

Unidad 3



Centro Don Bosco
Villamuriel de Cerrato

Creación de páginas web HTML

(incluido HTML 5)

Apuntes realizados para la asignatura de FP Grado Superior:
Lenguajes de Marcas y Sistemas de Gestión de Información
del ciclo Administración de Sistemas Informáticos en Red

Autor: Jorge Sánchez Asenjo (www.jorgesanchez.net)
Versión del documento: 2.0, Año 2012



Esta obra está bajo una licencia de Reconocimiento-NoComercial-CompartirIgual de Creative Commons
Para ver una copia de esta licencia, visite: <http://creativecommons.org/licenses/by-nc-sa/3.0/deed.es>



Atribución-NoComercial-CompartirIgual 3.0 Unported (CC BY-NC-SA 3.0)

Esto es un resumen fácilmente legible del [Texto Legal \(la licencia completa\)](#).

<http://creativecommons.org/licenses/by-nc-sa/3.0/legalcode>

Usted es libre de:

Compartir - copiar, distribuir, ejecutar y comunicar públicamente la obra
hacer obras derivadas

Bajo las condiciones siguientes:



Atribución — Debe reconocer los créditos de la obra de la manera especificada por el autor o el licenciante (pero no de una manera que sugiera que tiene su apoyo o que apoyan el uso que hace de su obra).



No Comercial — No puede utilizar esta obra para fines comerciales.



Compartir bajo la Misma Licencia — Si altera o transforma esta obra, o genera una obra derivada, sólo puede distribuir la obra generada bajo una licencia idéntica a ésta.

Entendiendo que:

Renuncia — Alguna de estas condiciones puede **no aplicarse** si se obtiene el permiso del titular de los derechos de autor

Dominio Público — Cuando la obra o alguno de sus elementos se halle en el **dominio público** según la ley vigente aplicable, esta situación no quedará afectada por la licencia.

Otros derechos — Los derechos siguientes no quedan afectados por la licencia de ninguna manera:

- Los derechos derivados de **usos legítimos** u otras limitaciones reconocidas por ley no se ven afectados por lo anterior.
- Los derechos **morales** del autor;
- Derechos que pueden ostentar otras personas sobre la propia obra o su uso, como por ejemplo **derechos de imagen** o de privacidad.

índice

(3.1) ¿qué es HTML?	9
(3.1.1) introducción	9
(3.1.2) historia de HTML	10
(3.1.3) herramientas HTML	14
(3.2) normalización. versiones de HTML	14
(3.3) protocolo http	16
(3.3.1) publicación de páginas web	17
(3.4) fundamentos básicos de HTML y XHTML	17
(3.4.1) HTML y XHTML	17
(3.4.2) estructura de una página web	18
(3.4.3) espacios en blanco	19
(3.4.4) atributos comunes	20
(3.5) codificación en HTML	21
(3.6) etiquetas básicas de párrafo	29
(3.6.1) párrafo simple	29
(3.6.2) títulos	30
(3.7) líneas y saltos	32
(3.7.1) salto de línea	32
(3.7.2) línea horizontal	33
(3.8) elementos para marcar el texto	33
(3.8.1) resaltado básico de caracteres	33
(3.8.2) marcado avanzado de caracteres	34
(3.8.3) marcado avanzado de párrafos	35
(3.8.4) correcciones	36
(3.9) listas	36
(3.9.1) con viñetas	36
(3.9.2) numéricas	37
(3.9.3) listas anidadas	37
(3.9.4) listas de términos	38
(3.10) enlaces	39
(3.10.1) URLs	39
(3.10.2) crear enlaces	41
(3.10.3) atributos del elemento a	42
(3.10.4) enlaces internos	43
(3.11) imágenes	43
(3.11.1) introducción	43
(3.11.2) inserción de imágenes	44

(3.11.3) imágenes de fondo	44
(3.11.4) mapas de imagen	44
(3.11.5) elemento canvas	46
(3.11.6) etiqueta svg	47
(3.11.7) lenguaje MathML. etiqueta math	48
(3.12) tablas	49
(3.12.1) introducción	49
(3.12.2) tablas simples	49
(3.12.3) celdas de cabecera	51
(3.12.4) títulos en las tablas	52
(3.12.5) etiquetas de agrupación de elementos de una tabla	52
(3.12.6) combinar celdas	53
(3.12.7) tablas dentro de tablas	54
(3.13) elementos span y div	55
(3.13.1) elemento div	55
(3.13.2) elemento span	55
(3.14) formularios	55
(3.14.1) etiqueta form	56
(3.14.2) cuadros de texto, input type="text"	56
(3.14.3) cuadro de contraseñas, input type="password"	57
(3.14.4) botones	57
(3.14.5) botones de radio	58
(3.14.6) casillas de verificación	58
(3.14.7) cuadros combinados	59
(3.14.8) cuadro de selección de archivo	60
(3.14.9) cuadro de texto multilínea	61
(3.14.10) agrupación de controles, fieldset	61
(3.14.11) controles de HTML 5	62
(3.15) etiquetas de cabecera	66
(3.15.1) elemento title	66
(3.15.2) elemento base	66
(3.15.3) elemento link	67
(3.15.4) etiqueta style	68
(3.15.5) etiqueta script	68
(3.15.6) etiqueta meta	69
(3.16) etiquetas semánticas de HTML5	70
(3.16.1) elemento header	70
(3.16.2) elemento footer	71
(3.16.3) elemento section	71
(3.16.4) elemento nav	72

(3.16.5) elemento article	72
(3.16.6) elemento hgroup	72
(3.16.7) elemento figure	72
(3.16.8) elemento figcaption	72
(3.16.9) elemento aside	73
(3.17) elementos multimedia _____	73
(3.17.1) el problema del vídeo y el audio	73
(3.17.2) etiqueta embed	74
(3.17.3) etiqueta object	74
(3.17.4) elemento param	74
(3.17.5) elemento iframe	75
(3.17.6) elemento video	75
(3.17.7) elemento audio	77

(3) creación de páginas web con HTML

(3.1) ¿qué es HTML?

(3.1.1) introducción

Como se comentó en el primer tema: a finales de los 80 se desarrolló el lenguaje de marcas SGML. En esa misma época **Tim Bernes Lee** utilizó SGML para definir un nuevo lenguaje de etiquetas que llamó **Hypertext Markup Language** (lenguaje de marcado de hipertexto) para crear documentos transportables a través de Internet en los que fuera posible el hipertexto; es decir la posibilidad que determinadas palabras marcadas de forma especial permitieran abrir un documento relacionado con ellas.

A pesar de tardar en ser aceptado, HTML fue un éxito rotundo y la causa indudable del éxito de Internet. Hoy en día casi todo en Internet se ve a través de documentos HTML, que popularmente se denominan **páginas web**.

Inicialmente estos documentos se veían con ayuda de intérpretes de texto (como por ejemplo el **Lynx** de **Unix**) que simplemente coloreaban el texto y remarcaban el hipertexto. Después el software se mejoró y aparecieron navegadores con capacidad más gráfica para mostrar formatos más avanzados y visuales.

Lógicamente desde 1989 hasta nuestros días HTML ha mejorado. Entre sus avances fundamentales:

- El lenguaje cada vez ha ido incorporando nuevas etiquetas más potentes, que permiten incluir en los documentos HTML, tablas, capas, marcos, imágenes,...
- Se han añadido lenguajes de **script** (como **JavaScript**) con código incrustado en las páginas HTML que permiten añadir funcionalidades y dinamismo a las páginas web
- Se han añadido técnicas en el lado del servidor con la misma finalidad como aplicaciones **CGI**, **PHP**, **ASP** o **JSP**.
- Se incorporaron lenguajes de estilo (como **CSS**) para generar un formato de documento más avanzado
- Se han añadido utilidades para gestión avanzada de JavaScript con **XML (AJAX)**
- Se ha permitido la inclusión de elementos avanzados en las páginas como Flash o los applets de Java para dar mayor funcionalidad.

- Se permiten elementos semánticos para dar mayor significado al contenido
- Se permite el dibujo libre de elementos en la página mediante el elemento **canvas**.
- Es posible añadir vídeo, audio y otros elementos multimedia de forma fácil

En la actualidad HTML sigue siendo el lenguaje fundamental de las páginas web; pero ahora Internet es la web, es decir todo en Internet se ve a través de una página web. Por eso hoy en día HTML es la capa superficial bajo la que se agolpan tecnologías muy diversas y muy distintas de HTML.

(3.1.2) historia de HTML

Se resaltan a continuación algunos de los eventos más importantes en la historia de HTML. Se indica el año y lo que ocurrió en él

- **1989.**
 - **Tim Bernes Lee**¹, científico británico que trabajaba en el **CERN**² centro de desarrollo nuclear ubicado en Suiza. Intenta trasladar el hipertexto a los documentos científicos, mediante el cual es posible avanzar de un documento a otro mediante enlaces existentes en el propio texto. Teorizó la forma de transportar este tipo de documentos (el actual protocolo **http**) y sobre el lenguaje de marcas a utilizar.
- **1991,**
 - **Tim Bernes Lee** acude a un grupo de discusión en Internet para discutir sobre cómo implementar el hipertexto de forma más conveniente. Con ello no pretende privatizar su invento sino hacerlo público desde el primer momento. Crea ese año un navegador propio (llamado **WorldWideWeb**) para pruebas.
- **1992**
 - La **NCSA**³ se interesa por la ya llamada **web** de **Bernes Lee** y crea el primer navegador: **Mosaic**. Entre sus creadores está **Eric Bina** y **Marc Andreessen** futuros millonarios gracias a la web.
- **1993.**
 - **Lou Montulli** desarrolla **Lynx** para los sistemas Unix, el primer navegador de texto en la web. Será ampliamente utilizado en los años siguientes, aunque luego quedará rápidamente superado por las capacidades de los navegadores gráficos
 - Se lanza comercialmente **Mosaic** por parte de la NCSA. Añade nuevos elementos al HTML original como por ejemplo la inclusión de imágenes.
- **1994**
 - Conferencia global sobre la web.
 - La **IETF**⁴ asigna un grupo de trabajo para estandarizar HTML.

¹ Premio Príncipe de Asturias 2002 junto a **Lawrence Roberts**, **Robert Kahn** y **Vinton Cerf**: considerados los **padres** de Internet

² <http://public.web.cern.ch/public/>

³ **National Center for Supercomputing Applications**, Centro Nacional de Estados Unidos de Supercomputación, creador de las primeras grandes redes de cálculo y supercomputadoras.

- El lenguaje HTML empieza a ser caótico porque aparecen numerosas etiquetas puestas por cada entidad privada. **Dan Connolly** recopila las etiquetas HTML de la época más utilizada y se crea el borrador de **HTML 2**.
- **Marc Andreessen** y **Jim Clark** abandona la NCSA y fundan **Mosaic Communications** (futura **Netscape**). Dejan también los estándares y crean elementos nuevos en el lenguaje HTML para crear páginas más vistosas para su navegador.
- A finales del año se crea la **World Wide Consortium**⁵ (**W3C**) fichando a algunos de los principales impulsores de la web (incluido **Tim Bernes Lee**). Se convertirá en el principal organismo de estandarización de las tecnologías relacionadas con la web en general y de HTML en particular

■ 1995

- Siguen apareciendo nuevos elementos en HTML que impulsan las posibilidades de las páginas web. Se crea el borrador **HTML 3**, que incluye tipos de letra y otras mejoras.
- La empresa **Mosaic Communications** se convierte en **Netscape Communications** y lanza el navegador **Netscape Navigator**. Se convertirá en los siguientes años en el navegador más utilizado.
- **Microsoft** crea **Internet Explorer** y lo incorpora rápidamente como parte del sistema operativo Windows 95. Comienza la primera guerra de los navegadores. Los contendientes son Explorer y Navigator.
- El grupo de trabajo de la IETF para HTML se desmantela por su escasa influencia. El **World Wide Consortium** queda como principal organismo de estandarización de HTML.
- A finales de año aparecen los primeros elementos de creación de hojas de estilo, raíz del lenguaje **CSS** que permite dar formato avanzado a las páginas web y que sigue siendo una de las tecnologías imprescindibles en la actualidad para crear páginas web.
- **Sun Microsystems** crea el lenguaje **Java**, que tendrá una enorme influencia en el desarrollo de Internet.
- Los hermanos **Allaire**, crean **ColdFusion**, un lenguaje basado en HTML que se ejecuta en el servidor que aloja las páginas web (servidores compatibles con esta tecnología) de modo que el cliente no necesita tener un software especial que reconozca esta tecnología. Al cliente le llegan páginas web normales que ha preparado el servidor tras traducir este lenguaje. Fue la primera tecnología de script en el lado del servidor.

■ 1996

- Se crea el **HTML ERB** (**Editorial Review Board**), en el que participan empresas como **IBM**, **Microsoft**, **Netscape**, **Novell**,... y el propio **W3C**. Es una reunión trimestral para ayudar en el estándar.
- Se crea **Yahoo!** la primera página exitosa que permite organizar la web para facilitar la búsqueda de otras páginas. La empresa será una de las de mayor valor en bolsa durante varios años.

⁴ **Internet Engineering Task Force**, grupo que estandariza diferentes aspectos de Internet

⁵ <http://www.w3.org> y en español <http://www.w3c.es/>

- Netscape desarrolla **JavaScript**, un lenguaje basado en C y Java que se incrusta dentro del código HTML de las páginas para darles una mayor potencia. Todavía sigue siendo uno de los lenguajes más influyentes en el desarrollo de páginas y aplicaciones para la web.
En definitiva es una tecnología de script en el lado del cliente. Los navegadores, debido a su éxito, tuvieron que ir poco a poco incorporando plugins (software añadido) que permitieran ejecutar código JavaScript.
- **Rasmus Lerdorf** crea el lenguaje **PHP**. Todavía sigue siendo la tecnología del lado del servidor más popular.
- La empresa **Macromedia** crea el software **Flash**. Se trata de una tecnología del lado del cliente (requiere un plugin en el navegador) que permitió que las páginas incluyeran todo tipo de elementos multimedia e interactivos que hacían de las páginas aplicaciones ricas semejantes a las aplicaciones de un escritorio de ordenador personal.
- **Hakom Wum Lie** crea el navegador **Opera**. Nunca ha alcanzado una gran cuota de público pero sigue presente después de todos esos años.

■ 1997

- Aparece la especificación estándar **HTML 3.2**, la primera en ser ampliamente aceptada. Incluye tablas, applets (pensadas para añadir elementos Java a las páginas) y otros formatos avanzados de formato.
- **Sun Microsystems** crea **Java Servlets** y **Microsoft** crea el lenguaje **ASP**. Son dos de las tecnologías del lado del servidor que tendrán una gran influencia en los años siguientes.

■ 1998

- La W3C lanza como estándares a **HTML 4.0** y a **CSS2**. Los estándares de la W3C cada vez se tienen más en cuenta y ambos alcanzan un gran éxito.
- La combinación **HTML+JavaScript+CSS** se conoce este año como **DHTML** (HTML dinámico). Alcanzará un enorme notoriedad y será la combinación habitual para hacer páginas web atractivas.
- Aparece **XML 1.0** por parte de la W3C, como el lenguaje que debió ser HTML (en palabras del propio **Tim Bernes Lee**). No ha llegado a suplantar a HTML pero sigue teniendo una enorme influencia en todo tipo de tecnologías.

■ 1999

- **Sun** crea **JSP** (páginas de servidor en lenguaje Java) y la plataforma de trabajo **J2EE (Java Enterprise)** con lo que pretende crear un entorno poderoso de trabajo para crear aplicaciones y servicios de Internet en los servidores.
- El navegador **Internet Explorer** de **Microsoft** domina el mercado poniendo fin a la primera guerra de navegadores.
- Aparece **RSS** un formato de contenido basado en XML que permite sindicarse y obtener información de forma veloz.
- La **W3C** presenta HTML 4.01, indicando que será la última versión del HTML clásico.

■ 2000

- Aparece el estándar W3C **XHTML 1.0**, versión de HTML basado en XML que pretende derrocar a HTML. A día de hoy sigue siendo el estándar más respetado para crear páginas web.
- **ISO** (organismo internacional de estándares) publica la norma **ISO 15445** con la que normaliza HTML. Esta norma es prácticamente la misma que la correspondiente al HTML 4.01 de la W3C.

■ 2001

- Estándar XHTML 1.1.
- **PHP** como tecnología en el lado del servidor y Flash en el lado del cliente, son las tecnologías dominantes para crear aplicaciones web enriquecidas (llamadas **RIA**, *Rich Internet Applications*)

■ 2002

- La fundación **Mozilla** recoge el testigo de Netscape y crea el navegador **Firefox**.
- Microsoft crea la plataforma de aplicaciones **.NET** con vocación de competir con **J2EE**.

■ 2003

- Apple lanza al mercado el navegador **Safari**.
- Se crea **Wordpress** el primer gestor de contenidos web (**CMS**). Permite crear páginas web (especialmente blogs) fácilmente y gestionar a diferentes usuarios que podrán editar contenidos de la web fácilmente.

■ 2004

- Se comercializa el navegador **Firefox**, comienza la segunda guerra de navegadores.
- Se forma el **WHATWG** para conseguir un HTML versión 5 que se convierta en nuevo estándar. Lo impulsan **Apple** y **Mozilla** entre otros.

