los fundadores de Sun rescató la idea para utilizarla en el ámbito de Internet y convirtieron a Java en un lenguaje potente, seguro y universal gracias a que lo puede utilizar todo el mundo y es gratuito. Una de los primeros triunfos de Java fue que se integró en el navegador Netscape y permitía ejecutar programas dentro de una página web, hasta entonces impensable con el HTML.

Actualmente Java se utiliza en un amplio abanico de posibilidades y casi cualquier cosa que se puede hacer en cualquier lenguaje se puede hacer también en Java y muchas veces con grandes ventajas. Para lo que nos interesa a nosotros, con Java podemos programar páginas web dinámicas, con accesos a bases de datos, utilizando XML, con cualquier tipo de conexión de red entre cualquier sistema. En general, cualquier aplicación que deseemos hacer con acceso a través web se puede hacer utilizando Java.

Qué son los Applets de Java

Es otra manera de incluir código a ejecutar en los clientes que visualizan una página web. Se trata de pequeños programas hechos en Java, que se transfieren con las páginas web y que el navegador ejecuta en el espacio de la página.

Los applets de Java están programados en Java y precompilados, es por ello que la manera de trabajar de éstos varía un poco con respecto a los lenguajes de script como Javascript. Los applets son más difíciles de programar que los scripts en Javascript y requerirán unos conocimientos básicos o medios del lenguaje Java.

La principal ventaja de utilizar applets consiste en que son mucho menos dependientes del navegador que los scripts en Javascript, incluso independientes del sistema operativo del ordenador donde se ejecutan. Además, Java es más potente que Javascript, por lo que el número de aplicaciones de los applets podrá ser mayor.

Como desventajas en relación con Javascript cabe señalar que los applets son más lentos de procesar y que tienen espacio muy delimitado en la página donde se ejecutan, es decir, no se mezclan con todos los componentes de la página ni tienen acceso a ellos. Es por ello que con los applets de Java no podremos hacer directamente cosas como abrir ventanas secundarias, controlar Frames, formularios, capas, etc.

Cómo es posible la multiplataforma en Java

Java es compatible con todos los sistemas porque basa su funcionamiento en los Byte Codes, que no es más que una precompilación del código fuente de Java.

Estos Byte Codes no son el programa en Java propiamente dicho, sino un archivo que contiene un código intermedio que puede manejar la Máquina Virtual de Java. Cada sistema operativo dispone de una Maquina Virtual de Java que puede interpretar los Byte Codes y transformarlos a sentencias ejecutables en el sistema en cuestión.

Qué es ASP

ASP (Active Server Pages) es la tecnología desarrollada por Microsoft para la creación de páginas dinámicas del servidor. ASP se escribe en la misma página web, utilizando el lenguaje Visual Basic Script o Jscript (Javascript de Microsoft).

Un lenguaje del lado del servidor es aquel que **se ejecuta en el servidor web**, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores. Podemos saber algo más sobre la programación del servidor y del cliente en el artículo qué es DHTML.

El tipo de servidores que emplean este lenguaje son, evidentemente, todos aquellos que funcionan con sistema Windows NT, aunque también se puede utilizar en un PC con windows 98 si instalamos un servidor denominado Personal Web Server. Incluso en sistemas Linux podemos utilizar las ASP si intalamos un componente denominado Chilisoft, aunque parece claro que será mejor trabajar sobre el servidor web para el que está pensado: Internet Information Server.

Con las ASP podemos realizar muchos tipos de aplicaciones distintas. Nos permite acceso a bases de datos, al sistema de archivos del servidor y en general a todos los recursos que tenga el propio servidor. También tenemos la posibilidad de comprar componentes ActiveX fabricados por distintas empresas de desarrollo de software que sirven para realizar múltiples usos, como el envio de correo, generar gráficas dinámicamente, y un largo etc.

Actualmente se ha presentado ya la segunda versión de ASP, el ASP.NET, que comprende algunas mejoras en cuanto a posibilidades del lenguaje y rapidez con la que funciona. ASP.NET tiene algunas diferencias en cuanto a sintaxis con el ASP, de modo que se ha de tratar de distinta manera uno de otro.

Qué es PHP

PHP es el acrónimo de Hipertext Preprocesor. **Es un lenguaje de programación del lado del servidor gratuito e independiente de plataforma**, rápido, con una gran librería de funciones y mucha documentación.

Un lenguaje del lado del servidor es aquel que **se ejecuta en el servidor web**, justo antes de que se envíe la página a través de Internet al cliente. Las páginas que se ejecutan en el servidor pueden realizar accesos a bases de datos, conexiones en red, y otras tareas para crear la página final que verá el cliente. El cliente solamente recibe una página con el código HTML resultante de la ejecución de la PHP. Como la página resultante contiene únicamente código HTML, es compatible con todos los navegadores. Podemos saber algo más sobre la programación del servidor y del cliente en el artículo qué es DHTML.

Una vez que ya conocemos el concepto de lenguaje de programación de scripts del lado del servidor podemos hablar de PHP. **PHP se escribe dentro del código HTML**, lo que lo hace realmente fácil de utilizar, al igual que ocurre con el popular ASP de Microsoft, pero con algunas ventajas como su gratuidad, independencia de plataforma, rapidez y seguridad. Cualquiera puede descargar a través de la página principal de PHP www.php.net y de manera gratuita, un módulo que hace que nuestro servidor web comprenda los scripts realizados en este lenguaje. Es independiente de plataforma, puesto que existe un módulo de PHP para casi cualquier servidor web. Esto hace que cualquier sistema pueda ser compatible con el lenguaje y significa una ventaja importante, ya que permite portar el sitio desarrollado en PHP de un sistema a otro sin prácticamente ningún trabajo.

PHP, en el caso de estar montado sobre un servidor Linux u Unix, es más rápido que ASP, dado que se ejecuta en un único espacio de memoria y esto evita las comunicaciones entre componentes COM que se realizan entre todas las tecnologías implicadas en una página ASP.

Por último señalábamos la seguridad, en este punto también es importante el hecho de que en muchas ocasiones PHP se encuentra instalado sobre servidores Unix o Linux, que son de sobra conocidos como más veloces y seguros que el sistema operativo donde se ejecuta las ASP, Windows NT o 2000. Además, PHP permite configurar el servidor de modo que se permita o rechacen diferentes usos, lo que puede hacer al lenguaje más o menos seguro dependiendo de las necesidades de cada cual.

Fue creado originalmente en 1994 por Rasmus Lerdorf, pero como **PHP está desarrollado en política de código abierto**, a lo largo de su historia ha tenido muchas contribuciones de otros desarrolladores. Actualmente PHP se encuentra en su **versión 4, que utiliza el motor Zend**, desarrollado con mayor meditación para cubrir las necesidades de las aplicaciones web actuales.

Este lenguaje de programación está preparado para realizar muchos tipos de aplicaciones web gracias a la extensa librería de funciones con la que está dotado. La librería de funciones cubre desde cálculos matemáticos complejos hasta tratamiento de conexiones de red, por poner dos ejemplos.

Algunas de las más importantes capacidades de PHP son: **compatibilidad con las bases de datos** más comunes, como MySQL, mSQL, Oracle, Informix, y ODBC, por ejemplo. Incluye funciones para el **envío de correo electrónico**, **upload de archivos**, crear **dinámicamente en el servidor imágenes en formato GIF**, incluso animadas y una lista interminable de utilidades adicionales.

Qué es XML

XML es una tecnología en realidad muy sencilla que tiene a su alrededor otras tecnologías que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Vamos a ver a lo largo de varios capítulos una introducción al mundo XML, es decir, al lenguaje así como a las tecnologías que trabajan con él, sus usos, ventajas y modos de llevar a cabo las tareas.

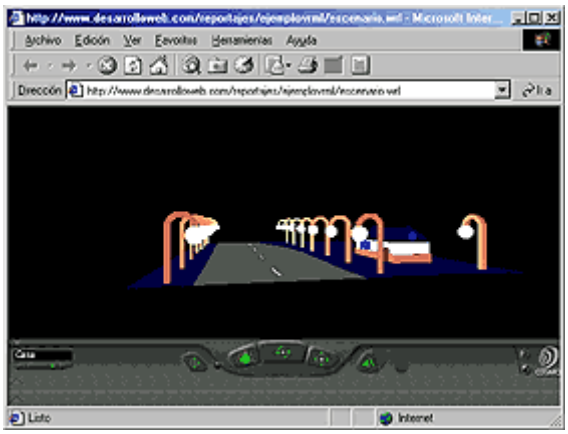
XML, con todas las tecnologías relacionadas, representa una manera distinta de hacer las cosas, más avanzada, cuya principal novedad consiste en permitir compartir los datos con los que se trabaja a todos los niveles, por todas las aplicaciones y soportes. Así pues, el XML juega un papel importantísimo en este mundo actual, que tiende a la globalización y la compatibilidad entre los sistemas, ya que es la tecnología que permitirá compartir la información de una manera segura, fiable, fácil. Además, XML permite al programador y los soportes dedicar sus esfuerzos a las tareas importantes cuando trabaja con los datos, ya que algunas tareas tediosas como la validación de estos o el recorrido de las estructuras corre a cargo del lenguaje y está especificado por el estándar, de modo que el programador no tiene que preocuparse por ello.

Vemos que XML no está sólo, sino que hay un mundo de tecnologías alrededor de él, de posibilidades, maneras más fáciles e interesantes de trabajar con los datos y, en definitiva, un avance a la hora de tratar la información, que es en realidad el objetivo de la informática en general. XML, o mejor dicho, el mundo XML no es un lenguaje, sino varios lenguajes, no es una sintaxis, sino varias y no es una manera totalmente nueva de trabajar, sino una manera más refinada que permitirá que todas las anteriores se puedan comunicar entre si sin problemas, ya que los datos cobran sentido. Todo esto lo veremos con calma en la Introducción a XML.

XML es interesante en el mundo de Internet y el e-bussiness, ya que existen muchos sistemas distintos que tienen que comunicarse entre si, pero como se ha podido imaginar, interesa por igual a todas las ramas de la informática y el tratamiento de datos, ya que permite muchos avances a la hora de trabajar con ellos.

En la introducción a XML, a lo largo de los siguientes capítulos, vamos a ver algunas características importantes de la tecnología que nos permitirán comprender mejor el mundo XML y cómo soluciona nuestros problemas a la hora de trabajar con los datos.

Qué es VRML



El Virtual Reality Modeling Language es un **lenguaje de modelado de mundos virtuales en tres dimensiones**. Igual que el HTML nos sirve para maquetar páginas web, VRML sirve para crear mundos en tres dimensiones a los que accedemos utilizando nuestro navegador, igual que si visitasemos una página web cualquiera, con la salvedad que nuestras visitas no se limitan a ver un simple texto y fotografías, sino que nos permite ver todo tipo de objetos y construcciones en 3D por los que podemos pasear o interactuar.

Este modo de visitar sitios en Internet es mucho más avanzado y posee grandes ventajas. Para empezar **la navegación se desarrolla de una manera mucho más intuitiva**, dado que la forma de actuar dentro del mundo virtual es similar a la de la vida real. Podemos movernos en todas las direcciones, no solo izquierda y derecha sino también adelante, atrás, arriba y abajo. Podemos tratar con los objetos como en la vida misma, tocarlos, arrastrarlos, etc. y en general todo lo que podais imaginar. También **los escenarios son mucho más reales**, pensemos en un ejemplo como podría ser una biblioteca virtual. En élla podríamos andar por cada una de las salas, tomar determinados libros y leerlos.

