UF1302:

CREACIÓN DE PÁGINAS WEB CON LENGUAJES DE MARCAS.

Manual teórico.

INDICE

1. Creación de páginas web con lenguajes de marcas.

1.1. Características de los lenguajes de marcas.

Los lenguajes de marcas o de marcado son sistemas de codificación de documentos caracterizados por el uso de etiquetas (marcas) que aportan información sobre la estructura y estilo de un texto.

Profundicemos un poco en esta definición. Lo primero es despejar una confusión que se produce muy comúnmente: los lenguajes de marcas no son lenguajes de programación; para que algo sea considerado así debe incluir el uso de variables, estructuras de control, etc. Y nada de esto lo hacen los lenguajes de marcas. Sin embargo, si son lenguajes en cuanto a que tienen una sintaxis, unas palabras clave y unas reglas fijas, más o menos estrictas, que hacen que a un documento de texto se le pueda identificar como HTML o XML por ejemplo.

Uno de los lenguajes de marcado más relevantes actualmente es XML (eXtensible Markup Language), lenguaje desarrollado por el World Wide Web Consortium (W3C) y que permite la definición de otros lenguajes de marcado. Para ello se basa en una serie de premisas y reglas cuya aplicación produce lo que se conoce como “documento bien formado”, es decir ajustado a las normas y por lo tanto correcto. HTML (HyperText Markup Language) es el lenguaje que nos va a ocupar en este manual, que no tiene obligatoriamente qué cumplir esas reglas, pero sí es muy recomendable. Esto es importante por varias razones: entre ellas y principalmente porque seguir estas normas hará que nuestro código sea más claro y de mejor calidad, y por otro lado tendremos el aspecto de la compatibilidad con los estándares de la W3C.

1.1.1. Utilización de etiquetas.

Las etiquetas van a ser los elementos que estructuren el documento. Ejemplos de etiquetas son: <html>, <img>.
Podemos ver que siempre utilizan los signos de menor y mayor, y entre ellos siempre va el nombre.
Además, por lo general tendremos lo que se conoce como etiqueta de cierre: </html> o </img>, que no serán más que la etiqueta inicial con una barra antes del nombre. Pero no todas las etiquetas tienen inicio y cierre, por ejemplo
 no requiere nada más.
Vamos ahora a ver cuáles son esas reglas que impone XHTML y que nosotros en HTML 4.01 utilizaremos como sugerencias para obtener un mejor código:

- Las etiquetas deberán estar escritas en minúscula. No es que en mayúsculas no funcionen, pero es altamente recomendable que todas las etiquetas estén escritas de la misma forma.

- Sólo podrá haber un único elemento raíz, esto es, una etiqueta que engloba a todas las demás. En nuestro caso <html></html>
- Los valores de los atributos deben ir siempre entre comillas. Como veremos, muchas etiquetas pueden tener "opciones". A estas se les llama atributos:

1.1.2. Compatibilidad.

Por compatibilidad entendemos la capacidad de una página HTML para poder ser visualizada de forma correcta en cualquier navegador. Pero, ¿quién dice lo que está bien y lo que está mal?
El World Wide Web Consortium, es, como su nombre indica, un consorcio internacional cuya tarea es la emisión de recomendaciones relacionadas con la web, entendiendo recomendación como normalización. Es decir, es el organismo que dice lo que es estándar y lo que no en el mundo de internet.
Si queremos que una página que hayamos diseñado y programado funcione correctamente en cualquier navegador de cualquier sistema operativo, deberemos utilizar lenguajes ya aprobados. Respecto al
HTML, el estándar actualmente aceptado es el 4.01 y el 5 está en fase de aprobación y estandarización.
¿Quiere decir esto que HTML 5 no funciona? No, funciona en una mayoría de navegadores incluyendo sus distintas versiones, pero podemos encontrar diferencias entre unos y otros, lo cual nos obliga (y estos es una recomendación aplicable a todo el manual) a verificar lo que hagamos en, al menos, los principales navegadores.

1.1.3. Editores de texto.

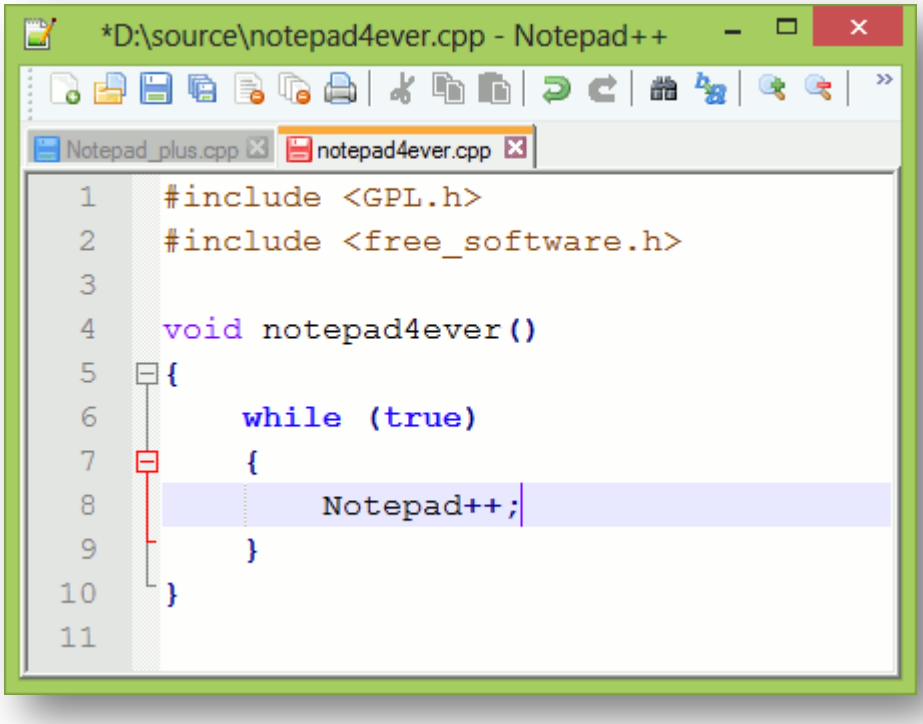
En este apartado vamos tratar de decidir con que herramienta crearemos nuestras páginas web. Inicialmente lo único que necesitamos es un editor de texto plano, tipo bloc de notas, pero si bien para hacer pequeños retoques nos sería suficiente, debemos pedir algo más a nuestro editor de HTML. Los aspectos que debemos pedir a esta herramienta son:

1. Que utilice una gama de colores para diferenciar etiquetas, selectores y propiedades.
2. Que cuando esté escribiendo el código nos sugiera posibilidades: de esta manera evitamos tener que recordar al pie de la letra la sintaxis y de sobre todo evitamos errores en el código. A esto se le conoce como ayuda contextual.
3. Como último deseo, podemos exigir que nos pueda ser útil además para desarrollar páginas más avanzadas, por ejemplo con css, javascript, php, etc. Esta tercera exigencia se retroalimenta de las dos anteriores, ya que podemos pedir que cada lenguaje adopte un color y además que nos sugiera el código tanto si estamos escribiendo HTML, como javascript e incluso PHP o java. De esta manera no tenemos que estar cambiando de editor según trabajemos con uno u otro.

Hagamos un repaso de herramientas partiendo de las más sencillas a las más complejas.

En primer lugar no fijaremos en notepad++ (http://notepad-plusplus.org/), que según su propia página web: es un editor de código “libre” que reemplaza al bloc de notas y que es compatible con varios lenguajes. Se ejecuta en el entorno MS Windows y su uso se rige por la licencia GPL. Equivalentemente en Linux tendremos gedit, scite o incluso emacs.

Como se puede observar se juega con el doble significado de la palabra free en inglés: gratis y libre. Pues bien notepad++ es ambas cosas y además es un editor sencillo que cumple el primer requisito:
Vemos que asigna un color distinto en función del tipo de código. También y por medio de un sistema de plugins podemos hacer que nos sugiera la sintaxis.



Es un programa completo y muy sencillo, la descarga ocupa tan solo 5,65 MB e instalado no mucho más. Además cuenta con la ventaja de que puede adaptarse a diversos lenguajes.
Otro tipo de editores HTML son aquellos que se nombran como tales.
En esta categoría podemos encuadrar a soluciones de pago tipo dreamweaver o alternativas gratuitas como BlueGriffon o Bluefish disponibles para Windows,

linux y MacOS X. Son parecidos entre sí, sugieren el código según se escribe y utilizan colores para diferenciar.
Así mismo permiten la estructuración del código, algo muy útil.

Hasta ahora tan solo hemos hablado de utilizar estos programas de una única forma, a través de su editor de código. Cada uno de los editores anteriores tienen además otro modo que puede parecernos a priori la solución más rápida y que menos esfuerzo nos exigirá este modo de codificar HTML (o cualquier otro lenguaje). Es utilizando herramientas WYSIWYG es decir What You See Is What You Get (en español, "lo que ves es lo que obtienes"). Para aprender rápido la diferencia entre un modo y otro, veamos en una imagen como obtener una página con HTML de las dos formas:

La vista dividir de Adobe Dreamweaver Cs6 nos permite ver a la vez la vista final del diseño y su estructura en código html.