■ 2005

- **AJAX**, tecnología que combina JavaScript, HTML, CSS y XML se populariza pasando a ser una de las tecnologías fundamentales para crear páginas web dinámicas.
- Se crean patrones **MVC (Modelo-Vista-Controlador)** que facilitan a los programadores la creación de servicios web.
- Se lanza **Joomla!** se convertirá en el CMS más popular (aunque actualmente parece que **Wordpress** le está superando).

■ 2007

- Apple comercializa el primer **iPhone**, comienza el éxito de los smartphones que poco a poco pasan a ser uno de los dispositivos que más páginas web visitan.
- Google presenta el sistema **Android** que en poco tiempo estará presente en la mayoría de smartphones.

■ 2008

- La guerra de los navegadores se recrudece con la llega de **Google Chrome**. Actualmente es el navegador más popular.
- Aparece el primer borrador de HTML 5

■ 2011

- Se acepta **HTML 5** y Flash empieza a dejar de utilizarse (aunque sigue siendo muy influyente)
- Aparecen numerosos borradores de CSS3
- La W3C acepta HTML5 y acuerda con la WHATWG el futuro estándar. EN 2013 se espera que aparezca la recomendación oficial de HTML 5 (y de CSS3).

(3.1.3) herramientas HTML

Para escribir HTML bastaría con un editor de texto plano como el **bloc de notas** de Windows o **gedit** de Linux. Pero los resultados se deben mostrar mediante un navegador o aún mejor, probar en varios navegadores para comprobar problemas (ya que hay elementos HTML que no son compatibles con todos los navegadores) .

Sin embargo lo ideal es trabajar mediante editores de código capaces de entender el lenguaje y colorear de diferente manera las etiquetas HTML para distinguirlas del texto normal y así trabajar mejor. Algunos populares son **Notepad++** o **Sublime Text**.

Los entornos de edición en XML como **Oxygene** también permiten trabajar en HTML y de hecho en el caso de utilizar XHTML son una gran opción.

Otra forma es trabajar con editores **WYSIWYG** (*What You See Is What You Get*, lo que se ve es lo que se obtiene), en los que se edita el documento al estilo de los procesadores de texto. De modo que se ve el resultado sin tocar el código y es el editor el que traduce el resultado al código correspondiente. Aunque en realidad no es posible un WYSIWYG real ya que los navegadores muestran el resultado de diferente forma.

(3.2) normalización. versiones de HTML

Los programas capaces de traducir el código HTML y producir una salida en pantalla de los mismos son los **navegadores** (*browsers* en inglés). Se trata de un software gráfico que se inició con la creación de **Mosaic** a principios de los 90 y que poco a poco produjo más productos hasta llegar a una guerra de navegadores a finales de los 90 entre **Internet Explorer** de **Microsoft** y **Navigator** de **Netscape** que ganó Microsoft pero que ahora continúa con otros navegadores como el propio **Internet Explorer**, **Mozilla Firefox** (sucesor de código abierto de Netscape), **Google Chrome**, **Opera** o **Apple Safari** entre otros.

El problema surgió en cuanto unos navegadores incorporaron elementos HTML que el resto no traducía, con lo que aparecieron diferentes dialectos HTML. Así una página se podía mostrar de forma totalmente diferente según el navegador.

La solución pasó por intentar estandarizar el lenguaje. Por ello el propio Tim Bernes Lee fundó la **World Wide Web Consortium** (abreviado **W3C**) como organismo de estandarización del lenguaje HTML ante la industria.

En la actualidad las directrices de W3C son seguidas por la mayoría de navegadores aunque no al 100%, lo que sigue generando problemas a los creadores de páginas web.

La situación, sin embargo se ha complicado en estos últimos años con la aparición de diferentes estándares, en concreto actualmente se consideran estándares a HTML 4.01, XHTML 1.0 y 1.1 y ya se considera de la misma forma a HTML 5. Las diferencias entre ellos son:

- **HTML 4.01.** Se trata de la versión estándar del HTML tradicional hecha en el año 1999 y que sigue teniendo mucha vigencia actualmente. Hay tres versiones: la transicional (que permite seguir usando algunas etiquetas que se consideran obsoletas), la estricta (que elimina numerosas etiquetas y atributos para forzar a crear un HTML con menos formato y más significado) y la *frameset* orientada a usar los ya muy poco utilizados marcos. La versión más popular es la transicional al ser más libre.

Para avisar de que nuestro documento sigue las normas de HTML 4.01, se coloca una etiqueta DOCTYPE que permite llegar al DTD estándar de esta versión. Para HTML 4.01 transicional es:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

HTML 4 no es compatible con XML y por ello a los antiguos diseñadores web les gusta más; pero precisamente el no ser compatible con XML no permite utilizar muchas aplicaciones creadas para ese lenguaje

- **XHTML.** XHTML era planteado como el sustituto de HTML, la primera versión, la 1.0 sigue siendo la más usada y en la fecha de escritura de este texto es la versión HTML más utilizada. Hay también versión estricta, transicional y (muy poco utilizada) *frameset*; la más utilizada, otra vez es la transicional porque permite el uso de numerosos atributos y elementos que se consideran obsoletos, pero que muchos diseñadores utilizan.

Las páginas XHTML obligan a que la escritura de HTML siga las reglas del lenguaje XML bien formado.

Para indicar que utilizamos XHTML 1.0 transicional se debe poner este DOCTYPE:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Además la etiqueta raíz **html** debe usarse de este modo (especificando el espacio de nombres y el lenguaje de la página):

```
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es">
```

En el caso de XHTML 1.1 el DOCTYPE sería:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Se puede observar que es más fiel al lenguaje XML. La etiqueta raíz html sería igual que la anterior

- **HTML 5.** Se considera el estándar de este año 2012. Poco a poco se está convirtiendo en el HTML más utilizado. Se creó fuera del **World Wide Web Consortium** en la entidad llamada WHATWG (**Web Hypertext Application Technology Working Group**), auspiciado por **Apple** y **Mozilla** entre otras empresas a las que se le han ido añadiendo muchas más.

La W3C decidió unirse a la WHATWG en 2008 para buscar un estándar. Actualmente el proyecto está en fase de borrador planificándose su paso a recomendación oficial en 2014. La WHATWG va por su lado mediante una filosofía de "estándar vivo" en la que ya hay una recomendación de uso por su parte, pero que va cambiando continuamente según se consideran nuevas mejoras.

Aunque no es una recomendación oficial y por ahora los navegadores aceptan sólo partes de HTML 5 (algunos casi todas, pero otros muy pocas) el hecho es que ya se anima a usarlo y la mayoría de páginas nuevas son HTML5. Para ello marcan un DOCTYPE mucho más sencillo:

```
<!DOCTYPE html>  
<html lang="es">
```

La etiqueta raíz HTML también es más sencilla al no indicar un espacio de nombres.

- **XHTML 5.** Se trata de una especificación en proceso. Es simplemente HTML 5 pero obligado a usar las reglas de XML bien formado. Es decir es HTML 5 siguiendo lo estricto de XML.

(3.3) protocolo http

La transmisión de páginas web (que en definitiva son documentos HTML) se realiza mediante el protocolo **http**, que es parte de la pila de protocolos **TCP/IP**. Se trata de un protocolo basado en una comunicación **petición-respuesta**; de modo que un cliente (también llamado **user agent**, agente de usuario) realiza una petición de recurso indicando su dirección, y un servidor responde a dicha petición bien transmitiendo al cliente el recurso solicitado o bien indicando un mensaje de error.

La dirección del recurso se indica utilizando la notación URL, que funciona así:

```
protocolo://servidor:puerto/rutaAlRecurso
```

En el caso de las páginas web, el protocolo es **http**. En la parte **servidor**, se indica la dirección del servidor (por ejemplo www.jorgesanchez.net) y la ruta es la ruta que hay que seguir por las carpetas y archivos del servidor para llegar al recurso. Ejemplo:

```
http://www.jorgesanchez.net/bd/sgbd.html
```

Permite al navegador mostrar en pantalla la página web **sgbd.html** alojada en la carpeta **bd** del servidor www.jorgesanchez.net con el que se comunica utilizando el protocolo **http** (puesto que no se ha indicado puerto, se utilizará el puerto 80).

A veces no se indica la ruta y en ese caso el servidor envía la llamada página por defecto, **página de inicio** o **home page** que generalmente es un documento HTML

llamado *index.html* o *default.html* o *home.html* y que se ubica en la carpeta raíz del servidor. También es posible indica una home page por cada carpeta del servidor.

La comunicación entre el servidor y el cliente, normalmente se realiza a través del puerto **80**. Pero se podría indicar otro puerto, ejemplo (usando el puerto **9123**):

<http://www.jorgesanchez.net:9123/bd/sgbd.html>

(3.3.1) publicación de páginas web

Se denomina **sitio web** al conjunto de páginas web y recursos de las mismas que contienen toda la información asociada a una determinada dirección de inicio en Internet.

Cuando una persona desea crear un nuevo sitio web, inicialmente le crea en su ordenador de trabajo y para ello debe crear una carpeta y en ella almacenar todas las páginas y recursos necesarios (imágenes, sonidos, vídeos, archivos auxiliares,...). Esa carpeta se deberá enviar al servidor web que hayamos contratado o del que dispongamos para publicar nuestra página en Internet.

Para ello normalmente se utiliza el protocolo de transmisión de ficheros conocido como **FTP**. Con copiar la carpeta en el sitio adecuado de nuestro servidor, la página estará publicada. Normalmente para ello se nos pide un usuario y contraseña que verifica que realmente somos los propietarios del espacio.

Hoy en día los herramientas avanzadas de diseño de páginas web tienen capacidad para transmitir los ficheros al servidor.

(3.4) fundamentos básicos de HTML y XHTML

(3.4.1) HTML y XHTML

La diferencia fundamental entre HTML y XHTML está en el cumplimiento que exigen de XML. Así XHTML es, en definitiva, un dialecto de XML que, por lo tanto, cumple sus reglas de forma absoluta. De tal manera que cumple las reglas del XML bien formado:

- Los elementos se escriben en minúsculas obligatoriamente
- Todo elemento se debe cerrar y además se cierra primero el último que se abrió. Los elementos vacíos también hay que cerrarles (por ejemplo se escribe `
` para el salto de línea)
- Todos los valores de los atributos van entrecomillados
- Todo atributo debe de tener un valor
- Hay un único elemento raíz (**html**).
- Los símbolos `<`, `>`, `&` y comillas deben utilizar entidades y no escribirse tal cual

Sin embargo HTML no es tan estricto y permite que:

- Los valores de los atributos no tienen que ir entrecomillados (a no ser que tengan espacios en blanco)
- No todos los atributos tienen valores. Es decir no siempre se usa *atributo=valor* hay atributos que no tienen valor
- Las etiquetas HTML se pueden escribir como queramos, en mayúsculas o minúsculas

- Las etiquetas vacías no es obligatorio cerrarlas (se puede escribir `
`)

Sin embargo aunque el reciente HTML 5 vuelve a ser más laxo con las reglas al estilo del HTML clásico, lo cierto es que se aconseja seguir las reglas del XHTML aun cuando se escriba HTML de esa forma podremos validar sin problemas nuestras páginas con herramientas XML.

(3.4.2) estructura de una página web

La estructura básica de una página web cambia según si hablamos de XHTML o de HTML. Como son tantas las posibilidades, en el presente documento hablaremos sólo de las dos versiones más populares en el momento de escribir estos apuntes que son XHTML 1.0 transicional y HTML5. Como ambas representan muy bien tanto a XHTML como a HTML respectivamente, sería fácil descubrir como es el esqueleto de una página web en cualquier otra versión.

XHTML (1.0 transicional)

La estructura de una página XHTML 1.0 transicional que esté escrita en español es:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="es" lang="es">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Título</title>
  </head>
  <body>
    Contenido de la página web
  </body>
</html>
```

Las partes son:

- La etiqueta **DOCTYPE** que permite especificar el documento DTD de validación del XHTML de la página. En caso de elegir otra versión de XHTML simplemente hay que cambiar las rutas al documento validador
- Elemento **html** que marca la raíz de la página y además debe especificar el espacio de nombres al que pertenecen las etiquetas (atributo **xmlns**) y el lenguaje en el que está escrita (atributo **xml:lang**).

Es conveniente (aunque el estándar no lo considera obligatorio) utilizar el atributo **lang**, que es común a todos los elementos HTML y que sirve también para marcar el lenguaje en el que está escrita la página. Usando tanto **xml:lang** como **lang** aseguramos que todos los navegadores queden avisados sobre el lenguaje utilizado.

El lenguaje español se puede indicar con **es** simplemente o con más detalle usando **es-ES** que es el símbolo internacional de español de España. Esta terminología sigue los códigos de dos letras definidos en la norma **ISO 639-1** (que contiene dos letras para cada lenguaje) y el código de dos letras de país de la **ISO 3166-1**. Otros ejemplos son:

- **ca** para catalán

- **eu** para euskera
 - **gl** para gallego
 - **fr** para francés
 - **en** para inglés
 - **en-US** para inglés de Estados Unidos
- Elemento **head** que marca el inicio de la cabecera. En la cabecera se coloca el título de la página (elemento **title**) y los elementos que sirven para incluir estilos CSS, código JavaScript y todo tipo de elementos que se utilizarán dentro de la página.
 - La etiqueta **meta** que advierte sobre el sistema de codificación del texto de la misma. En el código anterior se utiliza el más recomendable, que es UTF-8, Unicode de 8 bits
 - Elemento **body** que contiene el cuerpo de la página, el contenido visible por el navegador.

HTML5

En el caso de HTML5 la estructura es muy parecida pero simplifica muchas etiquetas:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8" >
    <title></title>
  </head>
  <body>

  </body>
</html>
```

Como diferencias más notables respecto a HTML:

- La etiqueta **DOCTYPE** está simplificada
- La etiqueta **meta** es mucho más sencilla para indicar el código del archivo (Unicode)
- El lenguaje se indica sólo con el atributo **lang** del elemento **html**.
- El resto de elementos no varía

(3.4.3) espacios en blanco

El texto dentro de las páginas web no mantiene los espacios en blanco, solo se considera el primero no se tienen en cuenta los siguientes. Solo la etiqueta **pre** permite preservarles (pero no es muy aconsejable su uso, ya que esa etiqueta no nos dice qué tipo de texto tenemos).

Ejemplo:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <title></title>
```

```

</head>
<body>
  Este texto
           tiene           muchos
           espacios
</body>
</html>
    