A la larga, el acceso a Internet se ha de convertir en una experiencia mucho más cercana a la que realizamos en la vida y las visitas a los lugares retratados en la Red serán mucho más reales. Sin embargo, en la actualidad VRML presenta muchas limitaciones con respecto a sus potencialidades, que se irán cubriendo con la llegada de máquinas más potentes y periféricos avanzados para la realidad virtual como pueden ser guantes o cascos.

Un poco de historia

El VRML surgió en la primavera de 1994, durante una reunión convocada por Tim Berners-Lee y Dave Ragget para tratar de acercar los desarrollos de realidad virtual a Internet. En esta reunión los asistentes llegaron a la conclusión que se tenía que desarrollar un lenguaje común para la descripción de los mundos en 3D.

De este modo, **en la Primera Conferencia Mundial de la WWW en Ginebra se aprobó el desarrollo de un nuevo lenguaje que permitiese crear mundos en tres dimensiones** a los que se pudiera acceder por la World Wide Web.

Con el tiempo se desarrolló el lenguaje dentro de varios requisitos: que fuese adaptable a la red, que no requiriese una línea de alta velicidad (anchos de banda elevados), que fuese multiplataforma, etc.

Materiales necesarios

Los materiales necesarios para comenzar con VRML son pocos, y posiblemente ya tengamos, sin saberlo, todos los ingredientes para introducirnos en el desarrollo y edición de mundos virtuales. Estos son:

- **Un editor de textos sencillo**. El Block de notas es ideal. También valdrá cualquier otro editor en modo ASCII. También podemos utilizar editores especializados como el VRML PAD.
- **Un visualizador VRML** para ver los resultados, que se instala en el navegador como cualquier otro plug-in. Posiblemente tu navegador ya esté preparado para ver los mundos en VRML, si no es así, tienes que instalarlo. Un visualizador muy conocido es el Cosmo Player.

Qué son las cookies

En nuestros viajes por la Red visitamos gran cantidad de páginas, muchas de ellas bastante complicadas que implementan distintos servicios de Internet. Estas páginas tienen que guardar distintas informaciones acerca de un usuario, por ejemplo su nombre, su edad o su color preferido. Para ello cuentan con una serie de mecanismos en el servidor como bases de datos u otro tipo de contenedores, pero hay un mecanismo mucho más interesante de guardar esa información que los propios recursos del servidor, que es el propio ordenador del usuario.

En nuestros ordenadores se guardan muchos datos que necesitan conocer las páginas web cada vez que entramos en la página, estas pequeñas informaciones son las cookies: **estados de variables que se conservan de una visita a otra en el ordenador del cliente**.

Como es un poco peligroso que las páginas web a las que accedemos se dediquen a introducir cosas en nuestro ordenador, las cookies están altamente restringidas. Para empezar, **solamente podemos guardar en ellas textos**, nunca programas, imágenes, etc. Además, los textos **nunca podrán ocupar más de 1 K**, de modo que nadie podría inundarnos el ordenador a base de cookies. Estas restricciones, unidas a la necesidad de **poner una fecha de caducidad** a las cookies para que estas se guarden, hacen que el aceptar cookies no signifique un verdadero problema para la integridad de nuestros sistemas.

Ejemplos de uso de las cookies

Un ejemplo típico de las cookies podría ser un **contador de las veces que accede un usuario a una página**. Podríamos poner una cookie en el ordenador del cliente donde tendríamos una variable que lleva la cuenta de las veces que ha accedido a la página y cada vez que se accede se incrementa en uno.

En desarrolloweb, igual que muchos otros sitios, utilizamos las cookies para **guardar la personalización de un usuario** de la página. En el momento de escribir este reportaje se puede ver en el diseño de desarrolloweb un botón que pone personalización y donde hay varios colores. Pulsando cada color la página cambia, para verse con ese color. Para que el color elegido se conserve de visita en visita se utilizan las cookies.

Otro ejemplo típico es el que **guarda el perfil del usuario**. Si el usuario accede a contenidos determinados podemos enviarle una cookie que le marca como interesado en un tema. A medida que va accediendo a distintos sitios le vamos encasillando como joven, adulto, hombre, mujer, o lo que proceda. Conociendo el perfil de un usuario se le pueden ofrecer un tipo de productos o servicios orientados a sus gustos o necesidades.

Cómo se utilizan las cookies

Para trabajar con cookies tenemos que **utilizar un lenguaje de programación avanzado como Javascript o ASP, PHP, etc**. No podemos entonces trabajar con cookies si solamente nos dedicamos a utilizar el HTML, que ya sabemos que es un poco limitado para cosas que se salgan de mostrar contenido en páginas estáticas.

En desarrolloweb tenemos las secciones de ASP y PHP, donde puedes encontrar algunos artículos que explican el uso de las cookies en estos lenguajes.

Polémica de las cookies

Existe un problema con estas galletitas y es que nos cortan nuestra intimidad. Lo que señalábamos anteriormente acerca de guardar el perfil de un usuario puede llegar a ser un problema para nosotros, por que nos están vigilando y apuntando cada uno de nuestros movimientos, lo que puede convertirse en un abuso de información que no tiene por qué pertenecer a nadie más que nosotros. Las empresas que más utilizan esta clasificación del personal son los AD-Servers (servidores de banners) como el de Doubleclick. No queremos entrar aquí más en esta polémica, pero si despertar la inquietud de que posiblemente estén entrando en nuestra intimidad.

Qué es SQL

Las aplicaciones en red son cada día más numerosas y versátiles. En muchos casos, el esquema básico de operación es una serie de scripts que rigen el comportamiento de una base de datos.

Debido a la diversidad de lenguajes y de bases de datos existentes, la manera de comunicar entre unos y otras sería realmente complicada a gestionar de no ser por la existencia de estándares que nos permiten el realizar las operaciones básicas de una forma universal.

Es de eso de lo que trata el Structured Query Language que no es mas que un lenguaje estándar de comunicación con bases de datos. Hablamos por tanto de un lenguaje normalizado que nos permite

trabajar con cualquier tipo de lenguaje (ASP o PHP) en combinación con cualquier tipo de base de datos (MS Access, SQL Server, MySQL...).

El hecho de que sea estándar no quiere decir que sea idéntico para cada base de datos. En efecto, determinadas bases de datos implementan funciones específicas que no tienen necesariamente que funcionar en otras.

Aparte de esta universalidad, el SQL posee otras dos características muy apreciadas. Por una parte, presenta una potencia y versatilidad notables que contrasta, por otra, con su accesibilidad de aprendizaje.

El manual de SQL pretende dar a conocer las operaciones básicas que se pueden realizar con SQL y que tienen una aplicación directa con la creación de aplicaciones en red sin profundizar más de lo estrictamente necesario. Buscamos con ello ofrecer al webmaster un manual de referencia práctico y aplicado.

Qué es un Webmaster

¿Qué es Ser WebMaster? La palabra WebMaster en definitiva es una palabra de origen Inglés, que si lo traducimos al castellano queda algo como Maestro Web. Pero muchas veces escuchamos estas palabras que soy WebMaster, conozco a un WebMaster y no sabemos realmente cual es la función especifica de estas personas en las cuales me incluyo.

Un WebMaster es la persona encargada de un sitio, vendría siendo como el director de una empresa, es la persona que decide las tecnologías que se van a usar, decide, los servidores, los chiches y la estructura.

Por empezar un WebMaster es una persona responsable, es la responsable propiamente dicho de Todo un Sitio, ya que es la que tiene por ejemplo, los códigos o passwords para hacer modificaciones en la pagina. Es la única persona autorizada a que si no le gusta la estructura o los contenidos del sitio puede decir que va o no va.

WebMasters Vs. Diseñadores Gráficos.

No tenemos que confundir los tantos, los WebMaster son WebMaster propiamente dicho, un diseñador es un diseñador por lo cual, en nuestro site si no sabemos de diseño démosle trabajo a la gente que sabe del tema, ya que un WebMaster abarca otros temas como explicábamos y tiene otras responsabilidades.

Muchas veces nosotros los WebMaster queremos dar nuestro toque de diseño a nuestro site, claro que podemos aportar muchas cosas para los encargados de diseño, pero una recomendación muy importante, no decidir en el tema de diseño, tenemos que tener para armar nuestro site, un encargado en el área, un experto en el tema de diseño para que nosotros solo nos aboquemos a nuestro trabajo que es el de armar todo lo que el grupo arma, en diseño y contenido.

Una cosa esencial que tenemos que ver a la hora de empezar a trabajar en nuestro site con los diseñadores es de explicarles los servidores en los que se va a trabajar, explicarles con porcentajes las tecnologías que convienen y las que no, en definitiva adaptarlos a lo que tenemos nosotros.

WebMasters y seguridad

Si bien hablamos de que los WebMaster son las personas que están encargadas de tener los códigos de acceso a la pagina, tenemos que hablar de que también puede ser el administrador del site, por lo tanto tiene que tener conocimientos amplios sobre seguridad informática para proteger de cualquier ataque a el sitio.

También tenemos que tratar los WebMaster es de toda la información que corra por nuestro site que sean lo mas privadas posibles, tenemos que asegurarnos siempre de que nuestros Server sean seguros para que no suframos ataques de hackers con malas intenciones y que nos perjudiquen nuestro trabajo.

Pero en fin ¿Quién se puede hacer llamar WebMaster?

Para ser WebMaster no hay que estudiar ninguna carrera universitaria ni menos tenemos que ir a algún curso para recibirnos de WebMaster. Los WebMaster somos personas altamente capacitadas en el ámbito de Internet por lo cual conocemos a fondo las tecnologías del mercado, somos los que en partes técnicas sabemos bien sobre lo que estamos trabajando, somos la parte mas importante del site porque somos las personas que unen a todas las personas que trabajan, los de contenido, los diseñadores, los noteros, los directivos, para hacer alguna publicación automáticamente pasan por nosotros los WebMaster.

Un punto fundamental que tenemos que tener muy pero muy en cuenta para ser WebMaster es la responsabilidad que tenemos, ya que si leímos lo que decía mas arriba de que somos la persona mas importante de el sitio, tenemos que estar siempre listos para cualquier desafio, y tenemos que afrontar

con responsabilidad nuestro puesto de trabajo.

Qué es Streaming

La tecnología de streaming se utiliza para aligerar la descarga y ejecución de audio y vídeo en la web, ya que permite escuchar y visualizar los archivos mientras se están descargando.

Si no utilizamos streaming, para mostrar un contenido multimedia en la Red, tenemos que descargar primero el archivo entero en nuestro ordenador y más tarde ejecutarlo, para finalmente ver y oir lo que el archivo contenía. Sin embargo, el streaming permite que esta tarea se realice de una manera más rápida y que podamos ver y escuchar su contenido durante la descarga.