Ahora puede surgir una pregunta ¿para qué queremos aprender código si la vista Diseño es mucho más sencilla de utilizar? Pues bien, la respuesta es sencilla y se puede resumir en dos puntos:

1. Los editores en modo Diseño nos pueden solucionar muchas cosas sencillas, pero si queremos hacer algo más complejo, es mejor y más rápido hacerlo en modo edición de código.
2. En segundo lugar, la vida de un programador, en este caso de webs, no está ocupada al 100% con nuevos desarrollos: al contrario, a menudo el trabajo consiste en retocar, arreglar o manipular lo que se hizo hace un tiempo o lo que hizo otra persona, y en este caso es imprescindible entender y manejar el código, donde los editores en modo diseño no son útiles.

Por último, dentro del software útil para elaborar webs, podemos referirnos a lo que se conoce como IDE (Integrated development enviroment, o en español entorno de desarrollo integrado). Es un conjunto de programas que forma un entorno gráfico a través del cual se facilita la tarea de programación. Este entorno puede ser utilizado para un solo lenguaje o adaptarse para varios. Si

bien empezamos el manual diciendo que HTML no es lenguaje de programación, la programación de páginas web involucra una serie de lenguajes, unos de programación y otros no, que tendrán que ser utilizados simultáneamente para obtener páginas de calidad. Será habitual encontrar páginas con HTML, CSS (hojas de estilo), javascript e incluso php. Estos dos últimos sí son lenguajes de programación y en cualquier caso desarrollando a un nivel profesional no es extraño que los programadores utilicen alguno de estos IDE.

Dentro de esta categoría tenemos Eclipse. Este software es de código abierto y multiplataforma, es decir puede descargarse y utilizarse libremente y funciona tanto en entorno Windows como Linux o Macintosh.
Pensado inicialmente para Java y desarrollado originalmente por IBM, es utilizado ampliamente en el mundo de la programación con diferentes lenguajes gracias a su sistema de plugins.
Mediante estos podemos extender la funcionalidad del programa y generar código en múltiples lenguajes.

En concreto y en el ámbito que nos interesa existe una adaptación de Eclipse pensada para el desarrollo web llamada Aptana Studio. Este IDE nos ofrece un entorno integrado y eficiente que cumple los tres requisitos que nos pusimos anteriormente.

1.2. Estructura de un documento creado con lenguaje de marcas.

HTML5 no es una nueva versión del antiguo lenguaje de etiquetas, ni siquiera una mejora de esta ya antigua tecnología, sino un nuevo concepto para la construcción de sitios web y aplicaciones en una era que combina dispositivos móviles, computación en la nube y trabajos en red.
HTML5 provee básicamente tres características: estructura, estilo y funcionalidad.
HTML5 es considerado el producto de la combinación de HTML, CSS y Javascript. Estas tecnologías son altamente dependientes y actúan como una sola unidad organizada bajo la especificación de HTML5. HTML está a cargo de la estructura, CSS presenta esa estructura y su contenido en la pantalla y Javascript hace el resto.

1.2.1. Estructura global.

Los documentos HTML se encuentran estrictamente organizados. Cada parte del documento está diferenciada, declarada y determinada por etiquetas específicas.

- **<!DOCTYPE>**

En primer lugar necesitamos indicar el tipo de documento que estamos creando. Esto en HTML5 es extremadamente sencillo:

```
<!DOCTYPE html>
```

> **IMPORTANTE:**
> Esta línea debe ser la primera línea del archivo, sin espacios o líneas que la precedan. De esta forma, el modo estándar del navegador es activado y las incorporaciones de HTML5 son interpretadas siempre que sea posible, o ignoradas en caso contrario.

- **<html>**

Luego de declarar el tipo de documento, debemos comenzar a construir la estructura HTML. Como siempre, la estructura tipo árbol de este lenguaje tiene su raíz en el elemento <html>. Este elemento envolverá al resto del código:

```
<!DOCTYPE html>
<html lang=”es”>
</html>
```

El atributo **lang** en la etiqueta de apertura **<html>** es el único atributo que necesitamos especificar en HTML5. Este atributo define el idioma del contenido del documento que estamos creando, en este caso **es** por español.

> **IMPORTANTE:**
> Para encontrar otros lenguajes para el atributo **lang** puede visitar el siguiente enlace:
> *www.w3schools.com/tags/ref_language_codes.asp*.

- **<head>**

El código HTML insertado entre las etiquetas **<html>** tiene que ser dividido entre dos secciones principales. Al igual que en versiones previas de HTML, la primera sección es la cabecera y la segunda el cuerpo. El siguiente paso, por lo tanto, será crear estas dos secciones en el código usando los elementos **<head>** y **<body>** ya conocidos**.**
El elemento **<head>** va primero, por supuesto, y al igual que el resto de los elementos estructurales tiene una etiqueta de apertura y una de cierre:

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

Dentro de las etiquetas **<head>** definiremos el título de nuestra página web, declararemos el set de caracteres correspondiente, proveeremos información general acerca del documento e incorporaremos los archivos externos con estilos, códigos Javascript o incluso imágenes necesarias para generar la página en la pantalla.
Excepto por el título y algunos íconos, el resto de la información incorporada en el documento entre estas etiquetas es invisible para el usuario.

- **<body>**

La siguiente gran sección que es parte principal de la organización de un documento HTML es el cuerpo. El cuerpo representa la parte visible de todo documento y es especificado entre etiquetas **<body>**. Estas etiquetas tampoco han cambiado en relación a versiones anteriores de HTML:

```
<!DOCTYPE html>
<html lang="es">
<head>
</head>
<body>
</body>
</html>
```

- **<meta>**

Es momento de construir la cabecera del documento. Algunos cambios e innovaciones fueron incorporados dentro de la cabecera, y uno de ellos es la etiqueta que define el juego de caracteres a utilizar para mostrar el documento. Ésta es una etiqueta **<meta>** que especifica cómo el texto será presentado en pantalla:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset=”iso-8859-1”>
</head>
<body>
</body>
</html>
```

La innovación de este elemento en HTML5, como en la mayoría de los casos, es solo simplificación. La nueva etiqueta **<meta>** para la definición del tipo de

caracteres es más corta y simple. Por supuesto, podemos cambiar el tipo **iso-8859-1** por el necesario para nuestros documentos y agregar otras etiquetas **<meta>** como **description** o **keywords** para definir otros aspectos de la página web, como es mostrado en el siguiente ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<meta name=”description” content=”Primer ejemplo de página con HTML5”>
<meta name=”keywords” content=”HTML5, CSS3, JavaScript”>
</head>
<body>
</body>
</html>
```

IMPORTANTE:
En HTML5 no es necesario cerrar etiquetas simples con una barra al final, pero recomendamos utilizarlas por razones de compatibilidad.

El ejemplo anterior de código se podría escribir de la siguiente manera:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1" />
<meta name="description" content=" Primer ejemplo de página con HTML5" />
<meta name="keywords" content="HTML5, CSS3, JavaScript" />
</head>
<body>
</body>
</html>
```

- **<title>**

La etiqueta **<title>**, como siempre, simplemente especifica el título del documento, y no hay nada nuevo para comentar:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<meta name="description" content="Ejemplo de HTML5">
<meta name="keywords" content="HTML5, CSS3, JavaScript">
<title>Este texto es el título del documento</title>
```

```
</head>
<body>
</body>
</html>
```

Conceptos básicos:
El texto entre las etiquetas **<title>** es el título del documento que estamos creando. Normalmente este texto es mostrado en la barra superior de la ventana del navegador.