```

El resultado mostraría en la página este texto:

Este texto tiene muchos espacios

Para mantener esas distancias, el texto tendría que estar dentro de un elemento **pre**, pero no se aconseja el uso de **pre** porque ayuda a adquirir malos hábitos y dificulta el aprendizaje de las formas avanzadas de maquetar páginas web.

La entidad ** ** permite añadir un espacio en blanco obligatorio. Por lo que si, por ejemplo, la escribimos cuatro veces seguidas, estaremos dejando cuatro espacios en blanco que el navegador sí tendrá en cuenta.

(3.4.4) atributos comunes

Todos los elementos XHTML de escritura tienen una serie de atributos comunes que se pueden utilizar de la misma forma en cualquier elemento. Son los siguientes:

atributo	significado
id	Identifica al elemento. Se trata de un identificador en el sentido XML (debe empezar por letra, no tener espacios, sí puede tener números y no puede repetirse su valor en dos elementos de la misma página web). Es uno de los atributos más importantes.
title	Permite poner un título al elemento. Es una especie de descripción corta que es muy útil en aquellos elementos que no muestran información que requiere una explicación (como un abreviatura, una sigla, una imagen,...). El navegador reacciona a este atributo mostrando un cartelito cuando el cursor del ratón se aproxima a un elemento que use este atributo (ver imagen inferior)
lang	Indica el lenguaje en el que está escrito el contenido del elemento (para español: lang="es"). Sólo se usa si algún elemento usa un lenguaje distinto al indicado para toda la página.
xml:lang	Procede de XML significa lo mismo que lang , pero es entendible por el software analizador de XML. Sólo se usa en XHTML
dir	Indica la dirección de lectura del texto contenido en el elemento. Puede ser uno de estos dos valores: ltr (<i>left to right</i> , de izquierda a derecha) o rtl (<i>right to left</i> , de derecha a izquierda).
class style	Permiten especificar formato de tipo CSS al atributo

atributo	significado
onclick	Permiten asignar código Javascript a alguna de esas acciones (un clic de ratón, un doble clic, pulsar tecla, dejar de pulsar tecla, mantener pulsada la tecla, entrar con el ratón en el elemento, mover el ratón por encima, sacar el ratón del elemento, pulsar tecla de ratón, liberar tecla de ratón,...)
ondblclick	
onkeyup	
onkeydown	
onkeypress	
onmouseover	
onmouseout	
onmousedown	
onmouseup	
onmousemove	

(3.5) codificación en HTML

Como Unicode no está realmente adoptado por todos los sistemas, es un peligro que el texto del cuerpo del documento XHTML utilice símbolos como la *eñe* por ejemplo que están fuera del código ASCII original, a riesgo de que en lugar de una eñe nos encontremos con otro símbolo cuando visualicemos nuestra página. ¿Por qué ocurre esto?

Los navegadores compatibles con HTML4 deben de ser capaces de poder decodificar texto en formato **ISO-8859-1** además de en **UTF-8** y **UTF-16**. Por lo que en principio parece que no hay ningún problema para escribir el código que sea. De hecho es cierto que hoy en día los principales navegadores respetan tanto a Unicode como al resto de codificaciones nacionales.

El problema es que nuestro documento HTML puede haber sido codificado usando una tabla distinta a la nuestra en el servidor en el que alojemos la página. Por ejemplo Windows en el bloc de notas por defecto cifra usando la tabla WIN1252 que no es estándar. Y cuando publiquemos por ejemplo en Linux, nos encontraremos con que decodifica con otra tabla. Resultado: los símbolos fuera del ASCII no aparecen como debieran.

Eso provoca que los únicos caracteres que sabemos con seguridad que se codifican igual entre distintos sistemas sean los correspondientes al ASCII original. Por ello desde hace muchos años hay diseñadores de páginas que sólo utilizan caracteres dentro del ASCII original.

Además hay símbolos que no pueden ser escritos como texto normal ya que se malinterpretarían por los navegadores como los símbolos **>** o **<** por ejemplo.

El problema es que el ASCII no permite símbolos de otros idiomas como la eñe y por ello se ideó poder escribir códigos nacionales usando una notación especial. Son las entidades.

De este modo para especificar códigos fuera del ASCII, se coloca el símbolo **&**, seguido del **nombre del código** y finalizado con el **punto y coma**. Por ejemplo **˜** es la entidad que representa a la eñe. Es lo mismo escribir ñ que escribir ˜

También es posible codificar utilizando la notación:

&#n;

n es el número del código dentro de la tabla Unicode (hay que recordar que los primeros 256 caracteres son los correspondientes a la tabla ISO-8859_1). Los símbolos que obligatoriamente se deben codificar de esta forma son:

Carácter	Número	Nombre
"	&#34;	&quot; ;
'	&#39;	&apos; ; (el navegador Explorer no le reconoce, por lo que no se aconseja su uso)
&	&#38;	&amp; ;
<	&#60;	&lt; ;
>	&#62;	&gt; ;

En definitiva las entidades de XML. Los caracteres fuera del código ASCII y dentro de la codificación 8859_1 son:

Carácter	Número	Nombre
À	&#192;	&Agrave;
Á	&#193;	&Aacute; ;
Â	&#194;	&Acirc; ;
Ã	&#195;	&Atilde; ;
Ä	&#196;	&Auml; ;
Å	&#197;	&Aring; ;
Æ	&#198;	&AElig; ;
Ç	&#199;	&Ccedil; ;
È	&#200;	&Egrave; ;
É	&#201;	&Eacute; ;
Ê	&#202;	&Ecirc; ;
Ë	&#203;	&Euml; ;
Ì	&#204;	&Igrave; ;
Í	&#205;	&Iacute; ;
Î	&#206;	&Icirc; ;
Ï	&#207;	&Iuml; ;
Ð	&#208;	&ETH; ;
Ñ	&#209;	&Ntilde; ;
Ò	&#210;	&Ograve; ;
Ó	&#211;	&Oacute; ;
Ô	&#212;	&Ocirc; ;
Õ	&#213;	&Otilde; ;
Ö	&#214;	&Ouml; ;
Ø	&#216;	&Oslash; ;
Ù	&#217;	&Ugrave; ;
Ú	&#218;	&Uacute; ;
Û	&#219;	&Ucirc; ;

Carácter	Número	Nombre
Ü	Ü	Ü
Ý	Ý	Ý
Þ	Þ	Þ
ß	ß	ß
à	à	à
á	á	á
â	â	â
ã	ã	ã
ä	ä	ä
å	å	å
æ	æ	æ
ç	ç	ç
è	è	è
é	é	é
ê	ê	ê
ë	ë	ë
ì	ì	ì
í	í	í
î	î	î
ï	ï	ï
ð	ð	ð
ñ	ñ	ñ
ò	ò	ò
ó	ó	ó
ô	ô	ô
õ	õ	õ
ö	ö	ö
ø	ø	ø
ù	ù	ù
ú	ú	ú
û	û	û
ü	ü	ü
ý	ý	ý
þ	þ	þ
ÿ	ÿ	ÿ

Otros símbolos de la tabla 8859_1 son:

Carácter	Número	Nombre
	&#160;	&nbsp; ; (espacio obligatorio, los navegadores respetan estos espacios en blanco)
¡	&#161;	&iexcl;
¢	&#162;	&cent;
£	&#163;	&pound;
¤	&#164;	&curren;
¥	&#165;	&yen;
	&#166;	&brvbar;
§	&#167;	&sect;
¨	&#168;	&uml;
©	&#169;	&copy;
ª	&#170;	&ordf;
«	&#171;	&laquo;
¬	&#172;	&not;
	&#173;	&shy;
®	&#174;	&reg;
—	&#175;	&macr;
°	&#176;	&deg;
±	&#177;	&plusmn;
²	&#178;	&sup2;
³	&#179;	&sup3;
´	&#180;	&acute;
µ	&#181;	&micro;
¶	&#182;	&para;
•	&#183;	&middot;
¸	&#184;	&cedil;
¹	&#185;	&sup1;
º	&#186;	&ordm;
»	&#187;	&raquo;
¼	&#188;	&frac14;
½	&#189;	&frac12;
¾	&#190;	&frac34;
¿	&#191;	&iquest;
×	&#215;	&times;
÷	&#247;	&divide;

Hay que tener en cuenta que en la tabla 8859_1 no aparece el símbolo del euro (sí en la 8859_1). Por lo que se permite usar el código `€` para el símbolo €.

Además es posible utilizar otros símbolos que son parte de Unicode, por ejemplo los matemáticos:

Carácter	Número	Nombre
∀	<code>&#8704;</code>	<code>&forall;</code>
∂	<code>&#8706;</code>	<code>&part;</code>
∃	<code>&#8707;</code>	<code>&exist;</code>
∅	<code>&#8709;</code>	<code>&empty;</code>
∇	<code>&#8711;</code>	<code>&nabla;</code>
∈	<code>&#8712;</code>	<code>&isin;</code>
∉	<code>&#8713;</code>	<code>&notin;</code>
∋	<code>&#8715;</code>	<code>&ni;</code>
∏	<code>&#8719;</code>	<code>&prod;</code>
∑	<code>&#8721;</code>	<code>&sum;</code>
-	<code>&#8722;</code>	<code>&minus;</code>
*	<code>&#8727;</code>	<code>&lowast;</code>
√	<code>&#8730;</code>	<code>&radic;</code>
∝	<code>&#8733;</code>	<code>&prop;</code>
∞	<code>&#8734;</code>	<code>&infin;</code>
∠	<code>&#8736;</code>	<code>&ang;</code>
∧	<code>&#8743;</code>	<code>&and;</code>
∨	<code>&#8744;</code>	<code>&or;</code>
∩	<code>&#8745;</code>	<code>&cap;</code>
∪	<code>&#8746;</code>	<code>&cup;</code>
∫	<code>&#8747;</code>	<code>&int;</code>
∴	<code>&#8756;</code>	<code>&there4;</code>
~	<code>&#8764;</code>	<code>&sim;</code>
≅	<code>&#8773;</code>	<code>&cong;</code>
≈	<code>&#8776;</code>	<code>&asymp;</code>
≠	<code>&#8800;</code>	<code>&ne;</code>
≡	<code>&#8801;</code>	<code>&equiv;</code>
≤	<code>&#8804;</code>	<code>&le;</code>
≥	<code>&#8805;</code>	<code>&ge;</code>
⊂	<code>&#8834;</code>	<code>&sub;</code>
⊃	<code>&#8835;</code>	<code>&sup;</code>
⊄	<code>&#8836;</code>	<code>&nsub;</code>
⊆	<code>&#8838;</code>	<code>&sube;</code>
⊇	<code>&#8839;</code>	<code>&supe;</code>
⊕	<code>&#8853;</code>	<code>&oplus;</code>

Carácter	Número	Nombre
⊗	&#8855;	&otimes;
⊥	&#8869;	&perp;
.	&#8901;	&sdot;

Símbolos griegos:

Carácter	Número	Nombre
A	&#913;	&Alpha;
B	&#914;	&Beta;
Γ	&#915;	&Gamma;
Δ	&#916;	&Delta;
E	&#917;	&Epsilon;
Z	&#918;	&Zeta;
H	&#919;	&Eta;
Θ	&#920;	&Theta;
I	&#921;	&Iota;
K	&#922;	&Kappa;
Λ	&#923;	&Lambda;
M	&#924;	&Mu;
N	&#925;	&Nu;
Ξ	&#926;	&Xi;
O	&#927;	&Omicron;
Π	&#928;	&Pi;
P	&#929;	&Rho;
Σ	&#931;	&Sigma;
T	&#932;	&Tau;
Υ	&#933;	&Upsilon;
Φ	&#934;	&Phi;
X	&#935;	&Chi;
Ψ	&#936;	&Psi;
Ω	&#937;	&Omega;
α	&#945;	&alpha;
β	&#946;	&beta;
γ	&#947;	&gamma;
δ	&#948;	&delta;
ε	&#949;	&epsilon;
ζ	&#950;	&zeta;
η	&#951;	&eta;
θ	&#952;	&theta;
ι	&#953;	&iota;

Carácter	Número	Nombre
κ	κ	κ
λ	λ	λ
μ	μ	μ
ν	ν	ν
ξ	ξ	ξ
ο	ο	ο
π	π	π
ρ	ρ	ρ
ς	ς	ς
σ	σ	σ
τ	τ	τ
υ	υ	υ
φ	φ	φ
χ	χ	χ
ψ	ψ	ψ
ω	ω	ω
θ	ϑ	ϑ
Υ	ϒ	ϒ
Ϝ	ϖ	ϖ

Y otros símbolos representables por HTML:

Carácter	Número	Nombre
Œ	Œ	Œ
œ	œ	œ
Š	Š	Š
š	š	š
ÿ	Ÿ	Ÿ
f	ƒ	ƒ
^	ˆ	ˆ
~	˜	˜
	  (deja un espacio del tamaño de la letra ene)	
	  (deja un espacio del tamaño de la letra eme)	 
	  (espacio corto)	 

Carácter	Número	Nombre
	&#8204; (espacio para el número cero sin coma)	&zwj;
	&#8205; (espacio para el número cero con coma)	&zwj;
	&#8206;	&lrm;
	&#8207;	&rlm;
-	&#8211;	&ndash;
—	&#8212;	&mdash;
‘	&#8216;	&lsquo;
’	&#8217;	&rsquo;
,	&#8218;	&sbquo;
“	&#8220;	&ldquo;
”	&#8221;	&rdquo;
„	&#8222;	&bdquo;
†	&#8224;	&dagger;
‡	&#8225;	&Dagger;
•	&#8226;	&bull;
...	&#8230;	&hellip;
%o	&#8240;	&permil;
’	&#8242;	&prime;
”	&#8243;	&Prime;
<	&#8249;	&lsaquo;
>	&#8250;	&rsaquo;
—	&#8254;	&oline;
€	&#8364;	&euro;
™	&#8482;	&trade;
←	&#8592;	&larr;
↑	&#8593;	&uarr;
→	&#8594;	&rarr;
↓	&#8595;	&darr;
↔	&#8596;	&harr;
↵	&#8629;	&crarr;
⌈	&#8968;	&lceil;
⌋	&#8969;	&rceil;
⌊	&#8970;	&lfloor;
⌋	&#8971;	&rfloor;
♠	&#9674;	&loz;
♠	&#9824;	&spades;

Carácter	Número	Nombre
♣	♣	♣
♥	♥	♥
♦	♦	&diamonds;

En cualquier caso salvo para poder escribir símbolos de fuera del teclado, actualmente con la difusión ya imparable de Unicode, lo normal es codificar las páginas Unicode y escribir de forma normal los símbolos nacionales.

(3.6) etiquetas básicas de párrafo

A diferencia del HTML clásico, en XHTML todo texto debe de estar encerrado en un elemento de párrafo. De esa forma se indica el tipo de texto que es. Los elementos de párrafo más importantes son:

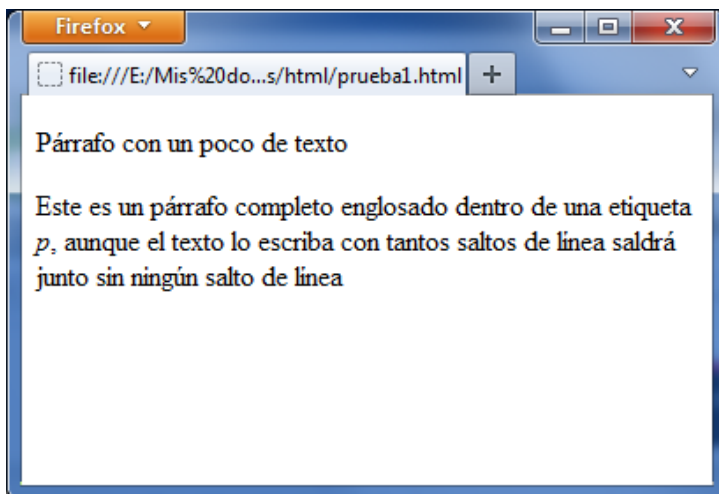
(3.6.1) párrafo simple

Lo marca la etiqueta `p`. El texto engrosado en una etiqueta `p`, aparecerá siempre junto sin espacios entre líneas. Por defecto cada párrafo de tipo `p` se separa con una línea respecto a los otros párrafos. Normalmente los navegadores utilizan el tipo de letra **Times** de tamaño 11pt para la letra de párrafo normal.

Es una de las etiquetas más utilizadas:

```
<p>
  Párrafo con un poco de texto
</p>
<p>
  Este es un párrafo completo engrosado dentro de una etiqueta <em>p</em>,
  aunque
  el texto lo escriba
  con tantos saltos de línea
  saldrá junto sin ningún
  salto
  de
  línea
</p>
```

En el navegador saldría:



En la imagen se observa el modo en el que el navegador muestra el párrafo

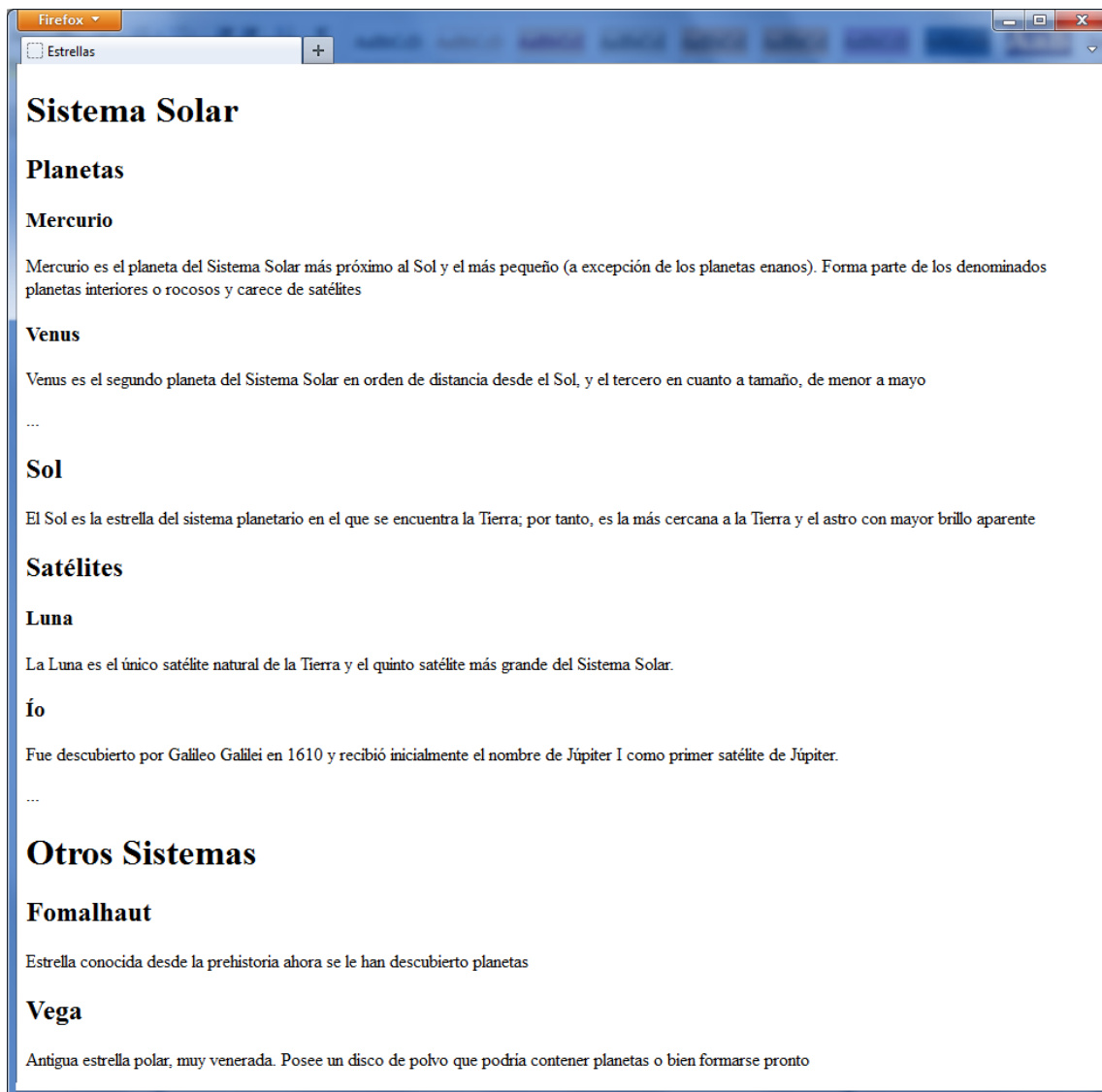
(3.6.2) títulos

Hay una serie de siete etiquetas que comienzan con la letra *h* y le sigue un número del 1 al 7, que sirven para marcar párrafos que se considerarán títulos del texto. De modo que el número 1 marcará títulos de primer nivel, es decir los títulos principales irán marcados con *h1*. Después se podría usar *h2* para designar párrafos que se considerarán títulos de segundo nivel. Ejemplo (XHTML completo):

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="es" xml:lang="es">
  <head>
    <title>Estrellas</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  </head>
  <body>
    <h1>Sistema Solar</h1>
    <h2> Planetas</h2>
    <h3>Mercurio</h3>
    <p>
      Mercurio es el planeta del Sistema Solar más próximo al Sol y el más
      pequeño (a excepción de los planetas enanos). Forma parte de los
      denominados planetas interiores o rocosos y carece de satélites
    </p>
    <h3>Venus</h3>
    <p>
      Venus es el segundo planeta del Sistema Solar en orden de distancia
      desde el Sol, y el tercero en cuanto a tamaño, de menor a mayo
    </p>
    <p>...</p>
    <h2>Sol</h2>
```

```
<p>
  El Sol es la estrella del sistema planetario en el que se encuentra la Tierra;
  por tanto, es la más cercana a la Tierra y el astro con mayor brillo aparente
</p>
<h2>Satélites</h2>
<h3>Luna</h3>
<p>
  La Luna es el único satélite natural de la Tierra y el quinto
  satélite más grande del Sistema Solar.
</p>
<h3>Ío</h3>
<p>
  Fue descubierto por Galileo Galilei en 1610 y recibió inicialmente
  el nombre de Júpiter I como primer satélite de Júpiter.
</p>
<p>...</p>
<h1>Otros Sistemas</h1>
<h2>Fomalhaut</h2>
<p>Estrella conocida desde la prehistoria ahora se le han descubierto
planetas</p>
<h2>Vega</h2>
<p>Antigua estrella polar, muy venerada. Posee un disco de polvo que podría
contener
  planetas o bien formarse pronto</p>
</body>
</html>
```

