

CURSO DE HTML4/5



HTML



POR
IVÁN RODRÍGUEZ RUIZ

ÍNDICE

1 INTRODUCCIÓN

- 1.1 ¿Que es HTML?
- 1.2 Historia de HTML
- 1.3 Especificación Oficial
- 1.4 HTML y XHTML
- 1.5 HTML y CSS

2 CARACTERÍSTICAS BÁSICAS

- 2.1 Lenguaje de Etiquetas
- 2.2 El Primer documento HTML
- 2.3 Etiquetas y Atributos
- 2.4 Elementos HTML
- 2.5 Sintaxis de las Etiquetas XHTML

3 TEXTO

- 3.1 Estructura
 - 3.1.1 Párrafos
 - 3.1.2 Secciones
- 3.2 Marcado Básico de Texto
 - 3.2.1 Etiquetas Básicas
 - 3.2.2 Otras etiquetas
- 3.3 Marcado Avanzado de Texto
- 3.4 Marcado Genérico de Texto
- 3.5 Espacios en blanco y nuevas líneas
 - 3.5.1 Nuevas líneas
 - 3.5.2 Espacios en blanco
 - 3.5.3 Texto Preformatado
- 3.6 Códificación de Caracteres

4 ENLACES

- 4.1 URL
- 4.2 Enlaces Relativos y Absolutos
- 4.3 Enlaces Básicos
- 4.4 Enlaces Avanzados
 - 4.4.1 Idioma del enlace (hreflang)
 - 4.4.2 Tipo de contenido (type)
 - 4.4.3 Tipo de relación (rel y rev)
 - 4.4.4 Codificación de caracteres (charset)
- 4.5 Otros tipos de Enlaces
- 4.6 Ejemplos de enlaces habituales

5 LISTAS

- 5.1 Listas No Ordenadas
- 5.2 Listas Ordenadas
- 5.3 Listas de Definición

6 IMÁGENES Y OBJETOS

- 6.1 Imágenes
- 6.2 Mapas de Imagen
- 6.3 Objetos

7 TABLAS

- 7.1 Tablas Básicas
- 7.2 Tablas Avanzadas

8 FORMULARIOS

- 8.1 Formularios Básicos
- 8.2 Elementos de Formulario
- 8.3 Formularios Avanzados
- 8.4 Otros elementos de Formulario

9 CAPAS Y METAINFORMACIÓN

- 9.1 Layouts o Capas
- 9.2 Estructura de la Cabecera
- 9.3 Metadatos
- 9.4 DOCTYPE

10 MAS ETIQUETAS

- 10.1 Comentarios
- 10.2 JavaScript
- 10.3 CSS
- 10.4 Iframes
- 10.5 Otras Etiquetas

11 ACCESIBILIDAD Y VALIDACIÓN

- 11.1 Requisitos del Nivel A de Accesibilidad
- 11.2 Validadores W3C
- 11.3 Otros Validadores

12 MARCADO SEMÁNTICO EN HTML5

- 12.1 Estructura básica
- 12.2 Estructura del cuerpo
- 12.3 Dentro del cuerpo
- 12.4 Nuevos Elementos
- 12.5 Ejemplo de Marcado Semántico

13 FORMULARIOS CON HTML5

- 13.1 Formularios Web
 - 13.1.1 El elemento <form>
 - 13.1.2 El elemento <input>
 - 13.1.3 Tipo email
 - 13.1.4 Tipo search
 - 13.1.5 Tipo url
 - 13.1.6 Tipo tel.
 - 13.1.7 Tipo number
 - 13.1.8 Tipo Range.
 - 13.1.9 Tipo date
 - 13.1.10 Tipo week.
 - 13.1.11 Tipo month.
 - 13.1.12 Tipo time
 - 13.1.13 Tipo datetime.
 - 13.1.14 Tipo datetime-local
 - 13.1.15 Tipo Color

- 13.2 Nuevos atributos.
 - 13.2.1 Atributo placeholder
 - 13.2.2 Atributo required
 - 13.2.3 Atributo multiple
 - 13.2.4 Atributo autofocus
 - 13.2.5 Atributo form
 - 13.2.6 Atributo pattern
- 13.3 Nuevos elementos para formularios
 - 13.3.1 El elemento <datalist>
 - 13.3.2 El elemento <progress>
 - 13.3.3 El elemento <meter>
 - 13.3.4 El elemento <output>

14 VÍDEO Y AUDIO EN HTML5

- 14.1 Reproduciendo vídeo con HTML5
 - 14.1.1 El elemento <video>
 - 14.1.2 Atributos para <video>
- 14.2 Formatos de video
- 14.3 Reproduciendo audio con HTML5
 - 14.3.1 El elemento <audio>

15 BIBLIOGRAFÍA Y ENLACES

- 15.1 Bibliografía recomendada
- 15.2 Enlaces de Interés

1 INTRODUCCIÓN

1.1 ¿Qué es HTML?

HTML, siglas de HyperText Markup Language («lenguaje de marcado de hipertexto»), hace referencia al lenguaje de marcado predominante para la elaboración de páginas web que se utiliza para describir y traducir la estructura y la información en forma de texto, así como para complementar el texto con objetos tales como imágenes. El HTML se escribe en forma de «etiquetas», rodeadas por corchetes angulares (<,>). HTML también puede describir, hasta un cierto punto, la apariencia de un documento (con CSS), y puede incluir un script (por ejemplo JavaScript), el cual puede afectar el comportamiento de navegadores web y otros procesadores de HTML.

1.2 Historia de HTML

La primera descripción de HTML disponible públicamente fue un documento llamado HTML Tags (Etiquetas HTML), publicado por primera vez en Internet por Tim Berners-Lee en 1991. Describe 22 elementos que incluyen el diseño inicial y relativamente simple de HTML. Trece de estos elementos todavía existen en HTML 4.

Berners-Lee consideraba a HTML una ampliación de SGML (Standard Generalized Markup Language o "Estándar de Lenguaje de Marcado Generalizado"), pero no fue formalmente reconocida como tal hasta la publicación de mediados de 1993, por la IETF (Internet Engineering Task Force o Fuerza de Tareas de Ingeniería de Internet), de una primera proposición para una especificación de HTML: el boceto Hypertext Markup Language de Berners-Lee y Dan Connolly, el cual incluía una Definición de Tipo de Documento SGML para definir la gramática.

El boceto expiró luego de seis meses, pero fue notable por su reconocimiento de la etiqueta propia del navegador Mosaic usada para insertar imágenes sin cambio de línea, que reflejaba la filosofía del IETF de basar estándares en prototipos con éxito. De la misma manera, el boceto competidor de Dave Raggett HTML+ (Hypertext Markup Format) (Formato de Marcaje de Hipertexto), de finales de 1993, sugería estandarizar características ya implementadas, como las tablas, imágenes y formularios.

A partir de 1996, los estándares de HTML los publica otro organismo de estandarización llamado W3C (<http://www.w3.org/>) (World Wide Web Consortium). La versión HTML 3.2 se publicó el 14 de Enero de 1997 y es la primera recomendación de HTML publicada por el W3C. Esta revisión incorpora los últimos avances de las páginas web desarrolladas hasta 1996, como applets de Java y texto que fluye alrededor de las imágenes.

HTML 4.0 se publicó el 24 de Abril de 1998 (siendo una versión corregida de la publicación original del 18 de Diciembre de 1997) y supone un gran salto desde las versiones anteriores. Entre sus novedades más destacadas se encuentran las hojas de estilos CSS, la posibilidad de incluir pequeños programas o scripts en las páginas web, mejora de la accesibilidad de las páginas diseñadas, tablas complejas y mejoras en los formularios.

La última especificación oficial de HTML se publicó el 24 de diciembre de 1999 y se denomina HTML 4.01. Se trata de una revisión y actualización de la versión HTML 4.0, por lo que no incluye novedades significativas.

Desde la publicación de HTML 4.01, la actividad de estandarización de HTML se detuvo y el W3C se centró en el desarrollo del estándar XHTML. Por este motivo, en el año 2004, las empresas Apple, Mozilla y Opera mostraron su preocupación por la falta de interés del W3C en HTML y decidieron organizarse en una nueva asociación llamada WHATWG (<http://www.whatwg.org/>) (Web Hypertext Application Technology Working Group).

La actividad actual del WHATWG se centra en el futuro estándar HTML 5, cuyo primer borrador oficial (<http://www.w3.org/TR/html5/>) se publicó el 22 de enero de 2008. Debido a la fuerza de las empresas que forman el grupo WHATWG y a la publicación de los borradores de HTML 5.0, en marzo de 2007 el W3C decidió retomar la actividad estandarizadora de HTML (<http://www.w3.org/2007/03/html-pressrelease>).

De forma paralela a su actividad con HTML, W3C ha continuado con la estandarización de XHTML, una versión avanzada de HTML y basada en XML. La primera versión de XHTML se denomina XHTML 1.0 y se publicó el 26 de Enero de 2000 (y posteriormente se revisó el 1 de Agosto de 2002).

1.3 Especificación Oficial

El organismo W3C (<http://www.w3.org/>) (World Wide Web Consortium) elabora las normas que deben seguir los diseñadores de páginas web para crear las páginas HTML. Las normas oficiales están escritas en inglés y se pueden consultar de forma gratuita en las siguientes direcciones:

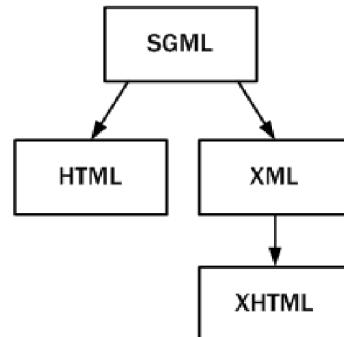
- Especificación oficial de HTML 4.01 (<http://www.w3.org/TR/html401>)
- Especificación oficial de XHTML 1.0 (<http://www.w3.org/TR/xhtml1>)

El estándar XHTML 1.0 incluye el 95% del estándar HTML 4.01, ya que sólo añade pequeñas mejoras y modificaciones menores. Afortunadamente, no es necesario leer las especificaciones recomendaciones oficiales de HTML para aprender a diseñar páginas con HTML o XHTML. Las normas oficiales están escritas con un lenguaje bastante formal y algunas secciones son difíciles de comprender. Por ello, en los próximos capítulos se explica de forma sencilla y con decenas de ejemplos la especificación oficial de XHTML.

1.4 HTML y XHTML

El lenguaje XHTML es muy similar al lenguaje HTML. De hecho, XHTML no es más que una adaptación de HTML al lenguaje XML. Técnicamente, HTML es descendiente directo del lenguaje SGML, mientras que XHTML lo es del XML (que a su vez, también es descendiente de SGML).

Actualmente, entre HTML 4.01 y XHTML 1.0, la mayoría de diseñadores escogen XHTML.



Esos sí, durante 2012 se ha visto incrementado la necesidad de usar, cada vez más, el estándar HTML5 que finalmente se ha impuesto al XHTML2 con el que trabajaba el W3C.

Fuente: <http://www.htmlcinco.com/xhtml-2-detenido-html-5-toma-sus-recursos/>

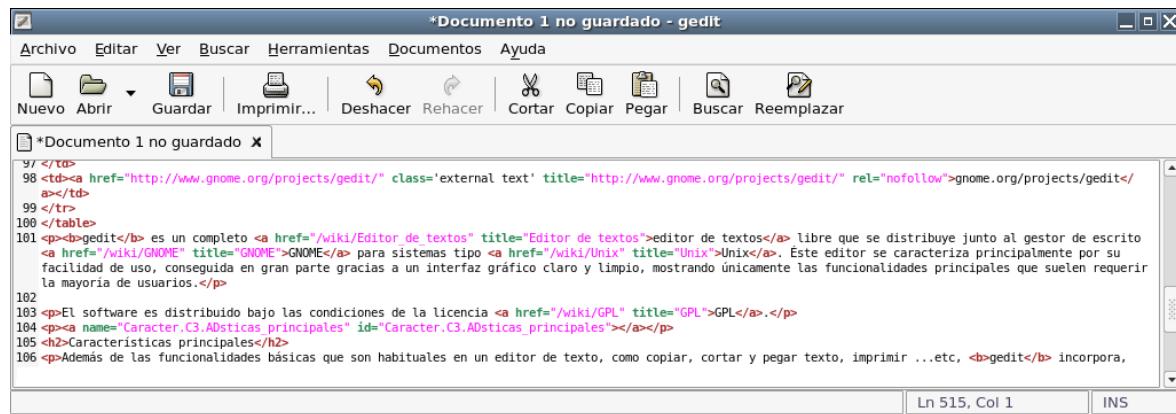
1.5 HTML y CSS

Originalmente, las páginas HTML sólo incluían información sobre sus contenidos de texto e imágenes. Con el desarrollo del estándar HTML, las páginas empezaron a incluir también información sobre el aspecto de sus contenidos: tipos de letra, colores y márgenes. La posterior aparición de tecnologías como JavaScript, provocaron que las páginas HTML también incluyeran el código de las aplicaciones (llamadas scripts) que se utilizan para crear páginas web dinámicas.

Incluir en una misma página HTML los contenidos, el diseño y la programación complica en exceso su mantenimiento. Normalmente, los contenidos y el diseño de la página web son responsabilidad de diferentes personas, por lo que es conveniente separarlos. CSS es el mecanismo que permite separar los contenidos definidos mediante XHTML y el aspecto que deben presentar esos contenidos. Esto lo veremos más adelante.

1.6 Editores

En principio, usaremos el mismo gedit que trae Ubuntu / Xubuntu.
En la captura inferior podemos ver la interfaz en español sobre Xubuntu.



Como alternativa, tenemos infinidad de Editores, desde los mas sencillos, como el propio Gedit, hasta los mas avanzados, como Eclipse.

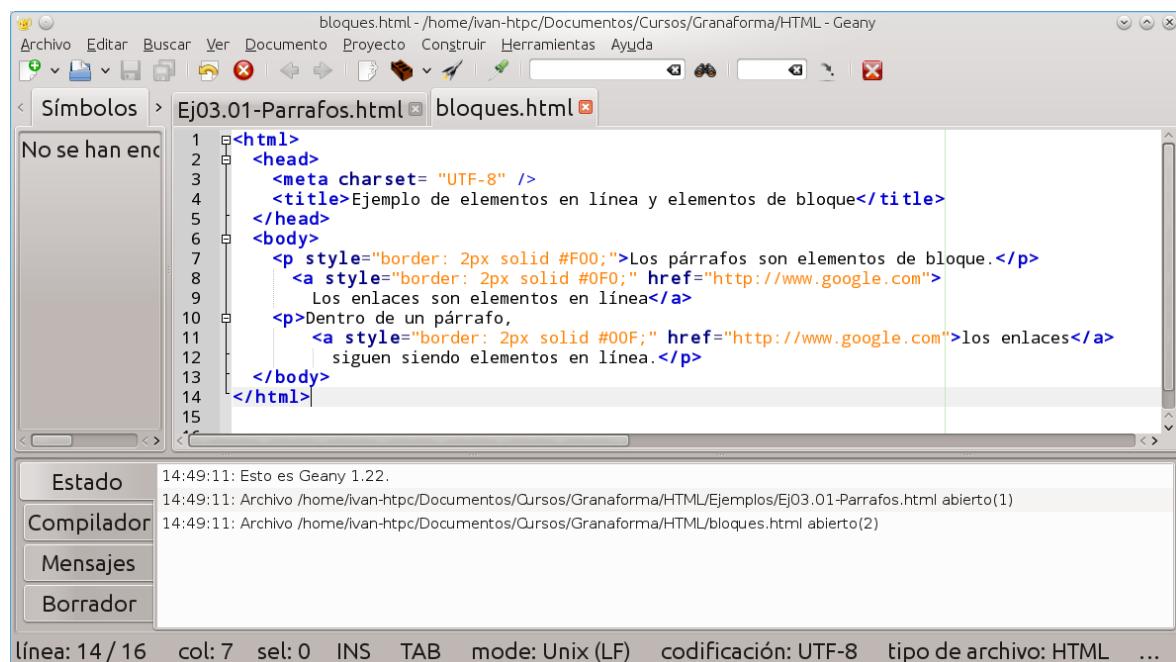
Una alternativa a Gedit, muy avanzada y excelente, es Geany, que se instala desde los repositorios oficiales de Ubuntu a través de la terminal. Para hacerlo haremos lo siguiente:

1. Abrimos el Terminal.

2. Escribimos los siguientes comandos:

```
sudo add-apt-repository ppa:geany-dev/ppa  
sudo apt-get update  
sudo apt-get install geany
```

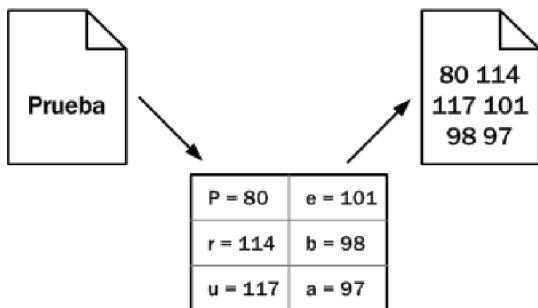
El resultado será el siguiente (versión 1.22 sobre Kubuntu):



2 CARACTERÍSTICAS BÁSICAS

2.1 Lenguaje de Etiquetas

Uno de los retos iniciales a los que se tuvo que enfrentar la informática fue el de cómo almacenar la información en los archivos digitales.



Como los primeros archivos sólo contenían texto sin formato, la solución utilizada era muy sencilla: se codificaban las letras del alfabeto y se transformaban en números.

De esta forma, para almacenar un contenido de texto en un archivo electrónico, se utiliza una tabla de conversión que transforma cada carácter en un número. Una vez almacenada la secuencia de números, el contenido del archivo se puede recuperar realizando el proceso inverso.

El proceso de transformación de caracteres en secuencias de números se denomina codificación de caracteres y cada una de las tablas que se han definido para realizar la transformación se conocen con el nombre de páginas de código. Una de las codificaciones más conocidas (y una de las primeras que se publicaron) es la codificación ASCII. La importancia de las codificaciones en HTML se verá más adelante.

Una vez resuelto el problema de almacenar el texto simple, se presenta el reto de almacenar los contenidos de texto con formato. En otras palabras, ¿cómo se almacena un texto en negrita? ¿y un texto de color rojo? ¿y otro texto azul, en negrita y subrayado?

Utilizar una tabla de conversión similar a las que se utilizan para textos sin formato no es posible, ya que existen infinitos posibles estilos para aplicar al texto.

Una solución técnicamente viable consiste en almacenar la información sobre el formato del texto en una zona especial reservada dentro del propio archivo. En esta zona se podría indicar dónde comienza y dónde termina cada formato.

No obstante, la solución que realmente se emplea para guardar la información con formato es mucho más sencilla: el archivo electrónico almacena tanto los contenidos como la información sobre el formato de esos contenidos. Si por ejemplo se quiere dividir el texto en párrafos y se desea dar especial importancia a algunas palabras, se podría indicar de la siguiente manera:

```
<párrafo>
Contenido de texto con <importante>algunas palabras</importante>
resaltadas de forma
especial.
</párrafo>
```

El principio de un párrafo se indica mediante la palabra `<párrafo>` y el final de un párrafo se indica mediante la palabra `</párrafo>`. De la misma manera, para asignar más importancia a ciertas palabras del texto, se encierran entre `<importante>` y `</importante>`.

El proceso de indicar las diferentes partes que componen la información se denomina marcar (markup en inglés). Cada una de las palabras que se emplean para marcar el inicio y el final de una sección se denominan etiquetas.

Aunque existen algunas excepciones, en general las etiquetas se indican por pares y se forman de la siguiente manera:

- Etiqueta de apertura: carácter <, seguido del nombre de la etiqueta (sin espacios en blanco) y terminado con el carácter >
- Etiqueta de cierre: carácter <, seguido del carácter /, seguido del nombre de la etiqueta (sin espacios en blanco) y terminado con el carácter >

Los caracteres <> también se llaman paréntesis angulares, corchetes angulares, "cuñas" o "corchángulos".

Así, la estructura típica de las etiquetas HTML es:

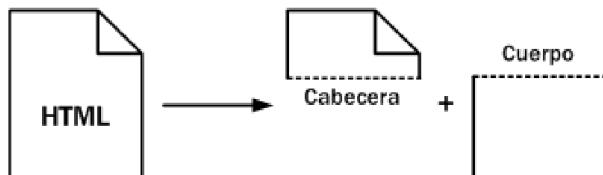
<nombre_etiqueta> ... </nombre_etiqueta>

HTML es un lenguaje de etiquetas (también llamado lenguaje de marcado) y las páginas web habituales están formadas por cientos o miles de pares de etiquetas. De hecho, las letras "ML" de la sigla HTML significan "markup language", que es como se denominan en inglés a los lenguajes de marcado. Además de HTML, existen muchos otros lenguajes de etiquetas como XML, SGML, DocBook y MathML.

La principal ventaja de los lenguajes de etiquetas es que son muy sencillos de leer y escribir por parte de las personas y de los sistemas electrónicos. La principal desventaja es que pueden aumentar mucho el tamaño del documento, por lo que en general se utilizan etiquetas con nombres muy cortos.

2.2 El Primer documento HTML

Las páginas HTML se dividen en dos partes: la cabecera y el cuerpo. La cabecera incluye información sobre la propia página, como por ejemplo su título y su idioma. El cuerpo de la página incluye todos sus contenidos, como párrafos de texto e imágenes.



El cuerpo (llamado `body` en inglés) contiene todo lo que el usuario ve en su pantalla y la cabecera (llamada `head` en inglés) contiene todo lo que no se ve (con la única excepción del título de la página, que los navegadores muestran como título de sus ventanas).

Para empezar creamos el primer código que se llamará [Ej02.02-holamundo.html](#).

```
<html>
  <head>
    <title> Titulo de la Página </title>
    <meta charset= "UTF-8" />
  </head>
  <body>
    <p> ;¡Hola Mundo!! </p>
  </body>
</html>
```

Si quieres probar este primer ejemplo, debes hacer lo siguiente:

1. Abre un editor de archivos de texto y crea un archivo nuevo
2. Copia el código HTML mostrado anteriormente y pégalo tal cual en el archivo creado
3. Guarda el archivo con el nombre `holamundo` junto con la extensión `.html`

Para que el ejemplo anterior funcione correctamente, es imprescindible que utilices un editor de texto sin formato. Si tu sistema operativo es Windows, puedes utilizar el Bloc de notas (`notepad`), Wordpad, EmEditor, UltraEdit, Notepad++, etc. pero no puedes utilizar un procesador de textos como Word o Open Office. Si utilizas sistemas operativos tipo Linux, puedes utilizar editores como Gedit, Kedit, Kate e incluso Vi, pero no utilices KOffice ni Open Office.

Luego lo abrimos directamente en cualquier navegador (aunque yo te recomiendo **Firefox** o **Chrome**). Si ya estás viendo tu primera página HTML en el navegador, prueba a pulsar sobre el menú contextual la opción Ver Código fuente de la página (en Firefox / Chrome) o y podrás ver el código HTML de la página que está cargada en el navegador. Prueba a ver el código HTML de tu página preferida y verás cuantas etiquetas puede llegar a tener una página compleja.

Volviendo al código HTML del primer ejemplo, es importante conocer las tres etiquetas principales de un documento HTML (`<html>`, `<head>`, `<body>`).

- `<html>`: indica el comienzo y el final de un documento HTML. Ninguna etiqueta o contenido puede colocarse antes o después de la etiqueta `<html>` (salvo alguna excepción que veremos). En el interior de la etiqueta `<html>` se definen la cabecera y el cuerpo del documento HTML y todo lo que se coloque fuera de la etiqueta `<html>` se ignora.
- `<head>`: delimita la parte de la cabecera del documento. La cabecera contiene información sobre el propio documento HTML, como por ejemplo su título y el idioma de la página. Los contenidos indicados en la cabecera no son visibles para el usuario, con la excepción de la etiqueta `<title>`, que se utiliza para indicar el título del documento y que los navegadores lo visualizan en la parte superior izquierda de la ventana.
- `<body>`: delimita el cuerpo del documento HTML. El cuerpo encierra todos los contenidos que se muestran al usuario (párrafos de texto, imágenes, tablas). En general, el `<body>` de un documento contiene cientos de etiquetas HTML, mientras que el `<head>` no contiene más que unas pocas.

2.3 Etiquetas y Atributos

HTML define 91 etiquetas que los diseñadores pueden utilizar para marcar los diferentes elementos que componen una página:

```
a, abbr, acronym , address, applet, area, b, base, basefont, bdo, big, blockquote, body, br, button, caption , center, cite, code, col, colgroup, dd, del, dfn, dir, div, dl, dt, em, fieldset, font, form, frame , frameset, h1, h2, h3, h4, h5, h6, head, hr, html, i, iframe, img, input, ins, isindex, kbd, label , legend, li, link, map , menu, meta,noframes, noscript, object, ol, optgroup, option, p, param , pre, q, s , samp, script, select, small, span, strike, strong, style, sub, sup, table, tbody, td, textarea, tfoot, th, thead, title, tr, tt, u, ul, var.
```

De todas las etiquetas disponibles, las siguientes se consideran obsoletas y no se deben utilizar:
`applet`, `basefont`, `center`, `dir`, `font`, `isindex` , `menu`, `s`, `strike`, `u`.

A pesar de que se trata de un número de etiquetas muy grande, no es suficiente para crear páginas complejas. Algunos elementos como las imágenes y los enlaces requieren cierta información adicional para estar completamente definidos.

La etiqueta `<a>` por ejemplo se emplea para incluir un enlace en una página. Utilizando sólo la etiqueta `<a>` no es posible establecer la dirección a la que apunta cada enlace. Como no es viable crear una etiqueta por cada enlace diferente, la solución consiste en personalizar las etiquetas HTML mediante cierta información adicional llamada atributos.

De esta forma, se utiliza la misma etiqueta `<a>` para todos los enlaces de la página y se utilizan los atributos para indicar la dirección a la que apunta cada enlace ([Ej02.03-Etiquetas.html](#)):

```
<html>
  <head>
    <title>Ejemplo de atributos en las etiquetas</title>
    <meta charset= "UTF-8" />
  </head>

  <body>
    <p>
      Los enlaces son muy fáciles de indicar:
      <a>Soy un enlace incompleto; no tengo dirección de destino</a>.
      <a href="http://www.google.com">Este enlace apunta a Google</a>.
    </p>
  </body>
</html>
```

El primer enlace del ejemplo anterior no está completamente definido, ya que no apunta a ninguna dirección. El segundo enlace, utiliza la misma etiqueta `<a>`, pero añade información adicional mediante un atributo llamado `href`. Los atributos se incluyen dentro de la etiqueta de apertura. Por ahora no es importante comprender la etiqueta `<a>` ni el atributo `href`, ya que se explicarán con todo detalle más adelante.

No todos los atributos se pueden utilizar en todas las etiquetas. Por ello, cada etiqueta define su propia lista de atributos disponibles. Además, cada atributo también indica el tipo de valor que se le puede asignar. Si el valor de un atributo no es válido, el navegador ignora ese atributo. Aunque cada una de las etiquetas HTML define sus propios atributos, algunos de los atributos son comunes a muchas o casi todas las etiquetas.

De esta forma, es habitual explicar por separado los atributos comunes de las etiquetas para no tener que volver a hacerlo cada vez que se explica una nueva etiqueta. Los atributos comunes se dividen en cuatro grupos según su funcionalidad:

1. Atributos básicos: se pueden utilizar prácticamente en todas las etiquetas HTML

Atributo	Descripción
<code>id="texto"</code>	Establece un identificador único a cada elemento dentro de una página HTML (se usa tanto para CSS como, sobre todo, para JavaScript)
<code>class="texto"</code>	Establece la clase CSS que se aplica a los estilos del elemento
<code>style="texto"</code>	Establece de forma directa los estilos CSS de un elemento
<code>title="texto"</code>	Establece el título a un elemento (mejora la accesibilidad y los navegadores lo muestran cuando el usuario pasa el ratón por encima del elemento)

La mayoría de páginas web actuales utilizan los atributos `id` y `class` de forma masiva. Sin embargo, estos atributos sólo son realmente útiles cuando se trabaja con CSS y con Javascript.

Respecto al valor de los atributos `id` y `class`, sólo pueden contener guiones medios (-), guiones bajos (_), letras y/o números, pero no pueden empezar por números. Además, los navegadores distinguen mayúsculas de minúsculas y no se recomienda utilizar letras como ñ y acentos, ya que no es seguro que funcionen correctamente en todas las versiones de todos los navegadores.

2. Atributos para internacionalización: los utilizan las páginas que muestran sus contenidos en varios idiomas o aquellas que quieren indicar de forma explícita el idioma de sus contenidos:

Atributo	Descripción
lang="codigo de idioma"	Indica el idioma del elemento mediante un código predefinido
xml:lang="codigo de idioma"	Indica el idioma del elemento mediante un código predefinido
dir	Indica la dirección del texto (útil para los idiomas que escriben de derecha a izquierda)

En las páginas XHTML, el atributo xml:lang tiene más prioridad que lang y es obligatorio incluirlo siempre que se incluye el atributo lang.

Como la palabra internacionalización es muy larga, se suele sustituir por la abreviatura i18n (el número 18 se refiere al número de letras que existen entre la letra i y la letra n de la palabra internacionalización).

3. Atributos de eventos: sólo se utilizan en las páginas web dinámicas creadas con JavaScript.

Atributo	Descripción
onclick, ondblclick, onmousedown, onmouseup, onmouseover, onmousemove, onmouseout, onkeypress, onkeydown, onkeyup	Permiten controlar los eventos producidos sobre cada elemento de la página

Cada vez que el usuario pulsa una tecla, mueve su ratón o pulsa cualquier botón del ratón, se produce un evento dentro del navegador. Utilizando JavaScript y los atributos anteriores, es posible responder de forma adecuada a cada evento.

4. Atributos para los elementos que pueden obtener el foco:

Cuando el usuario selecciona un elemento de la interfaz de una aplicación, se dice que "el elemento tiene el foco del programa". Si por ejemplo un usuario pincha con su ratón sobre un cuadro de texto y comienza a escribir, ese cuadro de texto tiene el foco del programa, llamado "focus" en inglés. Si el usuario selecciona después otro elemento, el elemento original pierde el foco y el nuevo elemento es el que tiene el foco del programa.

Los elementos de las páginas web también pueden obtener el foco de la aplicación (en este caso, el foco del navegador) y HTML define algunos atributos específicos para controlar cómo se seleccionan los elementos.

Atributo	Descripción
accesskey = "letra"	Establece una tecla de acceso rápido a un elemento HTML
tabindex = "numero"	Establece la posición del elemento en el orden de tabulación de la página. Su valor debe estar comprendido entre 0 y 32.767
onfocus, onblur	Controlan los eventos JavaScript que se ejecutan cuando el elemento obtiene o pierde el foco

Cuando se pulsa repetidamente la tecla del tabulador sobre una página web, el navegador selecciona de forma alternativa todos los elementos de la página que se pueden seleccionar (principalmente los enlaces y los elementos de formulario). El atributo `tabindex` permite alterar el orden en el que se seleccionan los elementos, por lo que es muy útil cuando se quiere controlar de forma precisa cómo se seleccionan los campos de un formulario complejo.

Por su parte, el atributo `accesskey` permite establecer una tecla para acceder de forma rápida a cualquier elemento. Aunque la tecla de acceso rápido se establece mediante HTML, la combinación de teclas necesarias para activar ese acceso rápido depende del navegador. En el navegador **Internet Explorer** se pulsa la tecla **ALT + la tecla definida**; en el navegador **Firefox** se pulsa **Alt + Shift + la tecla definida**; en el navegador **Opera** se pulsa **Shift + Esc + la tecla definida**; en el navegador **Safari** se pulsa **Ctrl + la tecla definida**.

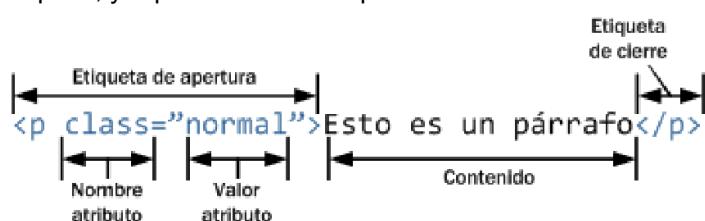
En el resto de la documentación, se emplearán las palabras "básicos", "i18n", "eventos" y "foco" respectivamente para referirse a cada uno de los grupos de atributos comunes definidos anteriormente.

2.4 Elementos HTML

Además de etiquetas y atributos, HTML define el término elemento para referirse a las partes que componen los documentos HTML.

Aunque en ocasiones se habla de forma indistinta de "elementos" y "etiquetas", en realidad un elemento HTML es mucho más que una etiqueta, ya que está formado por:

- Una etiqueta de apertura.
- Cero o más atributos.
- Texto encerrado por la etiqueta.
- Una etiqueta de cierre.



El texto encerrado por la etiqueta es opcional, ya que algunas etiquetas de

HTML no pueden encerrar ningún texto. El siguiente esquema muestra un elemento HTML, formado por una etiqueta `<p>`, atributos y contenidos de texto:

La estructura mostrada en el esquema anterior es un elemento HTML ya que comienza con una etiqueta de apertura (`<p>`), contiene cero o más atributos (`class="normal"`), dispone de un contenido de texto (Esto es un párrafo) y finaliza con una etiqueta de cierre (`</p>`).

Por tanto, si una página web tiene dos párrafos de texto, la página contiene dos elementos y cuatro etiquetas (dos etiquetas `<p>` de apertura y dos etiquetas `</p>` de cierre). De todas formas, aunque estrictamente no son lo mismo, es habitual intercambiar las palabras "elemento" y "etiqueta".

Por otra parte, el lenguaje HTML clasifica a todos los elementos en dos grupos: elementos en línea (inline elements en inglés) y elementos de bloque (block elements en inglés).

La principal diferencia entre los dos tipos de elementos es la forma en la que ocupan el espacio disponible en la página. Los elementos de bloque siempre empiezan en una nueva línea y ocupan todo el espacio disponible hasta el final de la línea, aunque sus contenidos no lleguen hasta el final de la línea. Por su parte, los elementos en línea sólo ocupan el espacio necesario para mostrar sus contenidos.

Si se considera el siguiente ejemplo ([Ej02.04-Elementos.html](#)):

```
<html>
  <head>
    <title>Ejemplo de elementos en línea y elementos de bloque</title>
    <meta charset= "UTF-8" />
  </head>
  <body>
    <p style="border: 2px solid #F00;">Los párrafos son elementos de bloque.</p>
    <a style="border: 2px solid #0F0;" href="http://www.google.com">
      Los enlaces son elementos en línea</a>
    <p>Dentro de un párrafo,
      <a style="border: 2px solid #00F;" href="http://www.google.com">los enlaces</a>
      siguen siendo elementos en línea.</p>
  </body>
</html>
```

Vemos cómo visualizan los párrafos en los navegadores el código HTML anterior (mediante CSS se han añadido bordes que muestran el espacio ocupado por cada elemento).

El primer párrafo contiene un texto corto que sólo ocupa la mitad de la anchura de la ventana del navegador. No obstante, el espacio reservado por el navegador para el primer párrafo llega hasta el final de esa línea, por lo que resulta evidente que los elementos `<p>` son elementos de bloque.

Por otra parte, el primer enlace del ejemplo anterior también tiene un texto corto que ocupa solamente la mitad de la anchura de la ventana del navegador. En este caso, el navegador sólo reserva para el enlace el sitio necesario para mostrar sus contenidos. Si se añade otro enlace en esa misma línea, se mostraría a continuación del primer enlace. Por tanto, los elementos `<a>` son elementos en línea.

Por último, el segundo párrafo sigue ocupando todo el espacio disponible hasta el final de cada línea (por ser un elemento de bloque) y el enlace que se encuentra dentro del párrafo sólo ocupa el sitio necesario para mostrar sus contenidos (por ser un elemento en línea).

La mayoría de elementos de bloque pueden contener en su interior elementos en línea y otros elementos de bloque. Los elementos en línea sólo pueden contener texto u otros elementos en línea. En otras palabras, un elemento de bloque no puede aparecer dentro de un elemento en línea. En cambio, un elemento en línea puede aparecer dentro de un elemento de bloque y dentro de otro elemento en línea.

Los elementos en línea definidos por HTML son: `a, abbr, acronym, b, basefont, bdo, big, br, cite, code, dfn, em, font, i, img, input, kbd, label, q, s, samp, select, small, span, strike, strong, sub, sup, textarea, tt, u, var`.

Los elementos de bloque definidos por HTML son: `address, blockquote, center, dir, div, dl, fieldset, form, h1, h2, h3, h4, h5, h6, hr, isindex, menu, noframes, noscript, ol, p, pre, table, ul`.

Los siguientes elementos también se considera que son de bloque: `dd, dt, frame-set, li, tbody, td, tfoot, th, thead, tr`.

Los siguientes elementos pueden ser en línea y de bloque según las circunstancias: `button, del, iframe, ins, map, object, script`.

2.5 Sintaxis de las Etiquetas XHTML

El lenguaje HTML original era muy permisivo en su sintaxis, por lo que era posible escribir sus etiquetas y atributos de muchas formas diferentes. Las etiquetas por ejemplo podían escribirse en mayúsculas, en minúsculas e incluso combinando mayúsculas y minúsculas. El valor de los atributos de las etiquetas se podían indicar con y sin comillas (""). Además, el orden en el que se abrían y cerraban las etiquetas no era importante.

La flexibilidad de HTML puede parecer un aspecto positivo, pero el resultado final son páginas con un código HTML desordenado, difícil de mantener y muy poco profesional.

Afortunadamente, XHTML soluciona estos problemas añadiendo ciertas normas en la forma de escribir las etiquetas y atributos.

A continuación se muestran las cinco restricciones básicas que introduce XHTML respecto a HTML en la sintaxis de sus etiquetas:

1. Las etiquetas se tienen que cerrar de acuerdo a como se abren, en función de su jerarquía. Veamos algunos ejemplos:

Ejemplo correcto en XHTML:

```
<p>Este es un párrafo con <a>un enlace</a></p>
```

Ejemplo incorrecto en XHTML (pero correcto en HTML):

```
<p>Este es un párrafo con <a>un enlace</p></a>
```

2. Los nombres de las etiquetas y atributos siempre se escriben en minúsculas:

Ejemplo correcto en XHTML:

```
<p>Este es un párrafo con <a href="http://www.google.com">un enlace</a></p>
```

Ejemplo incorrecto en XHTML (pero correcto en HTML):

```
<P>Este es un párrafo con <A HREF="http://www.google.com">un enlace</A></P>
```

3. El valor de los atributos siempre se encierra con comillas:

Ejemplo correcto en XHTML:

```
<p>Este es un párrafo con <a href="http://www.google.com">un enlace</a></p>
```

Ejemplo incorrecto en XHTML (pero correcto en HTML):

```
<p>Este es un párrafo con <a href=http://www.google.com>un enlace</a></p>
```

4. Los atributos no se pueden comprimir:

Ejemplo correcto en XHTML:

```
<dl compact="compact">...</dl>
```

Ejemplo incorrecto en XHTML (pero correcto en HTML):

```
<dl compact>...</dl>
```

Este tipo de atributos en los que el nombre coincide con su valor no son muy habituales.

5. Todas las etiquetas deben cerrarse siempre:

La mayoría de etiquetas HTML encierran un contenido de texto entre la etiqueta de apertura y la etiqueta de cierre. Sin embargo, algunas etiquetas especiales llamadas "etiquetas vacías" no necesitan encerrar ningún texto.