El streaming funciona de la siguiente manera. Primero nuestro ordenador (el cliente) conecta con el servidor y éste le empieza a mandar el fichero. El cliente comienza a recibir el fichero y construye un buffer donde empieza a guardar la información. Cuando se ha llenado el buffer con una pequeña parte del archivo, el cliente lo empieza a mostrar y a la vez continúa con la descarga. El sistema está sincronizado para que el archivo se pueda ver mientras que el archivo se descarga, de modo que cuando el archivo acaba de descargarse el fichero también ha acabado de visualizarse. Si en algún momento la conexión sufre descensos de velocidad se utiliza la información que hay en el buffer, de modo que se puede aguantar un poco ese descenso. Si la comunicación se corta demasiado tiempo, el buffer se vacía y la ejecución el archivo se cortaría también hasta que se restaurase la señal.

Programas de Streaming

En realidad, este proceso de streaming lo podemos haber visto en muchas ocasiones en nuestros ordenadores. Es lo que hacen programas como el Real Player o el Windows Media Player, programas que se instalan como plug-ins en los navegadores para recibir y mostrar contenidos multimedia por streaming.

Cuando pretendemos incluir audio o video en las páginas lo mejor pues, es utilizar la tecnología de streaming. Para ello simplemente tenemos que guardar los archivos multimedia con el formato de uno de los programas de streaming y seguir unas pequeñas normas a la hora de subirlos a Internet y colocarlos en la página. Las normas que seguir son propias de cada sistema y no las veremos aquí. Lo mejor para enterarse de cómo funcionan es visitar las correspondientes páginas web, señaladas más abajo.

Para convertir los archivos de audio y vídeo al formato de cada programa de streaming se utilizan unos programas especiales que se pueden descargar de las páginas de cada tecnología. Por ejemplo, el programa para convertir al formato que lee el Real Player se llama Real Producer.

A la hora de desarrollar el web con contenidos multimedia será necesario que nos decidamos a utilizar una tecnología de streaming en concreto y no las utilicemos todas para no obligar a nuestros usuarios a descargarse todos los plug-ins del mercado. A continuación vemos las tres posibles tecnologías de streaming del momento.

Real Media es posiblemente la más popular. También es la empresa con más experiencia en el sector y desarrolla muchos productos orientados a la distribución de archivos multimedia Su web: www.real.com

Windows Media es la apuesta de Microsoft. Ya posee una cuota de usuarios muy importante y seguramente aumentará con rapidez ya que Microsoft incluye el plug-in en la instalación típica de los sistemas operativos que está fabricando.

Quick Time es la tercera en discordia. Con menor cuota de mercado.

Servidores de Streaming

En principio no es necesario contar con un servidor especial para colocar archivos de audio o vídeo con descarga streaming en nuestras webs. Cualquier servidor normal puede mandar la información y es el cliente el que se encarga de procesarla para poder mostrarla a medida que la va recibiendo.

Sin embargo, existen servidores especiales preparados para transmitir streaming. Aunque en muchas ocasiones no es necesario utilizarlos nos pueden ofrecer importantes prestaciones como mandar un archivo de mayor o menor calidad dependiendo de la velocidad de nuestra línea.

En determinados casos, como la puesta en marcha de una radio o la transmisión de un evento en directo, si que será imprescindible contar con un servidor de streaming al que mandaremos la señal y con ella, la enviará a todos los clientes a medida que la va recibiendo.

Conclusión

No cabe duda que la transmisión de contenido multimedia a través de la web será cada vez más importante. La tecnología de streaming es un mercado con futuro y grandes compañías ya están luchando por el mercado. La velocidad de Internet aumentará con el tiempo y con ella aumentará la calidad de las transmisiones, para hacer posible tanto la radio como la televisión en Internet.

Qué es la programacion orientada a objetos

La programación Orientada a objetos (POO) es una forma especial de programar, más cercana a como expresaríamos las cosas en la vida real que otros tipos de programación.

Con la POO tenemos que aprender a pensar las cosas de una manera distinta, para escribir nuestros programas en términos de objetos, propiedades, métodos y otras cosas que veremos rápidamente para aclarar conceptos y dar una pequeña base que permita soltarnos un poco con este tipo de programación.

Motivación

Durante años, los programadores se han dedicado a construir aplicaciones muy parecidas que resolvían una y otra vez los mismos problemas. Para conseguir que los esfuerzos de los programadores puedan ser utilizados por otras personas se creó la POO. Que es una serie de normas de realizar las cosas de manera que otras personas puedan utilizarlas y adelantar su trabajo, de manera que consigamos que el código se pueda reutilizar.

La POO no es difícil, pero es una manera especial de pensar, a veces subjetiva de quien la programa, de manera que la forma de hacer las cosas puede ser diferente según el programador. Aunque podamos hacer los programas de formas distintas, no todas ellas son correctas, lo difícil no es programar orientado a objetos sino programar bien. Programar bien es importante porque así nos podemos aprovechar de todas las ventajas de la POO.

Cómo se piensa en objetos

Pensar en términos de objetos es muy parecido a cómo lo haríamos en la vida real. Por ejemplo vamos a pensar en un coche para tratar de modelizarlo en un esquema de POO. Diríamos que el coche es el elemento principal que tiene una serie de características, como podrían ser el color, el modelo o la marca. Además tiene una serie de funcionalidades asociadas, como pueden ser ponerse en marcha, parar o aparcar.

Pues en un esquema POO el coche sería el objeto, las propiedades serían las características como el color o el modelo y los métodos serían las funcionalidades asociadas como ponerse en marcha o parar.

Por poner otro ejemplo vamos a ver cómo modelizaríamos en un esquema POO una fracción, es decir, esa estructura matemática que tiene un numerador y un denominador que divide al numerador, por ejemplo 3/2.

La fracción será el objeto y tendrá dos propiedades, el numerador y el denominador. Luego podría tener varios métodos como simplificarse, sumarse con otra fracción o número, restarse con otra fracción, etc.

Estos objetos se podrán utilizar en los programas, por ejemplo en un programa de matemáticas harás uso de objetos fracción y en un programa que gestione un taller de coches utilizarás objetos coche. Los programas Orientados a objetos utilizan muchos objetos para realizar las acciones que se desean realizar y ellos mismos también son objetos. Es decir, el taller de coches será un objeto que utilizará objetos coche, herramienta, mecánico, recambios, etc.

Clases en POO

Las clases son declaraciones de objetos, también se podrían definir como abstracciones de objetos. Esto quiere decir que la definición de un objeto es la clase. Cuando programamos un objeto y definimos sus características y funcionalidades en realidad lo que estamos haciendo es programar una clase. En los ejemplos anteriores en realidad hablábamos de las clases coche o fracción porque sólo estuvimos definiendo, aunque por encima, sus formas.

Propiedades en clases

Las propiedades o atributos son las características de los objetos. Cuando definimos una propiedad normalmente especificamos su nombre y su tipo. Nos podemos hacer a la idea de que las propiedades son algo así como variables donde almacenamos datos relacionados con los objetos.

Métodos en las clases

Son las funcionalidades asociadas a los objetos. Cuando estamos programando las clases las llamamos métodos. Los métodos son como funciones que están asociadas a un objeto.

Objetos en POO

Los objetos son ejemplares de una clase cualquiera. Cuando creamos un ejemplar tenemos que especificar la clase a partir de la cual se creará. Esta acción de crear un objeto a partir de una clase se llama instanciar (que viene de una mala traducción de la palabra instace que en inglés significa ejemplar). Por ejemplo, un objeto de la clase fracción es por ejemplo 3/5. El concepto o definición de fracción sería la clase, pero cuando ya estamos hablando de una fracción en concreto 4/7, 8/1000 o cualquier otra, la llamamos objeto.

Para crear un objeto se tiene que escribir una instrucción especial que puede ser distinta dependiendo el lenguaje de programación que se emplee, pero será algo parecido a esto.

```
miCoche = new Coche()
```

Con la palabra new especificamos que se tiene que crear una instancia de la clase que sigue a continuación. Dentro de los paréntesis podríamos colocar parámetros con los que inicializar el objeto de la clase coche.

Estados en objetos

Cuando tenemos un objeto sus propiedades toman valores. Por ejemplo, cuando tenemos un coche la propiedad color tomará un valor en concreto, como por ejemplo rojo o gris metalizado. El valor concreto de una propiedad de un objeto se llama estado.

Para acceder a un estado de un objeto para ver su valor o cambiarlo se utiliza el operador punto.

```
miCoche.color = rojo
```

El objeto es miCoche, luego colocamos el operador punto y por último el nombre e la propiedad a la que deseamos acceder. En este ejemplo estamos cambiando el valor del estado de la propiedad del objeto a rojo con una simple asignación.

Mensajes en objetos

Un mensaje en un objeto es la acción de efectuar una llamada a un método. Por ejemplo, cuando le decimos a un objeto coche que se ponga en marcha estamos pasándole el mensaje “ponte en marcha”.

Para mandar mensajes a los objetos utilizamos el operador punto, seguido del método que deseamos imbocar.

```
miCoche.ponerseEnMarcha()
```

En este ejemplo pasamos el mensaje ponerseEnMarcha(). Hay que colocar paréntesis igual que cualquier llamada a una función, dentro irían los parámetros.

Otras cosas

Hay mucho todavía que conocer de la POO ya que sólo hemos hecho referencia a las cosas más básicas. También existen mecanismos como la herencia y el polimorfismo que son unas de las posibilidades más potentes de la POO.

La herencia sirve para crear objetos que incorporen propiedades y métodos de otros objetos. Así podremos construir unos objetos a partir de otros sin tener que reescribirlo todo.

El polimorfismo sirve para que no tengamos que preocuparnos sobre lo que estamos trabajando, y abstraernos para definir un código que sea compatible con objetos de varios tipos.

Son conceptos avanzados que cuesta explicar en las líneas de ese informe. No hay que olvidar que existen libros enteros dedicados a la POO y aquí solo pretendemos dar un repaso a algunas cosas para que os suenen cuando tengáis que poneros delante de ellas en los lenguajes de programación que debe conocer un desarrollador del web.

Que es un firewall

Un firewall es un dispositivo que funciona como cortafuegos entre redes, permitiendo o denegando las transmisiones de una red a la otra. Un uso típico es situarlo entre una red local y la red Internet, como dispositivo de seguridad para evitar que los intrusos puedan acceder a información confidencial.

Un firewal es simplemente un filtro que controla todas las comunicaciones que pasan de una red a la otra y en función de lo que sean permite o deniega su paso. Para permitir o denegar una comunicación el firewal examina el tipo de servicio al que corresponde, como pueden ser el web, el correo o el IRC. Dependiendo del servicio el firewall decide si lo permite o no. Además, el firewall examina si la comunicación es entrante o saliente y dependiendo de su dirección puede permitirla o no.

De este modo un firewall puede permitir desde una red local hacia Internet servicios de web, correo y ftp, pero no a IRC que puede ser innecesario para nuestro trabajo. También podemos configurar los accesos que se hagan desde Internet hacia la red local y podemos denegarlos todos o permitir algunos servicios como el de la web, (si es que poseemos un servidor web y queremos que accesible desde Internet). Dependiendo del firewall que tengamos también podremos permitir algunos accesos a la red local desde Internet si el usuario se ha autentificado como usuario de la red local.

Un firewall puede ser un dispositivo software o hardware, es decir, un aparatito que se conecta entre la red y el cable de la conexión a Internet, o bien un programa que se instala en la máquina que tiene el modem que conecta con Internet. Incluso podemos encontrar ordenadores computadores muy potentes y con softwares específicos que lo único que hacen es monitorizar las comunicaciones entre redes.