En el navegador aparecería:



(3.7) líneas y saltos

(3.7.1) salto de línea

A veces es necesario dentro del texto de un determinado párrafo hacer un salto de línea. El elemento que lo realiza no tiene cierre y es **br**. Ejemplo:

```
<p>Primera línea <br/>Segunda línea</p>
```

Aparecerá cada texto en una línea

(3.7.2) línea horizontal

Otra posibilidad es hacer un salto pero dejando una línea horizontal en el hueco de las palabras. Esto lo hace la etiqueta `hr` (que tampoco tiene cierre):

```
<p>Primera línea <hr/>Segunda línea</p>
```

Aunque los navegadores entienden este código. En realidad `hr` tiene que estar fuera de las etiquetas de párrafo, es decir lo correcto es:

```
<p>Primera línea </p>  
<hr/>  
<p>Segunda línea</p>
```

Resultado



(3.8) elementos para marcar el texto

(3.8.1) resaltado básico de caracteres

Son elementos que permiten marcar el texto de forma especial.

negrita

Lo hace el elemento `strong` y también (aunque se considera en desaparición) el elemento `b` (de **bold**, negrita).

cursiva

El elemento `em` y también `i` (de *italic*, aunque también en desaparición).

subíndice

Permite que el texto aparezca por debajo de la línea base y en un tamaño más pequeño. Lo hace el elemento `sub`, ejemplo:

```
<p>La fórmula del agua es H<sub>2</sub>O</p>
```

Obtendría: H₂O

superíndice

Parecida al anterior, pero ahora el texto marcado con el elemento `sup` aparecerá por encima y en pequeño. Ejemplo:

```
<p>La fórmula del agua es H<sup>2</sup>O</p>
```

Obtendría: H²O

(3.8.2) marcado avanzado de caracteres

En realidad son elementos poco utilizados para marcar texto, pero son muy interesantes para dar significado al mismo. La esperanza es que en el futuro haya cada vez más herramientas software capaces de manipular de forma adecuada estos elementos y así poder dar grandes posibilidades al texto escrito de las páginas web.

La lista de elementos relacionados con este marcado es:

elemento	significado	atributos que se suelen utilizar con el elemento	
abbr	Indica una abreviatura (por ejemplo O.N.U.). <code><abbr title="Su Alteza Real">SAR</abbr></code>	title	Permite indicar el significado de la abreviatura
acronym	Indica un acrónimo (por ejemplo tlfno) <code><acronym title="Teléfono">tlfno</acronym></code> En algunos navegadores una línea punteada bajo las abreviaturas y acrónimos permite indicar al usuario que al arrimar el ratón se explican los mismos	title	Permite indicar el significado del acrónimo
dfn	Permite indicar la definición de un término: <code><dfn title="usar dos conceptos de significado opuesto en una sola expresión">oxímoron</dfn></code> Los navegadores suelen mostrar este texto en negrita	title	Permite indicar la explicación de la definición
cite	Marca un texto como cita literal. <code><cite>Alea Jacta Est</cite></code> Los navegadores lo suelen mostrar en cursiva		
cod	Permite indicar que el texto al que engloba pertenece a código fuente de un lenguaje de programación.		
samp	Indica un ejemplo de salida por pantalla de un programa informático		
kbd	Se usa para indicar que el texto incluido indica una tecla del teclado o una combinación de teclas.	title	Para explicar la tecla
var	Indica nombres de variables		
time	Disponible desde HTML 5 permite indicar una hora (por ejemplo 10:00)		
bdo	Permite indicar la dirección del texto. Ejemplo: <code><bdo dir="rtl">Este texto sale al revés</bdo></code> Saldría <i>séver la elas otxet etsE</i>	dir	Posee dos valores ltr (el texto se muestra de izquierda a derecha) y rtl (el texto se muestra de derecha a izquierda)

(3.8.3) marcado avanzado de párrafos

Se trata de elementos de párrafo. Permiten utilizarse en lugar de los elementos habituales de párrafo (como **p**, **h1**,...) para marcar su texto con un significado especial. Los navegadores además les formatean de forma que resalte dicho significado.

elemento	significado	atributos que se suelen utilizar con el elemento	
blockquote	Permite indicar una cita de texto extensa. Ejemplo: <pre><blockquote cite="http://noticias.juridicas.com/base_datos/Admin/constitucion.tp.html#a1"> <p> Artículo 1 de la constitucion
 España se constituye en un Estado social y democrático de Derecho, que propugna como valores superiores de su ordenamiento jurídico la libertad, la justicia, la igualdad y el pluralismo político. </p> </blockquote></pre> Los navegadores suelen colocar el texto dejando margen tanto a izquierda como a derecha.	cite	Permite indicar una URL que señala al sitio de la web del que procede el texto literal.
address	Permite indicar una dirección postal. Los navegadores suelen mostrar el párrafo de dirección en cursiva y centrado.		

(3.8.4) correcciones

Permiten marcar texto para indicar como texto que se está revisando y que falta decidir su contenido final.

elemento	significado	atributos que se suelen utilizar con el elemento	
ins	Indica que el texto marcado con este elemento es provisional.	cite	URL a la dirección de la que procede el texto original
		datetime	Indica la fecha de la modificación con cuatro cifras para el año, dos para el mes y dos para el día.
del	Indica que el texto está provisionalmente marcado para su eliminación	cite	URL a la dirección de la que procede el texto original
		datetime	Indica la fecha de la modificación con cuatro cifras para el año, dos para el mes y dos para el día.

Ejemplo:

```
<p>  
La capital de Alemania es <del>Bonn</del>  
<ins datetime="20120312">Berlín</ins>  
</p>
```

Resultado:

La capital de Alemania es ~~Bonn~~Berlín

(3.9) listas

Las listas permiten crear párrafos agrupados y alineados mediante símbolos como viñetas o números y además crean párrafos alineados de forma especial para su correcta visibilidad.

(3.9.1) con viñetas

Las listas con viñetas se deben englobar dentro de un elemento **ul** (de *unordered list*, lista no ordenada), después cada párrafo de la lista estará dentro de elementos de tipo **li** (de *list item*, elemento de lista).

Ejemplo:

```
<ul>
  <li>Agua</li>
  <li>Vino</li>
  <li>Cerveza</li>
</ul>
```

Resultado:

- Lista de bebidas
- Agua
 - Vino
 - Cerveza

(3.9.2) numéricas

Las listas numéricas aparecen dentro del elemento `ol` (de *ordered list*, lista ordenada), después cada párrafo de la lista estará dentro de elementos de tipo `li`, al igual que las anteriores. La diferencia ahora es que cada párrafo con `li`, aparece con un número y no con una viñeta. Ejemplo:

```
<ol>
  <li>Agua</li>
  <li>Vino</li>
  <li>Cerveza</li>
</ol>
```

Resultado:

- Lista de bebidas
1. Agua
 2. Vino
 3. Cerveza

(3.9.3) listas anidadas

Es posible meter unas etiquetas con otras, por ejemplo:

```
<ul>
  <li>
    No alcohólicas
    <ul>
      <li>Agua</li>
    </ul>
  </li>
  <li>
    Alcohólicas
    <ul>
      <li>Vino</li>
      <li>Cerveza</li>
    </ul>
  </li>
</ul>
```

Con el resultado:

- Lista de bebidas
- No alcohólicas
 - Agua
 - Alcohólicas
 - Vino
 - Cerveza

También es posible anidar mezclando tipos de listas:

```
<ol>
<li>
  No alcohólicas
  <ul>
    <li>Agua</li>
  </ul>
</li>
<li>
  Alcohólicas
  <ul>
    <li>Vino</li>
    <li>Cerveza</li>
  </ul>
</li>
</ol>
```

Con el resultado:

- Lista de bebidas
1. No alcohólicas
 - Agua
 2. Alcohólicas
 - Vino
 - Cerveza

(3.9.4) listas de términos

Permite crear una lista de definiciones de términos. En ellas se indica el término a definir y su definición. Ejemplo:

```
<dl>
<dt>Windows</dt>
<dd>
  Sistema operativo de <strong>Microsoft</strong> disponible para PC
  disponible en versiones de 32 y 64 bits y para servidores,
  ordenadores e incluso tabletas y móviles.<br />
  La última versión es la 8 y la 2012 para servidores.
</dd>
<dt>Linux</dt>
```

```
<dd>
  Sistema operativo de código abierto disponible
  en numerosas distribuciones gratuitas y de pago.
  Es la base del sistema <strong>Android</strong>.
</dd>
<dt>Mac OS</dt>
<dd>
  Sistema operativo de los ordenadores de la empresa <strong>Apple</strong>
  <br />
  La última versión es la <strong>Snow Lion</strong>
</dd>
</dl>
```

Resultado:

Windows

Sistema operativo de **Microsoft** disponible para PC disponible en versiones de 32 y 64 bits y para servidores, ordenadores e incluso tabletas y móviles.
La última versión es la 8 y la 2012 para servidores.

Linux

Sistema operativo de código abierto disponible en numerosas distribuciones gratuitas y de pago. Es la base del sistema **Android**.

Mac OS

Sistema operativo de los ordenadores de la empresa **Apple**
La última versión es la **Snow Lion**

(3.10) enlaces

(3.10.1) URLs

Una URL (**Uniform Resource Location**) es la dirección concreta de un recurso (una página web, una imagen, un vídeo, un directorio,...) en Internet.

La gracia es que cada URL es única con lo que es una especie de DNI de un recurso. Si la URL es correcta sólo habrá un solo recurso posible al que se puede referir dicha URL.

Una URL se forma usando:

■ El **protocolo**. Ejemplos:

- http:// (para recursos de la web)
- https:// (para recursos de la web contenidos en un servidor seguro)
- ftp:// (recursos contenidos en un servidor de ficheros)
- ftps:// (recursos contenidos en un servidor de ficheros seguro)
- mailto: (dirección de correo electrónico)
- file:/// (recurso dentro de nuestra propia computadora)

■ **Servidor**. Nombre completo en Internet del servidor que aloja el recurso al que deseamos acceder. Por ejemplo: www.nasa.gov, www.centrodonbosco.es o www.jorgesanchez.net

■ **Puerto**. Puerto por el que se debe conectar con el servidor para obtener el recurso. Si no se indica (que es lo habitual) se toma el puerto por defecto. Por

ejemplo en http se usa el 80. Si queremos usar uno en particular se indica tras el servidor poniendo dos puntos y el puerto. Por ejemplo www.midb.com:1521

■ **Ruta.** Indica el recorrido dentro del servidor que hay que hacer en sus directorios para llegar al recurso que queremos. Se pone después del servidor. Ejemplos:

- [/index.html](#) Accede a la página index.html situada en la raíz del servidor.
- [/imagenes/paisajes/foto001.jpg](#) Accede a la imagen foto001.jpg dentro del directorio paisajes dentro, a su vez, del directorio paisajes

■ **Cadena de búsqueda.** Sólo aparece tras direcciones a páginas web que admitan recibir parámetros (como las páginas PHP, ASP o JSP por ejemplo).

Ejemplo [?pagina=5&idioma=es](#), pasará los parámetros [pagina](#) y [lenguaje](#) usando los valores [5](#) y [es](#) respectivamente.

Ejemplo de URL completa:

www.ejemplourl.com:9000/dirs/srv/pagina1.php?a=90&r=7

Por otro lado los caracteres presentes en una URL deben de pertenecer al ASCII, de otro modo tendremos el sempiterno problema de que no se lea bien la URL por diferentes tipos de codificación del texto. De hecho los **caracteres permitidos** en una URL son:

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
a b c d e f g h i j k l m n o p q r s t u v w x y z
0 1 2 3 4 5 6 7 8 9 0 - _ ~ .

Cualquier otro no es válido aunque sí aparecen los siguientes pero sólo para indicar elementos con significado especial (recorridos de la ruta, elementos de la parte de consulta de la URL,...). Estos símbolos están **reservados** para un uso concreto y no se pueden utilizar para ningún otro:

! # \$ % & ' () * + , / : ; = ? @ []

Si una dirección URL requiere utilizar símbolos más allá de estos definidos, entonces necesitamos codificar dicho símbolo

Si necesitamos codificar en la URL símbolos fuera de los caracteres permitidos entonces debemos codificarlos usando una notación que comienza con el símbolo % y le sigue el código hexadecimal correspondiente en el código ASCII. Así si una URL contiene un código reservado tendríamos que codificarlo mediante:

!	#	\$	%	&	'	()	*	+	,	/	:	;	=	?	@	[]
%21	%23	%24	%25	%26	%27	%28	%29	%2A	%2B	%2C	%2F	%3A	%3B	%3D	%3F	%40	%5B	%5D

La lista de símbolos que se pueden codificar es:

espacio	"	<	>	\	^	`	{		}	€	,	f	„	...	†	‡	^	‰
%20	%22	%3C	%3E	%5C	%5E	%60	%7B	%7C	%7D	%80	%82	%83	%84	%85	%86	%87	%88	%89
Š	‹	Œ	Ž	‘	’	“	”	•	–	—	~	™	š	›	œ	ž	Ÿ	ı
%8A	%8B	%8C	%8E	%91	%92	%93	%94	%95	%96	%97	%98	%99	%9A	%9B	%9C	%9E	%9F	%A1
ç	£	¥		§	“	©	≠	«	¬	-	®	-	°	±	²	³	´	µ
%A2	%A3	%A5	%A6	%A7	%A8	%A9	%AA	%AB	%AC	%AD	%AE	%AF	%B0	%B1	%B2	%B3	%B4	%B5
¶	·	¸	¹	º	»	¼	½	¾	¿	À	Á	Â	Ã	Ä	Å	Æ	Ç	È
%B6	%B7	%B8	%B9	%BA	%BB	%BC	%BD	%BE	%BF	%C0	%C1	%C2	%C3	%C4	%C5	%C6	%C7	%C8
É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ	Ò	Ó	Ô	Õ	Ö	Ø	Ù	Ú	Û	Ü
%C9	%CA	%CB	%CC	%CD	%CE	%CF	%D0	%D1	%D2	%D3	%D4	%D5	%D6	%D8	%D9	%DA	%DB	%DC
Ý	Þ	Ë	à	á	â	ã	ä	å	æ	ç	è	é	ê	ë	ì	í	î	ï
%DD	%DE	%DF	%E0	%E1	%E2	%E3	%E4	%E5	%E6	%E7	%E8	%E9	%EA	%EB	%EC	%ED	%EE	%EF
ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û	ü	ý	þ	ÿ	Salto de línea		
%F0	%F1	%F2	%F3	%F4	%F5	%F6	%F7	%F8	%F9	%FA	%FB	%FC	%FD	%FE	%FF	%0A ó %0D ó %0D%0A		

(3.10.2) crear enlaces

Dentro de una página web se pueden colocar enlaces que permitan cuando arrimamos el ratón hacer clic y que se cargue y se lance el destino de una URL.