La etiqueta `
` por ejemplo, se utiliza para indicar el comienzo de una nueva línea, tal y como se verá más adelante. Por sus características, la etiqueta `
` nunca encierra ningún contenido de texto.

Como el estándar XHTML obliga a cerrar todas las etiquetas abiertas, siempre que se incluya la etiqueta `
` se debería cerrar de forma seguida: `
</br>`. Para que el código resulte más cómodo de escribir, XHTML permite en estos casos escribir de forma abreviada una etiqueta que se abre y se cierra de forma consecutiva.

En lugar de abrir y cerrar de forma consecutiva la etiqueta (`
</br>`) se puede utilizar la sintaxis `
` para indicar que es una etiqueta vacía que se abre y se cierra en ese mismo punto. En la forma compacta es habitual equivocarse con la posición del carácter `/`.

Ejemplo correcto en XHTML:

`
`

Ejemplo incorrecto en XHTML (pero correcto en HTML):

`
`

Además de estas cinco restricciones básicas, XHTML incluye otros cambios más avanzados respecto a HTML:

- Antes de acceder al valor de un atributo, se eliminan todos los espacios en blanco que se encuentran antes y después del valor. Además, se eliminan todos los espacios en blanco sobrantes dentro del valor de un atributo. En otras palabras, si en el interior de un atributo se incluyen varios espacios en blanco seguidos, se eliminan todos salvo un único espacio en blanco utilizado para separar las diferentes palabras.
- Como se explicará más adelante al hablar de la etiqueta `<script>`, el código JavaScript debe encerrarse entre unas etiquetas especiales (`<! [CDATA[y]]>`) para evitar que el navegador interprete de forma errónea caracteres como `& y <`.
- Las páginas XHTML deben prescindir del atributo `name` para identificar de forma única a los elementos. En su lugar, siempre debe utilizarse el atributo `id`. De hecho, en la versión 1.0 del estándar XHTML, el atributo `name` se ha declarado obsoleto para las etiquetas `a`, `applet`, `form`, `frame`, `iframe`, `img` y `map`.

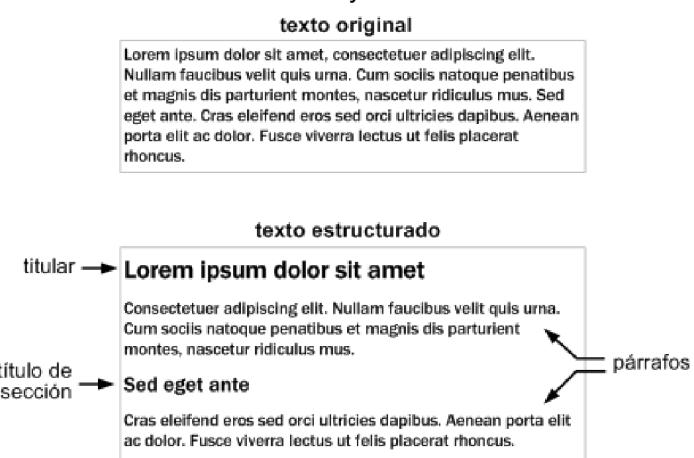
3 TEXTO

La mayor parte del contenido de las páginas HTML habituales está formado por texto, llegando a ser más del 90% del código de la página. Por este motivo, es muy importante conocer los elementos y etiquetas que define HTML para el manejo del texto.

El lenguaje HTML incorpora al tratamiento del texto muchas de las ideas y normas establecidas en otros entornos de publicación de contenidos. De esta forma, HTML define etiquetas para estructurar el contenido en secciones y párrafos y define otras etiquetas para marcar elementos importantes dentro del texto.

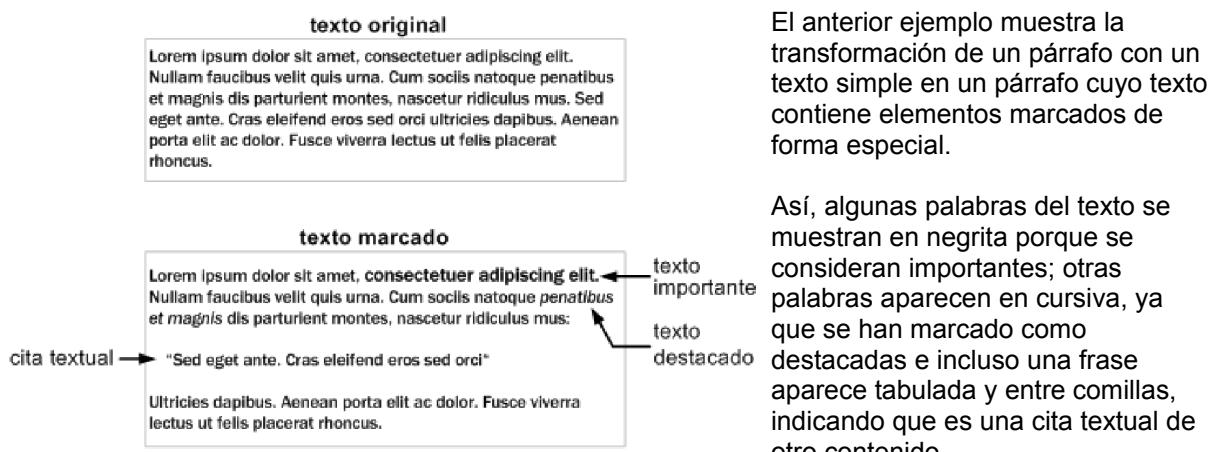
La tarea inicial del editor de contenidos HTML consiste en estructurar el texto original definiendo sus párrafos, titulares y títulos de sección, como se muestra en la siguiente imagen:

NOTA: *Lorem Ipsum* es un texto en latín usado como relleno en impresiones.
Mas información: <http://es.lipsum.com/>



El proceso de estructurar un texto simple consiste en indicar las diferentes zonas o secciones que componen el texto. De esta forma, los textos estructurados utilizan etiquetas para delimitar cada párrafo y títulos de sección para delimitar cada una de las secciones que forman el texto.

Una vez definida la estructura básica de los contenidos de la página, el siguiente paso consiste en marcar los diferentes elementos dentro del propio texto: definiciones, abreviaturas, textos importantes, textos modificados, citas a otras referencias, etc.



En las secciones siguientes se muestran todas las etiquetas que define HTML para estructurar y marcar el texto. Además, se hace una mención especial al tratamiento que hace HTML de los espacios en blanco y las nuevas líneas.

El anterior ejemplo muestra la transformación de un párrafo con un texto simple en un párrafo cuyo texto contiene elementos marcados de forma especial.

Así, algunas palabras del texto se muestran en negrita porque se consideran importantes; otras palabras aparecen en cursiva, ya que se han marcado como destacadas e incluso una frase aparece tabulada y entre comillas, indicando que es una cita textual de otro contenido.

3.1 Estructura

La forma más sencilla de estructurar un texto consiste en separarlo por párrafos. Además, HTML permite incluir títulos que delimitan cada una de las secciones.

3.1.1 Párrafos

Una de las etiquetas más utilizadas de HTML es la etiqueta `<p>`, que permite definir los párrafos que forman el texto de una página. Para delimitar el texto de un párrafo, se encierra ese texto con la etiqueta `<p>`, como muestra el siguiente ejemplo ([Ej03.01a-Párrafos.html](#)):

```
<html>
  <head>
    <title>Ejemplo de texto estructurado con párrafos</title>
    <meta charset= "UTF-8" />
  </head>
  <body>
    <p>Este es el texto que forma el primer párrafo de la página.
    Los párrafos pueden ocupar varias líneas y el navegador se encarga
    de ajustar su longitud al tamaño de la ventana.</p>

    <p>El segundo párrafo de la página también se define encerrando
    su texto con la etiqueta p. El navegador también se encarga de
    separar automáticamente cada párrafo.</p>
  </body>
</html>
```

La siguiente tabla recoge la definición formal de la etiqueta `<p>`:

<p>	Párrafos
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Delimita el contenido de un párrafo de texto

3.1.2 Secciones

Las páginas HTML habituales suelen tener una estructura más compleja que la que se puede crear solamente mediante párrafos. De hecho, es habitual que las páginas se dividan en diferentes secciones jerárquicas.

Los títulos de sección se utilizan para delimitar el comienzo de cada sección de la página. HTML permite crear secciones de hasta seis niveles de importancia. De esta forma, aunque una página puede definir cualquier número de secciones, sólo puede incluir seis niveles jerárquicos.

Las etiquetas que definen los títulos de sección son `<h1>`, `<h2>`, `<h3>`, `<h4>`, `<h5>` y `<h6>`. La etiqueta `<h1>` es la de mayor importancia y por tanto se utiliza para definir los titulares de la página. La importancia del resto de etiquetas es descendiente, de forma que la etiqueta `<h6>` es la que se utiliza para delimitar las secciones menos importantes de la página.

Las secciones en HTML son muy importantes para la Optimización de Motores de Búsqueda (search engine optimization o SEO), puesto que son indexadas por los principales buscadores indicando los contenidos más importantes de nuestra web.

Más información: <http://codatavern.com/optimizacion-etiquetas-h1-h2-h3-seo/>

<http://www.hectormainar.com/seo/h1-h2-y-h3-como-utilizar-correctamente-las-etiquetas-de-encabezado-de-html>

A continuación se muestra la definición formal de la etiqueta `<h1>`, siendo idéntica la definición del resto de etiquetas referidas a los títulos de sección:

<h1>	Títulos de sección
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Define los títulos de las secciones de mayor importancia de la página.

Al igual que la etiqueta `<p>`, las etiquetas de título de sección son elementos de bloque y no tienen atributos específicos.

Las etiquetas `<h1>`, ..., `<h6>` definen títulos de sección, no secciones completas. Por este motivo, no es necesario encerrar los contenidos de una sección con su etiqueta correspondiente. Solamente se debe encerrar con las etiquetas `<h1>`, ..., `<h6>` los títulos de cada sección.

El siguiente ejemplo muestra el uso real de las etiquetas de título de sección:

Fuente: <http://es.wikipedia.org/wiki/Usuario:Ivanrguez>

Veamos el siguiente ejemplo (**Ej03.01b-Secciones.html**):

```
<html>
  <head>
    <title>Usuario:Ivanrguez - Wikipedia, la enciclopedia libre</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <h1>Usuario:Ivanrguez</h1>

    <h2>Biografía</h2>
    <h3>Datos Personales</h3>
    <p>Mi Nombre es Iván Rodríguez. Naci en Sevilla, el 26 de Mayo de 1976. Soy
       Programador de Aplicaciones (Java, PHP, ASP.NET), Analista, Administrador de
       Sistemas (Gestor de Bases de Datos MySQL / Oracle 9i/10g, CMS Moodle, etc) y
       Dinamizador / Tutor de Acciones Formativas en Nuevas Tecnologías. Además, soy
       Socio del club de fútbol Real Betis Balompié.</p>

    <h3>Títulos</h3>
    <p>Poseo los siguientes estudios:</p>
  </body>
</html>
```

Los navegadores asignan de forma automáticamente el tamaño del título de cada sección en función de su importancia. Así, los títulos de sección `<h1>` se muestran con el tamaño de letra más grande, ya que son el nivel jerárquico superior, mientras que los títulos de sección `<h6>` se visualizan con un tamaño de letra muy pequeño, adecuado para el nivel jerárquico de menor importancia.

Evidentemente, el aspecto que los navegadores aplican por defecto a los títulos de sección se puede modificar utilizando las hojas de estilos de CSS.

3.2 Marcado Básico de Texto

3.2.1 Etiquetas básicas

Una vez estructurado el texto en párrafos y secciones, el siguiente paso es el marcado de los elementos que componen el texto. Los textos habituales están formados por elementos como palabras en negrita o cursiva, anotaciones y correcciones, citas a otros documentos externos, etc. HTML proporciona varias etiquetas para marcar cada uno de los diferentes tipos de texto.

Entre las etiquetas más utilizadas para marcar texto se encuentran `` y ``. La definición formal de estas dos etiquetas se muestra a continuación:

	Énfasis
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En línea
Descripción	Realza la importancia del texto que encierra (Cursiva)

	Énfasis mas acentuado
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En Línea
Descripción	Realza con la máxima importancia el texto que encierra (Negrita)

Veamos el siguiente ejemplo ([Ej03.02a-CursivaNegrita.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta em y strong</title>
  </head>
  <body>
    <h1>Marcado de Texto </h1>
    <p>El Lenguaje HTML permite marcar parte del texto como <em>Cursiva</em> o bien en
       <strong>negrita</strong>.</p>
  </body>
</html>
```

HTML también permite marcar de forma adecuada las modificaciones realizadas en el contenido de una página. En otras palabras, HTML permite indicar de forma clara el texto que ha sido eliminado y el texto que ha sido añadido a un determinado texto original. Las etiquetas utilizadas son `<ins>` y ``, cuya definición formal es la siguiente:

<ins>	Insertión
Atributos comunes	básicos, i18n y eventos
Atributos específicos	cite = "url" - Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se realizó la modificación. datetime = "fecha" - Especifica la fecha y hora en la que se realizó el cambio
Tipo de elemento	En Bloque y en Línea
Descripción	Se emplea para marcar una modificación en los contenidos originales consistente en la inserción de un nuevo contenido

	Borrado
Atributos comunes	básicos, i18n y eventos
Atributos específicos	cite = "url" - Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se realizó la modificación. datetime = "fecha" - Especifica la fecha y hora en la que se realizó el cambio
Tipo de elemento	En Bloque y en Línea
Descripción	Se emplea para marcar una modificación en los contenidos originales consistente en el borrado de cierto contenido

Las dos etiquetas cuentan con los mismos atributos específicos, que opcionalmente se pueden añadir para proporcionar más información sobre los cambios realizados. El atributo `cite` se emplea para indicar la dirección de un documento externo en el que se puede encontrar más información relacionada con la inserción o el borrado de texto. El atributo `datetime` puede utilizarse para indicar la fecha y la hora en la que se realizó cada cambio.

Veamos el siguiente ejemplo ([Ej03.02b-InsercionBorrado.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta ins y del</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <h3>Ejemplo de etiqueta ins y del</h3>

    <p>El HTML, acrónimo inglés de Hyper Text Markup Language (lenguaje de
    <del datetime="20121226" cite="http://es.wikipedia.org/wiki/HTML">
    marcado de hipertexto</del>
    <ins datetime="20091026" cite="http://es.wikipedia.org/wiki/HTML">
      marcas hipertextuales</ins>) es un lenguaje de marcación diseñado para estructurar
      textos y presentarlos en forma de hipertexto.</p>
  </body>
</html>
```

Por defecto, el texto eliminado (marcado con la etiqueta ``) se muestra **tachado** de forma que el usuario pueda identificarlo fácilmente como un texto que formaba parte del texto original y que ya no tiene validez. El texto insertado (marcado con la etiqueta `<ins>`) se muestra **subrayado**, de forma que el usuario pueda identificarlo como un texto nuevo que no formaba parte del texto original.

Por otra parte, en muchos tipos de páginas (artículos, noticias) es habitual citar literalmente un texto externo. HTML define la etiqueta `<blockquote>` para incluir citas textuales en las páginas web. La definición de la etiqueta HTML con el nombre más largo se muestra a continuación:

<blockquote>	Citas
Atributos comunes	básicos, i18n y eventos
Atributos específicos	cite = "url" - Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se realizó la modificación.
Tipo de elemento	En Bloque
Descripción	El texto que encierra es una cita textual de otro texto externo

Al igual que `<ins>` y ``, la etiqueta `<blockquote>` permite indicar mediante el atributo `cite` la dirección de un documento del que se ha extraído la cita.

Veamos el siguiente ejemplo ([Ej03.02c-Citas.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta blockquote</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p>Según el W3C, el valor del atributo <em>cite</em> en las etiquetas
      <strong>blockquote</strong> tiene el siguiente significado:</p>
      <blockquote cite="http://www.w3.org/TR/html401/struct/text.html">"El valor de este
        atributo es una dirección URL que indica el documento original de la cita."</blockquote>
    </body>
</html>
```

Para indicar de forma clara que el texto es una cita externa, los navegadores muestran por defecto el texto del elemento `<blockquote>` con un gran margen en la parte izquierda.

3.2.2 Otras Etiquetas

Además de las etiquetas anteriores tenemos una serie de etiquetas cuyo uso es menor, aunque entran dentro del estándar. Así tenemos etiquetas específicas para crear Superíndices (por ejemplo HM₃) o Subíndices (H₂O) – en inglés, Superscript / Subscript –.

<code><sup></code>	Superíndice
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En línea
Descripción	Muestra el texto por encima del nivel medio de la línea actual

<code><sub></code>	Suberíndice
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En línea
Descripción	Muestra el texto por debajo del nivel medio de la línea actual

Veamos un ejemplo sencillo ([Ej03.02d-SuperSubIndices.html](#)):

```
<html>
  <head>
    <title>Ejemplo de Superíndices - Subíndices</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p>Vemos un par de ejemplos de Superíndices y Subíndices </p>
    <p>Unidades de Espacio/Volumen: M2; H2O</p>
    <p>Fórmulas Químicas: CO2; H2O</p>
  </body>
</html>
```

Ejercicio1

Estructurar y marcar el siguiente texto extraído de la Wikipedia (http://es.wikipedia.org/wiki/Exploración_espacial) para que el navegador lo muestre con el aspecto de la siguiente imagen:

Exploración espacial

(Redirigido desde «[Exploracion espacial](#)»)

La **exploración espacial** designa los esfuerzos del hombre en estudiar el [espacio](#) y sus [astros](#) desde el punto de vista [científico](#) y de su explotación [económica](#).

Estos esfuerzos pueden involucrar tanto [seres humanos](#) viajando en [naves espaciales](#) como satélites con recursos de [telemetría](#) o [sondas](#) teleguiadas enviadas a otros [planetas](#) (orbitando o aterrizando en la superficie de estos cuerpos celestes).

La ciencia que estudia los vuelos espaciales y la [tecnología](#) relacionada con ellos se denomina [astronáutica](#). Las personas que pilotan naves espaciales, o son pasajeros en ellas, se llaman [astronautas](#) (en [Rusia](#): [cosmonautas](#); en [China](#): [taikonautas](#)). Técnicamente se considera astronauta a todo aquel que emprenda un [vuelo suborbital](#) (sin entrar en [órbita](#)) u orbital a como mínimo 100 km de altitud (considerado el límite externo de la [atmósfera](#)).

El [cielo](#) siempre ha atraído la atención y los sueños del hombre. Ya en [1634](#) se publicó la que se considera primera novela de ciencia ficción, [Somnium](#), de [Johannes Kepler](#), que narra un hipotético viaje a la [Luna](#). Más tarde, en [1865](#), en una famosa obra de ficción titulada "[De la Tierra a la Luna](#)", [Julio Verne](#) escribe sobre un grupo de hombres que viajó hasta la Luna usando un gigantesco [cañón](#). En [Francia](#), [Georges Méliès](#), uno de los pioneros del [cine](#), tomaba la novela de Verne para crear "[Le voyage dans la Lune](#)" ([1902](#)), una de las primeras [películas](#) de [ciencia ficción](#) en la que describía un increíble viaje a la Luna. En obras como "[La guerra de los mundos](#)" ([1898](#)) y "[The First Men in The Moon](#)" ([1901](#)), [H.G.Wells](#) también se concibieron ideas de exploración del espacio y de contacto con [civilizaciones extraterrestres](#).

Ejercicio2

Repetir el proceso anterior con este fragmento del artículo de la Wikipedia sobre la Molécula del agua (http://es.wikipedia.org/wiki/Mol%C3%A9cula_de_agua).

Otras propiedades

- pH neutro.
- Con ciertas sales forma [hidratos](#).
- Reacciona con los óxidos de metales formando [bases](#).
- Es [catalizador](#) en muchas reacciones químicas.
- Presenta un equilibrio de autoionización, en el cual hay [iones](#) H_3O^+ y OH^- .

Estudio Hidrobiológico

La realización de un estudio hidrobiológico permite:

- Proporcionar datos sobre el estado de un sistema acuático de forma regular.
- Documentar la variabilidad a corto y largo plazo de la calidad del agua por fenómenos naturales o actividades humanas.
- Evaluar el impacto de la polución producido por la actividad humana.
- Evaluar la influencia de ciertas zonas de muestreo sobre la fauna del lugar.
- Evaluar las características hidráulicas del cauce del río y la evolución del caudal mediante medidas de flujo. De esta manera, se puede establecer las variaciones de caudal que sufre el río a lo largo de ciclo estacional y anual.
- Realizar un estudio de la rivera.
- Evaluar los índices Biológicos.

3.3 Marcado Avanzado de Texto

Las páginas y documentos más avanzados suelen incluir otros elementos importantes que se deben marcar de forma adecuada. Por ello, HTML incluye muchas otras etiquetas que permiten marcar más elementos del texto. La etiqueta `<abbr>` marca las abreviaturas de un texto y la etiqueta `<acronym>` se emplea para marcar las siglas o acrónimos. Su definición es la siguiente:

<abbr>	Abreviaturas
Atributos comunes	básicos, i18n y eventos
Atributos específicos	title = "texto" - Indica el significado completo de la abreviatura
Tipo de elemento	En Línea
Descripción	Marca las abreviaturas del texto y proporciona su significado

<acronym>	Acrónimos o siglas
Atributos comunes	básicos, i18n y eventos
Atributos específicos	title = "texto" - Indica el significado completo del acrónimo o sigla
Tipo de elemento	En Línea
Descripción	Marca las siglas o acrónimos del texto y proporciona su significado

En ambos casos, el atributo title se puede utilizar para incluir el significado completo de la abreviatura o sigla. Veamos el siguiente ejemplo ([Ej03.03a-AbreviaturasAcrónimos.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta abbr y acronym</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p>El lenguaje
    <acronym title="HyperText Markup Language">HTML</acronym> es estandarizado por el
    <acronym title="World Wide Web Consortium">W3C</acronym>.</p>
    <p>Un ejemplo de abreviatura sería: <abbr title="Teléfono">Tel.</abbr></p>
  </body>
</html>
```

HTML5

En el estándar, desaparece la etiqueta `<acronym>` y queda solo `<abbr>`.

La mayoría de navegadores muestran por defecto un borde inferior punteado para todos los elementos `<abbr>` y `<acronym>`. Al posicionar el puntero del ratón sobre la palabra subrayada, el navegador muestra un pequeño recuadro (tooltip en inglés) con el valor del atributo title.

Por otra parte, en ocasiones resulta útil incluir la definición de una palabra extraña o cuyo uso está restringido a un entorno muy determinado. HTML incluye la etiqueta `<dfn>` para proporcionar al usuario la definición de todas las palabras para las que se considere apropiado.

<dfn>	Definición
Atributos comunes	básicos, i18n y eventos
Atributos específicos	title = "texto" - Indica el significado completo del término
Tipo de elemento	En Línea
Descripción	Marca las definiciones de ciertos términos y proporciona su significado

El siguiente ejemplo muestra cómo se utiliza la etiqueta `<dfn>` para incluir la definición completa de una palabra cuyo uso no es habitual fuera de los ámbitos médicos y psicológicos:

Veamos el siguiente ejemplo ([Ej03.03b-Definición.html](#)):

```
<html>
<head>
  <title>Ejemplo de etiqueta dfn</title>
  <meta charset="UTF-8" />
</head>
<body>
  <h2>Informe clínico: </h2>
  <p>Con estos síntomas, podría tratarse de un caso de
    <dfn title="Imagen o sensación subjetiva, propia de un sentido, determinada por otra
    sensación que afecta a un sentido diferente">sinestesia</dfn> </p>
</body>
</html>
```

HTML incluye dos etiquetas que se pueden utilizar para marcar un texto como una citación:

<cite>	Cita
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En Línea
Descripción	Se emplea para marcar una cita o una referencia a otras fuentes

<blockquote>	Referencia a una Cita
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En Línea
Descripción	Se emplea para marcar una referencia a una cita

En ocasiones, no está clara la diferencia entre `<cite>` y `<blockquote>`. El elemento `<cite>` marca el autor de la cita (persona, documento, etc.) y `<blockquote>` marca el contenido de la propia cita. En el siguiente ejemplo, `<blockquote>` encierra el contenido de una frase célebre y `<cite>` encierra el nombre de su autor ([Ej03.03c-Citas.html](#)):

```
<html>
<head>
  <title>Ejemplo de etiqueta dfn</title>
  <meta charset="UTF-8" />
</head>
<body>
  <p>Como dijo <cite>Mahatma Gandhi</cite>:
    <blockquote>Vive como si fueras a morir mañana y aprende como si fueras a vivir para
    siempre.</blockquote></p>
</body>
</html>
```

HTML5

En el estándar, las etiquetas `<dfn>`, `<cite>` y `<blockquote>` se mantienen inalteradas.

Ejercicio 03

Estructurar y marcar el siguiente texto para que el navegador lo muestre con el aspecto de la siguiente imagen:

El Ártico ha perdido en un año el 14% de su hielo

Quedan 10 años para reaccionar

El director del Instituto Goddard para Estudios Espaciales de la NASA, James Hansen, alertó el pasado miércoles de que el planeta dispone un plazo de 10 años para actuar de forma decisiva frente al calentamiento global y evitar una catástrofe climática. Hansen, uno de los investigadores decanos del cambio climático, recordó a los gobiernos que deben controlar el aumento de las emisiones de dióxido de carbono y mantener la subida de las temperaturas globales a 1 grado Celsius. De lo contrario, Hansen habla de la extinción del 50% de las especies, sequías, más huracanes y olas de calor más largas.

Otros avisos

10/08/06: La revista Science publica un estudio de la Universidad de Texas que dice que la pérdida de hielo en Groenlandia se ha triplicado desde 2004.

19/04/06: Un informe de CC OO revela que España aumentó sus emisiones de gases de efecto invernadero en un 52,8% entre 1990 y 2005.

18/03/06: El Consejo Mundial del Agua recuerda que los desastres naturales son la mayor amenaza para Europa, sobre todo para España.

19/02/06: El científico británico Chris Rapley alerta de que el deshielo en la Antártida podría subir hasta 5 metros el nivel del mar.

Robert Swan, explorador.

«La Antártida nos está lanzando un mensaje»

NOTA: Marcar la última frase como una cita.

3.4 Marcado Genérico de Texto

El estándar HTML/XHTML incluye numerosas etiquetas para marcar los contenidos de texto. No obstante, la infinita variedad de posibles contenidos textuales hace que no sean suficientes. Si se considera el siguiente ejemplo:

Importante: si quiere ponerse en contacto con la empresa ACME, puede hacerlo en el teléfono 900 555 555 o a través de la dirección de correo electrónico contacto@acme.org

El texto del ejemplo anterior contiene elementos de texto importantes, siglas, números de teléfono y direcciones de correo electrónico. XHTML define la etiqueta **para marcar los elementos importantes y para marcar las siglas:**

Importante: si quiere ponerse en contacto con la empresa <acronym>ACME</acronym>, puede hacerlo en el teléfono 900 555 555 o a través de la dirección de correo electrónico contacto@acme.org

Desafortunadamente, XHTML no define ninguna etiqueta específica para marcar números de teléfono o direcciones de correo electrónico. De la misma forma, no define etiquetas para otros posibles elementos que se pueden encontrar en los contenidos de texto.

Por este motivo, el estándar HTML/XHTML incluye una etiqueta llamada que se emplea para marcar cualquier elemento que no se puede marcar con las otras etiquetas definidas.

	Marca de Texto
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En Línea
Descripción	Se emplea para marcar contenidos y etiquetas en línea

Siguiendo con el ejemplo anterior, se usa para marcar el teléfono y la dirección de correo:

Importante: si quiere ponerse en contacto con la empresa <acronym>ACME</acronym>, puede hacerlo en el teléfono 900 555 555 o a través de la dirección de correo electrónico contacto@acme.org

La etiqueta se visualiza por defecto con el mismo aspecto que el texto normal. Por tanto es habitual utilizar esta etiqueta junto con los atributos `id` y `class` para modificar posteriormente su aspecto con CSS. Veamos el siguiente ejemplo ([Ej03.04-Span.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta span</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p><strong>Importante</strong>:</p>
    <p>Si quiere ponerse en contacto con la empresa <acronym>ACME</acronym>, puede hacerlo en el teléfono <span style="border: 1px solid #F00;">900 555 555</span> o a través del correo electrónico <span style="border: 1px solid #00F;">contacto@acme.org</span></p>
  </body>
</html>
```

La etiqueta sólo se puede utilizar para encerrar contenidos y etiquetas en línea. Cuando se quieren estructurar elementos de bloque, se utiliza la etiqueta `<div>`, tal y como se verá en capítulos posteriores.

3.5 Espacios en blanco y nuevas líneas

El aspecto más sorprendente del lenguaje HTML cuando se desarrollan los primeros documentos es el tratamiento especial de los "espacios en blanco" del texto. HTML considera espacio en blanco a los espacios en blanco, los tabuladores, los retornos de carro y el carácter de nueva línea (ENTER o Intro).

Veamos el siguiente ejemplo ([Ej03.05a-Espacios.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta p</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p>Este primer párrafo no contiene saltos de linea ni otro tipo de espaciado.</p>
    <p>Este segundo párrafo sí que contiene saltos
      de
      línea y
      otro     tipo
      de           espaciado.</p>
  </body>
</html>
```

Los dos párrafos de la imagen anterior se ven idénticos, aunque el segundo párrafo incluye varios espacios en blanco y está escrito en varias líneas diferentes. La razón de este comportamiento es que HTML ignora todos los espacios en blanco sobrantes, es decir, todos los espacios en blanco que no son el espacio en blanco que separa las palabras.

No obstante, HTML proporciona varias alternativas para poder incluir tantos espacios en blanco y tantas nuevas líneas como sean necesarias dentro del contenido textual de las páginas.

3.5.1 Nuevas líneas

Para incluir una nueva línea en un punto y forzar a que el texto que sigue se muestre en la línea inferior, se utiliza la etiqueta `
`. En cierta manera, insertar la etiqueta `
` en un determinado punto del texto equivale a presionar la tecla ENTER (o Intro) en ese mismo punto.

La definición formal de `
` se muestra a continuación:

<code>
</code>	Nueva Línea
Atributos comunes	básicos
Atributos específicos	-
Tipo de elemento	En Línea y etiqueta vacía
Descripción	Se emplea para marcar contenidos y etiquetas en línea

La etiqueta `
` es una de las pocas etiquetas especiales de HTML. La particularidad de `
` es que es una etiqueta vacía, es decir, no encierra ningún texto. De esta forma, la etiqueta debe abrirse y cerrarse de forma consecutiva: `
</br>`.

En estos casos, HTML permite utilizar un atajo para indicar que una etiqueta se está abriendo y cerrando de forma consecutiva: `
` (también se puede escribir como `
`). Utilizando la etiqueta `
` se puede rehacer el ejemplo anterior para que respete las líneas que forman el segundo párrafo.

Veamos un ejemplo ([Ej03.05b-Etiquetabr.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta br</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p>Este primer párrafo no contiene saltos de línea ni otro tipo de espaciado.</p>
    <p>Este segundo párrafo sí que contiene saltos <br/>
      de <br/>
      línea y <br/>
      otro     tipo <br/>
      de           espaciado.</p>
  </body>
</html>
```

3.5.2 Espacios en blanco

La solución al problema de los espacios en blanco no es tan sencilla como el de las nuevas líneas. Para incluir espacios en blanco adicionales, se debe sustituir cada nuevo espacio en blanco por el texto `&nbsp` (es importante incluir el símbolo `&` al principio y el símbolo `_` al final).

Así, el código HTML del ejemplo anterior se rehará así ([Ej03.05c-Caracter_nbsp.html](#)):

```
<html>
  <head>
    <title>Ejemplo de uso de nbsp</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p>Este primer párrafo no contiene saltos de línea ni otro tipo de espaciado.</p>
    <p>Este segundo párrafo sí que contiene saltos
      de
      línea y
      otro &nbsps;&nbsps; tipo
      de &nbsps;&nbsps;&nbsps;&nbsps;&nbsps; espaciado.</p>
  </body>
</html>
```

3.5.3 Texto Preformatado

En ocasiones, es necesario mostrar los espacios en blanco de un texto que no se puede modificar. Se trata de un caso habitual cuando una página web debe mostrar directamente el texto generado por alguna aplicación.

En estos casos, se puede utilizar la etiqueta `<pre>`, que muestra el texto tal y como se ha escrito, respetando todos los espacios en blanco y todas las nuevas líneas. La definición formal de la etiqueta se muestra a continuación:

<pre>	Texto Preformatado
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En Bloque
Descripción	Muestra el texto que encierra tal y como está escrito (respetando los espacios en blanco)

El siguiente ejemplo muestra el uso de la etiqueta `<pre>` ([Ej03.05d-TextoPre.html](#))

```
<html>
  <head>
    <title>Ejemplo de uso de pre</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <pre>
      La etiqueta pre
      respeta los espacios en blanco
      Y
      muestra el texto
      tal y como
      está escrito
    </pre>
    <p>
      La etiqueta pre
      respeta los espacios en blanco
      Y
      muestra el texto
      tal y como
      está escrito
    </p>
  </body>
</html>
```

El ejemplo anterior incluye el mismo texto (con espacios en blanco y varias líneas) dentro de una etiqueta `<pre>` y dentro de una etiqueta `<p>`.

El primer texto se ve en pantalla tal y como se ha escrito, respetando todos los espacios en blanco y todas las nuevas líneas. El segundo texto se ve como un párrafo normal, ya que HTML ha eliminado todos los espacios en blanco sobrantes. Los elementos `<pre>` son especiales, ya que los navegadores les aplican las siguientes reglas:

- Mantienen todos los espacios en blanco (tabuladores, espacios y nuevas líneas)
- Muestra el texto con un tipo de letra especial, denominado "de ancho fijo", ya que todas sus letras son de la misma anchura
- No se ajusta la longitud de las líneas (las líneas largas producen un scroll horizontal en la ventana del navegador)

Esta última característica diferencia por completo a los párrafos de los elementos `<pre>`. Como se ha visto, los navegadores ajustan la anchura de los párrafos de texto para que ocupen todo el tamaño de la ventana. Sin embargo, los elementos `<pre>` se muestran tal y como son originalmente, por lo que una línea muy larga dentro de un elemento `<pre>` provoca que la anchura de la página sea superior a la anchura de la ventana del navegador.

Si en el ejemplo anterior se añade más texto al final de la segunda línea (para producir una línea larga), el navegador muestra un scroll horizontal ya que el texto completo no cabe en el tamaño de la ventana y las líneas de los elementos `<pre>` nunca se ajustan.

Otra etiqueta relacionada con `<pre>` es la etiqueta `<code>`, que se utiliza para mostrar código fuente de cualquier lenguaje de programación. La definición formal de `<code>` es la siguiente:

<code>	Código Fuente
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En Línea
Descripción	Delimita el texto considerado un fragmento de código fuente

En la mayoría de páginas web, no tiene sentido utilizar la etiqueta <code>. Sin embargo, en muchas páginas web técnicas que incluyen listados de programas, trozos de código o etiquetas HTML, lo correcto es emplear la etiqueta <code>. Ejemplo ([Ej03.05e-Code.html](#))

```
<html>
  <head>
    <title>Ejemplo de etiqueta code</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <code>
      La etiqueta code
      no respeta los espacios en blanco
    </code>
    <p>La etiqueta code es similar a la
      etiqueta pre, sobre todo en el formato
      del texto.</p>
  </body>
</html>
```

Al igual que sucede con los elementos <pre>, el texto encerrado por la etiqueta <code> se muestra con un tipo de letra especial de ancho fijo. Por el contrario, el elemento <code> no respeta los espacios en blanco ni las líneas, por lo que su comportamiento es similar a la etiqueta <p>. La última diferencia es que <code> es un elemento en línea, mientras que <pre> es un elemento de bloque.

Ejercicio 04

Determinar el código HTML que corresponde al siguiente documento:

Características de los planetas

Nombre	Diametro relativo	Periodo orbital	Número de lunas

Mercurio	0,382	0,24 años	0
Venus	0,949	0,62 años	0
Tierra	1	1 año	1
Marte	0,532	1,88 años	2
Júpiter	11,209	11,86 años	49
Saturno	9,449	29,46 años	52
Urano	4,007	84,01 años	27
Neptuno	3,883	164,80 años	13

3.6 Codificación de Caracteres

Una consideración importante directamente relacionada con el texto de las páginas HTML es la codificación de los caracteres y la inserción de caracteres especiales. Algunos de los caracteres que se utilizan habitualmente en los textos no se pueden incluir directamente en las páginas web:

- Los caracteres que utiliza HTML para definir sus etiquetas (<, > y ") no se pueden utilizar libremente.
- Los caracteres propios de los idiomas que no son el inglés (ñ, á, ç, ¿, ¡, etc.) pueden ser problemáticos dependiendo de la codificación de caracteres utilizada.

La solución a la primera limitación consiste en sustituir los caracteres reservados de HTML por unas expresiones llamadas entidades HTML y que representan a cada carácter:

Caracteres especiales básicos

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

<	<	>	>
&	&	"	"

Caracteres especiales del HTML 2.0

Á	Á	À	À
É	É	È	È
í	Í	&lgrave;	Ì
Ó	Ó	Ò	Ò
Ú	Ú	Ù	Ù
á	á	à	à
é	é	è	è
í	í	ì	ì
ó	ó	ò	ò
ú	ú	ù	ù
Ä	Ä	Â	Â
Ë	Ë	Ê	Ê
&luml;	Ï	Î	Î
Ö	Ö	Ô	Ô
Ü	Ü	Û	Û
ä	ä	â	â
ë	ë	ê	ê
ï	ï	î	î
ö	ö	ô	ô
ü	ü	û	û
Ã	Ã	å	å
Ñ	Ñ	Å	Ã
Õ	Õ	Ç	Ç
ã	ã	ç	ç
ñ	ñ	Ý	Ý
õ	õ	ý	ý

Ø	Ø	ÿ	ÿ
ø	ø	Þ	þ
Ð	Ð	&torn;	þ
ð	ð	Æ	Æ
ß	ß	æ	æ

Caracteres especiales del HTML 3.2

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

Otros caracteres especiales

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

Mas información:

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

Ejercicio5

Determinar el código HTML que corresponde al siguiente documento:

Sintaxis de la etiqueta <blockquote>

La sintaxis de la etiqueta <blockquote> se muestra a continuación:

```
<blockquote cite="...direccion original de la cita...">Texto que  
se cita</blockquote>
```

RESUMEN – TEXTO

- | | |
|---------------------------|---|
| <p> | → Como ya hemos visto delimita un párrafo. |
| <h1> | → Indica un Titular / Sección. Tiene 6 niveles <h1> al <h6> |
|
 | |
| | → Muestra el texto en Negrita. |
| | → Muestra el texto en Cursiva. |
| <u> | → Muestra el texto Subrayado. |
| | → Indica texto eliminado. |
| <ins> | → Presenta el texto como una inserción |
|
 | |
| <sub> | → Muestra el texto como subíndice |
| <sup> | → Muestra el texto como superíndice |
|
 | |
| <abbr> | → Marca las abreviaturas del texto |
| <acronym> | → Marca las siglas o acrónimos del texto (HTML5 → NO) |
| <dfn> | → Marca las definiciones de ciertos términos |
| <cite> | → Se emplea para marcar una cita |
| <blockquote> | → Se emplea para marcar una referencia a una cita |
|
 | |
| | → Se emplea para marcar contenidos y etiquetas en línea |
|
 | → Realiza un salto de línea. Se presenta así:
. |
| <pre> | → Muestra el texto que encierra tal y como está escrito |
| <code> | → Delimita el texto considerado un fragmento de código fuente |

4 ENLACES

El lenguaje de marcado HTML se definió teniendo en cuenta algunas de las características que existían en ese momento para la publicación digital de contenidos. Entre los conceptos utilizados en su creación, se encuentra el mecanismo de "hipertexto".

De hecho, las letras "HT" de la sigla HTML significan "hipertexto" (hypertext en inglés), por lo que el significado completo de HTML podría traducirse como "lenguaje de marcado para hipertexto".

La incorporación del hipertexto fue una de las claves del éxito del lenguaje HTML, ya que permitió crear documentos interactivos que proporcionan información adicional cuando se solicita. El elemento principal del hipertexto es el "hiperenlace", también llamado "enlace web" o simplemente "enlace".

Los enlaces se utilizan para establecer relaciones entre dos recursos. Aunque la mayoría de enlaces relacionan páginas web, también es posible enlazar otros recursos como imágenes, documentos y archivos.

Una característica que no se suele tener en cuenta en los enlaces es que están formados por dos extremos y un sentido. En otras palabras, el enlace comienza en un recurso y apunta hacia otro recurso. Cada uno de los dos extremos se llaman "anchors" en inglés, que se puede traducir literalmente como "anclas".

4.1 URL

Antes de empezar a crear enlaces, es necesario comprender y dominar el concepto de URL. El acrónimo URL (del inglés Uniform Resource Locator) hace referencia al identificador único de cada recurso disponible en Internet. Las URL son esenciales para crear los enlaces, pero también se utilizan en otros elementos HTML como las imágenes y los formularios.

La URL de un recurso tiene dos objetivos principales:

- Identificar de forma única a ese recurso
- Permitir localizar de forma eficiente ese recurso

En primer lugar, las URL permiten que cada página HTML publicada en Internet tenga un nombre único que permita diferenciarla de las demás. De esta forma es posible crear enlaces que apunten de forma inequívoca a una determinada página.

Si se accede a la página principal de Google, la dirección que muestra el navegador es:
<http://www.google.es>

La cadena de texto <http://www.google.es> será la URL completa de la página principal de Google. La URL de las páginas es imprescindible para crear los enlaces, ya que permite distinguir una página de otra.

El segundo objetivo de las URL es el de permitir la localización eficiente de cada recurso de Internet. Para ello es necesario comprender las diferentes partes que forman las URL. Una URL sencilla siempre está formada por las mismas tres partes.

Por ejemplo, consideremos la siguiente URL:

<http://www.w3c.es/Divulgacion/html/logo/>

En este caso se trata del Logo oficial libre de HTML5 del consorcio W3C.

Las partes que componen la URL anterior son:

- Protocolo (`http://`): el mecanismo que debe utilizar el navegador para acceder a ese recurso. Todas las páginas web utilizan `http://`. Las páginas web seguras (por ejemplo las de los bancos y las de los servicios de email) utilizan `https://` (se añade una letra s).
- Servidor (`www.w3c.es`): simplificando mucho su explicación, se trata del ordenador en el que se encuentra guardada la página que se quiere acceder. Los navegadores son capaces de obtener la dirección de cada servidor a partir de su nombre.
- Ruta (`/Divulgacion/html/logo/`): camino que se debe seguir, una vez que se ha llegado al servidor, para localizar el recurso específico que se quiere acceder.

Por tanto, las URL no sólo identifican de forma única a cada recurso de Internet, sino que también proporcionan a los navegadores la información necesaria para poder llegar hasta ese recurso. La mayoría de URL son tan sencillas como la URL mostrada anteriormente. No obstante, existen URL complejas formadas por más partes.

<http://www.alistapart.com/comments/webstandards2008?page=5#42>

Las cinco partes que forman la URL anterior son:

- Protocolo (`http://`)
- Servidor (`www.alistapart.com`)
- Ruta (`/comments/webstandards2008`)
- Consulta (`?page=5`): información adicional necesaria para que el servidor localice correctamente el recurso que se quiere acceder. Siempre comienza con el carácter ? Y contiene una sucesión de palabras separadas por = y &
- Sección (`#42`): permite que el navegador se posicione automáticamente en una sección de la página web. Siempre comienza con el carácter #

Como las URL utilizan los caracteres :, =, & y / para separar sus partes, estos caracteres están reservados y no se pueden utilizar libremente. Además, algunos caracteres no están reservados pero pueden ser problemáticos si se utilizan en la propia URL.

Si es necesario incluir estos caracteres reservados y especiales en una URL, se sustituyen por combinaciones de caracteres seguros. Esta sustitución se denomina codificación de caracteres y el servidor realiza el proceso inverso (decodificación) cuando le llega una URL con los caracteres codificados.

A continuación se muestra la tabla para codificar los caracteres más comunes:

Carácter original	Carácter codificado	Carácter original	Carácter codificado
/	%2F	?	%3F
:	%3A	@	%40
=	%3D	&	%26
"	%22	\	%5C
'	%60	~	%7E
(espacio en blanco)	%20		%7C

Por otra parte, aunque desde hace tiempo ya es posible incluir en las URL caracteres de otros idiomas que no sean el inglés, aún no es completamente seguro utilizar estos caracteres en las URL. Si se utilizan letras como ñ, á, é o ç, es posible que algunos navegadores no las interpreten de forma correcta.

La solución consiste en codificar todos los caracteres que no existen en inglés. La siguiente tabla muestra la codificación de los caracteres más utilizados:

Carácter original	Carácter codificado	Carácter original	Carácter codificado
ñ	%F1	Ñ	%D1
á	%E1	Á	%C1
é	%E9	É	%C9
í	%ED	Í	%CD
ó	%F3	Ó	%D3
ú	%FA	Ú	%DA
ç	%E7	ç	%C7

Teniendo en cuenta las dos tablas anteriores de codificación de caracteres, es fácil crear las URL correctas sin caracteres problemáticos:

```
<!-- URL problemática -->
http://www.ejemplo.com/estaciones/otoño.html
<!-- URL correcta -->
http://www.ejemplo.com/estaciones/oto%F1o.html
<!-- URL problemática -->
http://www.ejemplo.com/ruta/nombre página.html
<!-- URL correcta -->
http://www.ejemplo.com/ruta/nombre%20p%Elgina.html
```

4.2 Enlaces Relativos y Absolutos

Las páginas web habituales suelen contener decenas de enlaces de diferentes tipos. La siguiente imagen muestra algunos de los tipos de enlaces de la página principal del sitio web <http://www.456bereastreet.com/archive/>

The screenshot shows the homepage of 456 Berea Street. At the top, there's a navigation bar with links for Home, About, Archive, Lab, Reviews, and Contact. Below the navigation, there's a main content area with a heading 'Archives'. To the right, there's a sidebar with sections for 'SUBSCRIBE / FOLLOW' (showing 44878 readers and a link to @rogerjohansson), 'SPONSORS' (listing 'Authentic Jobs' with a 'Comely Web HIRING' logo), and 'Externos y Absoluto' (listing job openings for 'Badass Senior PHP / LAMP Developer' at Sociagram and 'Senior Graphic Designer' at CorraTech). The main content area also contains links for categories like Accessibility, Browsers, Conferences, and CSS.

En esa página, cuando se pincha sobre algunos enlaces, el navegador abandona el sitio web para acceder a páginas que se encuentran en otros sitios. Estos enlaces se conocen como "enlaces externos". Sin embargo, la mayoría de enlaces de un sitio web apuntan a páginas del propio sitio web, por lo que se denominan "enlaces internos".

Además de internos/externos, la otra característica que diferencia a los enlaces (y por tanto, también a las URL) es si el enlace es absoluto o relativo. Las URL absolutas incluyen todas las partes de la URL (protocolo, servidor y ruta) por lo que no se necesita más información para obtener el recurso enlazado.

Las URL relativas prescinden de algunas partes de las URL para hacerlas más breves. Como se trata de URL incompletas, es necesario disponer de información adicional para obtener el recurso enlazado. En concreto, para que una URL relativa sea útil es imprescindible conocer la URL del origen del enlace.

Las URL relativas se construyen a partir de las URL absolutas y prescinden de la parte del protocolo, del nombre del servidor e incluso de parte o toda la ruta del recurso enlazado. Aunque las URL relativas pueden ser difíciles de entender para los que comienzan con HTML, son tan útiles que todos los sitios web las utilizan.

Imagina que dispones de una página publicada en <http://www.ejemplo.com/ruta1/ruta2/pag1.html> y quieres incluir en ella un enlace a otra página que se encuentra en <http://www.ejemplo.com/ruta1/ruta2/pag2.html>.

Como las URL identifican de forma única a los recursos de Internet y proporcionan la información necesaria para llegar hasta ellos, el enlace debería utilizar la URL completa de la segunda página.

Las URL completas también se llaman URL absolutas, ya que el navegador no necesita disponer de información adicional para localizar el recurso enlazado. Si se utilizan siempre las URL absolutas, los enlaces están completamente definidos.

Sin embargo, escribir siempre las URL completas es bastante aburrido, cuesta mucho tiempo y hace imposible cambiar la ubicación de los contenidos de un sitio web. Por ese motivo, casi todos los sitios web de Internet utilizan URL relativas siempre que es posible.

Una URL relativa es una versión abreviada de una URL absoluta. Su objetivo es eliminar todas las partes de la URL absoluta que se pueden adivinar a partir de la información de contexto de la página web. En otras palabras, las URL relativas aprovechan la inteligencia de los navegadores para crear URL incompletas que los navegadores pueden completar deduciendo la información que falta.

Considerando de nuevo el ejemplo anterior, la URL a la que se quiere enlazar utiliza el mismo protocolo y se encuentra en el mismo servidor que la página origen, por lo que la URL relativa puede prescindir de esas partes:

- URL absoluta: <http://www.ejemplo.com/ruta1/ruta2/pagina2.html>
- URL relativa: /ruta1/ruta2/pagina2.html

En el ejemplo anterior, las dos URL son equivalentes porque cuando no se indica el protocolo y el servidor de una URL, los navegadores suponen que son los mismos que los de la página origen. Por lo tanto, cuando el navegador encuentra la URL /ruta1/ruta2/pagina2.html, realiza el siguiente proceso:

1. La URL no es absoluta, por lo que se debe determinar la URL absoluta a partir de la URL relativa para poder cargar el recurso enlazado.
2. A la URL relativa le falta el protocolo y el servidor, por lo que se supone que son los mismos que los de la página origen (<http://> y www.ejemplo.com).
3. Se añaden las partes que faltan a la URL relativa para obtener la URL absoluta:
$$\begin{aligned} & \text{http://} + \text{www.ejemplo.com} + \text{/ruta1/ruta2/pagina2.html} \\ & = \\ & \text{http://www.ejemplo.com/ruta1/ruta2/pagina2.html.} \end{aligned}$$

Aunque el ejemplo mostrado es el caso más sencillo de URL relativa, existen otros casos más avanzados en los que se prescinde de parte o toda la ruta del recurso que se enlaza. A continuación se muestran los cuatro tipos diferentes de URL relativas:

1. El origen y el destino del enlace se encuentran en el mismo directorio

Si desde una página web se quiere enlazar un recurso que se encuentra en el mismo directorio del servidor, la URL relativa puede prescindir de todas las partes de la URL absoluta salvo el nombre del recurso enlazado.

Origen	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>
Recurso enlazado	Página web llamada <code>pagina2.html</code> y que se encuentra en el mismo directorio
URL absoluta	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina2.html</code>
URL relativa	<code>página2.html</code>

Cuando el navegador encuentra una URL relativa que sólo consiste en el nombre de un recurso, supone que el protocolo, servidor y directorio del recurso enlazado son los mismos que los del origen del enlace.

2. El destino del enlace se encuentra cerca de su origen y en un nivel superior

En este caso, el recurso que se enlaza no está en el mismo directorio que el origen del enlace pero sí que está cerca y en algún directorio superior. La URL relativa debe indicar de alguna manera que es necesario subir un nivel en la jerarquía de directorios para llegar hasta el recurso.

Para indicar al navegador que debe subir un nivel, se incluyen dos puntos y una barra (`..`) en la ruta del recurso enlazado. De esta forma, cada vez que aparece `..` en una URL relativa, significa que se debe subir un nivel.

Origen	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>
Recurso enlazado	Página web llamada <code>página2.html</code> y que se encuentra en el directorio superior llamado <code>ruta2</code>
URL absoluta	<code>http://www.ejemplo.com/ruta1/ruta2/página2.html</code>
URL relativa	<code>../página2.html</code>

Cuando el navegador encuentra la URL relativa `../página2.html`, sabe que para encontrar el recurso enlazado (`página2.html`) tiene que subir un nivel desde el lugar en el que se encuentra esa URL relativa.

La página que incluye esa URL se encuentra en el directorio `ruta1/ruta2/ruta3`, por lo que subir un nivel equivale entrar en el directorio `ruta1/ruta2`.

De la misma forma, si el destino se encuentra un par de niveles por encima, se debe incluir `..` dos veces seguidas:

Origen	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>
Recurso enlazado	Página web llamada <code>página2.html</code> y que se encuentra en el directorio superior llamado <code>ruta1</code>
URL absoluta	<code>http://www.ejemplo.com/ruta1/página2.html</code>
URL relativa	<code>../../página2.html</code>

Además de subir niveles, también se puede entrar en otros directorios para obtener los recursos:

Origen	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>
Recurso enlazado	Página web llamada <code>pagina2.html</code> y que se encuentra en el directorio <code>ruta4</code> que se encuentra en la raíz del servidor
URL absoluta	<code>http://www.ejemplo.com/ruta4/pagina2.html</code>
URL relativa	<code>../../../../ruta4/pagina2.html</code>

Si se intentan subir más niveles de los que es posible, el navegador ignora todos los `..` sobrantes. Si la página que tiene el enlace es `http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html` y la URL relativa que se incluye es `../../../../../../pagina2.html`, el navegador realmente la interpreta como `../../../../pagina2.html`.

Como el objetivo de las URL relativas es crear URL más cortas y sencillas que las URL absolutas, este método sólo se puede utilizar cuando el origen y el destino se encuentran cerca, porque de otro modo la URL relativa se complica demasiado.

3. El destino del enlace se encuentra cerca de su origen y en un nivel inferior

Este caso es muy similar al anterior, pero más sencillo. Si el recurso enlazado se encuentra en algún directorio inferior al que se encuentra el origen, sólo es necesario indicar el nombre de los directorios a los que debe entrar el navegador.

Origen	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>
Recurso enlazado	Página web llamada <code>pagina2.html</code> y que se encuentra en un directorio inferior llamado <code>ruta4</code>
URL absoluta	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/pagina2.html</code>
URL relativa	<code>ruta4/pagina2.html</code>

De la misma forma, se pueden indicar varios directorios seguidos para que el navegador descienda jerárquicamente por la estructura de directorios:

Origen	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>
Recurso enlazado	Página web llamada <code>pagina2.html</code> y que se encuentra en un directorio inferior llamado <code>ruta6</code> que está dentro del directorio <code>ruta5</code> y que a su vez está dentro del directorio <code>ruta4</code>
URL absoluta	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/ruta4/ruta5/ruta6/pagina2.html</code>
URL relativa	<code>ruta4/ruta5/ruta6/pagina2.html</code>

4. El origen y el destino del enlace se encuentran muy alejados

Cuando el origen y el destino de un enlace se encuentran muy alejados (pero en el mismo servidor) las URL relativas se pueden complicar en exceso. Aunque es posible utilizar `..` para subir por la jerarquía de directorios y se puede entrar en cualquier directorio indicando su nombre, las URL relativas que se obtienen son demasiado largas y complicadas.

En estos casos, lo más sencillo es indicar la ruta completa hasta el recurso enlazado comenzando desde la raíz del servidor web. Por lo tanto, estas URL relativas sólo omiten el protocolo y el nombre del servidor.

Origen	<code>http://www.ejemplo.com/ruta1/ruta2/ruta3/pagina1.html</code>
Recurso enlazado	Página web llamada <code>página2.html</code> y que se guarda en un directorio llamado <code>ruta7</code> que se encuentra en la raíz del servidor
URL absoluta	<code>http://www.ejemplo.com/ruta7/página2.html</code>
URL relativa	<code>/ruta7/página2.html</code>

Cuando la URL relativa comienza por `/`, el navegador considera que es la ruta completa comenzando desde la raíz del servidor, por lo que sólo le añade el protocolo y el nombre del servidor origen. A continuación se resumen los cuatro posibles casos de URL relativas y el procedimiento que sigue el navegador para convertirlas en URL absolutas:

Si la URL relativa...	El navegador la transforma en URL absoluta...
...sólo consiste en el nombre de un recurso	...añadiendo el protocolo, servidor y ruta completa del origen del enlace
...comienza porañadiendo el protocolo y servidor del origen del enlace, subiendo un nivel en la jerarquía de directorios y añadiendo el resto de la ruta incluida en la URL relativa
...comienza por /	...añadiendo el protocolo y servidor del origen del enlace
En cualquier otro caso	...añadiendo el protocolo, servidor y ruta completa del origen del enlace, a la que se añade la ruta incluida en la URL relativa

4.3 Enlaces Básicos

Los enlaces en HTML se crean mediante la etiqueta `<a>` (su nombre viene del inglés "anchor", literalmente traducido como "ancla"). A continuación se muestra la definición simplificada de `<a>` y más adelante se muestra su definición completa:

<a>	Enlaces
Atributos comunes	básicos, i18n, eventos y foco
Atributos específicos	<code>name = "texto"</code> - Permite nombrar al enlace para que se pueda acceder desde otros enlaces <code>href = "url"</code> - Indica la URL del recurso que se quiere enlazar
Tipo de elemento	En Línea
Descripción	Se emplea para enlazar todo tipo de recursos

El atributo más importante de la etiqueta `<a>` es `href`, que se utiliza para indicar la URL a la que apunta el enlace. Cuando el usuario pincha sobre un enlace, el navegador se dirige a la URL del recurso indicado mediante `href`. Las URL de los enlaces pueden ser absolutas, relativas, internas y externas.

Con la definición anterior, para crear un enlace que apunte a la página principal de Google solamente habría que incluir lo siguiente en un documento HTML:

```
<a href="http://www.google.com">Página principal de Google</a>
```

Como el atributo `href` indica una URL, un enlace puede apuntar a cualquier tipo de recurso al que pueda acceder el navegador. El siguiente enlace apunta a una imagen, que se mostrará en el navegador cuando el usuario pinche sobre el enlace:

```
<a href="http://www.ejemplo.com/fondo_escritorio.jpg">Imagen interesante para un fondo de escritorio</a>
```

De la misma forma, los enlaces pueden apuntar directamente a documentos PDF, Word, etc.

```
<a href="http://www.ejemplo.com/informe.pdf">Descargar informe completo [PDF]</a>
<a href="http://www.ejemplo.com/informe.doc">Descargar informe completo [DOC]</a>
```

Un truco muy útil con los enlaces es el uso de URL relativas para poder volver al inicio del sitio web desde cualquier página web interior:

```
<a href="/">Volver al inicio</a>
```

El enlace anterior utiliza una URL relativa con una ruta que apunta directamente a la raíz del servidor. Para obtener la URL absoluta, el navegador añade el mismo protocolo y el mismo nombre de servidor de la página en la que se encuentra el enlace. El resultado es que cuando se pincha ese enlace, siempre se vuelve al inicio del sitio web, independientemente de la página en la que se incluya el enlace.

El otro atributo básico de la etiqueta `<a>` es `name`, que permite definir enlaces dentro de una misma página web. Si una página es muy larga, puede ser útil mostrar enlaces de tipo "Saltar hasta la segunda sección", "Volver al principio de la página", etc.

Este tipo de enlaces son especiales, ya que la URL de la página siempre es la misma para todas las secciones y por tanto, debe añadirse otra parte a las URL para identificar cada sección.

(Ej04.02a-Secciones.html)

```
<a name="primera_seccion"></a>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id ligula eu felis
adipiscing ultrices. Duis gravida leo ut lectus. Praesent condimentum mattis ligula.</p>
...
<a name="segunda_seccion"></a>
<p>Pellentesque malesuada. In in lacus. Phasellus erat erat, lacinia a, convallis eu,
nonummy et, odio. Aenean urna elit, ultrices id, placerat varius, facilisis eget,
dolor.</p>
...
```

El atributo `name` permite crear "enlaces vacíos" que hacen referencia a secciones dentro de una misma página. Una vez definidos los "enlaces vacíos", es posible crear un enlace que apunte directamente a una sección concreta de una página (**Ej04.02b-Enlaces.htm**):

```
<!-- Enlace normal a la página -->
<a href="Ej04.02a-Secciones.htm">Enlace a la página 1</a>

<!-- Enlace directo a la segunda sección de la página -->
<a href="Ej04.02a-Secciones.htm#segunda_seccion">Enlace a la sección 2 de la página 1</a>
```

La sintaxis que se utiliza con estos enlaces es la misma que con los enlaces normales, salvo que se añade el símbolo `#` seguido del nombre de la sección a la que se apunta. Cuando el usuario pincha sobre uno de estos enlaces, el navegador accede a la página apuntada por la URL y baja directamente a la sección cuyo nombre se indica después del símbolo `#`.

También es posible utilizar este tipo de enlaces con URL relativas en una misma página. El siguiente ejemplo añade enlaces de tipo "Volver al inicio de la página" en varias secciones:

(Ej04.02c-URLRelativas.html)

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

<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris id ligula eu felis
adipiscing ultrices. Duis gravida leo ut lectus. Praesent condimentum mattis ligula.</p>
<a href="#inicio">Volver al inicio de la página</a>
...

<p>Pellentesque malesuada. In in lacus. Phasellus erat erat, lacinia a, convallis eu,
nonummy et, odio. Aenean urna elit, ultrices id, placerat varius, facilisis eget,
dolor.</p>
<a href="#inicio">Volver al inicio de la página</a>
...
```

Los enlaces directos a secciones también funcionan con el atributo id de cualquier elemento. El siguiente ejemplo es equivalente al ejemplo anterior:

([Ej04.02d-EnlacesID.html](#))

```
<h1 id="inicio">Título de la página</h1>

<p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Mauris id ligula eu felis
adipiscing ultrices. Duis gravida leo ut lectus. Praesent condimentum mattis ligula.</p>
<a href="#inicio">Volver al inicio de la página</a>
...
<p>Pellentesque malesuada. In in lacus. Phasellus erat erat, lacinia a, convallis eu,
nonummy et, odio. Aenean urna elit, ultrices id, placerat varius, facilisis eget,
dolor.</p>
<a href="#inicio">Volver al inicio de la página</a>
...
```

El nombre de la sección que se indica después del símbolo # puede utilizar el valor de los atributos id de cualquier elemento. De hecho, se recomienda utilizar los atributos id de los elementos ya existentes en la página en vez de crear "enlaces vacíos" de tipo

```
<a name="nombre_seccion"></a>.
```

Ejercicio 06

A partir de la estructura de directorios y archivos indicada a la derecha →

a) Crear la siguiente página llamada indice.html:

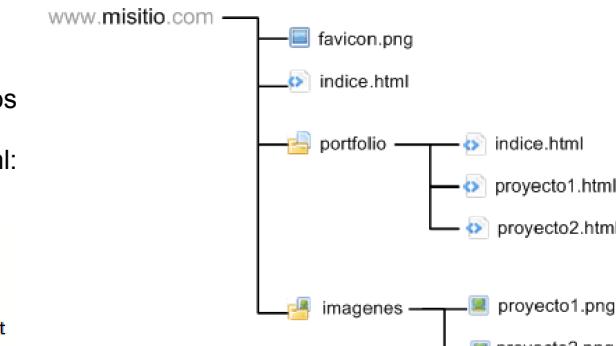
Mi Sitio

Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat. Aliquam dui ligula, porttitor eu, facilisis vitae, ornare sed, tortor.

Ultimos proyectos

Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat. Aliquam dui ligula, porttitor eu, facilisis vitae, ornare sed, tortor.

[Acceder a los últimos proyectos de Mi Sitio](#)



b) El último enlace inferior apuntará a la siguiente URL

<http://www.misitio.com/porfolio/indice.html>
(si lo deseamos podemos renombrar dicha página a porfolio.html):

NOTA: Se recomienda el uso del generador
Lorem Ipsum:
<http://es.lipsum.com/>

[Volver a la pagina principal](#)

Ultimos proyectos

Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat.

Proyecto 1

Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat.

[Ver imagen del Proyecto 1](#)

Proyecto 2

Etiam consectetuer, mauris vitae cursus scelerisque, dui turpis dignissim justo, et euismod enim odio sit amet erat.

[Ver imagen del Proyecto 2](#)

4.4 Enlaces Avanzados

Incluir enlaces básicos mediante la etiqueta `<a>` es muy sencillo. Sin embargo, la definición completa de `<a>` es muy compleja, ya que dispone de varios atributos específicos importantes. A continuación se muestra la definición completa de `<a>`:

<code><a></code>	Enlaces
Atributos comunes	básicos, i18n, eventos y foco
Atributos específicos	<code>name = "texto"</code> - Permite nombrar al enlace para que se pueda acceder desde otros enlaces <code>href = "url"</code> - Indica la URL del recurso que se quiere enlazar <code>hreflang = "codigo_idioma"</code> - Idioma del recurso enlazado <code>type = "tipo_de_contenido"</code> - Permite "avisar" al navegador sobre el tipo de contenido que se enlaza (imágenes, archivos, etc.) para que pueda prepararse en caso de que no entienda ese contenido <code>rel = "tipo_de_relacion"</code> - Describe la relación del documento actual con el recurso enlazado <code>rev = "tipo_de_relacion"</code> - Describe la relación del recurso enlazado con el documento actual <code>charset = "tipo_de_charset"</code> - Describe la codificación del recurso enlazado
Tipo de elemento	En Línea
Descripción	Se emplea para enlazar todo tipo de recursos

4.4.1 Idioma del enlace (hreflang)

El enlace puede indicar al navegador el idioma del recurso que se enlaza. Para establecer el valor del idioma, se utiliza un código estandarizado de dos letras. Además del idioma genérico, también se puede indicar una variación idiomática. Ejemplo de códigos de idioma más utilizados:

en (inglés), en-US (inglés – USA), es (español), es-AR (español, Argentina), es-ES (español, España), fr (francés), de (alemán), it (italiano), nl (holandés), el (griego), pt (portugués), ar (árabe), he (hebreo), ru (ruso), zh (chino), ja (japonés).

La lista completa de códigos de idioma está definida en el estándar [ISO 639](#).

4.4.2 Tipo de contenido (type)

Se utiliza para notificar al navegador sobre el tipo de contenido que se enlaza. Se indica mediante una cadena de texto cuyos posibles valores también están estandarizados. Los valores de los contenidos más utilizados son los siguientes: "text/html" (páginas HTML), "image/png" (imágenes con formato PNG), "image/gif" (imágenes con formato GIF), "text/css" (hojas de estilo CSS), "application/rss+xml" (archivos RSS).

La lista completa de tipos de contenido se define en los estándares [RFC 2045 y RFC 2046](#).

4.4.3 Tipo de relación (rel y rev)

Los enlaces pueden proporcionar información adicional muy útil para los navegadores y para los motores de búsqueda como Google. Los atributos rel y rev permiten indicar la relación que la página actual tiene con la página a la que se enlaza (atributo rel) y la relación que tiene la página enlazada con la página actual (atributo rev).

Los tipos de relación definidos son los siguientes:

- `alternate` – Indica que es una versión alternativa al documento actual (puede ser una versión en otro idioma o una versión preparada para otro medio, como una impresora o un dispositivo móvil)
- `stylesheet` – Indica que se ha enlazado una hoja de estilos
- `start` – Indica que se trata del primer documento de una colección de documentos (por ejemplo el primer capítulo de un libro)
- `next` – Indica que es el documento que sigue al actual dentro de una secuencia lógica de documentos (por ejemplo, los capítulos de un libro)
- `prev` – Indica que es el documento que precede al actual dentro de una secuencia lógica de documentos (por ejemplo, los capítulos de un libro)
- `contents` – Indica que el recurso enlazado es el documento que contiene la tabla de contenidos de la colección de documentos (por ejemplo, el índice de un libro).
- `bookmark` – Establece el enlace actual como un "marcador" o "favorito". Un marcador es un enlace que constituye un punto de entrada muy importante dentro del documento.

La [especificación oficial de HTML](#) define la lista completa de tipos de relaciones usables.

4.4.4 Codificación de caracteres (charset)

Además del idioma, tipo de contenido y relación del recurso que se enlaza, los enlaces también pueden indicar la codificación de caracteres que utiliza la página web enlazada.

Los valores que se pueden utilizar también están estandarizados y las codificaciones más utilizadas son UTF-8 y ISO-8859-1, aunque existen decenas de códigos definidos (ISO-10646-UCS-2, IBM852, Big5-HKSCS, windows-1252, HZ-GB-2312).

El organismo IANA publica la [lista completa de codificaciones](#) de caracteres disponibles.

Los ejemplos anteriores de enlaces básicos se pueden rehacer utilizando algunos de los atributos definidos por la etiqueta `<a>`:

```
<a href="http://www.google.com" hreflang="en" type="text/html" charset="UTF-8">
    Página principal de Google</a>

<a href="http://www.ejemplo.com/fondo_escritorio.jpg" type="image/jpg">
    Imagen interesante para un fondo de escritorio</a>
```

4.5 Otros tipos de Enlaces

Los enlaces mostrados en las secciones anteriores son los más utilizados por las páginas web. Los enlaces creados con la etiqueta `<a>` permiten enlazar cualquier tipo de recurso desde cualquier página. La característica más importante de estos enlaces es que el usuario debe activar la carga de los recursos. En otras palabras, el navegador no carga ningún recurso enlazado con la etiqueta `<a>` a menos que el usuario pinche sobre el enlace.

Además de estos enlaces, las páginas HTML pueden incluir otro tipo de enlaces que cargan los recursos automáticamente. Si una página HTML utiliza archivos CSS para aplicar estilos a sus contenidos, no es lógico que los enlace con la etiqueta `<a>` y espere a que el usuario pinche sobre el enlace para así cargar los archivos CSS. De la misma forma, muchas páginas web dinámicas necesitan que el navegador cargue varios archivos JavaScript para funcionar correctamente.

HTML define dos etiquetas para enlazar recursos que se deben cargar automáticamente:

- `<script>`
- `<link>`

Cuando el navegador encuentra alguna de estas dos etiquetas, descarga los recursos enlazados y los aplica a la página web.

La etiqueta `<script>` tiene dos modos de funcionamiento, ya que se emplea tanto para insertar un bloque de código JavaScript en la página como para enlazar un archivo JavaScript externo.

<code><script></code>	Código ejecutable
Atributos comunes	-
Atributos específicos	<code>src = "url"</code> – Indica la dirección del archivo que contiene el código <code>type = "tipo_de_contenido"</code> – Permite "avisar" al navegador sobre el tipo de código que se incluye (normalmente JavaScript) <code>defer = "defer"</code> – El código no va a modificar el contenido de la página web <code>charset = "tipo_de_charset"</code> – Describe la codificación del código enlazado
Tipo de elemento	Bloque y en línea (también puede ser una etiqueta vacía)
Descripción	Se usa para enlazar o definir un bloque de código (normalmente JavaScript)

Aunque la etiqueta `<script>` permite enlazar código de varios lenguajes de programación, el uso habitual de `<script>` consiste en enlazar un archivo JavaScript externo:

```
<head>
  <script type="text/javascript" src="http://www.ejemplo.com/js/inicializar.js"></script>
</head>
```

El atributo `type` utilizado habitualmente para los archivos JavaScript es "text/javascript". El atributo `src` es equivalente al atributo `href` de los enlaces creados con la etiqueta `<a>`. La URL indicada en el atributo `src` puede ser absoluta o relativa y externa o interna.

Además de enlazar un archivo JavaScript externo, la misma etiqueta `<script>` también permite incluir en la página web un bloque de código JavaScript.

Veamos un ejemplo ([Ej04.05-Script.html](#)):

```
<html>
  <head>
    <meta charset="UTF-8" />
    <script type="text/javascript">
      //<![CDATA[
        window.onload = function() {
          alert("La página se ha cargado completamente");
        }
      //]]&gt;
    &lt;/script&gt;
  &lt;/head&gt;
&lt;body&gt;
...
&lt;/body&gt;
&lt;/html&gt;</pre>
```

Cuando se incluye código JavaScript en la propia página XHTML, se debe insertar dentro de una sección especial llamada CDATA. Para ello, el código JavaScript se debe encerrar entre `<![CDATA[y]]>`. Cuando el navegador encuentra una sección de este tipo, no procesa su contenido como si fuera XHTML y por tanto no tiene en cuenta los posibles errores de validación de XHTML.

De esta forma, se pueden construir páginas XHTML válidas y código JavaScript correcto. En los capítulos posteriores se profundiza en el concepto de validación de páginas XHTML. Los caracteres `//` al comienzo y al final de la sección CDATA son necesarios para los navegadores que no son capaces de procesar correctamente estas secciones.

La etiqueta `<script>` (tanto cuando enlaza, como cuando incluye directamente el código) puede aparecer en cualquier parte del documento HTML, aunque normalmente se incluye dentro de la cabecera de la página (`<head>...</head>`).

La segunda etiqueta de XHTML para enlazar recursos es `<link>`, que permite enlazar y relacionar la página con otros recursos externos.

<link>	Enlazar recursos
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<ul style="list-style-type: none">Los siguientes con el mismo significado que para la etiqueta "a": <code>charset</code>, <code>href</code>, <code>hreflang</code>, <code>type</code>, <code>rel</code> y <code>rev</code><code>media = "tipo_de_medio"</code> - Indica el medio para el que debe aplicarse la relación
Tipo de elemento	Etiqueta vacía
Descripción	Enlaza y establece relaciones entre el documento y otros recursos

Al contrario que `<script>`, la etiqueta `<link>` solamente se puede incluir dentro de la cabecera del documento. Se pueden añadir tantas etiquetas `<link>` como sean necesarias, pero siempre dentro de `<head>...</head>`.

El atributo `media` hace referencia al medio para el que es válida la relación con el recurso enlazado. Los medios disponibles también están estandarizados, siendo los más comunes `screen` para los contenidos mostrados en pantalla, `print` para las impresoras y `handheld` para los dispositivos móviles.

El uso habitual de la etiqueta <link> es el de enlazar las hojas de estilos CSS utilizadas por las páginas web:

```
<head>
  ...
  <link rel="stylesheet" type="text/css" href="/css/comun.css" />
</head>
```

En este caso, es habitual establecer los atributos rel y type para indicar el tipo de recurso enlazado y su relación con la página web. La URL del recurso enlazado se indica en el atributo href, que admite tanto URL absolutas como relativas.

4.6 Ejemplos de enlaces habituales

4.6.1 Enlace al inicio del sitio web

```
<a href="/">Inicio</a>
```

Al pulsar el enlace anterior desde cualquier página web, se vuelve directamente a la página de inicio, home o página principal del sitio web.

4.6.2 Enlace a un email

```
<a href="mailto:nombre@direccion.com"
  title="Dirección de email para solicitar más información">
  Sigue la dirección de correo electrónico </a>
```

Al pinchar sobre el enlace anterior, se abre automáticamente el programa de correo electrónico del ordenador del usuario y se establece la dirección de envío al valor indicado después de mailto: La sintaxis es la misma que la de un enlace normal, salvo que se cambia el prefijo http:// por mailto:

La sintaxis de mailto: permite utilizarlo para otros ejemplos más complejos:

```
<!-- Envío del correo electrónico a varias direcciones a la vez -->
<a href="mailto:nombre@direccion.com, otro_nombre@direccion.com">
  Sigue la dirección de correo electrónico </a>

<!-- Añadir un "asunto" inicial al correo electrónico -->
<a href="mailto:nombre@direccion.com?subject=Solicitud de más información">
  Sigue la dirección de correo electrónico </a>

<!-- Añadir un texto inicial en el cuerpo del correo electrónico -->
<a href="mailto:nombre@direccion.com?body=Estaría interesado en solicitar más información
  sobre sus productos">Sigue la dirección de correo electrónico </a>
```

Todas las opciones anteriores se pueden combinar entre sí para realizar ejemplos más avanzados. Aunque el uso de mailto: puede parecer una ventaja, su uso está desaconsejado. Si se incluye una dirección de correo electrónico directamente en una página web, es muy probable que en poco tiempo esa dirección de email se encuentre llena de correo electrónico basura o "spam", ya que existen programas automáticos encargados de rastrear sistemáticamente todas las páginas web de Internet para encontrar direcciones de correo electrónico válidas.

La forma de mostrar las direcciones de correo electrónico en las páginas web consiste en incluir la dirección en una imagen o indicarla de forma que solamente los usuarios puedan entenderlo:

```
<p>La dirección de correo es <strong>nombre (arroba) direccion.com</strong></p>
<p>La dirección de correo es <strong>nombre_arroba_direccion.com</strong></p>
<p>La dirección de correo es <strong>nombreQUITAESTO@direccion.com</strong></p>
<p>La dirección de correo es <strong>nombre(ARROBA)direccion.com</strong></p>
<p>La dirección de correo es <strong>nombre @ direccion . com</strong></p>
```

4.6.3 Enlace a un archivo FTP

Para enlazar un archivo almacenado en un servidor FTP, la parte del protocolo de la URL debe cambiar de `http:*` a `ftp:*`:

```
<a href="ftp://ftp.ejemplo.com/ruta/archivo.zip" title="Archivo comprimido de los contenidos"> Descarga un ZIP con todos los contenidos </a>
```

4.6.4 Enlazar varias hojas de estilos CSS

La forma de hacerlo es poner una etiqueta `<link>` detrás de otra:

```
<link rel="stylesheet" type="text/css" href="/css/comun.css" />
<link rel="stylesheet" type="text/css" href="/css/secciones.css" />
```

4.6.5 Enlazar hojas de estilos CSS para diferentes medios

De nuevo sería añadiendo tantas etiquetas `<link>` como recursos css:

```
<link rel="stylesheet" type="text/css" href="/css/comun.css" media="screen, projection" />
<link rel="stylesheet" type="text/css" href="/css/impresora.css" media="print" />
<link rel="stylesheet" type="text/css" href="/css/movil.css" media="handheld" />
```

4.6.6 Enlazar el favicon

El favicon o icono para favoritos es el pequeño ícono que muestran las páginas en varias partes del navegador. Dependiendo del navegador que se utilice, este ícono se muestra en la barra de direcciones, en la barra de título del navegador y/o en el menú de favoritos/marcadores.

```
<link rel="shortcut icon" href="/favicon.ico" type="image/ico" />
```

Las características del favicon serán: La imagen debe medir 16 x 16 px; 16 colores (4 bits); Formato ICO, no BMP, no GIF, no JPG.