Si se quiere saber algo más sobre firewalls podemos examinar los enlaces que tenemos en el área de seguridad. Es especialmente interesante este enlace:
http://www.ciudadfutura.com/mundopc/cursos/firewalls/fire1.htm

Qué es CGI

Es el sistema más antiguo que existe para la programación de las páginas dinámicas de servidor. Actualmente se encuentra un poco desfasado por diversas razones entre las que destaca la dificultad con la que se desarrollan los programas y la pesada carga que supone para el servidor que los ejecuta.

Los CGI se escriben habitualmente en el lenguaje Perl, sin embargo, otros lenguajes como C, C++ o Visual Basic pueden ser también empleados para construirlos.

El funcionamiento básico de un programa CGI es parecido al apuntado para el conjunto de las páginas dinámicas del servidor, con algunas particularidades.

1. Se realiza una petición http, a la que pueden acompañar datos llegados o bien por un formulario o bien a través de la URL.
2. El servidor ejecuta los programas CGI a los que se accede y trabaja con los recursos necesarios para llevar a cabo las acciones, como por ejemplo bases de datos.
3. El programa CGI va escribiendo en la salida estándar el resultado de la ejecución del CGI, que incluye etiquetas HTML, ya que lo que se escribe es una página web.

Algunas desventajas de la programación en CGI son las siguientes:

- Los resultados se escriben directamente con el CGI, así que el código del programa se mezcla con el del HTML haciendo difícil su comprensión y mantenimiento.
- Cada programa CGI que se pone en marcha lo hace en un espacio de memoria propio. Así, si tres usuarios ponen en marcha un CGI a la vez se multiplicará por tres la cantidad de recursos que ocupe ese CGI. Esto significa una grave ineficiencia.

Para completar esta información sería interesante acceder a la sección CGI de nuestro directorio de enlaces, donde podemos encontrar sitios en Internet que ofrecen tutoriales sobre la tecnología y directorios de programas CGI ya creados para hacer cosas tan variadas como tiendas, foros, envio de formularios, etc.

Qué es Perl

Es un lenguaje de programación muy utilizado para construir aplicaciones CGI para el web. Perl es un acrónimo de Practical Extracting and Reporting Languaje, que viene a indicar que se trata de un lenguaje de programación muy práctico para extraer información de archivos de texto y generar informes a partir del contendido de los ficheros.

Es un lenguaje libre de uso, eso quiere decir que es gratuito. Antes estaba muy asociado a la plataforma Uníx, pero en la actualidad está disponible en otros sistemas operativos como Windows.

Perl es un lenguaje de programación interpretado, al igual que muchos otros lenguajes de Internet como Javascript o ASP. Esto quiere decir que el código de los scripts en Perl no se compila sino que cada vez que se quiere ejecutar se lee el código y se pone en marcha interpretando lo que hay escrito. Además es extensible a partir de otros lenguajes, ya que desde Perl podremos hacer llamadas a subprogramas escritos en otros lenguajes. También desde otros lenguajes podremos ejecutar código Perl.

Perl está inspirado a partir de lenguajes como C, sh, awk y sed (algunos provenientes de los sistemas Uníx), pero está enfocado a ser más práctico y fácil que estos últimos. Es por ello que un programador que haya trabajado con el lenguaje C y los otros tendrá menos problemas en entenderlo y utilizarlo rápidamente. Una diferencia fundamental de Perl con respecto a los otros lenguajes es que no limita el tamaño de los datos con los que trabaja, el límite lo pone la memoria que en ese momento se encuentre disponible.

Si queremos trabajar con Perl será necesario tener instalado el interprete del lenguaje. A partir de ese momento podemos ejecutar CGIs en nuestros servidores web. El proceso para conseguirlo puede variar de unos servidores a otros, pero se suelen colocar en un directorio especial del servidor llamado cgi-bin donde hemos colocado los correspondientes permisos CGI. Además, los archivos con el código también deberán tener permiso de ejecución.

Qué es C#

C# es el nuevo lenguaje de propósito general orientado a objetos creado por Microsoft para su nueva plataforma .NET.

Microsoft.NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando estos últimos años con el objetivo de mejorar tanto su sistema operativo como su modelo de componentes (COM) para obtener una plataforma con la que sea sencillo el desarrollo de software en forma de servicios web.

Los servicios web son un novedoso tipo de componentes software que se caracterizan a la hora de trabajar por su total independencia respecto a su ubicación física real, la plataforma sobre la que corre, el lenguaje de programación con el que hayan sido desarrollados o el modelo de componentes utilizado para ello.

El acceso a estos servicios se realiza en base a estándares de Internet, como son diferentes mecanismos del protocolo HTTP (GET y PUT) o el novedoso protocolo RPC conocido como SOAP (Simple Access Object Protocol), que no es más que una combinación de estándares como HTTP y XML para realizar llamadas a los miembros de estos servicios web. La idea detrás de SOAP consiste sencillamente en utilizar HTTP como medio de transporte para el envío de los mensajes de solicitud de ejecución de los miembros de servicios web remotos (lo que permite atravesar barreras tales como firewalls) y utilizar XML como lenguaje con el que escribir los cuerpos de estos mensajes.

Pero la plataforma .NET no son sólo los servicios web, sino que también ofrece numerosos servicios a las aplicaciones que para ella se escriban, como son un recolección de basura, independencia de la plataforma, total integración entre lenguajes (por ejemplo, es posible escribir una clase en C# que derive de otra escrita en Visual Basic.NET que a su vez derive de otra escrita en Cobol)

Como se deduce del párrafo anterior, es posible programar la plataforma .NET en prácticamente cualquier lenguaje, pero Microsoft ha decidido sacar uno nuevo porque ha visto conveniente poder disponer de un lenguaje diseñado desde 0 con vistas a ser utilizado en .NET, un lenguaje que no cuente con elementos heredados de versiones anteriores e innecesarios en esta plataforma y que por tanto sea lo más sencillo posible para programarla aprovechando toda su potencia y versatilidad.

C# combina los mejores elementos de múltiples lenguajes de amplia difusión como C++, Java, Visual Basic o Delphi. De hecho, su creador Anders Heljsberg fue también el creador de muchos otros lenguajes y entornos como Turbo Pascal, Delphi o Visual J++. La idea principal detrás del lenguaje es combinar la potencia de lenguajes como C++ con la sencillez de lenguajes como Visual Basic, y que además la migración a este lenguaje por los porgramadores de C/C++/Java sea lo más inmediata posible.

Además de C#, Microsoft propociona Visual Studio.NET, la nueva versión de su entorno de desarrollo adaptada a la plataforma .NET y que ofrece una interfaz común para trabajar de manera cómoda y visual con cualquiera de los lenguajes de la plataforma .NET (por defecto, C++, C#, Visual Basic.NET y JScript.NET, aunque pueden añadirse nuevos lenguajes mediante los plugins que proporcionen sus fabricantes).

El propio autor de este artículo ha publicado un excelente libro de C# donde se puede ampliar toda esta información y, por supuesto, aprender el lenguaje. Consta de unas 260 páginas y ha sido escrito con la idea de que cualquiera con unos conocimientos mínimos de programación pueda seguirlo, aunque sería recomendable que el lector conociese C++, Java o lenguajes similares.

Puede ser descargado, junto con otra documentación, de la web del autor: http://www.josanguapo.com/

Qué es .NET

Microsoft.NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años -y cuyo lanzamiento definitivo es inminente, estando ya disponible su primera versión Release Candidate- con los objetivos de:

- Mejorar sus sistemas operativos
- Mejorar su modelo de componentes COM+
- Obtener un entorno específicamente diseñado para el desarrollo y ejecución del software en forma de servicios que puedan ser tanto publicados como accedidos a través de Internet de forma independiente del lenguaje de programación, modelo de objetos, sistema operativo y hardware utilizados tanto para desarrollarlos como para publicarlos. Éste entorno es lo que se denomina la plataforma.NET, y los servicios antes mencionados son a los que se denomina servicios web.

Para el desarrollo y ejecución de aplicaciones en este nuevo entorno tecnológico Microsoft proporciona el conjunto de herramientas conocido .NET Framework SDK, que es posible descargarlo gratuitamente de su sitio web http://www.msdn.microsoft.com/net e incluye compiladores de lenguajes como C#, Visual Basic.NET, Managed C++ y JScript.NET específicamente diseñados para crear aplicaciones para él.

El corazón de la plataforma.NET es el CLR (Common Language Runtime), que es una aplicación similar a un máquina virtual que se encarga de gestionar la ejecución de las aplicaciones para ella escritas. A estas aplicaciones les ofrece numerosos servicios que facilita su desarrollo y mantenimiento y favorecen su fiabilidad y seguridad. Entre ellos los principales son:

- Modelo de programación consistente y sencillo, completamente orientado a objetos.
- Eliminación del temido problema de compatibilidad entre DLLs conocido como "infierno de las DLLs"
- Ejecución multiplataforma
- Ejecución multilenguaje, hasta el punto de que es posible hacer cosas como capturar en un programa escrito en C# una excepción escrita en Visual Basic.NET que a su vez hereda de un tipo de excepción escrita en Cobol.NET. Aunque más arriba se ha dicho que en el .NET Framework sólo se ofrecen compiladores de C#, MC++, VB.NET y JScript.NET, lo cierto es que aparte Microsoft y terceros han -o están- desarrollado versiones adaptadas a .NET de muchísimos otros lenguajes como APL, CAML, Cobol, Eiffel, Fortran, Haskell, Java, Mercury, ML, Mondrian, Oberon, Oz, Pascal, Perl, Python, RPG, Scheme o Smalltalk
- Recolección de basura
- Aislamente de memoria entre procesos y comprobaciones automáticas de seguridad de tipos en las conversiones
- Soporte multihilo
- Gestión del acceso a objetos remotos que permite el desarrollo de aplicaciones distribuidas de manera transparente a la ubicación real de cada uno de los objetos utilizados en las mismas.
- Seguridad avanzada, hasta el punto de que es posible limitar los permisos de ejecución del código en función de su procedencia (Internet, red local, CD-ROM, etc.), el usuario que lo ejecuta o la empresa que lo creó.

- Interoperabilidad con código preexistente, de manera que es posible utilizar con facilidad cualquier librería de funciones u objetos COM y COM+ creados con anterioridad a la aparición de la plataforma .NET
- Adecuación automática de la eficiencia de las aplicaciones a las caracterísitcas concretas de cada máquina donde se vaya a ejecutar

El propio autor de este artículo ha publicado un excelente libro donde podremos encontrar una descripción de .NET mucho más amplia. El libro en concreto trata sobre el lenguaje C#, desarrollado específicamente para la plataforma .NET.

Puede ser descargado, junto con otra documentación, de la web del autor: http://www.josanguapo.com/

Qué es JSP

JSP es un acrónimo de Java Server Pages, que en castellano vendría a decir algo como Páginas de Servidor Java. Es, pues, una tecnología orientada a crear páginas web con programación en Java.