- **<link>**

Otro importante elemento que va dentro de la cabecera del documento es **<link>**. Este elemento es usado para incorporar estilos, códigos Javascript, imágenes o iconos desde archivos externos. Uno de los usos más comunes para **<link>** es la incorporación de archivos con estilos CSS:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>
</body>
</html>
```

En HTML5 ya no se necesita especificar qué tipo de estilos estamos insertando, por lo que el atributo **type** fue eliminado. Solo necesitamos dos atributos para incorporar nuestro archivo de estilos: **rel** y **href**. El atributo **rel** significa “relación” y es acerca de la relación entre el documento y el archivo que estamos incorporando por medio de **href**.
En este caso, el atributo **rel** tiene el valor **stylesheet** que le dice al navegador que el archivo **misestilos.css** es un archivo CSS con estilos requeridos para presentar la página en pantalla.

Estilos Css:
Un archivo de estilos es un grupo de reglas de formato que ayudarán a cambiar la apariencia de nuestra página web (por ejemplo, el tamaño y color del texto). Sin estas reglas, el texto y cualquier otro elemento HTML sería mostrado en pantalla utilizando los estilos estándar provistos por el navegador.
Los estilos son reglas simples que normalmente requieren solo unas pocas líneas de código y pueden ser declarados en el mismo documento. No es estrictamente necesario obtener esta información de archivos externos pero es una práctica recomendada. Cargar las reglas CSS desde un documento externo (otro archivo) nos permitirá organizar el documento principal, incrementar la velocidad de carga y aprovechar las nuevas características de HTML5.

1.2.1. Estructura del Body.

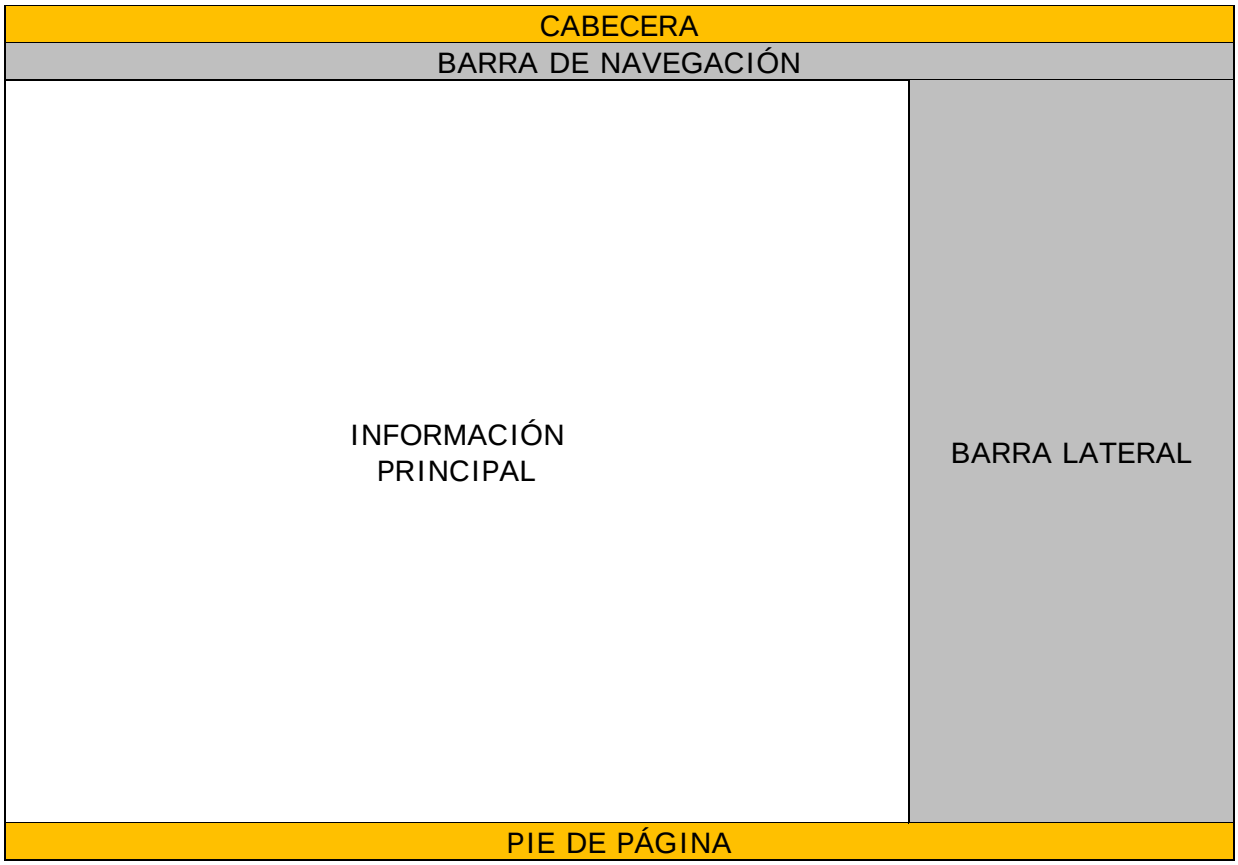
La estructura del cuerpo (el código entre las etiquetas **<body>**) generará la parte visible del documento. Este es el código que producirá nuestra página web.

Uno de los primeros elementos provistos para este propósito fue **<table>**. Las tablas permitían a los diseñadores acomodar datos, texto, imágenes y herramientas dentro de filas y columnas de celdas, incluso sin que hayan sido concebidas para este propósito. En los primeros días de la web, las tablas fueron una revolución, con respecto a la visualización de los documentos y la experiencia ofrecida a los usuarios.
Más adelante, el elemento ** ** comenzó a dominar la escena gradualmente. Reemplazando la función de **<table>**, el uso de ** ** nos permite lograr lo mismo con menos código, facilitando de este modo la creación, permitiendo portabilidad y ayudando al mantenimiento de los sitios web.
Con el surgir de webs más interactivas y la integración de HTML, CSS y Javascript, el uso de ** ** se volvió una práctica común.
Pero este elemento, así como **<table>**, no provee demasiada información acerca de las parte del cuerpo que está representando. Desde imágenes a menús, textos, enlaces, códigos, formularios, cualquier cosa puede ir entre las etiquetas de apertura y cierre de un elemento ** **.
** ** solo especifica una división en el cuerpo, como la celda de una tabla, pero no expresa qué clase de división es, cuál es su propósito o qué contiene.
Para los usuarios estas claves o indicios no son importantes, pero para los navegadores la correcta interpretación de qué hay dentro del documento que se está procesando puede ser crucial en muchos casos. A partir de la revolución de los dispositivos móviles y el surgir de diferentes formas en que la gente accede a la web, la identificación de cada parte del documento es una tarea que se ha vuelto más relevante que nunca.
Considerando todo esto, HTML5 incorpora nuevos elementos que ayudan a identificar cada sección del documento y organizar el cuerpo del mismo. En

HTML5 las secciones más importantes son diferenciadas y la estructura principal ya no depende más de los elementos ** ** o **<table>**

Normalmente una página o aplicación web está dividida entre varias áreas visuales para mejorar la experiencia del usuario y facilitar la interactividad. Las palabras claves que representan cada nuevo elemento de HTML5 están íntimamente relacionadas con estas áreas.

Veamos una representación visual de un clásico diseño web:



- En la parte superior, descripto como **Cabecera**, se encuentra el espacio donde usualmente se ubica el logo, título, subtítulos y una corta descripción del sitio web o la página.
- Inmediatamente debajo, podemos ver la **Barra de Navegación** en la cual casi todos los desarrolladores ofrecen un menú o lista de enlaces con el propósito de facilitar la navegación a través del sitio. Los usuarios son guiados desde esta barra hacia las diferentes páginas o documentos, normalmente pertenecientes al mismo sitio web.
- El contenido más relevante de una página web se encuentra, en casi todo diseño, ubicado en su centro. Esta sección presenta información y enlaces valiosos. La mayoría de las veces es dividida en varias filas y columnas.

En el ejemplo anterior se utilizaron solo dos columnas: **Información Principal** y **Barra Lateral**, pero esta sección es extremadamente flexible y normalmente diseñadores la adaptan acorde a sus necesidades insertando más columnas,

dividiendo cada columna entre bloques más pequeños o generando diferentes distribuciones y combinaciones.

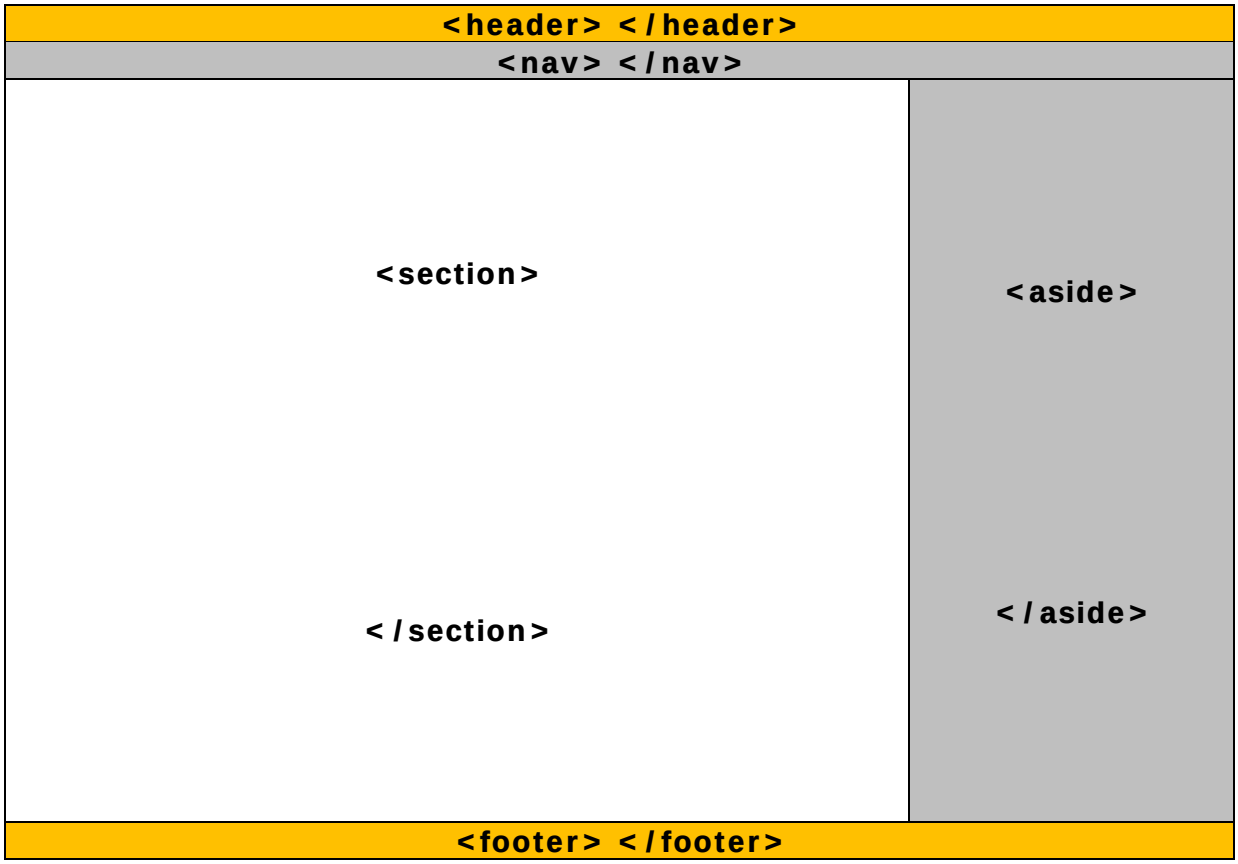
- En la base de un diseño web clásico siempre nos encontramos con una barra más que aquí llamamos **pie de página**. Esta es el área en donde normalmente se muestra información acerca del sitio web, el autor o la empresa, además de algunos enlaces con respecto a reglas, términos y condiciones y toda información adicional que el desarrollador considere importante compartir.

El **pie de página** es un complemento de la **Cabecera** y es parte de lo que se considera estos días la estructura esencial de una página web, como podemos apreciar en el siguiente ejemplo:

1.- Cabecera 2.- Barra de Navegación 3.- Información Principal
4.- Barra lateral 5.- Pie de página.

Esta simple representación de un blog nos puede ayudar a entender que cada sección definida en un sitio web tiene un propósito.

HTML5 considera esta estructura básica y provee nuevos elementos para diferenciar y declarar cada una de sus partes. A partir de ahora podemos decirle al navegador para qué es cada sección:



- **<header>**

Uno de los nuevos elementos incorporados en HTML5 es **<header>**. El elemento **<header>** no debe ser confundido con **<head>** usado antes para construir la cabecera del documento. Del mismo modo que **<head>**, la intención de **<header>** es proveer información introductoria (títulos, subtítulos, logos), pero difiere con respecto a **<head>** en su alcance. Mientras que el elemento **<head>** tiene el propósito de proveer información acerca de todo el documento, **<header>** es usado solo para el cuerpo o secciones específicas dentro del cuerpo:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>Aquí va el título principal del sitio web</h1>
```

```
</header>
</body>
</html>
```