Algunos navegadores permiten iconos más grandes, una gama cromática más amplia y otros formatos. Lo cierto es que si quieras con certeza que el favicon se vea en todos deberá seguir las características expuestas, 16x16 px, con 16 colores y en formato ICO. Podemos generar un favicon online en páginas como esta: <http://tools.dynamicdrive.com/favicon/>

4.6.7 Enlazar un archivo RSS

RSS son las siglas de Really Simple Syndication, un formato XML para indicar o compartir contenido en la web. Se utiliza para difundir información actualizada frecuentemente a usuarios que se han suscrito a la fuente de contenidos.

```
<link rel="alternate" type="application/rss+xml" title="Resumen de todos los artículos del blog" href="/feed.xml" />
```

4.6.8 Enlazar hojas de estilos, favicon y RSS

En una misma página se pueden incluir varias etiquetas `<link>`, por lo que es habitual que las páginas enlacen hojas de estilos, favicon y archivos RSS de forma conjunta:

```
<head>
...
<link rel="stylesheet" type="text/css" href="/css/impresora.css" media="print" />
<link rel="stylesheet" type="text/css" href="/css/movil.css" media="handheld" />
<style type="text/css" media="screen,projection">
  @import '/css/main.css';
</style>
<link rel="shortcut icon" href="/favicon.ico" type="image/ico" />
<link rel="alternate" type="application/rss+xml"
      title="Resumen de todos los artículos del blog" href="/feed.xml" />
...
</head>
```

4.6.9 Indicar que existe una versión de la página en otro idioma

```
<head>
  <title>English tutorial</title>
  <link lang="es" xml:lang="es" title="El tutorial en español" type="text/html"
    rel="alternate" hreflang="es" href="http://www.ejemplo.com/tutorial/espanol.html" />
</head>
```

4.6.10 Indicar que existe una versión de la página preparada para imprimir

```
<head>
  <link media="print" title="El tutorial en PDF" type="application/pdf"
    rel="alternate" href="http://www.ejemplo.com/tutorial/documento.pdf" />
</head>
```

4.6.11 Indicar que existe una página que es índice de la página actual

```
<head>
  <title>Tutorial - Capítulo 5</title>
  <link rel="start" title="El indice del tutorial" type="text/html"
    href="http://www.ejemplo.com/tutorial/indice.html" />
</head>
```

Ejercicio 07

Enlazar el favicon en todas las páginas del ejercicio 6 y añadir todos los atributos posibles a los enlaces.

NOTA: Generadores de favicon Online hay muchos y gratuitos. Algunos son...

<http://www.genfavicon.com/es/>

<http://www/favicon.cc/>

5 LISTAS

En ocasiones, es posible agrupar determinadas palabras o frases en un conjunto de elementos que tienen más significado de forma conjunta. El menú de navegación de un sitio web por ejemplo está formado por un grupo de palabras. Aunque cada palabra por separado tiene sentido, de forma conjunta constituyen el menú de navegación de la página, por lo que su significado conjunto es mayor que por separado.

El lenguaje HTML define tres tipos diferentes de listas para agrupar los elementos: listas no ordenadas (se trata de una colección simple de elementos en la que no importa su orden), listas ordenadas (similar a la anterior, pero los elementos están numerados y por tanto, importa su orden) y listas de definición (un conjunto de términos y definiciones similar a un diccionario).

5.1 Listas No Ordenadas

Las listas no ordenadas son las más sencillas y las que más se utilizan. Una lista no ordenada es un conjunto de elementos relacionados entre sí pero para los que no se indica un orden o secuencia determinados. La etiqueta `` encierra todos los elementos de la lista y la etiqueta `` cada uno de sus elementos.

<code></code>	Lista no ordenada
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir listas no ordenadas

<code></code>	Elemento de una lista
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Define los elementos de las listas (ordenadas y no ordenadas)

Veamos un ejemplo ([Ej05.01a-ListaNoOrdenada.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta UL</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <h1>Menú</h1>
    <ul>
      <li>Inicio</li>
      <li>Noticias</li>
      <li>Artículos</li>
      <li>Contacto</li>
    </ul>
  </body>
</html>
```

El navegador por defecto muestra los elementos de la lista tabulados y con una pequeña viñeta formada por un círculo negro. Como ya se sabe, el aspecto con el que se muestran los elementos de las listas se puede modificar mediante las hojas de estilos CSS.

De todos modos se puede indicar el tipo de viñeta a presentar con el atributo `type` (**¡marcado como obsoleto!**) y los valores `disc` (predefinido), `circle` (círculo vacío) o `square` (cuadrado negro).

5.2 Listas Ordenadas

Las listas ordenadas son casi idénticas a las listas no ordenadas, salvo que en este caso los elementos relacionados se muestran siguiendo un orden determinado. Cuando se crea por ejemplo una lista con las instrucciones de un producto, es importante el orden en el que se realiza cada paso. Cuando se muestra un índice o tabla de contenidos en un libro, es importante el orden de cada elemento del índice.

En todos estos casos, la lista más adecuada es la lista ordenada, que se define mediante la etiqueta ``. Los elementos de la lista se definen mediante la etiqueta ``, la misma que se utiliza en las listas no ordenadas.

<code></code>	Lista ordenada
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir listas ordenadas

Veamos un ejemplo ([Ej05.02a-ListaOrdenada.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta OL</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <h1>Instrucciones</h1>
    <ol>
      <li>Enchufar correctamente</li>
      <li>Comprobar conexiones</li>
      <li>Encender el aparato</li>
    </ol>
  </body>
</html>
```

El navegador muestra la lista de forma muy parecida a las listas no ordenadas, salvo que en este caso no se emplean viñetas gráficas en los elementos, sino que se numeran de forma consecutiva. El tipo de numeración empleada también se puede modificar aplicando hojas de estilos CSS a los elementos de la lista.

Por otra parte, aunque esté obsoleto, se puede indicar el tipo de numeración y el inicio del mismo mediante los siguientes atributos:

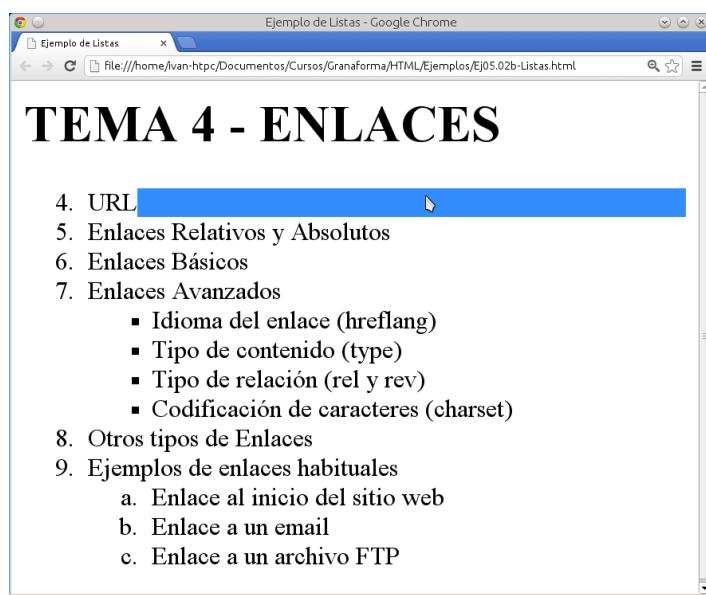
- `type`, cuyos valores pueden ser `i`, `I` (números romanos), `a`, `A` (alfabéticos), `1`,
- `start`, que indica el inicio de la ordenación (el valor será siempre numérico).

Estas listas pueden ser anidadas, es decir, contener o estar contenidas por otras listas. De hecho, la versatilidad es tal, que también pueden anidarse con listas desordenadas (elemento HTML `ul`).

Para lograr un código correcto, las sub-listas deben ser insertadas dentro de los ítems (elemento HTML `li`) y no directamente como contenido de la lista padre.

Veamos un ejemplo ([Ej05.02b-Listas.html](#)):

```
<html>
<head>
    <title>Ejemplo de Listas</title>
    <meta charset="UTF-8" />
</head>
<body>
    <h1>TEMA 4 - ENLACES</h1>
    <ol start="4">
        <li> URL</li>
        <li> Enlaces Relativos y Absolutos</li>
        <li> Enlaces Básicos</li>
        <li> Enlaces Avanzados</li>
        <ul type="square">
            <li> Idioma del enlace </li>
            <li> Tipo de contenido </li>
            <li> Tipo de relación </li>
            <li> Codificación de caracteres </li>
        </ul>
        <li> Otros tipos de Enlaces</li>
        <li> Ejemplos de enlaces habituales</li>
        <ol type="a">
            <li> Enlace al inicio del sitio web</li>
            <li> Enlace a un email</li>
            <li> Enlace a un archivo FTP</li>
        </ol>
    </ol>
</body>
</html>
```



Ejercicio 08

Determinar el código HTML que corresponde a la siguiente lista anidada simple.

Menú

- Inicio
- Noticias
 - [Recientes](#)
 - [Más leídas](#)
 - [Más valoradas](#)
- Artículos
 - 1. [XHTML](#)
 - 2. CSS
 - 3. JavaScript
 - 4. Otros
- Contacto

Email
nombre@direccion.com
Teléfono
900 900 900
Fax
900 900 900

Menú

- [Inicio](#)
- [Noticias](#)
 - [Recientes](#)
 - [Más leídas](#)
 - [Más valoradas](#)

Ejercicio 09

Determinar el código HTML que corresponde a la siguiente lista anidada compleja

5.3 Listas de Definición

Las listas de definición apenas se utilizan en la mayoría de páginas HTML. Su funcionamiento es similar al de un diccionario, ya que cada elemento de la lista está formado por términos y definiciones. La etiqueta `<dl>` crea la lista de definición y las etiquetas `<dt>` y `<dd>` definen respectivamente el término y la descripción de cada elemento de la lista.

<code><dl></code>	Lista de definición
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir listas de definición

<code><dt></code>	Término de una definición
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Define los términos de los elementos de una lista de definición

<code><dd></code>	Descripción de una definición
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Indica las definiciones de los elementos de una lista de definición

Veamos un ejemplo ([Ej05.03-Definicion.html](#)):

```
<html>
  <head>
    <title>Ejemplo de etiqueta DL</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <h1>Metalenguajes</h1>
    <dl>
      <dt>SGML</dt>
      <dd>Metalenguaje para la definición de otros lenguajes de marcado</dd>
      <dt>XML</dt>
      <dd>Lenguaje basado en SGML y que se emplea para describir datos</dd>
      <dt>RSS</dt>
      <dt>GML</dt>
      <dt>XHTML</dt>
      <dt>SVG</dt>
      <dt>XUL</dt>
      <dd>Lenguajes derivados de XML para determinadas aplicaciones</dd>
    </dl>
  </body>
</html>
```

Los navegadores formatean las listas de definición de forma similar a las otras listas, tabulando la definición y alineando a la izquierda los términos. Aunque no es habitual, cada término puede tener asociada más de una definición y cada definición puede tener asociada varios términos.

6 IMÁGENES Y OBJETOS

6.1 Imágenes

Las imágenes son uno de los elementos más importantes de las páginas web. De hecho, prácticamente todas las páginas web contienen alguna imagen y la mayoría incluyen decenas de imágenes. Dentro de las imágenes que se pueden incluir en una página HTML se deben distinguir dos tipos: las imágenes de contenido y las imágenes de adorno.

Las imágenes de contenido son las que proporcionan información y complementan la información textual. Las imágenes de adorno son las que se utilizan para hacer bordes redondeados, para mostrar pequeños iconos en las listas de elementos, para mostrar fondos de página, etc. Las imágenes de contenido se incluyen directamente en el código HTML mediante la etiqueta `` y las imágenes de adorno no se deberían incluir en el código HTML, sino que deberían emplearse hojas de estilos CSS para mostrarlas.

A continuación se muestra la definición de la etiqueta ``:

	Imagen
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<code>src = "url"</code> - Indica la URL de la imagen que se muestra <code>alt = "texto"</code> - Descripción corta de la imagen <code>longdesc = "url"</code> - Indica una URL en la que puede encontrarse una descripción más detallada de la imagen <code>name = "texto"</code> - Nombre del elemento imagen <code>height = "unidad_de_medida"</code> - Indica la altura con la que se debe mostrar la imagen (no es obligatorio que coincida con la altura original de la imagen) <code>width = "unidad_de_medida"</code> - Indica la anchura con la que se debe mostrar la imagen (no es obligatorio que coincida con la anchura original de la imagen)
Tipo de elemento	En línea y etiqueta vacía
Descripción	Se emplea para incluir imágenes en los documentos

Los dos atributos requeridos son `src` y `alt`. El atributo `src` es similar al atributo `href` de los enlaces, ya que establece la URL de la imagen que se va a mostrar en la página. Las URL indicadas pueden ser absolutas o relativas. El atributo `alt` permite describir el contenido de la imagen mediante un texto breve. Las descripciones deben tener una longitud inferior a 1024 caracteres y son útiles para las personas y dispositivos discapacitados que no pueden acceder a las imágenes.

Ejemplo sencillo para incluir una imagen (la etiqueta debe cerrarse con `/>`):

```

```

HTML no impone ninguna restricción sobre el formato gráfico que se puede utilizar en las imágenes, por lo que en principio la etiqueta `` puede incluir cualquier formato gráfico existente. Sin embargo, si la imagen utiliza un formato poco habitual, todos o algunos navegadores no serán capaces de mostrar esa imagen.

La recomendación es utilizar uno de los tres siguientes formatos gráficos que entienden todos los navegadores modernos: GIF, JPG y PNG. El formato PNG presenta el inconveniente de que los navegadores obsoletos como Internet Explorer 6 no muestran correctamente las imágenes con transparencias de 24 bits.

El atributo `longdesc` no se utiliza de forma habitual, pero permite indicar la URL en la que se puede encontrar más información sobre la imagen. Como el atributo `alt` sólo permite incluir descripciones de hasta 1024 caracteres, el atributo `longdesc` se emplea con las imágenes complejas que necesitan mucha información para ser descritas:

```


```

En el ejemplo anterior, las dos imágenes se encuentran en el mismo directorio del servidor (`/imagenes/`). Se trata de una estrategia habitual en la mayoría de sitios web: guardar todas las imágenes de contenido en un directorio especial independiente del resto de contenidos HTML. Además, el directorio siempre suele llamarse de la misma manera: "imagenes" o "images" en inglés.

Los atributos `width` y `height` se utilizan para indicar la anchura y altura con la que se muestran las imágenes, por lo que son los más contradictorios. Como ya se ha comentado, HTML estructura de forma correcta los contenidos de la página y CSS define el aspecto gráfico con el que se muestran los contenidos. En principio, la anchura y la altura con la que se muestra una imagen es parte de su aspecto gráfico, por lo que debería ser propio de CSS y no de XHTML.

Sin embargo, en la práctica no es viable establecer la anchura y altura de todas las imágenes de contenidos mediante CSS. Si el sitio web dispone de muchas imágenes, la sobrecarga de estilos diferentes que debería definir CSS sería contraproducente. Por este motivo, los atributos `width` y `height` son la excepción a la norma de que el código HTML no haga referencia al aspecto de los contenidos.

```


```

Si el valor del atributo `width` o `height` se indica mediante un número entero, el navegador supone que hace referencia a la unidad de medida píxel. Por tanto, en el ejemplo anterior, la primera foto se muestra con una anchura de 200 píxel y una altura de 350 píxel.

También es posible indicar la anchura y altura en forma de porcentaje. En este caso, el porcentaje hace referencia a la altura/anchura del elemento en el que está contenida la imagen. Si la imagen no se encuentra dentro de ningún otro elemento, hace referencia a la anchura/altura total de la página.

```
<div>
  
</div>
```

El ejemplo anterior mezcla los dos tipos de medidas que se pueden utilizar, para indicar que la foto tiene una anchura igual al 30% de la anchura del elemento `<div>` que la contiene y una altura de 350 píxel.

La anchura/altura con la que se muestra una imagen no tiene que coincidir obligatoriamente con la anchura/altura real de la imagen. Sin embargo, cuando estos valores no coinciden, las imágenes se muestran deformadas y el aspecto final es muy desagradable.

Si solamente se establece la altura de la imagen, el navegador calcula la anchura necesaria para que se mantenga la proporción de la imagen. De la misma forma, si sólo se establece la anchura de la imagen, el navegador calcula la altura que hace que la imagen se siga viendo con las mismas proporciones.

Además de los atributos anteriores, existen otros que han sido catalogados como obsoletos en el estándar HTML 4.01 y por extensión en XHTML pero que debemos conocer:

- **align:** Especifica la posición de la imagen respecto del contenido a su alrededor. Puede tomar uno de los siguientes valores (insensibles a mayúsculas/minúsculas):
 - **top:** el lado superior de la imagen es alineado verticalmente con la línea base.
 - **middle:** la imagen es centrada verticalmente respecto de la línea base.
 - **bottom:** el lado inferior de la imagen es alineado verticalmente con la línea base. Este es el valor por defecto.
 - **left:** la imagen flota a la margen izquierda.
 - **right:** la imagen flota a la margen derecha.
- **border:** Este atributo establece el ancho del borde en número de píxeles.
``
- **hspace:** Establece la cantidad de espacio en blanco (en píxeles) que será insertado como márgenes izquierdo y derecho de la imagen.
``
- **vspace:** Establece la cantidad de espacio en blanco (en píxeles) que será insertado como márgenes superior e inferior de la imagen.
``

Bancos de Imágenes

En internet tenemos multitud de bancos de imágenes gratuitos. Aquí va una lista de enlaces a los mas interesantes:

- <http://search.creativecommons.org/>
- <http://search.creativecommons.org/>
- <http://www.stockvault.net/>
- <http://flickrcc.bluemountains.net>
- <http://www.sxc.hu/>
- <http://www.morguefile.com/>
- <http://www.openphoto.net/>
- <http://www.photorack.net/>
- <http://www.unprofound.com/>
- <http://www.freedigitalphotos.net/>
- <http://www.freefoto.com>
- <http://www.freepixels.com/>
- <http://www.designpacks.com/>
- <http://www.everystockphoto.com/>
- <http://recursostic.educacion.es/bancoimagenes/web/>

Veamos un ejemplo ([Ej06.01-Imagen.html](#)):

NOTA: Teneis otra imagen aquí: <http://www.flickr.com/photos/62223880@N00/145129170>

```
<html>
  <head>    <title>Ejemplo de Imagen</title>
  <meta charset="UTF-8" /></head>
<body>
  <p> Ejemplo de Imagen </p><br><br>
  <a href="http://www.nasa.gov">
    
  </a>
</body>
</html>
```

Ejercicio 10

Estructurar y marcar el siguiente texto extraído de la Wikipedia (http://es.wikipedia.org/wiki/Exploración_espacial) para que el navegador lo muestre con el aspecto de la siguiente imagen:

Programa Apolo

El **Programa Apolo** comenzó en [julio de 1960](#) cuando la [NASA](#) anunció un proyecto, continuación de las misiones [Mercury](#), que tendría como objetivo el sobrevuelo tripulado de nuestro satélite para localizar una zona apropiada con vistas a un eventual [alunizaje de astronautas](#); se cumpliría así el viejo sueño del [viaje a la Luna](#) por parte del ser humano. Pero los planes iniciales se vieron modificados en [1961](#) con el anuncio del presidente [John F. Kennedy](#) de enviar y depositar un hombre en la [Luna](#), y traerlo de vuelta a salvo antes de que finalizara la década. La meta se alcanzó con 17 meses de sobra cuando el [20 de julio de 1969 Neil Armstrong y Edwin Buzz Aldrin](#) a bordo de la [Apolo 11](#) alunizaron en el [Mar de la Tranquilidad](#). Este hito histórico se retransmitió a todo el [planeta](#) desde las instalaciones del Observatorio Parkes ([Australia](#)). Inicialmente el paseo lunar iba a ser retransmitido a partir de la señal que llegase a la estación de seguimiento de Goldstone (California, Estados Unidos), perteneciente a la [Red del Espacio Profundo](#), pero ante la mala recepción de la señal se optó por utilizar la señal de la estación Honeysuckle Creek, cercana a Canberra ([Australia](#)).¹ Ésta retransmitió los primeros minutos del paseo lunar, tras los cuales la señal del Observatorio Parkes fue utilizada de nuevo durante el resto del paseo lunar.² Las instalaciones del [MDSCC](#) en [Robledo de Chavela \(Madrid, España\)](#) también pertenecientes a la [Red del Espacio Profundo](#), sirvieron de apoyo durante todo el viaje de ida y vuelta.^{3 4}



Insignia del programa Apolo. [\[editar\]](#)

El Proyecto Apolo fue uno de los triunfos más importantes de la tecnología moderna. Seis misiones lograron posarse sobre la superficie lunar ([Apolo 11, 12, 14, 15, 16 y 17](#)) con un solo fallo: la misión [Apolo 13](#) no pudo concretar su meta por la explosión del tanque de [oxígeno](#) líquido del módulo de servicio, pero la tripulación regresó a salvo. Previamente a las misiones con descenso proyectado a la superficie de la Luna, se probaron los sistemas de vuelo en varios lanzamientos automáticos (ver [Apolo 2, 3, 4, 5 y 6](#)), y después hubo dos pruebas tripuladas en órbita terrestre ([Apolo 7 y 9](#)), y dos misiones sólo orbitales (sin alunizaje) a la Luna ([Apolo 8 y 10](#)). En [1973](#), una vez finalizado el programa lunar, tres naves Apolo fueron usadas para enviar tripulaciones a la estación espacial [Skylab](#) (misiones SL-2, SL-3 y SL-4) y en [1975](#) fue lanzada la última nave Apolo, para la misión [Apolo-Soyuz](#).

Otra de las novedades de este programa fue la implementación de un sistema de encuentro y acople con otra nave en órbita lunar, bautizado *Lunar Orbit Rendezvous* o [LOR](#) («Encuentro de Órbita Lunar»), que fuera ideado por [John C. Houbolt](#), un ingeniero espacial de la NASA. A pesar de los riesgos que implicaba su uso, el LOR permitió a la NASA reemplazar el descomunal cohete «[NOVA](#)» originalmente planeado para este tipo de misiones, lo cual llevó a un significativo ahorro de dinero.

En especial, debemos incluir imágenes...

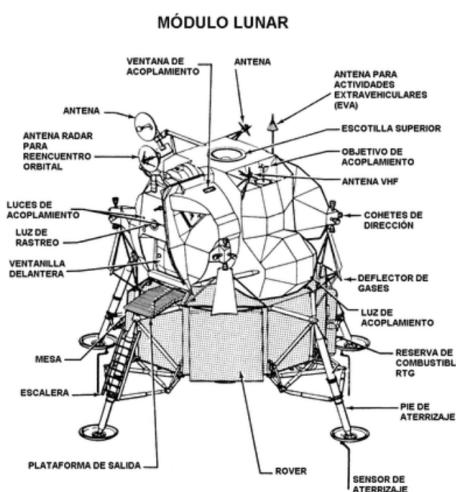
NOTA: Se recomienda copiar y pegar el texto de la página.

Descripción de la nave Apolo

[\[editar\]](#)

Artículo principal: [Nave Apolo](#).

El [módulo lunar](#) (LEM) Apolo fue la primera nave diseñada para volar en el [vacío](#) sin ninguna capacidad aerodinámica. El módulo estaba unido al módulo de mando y al módulo de servicio, y se separaba de éstos en la órbita lunar para emprender su descenso a la Luna con dos astronautas a bordo. Tenía unas patas tan débiles que no podían cargar el peso del módulo en gravedad terrestre, pero sí en la lunar (aproximadamente un sexto de la anterior). Al final de su estadía en la superficie, la etapa superior del módulo lunar despegaba para volver a unirse a los dos módulos en órbita lunar.



6.2 Mapas de Imagen

Aunque el uso de los mapas de imagen se ha reducido drásticamente en los últimos años, aún se utilizan en algunos sitios especializados. Muchas agencias de viaje y sitios relacionados utilizan mapas geográficos para seleccionar el destino del viaje. La mayoría de mapas se realiza hoy en día mediante Flash, aunque algunos sitios siguen recurriendo a los mapas de imagen.

Un mapa de imagen permite definir diferentes zonas "pinchables" dentro de una imagen. El usuario puede pinchar sobre cada una de las zonas definidas y cada una de ellas puede apuntar a una URL diferente. Siguiendo el ejemplo anterior, una sola imagen que muestre un mapa de todos los continentes puede definir una zona diferente para cada continente. De esta forma, el usuario puede pinchar sobre la zona correspondiente a cada continente para que el navegador muestre la página que contiene los viajes disponibles a ese destino.

Las zonas o regiones que se pueden definir en una imagen se crean mediante rectángulos, círculos y polígonos. Para crear un mapa de imagen, en primer lugar se inserta la imagen original mediante la etiqueta . A continuación, se utiliza la etiqueta <map> para definir las zonas o regiones de la imagen. Cada zona se define mediante la etiqueta <area>.

<map>	Mapa de imagen
Atributos comunes	básicos, i18n y eventos
Atributos específicos	name = "texto" - Nombre que identifica de forma única al mapa definido (es obligatorio indicar un nombre único)
Tipo de elemento	Bloque y en línea
Descripción	Se emplea para definir mapas de imagen

<area>	Area de un mapa de imagen
Atributos comunes	básicos, i18n y eventos
Atributos específicos	href = "url" - URL a la que se accede al pinchar sobre el área nohref = "nohref" - Se usa para las áreas que no son seleccionables shape = "default rect circle poly" - Indica el tipo de área que se define (toda la imagen, rectangular, circular o poligonal) coords = "lista de números" - Se trata de una lista de números separados por comas que representan las coordenadas del área. Rectangular = X1,Y1,X2,Y2 (coordenadas X e Y del vértice superior izquierdo y coordenadas X e Y del vértice inferior derecho). Circular = X1,Y1,R (coordenadas X e Y del centro y radio del círculo). Poligonal = X1,Y1,X2,Y2,...,XnYn (coordenadas de los vértices del polígono. Si las últimas coordenadas no son iguales que las primeras, se cierra automáticamente el polígono uniendo ambos vértices)
Tipo de elemento	Etiqueta vacía
Descripción	Define las distintas áreas que forman un mapa de imagen

Si una imagen utiliza un mapa de imagen, debe indicarlo con el atributo usemap. El valor de este, debe ser el nombre del mapa de imagen definido en otra parte del mismo documento HTML:

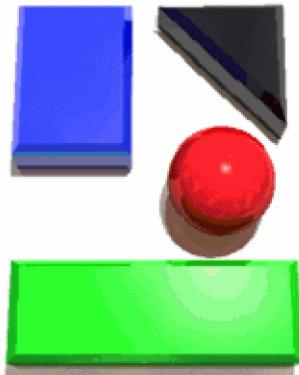
```

...
<map name="continentes">
...
</map>
```

Las áreas se definen mediante el atributo shape que indica el tipo de área y coords que es una lista de coordenadas cuyo significado depende del tipo de área definido. El enlace de cada área se define mediante el atributo href, con la misma sintaxis y significado que para los enlaces normales.

Veamos un ejemplo ([Ej06.02a-MapaImagen.html](#)):

```
<html>
  <head>
    <title>Ejemplo de Mapa de Imagen</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    
    <map name="mapa_zonas">
      <area shape="rect" coords="20,25,84,113" href="rectangulo.html" />
      <area shape="polygon" coords="90,25,162,26,163,96,89,25,90,24" href="triangulo.html" />
      <area shape="circle" coords="130,114,29" href="circulo.html" />
      <area shape="rect" coords="19,156,170,211" href="mailto:rectangulo@direccion.com" />
      <area shape="default" nohref="nohref" />
    </map>
  </body>
</html>
```



Otro ejemplo ([Ej06.02b-Los3Elefantes.html](#)):

NOTA: La imagen se puede obtener de esta página:

<http://www.regalospublicitariosm.com/LOS%203%20ELEFANTES%20CD.html>

```
<html>
  <head>
    <title>Ejemplo de Mapa de Imagen</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <p>Ejemplo de Mapa de Imágenes: </p>
    <p><br><br></p>
    
    <map name="elefantes">
      <area shape="RECT" coords="0,150,90,250"
            href="http://es.wikipedia.org/wiki/Sabana" alt="Nos vamos a la Sabana!!"/>
    </map>
  </p>
</body>
</html>
```

6.3 Objetos

Además de las imágenes, HTML permite incluir en las páginas web otros elementos mucho más complejos, como applets de Java y vídeos en formato QuickTime o Flash. La mayoría de este tipo de contenidos no los interpreta el navegador directamente, sino que hace uso de pequeños programas llamados plugins y que se encargan de tratar con este tipo de elementos complejos. La etiqueta <object> es la que permite incluir cualquier tipo de contenido complejo:

<object>	Objeto
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<code>data = "url"</code> - Indica la URL de los datos que utiliza el objeto <code>classid, codebase, codetype</code> - Información específica que depende del tipo de objeto <code>type</code> - Indica el tipo de contenido de los datos <code>height = "unidad_de_medida"</code> - Indica la altura con la que se debe mostrar el objeto <code>width = "unidad_de_medida"</code> - Indica la anchura con la que se debe mostrar el objeto
Tipo de elemento	Bloque y en línea
Descripción	Se emplea para embeber objetos en los documentos

El atributo data se emplea para indicar la URL del recurso que se va a incluir. El atributo type indica el tipo de contenido de los datos del objeto. Los posibles valores de type están estandarizados y coinciden con los del atributo type de la etiqueta <a> que se explicó anteriormente. El propio estándar de HTML incluye ejemplos de uso de esta etiqueta. Incluir un vídeo en formato MPEG:

```
<object data="PlanetaTierra.mpeg" type="application/mpeg" />
```

Aprovechando el ejemplo anterior, vamos a ver como usar un interesante plugin de Firefox que nos permite la descarga y conversión de vídeos: **Video-DownloadHelper**

<https://addons.mozilla.org/es/firefox/addon/video-downloadhelper/>

Por ejemplo, podemos descargar este video: <http://www.youtube.com/watch?v=lp2ZGND1I9Q> (también se puede encontrar buscando en Youtube el texto **Earth HD**)

Por otro lado, también se pueden incluir varias versiones alternativas de un mismo contenido. Así, si el navegador no es capaz de interpretar el formato por defecto, puede optar por cualquiera de los otros formatos alternativos. Veamos un ejemplo (**Ej06.03a-VideoTierra.html**):

```
<html>
  <head <title>Ejemplo de Objeto</title>
    <meta charset="UTF-8" /></head>
  <body>
    <object title="La Tierra vista desde el espacio"
      classid="http://www.observer.mars/TheEarth.py">
      <!-- Formato alternativo en forma de vídeo -->
      <object data="PlanetaTierra.mpeg" type="application/mpeg">
        <!-- Otro formato alternativo mediante una imagen GIF -->
        <object data="PlanetaTierra.gif" type="image/gif">
          <!-- Si el navegador no soporta ningún formato, se muestra el siguiente texto -->
          La <strong>Tierra</strong> vista desde el espacio.
        </object>
      </object>
    </object>
  </body>
</html>
```

A los objetos también se les puede pasar información adicional en forma de parámetros mediante la etiqueta `<param>`:

<param>	Parámetros de un objeto
Atributos comunes	Id
Atributos específicos	name = "texto" - Indica el nombre del parámetro value = "texto" - Indica el valor del parámetro
Tipo de elemento	Etiqueta vacía
Descripción	Se emplea para indicar el valor de los parámetros del objeto

Las etiquetas `<param>` siempre se incluyen en el interior de las etiquetas `<object>`:

```
<object data="..." type="...">
  <param name="parametro1" value="40" />
  <param name="parametro2" value="20" />
  <param name="parametro3" value="texto de prueba" />
</object>
```

Veamos un ejemplo ([Ej06.03b-TinoCasal.html](#)):

```
<html>
  <head>
    <meta charset="UTF-8" />
    <title> Título </title>
  </head>
  <body>
    <h1> Ejemplo de Video Vinculado (Tino Casal)</h1>
    <object>
      width="425" height="350"
      type="application/x-shockwave-flash"
      title="Tino Casal - Eloise"
      data="http://www.youtube.com/embed/McCfv0cM0Vw">
        <!-- Opciones Param (value: true|false): loop (bucle), menu,
        autostart, allowfullscreen y quality (value: low,medium, high)
        <param name="autostart" value="true" /> -->
    </object>
  </body>
</html>
```

El vídeo en cuestión es este:

http://www.youtube.com/watch?v=McCfv0cM0Vw&feature=player_embedded

Obsérvese que lo marcado en negrita en el código se corresponde con parte del URL.

Uno de los principales inconvenientes de `<object>` es la forma de incluir vídeos en formato Flash en las páginas HTML. Si se utiliza el siguiente código:

```
<object data="nombre_video.swf" type="application/x-shockwave-flash"></object>
```

El elemento anterior es correcto desde el punto de vista técnico, pero provoca que algunos navegadores como Internet Explorer no visualicen el vídeo hasta que se ha descargado completamente. Si se trata de un vídeo largo, esta solución no es válida para el usuario.

Por este motivo, se utiliza una solución alternativa para incluir vídeos Flash en las páginas HTML: el uso de la etiqueta `<embed>`. Aunque esta solución funciona correctamente, no se trata de una solución válida desde el punto de vista del estándar de XHTML, por lo que las páginas que incluyan esta solución no pasarán correctamente el proceso de validación.

Este es el motivo por el que los sitios web más populares de vídeos en formato Flash proporcionan un código similar al siguiente para incluir sus vídeos en las páginas HTML:

```
<object width="425" height="350">
  <param name="movie" value="http://www.youtube.com/v/MsH0rBWCYjs"></param>
  <param name="wmode" value="transparent"></param>
  <embed src="http://www.youtube.com/v/MsH0rBWCYjs" type="application/x-shockwave-flash"
wmode="transparent" width="425" height="350"></embed>
</object>
```

Una vez más, se debe tener en cuenta que la solución anterior de utilizar la etiqueta `<embed>` es correcta desde el punto de vista del usuario (no tiene que esperar a que el vídeo se descargue completamente para poder verlo) pero no es una solución técnicamente válida, ya que la etiqueta `<embed>` no es parte del estándar XHTML.

<embed>	Embeber objetos
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<code>src = "url"</code> - Indica la URL del archivo u objeto que se incluye en la página <code>type = "tipo_de_contenido"</code> - Indica el tipo de contenido del objeto (flash, quicktime, java, etc.) <code>height = "unidad_de_medida"</code> - Indica la altura con la que se debe mostrar el objeto <code>width = "unidad_de_medida"</code> - Indica la anchura con la que se debe mostrar el objeto
Tipo de elemento	Bloque
Descripción	Se emplea para embeber objetos en los documentos

7 TABLAS

Desde sus primeras versiones, HTML incluyó el soporte para crear tablas de datos en las páginas web. Además de ser sencillo, el modelo definido por HTML es muy flexible y bastante completo. Las tablas en HTML utilizan los mismos conceptos de filas, columnas, cabeceras y títulos que los que se utilizan en cualquier otro entorno de publicación de documentos:

Las tablas de HTML pueden contener elementos simples, agrupaciones de filas y de columnas, cabeceras y pies de tabla, subdivisiones, cabeceras múltiples y otros elementos complejos.

Cursos de diseño gráfico			
Nombre	Horas	Plazas	Horario
Introducción a XHTML	20	20	09:00 – 13:00
CSS avanzado	40	15	16:00 – 20:00
Taller de usabilidad	40	10	16:00 – 20:00
Introducción a AJAX	60	20	08:30 – 12:30

Diagrama de la estructura de la tabla:

- cabecera de tabla (arriba)
- cabecera de fila (izquierda)
- cabecera de columna (derecha)
- columna (abajo)
- fila (izquierda)

A pesar de que las tablas HTML son fáciles de comprender y utilizar, son uno de los elementos más polémicos de HTML. El problema de las tablas es que no siempre se utilizan adecuadamente. Aunque parezca obvio, las tablas se deben utilizar para mostrar información tabular.

Hasta hace unos años, las tablas también se utilizaban para definir la estructura de las páginas web. La cabecera de la página era una fila de una gran tabla, el pie de página era otra fila de esta tabla y la zona de contenidos estaba formada por varias columnas dentro de esa gran tabla.

Aunque algunos diseñadores siguen utilizando hoy en día las tablas para definir la estructura completa de las páginas web, se trata de una técnica obsoleta y nada recomendable. El motivo es que se complica en exceso el código HTML y su mantenimiento es muy complejo. La solución correcta para definir la estructura de las páginas consiste en la utilización de hojas de estilos CSS.

7.1 Tablas Básicas

Las tablas más sencillas de HTML se definen con tres etiquetas: `<table>` para crear la tabla, `<tr>` para crear cada fila y `<td>` para crear cada columna. Ejemplo ([Ej07.01a-TablaBasica.html](#)):

```
<html>
<head><title>Ejemplo de tabla sencilla</title>
<meta charset="UTF-8" />
<style type="text/css">
    th { background-color: #F00; color: #FFF; font-family: cursive; }
    table { margin: auto; }
</style></head>
<body>
    <h1>Listado de cursos</h1>
    <table border="1" summary="Tabla del contenido del Curso" height="80%" width="80%">
        <tr>
            <th><strong>Curso</strong></th>
            <th><strong>Horas</strong></th>
        </tr>
        <tr>
            <td>CSS</td>
            <td>20</td>
        </tr>
        <tr>
            <td>HTML</td>
            <td>20</td>
        </tr>
    </table>
</body>
</html>
```

La etiqueta <table> encierra todas las filas y columnas de la tabla. Las etiquetas <tr> (del inglés "table row") definen cada fila de la tabla y encierran todas las columnas. Por último, la etiqueta <td> (del inglés "table data cell") define cada una de las columnas de las filas, aunque realmente HTML no define columnas sino celdas de datos.

Al definir una tabla, se debe pensar en primer lugar en las filas que la forman y a continuación en las columnas. El motivo es que HTML procesa primero las filas y por eso las etiquetas <tr> aparecen antes que las etiquetas <td>.

<table>	Tabla
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<code>summary = "texto"</code> - Permite describir el contenido de la tabla (lo utilizan los buscadores y las personas discapacitadas)
Tipo de elemento	Bloque
Descripción	Se emplea para definir tablas de datos

<tr>	Fila de Tabla
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Se emplea para definir cada fila de las tablas de datos

<td>	Celda de Tabla
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<p><code>abbr = "texto"</code> - Permite describir el contenido de la celda (empleado sobre todo en los navegadores utilizados por personas discapacitadas)</p> <p><code>headers = "lista_de_id"</code> - Indica las celdas que actúan como cabeceras para esta celda (los títulos de las columnas y filas). Se indica como una lista de valores del atributo "id" de celdas</p> <p><code>scope = "col, row, colgroup, rowgroup"</code> - Indica las celdas para las que esta celda será su cabecera.</p> <p>Ej: <code>scope="col"</code> indica que esta celda es la cabecera de todas las demás celdas que están en la misma columna</p> <p><code>colspan = "numero"</code> - Número de columnas que ocupa esta celda</p> <p><code>rowspan = "numero"</code> - Número de filas que ocupa esta celda</p>
Tipo de elemento	Bloque
Descripción	Se emplea para definir cada celda de las tablas de datos

De todos los atributos disponibles para las celdas, los más utilizados son rowspan y colspan, que se emplean para construir tablas complejas como las que se ven más adelante. Entre los demás atributos, sólo se utiliza de forma habitual el atributo scope, sobre todo con las celdas de cabecera que se ven a continuación.

Normalmente, algunas de las celdas de la tabla se utilizan como cabecera de las demás celdas de la fila o de la columna. En este caso, HTML define la etiqueta `<th>` (del inglés "table header cell") para indicar que una celda es cabecera de otras celdas.

<th>	Celda Cabecera de Tabla
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<p><code>abbr = "texto"</code> - Permite describir el contenido de la celda (empleado sobre todo en los navegadores utilizados por personas discapacitadas)</p> <p><code>headers = "lista_de_id"</code> - Indica las celdas que actúan como cabeceras para esta celda (los títulos de las columnas y filas). Se indica como una lista de valores del atributo "id" de celdas</p> <p><code>scope = "col, row, colgroup, rowgroup"</code> - Indica las celdas para las que esta celda será su cabecera. Ej: <code>scope="col"</code> indica que esta celda es la cabecera de todas las demás celdas que están en la misma columna</p> <p><code>colspan = "numero"</code> - Número de columnas que ocupa esta celda <code>rowspan = "numero"</code> - Número de filas que ocupa esta celda</p>
Tipo de elemento	Bloque
Descripción	Se emplea para definir las celdas que son cabecera de una fila o de una columna de la tabla

Los atributos de la etiqueta `<th>` son idénticos que los atributos definidos para la etiqueta `<td>`. En este caso, el atributo más utilizado es `scope`, que permite indicar si la celda es cabecera de la fila o de la columna (`<th scope="row">` y `<th scope="col">` respectivamente).