> **Bibliografía:** Esta descripción de JSP está extraída de un PDF en inglés muy completo que introduce la tecnología, que se puede encontrar el la página corporativa de Java de Sun Microsystems, en su sección de aprendizaje online. A su vez, dicho manual proviene del portal Java **jGuru**.
>
> jGuru: Introduction to JavaServer Pages technology

Con JSP podemos crear aplicaciones web que se ejecuten en variados servidores web, de múltiples plataformas, ya que Java es en esencia un lenguaje multiplataforma. Las páginas JSP están compuestas de código HTML/XML mezclado con etiquetas especiales para programar scripts de servidor en sintaxis Java. Por tanto, las JSP podremos escribirlas con nuestro editor HTML/XML habitual.

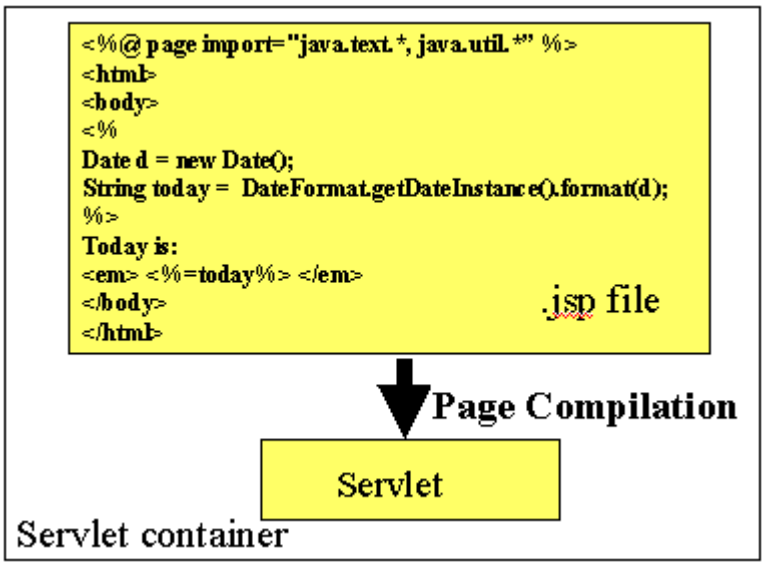
Motor JSP

El motor de las páginas JSP está basado en los servlets de Java -programas en Java destinados a ejecutarse en el servidor-, aunque el número de desarrolladores que pueden afrontar la programación de JSP es mucho mayor, dado que resulta mucho más sencillo aprender que los servlets.

En JSP creamos páginas de manera parecida a como se crean en ASP o PHP -otras dos tecnologías de servidor-. Generamos archivos con extensión .jsp que incluyen, dentro de la estructura de etiquetas HTML, las sentencias Java a ejecutar en el servidor. Antes de que sean funcionales los archivos, el motor JSP lleva a cabo una fase de traducción de esa página en un servlet, implementado en un archivo class (Byte codes de Java). Esta fase de traducción se lleva a cabo habitualmente cuando se recibe la primera solicitud de la página .jsp, aunque existe la opción de precompilar en código para evitar ese tiempo de espera la primera vez que un cliente solicita la página.

Ejemplo de página JSP

En la imagen siguiente se puede ver un ejemplo extremadamente simple de una página JSP y el esquema de conversión de esa página en un servlet.



Prerequisitos

Para aprender JSP, aparte de conocer HTML, será necesario comprender y tener algo de experiencia en la programación en Java, que es un lenguaje de programación Orientado a Objetos por completo. Una vez conocida la programación en Java se puede estudiar por encima el sistema de Servlets, lo que nos dará una mejor idea del funcionamiento interno del motor JSP.

Para aprender Java podemos consultar algunos enlaces del correspondiente directorio de nuestro buscador de enlaces.

Además, necesitaremos descargar e instalar Tomcat, el contenedor de servlets usado en la referencia oficial de implementación de JSP.

Qué es Oracle

Oracle es básicamente una herramienta cliente/servidor para la gestión de Bases de Datos. Es un producto vendido a nivel mundial, aunque la gran potencia que tiene y su elevado precio hace que sólo se vea en empresas muy grandes y multinacionales, por norma general. En el desarrollo de páginas web pasa lo mismo: como es un sistema muy caro no está tan extendido como otras bases de datos, por ejemplo, Access, MySQL, SQL Server, etc.

Vamos ahora en centrarnos en que es Oracle exactamente y como funciona la programación sobre éste. Oracle como antes he mencionado se basa en la tecnología cliente/servidor, pues bien, para su utilización primero sería necesario la instalación de la herramienta servidor (Oracle 8i) y posteriormente podríamos atacar a la base de datos desde otros equipos con herramientas de desarrollo como Oracle Designer y Oracle Developer, que son las herramientas básicas de programación sobre Oracle.

Para desarrollar en Oracle utilizamos PL/SQL un lenguaje de 5ª generación, bastante potente para tratar y gestionar la base de datos, también por norma general se suele utilizar SQL al crear un formulario.

> **Referencia:** Podemos aprender qué es el lenguaje SQL en un artículo de DesarrolloWeb.com. Además, existe un manual de SQL para el que desee profundizar.

Es posible lógicamente atacar a la base de datos a través del SQL plus incorporado en el paquete de programas Oracle para poder realizar consultas, utilizando el lenguaje SQL.

El Developer es una herramienta que nos permite crear formularios en local, es decir, mediante esta herramienta nosotros podemos crear formularios, compilarlos y ejecutarlos, pero si queremos que los otros trabajen sobre este formulario deberemos copiarlo regularmente en una carpeta compartida para todos, de modo que, cuando quieran realizar un cambio, deberán copiarlo de dicha carpeta y luego volverlo a subir a la carpeta. Este sistema como podemos observar es bastante engorroso y poco fiable pues es bastante normal que las versiones se pierdan y se machaquen con frecuencia. La principal ventaja de esta herramienta es que es bastante intuitiva y dispone de un modo que nos permite componer el formulario, tal y como lo haríamos en Visual Basic o en Visual C, esto es muy de agradecer.

Los problemas anteriores quedan totalmente resueltos con Designer que es una herramienta que se conecta a la base de datos y por tanto creamos los formularios en ella, de esta manera todo el mundo se conecta mediante Designer a la aplicación que contiene todos los formularios y no hay problemas de diferentes versiones, esto es muy útil y perfecto para evitar machacar el trabajo de otros. Pero el principal y más notable problema es la falta de un entorno visual para diseñar el formulario, es decir, nos aparece una estructura como de árbol en la cual insertamos un formulario, a la vez dentro de éste insertamos bloques o módulos que son las estructuras que contendrán los elementos del formularios, que pueden estar basados en tablas o no.

Por lo tanto si queremos hacer formularios para practicar o para probar qué es esto de Oracle, os recomiendo que uséis Developer pues es mucho más fácil e intuitivo al principio.

Qué es ActiveX

ActiveX es una tecnología de Microsoft para el desarrollo de páginas dinámicas. Tiene presencia en la programación del lado del servidor y del lado del cliente, aunque existan diferencias en el uso en cada uno de esos dos casos.

En el cliente:

Son pequeños programas que se pueden incluir dentro de páginas web y sirven para realizar acciones de diversa índole. Por ejemplo hay controles ActiveX para mostrar un calendario, para implementar un sistema de FTP, etc.

Son un poco parecidos a los Applets de Java en su funcionamiento, aunque una diferencia fundamental es la seguridad, pues un Applet de Java no podrá tomar privilegios para realizar acciones malignas (como borrarnos el disco duro) y los controles ActiveX sí que pueden otorgarse permisos para hacer cualquier cosa.

Los controles ActiveX son particulares de Internet Explorer.

En el servidor

También existen controles ActiveX del servidor y la gente que conozca ASP seguro que los utiliza ya, aunque sea sin darse cuenta. Por ejemplo, cuando realizamos una conexión con una base de datos, estamos utilizando un control ActiveX del servidor.

Desarrollo de ActiveX

Los controles ActiveX se desarrollan con entornos de Microsoft para la creación de aplicaciones Windows, como pueden ser Visual Basic Script o Visual C. Se nos escapa totalmente de este artículo el explicar algo del método de desarrollo, pero lo que si cabe señalar es que existen muchos controles ActiveX tanto del lado del servidor como del cliente, que están ya desarrollados y podemos incluirlos fácilmente en nuestras creaciones.

Qué son las Extensiones de Frontpage

Las extensiones de Frontpage son pequeñas funcionalidades añadidas por dicho programa a las páginas web básicas. Frontpage las pone a disposición del diseñador de páginas para que las incluya en su sitio web, si así lo desea.

Las extensiones de Frontpage son, por ejemplo, contadores, formularios que realizan pequeñas tareas automáticamente o ayudas para la publicación de la web. En definitiva, son pequeños scripts de servidor para hacernos la vida más sencilla o ampliar las funcionalidades de nuestras páginas.

Las tareas de publicación de la web se refiere a una manera muy cómoda de trabajar directamente con el servidor de alojamiento por la que a medida que vamos construyendo la página esta se va actualizando directamente en el servidor web, de modo que no tendremos que realizar acción alguna para publicar nuestro trabajo una vez realizado sino que ya estará publicado en Internet y accesible a todos los usuarios.

Instalar las extensiones de Frontpage

Para poder utilizar correctamente las extensiones, deben estar instaladas en el servidor donde vamos a publicar las páginas web. Si no disponemos de las extensiones en el servidor las funcionalidades avanzadas como los contadores o las asociadas a los formularios no estarán disponibles y puede darse el

caso de que en nuestro ordenador local sí que se vean bien y que en el servidor de Internet no.

Las extensiones se deben instalar de manera distinta dependiendo del alojador que hayamos contratado. No todos los proveedores de Hosting permiten el uso de las extensiones de Frontpage por lo que si estamos pensando utilizarlas es muy importante que nos informemos de si se permiten en el proveedor antes de contratar el espacio. No obstante, cabe señalar que casi todos los proveedores ofrecerán la posibilidad de instalarlas, aunque en muchos casos deberemos solicitar que las instalen o hacerlo nosotros mismos.

Para instalarlas es necesario que preguntemos en nuestro alojador cómo realizar esas acciones. Es posible que a través del típico panel de control del dominio se permita realizar la instalación. En cualquier caso, si están instaladas deberían aparecer un par de directorios llamados _vti_cnf y _vti_pvt. Lo más seguro que si los copiamos directamente o subimos por FTP dichas carpetas no nos sirva de mucho, así que informémonos como realizar la instalación antes de meter la pata.

Apéndice II
Frontpage 2000 para principiantes

Un análisis en profundidad de Frontpage y la respuesta a las preguntas más habituales planteadas por los usuarios novatos de este programa.

Introducción a Frontpage para principiantes

Es relativamente común oír por la Red, o entre colegas diseñadores —cuando se tienen amigos tan extraños como uno mismo—, que las peores páginas de Internet no fueron desarrolladas por un mal webmaster, sino escritas con FrontPage. Y tristemente es muy probable que lleven razón. (¿Triste? De momento retiro el adjetivo; no se puede sentir lástima de un producto fabricado por la mayor compañía de software del mundo.) ¿Por qué es así? Además, tengamos en cuenta que es una herramienta muy, muy extendida; ¿pueden equivocarse cientos de miles, quizás millones de personas? La respuesta a esta pregunta es sí, pueden equivocarse, y la otra cuestión requiere una argumentación un poco más extensa.