La etiqueta que permite realizar enlaces es la etiqueta **a**. El atributo **href** permite indicar la URL a la que se realiza el salto. Ejemplo:

A Augusto le sucedió el emperador

```
<a href="http://es.wikipedia.org/wiki/Tiberio">Tiberio</a>
```

la palabra Tiberio estará remarcada de modo que al hacer clic saltaremos a la URL <http://es.wikipedia.org/wiki/Tiberio>.

Ese ejemplo muestra un salto absoluto, es decir el enlace nos lleva a una dirección URL global. Pero es posible que el salto nos dirija a un recurso presente en nuestro propio servidor. Ejemplos:

```
<!-- Salto a la página tiberio.html que estará en el mismo directorio que la actual -->
```

```
<a href="tiberio.html">Tiberio</a>
```

```
<!-- Salto a la página tiberio.html que estará dentro del directorio emperadores que estará dentro del directorio actual -->
```

```
<a href="emperadores/tiberio.html">Tiberio</a>
```

```
<!-- Salto a la página tiberio.html que estará en el directorio padre, es decir el directorio que contiene al actual -->
```

```
<a href="../tiberio.html">Tiberio</a>
```

```
<!-- Salto a la página tiberio.html que estará en el directorio emperadores, dentro del directorio padre del actual -->
```

```
<a href="../emperadores/tiberio.html">Tiberio</a>
```

(3.10.3) atributos del elemento a

Realmente el único de obligado uso es **href**, pero es posible utilizar los siguientes atributos (además de los atributos comunes a todas las etiquetas comentados anteriormente):

atributo	significado
hreflang	Permite indicar un código de lenguaje (es, fr, en,...) indicando el lenguaje en el que está escrito el destino del enlace
media	Sólo válido en HTML5, permite indicar el medio idóneo para mostrar el contenido del enlace.
target	Permite indicar cómo se muestra la página de destino. Posibilidades: <ul style="list-style-type: none">• _blank. Abre el enlace en una nueva página• _parent. Abre el enlace en el marco de la página padre de ella.• _top. Abre la página en el marco superior• _self. Abre la página en el marco actual• nombre. EL nombre indicado será el del marco en el que se abrirá la página Salvo el primero, el resto no se usan por referirse a marcos.
rel	Informa sobre la función del enlace. Puede ser: <ul style="list-style-type: none">• alternate. Enlace alternativo• author.• bookmark. Página de marcadores• help. Página de ayuda• license. Información sobre la licencia• next. Si nuestra página pertenece a una serie ordenada, el enlace nos lleva al siguiente elemento dentro de la serie.• nofollow. Marca que los robots de búsqueda de empresas como Google no tengan en cuenta los enlaces externos y así evitar que dichos enlaces en las páginas se utilicen para subir su calificación en los buscadores. Así se ignoran por los robots los enlaces marcados de esta forma.• noreferrer. Un referrer es la información relativa a la página desde la que procede el visitante a un sitio. Con este valor en los enlaces, no se indicará al destino URL la página desde la que procedía el usuario.• prefetch. Permite descargar el enlace antes de que el usuario haga clic en él y así acelerar su carga. Se usa (aunque pocos navegadores soportan este valor) en enlaces de uso habitual.• prev. Si nuestra página pertenece a una serie ordenada, el enlace nos lleva al elemento anterior dentro de la serie.• search. Página de búsqueda dentro de nuestro sitio web.• tag. Página con etiquetas de temas (tags) de nuestro sitio web.

(3.10.4) enlaces internos

Hay un estilo de enlace que permite saltar a un punto concreto del documento. Se realiza de esta forma:

- (1) Se marca una posición en el documento usando la etiqueta **a** con el atributo **id** al que se le indica un identificador que servirá de marcador de la posición. Ejemplo:

```
<a id="salto1" />
```

- (2) Ponemos el enlace donde deseemos e indicamos en el atributo **href** el nombre del marcador indicado, pero anteponiendo al nombre el símbolo de la almohadilla (**#**). Ejemplo:

```
<a href="#salto1">Ir a la posición</a>
```

Aunque **id** es el atributo aconsejado actualmente para marcar una posición, lo cierto es que tradicionalmente se usaba el obsoleto atributo **name**. Puesto que hay navegadores que no reconocen el marcado de posiciones con **id**, conviene utilizar **name** (además de **id**).

Por otro lado es posible indicar direcciones de enlace que salten a una página y dentro de ella se coloque en una posición marcada, por ejemplo:

```
< a href="http://www.colegio.edu/seccion1/actividades.html#verano">  
Ver actividades de verano  
</a>
```

El enlace del ejemplo salta a la página [actividades.html](#) localizada en la ruta indicada y además se coloca en la posición indicada por la etiqueta marcada con el valor [verano](#).

(3.11) imágenes

(3.11.1) introducción

Las imágenes son fundamentales para que una página web sea más atractiva. Los navegadores no reconocen todas las imágenes, pero sí los formatos más importantes. Fundamentalmente:

- **Formato jpg.** Son imágenes que ocupan muy poco gracias a su alta compresión. No admiten animaciones ni zonas marcadas con transparencia. Son el formato habitual de la fotografía digital.
- **Formato gif.** Imágenes con hasta 256 colores que pueden contener animaciones y zonas marcadas como transparentes, a través de las cuales se mostraría lo que esté *por debajo* de la imagen. Son buenas para hacer animaciones (los populares gif animados) y para mostrar dibujos (que no fotos) en las páginas.
- **Formato png.** Está considerado como el mejor de los tres porque aúna ambas ventajas. Permite imágenes fotográficas con alta compresión y posibilidad además de utilizar canales alfa (zonas que permiten transparencia como los gif, pero además que admiten semitransparencias, mezclando colores de la imagen con los colores de los elementos que están por debajo de la imagen).

El tamaño en disco de las imágenes puede ser mayor o menor dependiendo de su tamaño y su compresión. De modo que un tamaño grande implica más tardanza al cargar la página, pero una mayor nitidez en la imagen.

(3.11.2) inserción de imágenes

Las imágenes se colocan mediante el atributo **src**, en el cual se indica la URL (relativa o absoluta) a la imagen.

Además conviene utilizar estos atributos:

atributo	significado
alt	Indica un texto alternativo. Ese texto aparece cuando la imagen no se ha podido cargar (o durante la carga). También suele aparecer cuando arrimamos el cursor a la imagen a fin de informarnos sobre ella. Es un texto también tenido en cuenta por los buscadores a fin de identificar lo que muestra la imagen.
width	Anchura de la imagen. No es aconsejable usarle para cambiar el tamaño de la imagen, ya que si la ampliamos no se verá en buena calidad y si la reducimos estaremos cargando una imagen grande para luego mostrarla en pequeño; sería más inteligente reducirla primero con un editor de imágenes. En cualquier caso es importante utilizar este atributo para que el navegador sepa de antemano el tamaño de la imagen y así que maquete la página correctamente.
height	Altura de la imagen. Tiene las mismas connotaciones que el atributo anterior.

(3.11.3) imágenes de fondo

El elemento **body** dispone de un atributo llamado **background**, con el que se permite indicar una imagen de fondo.

(3.11.4) mapas de imagen

introducción

Se trata de una técnica que permite seleccionar partes de una imagen a fin de que esas partes formen enlaces a otras páginas.

Se utiliza en mapas propiamente dichos, en los que el usuario selecciona partes de la imagen y en imágenes donde hay elementos claramente destacados.

uso

Los mapas se basan en una imagen previamente colocada en la página web mediante la etiqueta **img**. La etiqueta que indica que la imagen es un mapa es la etiqueta **map**. En dicha etiqueta se debe utilizar el atributo **name** para poner nombre al mapa de imágenes. El contenido de **name** es un identificador (por lo que no puede tener espacios en blanco ni nada que no sean letras del alfabeto inglés, números o guion bajo) que pondrá un nombre único al mapa.

Para que el mapa se asocie a la imagen correspondiente, la etiqueta **img** de la imagen debe utilizar el atributo **usemap**. En dicho atributo se hace referencia al nombre usando el carácter **#** seguido del nombre del mapa.

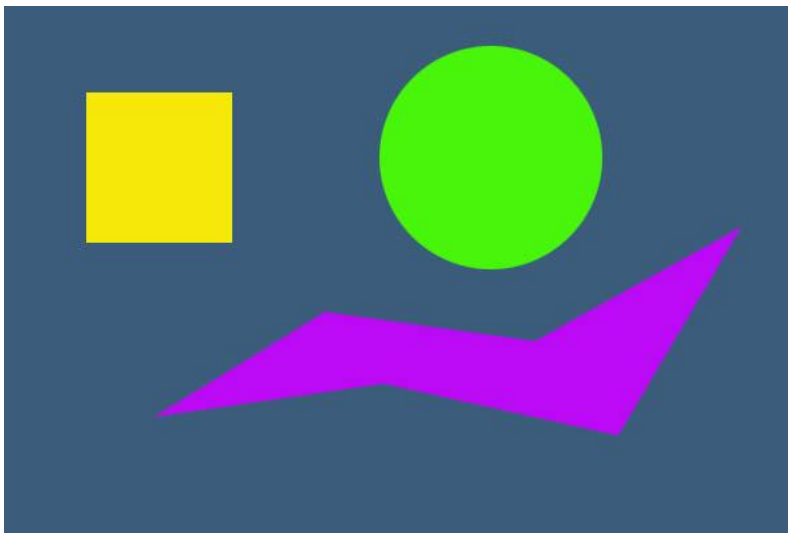
La etiqueta **map** debe contener dentro tantos elementos **area** como secciones en el mapa queramos crear. En cada elemento área rellenaremos dos atributos: **shape** (forma del área) y **coords** (coordenadas del área). Las posibilidades de este elemento son:

- **area="rect"**. Permite seleccionar un rectángulo en la imagen. En este caso el atributo **coords** indicará cuatro coordenadas. Las dos primeras marcan la coordenada X y la coordenada Y de la esquina superior izquierda de la imagen, las dos últimas las coordenadas X e Y de la esquina inferior derecha. Las coordenadas cuentan desde la esquina superior izquierda (que serían las coordenadas 0,0).
- **area="circle"**. Selecciona una región circular en la imagen. El atributo **coords** tendría tres coordenadas: las dos primeras son la coordenada X e Y del centro del círculo y la tercera el radio del mismo.
- **area="poly"**. Permite indicar una región poligonal. El atributo **coords** permite indicar las coordenadas X e Y de cada punto del polígono. Al final se cerrará el polígono.
- **area="default"**. Hace lo mismo que la opción **rect**.

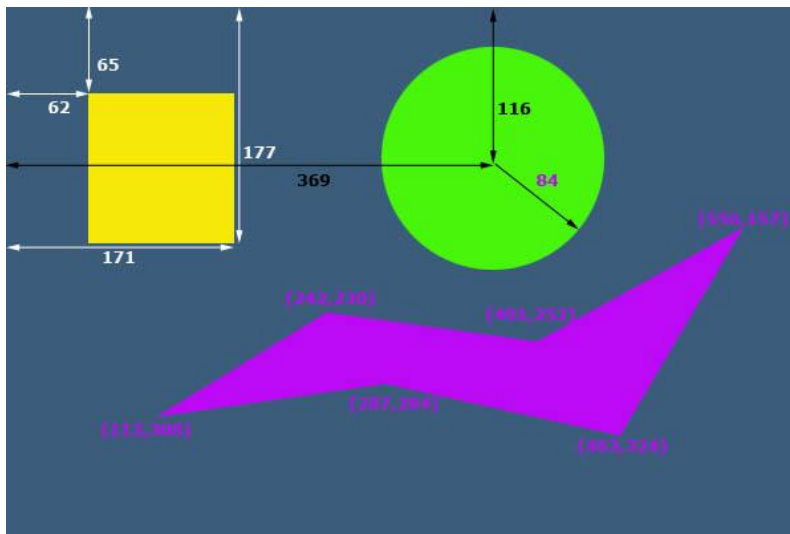
Además de estos atributos, en la etiqueta **area** deben rellenarse los atributos:

- **alt**. Texto alternativo en el área de la imagen, al igual que en la etiqueta **img**.
- **href**. Funciona como el atributo del mismo nombre en la etiqueta **a**: indica una URL a la que se dirigirá el navegador cuando el usuario haga clic sobre la imagen.

Por ejemplo en la siguiente imagen se detallan las coordenadas necesarias para hacer un mapa con tres áreas, una sobre cada forma geométrica de la misma:



Si deseamos que al hacer clic en el cuadrado, el círculo y la forma libre saltemos a otra dirección necesitamos conocer las coordenadas (en esto nos pueden ayudar los programas de retoque gráfico). En esta imagen se muestran las coordenadas necesarias:



El código HTML para hacer el mapa sería (suponiendo que la imagen se llama formas.jpg):

```

<map name="formas" id="formas">
  <area shape="rect" coords="62,65,171,177" href="rectangulo.html"
        alt="Rectángulo" />
  <area shape="circle" coords="369,116,84" href="circulo.html" alt="Círculo" />
  <area shape="poly" coords="113,308,242,230,401,252,556,157,463,324,287,284"
        href="poligono.html" alt="Polígono" />
</map>
```

El uso del atributo **id** en el elemento **map** no es obligatorio para que el mapa funcione, pero sí es necesario para cumplir la especificación XHTML donde se intentaba retirar el atributo **name** para sustituirle por **id**.

(3.11.5) elemento canvas

En inglés **canvas** significa *lienzo*, y define muy bien para que sirve este elemento creado para HTML5. Es uno de los componentes, de hecho, de HTML5 más famosos por el gran aporte que ha supuesto al dinamismo de las páginas web.

Mediante este elemento dispondremos de un área que podremos utilizar donde queramos para dibujar elementos gráficos mediante lenguaje JavaScript. Eso ha permitido (gracias a la potencia de JavaScript crear juegos, animaciones y elementos visuales atractivos en las páginas web).

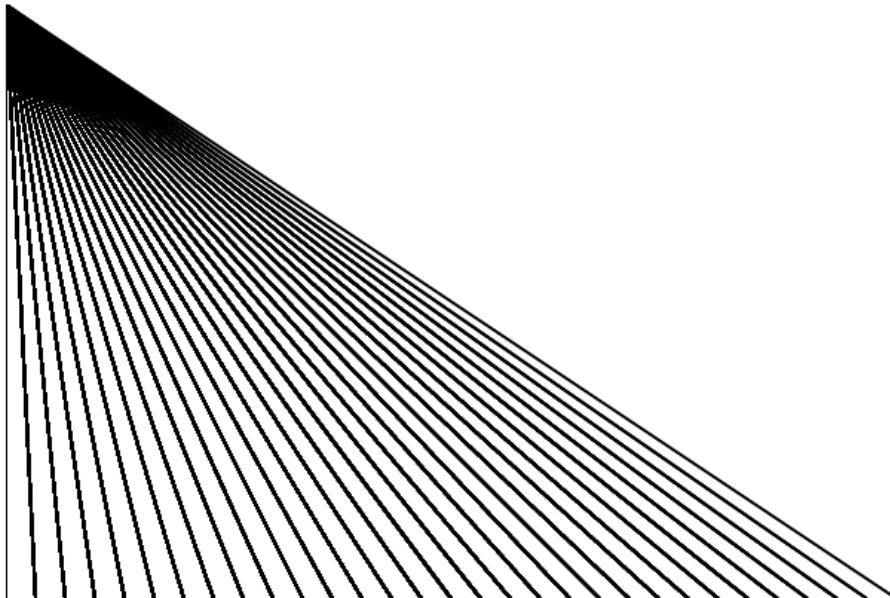
atributos de canvas

- **id**. Es el atributo que utilizan todos los elementos HTML para identificarles. En el caso de canvas es casi obligatorio su uso para poder hacer referencia al mismo.
- **width**. Anchura del lienzo (funciona igual que en el caso de **img**)
- **height**. Altura del lienzo.

Ejemplo de uso de canvas (usando JavaScript):

```
<canvas id="lienzo1" width="600" height="400" />
<script type="text/javascript">
  var canvas=document.getElementById("lienzo1");
  var contexto=canvas.getContext("2d");
  contexto.lineWidth=2;
  for(i=0;i<=600;i+=20){
    contexto.moveTo(0,0);
    contexto.lineTo(i,400)
    contexto.stroke();
  }
</script>tablas
```

Mediante la etiqueta script podemos colocar código en lenguaje JavaScript, desde ese código podemos utilizar el lienzo para dibujar. El resultado del código anterior es la imagen:



En cualquier caso es un elemento que no está reconocido en muchos navegadores (aunque sí en todos los modernos).

(3.11.6) etiqueta svg

El lenguaje SVG es un lenguaje ya veterano basado en XML que sirve para dibujar gráficos en una página web. Está aceptado desde hace tiempo por la W3C, sin embargo pocos navegadores soportan todavía esta etiqueta. Sin embargo HTML 5 sí soporta el uso de SVG.