Por otra parte, HTML define la etiqueta `<caption>` para establecer la leyenda o título de una tabla. La etiqueta debe colocarse inmediatamente después de la etiqueta `<table>` y cada tabla sólo puede incluir una etiqueta `<caption>`.

<caption>	Leyenda o Título de la Tabla
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	En Línea
Descripción	Se emplea para definir la leyenda o título de una tabla

Ejercicio 11

Su pedido

Determinar el código HTML necesario para crear la tabla que se muestra en la siguiente imagen:

Nombre producto	Precio unitario	Unidades	Subtotal
Reproductor MP3 (80 GB)	192.02	1	192.02
Fundas de colores	2.50	5	12.50
Reproductor de radio & control remoto	12.99	1	12.99
TOTAL	-	7	207.51

Resultado de la búsqueda

Imagen	Resultado de la búsqueda
	Portátil - 3 GHz - 4 GB RAM Comprar: 2.990 € 2.599 €
	Videocámara - Alta definición 1080p - 60 GB Comprar: 1.099 € 999 €
	Televisor - 46" - Full HD Comprar: 1.999 € 1.799 €
	Móvil - 3G - Wi-Fi - 8 GB Comprar: 399 € 349 €

El código será el siguiente:

([Ej07.01b-CombinarColumnas.html](#)):

```
<html>
<head><title>Combinación de Columnas</title>
<meta charset="UTF-8" /></head>
<body>
    <h1>Fusión de Columnas</h1>
    <table border="1">
        <tr>
            <td colspan="2">A</td>
        </tr>
        <tr>
            <td>B</td>
            <td>C</td>
        </tr>
    </table>
</body>
</html>
```

La primera fila de la tabla está formada sólo por una columna, mientras que la segunda fila está formada por dos columnas. En principio, podría pensarse en utilizar el siguiente código HTML para definir la tabla:

```
...
<table>
    <tr>
        <td>A</td>
    </tr>
    <tr>
        <td>B</td>
        <td>C</td>
    </tr>
</table>
...
```

Sin embargo, si se utiliza el código anterior, el navegador visualiza de forma incorrecta la tabla, ya que las tablas en HTML deben disponer de una estructura regular. En otras palabras, todas las filas de una tabla HTML deben tener el mismo número de columnas.

Ejercicio12

Determinar el código HTML necesario para crear la tabla que se muestra en la imagen de la izquierda.

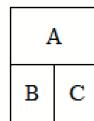
Utilizar las celdas de cabecera donde sea necesario y añadir todos los atributos posibles.

NOTA: Añadir en cada artículo un enlace Comprar que apunte a una página específica con las características del producto.

Las tablas complejas suelen disponer de una estructura irregular que junta varias columnas para formar una columna ancha o une varias filas para formar una fila más alta que las demás. Para fusionar filas o columnas, se utilizan los atributos rowspan y colspan respectivamente.

La siguiente imagen muestra una tabla compleja que ha fusionado dos columnas simples para formar una columna más ancha:

Fusión de columnas



Por lo tanto, si se quieren mostrar menos columnas en una fila, se fusionan mediante el atributo colspan, que indica el número de columnas simples que va a ocupar una determinada celda.

En el ejemplo anterior, la celda de la primera fila debe ocupar el espacio de dos columnas simples, por lo que el código HTML debe ser <td colspan="2">A</td>.

De forma equivalente, si se quiere diseñar una tabla HTML que fusiona filas como la de la siguiente imagen:

El código HTML que se debe utilizar para obtener la tabla de la imagen anterior es ([Ej07.01c-CombinarFilas.html](#)):

```
<html>
<head><title>Combinación de Filas</title>
    <meta charset="UTF-8" /></head>
<body>
    <h1>Fusión de Filas</h1>
    <table border="1">
        <tr>
            <td>A</td>
            <td rowspan="2">B</td>
        </tr>
        <tr>
            <td>C</td>
        </tr>
    </table>
</body>
</html>:
```

Fusión de filas

A	B
C	

De forma análoga a la fusión de columnas del ejemplo anterior, la fusión de filas debe indicarse de forma especial. Como las tablas HTML tienen que ser regulares, todas las columnas deben tener el mismo número de filas.

Utilizando los atributos rowspan y colspan, es posible diseñar tablas tan complejas como las que se muestran en los siguientes ejemplos ([Ej07.01d-FusionColumnas.html](#)):

```
<html>
<head><title>Ejemplo de columnas fusionadas</title>
    <meta charset="UTF-8" /></head>
<body>
    <h1>Fusión de columnas</h1>
    <table border="1">
        <tr>
            <td colspan="3">A</td>
            <td>B</td>
        </tr>
        <tr>
            <td>C</td>
            <td colspan="2">D</td>
            <td>E</td>
        </tr>
        <tr>
            <td colspan="4">F</td>
        </tr>
        <tr>
            <td>G</td>
            <td>H</td>
            <td>I</td>
            <td>J</td>
        </tr>
    </table>
</body>
</html>
```

Fusión de columnas

A		B	
C	D	E	
F			
G	H	I	J

Ejemplo de fusión de múltiples filas:

El código HTML necesario para fusionar las filas de la tabla situada a la derecha se muestra a continuación

([Ej07.01e-FusionFilas.html](#)):

```
<html>
<head><title>Ejemplo de Filas fusionadas</title>
<meta charset="UTF-8" /></head>
<body>
    <h1>Fusión de Filas</h1>
    <table border="1">
        <tr>
            <td>A</td>
            <td>B</td>
            <td rowspan="3">C</td>
            <td>D</td>
        </tr>
        <tr>
            <td rowspan="2">E</td>
            <td>F</td>
            <td rowspan="3">G</td>
        </tr>
        <tr>
            <td>H</td>
        </tr>
        <tr>
            <td>I</td>
            <td>J</td>
            <td>K</td>
        </tr>
    </table>
</body>
</html>
```

Ejercicio 13

Determinar el código HTML necesario para crear la tabla que se muestra en la siguiente imagen:

Fusión de filas

A	B		D
	F	C	
E	H		G
I	J	K	

Comparativa de reproductores MP3

Tabla comparativa de las características técnicas de los reproductores MP3

	MP3 mini			MP3 grande	
Capacidad de almacenamiento	4GB (1.000 canciones)	8GB (2.000 canciones)	16GB (4.000 canciones)	30GB (7.500 canciones)	80GB (20.000 canciones)
Colores	●	● ● ●	●	● ●	
Pantalla	LCD de 3 cm (diagonal) con retroiluminación			LCD de 6 cm (diagonal) con retroiluminación	
Tiempo de carga	Unas 3 horas			Unas 4 horas Unas 2 horas para alcanzar el 80% de la capacidad	

7.2 Tablas Avanzadas

Algunas tablas complejas están formadas por más elementos que filas y celdas de datos. Así, es común que las tablas más avanzadas dispongan de una sección de cabecera, una sección de pie y varias secciones de datos. Además, también es posible agrupar varias columnas de forma lógica para poder aplicar estilos similares a un determinado grupo de columnas.

Un ejemplo clásico de tablas avanzadas es el de las tablas utilizadas en contabilidad, como por ejemplo la tabla que muestra el balance de una empresa: (Ref: <http://bsolis.com/es/empresa/cifras>)

ACTIVO	EJERCICIO 2008
A) ACTIVO NO CORRIENTE	4.224.418,04
I. Inmovilizado intangible	2.738.315,41
II. Inmovilizado material.	1.156.054,48
III. Inversiones inmobiliarias.	33.326,04
IV. Inversiones en empresas del grupo y asociadas a largo plazo	49.239,27
V. Inversiones financieras a largo plazo.	2.929,77
VI. Activos por impuesto diferido.	244.553,07
B - ACTIVO CORRIENTE	68.280.278,35
II. Existencias.	15.783.454,09
III. Deudores comerciales y otras cuentas a cobrar.	40.429.579,29
IV. Inversiones en empresas del grupo y asociadas a corto plazo	4.017.815,28
V. Inversiones financieras a corto plazo.	723.907,10
VI. Periodificaciones a corto plazo.	513.848,25
VII. Efectivo y otros activos líquidos equivalentes.	6.811.674,04
TOTAL ACTIVO (A+B)	72.504.696,39
	
PATRIMONIO NETO Y PASIVO	EJERCICIO 2008
A) PATRIMONIO NETO	6.710.566,96
A-1) Fondos propios.	6.710.566,96
I. Capital.	480.800,00
II. Prima de emisión.	68.515,38
III. Reservas.	1.539.513,82
VII. Resultado del ejercicio.	4.621.737,76
B) PASIVO NO CORRIENTE	10.244.921,99
I. Provisiones a largo plazo.	50.000,00
II Deudas a largo plazo.	8.871.410,84
IV. Pasivos por impuesto diferido.	1.323.511,15
VI. Acreedores comerciales no corrientes.	0,00
C) PASIVO CORRIENTE	56.549.207,44
II. Provisiones a corto plazo.	1.088.381,81
III. Deudas a corto plazo.	5.275.109,71
V. Acreedores comerciales y otras cuentas a pagar.	49.185.715,92
TOTAL PATRIMONIO NETO Y PASIVO (A+B+C)	72.504.696,39

Las partes que componen las tablas complejas se definen mediante las etiquetas <thead>, <tbody> y <tfoot>. La cabecera de la tabla se define con la etiqueta <thead>, el pie de la tabla se define mediante <tfoot> y cada sección de datos se define con una etiqueta <tbody>.

<thead>	Cabecera de tabla
<tfoot>	Pie de tabla
<tbody>	Sección de una tabla
Atributos comunes	básicos, i18n y eventos
Atributos específicos	-
Tipo de elemento	Bloque
Descripción	Se emplean para agrupar varias filas en una cabecera (thead) un pie (tfoot) o una sección (tbody) de una tabla

Cada tabla puede contener solamente una cabecera y un pie, pero puede incluir un número ilimitado de secciones. Si se define una cabecera y/o un pie, las etiquetas <thead> y/o <tfoot> deben colocarse inmediatamente antes que cualquier etiqueta <tbody> (el navegador admite lo contrario, pero se hace para mejorar la carga de tablas grandes).

La siguiente imagen muestra una tabla avanzada con cabecera, pie y una sección de datos:

El código HTML necesario para crear la tabla de la imagen anterior hace uso de las etiquetas <thead>, <tbody> y <tfoot>:

([Ej07. 02a-TablasAvanzadas.html](#)):

```
<html>
<head><title>Tabla avanzada</title>
<meta charset="UTF-8" /></head>
<body>
    <h3>Análisis de ventas</h3>
    <table summary="Análisis de ventas anuales" border="1">
        <caption>Análisis de ventas anuales</caption>
        <thead>
            <tr>
                <th rowspan="2" scope="col">AÑO</th>
                <th colspan="4" scope="col">Expansión de ventas</th>
            </tr>
            <tr>
                <th scope="col">Producto A</th>
                <th scope="col">Producto B</th>
                <th scope="col">Producto C</th>
                <th scope="col">Producto D</th>
            </tr>
        </thead>
        <tfoot>
            <tr>
                <th rowspan="2" scope="col">AÑO</th>
                <th scope="col">Producto A</th>
                <th scope="col">Producto B</th>
                <th scope="col">Producto C</th>
                <th scope="col">Producto D</th>
            </tr>
            <tr>
                <th colspan="4" scope="col">Expansión de ventas</th>
            </tr>
        </tfoot>
        <tbody>
            <tr>
                <th scope="row">N-3</th><td>-</td><td>-</td><td>-</td><td>-</td>
            </tr>
            <tr>
                <th scope="row">N-2</th><td>3</td><td>5</td><td>8</td><td>4</td>
            </tr>
            <tr>
                <th scope="row">N-1</th><td>4</td><td>4</td><td>7</td><td>3</td>
            </tr>
            <tr>
                <th scope="row">N</th><td>5</td><td>7</td><td>6</td><td>2</td>
            </tr>
        </tbody>
    </table>
</body>
</html>
```

Análisis de ventas

AÑO	Análisis de ventas anuales			
	Producto A	Producto B	Producto C	Producto D
N-3	-	-	-	-
N-2	3	5	8	4
N-1	4	4	7	3
N	5	7	6	2
AÑO	Producto A	Producto B	Producto C	Producto D

Volviendo a lo anteriormente comentado, aunque al principio resulta extraño, el elemento `<tfoot>` siempre se escribe antes que cualquier elemento `<tbody>` en el código HTML. De hecho, si la etiqueta `<tfoot>` aparece después de un elemento `<tbody>`, la página no se considera válida (aunque se visualiza correctamente en muchos navegadores, como Firefox).

Análisis de ventas

La etiqueta `<tbody>` permite realizar agrupaciones de filas, pero en ocasiones se necesitan agrupar columnas. Aunque su uso no es muy común, HTML define dos etiquetas similares para agrupar columnas: `<col>` y `<colgroup>`.

La etiqueta `<col>` se utiliza para asignar los mismos atributos a varias columnas de forma simultánea.

De esta forma, la etiqueta `<col>` no agrupa columnas, sino que sólo asigna atributos comunes a varias columnas. El código HTML necesario para crear la tabla anterior se muestra a continuación: ([Ej07.02b-TablasCol.html](#)):

```

<html>
<head><title>Tabla avanzada</title>
<meta charset="UTF-8" /></head>
<body>
    <h3>Análisis de ventas</h3>
    <table summary="Análisis de ventas anuales" border="1">
        <caption>Análisis de ventas anuales</caption>
        <col style="width:10%;" />
        <col style="width:30%;" />
        <thead>
            <tr>
                <th scope="col">AÑO</th>
                <th scope="col">Producto A</th>
                <th scope="col">Producto B</th>
                <th scope="col">Producto C</th>
                <th scope="col">Producto D</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <th scope="row">N-3</th><td>-</td><td>-</td><td>-</td><td>-</td>
            </tr>
            <tr>
                <th scope="row">N-2</th><td>3</td><td>5</td><td>8</td><td>4</td>
            </tr>
            <tr>
                <th scope="row">N-1</th><td>4</td><td>4</td><td>7</td><td>3</td>
            </tr>
            <tr>
                <th scope="row">N</th><td>5</td><td>7</td><td>6</td><td>2</td>
            </tr>
        </tbody>
    </table>
</body>
</html>

```

Análisis de ventas anuales

AÑO	Producto A	Producto B	Producto C	Producto D
N-3	-	-	-	-
N-2	3	5	8	4
N-1	4	4	7	3
N	5	7	6	2

Con las líneas `<col style="width:10%;" />` y `<col style="width:30%;" />` indicamos que las 2 primeras columnas deben tener el 10% y el 30% de ancho. El resto se repartirá en partes iguales (en este caso, un 20% para las 3 columnas restantes).

Por otra parte, la etiqueta <colgroup> se emplea para agrupar de forma estructural varias columnas de la tabla. La forma habitual de indicar el número de columnas que abarca la agrupación es utilizar el atributo span, que establece el número de columnas de cada agrupación.

La siguiente imagen muestra una tabla avanzada con una agrupación de columnas realizada con la etiqueta <colgroup>:

El código HTML necesario para crear la tabla anterior se muestra a continuación:

([Ej07.02c-TabColGroup.html](#)):

```
<html>
<head><title>Tabla avanzada</title>
<meta charset="UTF-8" /></head>
<body>
    <h3>Análisis de ventas</h3>
    <table summary="Análisis de ventas anuales" border="1"  >
        <caption>Análisis de ventas anuales</caption>
        <colgroup span="1" style="background-color:silver" />
        <colgroup span="3" style="background-color:yellow"  />
        <thead>
            <tr>
                <th scope="col">AÑO</th>
                <th scope="col">Producto A</th>
                <th scope="col">Producto B</th>
                <th scope="col">Producto C</th>
                <th scope="col">Producto D</th>
            </tr>
        </thead>
        <tbody>
            <tr>
                <th scope="row">N-3</th><td>-</td><td>-</td><td>-</td><td>-</td><td>-</td>
            </tr>
            <tr>
                <th scope="row">N-2</th><td>3</td><td>5</td><td>8</td><td>4</td>
            </tr>
            <tr>
                <th scope="row">N-1</th><td>4</td><td>4</td><td>7</td><td>3</td>
            </tr>
            <tr>
                <th scope="row">N</th><td>5</td><td>7</td><td>6</td><td>2</td>
            </tr>
        </tbody>
    </table>
</body>
</html>
```

Análisis de ventas anuales				
AÑO	Producto A	Producto B	Producto C	Producto D
N-3	-	-	-	-
N-2	3	5	8	4
N-1	4	4	7	3
N	5	7	6	2

NOTA: Las etiquetas <colgroup> y <col> pueden dar problemas con los estilos CSS. La propiedad color no funciona correctamente ni en Firefox ni en Chrome.

El uso de las etiquetas <col> y <colgroup> no está muy extendido, debido a que la mayoría de navegadores no soportan muchas de sus funcionalidades.

Ejercicios de Tablas

Realizar las siguientes tablas respetando colores de fondo, combinación de celdas, etc.
Probar a usar las etiquetas <col> y <colgroup>

Animales en peligro de extinción			
Nombre	Cabezas	Previsión 2010	Previsión 2020
Ballena	6000	4000	1500
Oso Pardo	50	0	
Lince	10	0	
Tigre	300	210	

En la tabla inferior, a modo de indicación, debemos introducir sendas tablas en celdas específicas (es decir, tablas dentro de tablas).

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

8 FORMULARIOS

HTML es un lenguaje de marcado cuyo propósito principal consiste en estructurar los contenidos de los documentos y páginas web. Sin embargo, HTML también incluye elementos para crear aplicaciones web. El estándar HTML/XHTML permite crear formularios para que los usuarios interactúen con las aplicaciones web.

Los años transcurridos desde la publicación de los estándares de HTML y XHTML ha provocado que no estén disponibles todos los elementos utilizados por los formularios más avanzados y modernos. No obstante, HTML/XHTML incluye los suficientes elementos de formulario para crear desde los formularios sencillos que utilizan los buscadores hasta los formularios complejos de las aplicaciones más avanzadas.

8.1 Formularios Básicos

Los formularios más sencillos se pueden crear utilizando solamente dos etiquetas: <form> y <input>. Si se considera el formulario que muestra la siguiente imagen:

El código HTML necesario para definir el formulario anterior se muestra a continuación:

([Ej08.01a-FormularioSimple.html](#)):

```
<html>
<head>
    <title>Formulario</title>
    <meta charset="UTF-8" /></head>
<body>
    <h3>Formulario muy sencillo</h3>
    <form action="maneja_formulario.php" method="post">
        Escribe tu nombre:
        <input type="text" name="nombre" value="" /><br/>
        <input type="submit" value="Enviar" />
    </form>
</body>
</html>
```

NOTA IMPORTANTE: Para la gestión de formularios debemos usar lenguajes mas potentes que HTML. Por ejemplo, PHP (que ya veremos) con este archivo [maneja_formulario.php](#):

```
<?php
    // Si existe la variable nombre, lo recogemos
    if (isset ( $_POST ['nombre'])) {
        $nombre = $_POST ['nombre'];
        echo "<h1> Nombre: ".$nombre . "</h1>";
    }

    if (isset ( $_POST ['puesto'])) {
        foreach ( $_POST['puesto'] as $puestos) {
            echo $puestos."<br>";
        }
    }

    if (isset ( $_POST ['sexo'])) {
        $Sexo = $_POST ['sexo'];
        echo "<h1> Sexo: ".$Sexo . "</h1>";
    }
?>
```

IMPORTANTE: Para probarlo debemos hacer lo siguiente, instalar y arrancar XAMPP y colocar ambos archivos (el HTML y el PHP) en la carpeta HTDOCS, que tendrá permisos completos.

El proceso, dentro de la consola (Terminal) sería:

```
sudo mkdir /opt/lampp/htdocs/ejemplosHTML
sudo chmod 777 /opt/lampp/htdocs/ejemplosHTML).
```

Mas información: <http://www.apachefriends.org/en/xampp-linux.html>

La etiqueta <form> encierra todos los contenidos del formulario (botones, cuadros de texto, listas desplegables) y la etiqueta <input> permite definir varios tipos diferentes de elementos (botones y cuadros de texto).

<form>	Formulario
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<p><code>action = "url"</code> – Indica la URL que se encarga de procesar los datos del formulario</p> <p><code>method = "POST o GET"</code> – Método HTTP empleado al enviar el formulario</p> <p><code>enctype = "application/x-www-form-urlencoded O multipart/form-data"</code> – Tipo de codificación empleada al enviar el formulario al servidor (sólo se indica de forma explícita en los formularios que permiten adjuntar archivos)</p> <p><code>accept = "tipo_de_contenido"</code> – Lista separada por comas de todos los tipos de archivos aceptados por el servidor (sólo para los formularios que permiten adjuntar archivos)</p> <p>Otros: <code>accept-charset</code>, <code>onsubmit</code>, <code>onreset</code></p>
Tipo de elemento	Bloque
Descripción	Se emplea para insertar un formulario en la página

La mayoría de formularios utilizan sólo los atributos action y method. El atributo action indica la URL de la aplicación del servidor que se encarga de procesar los datos introducidos por los usuarios. Esta aplicación también se encarga de generar la respuesta que muestra el navegador.

El atributo method establece la forma en la que se envian los datos del formulario al servidor. Este atributo hace referencia al método HTTP, por lo que no es algo propio de HTML. Los dos valores que se utilizan en los formularios son GET y POST. De esta forma, casi todos los formularios incluyen el atributo `method="get"` o el atributo `method="post"`.

Al margen de otras diferencias técnicas, el método POST permite el envío de mucha más información que el método GET. En general, el método GET admite como máximo el envío de unos 500 bytes de información. La otra gran limitación del método GET es que no permite el envío de archivos adjuntos con el formulario. Además, los datos enviados mediante GET se ven en la barra de direcciones del navegador (se añaden al final de la URL de la página), mientras que los datos enviados mediante POST no se pueden ver tan fácilmente.

Si no sabes qué método elegir para un formulario, existe una regla general que dice que el método GET se debe utilizar en los formularios que no modifican la información (por ejemplo en un formulario de búsqueda). Por su parte, el método POST se debería utilizar cuando el formulario modifica la información original (insertar, modificar o borrar alguna información).

El ejemplo más común de formulario con método GET es el de los buscadores. Si realizas una búsqueda con tu buscador favorito, verás que las palabras que has introducido en tu búsqueda aparecen como parte de la URL de la página de resultados.

Del resto de atributos de la etiqueta <form>, el único que se utiliza ocasionalmente es enctype. Como se explica más adelante, este atributo es imprescindible en los formularios que permiten adjuntar archivos.

8.2 Elementos de Formulario

Los elementos de formulario como botones y cuadros de texto también se denominan "campos de formulario" y "controles de formulario". La mayoría de controles se crean con la etiqueta <input>, por lo que su definición formal y su lista de atributos es muy extensa:

<input>	Control de un formulario
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<p>type = "text password checkbox radio submit reset file hidden image button" - Indica el tipo de control que se incluye en el formulario</p> <p>name = "texto" - Asigna un nombre al control (es imprescindible para que el servidor pueda procesar el formulario)</p> <p>value = "texto" - Valor inicial del control</p> <p>size = "unidad_de_medida" - Tamaño inicial del control (para los campos de texto y de password se refiere al número de caracteres, en el resto de controles se refiere a su tamaño en píxel)</p> <p>maxlength = "numero" - Máximo número de caracteres para los controles de texto y de password</p> <p>checked = "checked" - Para los controles checkbox y radiobutton permite indicar qué opción aparece preseleccionada</p> <p>disabled = "disabled" - El control aparece deshabilitado y su valor no se envía al servidor junto con el resto de datos</p> <p>readonly = "readonly" - El contenido del control no se puede modificar</p> <p>src = "url" - Para el control que permite crear botones con imágenes, indica la URL de la imagen que se emplea como botón de formulario</p> <p>alt = "texto" - Descripción del control</p>
Tipo de elemento	En línea y etiqueta vacía
Descripción	Se emplean para insertar un control en un formulario

A continuación se muestran ejemplos para los diez controles que se pueden crear con la etiqueta <input>.

8.2.1 Cuadro de texto.

Se trata del elemento más utilizado en los formularios. En el caso más sencillo, se muestra un cuadro de texto vacío en el que el usuario puede escribir cualquier texto:

Nombre

A continuación se muestra el código HTML correspondiente al ejemplo anterior:

```
Nombre <br/>
<input type="text" name="nombre" value="" />
```

El atributo type diferencia a cada uno de los diez controles que se pueden crear con la etiqueta <input>. Para los cuadros de texto, su valor es text. El atributo name es el más importante en los campos del formulario. De hecho, si un campo no incluye el atributo name, sus datos no se envían al servidor. El valor que se indica en el atributo name es el nombre que utiliza la aplicación del servidor para obtener el valor de este campo de formulario.

Cuando el usuario pulsa el botón de envío del formulario, el navegador envía los datos a una aplicación del servidor para que procese la información y genere una respuesta adecuada. En el servidor, la aplicación que procesa los datos debe obtener en primer lugar toda la información introducida por el usuario. Para ello, utiliza el valor del atributo name para obtener los datos de cada control del formulario.

Como el valor del atributo name se utiliza en aplicaciones programadas, es esencial ponerse de acuerdo con el programador de la aplicación, no se debe modificar su valor sin modificar la aplicación y no se deben utilizar caracteres problemáticos en programación (espacios en blanco, acentos y caracteres como ñ o ç).

El atributo value se emplea para establecer el valor inicial del cuadro de texto. Si se crea un formulario para insertar datos, los cuadros de texto deberían estar vacíos. Por lo tanto, o no se añade el atributo value o se incluye con un valor vacío value="" . Si por el contrario se crea un formulario para modificar datos, lo lógico es que se muestren inicialmente los datos guardados en el sistema. En este caso, el atributo value incluirá el valor que se desea mostrar: <input type="text" name="nombre" value="Juan Pérez" />

Si no se especifica un tamaño, el navegador muestra el cuadro de texto con un tamaño predeterminado. El atributo size permite establecer el tamaño, en caracteres, con el que se muestra el cuadro de texto. Su uso es imprescindible en muchos formularios, en los que algunos campos como la dirección deben mostrar más caracteres de lo normal (<input size="100" ...) y otros campos como el código postal deben mostrar menos caracteres de lo normal (<input size="5" ...).

Además de controlar el tamaño con el que se muestra un cuadro de texto, también se puede limitar el tamaño del texto introducido. El atributo maxlength permite establecer el máximo número de caracteres que el usuario puede introducir en un cuadro de texto. Su uso es imprescindible para campos como el código postal, el número de la Seguridad Social y cualquier otro dato con formato predefinido y limitado.

Por último, el atributo readonly permite que el usuario pueda ver los contenidos del cuadro de texto pero no pueda modificarlos y el atributo disabled deshabilita un cuadro de texto de forma que el usuario no pueda modificarlo y además, el navegador no envía sus datos al servidor.

8.2.2 Cuadro de contraseña

La única diferencia entre este control y el cuadro de texto normal es que el texto que el usuario escribe en un cuadro de contraseña no se ve en la pantalla. En su lugar, los navegadores ocultan el texto utilizando asteriscos o círculos, por lo que es ideal para escribir contraseñas y otros datos sensibles.



A continuación se muestra el código HTML correspondiente al ejemplo anterior:

```
Contraseña <br/>
<input type="password" name="contrasena" value="" />
```

Cambiando el valor del atributo type por password se transforma el cuadro de texto normal en un cuadro de contraseña. Todos los demás atributos se utilizan de la misma forma y tienen el mismo significado.

8.2.3 Checkbox

Los checkbox o "casillas de verificación" son controles de formulario que permiten al usuario seleccionar y deseleccionar opciones individualmente. Aunque en ocasiones se muestran varios checkbox juntos, cada uno de ellos es completamente independiente del resto. Por este motivo, se utilizan cuando el usuario puede activar y desactivar varias opciones relacionadas pero no excluyentes.

A continuación, el código HTML correspondiente al ejemplo situado debajo:

(**Ej08.02a-FormularioCheckBox.html**):

```
<html>
<head>
    <title>Formulario Checkbox</title>
    <meta charset="UTF-8" />
</head>
<body>
    <form action="maneja_formulario.php" method="post">
        Puestos de trabajo buscados <br/>
        <input name="puesto[]" type="checkbox" value="Dirección"/> Dirección <br/>
        <input name="puesto[]" type="checkbox" value="Técnico"/> Técnico <br/>
        <input name="puesto[]" type="checkbox" value="empleado" checked="checked"/> Empleado
        <br/>
        <input type="submit" value="Enviar" />
    </form>
</body>
</html>
```

Puestos de trabajo buscados
□ Dirección
□ Técnico
□ Empleado

Y el fragmento de manejaformulario.php

```
// Para múltiples checkbox, usaremos un array
foreach ($_POST['puesto'] as $puestosElegidos) {
    echo $puestosElegidos . "<br>";
}
```

Aunque ya veremos el procesado con PHP, decir que en este caso hemos empleado como name dentro del form un array de datos (puesto[]) que pasaremos mediante POST al PHP.

Dentro de este, usaremos foreach para recorrer el array, asignarle un alias mediante AS y luego mostrar los valores con un echo.

El valor del atributo type para estos controles de formulario es checkbox. Como se muestra en el ejemplo anterior, el texto que se encuentra al lado de cada checkbox no se puede establecer mediante ningún atributo, por lo que es necesario añadirlo manualmente fuera del control del formulario. Si no se añade un texto al lado de la etiqueta <input /> del checkbox, el usuario sólo ve un pequeño cuadrado sin ninguna información relativa a la finalidad de ese checkbox.

El valor del atributo value, junto con el valor del atributo name, es la información que llega al servidor cuando el usuario envía el formulario.

Si se quiere mostrar un checkbox seleccionado por defecto, se utiliza el atributo checked. Si el valor del atributo es checked, el checkbox se muestra seleccionado. En cualquier otro caso, el checkbox permanece sin seleccionar. Aunque resulta redundante que el nombre y el valor del atributo sean idénticos, es obligatorio indicarlo de esta forma porque los atributos en XHTML no pueden tener valores vacíos:

```
<input type="checkbox" checked="checked" ... /> Checkbox seleccionado por defecto
```

8.2.4 Radiobutton

Los controles de tipo radiobutton son similares a los controles de tipo checkbox, pero presentan una diferencia muy importante: son mutuamente excluyentes. Los radiobutton se utilizan cuando el usuario solamente puede escoger una opción entre las distintas opciones relacionadas que se le presentan. Cada vez que se selecciona una opción, automáticamente se deselecciona la otra opción que estaba seleccionada.

Un ejemplo (añadimos estas líneas a [Ej08.02b-FormularioRadioButton.html](#)) :

```
...
<br/>
Sexo <br/>
<input type="radio" name="sexo" value="hombre" /> Hombre
<input type="radio" name="sexo" value="mujer" checked="checked" /> Mujer
<input type="submit" value="Enviar" />
...
Sexo
 Hombre
 Mujer
```

El valor del atributo type para estos controles de formulario es radio. El atributo name se emplea para indicar los radiobutton que están relacionados. Por lo tanto, cuando varios radiobutton tienen el mismo valor en su atributo name, el navegador sabe que están relacionados y puede deseleccionar una opción del grupo de radiobutton cuando se seleccione otra opción.

8.2.5 Botón de envío de formulario

La mayoría de formularios dispone de un botón para enviar al servidor los datos introducidos por el usuario:

Buscar

```
<input type="submit" name="buscar" value="Buscar" />
```

El valor del atributo type para este control de formulario es submit. El navegador se encarga de enviar automáticamente los datos cuando el usuario pincha sobre este tipo de botón. El valor del atributo value es el texto que muestra el botón. Si no se establece el atributo value, el navegador muestra el texto predefinido **Enviar consulta**.

8.2.6 Botón de reseteo del formulario

Aunque su uso era muy popular hace unos años, la mayoría de formularios modernos ya no utilizan este tipo de botón. Se trata de un botón especial que borra todos los datos introducidos por el usuario y devuelve el formulario a su estado original:

Borrar datos del formulario

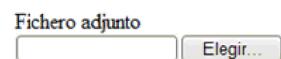
```
<input type="reset" name="limpiar" value="Borrar datos del formulario" />
```

El valor del atributo type para este control de formulario es reset. Cuando el usuario pulsa este botón, el navegador borra toda la información introducida y muestra el formulario en su estado original. Si el formulario no contenía originalmente ningún valor, el botón de reset lo vuelve a mostrar vacío. Si el formulario contenía información, el botón reset vuelve a mostrar la misma información original.

Como es habitual en los botones de formulario, el atributo value permite establecer el texto que muestra el botón. Si no se utiliza este atributo, el navegador muestra el texto predefinido del botón, que en este caso es **Restablecer**.

8.2.7 Ficheros adjuntos

Los formularios también permiten adjuntar archivos para subirlos al servidor. Aunque desde el punto de vista de HTML y del navegador no existe ninguna limitación sobre el número, tipo o tamaño total de los archivos que se pueden adjuntar, todos los servidores añaden restricciones por motivos de seguridad.



```
Fichero adjunto  
<input type="file" name="adjunto" />
```

El valor del atributo type para este control de formulario es file. El navegador se encarga de mostrar un cuadro de texto donde aparece el nombre del archivo seleccionado y un botón que permite navegar por los directorios y archivos del ordenador del usuario.

Si se incluye un control para adjuntar archivos, es obligatorio añadir el atributo enctype en la etiqueta <form> del formulario. El valor del atributo enctype debe ser multipart/form-data, por lo que la etiqueta <form> de los formularios que permiten adjuntar archivos siempre es:

```
<form action="..." method="post" enctype="multipart/form-data">  
...  
</form>
```

NOTA: Aunque no es parte de HTML, estimo conveniente incluir un ejemplo de subida de un archivo al servidor usando PHP. En concreto, la función @copy que copia el archivo elegido en el directorio donde se encuentra el php.

Ej08.02c-FormularioAdjuntos.html

```
<html>  
<head>  
    <title>Formulario Adjuntos</title>  
    <meta charset="UTF-8" />  
</head>  
<body>  
    <form action="maneja_formularioAdjuntos.php"  
        method="post" enctype="multipart/form-data">  
        Fichero adjunto  
        <input type="file" name="adjunto" />  
        <br/>  
        <input type="submit" value="Enviar" />  
    </form>  
</body>  
</html>
```

Puestos de trabajo buscados
 Dirección
 Técnico
 Empleado

Y ahora el archivo PHP que gestiona el adjunto **maneja_formularioAdjuntos.php**:

```
<?php  
  
// A partir de Archivo, podemos sacar su nombre, etc  
$archivo_nombre= $_FILES ['archivo'] ['name'];  
$archivo_temporal= $_FILES ['archivo'] ['tmp_name'];  
  
// Aquí usamos la función @copy, que copia el archivo al directorio temporal  
if (@copy ( $archivo_temporal, $archivo_nombre ) )  
{  
    echo "Nombre de Archivo: ".$archivo_nombre."  
";  
    echo "Archivo subido";  
}  
else  
    echo "Error al subir el archivo";  
?  
>
```

Para aumentar el tamaño de subida de archivos para PHP hacemos en consola:

sudo gedit /opt/lampp/etc/php.ini (o leafpad/mousepad en Xubuntu)

Buscamos: upload_max_filesize y cambiamos 2M por 10M.

Guardamos y REINICIAMOS XAMPP.

8.2.8 Campos ocultos

Los campos ocultos se emplean para añadir información oculta en el formulario:

```
<input type="hidden" name="url_previa" value="/articulo/primero.html" />
```

El valor del atributo type para este control de formulario es hidden. Los campos ocultos no se muestran por pantalla, de forma que el usuario desconoce que el formulario los incluye. Normalmente los campos ocultos se utilizan para incluir información que necesita el servidor pero que no es necesario o no es posible que la establezca el usuario.

8.2.9 Botón de imagen

El aspecto de los botones de formulario se puede personalizar por completo, ya que incluso es posible utilizar una imagen como botón:

```
<input type="image" name="enviar" src="accept.png" />
```



El valor del atributo type para este control de formulario es image. El atributo src indica la URL de la imagen que debe mostrar el navegador en lugar del botón normal. Su principal ventaja es que permite personalizar por completo la estética de los botones y mostrarlos con un aspecto homogéneo en todos los navegadores. El principal inconveniente es que ralentiza la carga del formulario y que si se quiere modificar su aspecto, es necesario crear una nueva imagen.

8.2.10 Botón

Algunos formularios complejos necesitan botones más avanzados que los de enviar datos (type="submit") y resetear el formulario (type="reset"). Por ese motivo, el estándar HTML/XHTML define un botón de tipo genérico:

```
<input type="button" name="guardar" value="Guardar Cambios" /> Guardar Cambios
```



El valor del atributo type para este control de formulario es button. Si pruebas a pulsar un botón de este tipo, verás que el navegador no hace nada: no envía los datos al servidor y no borra los datos introducidos. Este tipo de botones sólo son útiles si se utilizan junto con el lenguaje de programación JavaScript. Si la página incluye código JavaScript, los botones de este tipo se pueden programar para que realicen cualquier tarea compleja cuando se pulsa sobre ellos.

Ejercicio 14

Determinar el código HTML necesario para crear el formulario que se muestra en la siguiente imagen:

- Elegir el método más adecuado para el formulario (GET o POST) y cualquier otro atributo necesario.
- La aplicación que se encarga de procesar el formulario se encuentra en la raíz del servidor, carpeta "php" y archivo "insertar_cv.php".
- El nombre puede tener 20 caracteres como máximo, los apellidos 30 caracteres y la contraseña 10 caracteres como máximo.
- Asignar los atributos adecuados al campo del DNI.
- Por defecto, debe estar marcada la casilla de suscripción al boletín de novedades.

Rellena tu CV

Nombre

Apellidos

Contraseña

DNI

Sexo

- Hombre
- Mujer

Incluir mi foto:

No se ha seleccionado ningún archivo

Suscribirme al boletín de novedades

Guardar cambios Borrar los datos introducidos

8.3 Formularios Avanzados

Utilizando solamente las etiquetas `<form>` y `<input>` es posible diseñar la mayoría de formularios de las aplicaciones web. No obstante, HTML define algunos elementos adicionales para mejorar la estructura de los formularios creados.

La siguiente imagen muestra un formulario que agrupa sus elementos y añade etiquetas a cada campo para mejorar su estructura:

La etiqueta `<fieldset>` agrupa campos del formulario y la etiqueta `<legend>` asigna un nombre a cada grupo.