Conceptos básicos:
Entre las etiquetas **<header>** hay un **<h1>**. El elemento **<h1>** es un viejo elemento HTML usado para definir títulos. El número indica la importancia del título. El elemento **<h 1>** es el más importante y **<h6>** el de menor importancia, por lo tanto **<h1>** será utilizado para mostrar el título principal y los demás para subtítulos o subtítulos internos.

- **<nav>**

Siguiendo con nuestro ejemplo, la siguiente sección es la **Barra de Navegación**. Esta barra es generada en HTML5 con el elemento **<nav>**:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>A cerca de...</li>
<li>Imágenes</li>
<li>Contacto</li>
</ul>
</nav>
</body>
</html>
```

Como se puede apreciar en el ejemplo anterior, 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.

Anteriormente dijimos que la estructura y el orden que elegimos para colocar los elementos HTML5 dependen de nosotros. Esto significa que HTML5 es versátil y

solo nos otorga los parámetros y elementos básicos con los que trabajar, pero cómo usarlos será exclusivamente decisión nuestra. Un ejemplo de esta versatilidad es que el elemento **<nav>** podría ser insertado dentro del elemento **<header>** o en cualquier otra parte del cuerpo. Sin embargo, siempre se debe considerar que estas etiquetas fueron creadas para brindar información a los navegadores y ayudar a cada nuevo programa y dispositivo en el mercado a identificar las partes más relevantes del documento. Para conservar nuestro código portable y comprensible, recomendamos como buena práctica seguir lo que marcan los estándares y mantener todo tan claro como sea posible.
El elemento **<nav>** fue creado para ofrecer ayuda para la navegación, como en menús principales o grandes bloques de enlaces, y debería ser utilizado de esa manera.

Conceptos básicos:
Entre las etiquetas **<nav>** hay dos elementos que son utilizados para crear una lista. El propósito del elemento **<ul>** es definir la lista. Anidado entre las etiquetas **<ul>** encontramos varias etiquetas **<li>** con diferentes textos representando las opciones del menú. Las etiquetas **<li>**, como probablemente ya se ha dado cuenta, son usadas para definir cada ítem de la lista. El propósito de este libro no es enseñarle conceptos básicos sobre HTML, si necesita más información acerca de elementos regulares de este lenguaje visite nuestro sitio web y siga los enlaces correspondientes a este capítulo.

- **<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>**:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>A cerca de…</li>
<li>Imágenes</li>
<li>Contacto</li>
</ul>
</nav>
<section>
</section>
</body>
</html>
```

- **<aside>**

En un típico diseño web la columna llamada **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, que apuntan a cada una de las entradas del blog y ofrecen información adicional s. La información dentro de esta barra está relacionada con la información principal pero no es relevante por sí misma.

En HTML5 podemos diferenciar esta clase secundaria de información utilizando el elemento **<aside>**:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>A cerca de…</li>
<li>Imágenes</li>
<li>Contacto</li>
</ul>
</nav>
<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.

<footer>

Para finalizar la construcción de una estructura elemental de un documento HTML5, solo necesitamos un elemento más. Ya contamos con la cabecera del cuerpo, secciones con ayuda para la navegación, información importante y hasta una barra lateral con datos adicionales, por lo tanto lo único que nos queda por hacer es cerrar nuestro diseño para otorgarle un final al cuerpo del documento. HTML5 provee un elemento específico para este propósito llamado **<footer>**:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>A cerca de...</li>
<li>Imágenes</li>
<li>Contacto</li>
</ul>
</nav>
<section>
</section>
<aside>
<blockquote>Mensaje número uno</blockquote>
<blockquote>Mensaje número dos</blockquote>
</aside>
<footer>
Derechos Reservados &copy; 2010-2011
</footer>
</body>
```