FrontPage está programado por los empleados de Microsoft, la madre amantísima de Internet Explorer, y salta a la vista que se esmeraron en que los usuarios de Netscape nos sintiésemos defraudados. A eso se le llama simpatía corporativa. Pero ¿qué más da? Siempre nos quedará el semejante y más neutral y accesible Namo Web Editor. FrontPage genera lo que podríamos llamar código extraño, una especie de tiras de texto cifrado, cabalístico, que sólo utiliza el programa y seguramente nadie más en el mundo. A eso lo define como código fuente. El problema es que, además, FrontPage crea archivos y etiquetas basura, que aumentan el peso de nuestros websites y el tiempo de transferencia y publicación.

A pesar de todo esto, es una aplicación que inspira en los usuarios la simpatía de lo conocido. Forma parte de la suite Microsoft Office y su interfaz es muy parecida a la del omnipresente Word y a la del resto de productos de la gama. Por otra parte, sus archivos de ayuda son muy completos. Esto sin duda es una ventaja para el principiante. Sin embargo, el profesional del medio se encontrará con la imposible tarea de construir un palacio con herramientas de plástico, a diferencia de aquél que ejecuta software del prestigio y la calidad contrastada de Macromedia DreamWeaver o Adobe GoLive!, tan bien integrados, por si fuera poco, con otras aplicaciones comunes y muy potentes, de gran calidad visual y multimedia, como Flash y PhotoShop.

Pero siendo un producto muy usado, especialmente por aficionados cuyas finanzas no dependen de su talento creativo, se merece al menos una introducción a algunas de sus funciones. No es un manual para un uso avanzado, sino un repaso elemental de las preguntas más frecuentes que los usuarios plantean en los foros de discusión, así como una manera de hacer compatible su código con otros navegadores. Una vez instalado e iniciado el programa —digamos que cuando ya es demasiado tarde para echarse atrás—, nos encontramos con la pantalla principal de modo WYSIWYG; siglas en inglés de "lo que ves es lo que obtienes". La reconocemos porque en la parte inferior de la pantalla, sobre la barra de estado, vemos seleccionada la pestaña "normal"; observemos que hay dos más: "html" y "vista previa". La vista html nos muestra el código fuente, que es lo que interpreta el navegador, y es especialmente importante en un producto de las características de FrontPage, dado que su código debe ser revisado y corregido para adaptarlo a navegadores diversos y para incluir código javascript, por ejemplo. La última pestaña, vista previa, nos muestra el resultado tal y como se verá en Internet Explorer, y además ejecuta una función de depuración de código: si hay errores, nos lo hará saber mediante una ventana emergente.

Barras de herramientas de Frontpage

Existen varias barras de herramientas, y la mejor idea es probar todas ellas, con todas sus funciones, hasta descubrir cuáles son las que realmente resultan útiles de forma frecuente, y aquellas de uso más inusual. Al final es normal que creemos nuestra propia barra, para eliminar del escritorio elementos innecesarios.

Las barras de herramientas más comunes para el usuario principiante son: estándar, con los botones típicos de guardar, abrir, página nueva, deshacer, rehacer, etc.; formato, para aplicar tipografías y párrafos al texto (FrontPage admite la justificación de párrafos, pero solo desde Formato> párrafo> alineación, sin disponer de un botón específico en la barra de formato); tablas, para construir tablas y modificar sus propiedades; imágenes, para insertar gráficos, cambiarles el tamaño y la resolución, crear mapas de imágenes, modificar luz, contraste, etc.; efectos DHTML, para aplicar animaciones automáticas o de reacción a las páginas; exploración, para controlar la estructura de un sitio web; e informes, que nos detalla las páginas lentas, anticuadas, etc.

Existen otras barras de herramientas, como por ejemplo de formulario, pero requieren ciertos conocimientos adicionales y, de momento, el usuario novel no las necesita... ni entiende.

Creando un web en local con Frontpage

Para empezar, hay que tener claras dos nociones elementales y sencillas.

- Una página web es un único archivo, una estructura de datos e informaciones que instruyen al navegador sobre cómo debe mostrar dicha página.
- **Un sitio web** es una carpeta o grupo de carpetas que contiene a su vez páginas web individuales y, posiblemente, otros archivos como imágenes, animaciones Flash, documentos JavaScript, películas QuickTime o Real, etc.

En FrontPage debemos crear primero una carpeta web, que es algo así como la versión local, en nuestro disco duro, del sitio web accesible al público y colocado en un servidor. En realidad nunca deben crearse páginas arbitrariamente, en un folder cualquiera, ya que eso dificulta considerablemente la tarea de publicación. La operación es fácil: clic sobre archivo, clic sobre nuevo, clic sobre web... La aplicación nos da la opción de elegir entre algunas plantillas; mi opción predilecta, más libre, es "web de una sola página", de modo que yo me encargo de todo el trabajo posterior, asumiendo un control absoluto.

Se introduce la ubicación y el nombre de la carpeta web, en el campo al efecto, y se acepta. Unos segundos después, ya disponemos de nuestra web local, donde FrontPage ubicará todas las páginas y el resto de archivos. Así, al publicar, no tendremos pérdida.

Definiendo la resolución con Frontpage

Un problema fundamental del diseño web es la proyección de nuestras páginas, es decir, cómo pensamos que serán vistas por el usuario. ¿Qué tamaño tiene su pantalla? ¿Es igual que la nuestra, más grande?

Inevitablemente ocurrirá una de las dos cosas, e incluso ambas. Aunque es cierto que los monitores de 800x600 son muy comunes, también lo son los de 17". Por ello, si pensamos en el primero, dejaremos vacío gran parte del espacio del usuario de 17"; y si pensamos en éste, obligaremos al navegante de 15" a utilizar las barras de desplazamiento horizontal para moverse por una página, y eso es muy incómodo.

> Referencias: Tenemos algunos artículos que tratan el tema de las resoluciones de pantalla en páginas web con más detenimiento y de manera general a todos los programas:
>
> - Definiciones de pantalla. Cómo tratarlas.
> - Páginas fluidas (las que se ajustan al tamaño de la ventana del explorador) Cuando son apropiadas.

La solución predilecta del diseñador suele consistir en crear tablas con un ancho y alto de tamaño definido porcentualmente, es decir, adaptable a cada monitor; el 100% es una buena opción. El procedimiento es simple: Tablas > Insertar: a continuación aparece una ventana, donde introducimos a voluntad ciertos datos, como grosor del borde, número de filas (horizontales) y columnas (verticales), color o imagen de fondo (la imagen de fondo en las tablas ya es reconocida en Netscape 7). En esta primera ventana podemos definir el ancho: 100%, esto es, adaptable al monitor. Una vez que aceptemos, ya disponemos de nuestra tabla en la pantalla.

Ahora, haciendo clic con el botón derecho del ratón sobre la misma tabla, seleccionamos Propiedades de tabla en el menú contextual. Ahora sí, definimos también el alto en 100%. Conviene advertir, en todo caso, que este código es interpretado por Internet Explorer, pero no por Netscape en ninguna de sus versiones. Para hacerlo accesible a todos, podemos cambiar a la vista html y sustituir la tabla (definida por las etiquetas <table> </table>) por este código:

```
<table cellpadding="0" cellspacing="0" border="0" width="100%" height="100%" style="text-align:
left;">
<tbody>
<tr>
<td valign="top">
</td>
</tr>
</tbody>
</table>
```

Este codigo ha sido generado por Netscape Composer y será asimilado por FrontPage, con posibilidad de editarse normalmente sin que se pierda su compatibilidad (conviene que, en FrontPage, sigamos estas instrucciones: abrimos Herramientas> Opciones de página> pestaña Código fuente html y seleccionamos "mantener el código fuente html", de manera que FrontPage no rescriba el código generado por otras aplicaciones). Ahora, todo lo que compongamos dentro de la tabla se adaptará a los diversos monitores, siempre quedando dentro de la pantalla.

Si tan sólo queremos definir el ancho de la tabla, nos limitamos a seguir las primeras instrucciones. En todo caso, ningún usuario se verá obligado a mover el scroll horizontal cuando navegue nuestros sitios. (A excepción de aquellas ocasiones en que incluyamos objetos muy grandes sin posibilidad de reestructuración automática, como por ejemplo las imágenes; si bien sería posible definir su tamaño como porcentaje de la ventana, esta opción tan solo es útil en gráficos sin degradados, ya que las fotografías quedarían deformadas y pixelizadas).

Funciones especiales de Frontpage: Las extensiones de servidor

Conviene precisar que esta aplicación de Microsoft incluye algunas opciones particulares, muy cómodas e interesantes, como por ejemplo la creación de foros totalmente personalizados o la inclusión de contadores. Sin embargo, cuenta con la desventaja de que estas características dependen del servidor donde alojemos nuestras páginas, y si éste no cuenta con las extensiones apropiadas, la mejor idea es desechar todas estas funciones adicionales... porque sencillamente no funcionarán. Si ya conocemos nuestro proveedor de espacio web, gratuito o de pago, podemos consultar sus servicios y ver si cuentan con estas extensiones; normalmente muestran esta información de manera explícita, ya que es una pregunta muy común. Si no disponen de las extensiones, podemos deshabilitarlas todas de manera automática...

Hacemos clic en el menú Herramientas> Opciones de página> pestaña Compatibilidad. Deseleccionamos la casilla "habilitado con las extensiones de servidor de Microsoft FrontPage". Así, sus las funciones asociadas quedan atenuadas e inoperativas en los menús, de manera que no nos ocurrirá algo muy común e irritante: descubrir al final que parte de nuestro trabajo ha resultado en vano.

En esta ventana, además, contamos con otras opciones muy útiles, y con algunas más que podrían sonar a chiste. La mejor idea suele consistir en seguir estas instrucciones:

—En la entrada Exploradores, seleccionar "personalizado". Es de muy mal gusto planear un web pensando tan sólo en un grupo cerrado de usuarios.

—En la entrada Versiones de exploradores, seleccionar "personalizado". En FrontPage 2000 se tratan tan sólo hasta las versiones 4.0, mientras que en la actualidad Internet Explorer avanza por la 6, y Netscape, por la 7.

En el listado de tecnologías cada usuario puede elegir las opciones que le convenzan. Para el caso de ActiveX, por ejemplo, hay que considerar el hecho de que éste no es soportado por Netscape, mientras que existen otras opciones compatibles con ambos navegadores, si lo que buscamos es incluir contenido multimedia en nuestras webs.

Multimedia con FrontPage 2000

Como decía, los controles ActiveX son particulares de Internet Explorer. Se trata de pequeños códigos visibles en la fuente del documento que indican al navegador ciertas instrucciones para definir el comportamiento de contenido multimedia y dinámico. En general suelen ser entradas complicadas para el usuario novel, y llegan a variar mucho de un control a otro; es decir, quizás tengas los conocimientos suficientes para incluir flash, pero no los apropiados para añadir RealVideo. Si no se conocen, son inútiles. En todo caso, usualmente incluyen la opción de "representación alternativa" para exploradores que no soportan esta tecnología.

Entonces, **¿qué hacer para insertar músicas y contenidos especiales?** Como curiosidad valga decir que la ventana propiedades de página (fácilmente accesible haciendo clic con el botón secundario del ratón sobre la pantalla principal) tiene la opción de música de fondo, pero en general es una mala idea. Ocupa ancho de banda, puede resultar pesada y la mejor alternativa por su escaso peso, la música midi, suena soporífera e insustancial.