La forma de incorporar SVG en un documento HTML 5 es mediante la etiqueta svg:

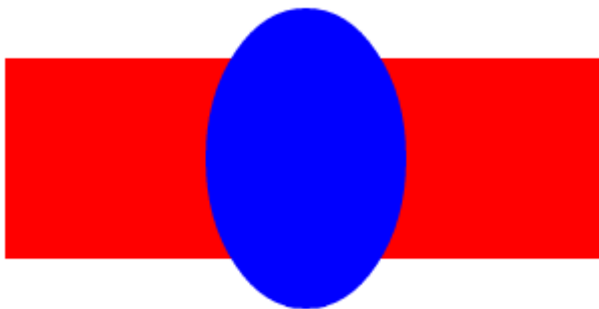
```
<svg xmlns="http://www.w3.org/2000/svg">
  ..etiquetas svg
</svg>
```

Dentro del elemento **svg** se colocan las etiquetas que permiten dibujar en el área del svg. En principio si no indicamos tamaño, todo el área de la página web se toma para dibujar en svg, pero podemos indicar los atributos **width** y **height** para indicar tamaños concretos.

Las etiquetas que se pueden utilizar dentro de svg pertenecen al lenguaje SVG y rebasan el propósito de este manual, pero un ejemplo de ellas sería:

```
<svg xmlns="http://www.w3c.org/2000/svg" height="400">
  <rect id="rect1" x="50" y="50" width="300" height="100" fill="red" />
  <ellipse id="elips1" cx="200" cy="100" rx="50" ry="75" fill="blue" />
</svg>
```

El resultado:



(3.11.7) lenguaje MathML. etiqueta math

El lenguaje MathML es un clásico de los lenguajes derivados de XML y ya se considera parte de HTML 5. Aunque no sirve para incrustar imágenes, permite colocar fórmulas matemáticas por complejas que sean.

No todos los navegadores soportan añadir código MathML a una página web, pero poco a poco cada vez son más los que lo permiten.

Ejemplo de código MathML:

```
<math>
  <msub><mi>x</mi><mtext>0</mtext></msub>
  <mo>=</mo>
  <mn>2</mn>
  <msup>
    <mfenced open="(" close=")">
      <mfrac>
        <msup>
          <mi>y</mi><mn>2</mn>
        </msup>
        <msup>
          <mi>z</mi><mn>3</mn>
        </msup>
      </mfrac>
    </mfenced>
  </msup>
  <mn>2</mn>
</math>
```


Obtiene el resultado:

$$x_0 = 2\left(\frac{y^2}{z^3}\right)^2$$

(3.12) tablas

(3.12.1) introducción

Las tablas son uno de los elementos fundamentales de HTML para mejorar la puesta visual de las páginas web. Una tabla normal es un conjunto de filas y columnas en las que se dispone el texto o las imágenes que deseemos.

Las tablas HTML permiten todo tipo de formato y en especial permiten unir varias celdas para formar tablas con una disposición irregular, lo que permiten diseñar disposiciones muy complejas en las páginas. Aún más: se permite incluso colocar una tabla dentro de una celda lo que aún permite más libertad en la maquetación de páginas.

(3.12.2) tablas simples

La definición de una tabla comienza con la etiqueta **table**, por cada fila se indica **tr** y por cada columna se indica **td**. En HTML 5 y XHTML a la etiqueta **table** sólo se le permite el atributo **border** para indicar la anchura del borde de la tabla (además de, por supuesto, los atributos comunes a todos los elementos de HTML como **id**, **lang**,...).

El atributo **width** no se recomienda utilizar, ya que sus posibilidades se hacen en todos los navegadores, pero es muy utilizado (junto a otros atributos que ya no se recomiendan) para determinar la anchura de una tabla.

Ejemplo de tabla:

```
<table border="1">
  <tr>
    <td></td>
    <td>Lunes</td>
    <td>Martes</td>
    <td>Miércoles</td>
    <td>Jueves</td>
    <td>Viernes</td>
  </tr>
  <tr>
    <td>10:30</td>
    <td>Matemáticas</td>
    <td>Geografía</td>
    <td>Física</td>
    <td>Dibujo</td>
    <td>Matemáticas</td>
  </tr>
```

```

<tr>
  <td>11:30</td>
  <td>Inglés</td>
  <td>Lenguaje</td>
  <td>Geografía</td>
  <td>Química</td>
  <td>Física</td>
</tr>
</table>

```

Salen:

	Lunes	Martes	Miércoles	Jueves	Viernes
10:30	Matemáticas	Geografía	Física	Dibujo	Matemáticas
11:30	Inglés	Lenguaje	Geografía	Química	Física

atributos de la etiqueta **table**

En realidad actualmente en los estándares tanto XHTML como HTML5 sólo se permite el atributo **border**, pero aun funcionan todos los que vamos a comentar aquí en los navegadores. La razón de que no se usen se debe a que mediante CSS podemos conseguir un mejor resultado. Los atributos son:

atributo	significado
border	Indica la anchura en píxeles del border de la tabla
cellpadding	Especifica el espacio en píxeles entre el borde de la celda y el contenido de la misma. En definitiva es el espacio interior de la celda.
cellspacing	Espacio en píxeles existente entre celda y celda
width	Anchura de la tabla, puede ser en píxeles o en porcentaje sobre la anchura de la página (usando el signo %)
rules	Indica qué bordes de la tabla se mostrarán. Posibilidades: <ul style="list-style-type: none"> • none. Sin líneas • groups. Pone líneas sólo para separar las etiquetas de grupos • rows. Bordes horizontales (entre filas) • cols. Bordes verticales (entre columnas) • all. Todos los bordes de la tabla

No se recomienda oficialmente usar las cuatro últimas, pero lo cierto es que están muy aceptadas y casi todos los diseñadores las siguen usando preferentemente sobre la opción de usar CSS en su lugar.

(3.12.3) celdas de cabecera

Es muy habitual que las tablas muestren datos y que estos posean celdas que sirvan para describirles. Esas celdas se consideran de cabecera y se marcan con **th**. Ejemplo:

```
<table border="1">
  <tr>
    <th>&nbsp;</th>
    <th>Lunes</th>
    <th>Martes</th>
    <th>Miércoles</th>
    <th>Jueves</th>
    <th>Viernes</th>
  </tr>
  <tr>
    <th>10:30</th>
    <td>Matemáticas</td>
    <td>Geografía</td>
    <td>Física</td>
    <td>Dibujo</td>
    <td>Matemáticas</td>
  </tr>
  <tr>
    <th>11:30</th>
    <td>Inglés</td>
    <td>Lenguaje</td>
    <td>Geografía</td>
    <td>Química</td>
    <td>Física</td>
  </tr>
</table>
```

En el resultado sólo se aprecia que el navegador colorea en negrita las celdas de cabecera. Pero con ayuda de CSS podríamos diferenciarlas más (como veremos en la siguiente unida).

	Lunes	Martes	Miércoles	Jueves	Viernes
10:30	Matemáticas	Geografía	Física	Dibujo	Matemáticas
11:30	Inglés	Lenguaje	Geografía	Química	Física

(3.12.4) títulos en las tablas

A las tablas se les puede poner un título con ayuda de la etiqueta **caption**. Ejemplo:

```
<table border="1">
  <caption>Ventas</caption>
  <tr>
    <th>Hardware</th>
    <td>12.190 €</td>
  </tr>
  <tr>
    <th>Software</th>
    <td>9.870 €</td>
  </tr>
</table>
```

Resultado:

Hardware	12.190 €
Software	9.870 €

(3.12.5) etiquetas de agrupación de elementos de una tabla

Hay tres elementos HTML que sirven para diferenciar las tres partes principales de una tabla, son:

- **thead**. Sirve para indicar las filas que forman la cabecera de la tabla
- **tfoot**. Indica el pie de la tabla
- **tbody**. Indica el cuerpo de la tabla

De esa forma se podrá más adelante dar formato diferencia a cada parte. Ejemplo de uso:

```
<table border="1" rules="groups">
  <caption>Ventas por secciones</caption>
  <thead>
    <tr>
      <td>&nbsp;</td>
      <td>Hardware</td>
      <td>Software</td>
    </tr>
  </thead>
```

```
<tfoot>
  <tr>
    <th>Total</th>
    <th>25000</th>
    <th>22000</th>
  </tr>
</tfoot>
<tbody>
  <tr>
    <th>Enero</th>
    <td>12000</td>
    <td>15000</td>
  </tr>
  <tr>
    <th>Febrero</th>
    <td>13000</td>
    <td>9000</td>
  </tr>
</tbody>
</table>
```

Resultado:

Ventas por secciones

	Hardware	Software
Enero	12000	15000
Febrero	13000	9000
Total	25000	22000

(3.12.6) combinar celdas

Es posible unir celdas y de esta forma conseguir tablas de formas caprichosas que permiten una maquetación más poderosa.

Las etiquetas de columna (**td** y **th**) son las que poseen los atributos que permiten esta operación. En concreto son los atributos:

atributo	significado
colspan	Combina la celda actual con el número de celdas a la derecha que se indique. Por ejemplo <i>colspan="3"</i> une esta celda con las dos que tiene a su derecha, formando una combinación de tres celdas en horizontal.
rowspan	Combina la celda actual con el número de celdas hacia abajo que se indique. Por ejemplo <i>rowspan="3"</i> une esta celda con las dos que tiene hacia abajo, formando una combinación de tres celdas en vertical.

Ejemplo de uso:

```
<table border="1" width="100%">
  <tr>
    <td>&nbsp;</td>
    <td>&nbsp;</td>
    <td colspan="2">&nbsp;</td>
  </tr>
  <tr>
    <td>&nbsp;</td>
    <td colspan="2">&nbsp;</td>
    <td rowspan="2">&nbsp;</td>
  </tr>
  <tr>
    <td rowspan="3">&nbsp;</td>
    <td>&nbsp;</td>
    <td>&nbsp;</td>
  </tr>
  <tr>
    <td colspan="3">&nbsp;</td>
  </tr>
  <tr>
    <td colspan="3">&nbsp;</td>
  </tr>
</table>
```

Resultado:

(3.12.7) tablas dentro de tablas

Para generar tablas aún más complejas, se pueden meter elementos **table**, dentro de otras tablas. Lo que se hace es meter una etiqueta **table** (con todos sus elementos de fila y columna) en un **td** o **th**.

Esta combinación permite realizar formatos de tabla a capricho y de esa forma conseguir disposiciones de páginas extremadamente complejas.

(3.13) elementos span y div

(3.13.1) elemento div

Se trata de un elemento clásico (presente en las primeras versiones de HTML) que permite agrupar el contenido. Inicialmente se usaba para usar su atributo **align** para que todo un bloque de párrafos tuviera la misma alineación. Posteriormente se ha utilizado con CSS a fin de definir secciones y así dar un formato concreto a toda la sección.

En especial se ha utilizado para definir capas en las páginas web y ese sigue siendo su principal uso.

Dentro de **div** se pueden colocar todo tipo de etiquetas (tablas, párrafos, imágenes,...)

La mayoría de navegadores no da un formato concreto a div, pero si deja un espacio y otro por detrás de la etiqueta.

(3.13.2) elemento span

Es muy similar al anterior, pero como los navegadores no agregan nada (ni siquiera espacios) a esta etiqueta cuando se muestra por pantalla, en lugar de para definir capas (aunque se podría) se usa para marcar contenido dentro de un párrafo, a fin de que a ese contenido se le pueda dar un formato especial mediante CSS. Ejemplo:

Dentro de este texto `` esta frase sale de color azul``. Esto vuelve a salir normal

`color:blue` es código CSS que permite colorear de color azul (como se verá en la siguiente unidad).

(3.14) formularios

Los formularios son un elemento en las páginas web que permiten recabar información para enviarla a un servidor de aplicaciones y que la procese. Por ejemplo podemos colocar cuadros de texto en los que el usuario pone sus datos y estos pasan a una página **PHP** que tendrá la capacidad de recoger dichos datos y actuar en consecuencia.

Ejemplo de formulario:

Escriba su nombre

Escriba sus apellidos

Escriba su sexo:
 Hombre Mujer

(3.14.1) etiqueta form

Todo formulario HTML comienza con una etiqueta **form**, dentro de ella se colocan todos los controles del formulario.

Fundamentalmente la etiqueta **form** posee dos atributos:

- **action**. Es el atributo que contiene la URL de la página web o servicios que procesará los datos del formulario.
- **method**. Permite indicar qué instrucción http utilizaremos para pasar la información al destino de nuestro formulario. Las opciones habituales son **GET** y **POST**. No se distingue entre mayúsculas y minúsculas. La diferencia es que GET genera una cadena de búsqueda en la URL que contiene los datos del formulario; POST, por su parte, pasa los datos pero de forma más oculta.

Ejemplo:

```
<form action="control_form.php" method="GET">
...
</form>
```

(3.14.2) cuadros de texto, input type="text"

Los cuadros de texto permiten recoger texto que escriba el usuario. Su sintaxis es:

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

Es saca en la página un cuadro en el que el usuario puede introducir texto. Además podemos utilizar estos atributos:

- **value**. Da un valor inicial al cuadro, se usa para indicar un texto de ayuda al relleno.
- **id**. Identificador del cuadro. La W3C (organismo que estandariza XML y HTML) recomienda usar **id** en lugar de **name**, pero lo cierto es que PHP recoge los valores gracias a **name**, no funciona con **id**. Por lo que para más seguridad los diseñadores suelen utilizar ambos atributos.

Ejemplo:

```
<form action="control_form.php" method="get">
Escriba su nombre y apellidos<input type="text" name="nombre" />
</form>
```

Ese código da como resultado:

Escriba su nombre y apellidos

atributos de los cuadros de texto

Todos los cuadros de texto tienen los siguientes atributos

atributo	posibles valores	uso
maxlength	Números	Indica el máximo número de caracteres que se le permitirá escribir a la usuaria o usuario.
size	Números	Anchura, en caracteres, del cuadro de texto. No tiene sentido que sea mayor que el anterior (sí que sea menor)
value	Texto	Permite rellenar el cuadro con un texto inicial

En realidad estos atributos les usan todos los tipos de cuadros.

(3.14.3) cuadro de contraseñas, **input type="password"**

Funcionan como los cuadros de texto, sólo que el texto que se introduce se oculta, mostrando sólo puntos o asteriscos. La sintaxis es:

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

Usa los mismos atributos que los cuadros de texto. Si usamos método GET, la contraseña es visible en la parte superior del navegador. Con POST esto no ocurre, pero aún así podríamos averiguarla. Por ello lo ideal es pasar cifrada la contraseña.

(3.14.4) botones

Los botones son controles del formulario en los que no se puede escribir, lo que sí permiten es cargar una imagen mediante su atributo **src** (que funciona como el atributo **src** de las imágenes)

botón de envío, **input type="submit"**

Sirve para llevar a cabo la comunicación entre el formulario y la página que recoge sus datos. La sintaxis es:

```
<input type="submit" value="texto del botón" />
```

En cuanto se pulsa este botón, los datos del resto de controles se envían a la página receptora del formulario (que se corresponde a la URL indicada en el apartado **action** de la etiqueta **form**).

botón de restablecer, **input type="reset"**

La sintaxis de este botón es:

```
<input type="reset" value="texto del botón" />
```

Este tipo de botones lo que hacen es anular los datos que se han introducidos en los controles del formulario. Deja todos los controles en su estado inicial.

botón genérico, `input type="button"`

Un botón genérico se marca indicando `type="button"` en la etiqueta `type`. En los formularios no se usa para enviar o configurar la información, sino que se utiliza normalmente para capturar su pulsación (mediante `JavaScript` es lo más habitual) y responder en consecuencia.

(3.14.5) botones de radio

Se trata de un control fácil de usar que permite elegir una de entre varias opciones. Todas las opciones deben de tener el mismo nombre y sólo cambia el valor de las mismas. Ejemplo:

```
<form action="control_form.php" method="get">
  <p>Sexo:</p>
  <input type="radio" name="sexo" value="hombre"/> Hombre<br />
  <input type="radio" name="sexo"
    value="mujer" checked="checked"/> Mujer<br />
</form>
```

El resultado es:

Sexo:

- Hombre
- Mujer

El atributo `checked` (que sólo admite el valor `checked`) hace que el botón en el que se usa, aparezca chequeado por defecto

(3.14.6) casillas de verificación

Se usan igual que los botones de radio, aunque no se recomienda agrupar varios con el mismo nombre; no porque no se pueda, sino porque las casillas se asocian a valores que se activan o desactivan.

La diferencia es que el tipo (`type`) es `checkbox` en lugar de `radio`. También posee el atributo `checked` para que inicialmente la casilla aparezca pulsada.

Ejemplo:

```
<form action="control_form.php" method="get">
  <p>Sexo:</p>
  <input type="checkbox" name="sexo" value="hombre"/> Hombre<br />
  <input type="checkbox" name="sexo" value="mujer"
    checked="checked"/> Mujer<br />
  Estado civil: Casado <input type="checkbox" name="estadocivil"
    value="casado" checked="checked"/>
  <br />
  <input type="submit" value="Enviar" />
</form>
```

El resultado:

Sexo:

Hombre

Mujer

Estado civil: Casado

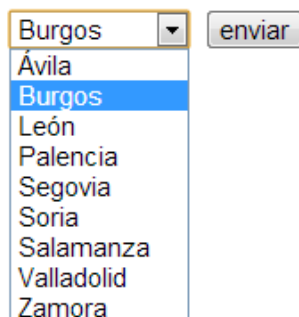
(3.14.7) cuadros combinados

Un cuadro combinado permite el uso de una lista de opciones en la que se puede elegir la deseada. Todo cuadro comienza con una etiqueta **select** que es la encargada de dar nombre (**name**) al control. Dentro cada opción del cuadro se indica con una etiqueta **option** a la que se da valor mediante el atributo **value**. Dentro de la etiqueta **option** se coloca el texto que verá el usuario.