```
</html>
```

Generalmente, el elemento **<footer>** representará el final del cuerpo de nuestro documento y tendrá el propósito descripto anteriormente. 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>**). Estudiaremos esta última característica más adelante.

1.2.2. Dentro del Body.

La mayoría de los elementos ya estudiados fueron creados para construir una estructura para el documento HTML que pueda ser identificada y reconocida por los navegadores y nuevos dispositivos.

Hemos aprendido acerca de las etiquetas **<body>**, **<header>**, **<nav>**,**<section>**, **<aside>** y **<footer>**. Pero ninguno de estos elementos declara algo acerca del contenido. Todos tienen un específico propósito estructural.

Ahora, con más profundidad nos introducimos dentro del documento, nos encontramos más cerca de la definición del contenido. Esta información estará compuesta por diferentes elementos visuales como títulos, textos, imágenes, videos y aplicaciones interactivas, entre otros.
Necesitamos poder diferenciar estos elementos y establecer una relación entre ellos dentro de la estructura.

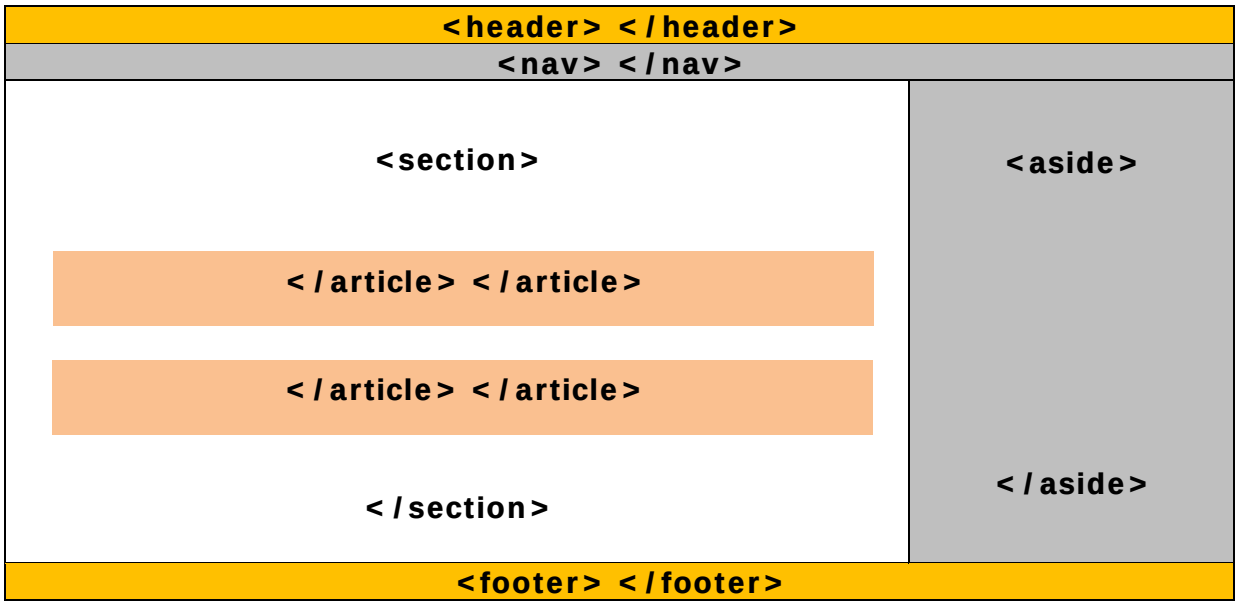
<article>

Del mismo modo que los blogs están divididos en entradas, los 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="iso-8859-1">
<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>A cerca de…</li>
<li>Imágenes</li>
<li>Contacto</li></ul>
</nav>
<section>
<article>
Este es el texto de mi primer mensaje
</article>
<article>
Este es el texto de mi segundo mensaje
</article>
</section>
<aside>
<blockquote>Mensaje número uno</blockquote>
<blockquote>Mensaje número dos</blockquote>
</aside>
<footer>
Derechos Reservados &copy; 2010-2011
</footer>
</body>
</html>
```

Como puede observarse en el código anterior, 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. Y al igual que cada elemento hijo del cuerpo, las etiquetas **<article>** son ubicadas una sobre otra.



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>** vistos 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:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>A cerca de…</li>
<li>Imágenes</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>Titulo 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 &copy; 2010-2011
</footer>
</body>
</html>
```

Los dos mensajes insertados en el código anterior 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.

- **<hgroup>**

Dentro de cada elemento **<header>**, en la parte superior del cuerpo o al comienzo de cada **<article>**, incorporamos elementos **<h1>** para declarar un título. Básicamente, las etiquetas **<h1>** son todo lo que necesitamos para crear una línea de cabecera para cada parte del documento, pero es normal que necesitemos también agregar subtítulos o más información que especifique de qué se trata la página web o una sección en particular. De hecho, el elemento **<header>** fue creado para contener también otros elementos como tablas de contenido, formularios de búsqueda o textos cortos y logos.
Para construir este tipo de cabeceras, podemos aprovechar el resto de las etiquetas H, como **<h1>, <h2>**, **<h3>**, **<h4>**, **<h5>** y **<h6>**, pero siempre considerando que por propósitos de procesamiento interno, y para evitar generar múltiples secciones durante la interpretación del documento por parte del navegador, estas etiquetas deben ser agrupadas juntas. Por esta razón, HTML5 provee el elemento **<hgroup>**:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>A cerca de...</li>
<li>Imágenes</li>
<li>Contacto</li>
```

```
</ul>
</nav>
<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>
<aside>
<blockquote>Mensaje número uno</blockquote>
<blockquote>Mensaje número dos</blockquote>
</aside>
<footer>
Derechos Reservados &copy; 2010-2011
</footer>
</body>
</html>
```

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.

En el ejemplo anterior, agregamos un subtítulo y datos adicionales a cada mensaje. Los títulos y subtítulos fueron agrupados juntos utilizando **<hgroup>**, recreando de este modo la jerarquía **<h1>** y **<h2>** en cada elemento **<article>**.

Importante:
El elemento **<hgroup>** es necesario cuando tenemos un título y subtítulo o más etiquetas H juntas en la misma cabecera. Este elemento puede contener solo etiquetas H y esta fue la razón por la que en nuestro ejemplo dejamos los datos adicionales afuera. Si solo disponemos de una etiqueta **<h1>** o la etiqueta **<h1>** junto con datos adicionales, no tendremos que agrupar estos elementos juntos. Por ejemplo, en la cabecera del cuerpo (**<header>**) no usamos este elemento porque solo tenemos una etiqueta H en su interior. **<hgroup>** fue creado solo con la intención de agrupar etiquetas H, exactamente como su nombre lo indica.

- **<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:

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<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>A cerca de…</li>
<li>Imágenes</li>
<li>Contacto</li>
</ul>
</nav>
<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
<figure>
<img src="http://minkbooks.com/content/myimage.jpg">
<figcaption>
Esta es la imagen del primer mensaje
</figcaption>
</figure>
<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>
<aside>
<blockquote>Mensaje número uno</blockquote>
<blockquote>Mensaje número dos</blockquote>
</aside>
<footer>
Derechos Reservados &copy; 2015-2016
</footer>
</body>
</html>
```

En el código anterior, en el primer mensaje, luego del texto insertamos una imagen (**<img src="http://minkbooks.com/content/myimage.jpg">**). 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.
También se puede observar un elemento extra dentro de **<figure>**. 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.

1.2.3. Diferencias entre HTML5 y HTML4.

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.

- **<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 la parte del texto que concuerda con el texto buscado:

```
<span>Mi <mark>coche</mark> es rojo</span>
```

Si un usuario realiza una búsqueda de la palabra “coche”, por ejemplo, los resultados podrían ser mostrados con el código del ejemplo anterior. La frase del ejemplo representa los resultados de la búsqueda y las etiquetas **<mark>** en el medio encierran lo que era el texto buscado (la palabra “coche”). En algunos navegadores, esta palabra será resaltada con un fondo amarillo por defecto, pero siempre podemos sobrescribir estos estilos con los nuestros utilizando CSS, como veremos en próximos capítulos.
En el pasado, normalmente obteníamos similares resultados usando el elemento **<b>**.
El agregado de **<mark>** tiene el objetivo de cambiar el significado y otorgar un nuevo propósito para éstos y otros elementos relacionados:

- **** es para indicar énfasis (reemplazando la etiqueta **<i>** que utilizábamos anteriormente).
- **** es para indicar importancia.
- **<mark>** es para resaltar texto que es relevante de acuerdo con las circunstancias.
- **<b>** debería ser usado solo cuando no hay otro elemento más apropiado para la situación.

- **<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, descargos, etc…

```
<small>Derechos Reservados &copy; 2015 Cecot</small>
```

- **<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…

```
<span>Amo la película <cite>Tentaciones</cite></span>
```

- **<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>,** como en el siguiente ejemplo:

```
<article>
<header>
<h1>Título del mensaje </h1>
</header>
Este es el texto del mensaje
<footer>
<address>
<a href=”http://www.jdgauchat.com”>JD Gauchat</a>
</address>
</footer>
</article>
```

- **<time>**

En cada **<article>** de nuestra última plantilla, 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 en HTML5 específico para este propósito. El elemento **<time>** nos permite declarar un texto comprensible para humanos y navegadores que representa fecha y hora:

```
<article>
<header>
<h1>Título del mensaje dos</h1>
<time datetime="2015-02-09" pubdate>publicado 09-02-2015</time>
</header>
Este es el texto del mensaje
</article>
```

En el Listado anterior, el elemento **<p>** usado en ejemplos previos fue reemplazado por el nuevo elemento **<time>** para mostrar la fecha en la que el mensaje fue publicado. 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: **2011-10-12T12:10:45**. También incluimos el atributo **pubdate**, el cual solo es agregado para indicar que el valor del atributo **datetime** representa la fecha de publicación.

Etiquetas eliminadas de HTML4

- <acronym>
- <applet>
- <basefont>
- <big>
-
- <dir>
- <font>
- <frame>
- <frameset>
- <noframes>
- <strike>
- <tt>
- <u>
- <xmp>

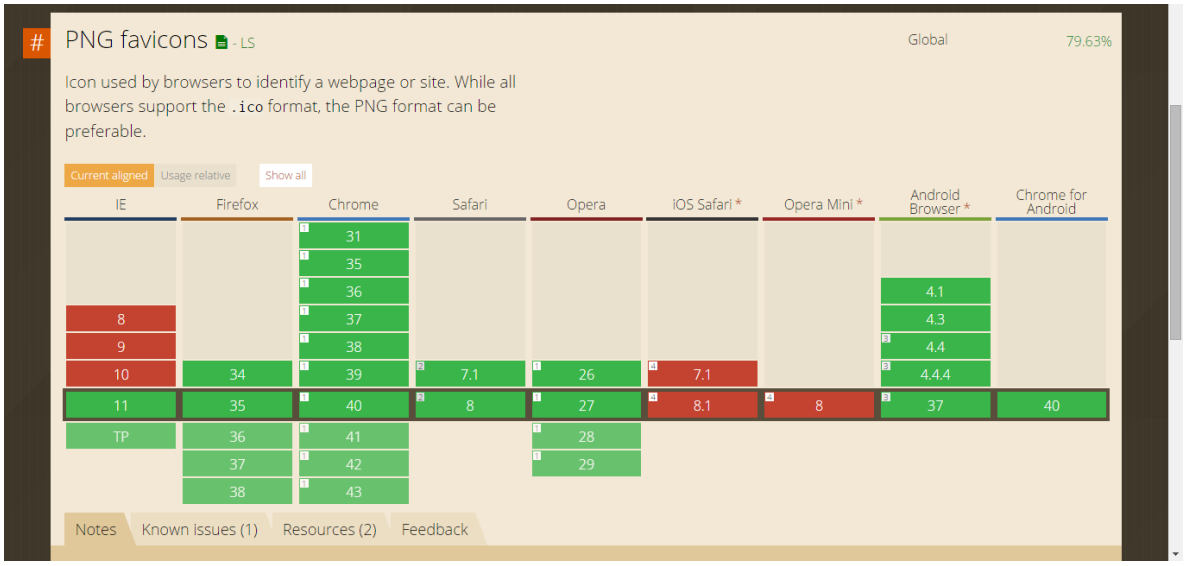
1.3. Navegadores web:

En términos generales debemos decir que ningún navegador es 100% compatible con todas las características de HTML5 y CSS3. Esta regla se mantiene aún hoy y seguirá por un tiempo, hasta que se estandaricen todas sus características. Claro que en este sentido encontraremos navegadores que presentan mayor compatibilidad que otros.
Para comprender un poco mejor esto, debemos saber que HTML5 comenzó a transitar su ruta hacia la estandarización en el año 2008, con sus primeros borradores. En el 2009 algunos navegadores comenzaron a ofrecer soporte a determinadas características, aunque recién en los años siguientes encontraríamos un soporte más extendido.

En este momento ya todos los navegadores soportan HTML5 y la mayoría de sus funciones se encuentran actualmente en estado de desarrollo.
Google Chrome ya implementa muchas de las características de HTML5 y además es una buena plataforma para pruebas. Por otro lado, Firefox es uno de los mejores navegadores para desarrolladores y también provee total soporte para HTML5.

Las versiones más recientes de los navegadores más populares del mercado soportan HTML5. Entre ellos: Internet Explorer 9 (muy mejorada en IE10), Firefox 9, Chrome 16, Safari 5.1 y Opera 11.60.

Tablas compatibilidad HTML5 en navegadores:
Conjunto de tablas donde mostrar el soporte de HTML5 en los principales navegadores de escritorio y móviles.
Pese al esfuerzo por la definitiva llegada de HTML5, lo cierto es que cada navegador implementa las mejoras de los estándares a su propio ritmo, por lo que tener a mano estas tablas puede resultar muy útil.
Aquellos desarrolladores y/o diseñadores web interesados pueden acceder a esta plataforma web desde la que informarse de forma actualizada acerca de los navegadores con soporte HTML 5 desde ***caniuse.com***.



Ejemplo compatibilidad favicon en png

1.4. Marcas para dar formato al documento

Vamos a ver una serie de etiquetas y atributos básicos para la creación de páginas webs. No entraremos a describir una a una todas las etiquetas y atributos de que HTML dispone. Simplemente indicaremos las que han sido más utilizadas en el pasado, aunque algunas se consideren deprecated (no recomendadas).

ETIQUETA	USO	OBSERVACIONES
…	Poner texto en negrita	Puede ser sustituido por CSS.
…	Poner texto en negrita	Puede ser sustituido por CSS.
…	Poner texto en cursiva	Puede ser sustituido por CSS.
…	Poner texto en cursiva	Puede ser sustituido por CSS.
<u>…</u>	Poner texto subrayado	Deprecated. Sustituir por CSS.
…	Poner texto más pequeño	Puede ser sustituido por CSS.
<big>…</big>	Poner texto más grande	Puede ser sustituido por CSS.
_…	Poner texto subíndice	Puede ser sustituido por CSS.
[…]	Poner texto superíndice	Puede ser sustituido por CSS.
<strike>…</strike>	Poner texto como tachado	Deprecated. Sustituir por CSS.
<s>…</s>	Poner texto como tachado	Deprecated. Sustituir por CSS.
~~…~~	Poner texto como tachado	Puede ser sustituido por CSS.

Algunas de las etiquetas que vamos a explicar están obsoletas (deprecated en inglés). Estas etiquetas en principio no deben de ser usadas porque dejarán de existir en las nuevas versiones a partir de HTML 5 y los navegadores es posible que dejen de reconocerlas en un futuro. Los motivos para seguir conociéndolas son:

- Son etiquetas que han sido muy populares en el pasado y nos podemos encontrar muchas páginas webs que hacen uso de ellas.

- Son etiquetas reconocidas por prácticamente todos los navegadores actuales.

- Son una buena forma de introducirnos en los lenguajes propios de desarrollos webs desde el punto de vista didáctico. Una vez se entiendan estos conceptos, es más fácil abordar aspectos más avanzados como las hojas de estilo CSS.

Las etiquetas deben rodear al texto. Es decir, la etiqueta debe abrirse y cerrarse, conteniendo el texto o la palabra que queramos transformar en su interior. Por ejemplo:

Este texto aparecerá escrito en negrita. Se pueden combinar diferentes formatos, o sea, diferentes etiquetas. Por ejemplo, si queremos que un texto esté en negrita y en cursiva escribiríamos esto: Este texto aparecerá escrito en negrita y en cursiva.

Cuando combinemos, debemos tener cuidado a la hora de cerrar las etiquetas. Debemos cerrar las etiquetas por orden, de la más interior a la más exterior.

- **NEGRITA**

Existen dos etiquetas que hacen que nuestro texto se convierta en negrita. La utilización de cualquiera de ellas es en principio indiferente (aunque pueda atribuírseles un significado diferente a cada una de ellas no vamos a prestarle atención a esto ahora). La primera es la etiqueta y la otra es la etiqueta . Aquí va un ejemplo de código y lo que veríamos en pantalla:

```
Esta palabra la vamos a poner en <b>negrita</b> y esta
otra <strong>también</strong>
```

Esta palabra la vamos a poner en **negrita** y esta otra **también**

Normalmente es preferible usar técnicas CSS en lugar de esta etiqueta, pero es una etiqueta que debemos conocer.

- **CURSIVA**

Para escribir un texto en cursiva se ha utilizado mucho en el pasado la etiqueta (que por supuesto debes cerrarla con la etiqueta). También se ha utilizado la etiqueta . Como en el caso de la negrita, aunque podrían atribuírseles distintos significados no vamos a prestarle atención a esta cuestión ahora. Aquí presentamos un ejemplo:

```
Esta palabra la vamos a poner en <i>cursiva</i> y esta
otra <em>también</em>
```

Esta palabra la vamos a poner en *cursiva* y esta otra *también*

- **SUBRAYADO U (DEPRECATED)**

Para que la palabra o el texto quedara subrayado se usó en el pasado el rodearlo con la etiqueta <u> y cerrarlo con su correspondiente etiqueta </u>. Así se subrayaría una frase:

```
<u>Así subrayaríamos una frase importante</u>
```

Así subrayaríamos una frase importante

Está etiqueta está obsoleta (deprecated), lo que significa que ya no se recomienda su uso. Para lograr el resultado deseado se deben usar hojas de estilo CSS como veremos más adelante.

- **PALABRAS MÁS GRANDES O MÁS PEQUEÑAS**

Puede que en una frase queramos destacar un palabra por medio de una variación de tamaño sin necesidad de utilizar los encabezados (los encabezados son etiquetas especiales que explicaremos más adelante). La variación de tamaño se podía conseguir gracias a las etiquetas <big> y <small>. Sus propios nombres en inglés nos indican cuáles eran sus funciones: <big> agrandará el texto y <small> lo disminuirá. No recomendamos su uso ya que las nuevas versiones de HTML no van a admitir esta etiqueta. La modificación del tamaño del texto se debe hacer a través de técnicas CSS.

Cada vez que se escribe una etiqueta big, se hace el texto un punto más grande. Estas etiquetas también se podían combinar, por lo que si escribimos dos veces la etiqueta <big>, haremos crecer la palabra dos puntos. Un ejemplo sería el siguiente:

```
Esta palabra se va a escribir <small>pequeñita</small>, esta se va a escribir <big>más grande</big> y ésta <big><big>más grande aún</big></big>.
```

Esta palabra se va a escribir <small>pequeñita</small>, esta se va a escribir <big>más grande</big> y ésta <big><big>más grande aún</big></big>.

- **SUPERÍNDICES Y SUBÍNDICES**

Mediante HTML también podemos escribir expresiones con símbolos matemáticos. Gracias a las etiquetas siguientes podrás escribir subíndices y superíndices fácilmente. La etiqueta <sub> te servirá para escribir un subíndice y <sup> será la etiqueta para un superíndice. Así nos queda un ejemplo como el siguiente:

```
Gracias a estas etiquetas podemos escribir cualquier expresión con símbolos matemáticos como esta: H<sub>2</sub>O o números elevados a potencias 7<sup>3</sup>.
```

Gracias a estas etiquetas podemos escribir cualquier expresión con símbolos matemáticos como esta: H_2O o números elevados a potencias 7^3.

Los subíndices y superíndices con estas etiquetas pueden ser sustituidos por técnicas de CSS, pero muchas personas prefieren usar estas etiquetas.

- **TEXTO TACHADO**

Existen tres etiquetas que se han venido usando para conseguir que un texto quede tachado. Hablamos de las etiquetas <strike>, <s> y <del>. Todas ellas ofrecen el mismo resultado. Aquí presentamos una muestra:

```
Puedo proceder a tachar una palabra <strike>así</strike>,
<s>así</s> o <del>así</del>
```

Puedo proceder a tachar una palabra ~~así~~, ~~así~~ o ~~así~~

La etiqueta strike está deprecated, lo que significa que ya no se recomienda su uso. La etiqueta s también fue deprecated, aunque a partir de HTML 5 se ha redefinido su significado. Para lograr el tachado de un texto se recomienda usar técnicas CSS (hojas de estilo) como veremos más adelante.

1.5. Enlaces y direccionamientos

Los hipervínculos son más de lo que parecen. Ellos muestran un texto en la página, pero cuando el usuario hace clic en el texto, el navegador carga una página diferente en el navegador.

Código para añadir o crear Vínculos (links)

```
<!DOCTYPE HTML>

<html lang = "es">

  <head>

    <title>Vinculos y links</title>

    <meta charset = "utf-8" />

  </head>

  <body>

    <h1>Crear Vínculos</h1>

    <h2>Vínculo Relativo a una Imagen</h2>

    <p>

     Este es el texto que vincula con la imagen de google al dar <a href =
"imgages/logogoogle.jpg">clic aqui</a>
```

```
    </p>

    <h2>Vínculo Absoluto a una URL</h2>

    <p>

     Este vínculo, nos envia a la <a href = "http://www.google.es ">Página de
google</a>

    </p>

  </body>

</html>
```

Los hipervínculos los agregamos a menudo para enlazar o tras paginas externas o internas en nuestra página, pero también podemos utilizarlos para crear hipervínculos a nuestra podría página web, veamos un ejemplo más claro:

```
<a href="http://www.elprofedemicurso.es" target="_blank" title="mensaje de
link">página web del profe de mi curso</a>
```

href = colocamos la dirección de la página a direccionar.

El href puede ser relativo o absoluto

- Relativa: El href puede ser un simple nombre.
    1. Si se vincula a un archivo que esté en el mismo directorio que la página Web, usted puede simplemente indicar el nombre del archivo.
    2. Esto se conoce como una referencia relativa debido a que el navegador asume que el archivo vinculado está en el directorio actual.
    3. En el primer ejemplo, el vínculo de la imagen apunta hacia una carpeta llamada (images), que está dentro del directorio y después a la imagen llamada (logogoogle.jpg).
- Absoluto: El href también puede ser una dirección Web completa.
    1. Si se prefiere, podemos dar toda la dirección de un sitio Web.

2. Esto se conoce como referencia absoluta, ya que explica cómo encontrar el archivo, independientemente de la ubicación de la página actual.

target = es una etiqueta para abrir esa página pero en una nueva pestaña puede ser muy molestoso para los usuarios así que no es utilizada muy frecuentemente
title= colocaremos el mensaje que queremos que salga cuando el mouse pasa por encima del hipervínculo.

- **HIPERVINCULOS DENTRO DE LA MISMA PÁGINA**

```
<a id="diseño">Hipervinculos</a>
```

Esta etiqueta es para identificar el hipervínculo dentro de la misma página así como Wikipedia.

```
<a href="#diseño">Hipervínculo</a>
```

El # indica que en alguna parte de la página hay un id al cual se va a re direccionar este hipervínculo.

Coloquemos el texto que será visible en el cuerpo entre la etiqueta <a> y </ a>. Cualquier texto que aparece entre las etiquetas <a> y </ a> se mostrará en la pantalla en un formato predefinido que lo indica como un enlace.

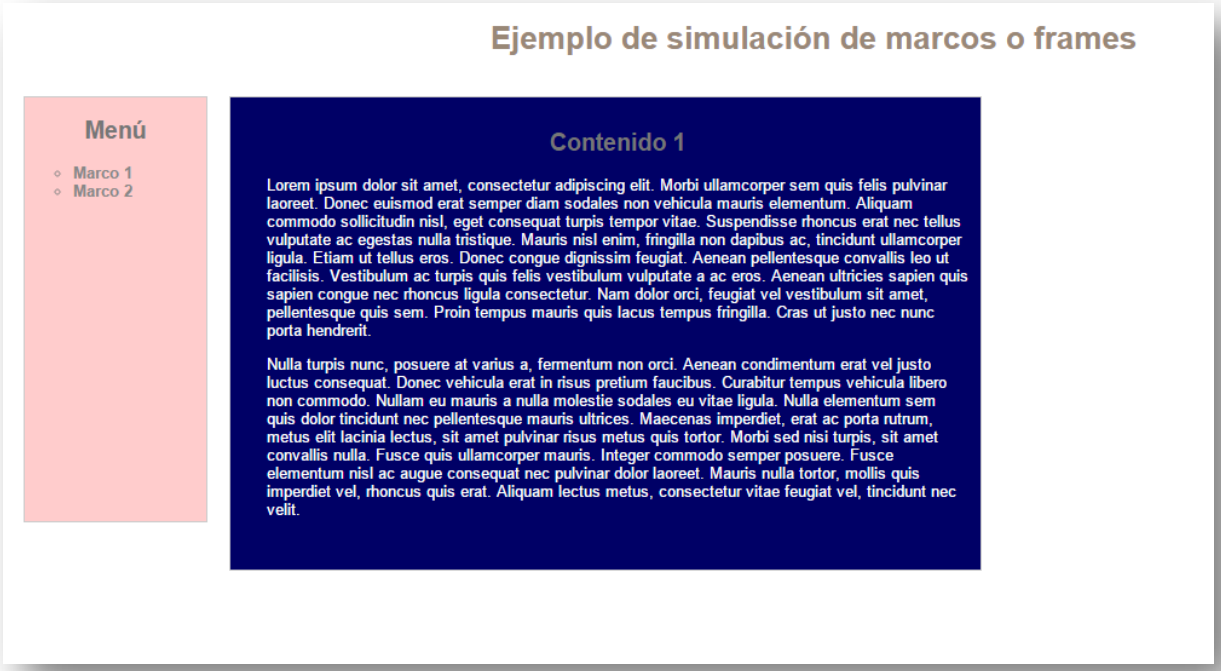
El formato por defecto de un enlace es un texto azul subrayado, pero más adelante se puede cambiar haciendo uso de CSS3.

1.6. Marcos y capas

Los marcos pasan a ser etiquetas obsoletas en HTML5. Los marcos solo nos daban dificultades en el posicionamiento o en la accesibilidad de la página.

Para sustituir ese efecto son varias las soluciones, desde el uso de algún framewoork como el uso de capas y CSS3.

En el siguiente ejemplo vamos a ver cómo conseguir el efecto de marcos con Html5 y CSS3:



- **Código HTML del ejemplo**

```
<!DOCTYPE html>
<html lang="es">
<head>
<meta charset="iso-8859-1">
<meta name="description" content="Ejemplo de Simular marcos">
<meta name="keywords" content="HTML5, Capas, CSS3">
<title>Simular marcos con Css</title>
 <link rel="stylesheet" type="text/css" href="estilo.css" />
</head>
<body>
<h1>Ejemplo de simulación de marcos o frames</h1>

<aside>


<h2>Menú</h2>
<ul>
  <li><a href="#marco1">Marco 1</a></li>
  <li><a href="#marco2">Marco 2</a></li>
</ul>

 </aside>
<section>
<div id="marco1">
<h2>Contenido 1</h2>
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Morbi ullamcorper sem quis felis pulvinar laoreet. Donec euismod erat semper diam sodales non vehicula mauris elementum. Aliquam commodo sollicitudin nisl, eget consequat
```