Datos personales

Nombre

Apellidos

DNI

Datos de conexión

Nombre de usuario

Contraseña

Repite la contraseña

<fieldset>	Agrupación de campos
Atributos comunes	básicos, i18n y eventos
Atributos específicos	–
Tipo de elemento	Bloque
Descripción	Se usa para agrupar de forma lógica varios campos de un formulario

Veamos el código HTML del formulario anterior y que hace uso de `<fieldset>` y `<legend>` para agrupar los campos del formulario [Ej08.03-FormularioAvanzado.html](#):

```
<html>
<head>
    <title>Formulario Avanzado</title>
    <meta charset="UTF-8" />
</head>

<body>
<form action="maneja_formulario.php" method="post">
    <fieldset>
        <legend>Datos personales</legend>
        Nombre <br/><input type="text" name="nombre" value="" /><br/>
        Apellidos <br/><input type="text" name="apellidos" value="" /><br/>
        DNI <br/><input type="text" name="dni" value="" size="10" maxlength="9" />
    </fieldset>

    <fieldset>
        <legend>Datos de conexión</legend>
        Nombre de usuario<br/><input type="text" name="nombre" value="" maxlength="10" /><br/>
        Contraseña<br/><input type="password" name="pass" value="" maxlength="10" /><br/>
        Repite la contraseña<br/><input type="password" name="pass2" value="" maxlength="10" />
    </fieldset>
</form>

</body>
</html>
```

La etiqueta `<fieldset>` agrupa todos los controles de formulario a los que encierra. El navegador muestra por defecto un borde resaltado para cada agrupación. La etiqueta `<legend>` se incluye dentro de cada etiqueta `<fieldset>` y establece el título que muestra el navegador para cada agrupación de elementos.

Por otra parte, todos los controles de formulario salvo los botones presentan una carencia muy importante: no disponen de la opción de establecer el título o texto que se muestra junto al control. En el código HTML del ejemplo anterior, el nombre de cada campo se incluye en forma de texto normal, sin ninguna relación con el campo al que hace referencia.

Afortunadamente, el lenguaje HTML incluye una etiqueta denominada `<label>` y que se utiliza para establecer el título de cada campo del formulario. Su definición formal es la siguiente:

<legend>	Título o leyenda de un campo de formulario
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<code>for = "id_de_elemento"</code> – Indica el ID del campo del formulario para el que este elemento es su título Otros: <code>accesskey</code> , <code>onfocus</code> y <code>onblur</code>
Tipo de elemento	Bloque
Descripción	Se emplea para definir el título o leyenda de un conjunto de campos de formulario agrupados con la etiqueta <code>fieldset</code>

El único atributo que suele utilizarse con la etiqueta `<label>` es `for`, que indica el identificador (atributo `id`) del campo de formulario para el que esta etiqueta hace de título.

En el anterior ejemplo, el nombre de los campos de formulario se incluía mediante un texto normal:

```
Nombre <br/>
<input type="text" name="nombre" value="" />

Apellidos <br/>
<input type="text" name="apellidos" value="" />

DNI <br/>
<input type="text" name="dni" value="" size="10" maxlength="9" />
```

Utilizando la etiqueta `<label>`, cada campo de formulario puede disponer de su propio título:

```
<label for="nombre">Nombre</label> <br/>
<input type="text" id="nombre" name="nombre" value="" />

<label for="apellidos">Apellidos</label> <br/>
<input type="text" id="apellidos" name="apellidos" value="" />

<label for="dni">DNI</label> <br/>
<input type="text" id="dni" name="dni" value="" size="10" maxlength="9" />
```

La principal ventaja de utilizar `<label>` es que el código HTML está mejor estructurado y se mejora su accesibilidad. Además, al pinchar sobre el texto del `<label>`, el puntero del ratón se posiciona automáticamente para poder escribir sobre el campo de formulario asociado. Este comportamiento es especialmente útil para los campos de tipo `radio`button y `checkbox`.

8.4 Otros elementos de Formulario

La etiqueta <input> permite crear diez tipos diferentes de controles de formulario. Sin embargo, algunas aplicaciones web utilizan otros elementos de formulario que no se pueden crear con <input>. Las listas desplegables y las áreas de texto disponen de sus propias etiquetas (<select> y <textarea> respectivamente).

Las áreas de texto son útiles cuando se debe introducir una gran cantidad de texto, ya que es mucho más cómodo de introducir que en un campo de texto normal:

Nombre del producto

Descripción del producto

El código HTML del ejemplo anterior es ([Ej08.04a-FormularioTextArea.html](#)):

```
<html>
<head>
<title>Formulario Avanzado</title>
<meta charset="UTF-8" />
</head>
<body>
    <form action="insertar_producto.php" method="post">
        <label for="nombre">Nombre del producto</label> <br/>
        <input type="text" id="nombre" name="nombre" value="" />

        <label for="descripcion">Descripción del producto</label> <br/>
        <textarea id="descripcion" name="descripcion" cols="40" rows="5"></textarea>
    </form>
</body>
</html>
```

La definición formal de la etiqueta <textarea> es:

<textarea>	Área de texto
Atributos comunes	básicos, i18n y eventos
Atributos específicos	rows = "numero" – Número de filas de texto que mostrará el textarea cols = "numero" – Número de caracteres que se muestran en cada fila del textarea Otros: name, disabled, readonly, onselect, onchange, onfocus, onblur
Tipo de elemento	En línea
Descripción	Se emplea para incluir un área de texto en un formulario

Los atributos más utilizados en las etiquetas <textarea> son los que controlan su anchura y altura. La anchura del área de texto se controla mediante el atributo cols, que indica las columnas o número de caracteres que se podrán escribir como máximo en cada fila. La altura del área de texto se controla mediante rows, que indica directamente las filas de texto que serán visibles.

El principal inconveniente de los elementos <textarea> es que el lenguaje HTML no permite limitar el número máximo de caracteres que se pueden introducir. Mientras los elementos <input type="text"> disponen del atributo maxlength, las áreas de texto no disponen de un atributo equivalente, por lo que sólo es posible limitar el número de caracteres mediante su programación con JavaScript.

Por otra parte, el otro control disponible para los formularios es el de las listas desplegables:

La imagen anterior muestra los tres tipos de listas desplegables disponibles. El primero es el de las listas más utilizadas que sólo muestran un valor cada vez y sólo permiten seleccionar un valor. El segundo tipo de lista es el que sólo permite seleccionar un valor pero muestra varios a la vez. Por último, el tercer tipo de lista desplegable es aquella que muestra varios valores y permite realizar selecciones múltiples.

El código HTML del ejemplo anterior es
(Ej08.04b-FormularioSelect.html):

```
<html>
<head><title>Formulario Avanzado</title>
<meta charset="UTF-8" /></head>
<body>
<form action="elegir_SistOperativo.php" method="post">
<label for="so">Sistema operativo</label> <br/>
<select id="so" name="so">
<option value="" selected="selected">- selecciona -</option>
<option value="linux">Linux</option>
<option value="windows">Windows</option>
<option value="mac">Mac</option>
<option value="otro">Otro</option>
</select>

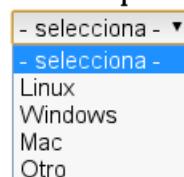
<br/><br/><br/><br/><br/>
<label for="so2">Sistema operativo</label> <br/>
<select id="so2" name="so2" size="5">
<option value="linux" selected="selected">Linux</option>
<option value="mac">Mac</option>
<option value="windows">Windows</option>
<option value="otro">Otro</option>
</select>

<label for="so3">Sistema operativo</label> <br/>
<select id="so3" name="so3" size="5" multiple="multiple">
<option value="linux" selected="selected">Linux</option>
<option value="mac">Mac</option>
<option value="windows">Windows</option>
<option value="otro">Otro</option>
</select>
</form>
</body>
</html>
```

Los tres tipos de listas desplegables se definen con la misma etiqueta `<select>` y cada elemento de la lista se define mediante la etiqueta `<option>`:

<select>	Lista desplegable
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<p><code>size = "numero"</code> - Número de filas que se muestran de la lista (por defecto sólo se muestra una)</p> <p><code>multiple = "multiple"</code> - Si se incluye, se permite seleccionar más de un elemento</p> <p>Otros: <code>name</code>, <code>disabled</code>, <code>onchange</code>, <code>onfocus</code>, <code>onblur</code></p>
Tipo de elemento	En línea
Descripción	Se emplea para incluir una lista desplegable en un formulario

Sistema operativo



Sistema operativo



Sistema operativo



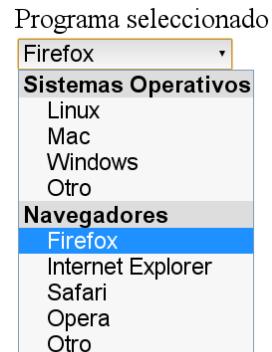
<option>	Elemento de una lista desplegable
Atributos comunes	básicos, i18n y eventos
Atributos específicos	<p><code>selected = "selected"</code> - Indica si el elemento aparece seleccionado por defecto al cargarse la página</p> <p><code>value = "texto"</code> - El valor que se envía al servidor cuando el usuario elige esa opción</p> <p>Otros: <code>label</code>, <code>disabled</code></p>
Tipo de elemento	-
Descripción	Se emplea para definir cada elemento de una lista desplegable

La inmensa mayoría de listas desplegables que utilizan las aplicaciones web son simples, por lo que el código HTML habitual de las listas desplegables es:

```
<label for="so">Sistema operativo</label> <br/>
<select id="so" name="so">
  <option value="" selected="selected">- selecciona -</option>
  <option value="linux">Linux</option>
  <option value="windows">Windows</option>
  <option value="mac">Mac</option>
  <option value="otro">Otro</option>
</select>
```

La etiqueta `<select>` define la lista y encierra todas las opciones que muestra la lista. Cada una de las opciones de la lista se define mediante una etiqueta `<option>`. El atributo `value` de cada opción es obligatorio, ya que es el dato que se envía al servidor cuando el usuario envía el formulario. Para seleccionar por defecto una opción al mostrar la lista, se añade el atributo `selected` a la opción deseada.

Por otra parte, las listas desplegables permiten agrupar sus opciones de forma que el usuario pueda encontrar fácilmente las opciones cuando la lista es muy larga:



El código HTML del ejemplo anterior es ([Ej08.04c-FormOptgroup.html](#)):

```
<html>
<head><title>Formulario Avanzado</title>
<meta charset="UTF-8" /></head>
<body>
  <form action="elegir_SistOperativo.php" method="post">
    <label for="programa">Programa seleccionado</label> <br/>
    <select id="programa" name="programa">
      <optgroup label="Sistemas Operativos">
        <option value="Linux" selected="selected">Linux</option>
        <option value="Mac">Mac</option>
        <option value="Windows">Windows</option>
        <option value="Other">Otro</option>
      </optgroup>
      <optgroup label="Navegadores">
        <option value="Firefox" selected="selected">Firefox</option>
        <option value="Internet Explorer">Internet Explorer</option>
        <option value="Safari">Safari</option>
        <option value="Opera">Opera</option>
        <option value="Other">Otro</option>
      </optgroup>
    </select>
  </form>
</body>
</html>
```

La etiqueta <optgroup> permite agrupar opciones relacionadas dentro de una lista desplegable. Su definición formal se muestra a continuación:

<optgroup>	Agrupación de elementos de una lista desplegable
Atributos comunes	básicos, i18n y eventos
Atributos específicos	label = "texto" – Texto que se muestra como título de la agrupación de opciones Otros: selected, disabled
Tipo de elemento	–
Descripción	Se emplea para definir una agrupación lógica de opciones de una lista desplegable

El único atributo que suele utilizarse con la etiqueta <optgroup> es label, que indica el nombre de cada agrupación. Los navegadores muestran de forma destacada el título de cada agrupación, de forma que el usuario pueda localizar más fácilmente la opción deseada.

Ejercicio 15

Determinar el código HTML necesario para crear el formulario que se muestra en la siguiente imagen:
Y además, el PHP que lo gestiona.

Rellena tu CV

Datos personales

Provincia

- selecciona -
Álava/Araba
Albacete
Alicante/Alacant
Almería

Fecha de nacimiento

_____ de Enero de _____

Temas de interés

Administración de bases de datos
Análisis y programación
Arquitectura
Calidad
ERP, CRM, Business Intelligence

Información sobre el producto

Datos básicos

Nombre

Descripción

Foto Seleccionar archivo No se ha seleccionado ningún archivo

Añadir contador de visitas

Datos económicos

Precio € Impuestos
 4%
 7%
 16%
 25%

Promoción

Ninguno
 Transporte gratuito
 Descuento 5%

Ejercicio 16

Determinar el código HTML necesario para crear el formulario que se muestra en la imagen situada a la izquierda:

9 CAPAS Y METAINFORMACIÓN

Los capítulos anteriores muestran las decenas de etiquetas XHTML disponibles para marcar y estructurar cada elemento individual de las páginas web: tablas, listas, enlaces, párrafos, imágenes, etc. Aunque combinando esas etiquetas es posible crear cualquier página web, no es posible hacer que las páginas muestren estructuras complejas.

La mayoría de páginas HTML disponen de estructuras complejas formadas por varias columnas de contenidos y otro tipo de divisiones. Utilizando exclusivamente HTML no es posible crear estas estructuras complejas, ya que es imprescindible emplear las hojas de estilos CSS.

No obstante, los estilos de CSS necesitan la ayuda de HTML/XHTML para crear los diseños más avanzados. En concreto, el código HTML se encarga de agrupar los elementos de la página en diferentes divisiones en función de su finalidad: la zona de la cabecera de la página, la zona de contenidos, una zona lateral para el menú y otras secciones menores, la zona del pie de página, etc. Veamos un ejemplo en la página <http://www.alistapart.com/>

The screenshot shows the homepage of A List Apart (ALa). At the top, there's a navigation bar with links: ARTICLES • TOPICS • ABOUT • CONTACT • CONTRIBUTE • FEED. To the left, there's a sidebar with the A La logo and a badge saying 'No. 367'. The main content area has several

elements highlighted with different colors: a green border surrounds the main article content ('Vexing Viewports' by Peter-Paul Koch, Luke Wroblewski, Stephanie Rieger, Lyza Danger Gardner); a red border surrounds the 'AN EVENT APART' section; and a blue border surrounds the right sidebar. The sidebar itself contains a search field, a checkbox for 'include discussions', and a 'Topics' section with links to 'Code >', 'Content >', 'Culture >', 'Design >', 'Mobile >', 'Process >', and 'User Science >'.

Los <div> están marcados en colores:

- En rojo, la Navegación
- En azul, el Principal que ha su vez engloba varios <div>:
 - En verde, los contenidos.
 - En rosa, la columna secundaria.
 - En verde oscuro, el lateral

Para agrupar los elementos que forman cada zona o división de la página se usa la etiqueta <div>:

<div>	Divisiones
Atributos comunes	básicos, i18n y eventos
Atributos específicos	–
Tipo de elemento	Bloque
Descripción	Agrupa elementos de bloque

El nombre de la etiqueta div tiene su origen en la palabra división, ya que esta etiqueta define zonas o divisiones dentro de una página HTML. En cualquier caso, casi todos los diseñadores web utilizan la palabra "capa" para referirse a una "división". Aunque se trata de un error (las capas se crean mediante una propiedad de CSS llamada z-index) es preferible seguir llamando "capas" a las zonas definidas con la etiqueta <div> para poder entenderse con el resto de diseñadores.

9.1 Layouts o Capas

Las páginas web complejas que están bien diseñadas utilizan decenas de etiquetas <div>. Con mucha diferencia, los atributos más utilizados con esta etiqueta son id (para identificar la capa de forma única) y class (para aplicar a la capa estilos CSS).

No se va a profundizar en el proceso de diseñar una página web mediante <div>, ya que no es posible diseñar una página web compleja usando elementos <div> sin utilizar CSS.

Por último, si observas el código HTML de algunas páginas web complejas, verás que la mayoría utilizan los mismos nombres para identificar sus divisiones. Los nombres más comunes, y sus equivalentes en inglés, se muestran a continuación:

- **contenedor** (wrapper) suele encerrar la mayor parte de los contenidos de la página y se emplea para definir las características básicas de la página: su anchura, sus bordes, imágenes laterales, si se centra o no respecto de la ventana del navegador, etc.
- **cabecera** (header) que incluye todos los elementos invariantes de la parte superior de la página (logotipo, imagen o banner, cuadro de búsqueda superior, etc.)
- **contenido** (content) engloba el contenido principal del sitio (la zona de noticias, la zona de artículos, la zona de productos, etc. dependiendo del tipo de sitio web)
- **menu** (menu) se emplea para agrupar todos los elementos del menú lateral de navegación de la página
- **pie** (footer) que incluye todos los elementos invariantes de la parte inferior de la página (aviso de copyright, política de privacidad, términos de uso, etc.)
- **lateral** (sidebar) se emplea para agrupar los elementos de las columnas laterales y secundarias de la página.

De esta forma, el esqueleto de una página HTML compleja suele ser similar al siguiente:

```
...
<div id="contenedor">
  <div id="cabecera">
    ...
  </div>
  <div id="contenido">
    <div id="menu">
      ...
    </div>
    ...
  </div>
  <div id="pie">
    ...
  </div>
</div>
...
```

El equivalente para las páginas en inglés sería el siguiente:

```
...
<div id="wrapper">
  <div id="header">
    ...
  </div>
  <div id="content">
    <div id="menu">
      ...
    </div>
    ...
  </div>
  <div id="footer">
    ...
  </div>
</div>
```

Veamos un ejemplo con Secciones y usando CSS
Para los colores hemos empleado la siguiente Url:
<https://kuler.adobe.com>

veamos un ejemplo (**Ej09.01-Divisiones.html**):

```
<html>
<head>
    <meta charset="UTF-8" />
    <title> Título </title>
    <style type="text/css">
        .navegacion {
            position: absolute;
            width: 80%;
            height: 20%;
            margin: auto;
            background-color: rgb(255,140,13);
            color: white;
            font-weight: bold;
            font-size: 50px;
        }

        .contenido {
            position: absolute;
            width: 60%;
            height: 50%;
            margin: auto;
            top: 150px;
            left: 50px;
            background-color: blue;
            color: white;
        }

        .pie {
            position: absolute;
            width: 80%;
            height: 20%;
            margin: auto;
            top: 500px;
            background-color: red;
            color: white;
            font-weight: bold;
            font-size: xx-large;
        }
    </style>
</head>
<body>
    <div class="navegacion">
        <p> Esto es la sección de navegación</p>
    </div>

    <div class="contenido">
        <p> Esto es la sección de contenido</p>
    </div>

    <div class="pie">
        <p> Esto es la sección de contenido</p>
    </div>
</body>
</html>
```

9.2 Estructura de la Cabecera

Las páginas y documentos HTML incluyen más información de la que los usuarios ven en sus pantallas. Estos datos adicionales siempre están relacionados con la propia página, por lo que se denominan metainformación o metadatos. La metainformación siempre se incluye en la sección de la cabecera, es decir, dentro de la etiqueta <head>.

Como ya se explicó anteriormente, las páginas XHTML se dividen en dos partes denominadas cabecera y cuerpo. La sección de la cabecera está formada por todas las etiquetas encerradas por la etiqueta <head>:

<head>	Cabecera
Atributos comunes	i18n
Atributos específicos	profile = "url" - Especifica la URL del perfil o perfiles que utilizan los metadatos lang = "codigo_de_idioma" - Especifica el idioma principal de los contenidos de la página
Tipo de elemento	-
Descripción	Define la cabecera del documento HTML

La cabecera típica de una página HTML completa presenta la siguiente estructura:

El código HTML del ejemplo anterior es ([Ej09.02a-Cabecera.html](#)):

```
<html>
<head>
<!-- Zona de etiquetas META --&gt;
&lt;meta http-equiv="Content-Type" content="text/html; charset=UTF-8" /&gt;

<!-- Zona de título --&gt;
&lt;title&gt;El título del documento&lt;/title&gt;

<!-- Zona de recursos enlazados (CSS, RSS, JavaScript) --&gt;
&lt;link rel="stylesheet" href="#" type="text/css" media="screen" /&gt;
&lt;link rel="stylesheet" href="#" type="text/css" media="print" /&gt;

&lt;link rel="alternate" type="application/rss+xml" title="RSS 2.0" href="#" /&gt;

&lt;script src="#" type="text/javascript"&gt;&lt;/script&gt;
&lt;/head&gt;
&lt;body&gt;
  &lt;h1&gt;Hola Mundo! &lt;/h1&gt;
&lt;/body&gt;
&lt;/html&gt;</pre>
```

La etiqueta <title> establece el título de la página. Los navegadores muestran este título como título de la propia ventana del navegador. Los buscadores utilizan este título como título de sus resultados de búsqueda.

Por tanto, el valor de <title> no sólo es importante para los usuarios, sino que también es importante para que los usuarios encuentren las páginas a través de los buscadores. Un error común de muchos sitios web consiste en mostrar un mismo título genérico en todas sus páginas. Cada página debe mostrar un título corto, adecuado, único y que describa inequívocamente los contenidos de la página.

Las páginas XHTML deben tener definido un título y sólo uno, por lo que todas las páginas web deben incluir obligatoriamente una etiqueta <title>, cuya definición formal se muestra a continuación:

<title>	Título del documento
Atributos comunes	i18n
Atributos específicos	lang = "codigo_de_idioma" – Especifica el idioma principal del título de la página
Tipo de elemento	–
Descripción	Define el título del documento HTML

Por último, la etiqueta <head> permite definir en el atributo profile la URL de un documento externo que contiene el perfil que siguen los metadatos de la cabecera. Los blogs creados con el programa WordPress incluyen por ejemplo el siguiente perfil en su cabecera:

```
<head profile="http://gmpg.org/xfn/11">
  ...
</head>
```

El documento <http://gmpg.org/xfn/11> es un perfil que define atributos adicionales para establecer la relación entre sitios web.

9.3 Metadatos

Una de las partes más importantes de la metainformación de la página son los metadatos, que permiten incluir cualquier información relevante sobre la propia página.

La especificación oficial de HTML no define la lista de metadatos que se pueden incluir, por lo que las páginas tienen libertad absoluta para definir los metadatos que consideren adecuados. La etiqueta empleada para la definición de los metadatos es <meta>.

<meta>	Metadatos
Atributos comunes	i18n
Atributos específicos	<p>name = "texto" – El nombre de la propiedad que se define (no existe una lista oficial de propiedades)</p> <p>content = "texto" – El valor de la propiedad definida (no existe una lista de valores permitidos)</p> <p>http-equiv = "texto" – En ocasiones, reemplaza al atributo "name" y lo emplean los servidores para adaptar sus respuestas al documento</p> <p>scheme = "texto" – Indica el esquema que se debe emplear para interpretar el valor de la propiedad</p>
Tipo de elemento	–
Descripción	Permite definir el valor de los metadatos que forman la metainformación del documento

Los metadatos habituales utilizan solamente los atributos name y content para definir el nombre y el valor del metadato:

```
<meta name="autor" content="Iván Rodríguez Ruiz" />
```

No obstante, algunas etiquetas <meta> muy utilizadas hacen uso del atributo http-equiv. Este atributo se utiliza para indicar que el valor establecido por este metadato puede ser utilizado por el servidor al entregar la página al navegador del usuario. El siguiente metadato indica al servidor que el contenido de la página es código HTML y su codificación de caracteres es UTF-8:

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

El atributo scheme no suele utilizarse, aunque permite proporcionar información de contexto para que el navegador interprete correctamente el valor del metadato. En el siguiente ejemplo, el atributo scheme indica al navegador que el valor del metadato hace referencia al código ISBN:

```
<meta scheme="ISBN" name="identificador" content="84-8450-044-6">
```

Aunque no existe una lista oficial con los metadatos que se pueden definir, algunos de ellos se utilizan en tantas páginas que se han convertido prácticamente en un estándar. A continuación se muestran los metadatos más utilizados:

Definir el autor del documento:

```
<meta name="author" content="Iván Rodríguez Ruiz" />
```

Definir el programa con el que se ha creado el documento:

```
<meta name="generator" content="WordPress 2.8.4" />
```

Definir la codificación de caracteres del documento:

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

```
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1" />
```

Definir el copyright del documento:

```
<meta name="copyright" content="www.granaforma.com" />
```

Definir el comportamiento de los buscadores:

```
<meta name="robots" content="index, follow" />
```

Definir las palabras clave que definen el contenido del documento:

```
<meta name="keywords" content="diseño, css, hojas de estilos, web, html" />
```

Definir una breve descripción del sitio:

```
<meta name="description" content="Apuntes de diseño web" />
```

La etiqueta que define la codificación de los caracteres (http-equiv="Content-Type") se emplea prácticamente en todas las páginas y las etiquetas que definen la descripción (description) y las palabras clave (keywords) también son muy utilizadas.

9.4 DOCTYPE

El estándar XHTML deriva de XML, por lo que comparte con el muchas de sus normas y sintaxis. Uno de los conceptos fundamentales de XML es la utilización del DTD o Document Type Definition ("Definición del Tipo de Documento").

Un DTD es un documento que recoge el conjunto de normas y restricciones que deben cumplir los documentos de un determinado tipo. Si por ejemplo se define un DTD para los documentos relacionados con libros, se puede fijar como norma que cada libro tenga un título y sólo uno, que tenga uno o más autores, que la información sobre el número de páginas pueda ser opcional, etc.

El conjunto de normas, obligaciones y restricciones que se deben seguir al crear un documento de un determinado tipo, se recogen en su correspondiente DTD. El estándar XHTML define el DTD que deben seguir las páginas y documentos XHTML. En este documento se definen las etiquetas que se pueden utilizar, los atributos de cada etiqueta y el tipo de valores que puede tener cada atributo.

En realidad, la versión 1.0 del estándar de XHTML define tres DTD diferentes. Para indicar el DTD utilizado al crear una determinada página, se emplea una etiqueta especial llamada doctype. La etiqueta doctype es el único elemento que se incluye fuera de la etiqueta <html> de la página. De hecho, la declaración del doctype es lo primero que se debe incluir en una página web, antes incluso que la etiqueta <html>.

Como se verá más adelante, para que una página XHTML sea correcta y válida es imprescindible que incluya el correspondiente doctype que indica el DTD utilizado. A continuación se muestran los tres DTD que se pueden utilizar al crear páginas XHTML:

XHTML 1.0 Estricto

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

Se trata de la variante con las normas más estrictas y las restricciones más severas. Las páginas web que incluyan este doctype, no pueden utilizar atributos relacionados con el aspecto de los contenidos, por lo que requiere una separación total de código HTML y estilos CSS.

XHTML 1.0 Transitorio

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

Se trata de una variante menos estricta que la anterior, ya que permite el uso de algunos atributos HTML relacionados con el aspecto de los elementos.

XHTML 1.0 Frameset

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

Esta última variante la utilizan las páginas que están formadas por frames, una práctica completamente desaconsejada y que hoy en día sólo utilizan los sitios web obsoletos.

Si no tienes claro el DTD que más te conviene, deberías utilizar el XHTML 1.0 transitorio, ya que es más fácil crear páginas web válidas. Si tienes conocimientos avanzados de XHTML, puedes utilizar XHTML 1.0 estricto.

Por otra parte, además del DOCTYPE apropiado, también es necesario que las páginas web indiquen el namespace asociado. Un namespace en un documento XML permite diferenciar las etiquetas y atributos que pertenecen a cada lenguaje.

Si en un mismo documento se mezclan etiquetas de dos o más lenguajes derivados de XML (XHTML y SVG por ejemplo) y que tienen el mismo nombre, no se podría determinar a qué lenguaje pertenece cada etiqueta y por tanto, no se podría interpretar esa etiqueta o ese atributo. Los namespaces se indican mediante una URL.

El namespace que utilizan todas las páginas XHTML (independientemente de la versión y del DOCTYPE) es <http://www.w3.org/1999/xhtml> y se indica de la siguiente manera:

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

De esta forma, es habitual que las páginas XHTML comiencen con el siguiente código:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
      "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">  
<html xmlns="http://www.w3.org/1999/xhtml" lang="es" xml:lang="es">
```

Aunque el código anterior es mucho más complicado que una simple etiqueta <html>, es imprescindible para que las páginas XHTML creadas sean correctas y superen satisfactoriamente el proceso de validación que se muestra en los capítulos siguientes.

Afortunadamente, si utilizas un editor avanzado como Dreamweaver para crear las páginas, todo el código anterior se incluye de forma automática. Si creas las páginas a mano, sólo tienes que copiar y pegar ese código en cada nueva página.

<meta>	Metadatos
Atributos comunes	i18n
Atributos específicos	<p>name = "texto" – El nombre de la propiedad que se define (no existe una lista oficial de propiedades)</p> <p>content = "texto" – El valor de la propiedad definida (no existe una lista de valores permitidos)</p> <p>http-equiv = "texto" – En ocasiones, reemplaza al atributo "name" y lo emplean los servidores para adaptar sus respuestas al documento</p> <p>scheme = "texto" – Indica el esquema que se debe emplear para interpretar el valor de la propiedad</p>
Tipo de elemento	–
Descripción	Permite definir el valor de los metadatos que forman la metainformación del documento

10 MAS ETIQUETAS

10.1 Comentarios

Al igual que la mayoría de lenguajes de marcado, HTML permite incluir comentarios dentro de su código para añadir información que no se debe mostrar por pantalla.

Normalmente, los diseñadores y programadores incluyen comentarios para marcar el comienzo y el final de las secciones de las páginas, para incluir avisos y notas para otros diseñadores o para incluir explicaciones sobre la forma en la que se ha creado el código HTML.

Aunque los comentarios no se muestran por pantalla y por tanto son invisibles para los usuarios, sí que se descargan con el código HTML de la página. Por este motivo, nunca debe incluirse información sensible o confidencial en los comentarios.

La sintaxis de los comentarios es la siguiente:

Apertura del comentario: <!--

Contenido del comentario: (cualquier texto)

Cierre del comentario: -->

Los comentarios de HTML puede ocupar varias líneas. Sin embargo, los comentarios no se pueden anidar, es decir, no se puede incluir un comentario dentro de otro comentario.

10.2 JavaScript

Como ya se explicó en los capítulos anteriores, la etiqueta <script> se utiliza para enlazar archivos JavaScript externos y para incluir bloques de código JavaScript en las páginas. Sin embargo, algunos navegadores no disponen de soporte completo de JavaScript, otros navegadores permiten bloquearlo parcialmente e incluso algunos usuarios bloquean completamente el uso de JavaScript porque creen que así navegan de forma más segura.

Si JavaScript está bloqueado o deshabilitado y la página web requiere su uso para un correcto funcionamiento, es habitual incluir un mensaje de aviso al usuario indicándole que debería activar JavaScript para disfrutar completamente de la página. HTML define la etiqueta <noscript> para incluir un mensaje que los navegadores muestran cuando JavaScript se encuentra bloqueado o deshabilitado.

<noscript>	Sin soporte de scripts
Atributos comunes	básicos, i18n y eventos
Atributos específicos	–
Tipo de elemento	Bloque
Descripción	Define un mensaje alternativo que se muestra al usuario cuando su navegador no soporta la ejecución de scripts

Veamos un ejemplo ([Ej10.02-NoScript.html](#)):

```
<html>
    <head><title>Ejemplo NoScript</title>
        <meta charset="UTF-8" /></head>
    <body>
        <noscript>
            <p>Bienvenido a Mi Sitio</p>
            <p>La página que estás viendo requiere para su funcionamiento el uso de JavaScript.
                Si lo has deshabilitado intencionadamente, por favor vuelve a activarlo.</p>
        </noscript>
    </body></html>
```

10.3 CSS

Algunos de los atributos más utilizados en la creación de páginas web son id, class y style. Los tres atributos están muy relacionados con CSS, sobre todo class y style.

El atributo id se emplea para asignar un identificador único a cada elemento de la página, lo que es útil tanto para aplicar estilos CSS a ese elemento como para programar aplicaciones con JavaScript.

Por otra parte, el atributo class se emplea para definir la clase CSS que se aplica a un elemento. La clase CSS es el nombre de un conjunto de estilos que se definen en la hoja de estilos y que se quieren aplicar a un elemento:

```
<p class="resumen">Lorem ipsum dolor sit amet, consectetuer adipiscing elit.  
Maecenas at diam id enim viverra semper. Nulla id urna. Donec sodales.</p>
```

El párrafo del ejemplo anterior se muestra por pantalla con el aspecto definido por el conjunto de estilos llamado resumen y que se define en la hoja de estilos CSS enlazada por la página web.

El atributo style se emplea para definir estilos CSS directamente sobre los elementos HTML, tal y como se muestra en el siguiente ejemplo: ([Ej10.03a-EjCSS.html](#)):

```
<html>  
  <head>  
    <title>Ejemplo CSS</title>  
    <meta charset="UTF-8" />  
  </head>  
  <body>  
    <p>Algunas palabras de esta frase se muestran de  
      <span style="color:red">color rojo</span>  
    </p>  
  </body>  
</html>
```

No se debe confundir el atributo style con la etiqueta `<style>` que se explicó anteriormente. La etiqueta `<style>` se utiliza para incluir bloques de código CSS ([Ej10.03b-OtroCSS.html](#)):

```
<html>  
  <head>  
    <title>Ejemplo CSS</title>  
    <meta charset="UTF-8" />  
    <style type="text/css">  
      span {color:red;}  
    </style>  
  </head>  
  <body>  
    <p><span>En este caso, TODAS las palabras están en rojo</span>  
    </p>  
  </body>  
</html>
```

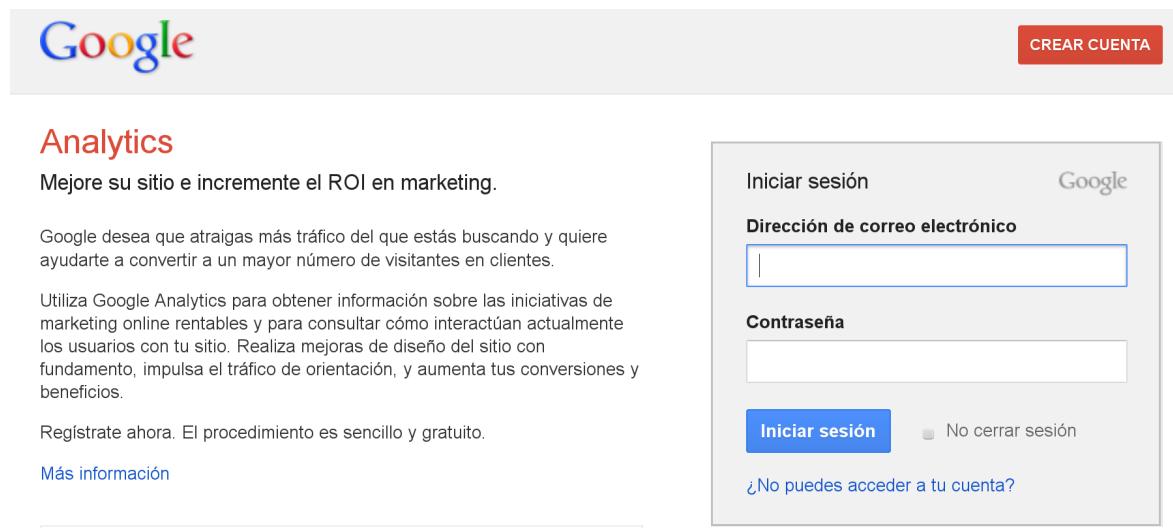
10.4 Iframes

Aunque su uso no es muy común, la etiqueta <iframe> puede ser muy útil en determinadas ocasiones, ya que permite insertar un documento HTML dentro de otro documento HTML. Un iframe puede considerarse como un agujero que se abre en una página web y a través del cual se muestra otra página web.

En ocasiones se utiliza para mostrar contenidos externos al sitio web como si fueran parte del mismo sitio. Otra veces se emplea para incluir una aplicación común a varios sitios web de una misma empresa.

La página principal de Google Analytics emplea un <iframe> para incluir en un pequeño recuadro la página correspondiente a la validación de usuario:

<https://accounts.google.com/ServiceLogin?service=analytics>



The screenshot shows the Google Analytics login interface. At the top, there's a navigation bar with the Google logo and a 'CREAR CUENTA' button. Below it, the 'Analytics' logo is displayed with the tagline 'Mejore su sitio e incremente el ROI en marketing.' A descriptive paragraph explains that Google wants to attract more traffic from the user's site. It encourages users to use Google Analytics to get information about online marketing initiatives and to consult how users interact with their site. It also mentions that Google helps users improve site design, which in turn drives traffic and increases conversion rates. A registration link is provided for those who haven't registered yet. On the right side, there's a login form with fields for 'Iniciar sesión' (Email) and 'Contraseña' (Password), a 'Iniciar sesión' button, a 'No cerrar sesión' checkbox, and a link for users who can't access their account.

<iframe>	Marco (frame) en línea
Atributos comunes	básicos
Atributos específicos	<code>src = "url"</code> - URL del documento HTML que se visualiza en el iframe <code>height = "longitud"</code> - Altura que ocupará el iframe en el documento <code>width = "longitud"</code> - Anchura que ocupará el iframe en el documento <code>name = "texto"</code> - Nombre que identifica al iframe <code>longdesc = "url"</code> - Dirección en la que puede encontrarse una descripción larga del contenido del iframe <code>scrolling = "yes no auto"</code> - Indica si el iframe debe mostrar barras de scroll (horizontal y vertical) cuando el contenido incluido no cabe en el iframe
Tipo de elemento	Bloque y en línea
Descripción	Se emplea para incluir en la página un marco que muestra otro documento HTML

El siguiente ejemplo define la altura y anchura del iframe, indica la URL que se debe mostrar y mediante el atributo scrolling se indica que el iframe no debe mostrar barras de scroll ni siquiera en el caso de que el contenido mostrado no quepa en el iframe definido, además de no incluir un marco. El archivo se llamará **Ej10.04-Iframe.html**:

NOTA: Este caso es muy usado en multitud de webs para indicar la localización.

Nos vamos a <http://maps.google.es/>
Realizamos una búsqueda. Al localizar el sitio, pulsamos el botón Enlazar (marcado en rojo) y copiamos el texto que aparece en Pegar HTML para insertar en sitio web (en azul).



```
<html>
  <head>
    <title>Ejemplo CSS</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <h1>Ejemplo de Iframe</h1>
    <iframe width="425" height="350" frameborder="0" scrolling="no" marginheight="0"
marginwidth="0" src="http://maps.google.es/maps?
f=q&source=s_q&hl=es&geocode=&q=Calle+Alfonso+de+Coss%20Ado,
+Sevilla&aq=0&oq=Calle+Alfonso+de+Coss%20Ado,&sll=37.376543,-
5.981345&sspn=0.001227,0.002064&t=w&ie=UTF8&hq=&hnear=Calle+Alfonso+de+
Coss%20Ado,+41004+Sevilla&ll=37.376491,-
5.980617&spn=0.001227,0.002064&z=14&output=embed"></iframe><br /><small><a
href="http://maps.google.es/maps?
f=q&source=embed&hl=es&geocode=&q=Calle+Alfonso+de+Coss%20Ado,
+Sevilla&aq=0&oq=Calle+Alfonso+de+Coss%20Ado,&sll=37.376543,-
5.981345&sspn=0.001227,0.002064&t=w&ie=UTF8&hq=&hnear=Calle+Alfonso+de+
Coss%20Ado,+41004+Sevilla&ll=37.376491,-5.980617&spn=0.001227,0.002064&z=14"
style="color:#0000FF;text-align:left">Ver mapa más grande</a></small>
  </body>
</html>
```

10.5 Otras Etiquetas

La etiqueta `<address>` es una de las etiquetas más desconocidas de HTML, por lo que uso no está muy extendido. La etiqueta `<address>` se utiliza para proporcionar información de contacto.

<address>	Direcciones
Atributos comunes	básicos, i18n y eventos
Atributos específicos	–
Tipo de elemento	Bloque
Descripción	Define la información de contacto de un documento

El siguiente ejemplo sencillo muestra directamente el nombre, dirección y teléfono de contacto de una empresa:

```
<address>
    Nombre de la empresa
    Dirección completa
    Teléfono y Fax
</address>
```

Veamos un ejemplo (que cada uno ponga su propia información) [Ej10.05a-Address.html](#):

```
<html>
    <head>
        <title>Ejemplo Address</title>
        <meta charset="UTF-8" />
    </head>
    <body>
        <h1>Información de Contacto</h1>
        <address>
            Autor: Iván Rodríguez Ruiz<br />
            <a href="http://www.ivanrguez.es">http://www.ivanrguez.es</a>
        </address>
    </body>
</html>
```

Hasta hace unos años, la etiqueta `<hr>` era una de las más utilizadas, ya que permite mostrar una línea horizontal de separación. Sin embargo, hoy en día apenas se utiliza, ya que se considera un elemento puramente estético, del que no debería preocuparse HTML y para el que CSS ofrece alternativas mucho mejores.

<hr>	Línea horizontal
Atributos comunes	básicos, i18n y eventos
Atributos específicos	–
Tipo de elemento	Bloque
Descripción	Permite incluir una línea horizontal de separación

Veamos un ejemplo [Ej10.05b-LineaHR.html](#):

```
<html>
    <head><title>Ejemplo HR</title>
    <meta charset="UTF-8" /></head>
    <body>
        <h1>Información de Contacto</h1>
        <p>El primer párrafo de texto del documento</p>
        <hr/>
        <p>Este es el segundo párrafo de texto del documento</p>
    </body>
</html>
```

Esta etiqueta tiene una serie de atributos que han sido marcados como obsoletos en el estándar. Son los siguientes:

noshade

Cuando este atributo booleano está presente, la regla es dibujada con una línea sólida en lugar de utilizar el efecto 3D predeterminado. Recuerda que XHTML requiere que los atributos booleanos se declaren con sus nombres como valores.