Bien, la opción entonces es la inclusión de complementos (Insertar> Avanzadas> Complemento), que generará unas etiquetas <embed> </embed> interpretadas tanto por Netscape como por Explorer. Las entradas que debemos cumplimentar en la ventana que aparece son los siguientes: "origen de datos",

que determina la fuente o archivo de origen; el tamaño en alto y ancho (aunque puede ocultarse el complemento si se desea, seleccionando la casilla Ocultar, que creará la etiqueta "hidden"); y ciertas variables de diseño que pueden ser editadas más tarde, y más cómodamente, desde los controles de la barra de formato.

¿Qué podemos hacer con los complementos o archivos embebidos? Pues en realidad agregar cualquier archivo que un navegador sea capaz de interpretar mediante plugins; desde Flash de Macromedia hasta QuickTime o músicas mp3. Sin embargo, cada archivo específico tiene unas características particulares cuyo código debe ser conocido y editado desde la vista de código fuente. Por ejemplo, las animaciones flash incluyen menu="0/—1", que muestra o no, el menú contextual, etc. Quizás pueda verse de una forma más clara el resultado del código, que deberá editarse manualmente, en la siguiente línea:

<embed src="archivo.xxx" width=320 height=200 AUTOSTART=true LOOP=true></embed>

Desde FrontPage podremos añadir autostart, que indicará si el inicio de la reproducción del archivo es automático o corresponde a un evento, y loop, que define si el fichero se ejecuta indefinidamente o si tan solo se reproduce una vez y luego se detiene. Existen más variables genéricas y otras específicas de formato; en todo caso, la página de desarrollo de Netscape incluye información complementaria muy interesante.

HTML limpio

Como ya se dijo anteriormente, FrontPage trabaja también en modo vista de código fuente, que es completamente editable y cuyos resultados se actualizan en tiempo real en la vista WYSIWYG. Sin embargo, el usuario novel, para quien va destinada esta introducción elemental, quizás no conozca dicho lenguaje, y lo que desea hacer es simplemente cortar y pegar códigos que tal vez ha bajado de la Red. **¿Es esto posible?** Desde luego, y de hecho de dos formas distintas. Para una de ellas usaremos el comando Insertar> Avanzadas> Html, que abre una ventana en blanco donde pegar cualquier secuencia. Antes deberemos especificar con el ratón dónde queremos insertar el nuevo código. No obstante, éste se incluirá según está escrito, sin análisis, y su contenido no se mostrará en la vista normal.

Para eliminar esta característica deberemos borrar los comentarios que el software genera automática. Acudimos a la vista html y borramos completamente y sin temor las líneas: <!--webbot bot="HTMLMarkup" startspan -->, endspan -->, al principio y al final.

La otra forma la usaremos para insertar código en el encabezado del documento, que está delimitado por las etiquetas <head>,</head>, donde la función insertar html no es efectiva.

Para ello pasamos a la vista de código, donde pegamos cualquier código de la forma clásica, abriendo un menú contextual con el ratón y seleccionando pegar. No obstante, puede haber un problema: si el texto copiado tiene formato, FrontPage lo conservará, haciendo anotaciones de formateado, e inutilizará el script. Puesto que corregirlo manualmente sería un engorro, podemos seguir un pequeño truco: primero pegamos el texto en el bloc de notas, de manera que se pierde el formato, y allí volvemos a copiar, apareciendo limpio de etiquetas basura.

Conclusiones:

Bien, éstas son algunas de las preguntas que comúnmente se pueden leer en los foros de usuarios de FrontPage, y aquí exponemos algunas posibles respuestas. No son todas, ni siquiera tienen que ser infalibles; hay más opciones creativas que el usuario puede descubrir con la práctica, disfrutando así de un trabajo hecho con cierto ingenio.

Apendice III
Instalación de IIS en Windows XP profesional

Descripción detallada del proceso de instalación y configuración de Internet Information Server en Windows XP profesional. Conceptos básicos necesarios para empezar la administración.

Internet Information Server (IIS) es el servidor de páginas web avanzado de la plataforma Windows. Se distribuye gratuitamente junto con las versiones de Windows basadas en NT, como pueden ser Windows 2000 Profesional o Windows 2000 Server, así como Windows XP, también en sus versiones Profesional y Server.

Estas normas de instalación son aplicables, a nivel general, a las que podemos encontrarnos en las distintas versiones de los sistemas operativos comentados antes, si bien hemos tomado Windows XP profesional para relatar los pasos y tomar las imágenes de las pantallas.

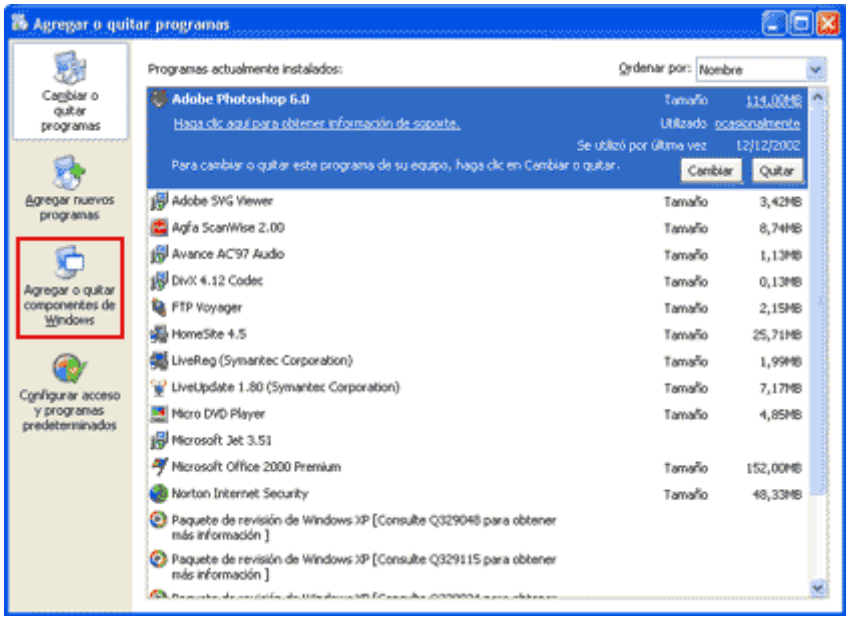
> **Nota:** Windows 95, 98, las versiones Home, de Windows XP, y ME, de Windows 2000, no se admite la instalación de IIS. En su lugar podemos probar a instalar el Personal Web Server.

Agregar componentes adicionales de Windows

IIS se puede encontrar en el propio CD de instalación de Windows XP Profesional. Hay que acceder a la opción de "Instalar componentes opcionales de Windows" para poder cargarlo en nuestro sistema. Para ello tenemos dos opciones:

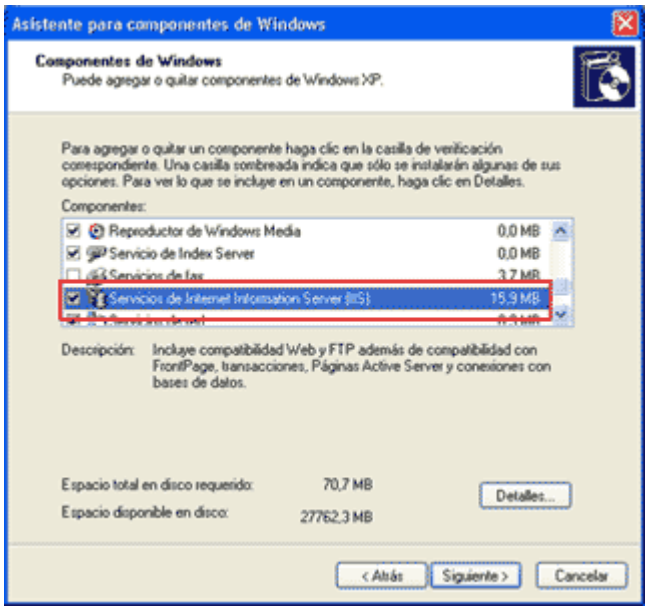
1) Insertar el CD de instalación de Windows y en la ventana de autoarranque que se muestra, seleccionar la opción que pone "Instalar componentes opcionales de Windows"

2) En el Panel de control, seleccionar la opción de "Agregar o quitar programas" y en la ventana que sale, pulsar sobre el icono de la izquierda marcado como "Seleccionar o quitar componentes de Windows".



Ahora nos muestra la ventana para seleccionar los componentes adicionales de Windows que hay disponibles. En la lista, marcamos la opción "Servicios de Internet Information Server (IIS)". Por defecto se seleccionan unos cuantos componentes, dentro de los que ofrece la instalación de IIS. Nosotros podemos elegir qué componentes deseamos instalar apretando el botón marcado como "Detalles". Entre los componentes posibles se encuentran las extensiones de Frontpage, documentación, servicios adicionales de IIS, un servidor de FTP (para la transferencia de ficheros con el servidor por FTP), incluso uno de SMTP (para el envío de correos electrónicos).

Si no sabemos qué componentes instalar podemos dejar las opciones como aparecen en un principio, pues para la mayoría de los casos serán válidas. Sólo un detalle: puede ser adecuado no instalar las extensiones de Frontpage en caso de que no pensemos que se vayan a utilizar.



Una vez hemos instalado los componentes deseados, podemos y apretar el botón de "Siguiente" para comenzar la instalación, que se alargará unos minutos.

Acceder al servidor web

Podemos acceder al servidor web para comprobar si se ha instalado correctamente IIS. Para ello simplemente debemos escribir http://localhost en Internet Explorer y debería aparecer una página web informando que IIS está correctamente instalado. Además, aparecerá la documentación de IIS en una ventana emergente, si es que fue instalada.

Sitio web predeterminado en IIS

Lo que se muestra cuando accedemos a http://localhost es el sitio web predeterminado, que se guarda en nuestro disco duro, concretamente en la carpeta C:\Inetpub\wwwroot

Si accedemos a dicha carpeta desde Mi PC podremos ver los archivos que se están sirviendo como sitio web predeterminado. Encontraremos, entre otros archivos uno llamado "iisstart.asp" que es el que se pone en marcha al acceder a este directorio.

Colocar nuestras propias páginas

Lo lógico ahora es que deseemos colocar nuestras propias páginas web para que las sirva IIS. Si nuestro sitio web es bastante simple podríamos colocar todos los archivos dentro de la carpeta del sitio web predeterminado.

Por ejemplo, para hacer una prueba, podríamos colocar un archivo llamado "hola.asp" en la carpeta C:\Inetpub\wwwroot. Para acceder a este archivo desde nuestro explorador deberíamos escribir la dirección http://localhost/hola.asp

Si deseamos un código simple para el archivo hola.asp, en el que se pruebe si está funcionando o no el motor ASP, podemos utilizar el siguiente:

```
<html>
<head>
<title>Pobando ASP</title>
</head>
<body>
<%
for i=1 to 7
response.write "<font size=" & i & ">Hola Mundo!</font><br>"
next
%>
</body>
</html>
```

> **Atención:** No se debe acceder al archivo utilizando una ruta como esta: C:\Inetpub\wwwroot\hola.asp, pues de esa manera no se estaría pasando a través del servidor web y la página ASP no se ejecutaría.