Ejemplo:

```
<form action="control_form.php" method="get">
  <select name="provincia">
    <option value="av">Ávila</option>
    <option value="bu">Burgos</option>
    <option value="l">León</option>
    <option value="p">Palencia</option>
    <option value="sg">Segovia</option>
    <option value="so">Soria</option>
    <option value="sa">Salamanca</option>
    <option value="va">Valladolid</option>
    <option value="za">Zamora</option>
  </select>
  <input type="submit" value="enviar"/>
</form>
```

Resultado:

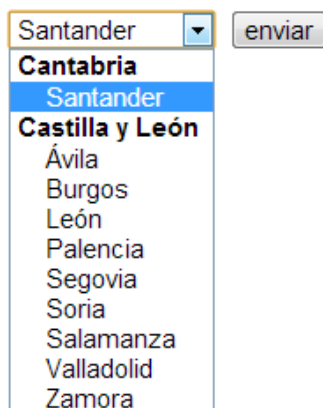


The screenshot shows a web form with a dropdown menu and a submit button. The dropdown menu is open, showing a list of provinces: Ávila, Burgos, León, Palencia, Segovia, Soria, Salamanca, Valladolid, and Zamora. 'Burgos' is currently selected and highlighted in blue. To the right of the dropdown is a button labeled 'enviar'.

Se pueden agrupar opciones dentro del cuadro usando el elemento **optgroup**. Ejemplo:

```
<form action="control_form.php" method="get">
  <select name="provincia">
    <optgroup label="Cantabria">
      <option value="s">Santander</option>
    </optgroup>
    <optgroup label="Castilla y León">
      <option value="av">Ávila</option>
      <option value="bu">Burgos</option>
      <option value="l">León</option>
      <option value="p">Palencia</option>
      <option value="sg">Segovia</option>
      <option value="so">Soria</option>
      <option value="sa">Salamanca</option>
      <option value="va">Valladolid</option>
      <option value="za">Zamora</option>
    </optgroup>
  </select>
  <input type="submit" value="enviar"/>
</form>
```

Da como resultado:



The image shows a web browser window with a dropdown menu. The dropdown menu is open, showing a list of provinces. The first province is 'Cantabria', with 'Santander' selected. Below it, the provinces of 'Castilla y León' are listed, including 'Ávila', 'Burgos', 'León', 'Palencia', 'Segovia', 'Soria', 'Salamanca', 'Valladolid', and 'Zamora'. To the right of the dropdown menu is a button labeled 'enviar'.

(3.14.8) cuadro de selección de archivo

Cuando se usa `type="file"` en una etiqueta `input`, entonces aparece un botón que al pulsarlo hace aparecer un cuadro de selección de archivos mediante el cual podremos elegir un archivo. La ruta local a dicho archivo es lo que se guarda para ser enviado al sitio o página que recibe los valores del formulario. Ejemplo:

```
<form action="recogida1.php" method="get">
  Elija el archivo
  <input type="file" name="archivo" />
  <input type="submit" value="enviar"/><br />
</form>
```

Resultado:

Elija el archivo No se ha seleccionado ningún archivo

Al pulsar *seleccionar archivo* aparece un cuadro, cuando elijamos el archivo aparece su ruta, que será lo que se envíe con el formulario..

(3.14.9) cuadro de texto multilinea

La etiqueta `textarea` permite coloca un cuadro de texto de varias líneas para que el usuario puede introducir un texto largo. El atributo `rows` permite indicar la altura en líneas de texto del cuadro y el atributo `cols`, la anchura en caracteres (los demás atributos son como los de la etiqueta `input type="text"`). Ejemplo:

```
<form action="control_form.php" method="get">  
  Escribe un texto descriptivo: <br />  
  <textarea rows="10" cols="40" name="texto"></textarea><br />  
  <input type="submit" value="enviar"/><br />  
</form>
```

Resulta:

Escribe un texto descriptivo:

Entre la etiqueta `textarea` se puede colocar texto que aparecerá inicialmente dentro del cuadro.

(3.14.10) agrupación de controles, fieldset

La etiqueta `fieldset` permite agrupar controles para que visualmente sea más cómodo el relleno de los controles. Visualmente los controles aparecerán recuadrados y se suele utilizar una etiqueta inmediatamente interior a `fieldset` que es `legend` que contiene el texto que encabezará al grupo de controles.

Ejemplo:

```
<form action="control_form.php" method="get">
  <fieldset >
    <legend>Datos personales</legend>
    <input type="radio" name="sexo" value="hombre"/>Hombre<br />
    <input type="radio" name="sexo" value="mujer"
      checked="checked"/>Mujer<br />
    Estado civil: Casado <input type="checkbox" name="estadocivil"
      value="casado" checked="checked"/>
    <br />
  </fieldset>
  <input type="submit" value="Enviar" />
</form>
```

El resultado:



(3.14.11) controles de HTML 5

El nuevo estándar HTML ha mejorado notablemente los formularios gracias a nuevos controles. La pega es que no todos los controles son soportados por los navegadores; hay controles que algunos navegadores no permiten utilizar. cuando un navegador no comprende un control HTML 5, suele traducirle como un cuadro de texto normal.

cuadros de texto especializados

La diferencia entre ellos es el uso del atributo **type**. Los buenos navegadores que reconocen estos cuadros, varían su uso para adaptarles a su función y así facilitar al usuario la entrada de datos. De hecho los navegadores validan los datos y no les dejan enviar hasta que sean correctos. Posibilidades:

- **input type="number"**. Acepta sólo números. El navegador se asegura de no aceptar texto. El atributo **maxlength** permite indicar un valor máximo para el cuadro. Si, por ejemplo, ponemos **maxlength="5"** y entonces sólo se aceptarán cinco caracteres.
- **input type="email"**. Acepta sólo direcciones de correo electrónico
- **input type="url"**. Acepta sólo direcciones URL.
- **input type="date"**. Acepta sólo fechas válidas. Usa el formato de fecha configurado en el sistema operativo del usuario que visita la página. Los

navegadores además proporcionan un cuadro visual más sencillo para recoger la fecha.

- **input type="time"**. Acepta sólo horas válidas; funciona igual que el cuadro anterior.
- **input type="datetime"**. Acepta fecha y hora.
- **input type="month"**. Acepta sólo números del 1 al 12, referidos a un mes.
- **input type="search"**. Presenta un cuadro de texto pensado para hacer búsquedas.
- **input type="tel"**. Permite introducir números de teléfono.
- **input type="range"**. Presenta un control para elegir datos entre un rango. Los atributos **max** y **min** establecen el rango máximo y mínimo del control. El atributo **step** indica cuánto se mueve el control (si de uno en uno, de dos en dos,...).
- **input type="color"**. Presenta un control de selección de colores. El color se toma en formato **#xxxxxx** donde cada **x** es una cifra hexadecimal. Es decir lo toma en el formato habitual de colores de HTML.

etiquetado de controles

En lugar de poner la información de los controles poniendo texto directamente, se recomienda usar la etiqueta **label** que utiliza un atributo **for** que sirve para asociar el texto interno a la etiqueta **label** respecto al nombre del control que se indica con el atributo **for**.

Esto permite una mejor usabilidad (los navegadores reconocen **label** y, por ejemplo, al hacer clic en el texto de la etiqueta, el control asociado recibe el foco).

El ejemplo anterior usando **label** sería:

```
<form action="control_form.php" method="get">
  <label for="texto">Escribe un texto descriptivo:</label>
  <textarea rows="10" cols="40" name="texto" ></textarea><br />
  <input type="submit" value="enviar"/><br />
</form>
```

datalist

No hay muchos navegadores que acepten esta etiqueta, pero es muy potente permite añadir entradas a un control de cuadro de texto (y también a cuadros especializados como los de email, url,...). La forma de usar consiste en usar el atributo HTML5 **list** existente en la etiqueta **input**. Ese atributo asociará el cuadro de texto a la lista de valores. Luego dentro de **datalist** se colocan etiquetas **option** para cada opción en la lista.

Ejemplo:

```
<form action="recogida.php" method="get">
<label for="gustos">
  Escribe en qué te gusta pasar tu tiempo de ocio
</label>
<input type="text" id="gustos" name="gustos" list="listaGustos" />
<datalist id="listaGustos">
  <option label="deportes" value="Deportes" />
  <option label="teatro" value="Teatro" />
  <option label="cine" value="Cine" />
  <option label="leer" value="Leer" />
</datalist>
<input type="submit" value="enviar"/><br />
</form>
```

No aparece un cuadro combinado, será un cuadro de texto que permitirá que aparezca la lista de opciones. Pero podremos escribir lo que queramos.

Sin embargo esta forma no funciona en la mayoría de navegadores y por ello se usa un truco que hace que los navegadores no compatibles con HTML5 vean la lista como un cuadro combinado y los compatibles como un cuadro de texto en el que se podría escribir:

```
<form action="recogida.php" method="get">
<label for="gustos">
  Escribe en qué te gusta pasar tu tiempo de ocio
</label>
<datalist id="listaGustos">
  <select name="listaGustos">
    <option value="deportes" >Deportes</option>
    <option value="teatro" >Teatro</option>
    <option value="cine" >Cine</option>
    <option value="leer" >Leer</option>
  </select>
</datalist>
<input id="gustos" name="gustos" list="listaGustos" />
<input type="submit" value="enviar"/><br />
</form>
```

El truco consiste en meter **select** dentro del **datalist**, los navegadores modernos ignorarán la etiqueta **select** y los viejos el **datalist**.

atributo **required**

Este atributo obliga a rellenar con algún valor el control en el que se usa. Es decir hace que un determinado control sea de obligado rellenado en un formulario. Uso:

```
<input type="text" name="texto" id="texto" required />
```

O bien, usando una forma más compatible con XMLN:

```
<input type="text" name="texto" id="texto" required="required" />
```


atributo **multiple**

Permite en los cuadros de entrada de texto que el usuario pueda indicar varios valores si les separa con comas.

atributo **pattern**

Permite colocar una expresión regular en un cuadro de texto que, obligatoriamente, tendrá que cumplir el cuadro en el que se use el atributo. Ejemplo (cuadro de texto que sólo acepta introducir 5 letras mayúsculas y tres números):

```
<form action="recogida1.php" method="get">
  <label for="texto">
    Escribe el nº de serie (5 letras y tres números)
  </label>
  <input type="text" pattern="[A-Z]{5}[0-9]{3}" id="gustos" name="texto"
  />
  <input type="submit" value="enviar"/><br />
</form>
```

atributo **placeholder**

Un **placeholder** es un texto que ayuda a rellenar un cuadro de un formulario (está especialmente pensado para los cuadros de texto) colocando un texto inicial en el cuadro que se va en cuanto el cuadro obtiene el foco del usuario (porque, por ejemplo, el usuario le hace clic). Ejemplo:

```
<form action="recogida1.php" method="get">
  <label for="texto">
    Escribe el nº de serie
  </label>
  <input type="text" pattern="[A-Z]{5}[0-9]{3}" id="gustos" name="texto"
  placeholder="5 letras y tres números" />
  <input type="submit" value="enviar"/><br />
</form>
```

El resultado es:

Escribe el nº de serie

atributo **autocomplete**

Permite activar (valor **on**) o desactivar (valor **off**) el autocompletado del navegador. El autocompletado es la opción que permite a los usuarios cuando rellenan un formulario ver entradas habituales que han escrito en el mismo u otros formularios. a veces conviene desactivar cuando lo que se escribe son datos sensibles (nombre de usuario, contraseñas).

atributos **min,max y range**

Son atributos que se pueden utilizar en muchos tipos de cuadros (**number, date, time, range,..**) que almacenan valores numéricos o similares. **min** y **max** establecen los límites del cuadro. Por ejemplo si el cuadro es numérico y ponemos **min=1** y **max=50** y eso impedirá poner valores en el cuadro fuera de esos topes. El parámetro **step** indica el mínimo incremento de valor en el cuadro; si en el ejemplo anterior ponemos **step=3** del

valor 1 saltaremos al 4. Los controles del cuadro (en el caso de los cuadros numéricos, las flechitas) obedecen a esa configuración.

(3.15) etiquetas de cabecera

El elemento **head** de las páginas web sirve para declarar elementos que no aparecen dentro del contenido de la página pero que son muy importantes para su correcto funcionamiento.

(3.15.1) elemento title

Title es obligatorio en las páginas web, sirve para indicar el título de la página (que los navegadores suelen mostrar en la barra de título de la ventana de la página).

Es un elemento muy importante porque los buscadores otorgan a su texto prioridad en las búsquedas que realizan. De modo que una página puede salir más arriba o abajo en los resultados del buscador dependiendo del resultado de las páginas.

(3.15.2) elemento base

No es muy utilizado pero permite indicar la raíz desde la que los enlaces en la página tomarán para sus relativos. Si no se indica, los enlaces relativos parten del directorio que alberga a la página, de esta forma podemos hacer que los enlaces partan del directorio que digamos. Ejemplo:

```
<!DOCTYPE html>
<html lang="es-ES">
  <head>
    <meta charset="UTF-8">
    <base href="http://www.mipagina.com/conts" />
    <title>Página 3</title>
  </head>
```

Todos los enlaces relativos se buscarán en la dirección indicada. Así el enlace:

```
<a href="pg4.html">
```

Ese enlace saltará a la dirección <http://www.mipagina.com/conts/pg4.html>

base es un elemento vacío.

(3.15.3) elemento link

Permite invocar a un recurso externo a la página a fin de incorporar a la misma el contenido del recurso. Usa los siguientes atributos (obligatorios son **href** y **rel**):

atributo	uso
rel	<p>Indica la relación del recurso con la página. Posibilidades:</p> <ul style="list-style-type: none">• stylesheet. Es la que más se utiliza, indica que el recurso es una hoja de estilos (CSS).• alternate. Indica que el recurso es una versión alternativa del documento actual. Se usa bastante.• icon. Indica que el recurso es el icono de la página web. Se suele usar icon para iconos de tamaño 60x60, mientras que se usa short icon para iconos de tamaño 12x12 (normalmente se usan ambas entradas, escribiendo una etiqueta link distinta para cada tipo de icono). <p>Los siguientes, apenas se usan:</p> <ul style="list-style-type: none">• author. Información sobre el autor.• next. Documento siguiente al actual si se trata de parte de una serie de documentos relacionados.• prev. Documento anterior al actual si se trata de parte de una serie de documentos relacionados.• first. Es la primera página de la serie de documentos relacionados• last. Es la última página de la serie de documentos relacionados• up. Página de nivel superior en la estructura del sitio web.• prefetch. Hace una carga previa del recurso en los tiempos en los que no se está haciendo nada (es decir de forma invisible al usuario).• bookmark. El recurso es un mapa de navegación. Contendrá enlaces.• external. Es un recurso externo de tipo indefinido.• help. Es una página de ayuda.• search. Es una página de búsqueda.• license. Información sobre la licencia.• tag. Página de etiquetas.• nofollow. Marca que los robots de búsqueda de empresas como Google no tengan en cuenta los enlaces externos y así evitar que dichos enlaces en las páginas se utilicen para subir su calificación en los buscadores. Así se ignoran por los robots los enlaces marcados de esta forma.• noreferrer. Un referrer es la información relativa a la página desde la que procede el visitante a un sitio. Con este valor en los enlaces, no se indicará al destino URL la página desde la

atributo	uso
	que procedía el usuario. <ul style="list-style-type: none"> • pingback. Es un recurso que solicita información al que le enlaza.
href	Dirección URL al recurso solicitado; puede ser relativa o absoluta.
hreflang	Propio de HTML 5. Indica el lenguaje en el que está escrito el recurso.
type	Tipo MIME del documento relacionado. Los tipos MIME son indicaciones pertenecientes al protocolo http que usan los servidores web para indicar el tipo de información. Así por ejemplo text/css es el tipo MIME que indica que un recurso es una hoja de estilos CSS.
media	Permite indicar el tipo de dispositivo más adecuado para mostrar el recurso.

(3.15.4) etiqueta style

Permite colocar código CSS en la página web. Se usa un atributo **type** para indicar el tipo de contenido que casi unánimemente será **text/css**:

```
<style type="text/css">
  p{
    font-face: Arial, Helvetica, sans-serif;
  }
</style>
```

Ese código hace que el texto de las etiquetas p, use tipo de letra **Arial** en primera opción, **Helvetica** en segunda y **sans-serif** en tercera.

(3.15.5) etiqueta script

Permite colocar código script en la página web. Hoy en día lo habitual es utilizar esta etiqueta para incorporar código **JavaScript** a la misma.

Hoy en día no se suelen usar atributos pero podría usar los siguientes:

atributo	uso
type	Tipo MIME de contenido que se colocará dentro de la etiqueta. El habitual es text/javascript . Es el que se asume por defecto.
charset	Codificación usada para el texto del script. Por defecto toma el de la página
src	Permite indicar la URL a un archivo externo que contendrá código (normalmente en JavaScript) que se colocará en la página.