```
<hr noshade="noshade" />
```

size (pixels)

Este atributo especifica la altura de la regla en píxeles. Su valor predeterminado depende directamente del navegador.

```
<hr size="3" />
```

width (length)

Especifica el ancho de la regla. El valor predeterminado es 100%.

```
<hr width="80%" />
```

11 ACCESIBILIDAD Y VALIDACIÓN

El principal objetivo de la accesibilidad web es el de permitir a cualquier usuario, independientemente del tipo de discapacidad que sufra, el acceso a los contenidos del sitio y permitirle la navegación necesaria para realizar las acciones deseadas.

Los sitios web accesibles no solamente facilitan el acceso de sus contenidos a los usuarios discapacitados, sino que también permiten ofrecer la misma funcionalidad con dispositivos muy limitados (dispositivos sin pantalla o con pantallas minúsculas, dispositivos sin teclado ni ratón, etc.).

Las cuatro principales ventajas de diseñar un sitio web completamente accesible son las siguientes:

- Los sitios accesibles separan completamente diseño y contenido.
- Un sitio accesible puede ser accedido por multitud de dispositivos diferentes sin necesidad de reescribir el código HTML.
- Los sitios accesibles son los únicos con una audiencia potencial global, ya que permiten el acceso a todos los usuarios y a todos los dispositivos.
- Generalmente, los sitios accesibles son más fáciles de utilizar también para los usuarios sin discapacidades.

La creación de sitios accesibles puede realizarse siguiendo las recomendaciones establecidas por el W3C y que se recogen en el documento Web Content Accessibility Guidelines (WCAG).

La versión WCAG 1.0 que se utiliza en la actualidad se publicó en 1999, mientras que la versión WCAG 2.0 se encuentra todavía en borrador y se actualizó por última vez el día 30 de abril de 2008.

Las recomendaciones del WCAG 1.0 están formadas por 65 requisitos que un sitio web debe cumplir para considerarse accesible. Los requerimientos se agrupan en prioridades.

Los requisitos de prioridad 1 son de obligado cumplimiento, los de prioridad 2 son recomendables y los de prioridad 3 son deseables. Si un sitio web cumple con todos los requisitos de prioridad 1, se considera que el sitio es conforme al nivel A de accesibilidad.

El nivel AA de accesibilidad está reservado para los sitios que cumplen todos los requisitos de prioridad 1 y prioridad 2. Finalmente, los sitios que cumplen los 65 requisitos, son conformes al nivel AAA de accesibilidad.

11.1 Requisitos del Nivel A de Accesibilidad

Los requisitos de accesibilidad que exige el nivel A son los siguientes:

11.1.1 Generales

- Proporcionar un texto alternativo para todas las imágenes, objetos y otros elementos no textuales (mediante los atributos alt y longdesc).
- Asegurar que toda la información que utilice el color como elemento informativo pueda ser entendida por las personas o dispositivos que no pueden distinguir los colores.
- Marcar claramente (mediante los atributos lang y xml:lang) las variaciones del idioma del texto o de los elementos textuales (<caption>) respecto del idioma principal de la página.
- El documento debe poder leerse completamente cuando no se utilicen hojas de estilos.
- La información equivalente para los contenidos dinámicos debe adaptarse a los cambios de los contenidos dinámicos.
- Ningún elemento debe parpadear en la pantalla.
- El contenido del sitio se debe escribir con un lenguaje sencillo y limpio.

11.1.2 Si se utilizan mapas de imagen

- Proporcionar un enlace textual por cada una de las regiones del mapa de imagen
- Utilizar mapas de imagen en el cliente, en vez de mapas de imagen de servidor.

11.1.3 Si se utilizan tablas

- Utilizar cabeceras de fila y de columna.
- Si la tabla tiene varios niveles de cabeceras, utilizar las agrupaciones disponibles (<thead>, <tfoot>).

11.1.4 Si se utilizan frames

- Indicar un título a cada frame para su identificación y facilitar la navegación.

11.1.5 Si se utilizan applets y scripts.

- Asegurar que la página también se pueda utilizar cuando no se ejecutan los applets y los scripts. Si no es posible, proporcionar informaciones equivalentes o páginas alternativas que sean accesibles.

11.1.6 Si se utilizan contenidos multimedia (audio y vídeo).

- Incluir una descripción textual del contenido multimedia.
- Para los contenidos basados en vídeo o animaciones, sincronizar las alternativas textuales con la presentación.

11.1.7 Si no se pueden cumplir los anteriores requisitos

- Proporcionar una página alternativa con la mayor cantidad posible de contenidos y que cumpla con los requisitos anteriores.

La lista completa con los 65 requisitos de los tres niveles de accesibilidad se puede consultar en <http://www.w3.org/TR/WCAG10/full-checklist.html>

11.2 Validadores W3C

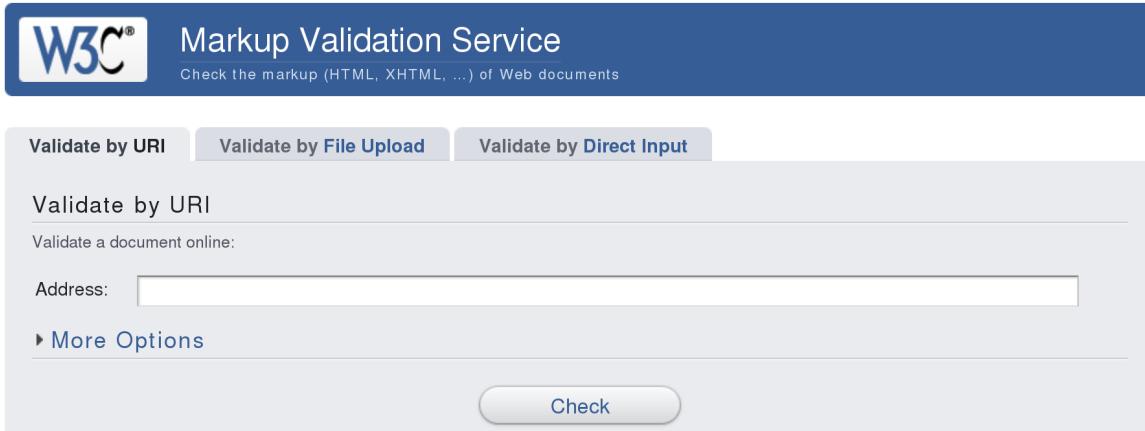
La validación es el proceso que asegura que un documento escrito en un determinado lenguaje (por ejemplo XHTML) cumple con las normas y restricciones de ese lenguaje. Las normas y restricciones de los documentos escritos en XML (y en sus lenguajes derivados, como XHTML) se definen en el DTD o Document Type Definition ("Definición del Tipo de Documento").

El concepto de validación es objeto de controversia en el ámbito del diseño web. Por una parte, la validación no es obligatoria y las páginas web se pueden ver bien sin que sean válidas. Por otra parte, una página válida es más correcta que otra página que no lo sea, ya que cumple con las normas y restricciones impuestas por XHTML.

Debido a esta controversia, algunos diseñadores consideran que se da demasiada importancia a la validación de las páginas y a la creación de páginas válidas. El resto de diseñadores argumentan que si la especificación de XHTML impone una serie de normas y restricciones, lo más correcto es que las páginas web las cumplan, aunque no sea obligatorio.

En cualquier caso, el proceso de validación consiste en probar página a página si su código HTML pasa la prueba de validación. Los validadores son las herramientas que se utilizan para validar cada página. Algunos editores de páginas web incluyen sus propios validadores y el organismo W3C ha creado una herramienta gratuita para la validación de las páginas.

La validación de las páginas web no requiere el uso de editores avanzados, ya que el organismo W3C ha creado una herramienta que se puede usar gratuitamente: <http://validator.w3.org/>



Aunque la herramienta sólo está disponible en inglés, su uso es muy intuitivo:

- Validate by URI, permite escribir la URL de la página que se quiere validar. Esta opción es la más sencilla para validar las páginas que ya están publicadas en Internet.
- Validate by File Upload, muestra un formulario mediante el que se puede subir el archivo HTML correspondiente a la página que se quiere validar. Esta opción es la mejor para validar las páginas web que has desarrollado y que aún no has publicado en Internet.
- Validate by Direct Input, permite validar código HTML de forma directa. Se trata de la opción más rápida para validar trozos o páginas HTML completas. Esta opción es la mejor cuando estás desarrollando las páginas y quieres asegurarte que el código sea válido.

Si la página no pasa correctamente la prueba de validación, se muestra el listado completo de fallos junto con la ayuda necesaria para resolver cada uno de los errores.

11.3 Otros Validadores

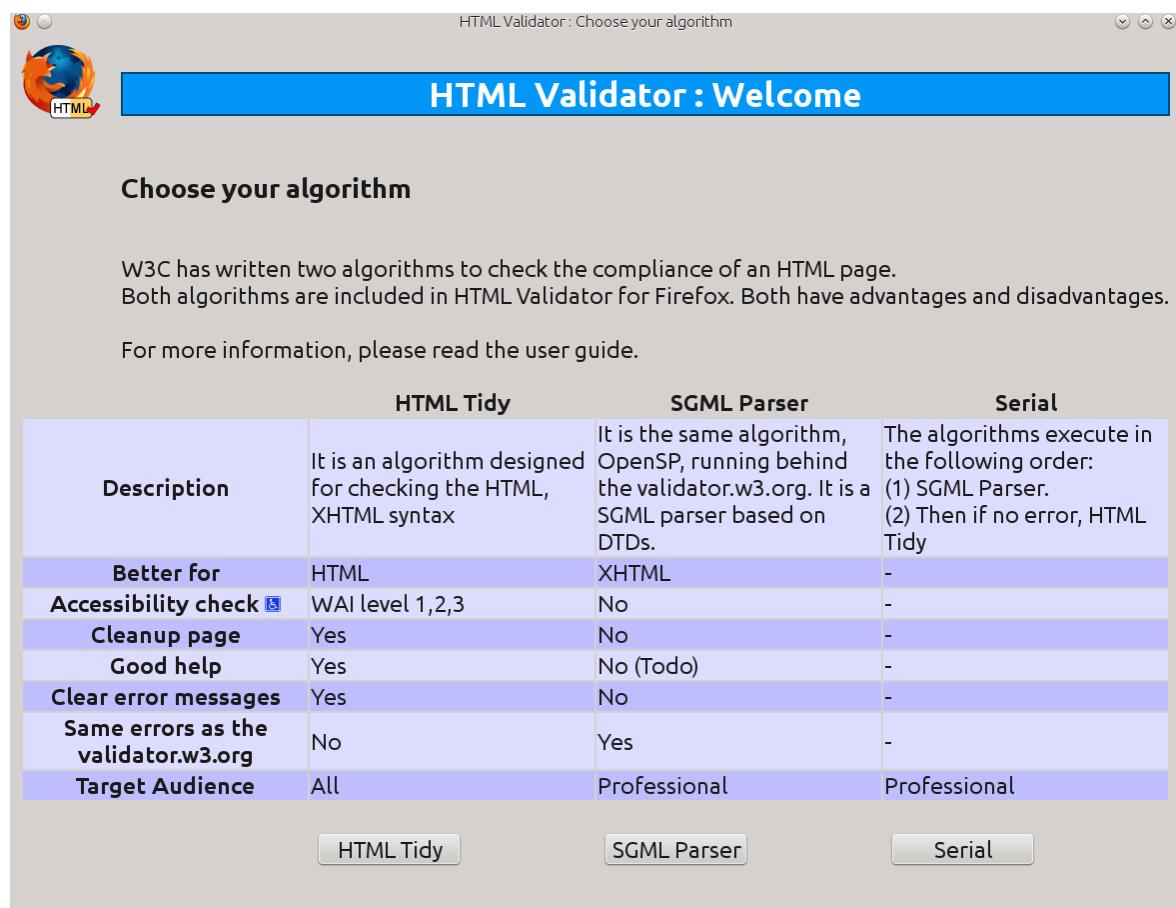
Además de los validadores disponibles en herramientas como Dreamweaver y de los validadores gratuitos disponibles en Internet, existe otro método de validación sencillo, gratuito y muy rápido. La única limitación de este validador es que necesariamente se debe utilizar el navegador Firefox.

Si ya dispones del navegador Firefox, puedes instalar el validador mediante un complemento llamado HTML Validator. La instalación se realiza como cualquier otro complemento, aunque en este caso la descarga dura un poco más de lo normal porque ocupa más de 2 MB.

La dirección es la siguiente:

http://users.skynet.be/mgueury/mozilla/download_090.html

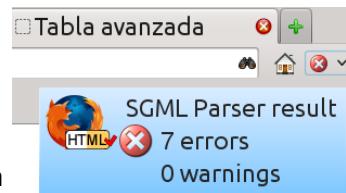
Tras su instalación, la primera vez que se reinicia Firefox se muestra la siguiente ventana:



En la ventana que se muestra se solicita al usuario que configure el tipo de validación que se va a realizar. Las opciones para elegir son: HTML Tidy (que ofrece ayuda para resolver los errores y es mejor para HTML), SGML Parser (ofrece menos ayuda, pero es el mismo que el validador del W3C) o Serial (que realiza las dos validaciones de forma seguida).

Si no sabes cual elegir, la opción Serial es una buena alternativa, ya que primero realiza la validación SGML Parser y a continuación, si no se han producido errores, realiza la validación HTML Tidy.

Una vez configurado el validador, abre cualquier página web y verás cómo en la esquina inferior derecha de Firefox (en las nuevas versiones estará en la esquina superior derecha) se muestra un pequeño ícono que indica si la página es válida o no. Cuando la página no es válida, aparece un ícono correspondiente a un error. Si colocas el puntero del ratón sobre el ícono, se muestra la información específica sobre los errores encontrados:



Si pulsas dos veces sobre el ícono, aparece una nueva ventana en la que se muestra la lista completa de errores, el lugar exacto en el que se han producido y las posibles soluciones para corregirlos.

Para ver directamente el número de errores de la página, puedes pulsar el botón derecho del ratón sobre el ícono del validador y seleccionar la opción Show y después Icon and Text. Después de activar esta opción, cada vez que cargues una página web se muestra toda la información de validación.

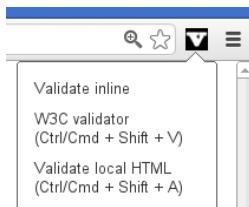
Aunque existen muchos otros validadores, el uso de Firefox junto con Html Validator es la única alternativa que permite validar las páginas web sin esfuerzo. Abriendo cualquier página en el navegador Firefox, es posible visualizar al instante si la página es válida o no y el número de errores que se han encontrado.

Otro validador en local sería con Google Chrome:

<http://robertnyman.com/html-validator/>

La dirección de la descarga es:

<https://chrome.google.com/webstore/detail/html-validator/cgnfbhngibokieehnjhbjkkhbfmhojo>



12 MARCADO SEMÁNTICO EN HTML5

12.1 Estructura básica

El primer elemento a incluir en nuestras páginas será el tipo de documento que se realiza mediante el DOCTYPE con la siguiente línea:

```
<!DOCTYPE html>
```

Esta línea debe ir al principio de documento sin nada (espacios, caracteres, etc) delante.

Después incluiremos la etiqueta ya vista html pero añadiendo el atributo lang con el valor "es" para indicar que el documento está construido en español.

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

Posteriormente añadimos las etiquetas <head> y <body> que ya estudiamos junto al <meta> que indica el conjunto de caracteres a añadir junto a otras características. Además, podemos añadir la etiqueta <link> que se mantiene igual pero quitando el atributo type. Así

Por tanto nuestro primer ejemplo quedará ([Ej12.01a-Doctype.html](#)):

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8" />
    <meta name="description" content="Mi primer ejemplo HTML5" />
    <title>Mi Primer HTML5</title>
    <link rel="stylesheet" href="comun.css" />
  </head>
  <body>
    <p>Algunas palabras de esta frase se muestran de
       <span>Color Azul</span>
    </p>
  </body>
</html>
```

Por otro lado tendremos el archivo [Comun.css](#)

```
span {
  color:blue;
}
```

12.2 Estructura del cuerpo

En temas anteriores hemos visto como estructurar una página mediante tablas (en las primeras versiones de HTML) o divisiones (a partir de la versión 4).

En HTML las <div> dan paso a un conjunto de etiquetas mucho más "semánticas".

Un ejemplo muy común dentro de las páginas webs actuales son los blogs. A partir de un ejemplo veremos las distintas partes que componen el <body> del documento:

- 1) Cabecera o <header>
- 2) Barra de navegación o <nav>
- 3) Sección de Información principal o <section>
- 4) Barra Lateral o <aside>
- 5) Pie de página o <footer>

Estos cinco componentes nos adentran en el marcado semántico, que emplea las etiquetas siguientes: <header>, <nav>, <section>, <aside>, <footer>, <article>, <hgroup>, <figure> y <figcaption>



12.2.1 El elemento <header>

Este elemento no debe ser confundido con <head>. Mientras este último se refiere al documento y no se presenta, <header> muestra información introductoria al cuerpo del documento, como títulos o logos. Veamos un fragmento de código:

```
<header>
  <h1>Este es el título principal del sitio web</h1>
</header>
```

12.2.2 El elemento <nav>

Siguiendo con nuestro ejemplo, la siguiente sección es la Barra de Navegación. Esta barra es generada en HTML5 con el elemento <nav>. Un ejemplo: [Ej12.02a-Nav.html](#):

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>
    <nav>
      <ul>
        <li>principal</li>
        <li>fotos</li>
        <li>videos</li>
        <li>contacto</li>
      </ul>
    </nav>
  </body>
</html>
```

Como se puede apreciar en el Listado 1-11, el elemento <nav> se encuentra dentro de las etiquetas <body> pero es ubicado después de la etiqueta de cierre de la cabecera (</header>), no dentro de las etiquetas <header>. Esto es porque <nav> no es parte de la cabecera sino una nueva sección.

12.2.3 El elemento <section>

La columna Información Principal contiene la información más relevante del documento y puede ser encontrada en diferentes formas (por ejemplo, dividida en varios bloques o columnas). Debido a que el propósito de estas columnas es más general, el elemento en HTML5 que especifica estas secciones se llama simplemente <section>.

Al igual que la Barra de Navegación, la columna Información Principal es una sección aparte. Por este motivo, la sección para Información Principal va debajo de la etiqueta de cierre </nav>.

```
....<nav>....</nav><section>....</section></body></html>
```

12.2.4 El elemento <aside>

La Barra Lateral se ubica al lado de la columna Información Principal. Esta es una columna o sección que normalmente contiene datos relacionados con la información principal pero que no son relevantes o igual de importantes.

En el diseño de un blog, por ejemplo, la Barra Lateral contendrá una lista de enlaces. La información dentro de esta barra está relacionada con la información principal pero no es relevante por sí misma. En HTML5 podemos diferenciar esta información secundaria usando <aside>:

```
....  
    <section>  
    ....  
    </section>  
    <aside>  
        <blockquote>Mensaje número uno</blockquote>  
        <blockquote>Mensaje número dos</blockquote>  
    </aside>  
    </body>  
</html>
```

El elemento <aside> podría estar ubicado del lado derecho o izquierdo de nuestra página de ejemplo, la etiqueta no tiene una posición predefinida. El elemento <aside> solo describe la información que contiene, no el lugar dentro de la estructura.

Este elemento puede estar ubicado en cualquier parte del diseño y ser usado siempre y cuando su contenido no sea considerado como el contenido principal del documento. Por ejemplo, podemos usar <aside> dentro del elemento <section> o incluso insertado entre la información relevante, como en el caso de una cita.

12.2.5 El elemento <footer>

Lo único que nos queda por hacer es cerrar nuestro diseño para otorgarle un final al cuerpo del documento. Para ello usaremos <footer>. Un ejemplo: [Ej12.02b-Footer.html](#)):

```
<!DOCTYPE html>  
<html lang="es">  
    <head>  
        <meta charset="UTF-8">  
        <title>Este texto es el título del documento</title>  
        <link rel="stylesheet" href="misestilos.css">  
    </head>  
    <body>  
        <header>  
            <h1>Este es el título principal del sitio web</h1>  
        </header>  
        <nav>  
            <ul>  
            </ul>  
        </nav>  
        <section>  
            ....  
        </section>  
        <aside>  
            <blockquote>Mensaje número uno</blockquote>  
            <blockquote>Mensaje número dos</blockquote>  
        </aside>  
        <footer>  
            Derechos Reservados © 2010-2011  
        </footer>  
    </body>  
</html>
```

Generalmente, <footer> representará el final del cuerpo de nuestro documento. Sin embargo, <footer> puede ser usado múltiples veces dentro del cuerpo para representar también el final de diferentes secciones (del mismo modo que la etiqueta <header>).

12.3 Dentro del cuerpo

Ya hemos visto la estructura básica de un página web en HTML5, usando etiquetas semánticas. De todos modos vamos a adentrarnos mas profundamente en la sección de Información principal, que está dentro del elemento <section>.

12.3.1 El elemento <article>

Del mismo modo que los blogs están divididos en entradas, sitios web normalmente presentan información relevante dividida en partes que comparten similares características. El elemento <article> nos permite identificar cada una de estas partes.

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    ...
    <section>
      <article>
        Este es el texto de mi primer mensaje
      </article>
      <article>
        Este es el texto de mi segundo mensaje
      </article>
    </section>
    ...
  </body>
</html>
```

Las etiquetas <article> se encuentran ubicadas dentro del elemento <section>. Las etiquetas <article> en nuestro ejemplo pertenecen a esta sección, son sus hijos, del mismo modo que cada elemento dentro de las etiquetas <body> es hijo del cuerpo.

Es importante reseñar que la estructura de un documento HTML puede ser descripta como un árbol, con el elemento <html> como su raíz. Otra forma de describir la relación entre elementos es nombrarlos como padres, hijos y hermanos, de acuerdo a la posición que ocupan dentro de esa misma estructura.

Por ejemplo, en un típico documento HTML el elemento <body> es hijo del elemento <html> y hermano del elemento <head>. Ambos, <body> y <head>, tienen al elemento <html> como su padre.

El elemento <article> no está limitado por su nombre (no se limita, por ejemplo, a artículos de noticias). Este elemento fue creado con la intención de contener unidades independientes de contenido, por lo que puede incluir mensajes de foros, artículos de una revista digital, entradas de blog, comentarios de usuarios, etc... Lo que hace es agrupar porciones de información que están relacionadas entre sí independientemente de su naturaleza.

Como una parte independiente del documento, el contenido de cada elemento <article> tendrá su propia estructura. Para definir esta estructura, podemos aprovechar la versatilidad de los elementos <header> y <footer> estudiados anteriormente. Estos elementos son portables y pueden ser usados no solo para definir los límites del cuerpo sino también en cualquier sección de nuestro documento.

Un ejemplo completo: **Ej12.03a-Article.html**:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <meta name="description" content="Ejemplo de HTML5">
    <meta name="keywords" content="HTML5, CSS3, JavaScript">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>
    <nav>
      <ul>
        <li>principal</li>
        <li>fotos</li>
        <li>videos</li>
        <li>contacto</li>
      </ul>
    </nav>
    <section>
      <article>
        <header>
          <h1>Título del mensaje uno</h1>
        </header>
        Este es el texto de mi primer mensaje
        <footer>
          <p>comentarios (0)</p>
        </footer>
      </article>
      <article>
        <header>
          <h1>Tituto del mensaje dos</h1>
        </header>
        Este es el texto de mi segundo mensaje
        <footer>
          <p>comentarios (0)</p>
        </footer>
      </article>
    </section>
    <aside>
      <blockquote>Mensaje número uno</blockquote>
      <blockquote>Mensaje número dos</blockquote>
    </aside>
    <footer>
      Derechos Reservados © 2010-2011
    </footer>
  </body>
</html>
```

Los dos mensajes insertados en el código fueron construidos con el elemento `<article>` y tienen una estructura específica. En la parte superior de esta estructura incluimos las etiquetas `<header>` conteniendo el título definido con el elemento `<h1>`, debajo se encuentra el contenido mismo del mensaje y sobre el final, luego del texto, vienen las etiquetas `<footer>` especificando la cantidad de comentarios recibidos.

12.3.2 El elemento <hgroup>

Este elemento sirve para agrupar secciones del documento creadas con las etiquetas h1...h6. Veamos un ejemplo completado el ejemplo anterior ([Ej12.03a-Article.html](#)):

```
...
<section>
  <article>
    <header>
      <hgroup>
        <h1>Título del mensaje uno</h1>
        <h2>Subtítulo del mensaje uno</h2>
      </hgroup>
      <p>publicado 10-12-2011</p>
    </header>
    Este es el texto de mi primer mensaje
    <footer>
      <p>comentarios (0)</p>
    </footer>
  </article>
  <article>
    <header>
      <hgroup>
        <h1>Título del mensaje dos</h1>
        <h2>Subtítulo del mensaje dos</h2>
      </hgroup>
      <p>publicado 15-12-2011</p>
    </header>
    Este es el texto de mi segundo mensaje
    <footer>
      <p>comentarios (0)</p>
    </footer>
  </article>
</section>
...
```

Las etiquetas H deben conservar su jerarquía, lo que significa que debemos primero declarar la etiqueta <h1>, luego usar <h2> para subtítulos y así sucesivamente. Sin embargo, a diferencia de anteriores versiones de HTML, HTML5 nos deja reusar las etiquetas H y construir esta jerarquía una y otra vez en cada sección del documento.

Es importante reseñar que hgroup solo se usa cuando tenemos mas de un elemento <h(x)> dentro de la cabecera del artículo. Es decir, sólo se emplea para agrupar elementos <h(x)>

Navegadores y programas que ejecutan y presentan en la pantalla sitios webs leen el código HTML y crean su propia estructura interna para interpretar y procesar cada elemento. Esta estructura interna está dividida en secciones que no tienen nada que ver con las divisiones en el diseño o el elemento <section>. Estas son secciones conceptuales generadas durante la interpretación del código.

El elemento <header> no crea una de estas secciones por sí mismo, lo que significa que los elementos dentro de <header> representarán diferentes niveles e internamente pueden generar diferentes secciones.

El elemento <hgroup> fue creado con el propósito de agrupar las etiquetas H y evitar interpretaciones incorrectas por parte de los navegadores.

12.3.3 Elementos <figure> y <figcaption>

La etiqueta <figure> fue creada para ayudarnos a ser aún más específicos a la hora de declarar el contenido del documento. Antes de que este elemento sea introducido, no podíamos identificar el contenido que era parte de la información pero a la vez independiente, como ilustraciones, fotos, videos, etc... Normalmente estos elementos son parte del contenido relevante pero pueden ser extraídos o movidos a otra parte sin afectar o interrumpir el flujo del documento. Cuando nos encontramos con esta clase de información, las etiquetas <figure> pueden ser usadas para identificarla:

```
...
<article>
  <header>
    <hgroup>
      <h1>Título del mensaje uno</h1>
      <h2>Subtítulo del mensaje uno</h2>
    </hgroup>
    <p>publicado 09-01-2013</p>
  </header>
  Este es el texto de mi primer mensaje
  <figure>
    
    <figcaption>
      Esta es la imagen del primer mensaje
    </figcaption>
  </figure>
  <footer>
    <p>comentarios (0)</p>
  </footer>
</article>
...
```

Esta es una práctica común, a menudo el texto es enriquecido con imágenes o videos. Las etiquetas <figure> nos permiten envolver estos complementos visuales y diferenciarlos así de la información más relevante.

Por otro lado, normalmente, unidades de información como imágenes o videos son descriptas con un corto texto debajo. HTML5 provee un elemento para ubicar e identificar estos títulos descriptivos. Las etiquetas <figcaption> encierran el texto relacionado con <figure> y establecen una relación entre ambos elementos y su contenido.

12.4 Nuevos Elementos

HTML5 fue desarrollado con la intención de simplificar, especificar y organizar el código. Para lograr este propósito, nuevas etiquetas y atributos fueron agregados y HTML fue completamente integrado a CSS y Javascript. Estas incorporaciones y mejoras de versiones previas están relacionadas no solo con nuevos elementos sino también con cómo usamos los ya existentes.

12.4.1 El elemento <mark>

La etiqueta <mark> fue agregada para resaltar parte de un texto que originalmente no era considerado importante pero ahora es relevante acorde con las acciones del usuario. El ejemplo que más se ajusta a este caso es un resultado de búsqueda. El elemento <mark> resaltará en **amarillo la parte del texto que ha sido marcado**. Un ejemplo: [Ej12.04-Elementos.html](#):

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="UTF-8">
    <title>Este texto es el título del documento</title>
    <link rel="stylesheet" href="misestilos.css">
  </head>
  <body>
    <header>
      <h1>Este es el título principal del sitio web</h1>
    </header>
    <nav>
    </nav>
    <section>
      <article>
        <header>
          <h1>Ejemplo de Entrada</h1>
        </header>
        <span>Mi <mark>coche</mark> es negro</span>
      </article>
    </section>
  </body>
</html>
```

12.4.2 El elemento <small>

La nueva especificidad de HTML es también evidente en elementos como <small>. Previamente este elemento era utilizado con la intención de presentar cualquier texto con letra pequeña. La palabra clave referenciaba el tamaño del texto, independientemente de su significado. En HTML5, el nuevo propósito de <small> es presentar la llamada letra pequeña, como impresiones legales, descargas, etc.... Veamos un ejemplo (seguimos con [Ej12.04-Elementos.html](#)):

```
...
  <section>
    <article>
      <header>
        <h1>Ejemplo de Entrada</h1>
      </header>
      <span>Mi <mark>coche</mark> es negro</span><br />
      <small>Derechos Reservados &copy; 2013 Iván Rodríguez</small>
    </article>
  </section>
</body>
</html>
```

12.4.3 El elemento <cite>

Otro elemento que ha cambiado su naturaleza para volverse más específico es <cite>. Ahora las etiquetas <cite> encierran el título de un trabajo, como un libro, una película, una canción, etc... Veamos un ejemplo (seguimos con [Ej12.04-Elementos.html](#)):

```
...
<section>
  <article>
    <header>
      <h1>Ejemplo de Entrada</h1>
    </header>
    <span>Mi <mark>coche</mark> es negro</span><br />
    <small>Derechos Reservados © 2013 Iván Rodríguez</small><br />
    <span>Me encanta la película <cite>Blade Runner</cite></span>
  </article>
</section>
</body>
</html>
```

12.4.4 El elemento <address>

El elemento <address> es un viejo elemento convertido en un elemento estructural. No necesitamos usarlo previamente para construir nuestra plantilla, sin embargo podría ubicarse perfectamente en algunas situaciones en las que debemos presentar información de contacto relacionada con el contenido del elemento <article> o el cuerpo completo. Este elemento debería ser incluido dentro de <footer>. Un ejemplo (seguimos con [Ej12.04-Elementos.html](#)):

```
...
<section>
  <article>
    <header>
      <h1>Ejemplo de Entrada</h1>
    </header>
    <span>Mi <mark>coche</mark> es negro</span><br />
    <small>Derechos Reservados © 2013 Iván Rodríguez</small><br />
    <span>Me encanta la película <cite>Blade Runner</cite></span>
    <footer>
      <address>
        Mas info: <a href="http://www.htmlcinco.com/">HTMLCinco</a>
      </address>
    </footer>
  </article>
</section>
</body>
</html>
```

12.4.5 El elemento <time>

En cada <article>, incluimos la fecha indicando cuándo el mensaje fue publicado. Para esto usamos un simple elemento <p> dentro de la cabecera (<header>) del mensaje, pero existe un elemento HTML5 específico para este propósito. El elemento <time> nos permite declarar un texto comprensible para humanos y navegadores que representa fecha y hora.

Seguimos con [Ej12.04-Elementos.html](#):

```
...
<section>
  <article>
    <header>
      <h1>Ejemplo de Entrada</h1>
      <time datetime="2013-01-09" pubdate="pubdate">Publicado 09-01-2013</time>
    </header>
    <span>Mi <mark>coche</mark> es negro</span><br />
    <small>Derechos Reservados © 2013 Iván Rodríguez</small><br />
    <span>Me encanta la película <cite>Blade Runner</cite></span>
    <footer>
      <address>
        Mas info: <a href="http://www.htmlcinco.com/">HTMLCinco</a>
      </address>
    </footer>
  </article>
</section>
</body>
</html>
```

El atributo datetime tiene el valor que representa la fecha comprensible para el navegador (timestamp). El formato de este valor deberá seguir un patrón similar al del siguiente ejemplo: 2013-01-09T20:03:45. También incluimos el atributo pubdate, el cual solo es agregado para indicar que el valor del atributo datetime representa la fecha de publicación.

12.5 Ejemplo de Marcado Semántico

Veamos un ejemplo completo añadiendo CSS ([Ej12.05-Semantico.html](#)):

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <style>
      body { margin: 0 auto; width: 600px; font: 75% Lucida Grande, Trebuchet MS }
      /* La sección se muestra como un bloque, a la izquierda y con margen interno 10px */
      section { display: block; }
      section#articles { width: 440px; float: left; padding: 10px; background-color: #ffc; }

      /* La cabecera del artículo se subraya; margen inferior: 10px */
      article > header { text-decoration: underline; margin-bottom: 10px; }
      /* Columna Derecha, alineado a la izquierda, ancho 100px y margen interno 10px */
      aside { float: left; width: 100px; padding: 10px; }

      /*overflow: hidden; clear: both -> No se permiten elementos flotantes laterales*/
      footer { overflow: hidden; clear: both; text-align: center; padding: 20px; }

      /* En la sección Navegación, fondo gris oscuro, centrado y sin los puntitos */
      nav li { float: left; width: 100px; text-align: center; padding: 10px; }
      nav ul { list-style: none; overflow: hidden; padding: 0; margin: 0; background-color: #444; }

      /* Coloreando enlaces para nav en blanco con subrayado en rojo*/
      nav a { text-decoration: none; border-bottom: 1px solid red; }
      nav li a { color: #fff; }
      /* Coloreando enlaces para aside en verde*/
      aside ul a { text-decoration: none; color: #65ac1c; border-bottom: 1px solid red; }
    </style>
  <title>Ejemplo de HTML5</title>
  <meta charset="UTF-8">
```

```

</head>
<body>
  <header>
    <hgroup>
      <h1>Ejemplo de página semántica</h1>
      <h2>Usando HTML5...</h2>
    </hgroup>
  </header>
  <nav>
    <ul>
      <li><a href="#">Portada</a></li>
      <li><a href="#">Tutoriales</a></li>
      <li><a href="#">Sobre mi</a></li>
      <li><a href="#">Contacto</a></li>
    </ul>
  </nav>
  <section id="articles">
    <article>
      <header>
        <h2><a href="#">Article Title</a></h2>
      </header>
      <section>
        Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor
        incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
        exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor
        in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur
        sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est
        laborum.
      </section>
    </article>
  /78    <article>
    <header>
      <h2><a href="#">Article Title</a></h2>
    </header>
    <section>
      Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor
      incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud
      exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor
      in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur
      sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est
      laborum.
    </section>
  </article>
  </section>
  <aside>
    <h2>Enlaces Principales</h2>
    <ul>
      <li><a ref="external" href="#">Enlace 1</a></li>
      <li><a ref="external" href="#">Enlace 2</a></li>
      <li><a ref="external" href="#">Enlace 3</a></li>
      <li><a ref="external" href="#">Enlace 4</a></li>
      <li><a ref="external" href="#">Enlace 5</a></li>
    </ul>
  946 </aside>

  <footer>
    <small>Derechos Reservados © 2013 Iván Rodríguez</small><br />
    <address>
      Mas info: <a href="http://www.ivanrguez.es/">ivanrguez.es</a>
    </address>
  </footer>
</body>
</html>
</html>

```


13 FORMULARIOS CON HTML5

13.1 Formularios Web

La Web 2.0 está completamente enfocada en el usuario. Y cuando el usuario es el centro de atención, todo está relacionado con interfaces, en cómo hacerlas más intuitivas, más naturales, más prácticas, y por supuesto más atractivas. Los formularios web son la interface más importante de todas, permiten a los usuarios insertar datos, tomar decisiones, comunicar información y cambiar el comportamiento de una aplicación. Durante los últimos años, códigos personalizados y librerías fueron creados para procesar formularios en el ordenador del usuario. HTML5 vuelve a estas funciones estándar agregando nuevos atributos, elementos y una API completa. Ahora la capacidad de procesamiento de información insertada en formularios en tiempo real ha sido incorporada en los navegadores y completamente estandarizada.

13.1.1 El elemento <form>

Los formularios en HTML no han cambiado mucho. La estructura sigue siendo la misma, pero HTML5 ha agregado nuevos elementos, tipos de campo y atributos para expandirlos tanto como sea necesario y proveer así las funciones actualmente implementadas en aplicaciones web.

Existen nuevos atributos para el elemento <form>:

- **autocomplete:** Puede tomar dos valores: on y off. El valor por defecto es on. Cuando es configurado como off los elementos <input> pertenecientes a ese formulario tendrán la función de autocompletar desactivada, sin mostrar entradas previas como posibles valores. Puede ser implementado en el elemento <form> o en cualquier elemento <input> independientemente.
- **novalidate.** Una de las características de formularios en HTML5 es la capacidad propia de validación. Los formularios son automáticamente validados. Para evitar este comportamiento, podemos usar el atributo novalidate. Para lograr lo mismo para elementos <input> específicos, existe otro atributo llamado formnovalidate. Ambos atributos son booleanos, ningún valor tiene que ser especificado (su presencia es suficiente para activar su función).

MUY IMPORTANTE: Los códigos a partir de ahora deben ser probados preferentemente con Google Chrome (y en su defecto con Firefox, aunque este puede dar problemas).

Vamos a definir una plantilla para manejar formularios en PHP:

maneja_formularioHTML5.php:

```
<?php
    // Si existe la variable nombre, lo recogemos
    if (isset ($_POST ['nombre']))
    {
        $nombre = $_POST ['nombre'];
        echo "<h1> Nombre: ".$nombre . "</h1>";
    }

    // Para múltiples checkbox, usaremos un array
    if (isset ($_POST ['puesto']))
    {
        foreach ($_POST['puesto'] as $puestos)
        {
            echo $puestos."<br>";
        }
    }

    if (isset ($_POST ['sexo']))
    {
        $Sexo = $_POST ['sexo'];
        echo "<h1> Sexo: ".$Sexo . "</h1>";
    }
?>
```

Veamos una plantilla para Formulario HTML5 ([Ej13.01a-PlantillaFormulario.html](#)):

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Formularios</title>
</head>
<body>
    <section>
        <form action="maneja_formularioHTML5.php" name="miformulario" id="miformulario"
            method="post" enctype="multipart/form-data" autocomplete="on">
            Nombre: <input type="text" name="nombre" id="nombre" novalidate="novalidate" />
            <br />
            <input type="submit" value="Enviar">
        </form>
    </section>
</body>
</html>
```

13.1.2 El elemento <input>

El elemento más importante en un formulario es <input>. Este elemento puede cambiar sus características gracias al atributo type (tipo). Este atributo determina qué clase de entrada es esperada desde el usuario. Los tipos disponibles hasta el momento eran el multipropósitos text (para textos en general) y solo unos pocos más específicos como password o submit. HTML5 ha expandido las opciones incrementando de este modo las posibilidades para este elemento.