Si tuviéramos un sitio medianamente complejo, convendría crear una carpeta dentro de wwwroot con el nombre de nuestro sitio y dentro podríamos colocar todos los archivos. Si el directorio tuviera una ruta como C:\Inetpub\wwwroot\mi_web, accederíamos colocando esta dirección en nuestro navegador: http://localhost/mi_web.

Cuando se accede a ese directorio se sirve el documento por defecto que se haya configurado en el

servidor web. El documento por defecto es aquel que se sirve si no se ha especificado ninguno en la ruta de acceso, es decir, si no indicamos ningún archivo en la dirección que colocamos en Internet Explorer (una dirección como http://localhost/mi_web especifica un directorio, pero ningún archivo) se sirve el documento por defecto.

> **Referencia:** Se habla más sobre el documento por defecto en un artículo del manual de publicar en Internet. http://www.desarrolloweb.com/articulos/200.php?manual=3

En principio, el documento por defecto en IIS está configurado a Default.asp o Default.htm. Esto quiere decir que deberíamos colocar un archivo con uno de esos nombres en nuestro directorio para que se sirva si el usuario no indica ningún nombre de archivo. Luego veremos cómo cambiar esta configuración.

Para probar todo esto, podemos crear un archivo llamado Default.asp y guardarlo en nuestro directorio C:\Inetpub\wwwroot\mi_web. El código que podríamos utilizar sería por ejemplo este:

```
<html>
<head>
<title>Archivo por defecto en mi_web</title>
</head>
<body>
<h1>Archivo por defecto en mi_web</h1>
<%
for i=0 to 9
response.write i
next
%>
</body>
</html>
```

Recordar que para ver este archivo habría que componer una dirección como esta http://localhost/mi_web. Automáticamente se sirve el archivo Default.asp, aunque no se especifique nada, pues ese es el documento por defecto. También podríamos acceder al archivo especificando su ruta completa: http://localhost/mi_web/Default.asp

> **Nota:** Podemos llamar al archivo Default.asp o bien default.asp (con mayúscula o minúscula en su inicial). Cualquiera de las dos opciones es válida, pues en Windows no se tienen en cuenta las mayúsculas y minúsculas en nombres de archivos.

Administración de IIS

Para administrar el servidor Internet Information Server en Windows XP, disponemos de un panel de control llamado "Servicios de Internet Information Server" al que podemos acceder de varias maneras.

1) Pulsando con el botón derecho en MI PC y seleccionando la opción que pone "Administrar". Esto nos abre "Microsoft Management Console" o, dicho en castellano, la "Administración de equipos". En la lista de la izquierda, en la parte de abajo aparece "Servicios y aplicaciones", entre los que encontraremos una opción buscada: "Servicios de Internet Information Server"

2) Podemos acceder desde el panel de control. Si tenemos configurada la vista clásica encontraremos un icono que pone "Herramientas administrativas" y haciendo doble clic, encontraremos el icono para administrar IIS. Si teníamos configurada la vista por categorías del panel de control (la que aparece por defecto en Windows XP) la búsqueda de la opción es un poco más compleja: Seleccionamos "Rendimiento y mantenimiento" y dentro ya encontraremos el icono de "Herramientas administrativas", al que teníamos que hacer doble clic para encontrar, entre otros, el icono para acceder a "Servicios de Internet Information Server".

3) Otra manera de acceder aparece en la ayuda de Internet Information Server. Se trata de hacer una búsqueda del archivo llamado "inetmgr.exe". Una vez localizado se puede ejecutar y aparece la consola de administración de IIS. Si se desea, se puede hacer un acceso directo a dicho archivo para no tener que buscarlo cada vez que se desee ejecutar.

Una vez hemos accedido al panel "Servicios de Internet Information Server" tenemos ante nosotros la posibilidad de configurar nuestro servidor web en muchos aspectos, por ejemplo podemos, definir el documento por defecto, crear directorios virtuales, modificar las opciones de seguridad, etc.

Definir el documento por defecto en IIS

Hablamos antes sobre el documento por defecto, que en IIS viene definido en un principio en los archivos default.asp, default.htm o index.htm. Estos archivos son muy específicos de Windows, pero no suelen utilizarse en la mayoría de los proveedores, así que es mejor que utilicemos un documento por defecto idéntico al que utilizan la mayoría de los proveedores de hosting.

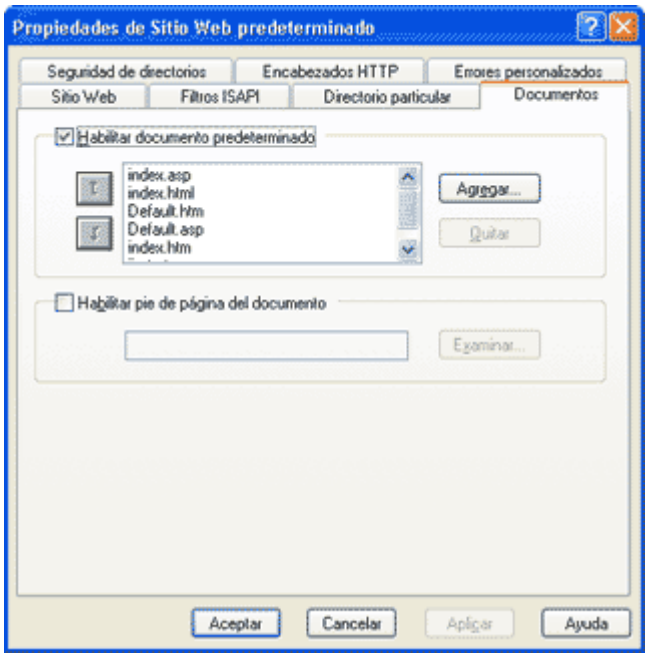
Si nuestro documento por defecto no es el mismo en local (nuestro servidor IIS) y en remoto (espacio en nuestro albergue en un servidor web conectado permanentemente a Internet), puede que tengamos que realizar algunas modificaciones en los nombres de los archivos, para que todo funcione perfectamente al subir el sitio web a Internet, lo que resultaría un engorro adicional e innecesario, de haber configurado nuestro servidor desde un principio.

Documentos por defecto típicos son index.html, index.asp (Si es que estamos programando páginas ASP) o index.php (si es que estamos programando con PHP).

Para definir el documento por defecto debemos apretar con el botón derecho el sitio web que deseamos modificar y seleccionar la opción "Propiedades".

Entonces aparece la ventana de propiedades de este sitio, donde debemos seleccionar la pestaña marcada como "Documentos" para poder definir el documento o los documentos por defecto.

Podemos definir uno o varios documentos por defecto, de modo que, si no existe el primer archivo seleccionado como documento por defecto, se intentaría con el segundo, el tercero y cuantos haya configurados hasta que se encuentre un archivo que mostrar o se acabe la lista. Por tanto, el orden de los distintos archivos configurados como documento por defecto si importa y se puede modificar utilizando las flechas de la izquierda de la lista de posibles documentos.



Si no hay ningún archivo en el directorio cuyo nombre sea alguno de los documentos por defecto, no se mostraría ningún archivo y en su lugar recibiríamos un error 404 o el listado de ese directorio, depende de cómo esté configurado IIS para este caso.

Directorios virtuales en IIS

Un directorio virtual es un directorio del servidor que no está dentro del directorio de publicación habitual, es decir, un directorio que no depende de C:\Inetpub\wwwroot pero que sí que se puede acceder a través del servidor web como si estuviera dentro de dicho directorio.

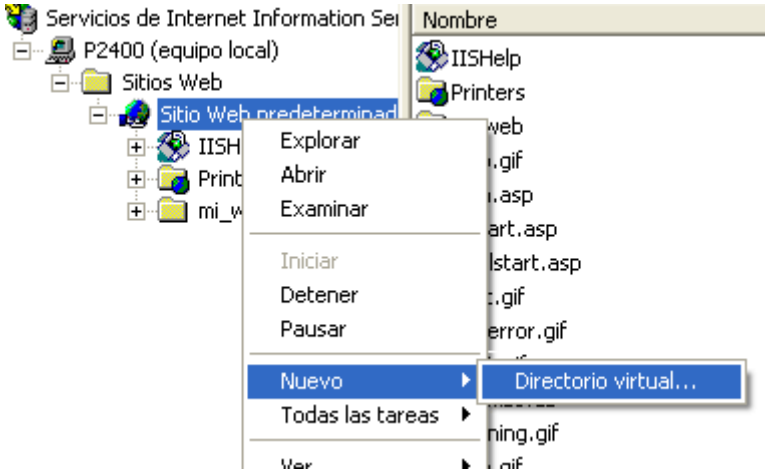
Como ya habíamos comentado, para acceder a nuestro IIS necesitamos escribir una dirección como esta: http://localhost. Así se accede al directorio C:\Inetpub\wwwroot, que es llamado directorio

particular. El directorio virtual se accede con algo como http:///localhost/directorio_virtual, pero no tiene porque existir una correspondencia en disco de este directorio dentro de la carpeta de publicación, es decir, no tiene porque existir el directorio C:\Inetpub\wwwroot\directorio_virtual, sino que dicho directorio podría estar en cualquier otro sitio de nuestro disco duro, por ejemplo C:\mis_paginas.

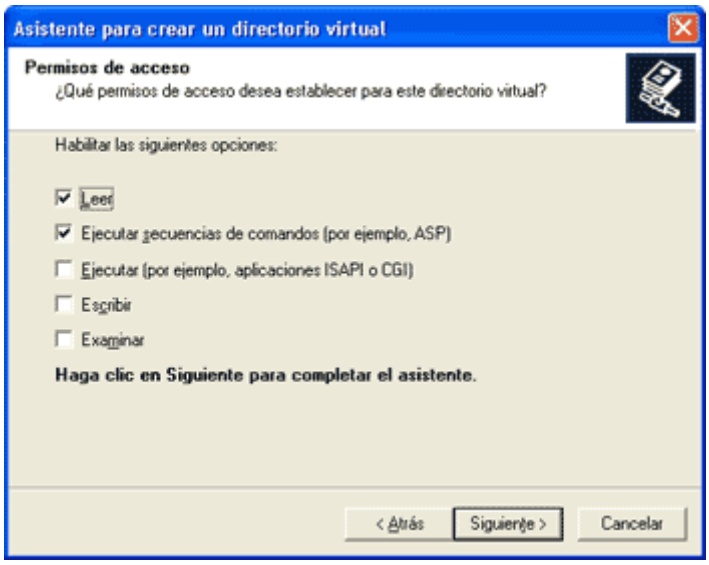
Los directorios virtuales se pueden mapear hacia otro directorio de nuestros discos duros o incluso a otro directorio situado en otro ordenador de la red.

Crear un directorio virtual

Para definir un directorio virtual se puede pulsar con el botón derecho del ratón sobre el sitio web en el que queremos definirlo y seleccionar la opción "Nuevo > Directorio Virtual...". Entonces aparece un asistente que nos guiará paso a paso en el proceso.



El primer paso del asistente nos pregunta el "alias" o nombre lógico que queremos darle al directorio. El segundo paso nos pide la localización física de ese directorio en nuestro disco duro o en la red local. Finalmente nos solicita los permisos que deseamos asignar a ese directorio. El permiso de lectura y el de ejecución de secuencias de comandos (Por ejemplo, ASP) suelen ser suficientes para la mayoría de los casos.



Una vez finalizado el asistente queda creado el directorio virtual y podremos accederlo a través del alias que hayamos seleccionado.