(3.15.6) etiqueta meta

La etiqueta meta permite indicar información extra a la página web para que sea recogida por otro software o para que el propio protocolo http la tenga en cuenta.

Sus atributos son:

atributo	uso
content	Establece el contenido al que se refieren los atributos name y http-equiv (que no se pueden utilizar a la vez)
name	Permite indicar información sobre la página que será tenida en cuenta por los programas que analicen el código de la misma. Posibles valores: <ul style="list-style-type: none">• keywords. Permite indicar una lista de palabras clave separadas por comas para que se tengan en cuenta por los robots de los buscadores y así establecer palabras y términos relacionados con la página que faciliten que la página sea encontrado por los usuarios al usar los buscadores• author. Indica el nombre del autor de la página web• description. Descripción de la página• generator. Software utilizado para crear la página
http-equiv	Establece una cabecera http para diversos fines. Contenidos habituales: <ul style="list-style-type: none">• content-type. Permite establecer el tipo de documento y el lenguaje web con el que está creado.• refresh. Hace que la página web se recarga tras el tiempo especificado en el atributo content. Se puede incluso indicar una URL a la que el navegador se dirigirá tras ese tiempo.• expires. Fecha en la que expira la página• date. Fecha en la que se creó la página• last-modified. Fecha de última modificación de la página• default-style. URL a la hoja de estilos alternativa del documento. EL protocolo http envía estas cabeceras, pero si las usamos en nuestra página, en teoría tienen prioridad sobre lo que envíe el http.
charset	Disponible desde HTML 5, sirve sólo para indicar la forma en la que está codificado el texto, en sustitución de la versión larga mediante http-equiv="content-Type" de HTML 4 y XHTML.

Ejemplos de uso con el atributo **name**:

```
<meta name="author" content="Jorge Sanchez" />  
<meta name="generator" content="Sublime Text 2" />  
<meta name="description" content="Página sobre Palencia" />
```

```
<meta name="keywords" content="Palencia, Castilla, monumentos, ciudad, castellana, Tierra de Campos, Cerrato" />
```

Con name todos los atributos son descriptivos de la página. Sin embargo, el elemento **meta** puede realizar indicaciones http con ayuda de su atributo **http-equiv**. Ejemplos:

```
<!-- Codificación Unicode-->
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

Este es quizá el meta más utilizado de las páginas web escritas en HTML anterior a la versión 5. HTML 5 ha simplificado esta etiqueta:

```
<meta charset="UTF-8" />
```

Que es mucho más fácil de recordar y escribir. Otros usos de meta con **http-equiv** son:

```
<meta http-equiv="Refresh" content="3" />
<!--La página se recarga cada tres segundos-->
<meta http-equiv="Refresh" content="3;URL=http://www.google.es" />
<!--A los tres segundos se carga la página google-->
<meta http-equiv="Expires" content="Fri, 21 Dec 2012 23:59:59 GMT" />
<!--La página caduca el 21/12/2012 a las 23:59:59 horario de Greenwich-->
```

(3.16) etiquetas semánticas de HTML5

La idea en los nuevos estándares de HTML (especialmente en HTML5) es que los elementos HTML más que para formato, sirvan para dar valor semántico al contenido. Es decir, para indicar qué tipo de contenido es. Eso vale para casi todos los elementos ya conocidos como **p** (párrafo), **abbr** (abreviatura), **strong** (remarcado fuerte, negrita), etc.

La idea es marcar la semántica con HTML y el formato con las hojas de estilo CSS que aprenderemos en la siguiente unidad.

A este respecto hay una serie de elementos HTML aparecidos con HTML 5 que no dan ningún formato al texto pero permiten remarcarle dándole un significado y además permitiendo posteriormente que ese contenido tenga un formato especial mediante CSS.

Todos estos elementos no son comprendidos excepto por navegadores compatibles con HTML 5.

(3.16.1) elemento header

Permite marcar una cabecera. Esto no tiene que ver con el elemento **head**, se trata de una serie párrafos que se marcan de esta forma para indicar que pertenecen a la cabecera de la página.

Realmente una página puede tener varios elementos **header**, al nivel de la etiqueta **body** indica que su contenido es la cabecera de la página completa (por ejemplo donde se muestra el título general, logotipo,...). Pero dentro de una etiqueta, por ejemplo, **article** indicaría que su contenido es la cabecera del artículo.

No se puede meter un header dentro de otro header ni dentro del elemento **foot** o **address**.

(3.16.2) elemento footer

Similar al anterior, sirve para marcar el pie de una página, sección, artículo etc. Dependiendo del contexto en el que se coloque servirá para unas cosas u otras. En el caso del elemento footer que se coloque al nivel del elemento **body**, servirá para agrupar los elementos de pie de página que suelen ser información sobre el autor, copyright, términos de uso de la página, etc.

(3.16.3) elemento section

Es un elemento que permite dividir en diferentes partes o secciones un documento. Ejemplo de página dividida en secciones:

```
<!DOCTYPE html>
<html lang="es-ES">
  <head>
    <meta charset="UTF-8">
    <title></title>
  </head>
  <body>
    <header>
      <h1>Historia de HTML</h1>
      <p>Desde 1989 hasta nuestros días</p>
    </header>
    <section>
      <h2>Inicios en HTML. Tim Bernes Lee</h2>
      <p>.....</p>
    </section>
    <section>
      <h2>Creación de la web. Primeros navegadores</h2>
      <p>.....</p>
    </section>
    <section>
      <h2>La guerra de los navegadores</h2>
      <p>.....</p>
    </section>
    <section>
      <h2>De HTML 4 a HTML 5 pasando por XHTML</h2>
      <p>...</p>
    </section>
    <footer>
      <p>Realizado por Jorge Sánchez</p>
    </footer>
  </body>
</html>
```

(3.16.4) elemento nav

Se trata de un elemento que marca a su contenido como una sección de enlaces, es decir una barra de navegación. Más adelante con CSS se puede dar un formato especial a dichos enlaces. **nav** se puede escribir dentro de cualquier elemento HTML de sección (como **section**, **article**, **header**, **footer**,..).

Los elementos marcados por **nav** pueden ser omitidos por los lectores digitales de páginas web que utilizan, por ejemplo, las personas invidentes. Lo cual facilita la comprensión del texto) y así que dicho contenido quede marcado sólo para utilizar los enlaces interiores a **nav**.

(3.16.5) elemento article

Si se observan los elementos descritos anteriormente parece claro que HTML 5 utiliza como metáfora la forma de distribuir contenidos de un periódico. Así hay cabeceras, pies, secciones y, con este elemento, artículos.

La idea es colocar dentro de este elemento (que tiene sentido dentro del elemento **section**, o incluso aparecer de forma independiente) es colocar contenido que pueda ser entendido como un **todo** que describa un tema de forma íntegra.

Nuevamente el formato mediante CSS permitiría que apareciera con el estilo que deseemos.

(3.16.6) elemento hgroup

Permite agrupar varios títulos (elementos **h1** a **h6**) dentro de esta etiqueta para darles estilo común. Se utiliza dentro de **article** para marcar su zona de títulos.

(3.16.7) elemento figure

Sirve para agrupar los elementos relativos a una imagen como la propia imagen (elemento **img**), el título de la misma, el pie, los párrafos relativos, etc.

(3.16.8) elemento figcaption

Permite indicar el título de una imagen, dentro de un elemento figure. Ejemplo de uso:

```
<article>
  <hgroup>
    <h2>Palencia</h2>
    <h3>Paisajes</h3>
  </hgroup>
  <figure>
    
    <figcaption>
      Vista de la Montaña Palentina desde Camporredondo
    </figcaption>
  </figure>
```



```
<p>  
  Los paisajes en la provincia de Palencia.....  
</p>  
...  
</article>
```

(3.16.9) elemento aside

Permite marcar texto dentro de un artículo para que no se tenga en cuenta como parte del texto del artículo, sino como un texto aparte que permite realizar aclaraciones al artículo, referencias, resúmenes remarcados y sobre todo cuadros de texto de estilo periodístico para destacar partes del artículo. Para que este texto aparezca de manera especial, debe dársele formato con CSS.

(3.17) elementos multimedia

Nuevamente en HTML 5 ha habido una preocupación sobre cómo utilizar elementos multimedia en una página web. Hasta la aceptación de HTML 5 ha sido problemática la forma de añadir elementos multimedia.

El uso de **Flash** simplificó esta posibilidad dada la potencia de esta tecnología, pero dificultaba el aprendizaje para realizar páginas y exigía utilizar una forma de trabajar ajena a HTML y que además dependía de una empresa en concreto (**Adobe**). Actualmente la idea de muchos creadores es ir retirando Flash y aprovechar las nuevas capacidades de HTML 5 para la multimedia.

En especial HTML 5 avanza enormemente en cuestiones relacionadas con el vídeo (ya se ha tratado anteriormente en estos apuntes como utiliza HTML 5 las imágenes).

(3.17.1) el problema del vídeo y el audio

EL problema es que hay numerosos formatos de vídeo y audio y que cada navegador tiene capacidades distintas al respecto.

Por ello muchos usuarios conocedores de estos problemas saben que en los ordenadores hay que instalar plugins (extensiones) para poder ver vídeos u oír música.

De esta forma si incorporamos vídeo a nuestra página a través de un enlace normal:

```
<a href="video1.mpeg">Ver vídeo</a>
```

El navegador buscará si disponemos del plugin apropiado para ver el vídeo, si no es así simplemente lo descargará en nuestro ordenador al no poder mostrar él el contenido.

(3.17.2) etiqueta embed

Se trata de una etiqueta ya veterana que se utiliza para colocar en una página web elementos no pertenecientes al lenguaje HTML como animaciones Flash, vídeo, audio, etc.

Usa los siguientes atributos:

atributo	uso
src	URL al recurso que se desea mostrar
type	Tipo MIME que indica el contenido del recurso que se incorpora con la etiqueta
height	Altura de la ventana que mostrará el recurso
width	Anchura de la ventana que mostrará el recurso

(3.17.3) etiqueta object

Esta etiqueta está orientada a sustituir a la anterior y permite incorporar cualquier tipo de contenido a una página web. Los atributos posibles son:

atributo	uso
data	URL al recurso que se desea mostrar
type	Tipo MIME que indica el contenido del recurso que se incorpora con la etiqueta.
height	Altura de la ventana que mostrará el recurso
width	Anchura de la ventana que mostrará el recurso
usemap	Permite indicar el nombre de un mapa de imágenes (usando <i>#nombre</i>) que actuará sobre el objeto.
name	Permite indicar el nombre del objeto
form	Indica el nombre del formulario al que pertenece este objeto

(3.17.4) elemento param

Elemento interior a **object**, que permite especificar parámetros a los plugins de los navegadores encargados de mostrar el objeto. A través de **param** pasamos instrucciones al plugin que reproduzca el objeto. La forma es mediante los atributos **name** (en el que se indica el nombre del atributo) y **value** (valor del atributo)

Ejemplo:

```
<object data="Wildlife.wmv" type="video/x-ms-wmv" width="500" height="300">  
  <param name="autoplay" value="true" />  
</object>
```

En el ejemplo el vídeo (en caso de que el navegador pueda reproducirle), se ejecutará automáticamente al cargar la página.

(3.17.5) elemento **iframe**

Es un elemento que había desaparecido en el estándar pero que HTML 5 ha recuperado. La idea original es colocar un documento dentro de otro documento, es decir sirve para incrustar contenido de una dirección dentro de la página que le hace referencia. Su popularidad actual se debe a **youtube** ya que es el elemento que sugiere la página para colocar vídeos youtube en nuestras páginas.

Atributos:

atributo	uso
src	URL al recurso que se desea mostrar
width	Anchura del objeto en nuestra página
height	Altura del objeto en nuestra página

(3.17.6) elemento **video**

Es un elemento que sólo funciona en HTML 5 y es el método propuesto para incorporar vídeos a una página web.

atributo	uso
src	URL al vídeo que se desea mostrar
width	Anchura del vídeo en nuestra página
height	Altura del vídeo en nuestra página
autoplay	Usa el valor fijo autoplay para indicar que el vídeo se inicia automáticamente en cuanto se descargue
loop	Usa el valor fijo loop para indicar que el vídeo se ejecuta automáticamente una y otra vez
controls	Con valor controls indica que el vídeo mostrará los controles de reproducción (pausa, play,...)
preload	Recomendación sobre cómo debemos realizar la descarga. Posibilidades: <ul style="list-style-type: none">• auto. El vídeo se descarga en cuanto se carga la página• none. El vídeo no se descarga. Cuando el usuario pulse play, se descargará.• metadata. Descarga los metadatos del vídeo, no el vídeo en sí
poster	Permite indicar la dirección URL a una imagen que se mostrará mientras el vídeo no se está reproduciendo. Si no se usa este atributo, se usa el primer fotograma del vídeo como póster.

codecs de vídeo. elemento **source**

Se denomina códec (de codificar, descodificar), al software que se usa para crear los vídeos. Un vídeo actual usa un formato comprimido de datos, por lo que para crearlos se codifican los datos y para mostrarle, se descodifica.

Eso significa que para ver un vídeo en un formato concreto, se necesita el códec apropiado. Para más inri, cada navegador incorpora unos codecs. HTML 5 ha conseguido subsanar la situación, mediante la etiqueta **source**.

La idea es que el vídeo se codifica en varios formatos distintos y luego les hacemos referencia dentro de la etiqueta video y el navegador usará el formato que sea capaz de traducir (del que disponga de codecs).

El elemento **source** fundamentalmente dispone de dos atributos: **src** para indicar la URL que nos lleva al vídeo y **type** que indica el tipo MIME del vídeo. Además dentro del tipo MIME se pueden indicar los codecs concretos que necesitamos para decodificar el vídeo.

La idea es la siguiente:

```
<video autoplay="autoplay" controls="controls" poster="foto1.jpg" >
  <source src="video.mp4" type="video/mp4;codecs='avc1.42E01E, mp40a.40.2' " />
  <source src="video.ogv" type="video/ogg;codecs='theora, vorbis' " />
  El navegador no puede mostrar el vídeo
</video>
```

Dentro de **type**, el uso de codecs es opcional, ya que si el navegador no reconoce el formato no suele hacer caso a los **codecs** que se indiquen (aunque a veces les descarga). La idea es que si el primer formato no se reconoce (primer elemento **source**), se intenta el segundo y así sucesivamente. Si ninguno es reproducible por el navegador actual, éste mostrará la frase final tras el último **source**.

subtítulos

Aunque los navegadores actuales no permiten esta posibilidad (se consigue a través de JavaScript) de hecho sí hay una etiqueta reciente que se coloca dentro del elemento **video** que sirve para indicar subtítulos. Se trata de **track**. Sus atributos son:

atributo	uso
src	URL al archivo VTT o SRT que contiene los subtítulos
kind	Tipo de subtítulos. Posibilidades: <ul style="list-style-type: none"> • subtitles. Subtítulos normales • captions. Igual que el anterior pero con posibilidad de añadir sonidos. • descriptions. Texto alternativo e independiente de la película • chapters. Contiene los capítulos que permiten navegar de forma ágil por la película • metadata. Metadatos de la película
label	Nombre de los subtítulos para reconocerlos en caso de haber indicado varios
srclang	Idioma de los subtítulos (por ejemplo es para español)

Ejemplo:

```
<video autoplay="autoplay" controls="controls" poster="foto1.jpg" >
  <source src="video.mp4" type="video/mp4;codecs='avc1.42E01E, mp40a.40.2' " />
  <source src="video.ogv" type="video/ogg;codecs='theora, vorbis' " />
  <source src="spa.vtt" kind="subtitles" srclang="es" label="Spanish" />
  <source src="deu.vtt" kind="subtitles" srclang="de" label="German" />
  El navegador no puede mostrar el vídeo
</video>
```

(3.17.7) elemento audio

Se trata de un elemento comparable al video, pero en este caso para incorporar audio a la página.

atributo	uso
src	URL al vídeo que se desea mostrar
autoplay	Usa el valor fijo <i>autoplay</i> para indicar que el vídeo se inicia automáticamente en cuanto se descargue
loop	Usa el valor fijo <i>loop</i> para indicar que el vídeo se ejecuta automáticamente una y otra vez
controls	Con valor <i>controls</i> indica que el vídeo mostrará los controles de reproducción (pausa, play,...)
preload	Recomendación sobre cómo debemos realizar la descarga. Posibilidades: <ul style="list-style-type: none">• auto. El vídeo se descarga en cuanto se carga la página• none. El vídeo no se descarga. Cuando el usuario pulse play, se descargará.• metadata. Descarga los metadatos del vídeo, no el vídeo en sí

Con el audio hay el mismo problema con la cuestión de los codecs y los formatos, por lo que también es habitual convertir el audio a distintos formatos y dar opciones dentro de la etiqueta audio gracias a la etiqueta **source**. Ejemplo:

```
<audio controls="controls" autoplay="autoplay">  
  <source src="audio1.ogg" type="audio/ogg">  
  <source src="audio2.mp3" type="audio/mpeg">  
  No se puede reproducir el archivo de audio  
</audio>
```