En HTML5 estos nuevos tipos no solo están especificando qué clase de entrada es esperada sino también diciéndole al navegador qué debe hacer con la información recibida. El navegador procesará los datos ingresados de acuerdo al valor del atributo type y validará la entrada o no. El atributo type trabaja junto con otros atributos adicionales para ayudar al navegador a limitar y controlar en tiempo real lo ingresado por el usuario.

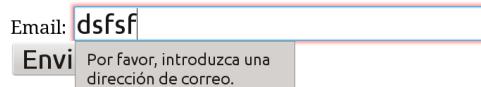
13.1.3 Tipo email

Casi todo formulario en la web ofrece un campo para ingresar una dirección de email, pero hasta ahora el único tipo de campo disponible para esta clase de datos era text. El tipo text representa un texto general, no un dato específico, por lo que teníamos que controlar la entrada con código Javascript para estar seguros de que el texto ingresado correspondía a un email válido. Ahora el navegador se hace cargo de esto con el nuevo tipo email.

El texto insertado en el campo email será controlado por el navegador y validado como un email. Si la validación falla, el formulario no será enviado. Cómo cada navegador responderá a una entrada inválida no está determinado en la especificación de HTML5. Por ejemplo, algunos navegadores mostrarán un borde rojo alrededor del elemento <input> que produjo el error y otros lo mostrarán en azul. Existen formas de personalizar esta respuesta, pero las veremos más adelante.

Veamos un ejemplo [Ej13.01b-InputEmail.html](#):

```
<!DOCTYPE html>
<html lang="es">
<head><title>Formularios</title></head>
<body>
    <section>
        <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">
            Email: <input type="email" name="nombre" id="nombre" />
            <br />
            <input type="submit" value="Enviar" />
        </form>
    </section>
</body>
</html>
```



13.1.4 Tipo search.

El tipo search (búsqueda) no controla la entrada, es solo una indicación para los navegadores. Al detectar este tipo de campo algunos navegadores cambiarán el diseño del elemento para ofrecer al usuario un indicio de su propósito.

```
<input type="search" name="busqueda" id="busqueda" />
```

13.1.5 Tipo url.

Este tipo de campo trabaja exactamente igual que el tipo email pero es específico para direcciones web. Está destinado a recibir solo URLs absolutas y retornará un error si el valor es inválido.

NOTA: Como URL absoluta se entienda algo similar a <http://www.mipagina.com>

```
<input type="url" name="miurl" id="miurl">  
Un ejemplo (Ej13.01c-SearchURL.html):  
<!DOCTYPE html>  
<html lang="es">  
<head><title>Formularios</title></head>  
<body>  
    <section>  
        <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">  
            Buscador: <input type="search" name="busqueda" id="busqueda"><br />  
            URL Solicitada: <input type="url" name="miurl" id="miurl"><br />  
            <input type="image" src="imagenes/Lupa.png" />  
        </form>  
    </section>  
</body>  
</html>
```

Buscador: URL Solicitada: Enviar
Por favor, introduzca una URL.

13.1.6 Tipo tel.

Este tipo de campo es para números telefónicos. A diferencia de los tipos email y url, el tipo tel no requiere ninguna sintaxis en particular. Es solo una indicación para el navegador en caso de que necesite hacer ajustes de acuerdo al dispositivo en el que la aplicación es ejecutada.

```
<input type="tel" name="telefono" id="telefono">
```

13.1.7 Tipo number

Como su nombre lo indica, el tipo number es sólo válido cuando recibe una entrada numérica. Existen algunos atributos nuevos que pueden ser útiles para este campo:

- **min** El valor de este atributo determina el mínimo valor aceptado para el campo.
- **max** El valor de este atributo determina el máximo valor aceptado para el campo.
- **step** El valor de este atributo determina el tamaño en el que el valor será incrementado o disminuido en cada paso.

No es necesario especificar ambos atributos (min y max), y el valor por defecto para step es 1.

```
<input type="number" name="numero" id="numero" min="0" max="10" step="5">
```

Un ejemplo Ej13.01d-TelNumber.html:

```
<!DOCTYPE html>  
<html lang="es">  
<head><title>Formularios</title>  
    <meta charset="UTF-8" /></head>  
<body>  
    <section>  
        <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">  
            Teléfono: <input type="tel" name="telefono" id="telefono"><br />  
            Edad: <input type="number" name="numero" min="0" max="10" step="5"><br />  
            <input type="submit" value="Enviar">  
        </form>  
    </section>  
</body></html>
```

Teléfono: DNI:

Ejercicio Especial:

El formulario:

http://www.mclibre.org/consultar/amaya/ejercicios/hoja_inscripcion/hoja_inscripcion_cfs_dogv030428.html

En vez de Fecha Nacimiento, poner edad con type number

En vez de nacido en poner Correo electrónico, type email.

Añadir type search donde queráis y type tel.

Incluir el PHP.

13.1.8 Tipo Range.

Este tipo de campo hace que el navegador construya una nueva clase de control que no existía previamente. Este nuevo control le permite al usuario seleccionar un valor a partir de una serie de valores o rango. Normalmente es mostrado en pantalla como una puntero deslizable o un campo con flechas para seleccionar un valor entre los predeterminados, pero no existe un diseño estándar hasta el momento.

El tipo range usa los atributos min y max estudiados previamente para configurar los límites del rango. También puede utilizar el atributo step para establecer el tamaño en el cual el valor del campo será incrementado o disminuido en cada paso.

Podemos declarar el valor inicial utilizando el viejo atributo value y usar Javascript para mostrar el número seleccionado en pantalla como referencia. Experimentaremos con esto y el nuevo elemento <output> más adelante.

Un ejemplo **Ej13.01e-Range.html**:

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

    <script>
        /* Aquí va la parte de JavaScript*/
        /* Se copia el nombre de la función con los argumentos
        pero en HTML van con '' y aquí en el Script SE QUITAN*/
        function imprimeValor (rangoPasado,valorRangoPasado) {
            var barraRango = document.getElementById(rangoPasado);
            var campoTexto = document.getElementById(valorRangoPasado);
            campoTexto.value = barraRango.value;
        }

        window.onload = function () {
            imprimeValor ('rango1','valordelRango');
        }
    </script>
</head>
<body>
    <section>
        <form action="maneja_formularioHTML5.php"
            method="post" autocomplete="on">
            <p>Mínimo = 100, Máximo = 500, Salto = 10</p>
            <br />
            <input type="range" min="100" max="500" step="10" id="rango1"
                onchange ="imprimeValor ('rango1','valordelRango')"
                name="mirango" class="miclaseRango" />
            <input type="text" maxlength="3" size="3" id="valordelRango" />
            <br />
            <input type="submit" value="Enviar" />
        </form>
    </section>
</body>
</html>
```

13.1.9 Tipo date

Este es otro tipo de campo que genera una nueva clase de control. En este caso fue incluido para ofrecer una mejor forma de ingresar una fecha. Algunos navegadores muestran en pantalla un calendario que aparece cada vez que el usuario hace clic sobre el campo. El calendario le permite al usuario seleccionar un día que será ingresado en el campo junto con el resto de la fecha. Un ejemplo de uso es cuando necesitamos proporcionar un método para seleccionar una fecha para un vuelo o la entrada a un espectáculo. Gracias al tipo date ahora es el navegador el que se encarga de construir un almanaque o las herramientas necesarias para facilitar el ingreso de este tipo de datos.

Elija Fecha

A screenshot of a date picker interface. At the top, there's a dropdown menu labeled "Día/Mes/Año" with a downward arrow. Below it, a small calendar shows the month of January 2013. The days of the week are labeled from "lun" to "dom". The date "10" is highlighted with a blue background, indicating it is the selected date. There are navigation buttons for "Hoy" (Today) and "Eliminar" (Delete). At the bottom right of the calendar, there's a button labeled "Enviar" (Send).

Un ejemplo [Ej13.01f-Date.html](#):

```
<!DOCTYPE html>
<html lang="es">
<head>
    <meta charset="UTF-8" />
    <title>Formularios</title>
</head>
<body>
    <section>
        <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">
            <p> Elija Fecha </p>
            <input type="date" name="fecha" id="fecha">
            <br /><br /><br /><br /><br /><br /><br /><br /><br />
            <input type="submit" value="Enviar">
        </form>
    </section>
</body>
</html>
```

La interfaz no fue declarada en la especificación. Cada navegador provee su propia interface y a veces adaptan el diseño al dispositivo en el cual la aplicación está siendo ejecutada. Normalmente el valor generado y esperado tiene la sintaxis año-mes-día.

13.1.10 Tipo week.

Este tipo de campo ofrece una interface similar a date, pero solo para seleccionar una semana completa. Normalmente el valor esperado tiene la sintaxis 2011-W50 donde 2011 es al año y 50 es el número de la semana.

```
<input type="week" name="semana" id="semana">
```

13.1.11 Tipo month.

Similar al tipo de campo previo, éste es específico para seleccionar meses. Normalmente el valor esperado tiene la sintaxis año-mes.

```
<input type="month" name="mes" id="mes">
```

13.1.12 Tipo time.

El tipo de campo time es similar a date, pero solo para la hora. Toma el formato de horas y minutos, pero su comportamiento depende de cada navegador en este momento. Normalmente el valor esperado tiene la sintaxis hora:minutos:segundos, pero también puede ser solo hora:minutos.

```
<input type="time" name="hora" id="hora">
```

13.1.13 Tipo datetime.

El tipo de campo datetime es para ingresar fecha y hora completa, incluyendo la zona horaria.

<input type="datetime" name="fechahora" id="fechahora">

Más información: <http://www.w3.org/TR/html-markup/input.datetime.html#input.datetime>

13.1.14 Tipo datetime-local.

El tipo de campo datetime-local es como el tipo datetime sin la zona horaria.

<input type="datetime-local" name="tiempolocal" id="tiempolocal">

Veamos un ejemplo con los 5 apartados anteriores:

Ej13.01g-Fechas.html:

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

<body>
<section>
    <form action="maneja_formularioHTML5.php"
        method="post" autocomplete="on">
        <p> Elija Fecha </p>
        Semana: <input type="week" name="semana" id="semana"><br />
        Mes: <input type="month" name="mes" id="mes"><br />
        Hora:<input type="time" name="hora" id="hora"><br />
        Dia y Hora: <input type="datetime" name="fechahora" id="fechahora"><br />
        Dia y Hora (Local): <input type="datetime-local"
            name="tiempolocal" id="tiempolocal"><br />
        <input type="submit" value="Enviar">
    </form>
</section>
</body>
</html>
```

Elija Fecha

Semana:

Mes:

Hora:

Dia y Hora:

Dia y Hora (Local):

13.1.15 Tipo Color

Además de los tipos de campo para fecha y hora existe otro tipo que provee una interfaz predefinida similar para seleccionar colores. Normalmente el valor esperado para este campo es un número hexadecimal, como #00FF00.

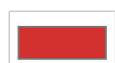
<input type="color" name="micolor" id="micolor">

Ninguna interfaz fue especificada como estándar en HTML5 para el tipo de campo color.

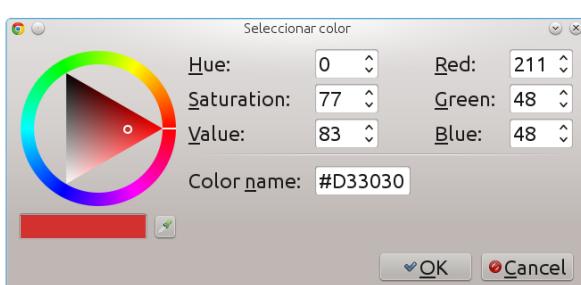
En la imagen inferior, el selector que aparece al usar el navegador Google Chrome.

NOTA: Para que aparezca el valor en un campo de texto hay que usar JavaScript, como aparece en el código de la página siguiente (muy similar al ya visto con el input range).

Elija Color:



#d33030



Un ejemplo Ej13.01h-Color.html:

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Formularios</title>
    <meta charset="UTF-8" />
    <script>
        // Igualamos el valor del input color con el Input Text (con el color elegido)
        function imprimeValor(selector, cajatexto) {
            var x = document.getElementById(cajatexto);
            var y = document.getElementById(selector);
            x.value = y.value;
        }

        // Al cargar la página se ve el valor del input color y del Input Text
        window.onload = function() { imprimeValor('micolor', 'valorColor'); }
    </script>
</head>
<body>
    <section>
        <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">
            <p> Elija Color: </p>
            <input type="color" name="micolor" id="micolor"
                onchange="imprimeValor('micolor', 'valorColor')"><br />
            <input id="valorColor" type="text" size="10"/>
            <br />
            <input type="submit" value="Enviar">
        </form>
    </section>
</body>
</html>
```

13.2 Nuevos atributos.

Algunos tipos de campo requieren de la ayuda de atributos, como los anteriormente estudiados min, max y step. Otros tipos de campo requieren la asistencia de atributos para mejorar su rendimiento o determinar su importancia en el proceso de validación. Ya vimos algunos de ellos, como novalidate para evitar que el formulario completo sea validado o formnovalidate para hacer lo mismo con elementos individuales. El atributo autocomplete, también estudiado anteriormente, provee medidas de seguridad adicionales para el formulario completo o elementos individuales. Aunque útiles, estos atributos no son los únicos incorporados por HTML5. Es momento de estudiar el resto.

13.2.1 Atributo placeholder

Especialmente en tipos de campo search, pero también en entradas de texto normales, el atributo placeholder representa una sugerencia corta, una palabra o frase provista para ayudar al usuario a ingresar la información correcta. El valor de este atributo es presentado en pantalla por los navegadores dentro del campo, como una marca de agua que desaparece cuando el elemento es enfocado.

```
<input type="search" name="busqueda" placeholder="escriba su búsqueda">
```

13.2.2 Atributo required

Este atributo booleano no dejará que el formulario sea enviado si el campo se encuentra vacío. Por ejemplo, cuando usamos el tipo email para recibir una dirección de email, el navegador comprueba si la entrada es un email válido o no, pero validará la entrada si el campo está vacío. Cuando el atributo required es incluido, la entrada será válida sólo si se cumplen las dos condiciones: que el campo no esté vacío y que el valor ingresado esté de acuerdo con los requisitos del tipo de campo.

```
<input type="email" name="miemail" id="miemail" required="required">
```

13.2.3 Atributo multiple

El atributo "multiple" es otro atributo booleano que puede ser usado en algunos tipos de campo (por ejemplo, email o file) para permitir el ingreso de entradas múltiples en el mismo campo. Los valores insertados deben estar separados por coma para ser válidos.

```
<input type="email" name="miemail" id="miemail" multiple="multiple">
```

El código anterior permite la inserción de múltiples valores separados por coma, y cada uno de ellos será validado por el navegador como una dirección de email.

Ej13.02a-PlaceRequiredMultiple.html:

```
<!DOCTYPE html>
<html lang="es">
<head> <title>Formularios</title>
    <meta charset="UTF-8" /></head>
<body>
    <section>
        <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">
            <p> Formulario: </p>
            <input type="search" name="busqueda" placeholder="escriba su búsqueda"><br />
            <input type="email" name="miemail" id="miemail"
                multiple="multiple" required="required"><br />
            <input type="submit" value="Enviar">
        </form>
    </section>
</body>
</html>
```

Formulario:

escriba su búsqueda
as

Enviar

Introduce una lista de direcciones de correo electrónico separada por comas

vamos a ver el código del PHP

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

</head>
<?php
    /* if -> SI
    Si la variable búsqueda se ha mandado vacía ("") ...*/
    if ($_POST ['busqueda']=="")
    {
        echo "Búsqueda: NO.<br />";
        echo "Realice una búsqueda en la página anterior. <br />";
    }
    /*Si no (else) es que la variable se ha completado...*/
    else {
        $mibusqueda = $_POST ['busqueda'];
        echo "Búsqueda: ".$mibusqueda."<br />";
    }

    $mis correos = $_POST ['correo'];
    echo "Correos: ".$mis correos;
?>
```

13.2.4 Atributo autofocus

El atributo autofocus enfocará la página web sobre el elemento seleccionado pero considerando la situación actual. No moverá el foco cuando ya haya sido establecido por el usuario sobre otro elemento. Ejemplo: Modificamos **Ej13.02a-PlaceRequiredMultiple.html**:

```
<input type="search" name="busqueda" id="busqueda"
placeholder="escriba su búsqueda" autofocus="autofocus">
```

13.2.5 Atributo form

El atributo form es una adición útil que nos permite declarar elementos para un formulario fuera del ámbito de las etiquetas `<form>`. Hasta ahora, para construir un formulario teníamos que escribir las etiquetas `<form>` de apertura y cierre y luego declarar cada elemento del formulario entre ellas. En HTML5 podemos insertar los elementos en cualquier parte del código y luego hacer referencia al formulario que pertenecen usando su nombre y el atributo form:

Un ejemplo (**Ej13.02c-AtributoForm.html**):

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Formularios</title>
    <meta charset="UTF-8" />
  </head>
  <body>
    <nav>
      BARRA NAVEGACIÓN DEL SITIO --
      <input type="search" name="busqueda"
        placeholder="escriba su búsqueda" form="formulario">
    </nav>
    <section>
      <form name="formulario" id="formulario" method="get">
        <input type="text" name="nombre" id="nombre">
        <input type="submit" value="Enviar">
      </form>
    </section>
  </body>
</html>
```

13.2.6 Atributo pattern

El atributo pattern es para propósitos de validación. Usa expresiones regulares para personalizar reglas de validación. Algunos de los tipos de campo ya estudiados validan cadenas de texto específicas, pero no permiten hacer validaciones personalizadas, como un código postal.

No existe ningún tipo de campo predeterminado para esta clase de entrada. El atributo pattern nos permite crear nuestro propio tipo de campo para controlar esta clase de valores no ordinarios. Puede incluso incluir un atributo title para personalizar mensajes de error.

Algunos patrones: <http://html5pattern.com/>

Carácter	Texto buscado
^	Principio de entrada o línea.
\$	Fin de entrada o línea.
*	El carácter anterior 0 o más veces.
+	El carácter anterior 1 o más veces.
?	El carácter anterior una vez como máximo (el carácter anterior es opcional).
.	Cualquier carácter individual, salvo el de salto de línea.
x y	x o y.
{n}	Exactamente n apariciones del carácter anterior.
{n,m}	Como mínimo n y como máximo m apariciones del carácter anterior.
[abc]	Cualquiera de los caracteres entre corchetes. Especifique un rango de caracteres con un guión (por ejemplo, [a-f] es equivalente a [abcdef]).
[^abc]	Cualquier carácter que no esté entre corchetes. Especifique un rango de caracteres con un guión (por ejemplo, [^a-f] es equivalente a [^abcdef]).
\b	Límite de palabra (como un espacio o un retorno de carro).
\B	Cualquiera que no sea un límite de palabra.
\d	Cualquier carácter de dígito. Equivalente a [0-9].
\D	Cualquier carácter que no sea de dígito. Equivalente a [^0-9].
\f	Salto de página.
\n	Salto de línea.
\r	Retorno de carro.
\s	Cualquier carácter individual de espacio en blanco (espacios, tabulaciones, saltos de página o saltos de línea).
\S	Cualquier carácter individual que no sea un espacio en blanco.
\t	Tabulación.
\w	Cualquier carácter alfanumérico, incluido el de subrayado. Equivalente a [A-Za-z0-9_].
\W	Cualquier carácter que no sea alfanumérico. Equivalente a [^A-Za-z0-9_].

Os dejo algunos ejemplos ya hechos para que los uséis:

Cualquier letra en minúscula	[a-z]
Entero	$^{:} (+ -) ? \d{+} \$$ (como alternativa [0-9])
URL	$^{:} (ht f) tp (s?) : / / [0-9a-zA-Z] ([-.\w] * [0-9a-zA-Z]) * ((0-9) *) * (/?) ([a-zA-Z0-9-\-.?\\, \'\\"/\\"/+&%\$#_] *) ? \$$
Contraseña segura	$^{:} (!^{:} [0-9] * \$) (!^{:} [a-zA-Z] * \$) ^{:} ([a-zA-Z0-9] {8,10}) \$$ (Entre 8 y 10 caracteres, por lo menos un dígito y un alfanumérico, y no puede contener caracteres especiales)
Fecha	$^{:} \d{1,2} / \d{1,2} / \d{2,4} \$$ (Por ejemplo 01/01/2007)
Hora	$^{:} (0[1-9] 1\d{2}[0-3]) : ([0-5]\d) : ([0-5]\d) \$$ (Por ejemplo 10:45:23)
Número tarjeta de crédito	$^{:} ((67\d{2}) (4\d{3}) (5[1-5]\d{2}) (6011)) (-?\s?\d{4}) \$$ {3} (3[4,7]) \d{2}-?\s?\d{6}-?\s?\d{5} \\$
Número teléfono	$^{:} [0-9] {2,3} - ? ? [0-9] {6,7} \$$
Código postal	$^{:} ([1-9] {2} [0-9] [1-9] [1-9] [0-9]) [0-9] {3} \$$
Certificado Identificación Fiscal	$^{:} (X(- \.) ? 0 ? \d{7} (- \.) ? [A-Z] [A-Z] (- \.) ? \d{7} (- \.) ? [0-9A-Z] \d{8} (- \.) ? [A-Z]) \$$
Dirección IPv4	$^{:} ((^ \.) ((25[0-5]) (2[0-4]\d) (1\d\d) ([1-9]?\d))) {4} \$$

Un ejemplo ([Ej13.02b-Pattern.html](#)):

```
<!DOCTYPE html>
<html lang="es">
<head>
    <title>Formularios</title>
    <meta charset="UTF-8" />
</head>
<body>
    <section>
        <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">
            <p> Formulario: </p>
            <input pattern="[0-9]{5}" name="cp" title="Inserte Código Postal" /><br />
            <input type="submit" value="Enviar" />
        </form>
    </section>
</body>
</html>
```

13.3 Nuevos elementos para formularios

Ya hemos visto los nuevos tipos de campos disponibles en HTML5, por lo tanto es momento de estudiar los nuevos elementos HTML incorporados con la intención de mejorar o expandir las posibilidades de los formularios.

13.3.1 El elemento <datalist>

El elemento <datalist> es un elemento específico de formularios usado para construir una lista de ítems que luego, con la ayuda del atributo list, será usada como sugerencia en un campo del formulario.

```
<datalist id="informacion">
<option value="123123123" label="Teléfono 1">
<option value="456456456" label="Teléfono 2">
</datalist>
```

Formulario:

<input type="text"/>	<input type="button" value="Enviar"/>
123123123	Teléfono 1
456456456	Teléfono 2

Este elemento utiliza el elemento <option> en su interior para crear la lista de datos a sugerir. Con la lista ya declarada, lo único que resta es referenciarla desde un elemento <input> usando el atributo list:
<input type="tel" name="telefono" id="telefono" list="informacion">

IMPORTANTE: El elemento <datalist> fue solo implementado en Opera y Firefox Beta en este momento. Un ejemplo ([Ej13.03a-Datalist.html](#)):

```
<!DOCTYPE html>
<html lang="es">
<head> <title>Formularios</title>
      <meta charset="UTF-8" /> </head>
<body>
  <section>
    <form action="maneja_formularioHTML5.php" method="post" autocomplete="on">
      <p> Formulario: </p>
      <datalist id="informacion">
        <option value="123123123" label="Teléfono 1">
        <option value="456456456" label="Teléfono 2">
      </datalist>
      <input type="tel" name="telefono" id="telefono" list="informacion">
      <input type="submit" value="Enviar" />
    </form>
  </section>
</body>
</html>
```

13.3.2 El elemento <progress>

Este elemento no es específico de formularios, pero debido a que representa el progreso en la realización de una tarea, y usualmente estas tareas son comenzadas y procesadas a través de formularios, puede ser incluido dentro del grupo de elementos para formularios.

El elemento <progress> utiliza dos atributos para configurar su estado y límites. El atributo value indica qué parte de la tarea ya ha sido procesada, y max declara el valor a alcanzar para que la tarea se considere finalizada.. Un ejemplo, ([Ej13.03b-ProgressMeter.html](#)):

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" /> </head>
<body>
  <section>
    <progress value="60" max="100">
      <span id="descargando">60</span>%
    </progress>
  </section>
</body>
</html>
```

Porcentaje enviado:



13.3.3 El elemento <meter>

Similar a <progress>, el elemento <meter> es usado para mostrar una escala, pero no de progreso. Este elemento tiene la intención de representar una medida, como el tráfico del sitio web, por ejemplo.

El elemento <meter> cuenta con varios atributos asociados: min y max configuran los límites de la escala, value determina el valor medido, y low, high y optimum son usados para segmentar la escala en secciones diferenciadas y marcar la posición que es óptima

Ej13.03b-ProgressMeter.html (modificamos el ejemplo anterior).

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" /> </head>
<body>
<section>
  Porcentaje enviado: <br />
  <meter value="2" min="0" max="10">2 sobre 10</meter><br>
  <meter value="0.6">60%</meter>
</section>
</body>
</html>
```

13.3.4 El elemento <output>

Este elemento representa el resultado de un cálculo. Normalmente ayudará a mostrar los resultados del procesamiento de valores provistos por un formulario. El atributo for asocia el elemento <output> con el elemento fuente que participa del cálculo, pero este elemento deberá ser referenciado y modificado desde código Javascript. Su sintaxis es <output>valor</output>.

Un Ejemplo: **Ej13.03c-Output.html**

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" />
<script>
  // Igualamos el valor del input range con el Input Text (con el número elegido)
  function imprimeValor(rango,cuadro) {
    var x = document.getElementById(cuadro);
    var y = document.getElementById(rango);
    x.value = y.value;
  }
  // Al cargar la página se ve el valor del input range y del Input Text aquí puestos
  window.onload = function() {
    imprimeValor('rango1', 'valordelRango'); }
</script>

</head>
<body>
<section>
  <!-- Dentro del form uso el evento OnInput para realizar el cálculo -->
  <form oninput="resultado.value= rango1.valueAsNumber* cuadro1.valueAsNumber">
    <input id="rango1" name="rango1" type="range" min="0" max="100" step="1"
      onchange="imprimeValor('rango1','valordelRango')"/>100 <br />
    <input id="valordelRango" type="text" size="2"/>
    *
    <input type="number" name="cuadro1" value="50">
    =
    <output name="resultado" for="rango1 cuadro1"></output>
  </form>
</section>
</body>
</html>
```

NOTA: Hay un ejemplo mas sencillo en este enlace:

<http://surpatterns.com/sitio/html5-en-espanol-surpatterns/tutorial-html5-en-espanol-el-elemento-output/>

14 VÍDEO Y AUDIO EN HTML5

14.1 Vídeo en HTML5.

Una de las características más mencionadas de HTML5 fue la capacidad de procesar vídeo. El entusiasmo nada tenía que ver con las nuevas herramientas provistas por HTML5 para este propósito, sino más bien con el hecho de que desde los vídeos se volvieron una pieza esencial de Internet, todos esperaban soporte nativo por parte de los navegadores. Era como que todos conocían la importancia de los vídeos excepto aquellos encargados de desarrollar las tecnologías para la web.

Pero ahora que ya disponemos de soporte nativo para vídeos e incluso un estándar que nos permitirá crear aplicaciones de procesamiento de vídeo compatibles con múltiples navegadores, podemos comprender que la situación era mucho más complicada de lo que nos habíamos imaginado. Desde codificadores hasta consumo de recursos, las razones para no implementar vídeo de forma nativa en los navegadores eran mucho más complejas que los códigos necesarios para hacerlo.

A pesar de estas complicaciones, HTML5 finalmente introdujo un elemento para insertar y reproducir vídeo en un documento HTML. El elemento `<video>` usa etiquetas de apertura y cierre y solo unos pocos parámetros para lograr su función. La sintaxis es extremadamente sencilla y solo el atributo `src` es obligatorio,

Veamos un ejemplo: Nos vamos a Youtube y buscamos Daniel Lopez.

Pulsamos en el vídeo titulado *El Cielo de Canarias - Daniel López*

Nos lo descargamos en MP4 y lo metemos en htdocs.

Veamos un código sencillo: [Ej14.01a-VideoSimple.html](#) .

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" /> </head>
<body>
  <section>
    Ejemplo de vídeo: <br />
    <video src="videos/El_Cielo_de_Canarias-Daniel_López[HD].mp4" controls="controls">
      </video>
    </section>
</body>
</html>
```

NOTA: El formato MP4 no funciona en Ubuntu en Firefox ni en Chromium (usar Chrome).

El código anterior debería ser suficiente. Pero debemos proveer al menos dos archivos diferentes con formatos de video diferentes: OGG y MP4. Esto es debido a que a pesar de que el elemento `<video>` y sus atributos son estándar, no existe un formato estándar de video. Primero, algunos navegadores soportan un codificador de video que otros no, y segundo el codificador utilizado en el formato MP4 (el único soportado por importantes navegadores como Safari e Internet Explorer) se encuentra bajo licencia comercial.

Los formatos OGG y MP4 son contenedores de video y audio. OGG contiene codificadores de video Theora y de audio Vorbis, y los disponibles para el contenedor MP4 son H.264 para video y AAC para audio. En este momento OGG es reconocido por Firefox, Google Chrome y Opera, mientras que MP4 trabaja en Safari, Internet Explorer y también Google Chrome.

14.1.1 El elemento <video>

Este elemento ofrece varios atributos para establecer su comportamiento y configuración. Los atributos width y height, al igual que en otros elementos HTML ya conocidos, declaran las dimensiones para el elemento o ventana del reproductor. El tamaño del video será automáticamente ajustado para entrar dentro de estos valores, pero no fueron considerados para redimensionar el video sino limitar el área ocupada por el mismo para mantener consistencia en el diseño. El atributo src especifica la fuente del video. Este atributo puede ser reemplazado por el elemento <source> y su propio atributo src para declarar varias fuentes con diferentes formatos.

Vamos a modificar el ejemplo anterior: [Ej14.01a-VideoSimple.html](#).

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" /> </head>
<body>
    <section>
        Ejemplo de vídeo: <br />
        <video id="mivideo" width="720" height="400" controls="controls"
            src="videos/El_Cielo_de_Canarias-Daniel_López[HD].mp4"
            src="videos/El_Cielo_de_Canarias-Daniel_López[HD].ogg">
        </video>
    </section>
</body>
</html>
```

14.1.2 Atributos para <video>

Ya hemos puesto uno: controls. El atributo controls es uno de varios atributos disponibles para este elemento. Éste, en particular, muestra controles de video provistos por el navegador por defecto. Cuando el atributo está presente cada navegador activará su propia interface, permitiendo al usuario comenzar a reproducir el video, pausarlo o saltar hacia un cuadro específico, entre otras funciones.

Junto con controls, también podemos usar los siguientes:

- **autoplay**: Cuando este atributo está presente, el navegador comenzará a reproducir el video automáticamente tan pronto como pueda. Valor igual al nombre del atributo.
- **loop**: Si este atributo es especificado, el navegador comenzará a reproducir el video nuevamente cuando llega al final. Valor igual al nombre del atributo.
- **poster**: Este atributo es utilizado para proveer una imagen que será mostrada mientras esperamos que el video comience a ser reproducido. Valor igual al nombre del atributo.
- **preload**: Este atributo puede recibir tres valores distintos: none, metadata o auto. El primero indica que el video no debería ser cacheado, por lo general con el propósito de minimizar tráfico innecesario. El segundo valor, metadata, recomendará al navegador que trate de capturar información acerca de la fuente (por ejemplo, dimensiones, duración, primer cuadro, etc...). El tercer valor, auto, es el valor configurado por defecto que le sugerirá al navegador descargar el archivo tan pronto como sea posible.

Vamos a modificar el ejemplo anterior: [Ej14.01a-VideoSimple.html](#).

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" /> </head>
<body>
    <section>
        Ejemplo de vídeo: <br />
        <video id="mivideo" width="720" height="400" preload="preload" controls="controls"
            loop="loop" poster="http://www.cielosdelteide.com/fotos/2011/Tindaya-DLopez.jpg"
            src="videos/El_Cielo_de_Canarias-Daniel_López[HD].mp4"
            src="videos/El_Cielo_de_Canarias-Daniel_López[HD].ogg">
        </video>
    </section>
</body>
</html>
```

14.2 Formatos de video

Por el momento no existe un estándar para formatos de video y audio en la web. Existen varios contenedores y diferentes codificadores disponibles, pero ninguno fue totalmente adoptado y no hay consenso alguno de parte de los fabricantes de navegadores para lograr un estándar en el futuro cercano.

Los contenedores más comunes son OGG, MP4, FLV y el nuevo propuesto por Google, WEBM. Normalmente estos contenedores contienen vídeo codificado con los codificadores Theora, H.264, VP6 o VP8, respectivamente. Esta es la lista de los más usados:

- OGG codificador de vídeo Theora y audio Vorbis.
- MP4 codificador de vídeo H.264 y audio AAC.
- FLV codificador de vídeo VP6 y audio MP3. También soporta H.264 y AAC.
- WEBM codificador de vídeo VP8 y audio Vorbis.

Los codificadores utilizados para OGG y WEBM son gratuitos, pero los utilizados para MP4 y FLV están patentados, lo que significa que si queremos usar MP4 o FLV para nuestras aplicaciones deberemos pagar. Algunas restricciones son anuladas para aplicaciones gratuitas.

El tema es que en este momento Safari e Internet Explorer no soportan la tecnología gratuita. Ambos solo trabajan con MP4 y solo Internet Explorer anunció la inclusión del codificador VP8 en el futuro. Esta es la lista de los navegadores más populares:

- Firefox codificador de vídeo Theora y audio Vorbis.
- Google Chrome codificador de vídeo Theora y audio Vorbis. También soporta codificador de vídeo H.264 y audio AAC.
- Opera codificador de vídeo Theora y audio Vorbis.
- Safari codificador de vídeo H.264 y audio AAC.
- Internet Explorer codificador de vídeo H.264 y audio AAC.

Un mayor soporte para el formato WEBM en el futuro podría mejorar la situación, pero probablemente no habrá un formato estándar por al menos los próximos dos o tres años y tendremos que considerar diferentes alternativas de acuerdo a la naturaleza de nuestra aplicación y nuestro negocio.

14.3 Reproduciendo audio con HTML5

Audio no es un medio tan popular como video en Internet. Podemos filmar un video con una cámara personal que generará millones de vistas en sitios web como www.youtube.com, pero crear un archivo de audio que obtenga el mismo resultado es prácticamente imposible. Sin Embargo, el audio se encuentra aún disponible, ganando su propio mercado en shows de radio y podcasts en toda la red.

HTML5 provee un nuevo elemento para reproducir audio en un documento HTML. El elemento, por supuesto, es `<audio>` y comparte casi las mismas características del elemento `<video>`.

Para probarlo vamos a usar el banco de imágenes del Ministerio de Educación y Deporte.

<http://recursostic.educacion.es/bancoimagenes/web/>

En el buscador ponemos el siguiente texto: **Superspace**. Bajamos las versiones MP3 y OGG

Un ejemplo: **Ej14.03a-Audio.html** .

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" /> </head>
<body>
    <section id="reproductor">
        <audio src="audios/k01444.mp3" controls="controls">
        </audio>
    </section>
</body>
</html>
```

14.3.1 El elemento `<audio>`

El elemento `<audio>` trabaja del mismo modo y comparte varios atributos con el elemento `<video>`:

- `src` Este atributo especifica la URL del archivo a ser reproducido. Al igual que en el elemento `<video>` normalmente será reemplazado por el elemento `<source>` para ofrecer diferentes formatos de audio entre los que el navegador pueda elegir.
- `controls` Este atributo activa la interface que cada navegador provee por defecto para controlar la reproducción del audio.
- `autoplay` Cuando este atributo está presente, el audio comenzará a reproducirse automáticamente tan pronto como sea posible.
- `loop` Si este atributo es especificado, el navegador reproducirá el audio una y otra vez de forma automática.
- `preload` Este atributo puede tomar tres valores diferentes: `none`, `metadata` o `auto`.
 - `none`: indica que el audio no debería ser cacheado, normalmente con el propósito de minimizar tráfico innecesario.
 - `metadata`, pide al navegador obtener información sobre el medio (ej: duración).
 - `auto`, (por defecto) solicita al navegador descargar el archivo lo antes posible

De nuevo tenemos problemas con los codificadores, así que tendremos que poner el archivo en ogg y mp3. Modificamos el ejemplo anterior: **Ej14.03a-Audio.html** .

```
<!DOCTYPE html>
<html lang="es">
<head> <meta charset="UTF-8" /> </head>
<body>
    <section id="reproductor">
        <audio id="medio" controls="controls">
            <source src="audios/k01444.mp3" />
            <source src="audios/k01444.ogg" />
        </audio>
    </section>
</body>
</html>
```

15 BIBLIOGRAFÍA Y ENLACES

15.1 Bibliografía recomendada

Los manuales recomendados son los siguientes:

1. [EL GRAN LIBRO DE HTML5, CSS3 Y JAVASCRIPT](#)

Autor: Gauchat, Juan Diego

Editorial: MARCOMBO, S.A.

ISBN: 9788426717702

En Google Books:

<http://books.google.es/books?>

[id=szDMIRzwzuUC&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q=f=false](http://books.google.es/books?id=szDMIRzwzuUC&printsec=frontcover&hl=es&source=gbs_ge_summary_r&cad=0#v=onepage&q=f=false)

2. [HTML5 Y CSS3 DOMINE LOS ESTÁNDARES DE LAS APLICACIONES WEB](#)

Autor: Luc VAN LANCKER

Editorial: Eni Ediciones

ISBN: ISBN : 978-2-7460-6816-2

En Google Books:

<http://books.google.es/books?>

[id=25p0iG6tW7MC&printsec=frontcover&dq=HTML5+y+CSS3&hl=es](http://books.google.es/books?id=25p0iG6tW7MC&printsec=frontcover&dq=HTML5+y+CSS3&hl=es)

15.2 Enlaces de Interés

Estos son algunos enlaces de interés para ampliar información:

- <http://www.w3.org/TR/html-markup/> → La especificación oficial HTML5
- <http://www.htmlcinco.com> → Excelente blog sobre el estándar HTML5
- <http://www.w3schools.com> → Sitio sobre Diseño Web (inglés)
- <http://www.html5rocks.com/es> → Web sobre desarrollo en HTML5
- <http://www.librosweb.es> → Colección de manuales HTML, CSS, etc
- <http://webintenta.com> → Recursos Web