```
turpis tempor vitae. Suspendisse rhoncus erat nec tellus vulputate ac egestas
nulla tristique. Mauris nisl enim, fringilla non dapibus ac, tincidunt ullamcorper
ligula. Etiam ut tellus eros. Donec congue dignissim feugiat. Aenean pellentesque
convallis leo ut facilisis. Vestibulum ac turpis quis felis vestibulum vulputate a ac
eros. Aenean ultricies sapien quis sapien congue nec rhoncus ligula consectetur.
Nam dolor orci, feugiat vel vestibulum sit amet, pellentesque quis sem. Proin
tempus mauris quis lacus tempus fringilla. Cras ut justo nec nunc porta
hendrerit.</p>
<p>Nulla turpis nunc, posuere at varius a, fermentum non orci. Aenean
condimentum erat vel justo luctus consequat. Donec vehicula erat in risus
pretium faucibus. Curabitur tempus vehicula libero non commodo. Nullam eu
mauris a nulla molestie sodales eu vitae ligula. Nulla elementum sem quis dolor
tincidunt nec pellentesque mauris ultrices. Maecenas imperdiet, erat ac porta
rutrum, metus elit lacinia lectus, sit amet pulvinar risus metus quis tortor. Morbi
sed nisi turpis, sit amet convallis nulla. Fusce quis ullamcorper mauris. Integer
commodo semper posuere. Fusce elementum nisl ac augue consequat nec
pulvinar dolor laoreet. Mauris nulla tortor, mollis quis imperdiet vel, rhoncus quis
erat. Aliquam lectus metus, consectetur vitae feugiat vel, tincidunt nec velit.
</p>
</div>

<div id="marco2">
<h2>Contenido 2</h2>
<p>Nam urna nisl, blandit vitae molestie id, mattis ut augue. Ut sit amet libero
felis, at scelerisque neque. Nulla luctus porta sapien, vel imperdiet odio euismod
et. Donec id adipiscing felis. Nam pellentesque mollis pellentesque. Aliquam vel
diam nec ante consectetur auctor sed vitae augue. Fusce erat massa, volutpat ac
vulputate ac, dictum non arcu. Phasellus suscipit bibendum massa vel iaculis.
Nullam bibendum viverra orci id aliquet. Duis consequat neque id lectus aliquet
sed feugiat nisl adipiscing. Nunc suscipit est nec purus faucibus ultrices. </p>
<p>Mauris at velit nulla, id bibendum neque. Nullam et augue id elit convallis
fringilla vel non urna. Nulla lectus ante, fermentum nec dapibus vitae, tincidunt
vel lectus. Pellentesque luctus quam a mi mollis gravida. Curabitur est lorem,
aliquam sed varius nec, adipiscing vel sapien. Donec at tortor tellus. Suspendisse
vitae neque vitae odio mollis ultricies ac sed mi. Aenean et nisl non mauris
molestie tempor et sed eros. Maecenas scelerisque placerat eros et bibendum.
Mauris venenatis, enim sodales elementum consectetur, tellus nisi euismod sem,
dapibus pulvinar lectus lorem nec enim. Quisque auctor fermentum scelerisque.
Etiam sed nulla quis eros pellentesque convallis sit amet eu neque. Aenean
mattis placerat vehicula. Cras eget lacus odio. Cras commodo fringilla pharetra.
In eget lacus metus, dignissim rutrum diam. Fusce sed magna ut nulla cursus
feugiat. Fusce fringilla quam id quam auctor feugiat. Nulla tincidunt venenatis
diam, in varius libero imperdiet vitae. </p>
</div>
</section>

</body>
</html>
```

- **Código CSS del ejemplo**

```
html {
 font: normal 13px arial,helvetica,sans-serif;
 }

h1 {
 text-align: center;
 color: #987;
 }

h2 {
 text-align: center;
 color: #777;
 }

ul {
 list-style: circle;
 color: #888;
 }

ul li a {
 color: #888;
 text-decoration: none;
 font-weight: bold;
 }

ul li a:hover {
 color: #f00;
```

```
  }

p {
  color: #FFF;
  }

aside {
  border: solid 1px #ccc;
  position: absolute;
  left: 30px;
  top: 80px;
  width: 150px;
  height: 350px;
  background-color: #FCC;
}
section {
  border: solid 1px #ccc;
  position: absolute;
  overflow: hidden;
  left: 200px;
  top: 80px;
  width: 580px;
  height: 350px;
  padding: 20px 20px 20px 20px;
  background-color: #006;
}
 /* Capas del ejemplo  */
```

```
div#marco1, div#marco2 {
  height: 100%;
  width: 100%;
  padding: 10px 5px 5px 10px;
}
```

2. Imágenes y elementos multimedia.

2.1. Inserción de imágenes: formatos.

Uno de las principales decisiones a la hora de incluir gráficos en la web será elegir el formato correcto para cada tipo de imagen de manera que logremos una correcta relación entre la calidad visual de la misma y su peso en Kb.

Cuando hablamos de formatos de imagen en la web, tenemos que limitarnos a 4, ya que son los únicos soportados por los navegadores de internet.

- **GIF (Graphic Image File Format).**

  Sus características son:
  - Número de colores: de 2 a 256 de una paleta de 24 bits.
  - Formato de compresión sin pérdida basado en el algoritmo LZW.
  - Carga progresiva en el navegador.
  - Máscara de trasparencia de 1 bit.
  - Permite la animación simple.

Es el formato más adecuado para aquellas imágenes sencillas, de formas simples y en las que no existe un elevado número de colores.

- **JPEG (Joint Photographic Experts Group).**

Fue diseñado para la compresión de imágenes fotográficas, basándose en el hecho de que el ojo humano no es perfecto y no es capaz de captar toda la información que se puede almacenar el una imagen de 24 bits.
El formato JPEG intenta eliminar la información que el ojo humano no es capaz de distinguir, por eso se dice que posee un formato de compresión con pérdida, porque elimina información.

Las características de este formato son:

- Número de colores: 24 bits color o 8 bits B/N

- Elevado grado de posibilidad de compresión.
- Formato de compresión con pérdida.
- No permite trasparencia, ni canal alfa.
- No permite la animación.

Por regla general, es el más indicado para aquellas imágenes que son fotografias.

- **PNG** (Portable Network Graphics).

Proporciona un formato compresión de imágenes sin pérdida.

Las características de este formato son:

- Color indexado hasta 256 colores y TrueColor hasta 48 bits por pixel.
- Mayor compresión que el formato GIF (+10%)
- Compresión sin pérdida.
- Canal alfa. (Transparencia variable)
- No permite animación.

El más adecuado para imágenes de elementos renderizados, ya que se logran unos degradados muy suaves y una buena definición de las líneas.

A continuación vamos a ver algunas pruebas para que podamos ver cómo afecta a cada tipo de imagen el formato de exportación elegido tanto en calidad de la imagen como en peso:

El caso de una fotografía digital.

JPG	GIF	PNG
		
Jpg con 20% de compresión. 11Kb	Gif con 256 colores. 22 Kb	Png 24 bits. 68 Kb

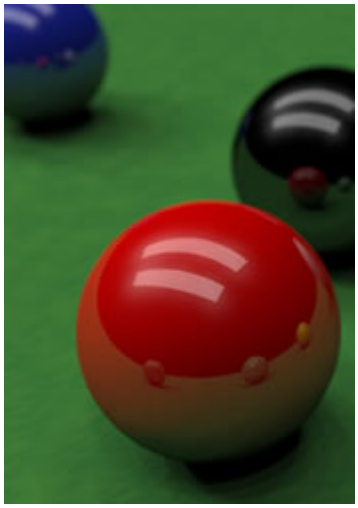
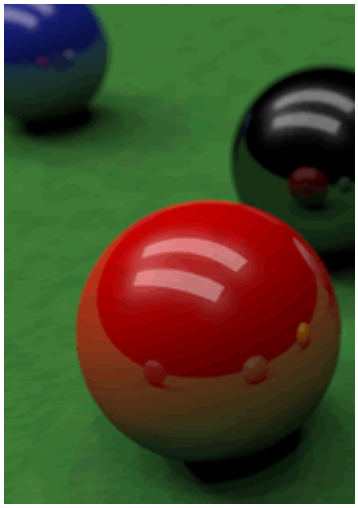
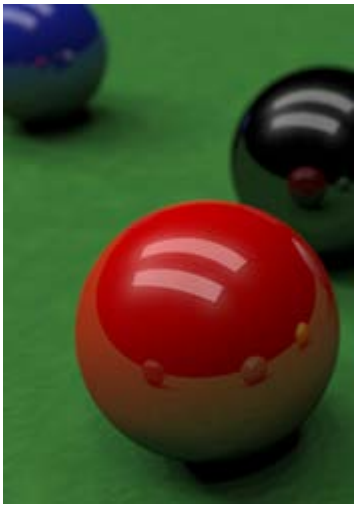
Vemos que para obtener una calidad similar los tamaños de la imágen son muy diferentes, en este caso, lo más adecuado será optar por el formato JPG.

El caso de una imagen simple

JPG	GIF	PNG
		
Jpg con 20% de compresión. 13Kb	Gif con 256 colores. 5 Kb	Png 24 bits. 8 Kb

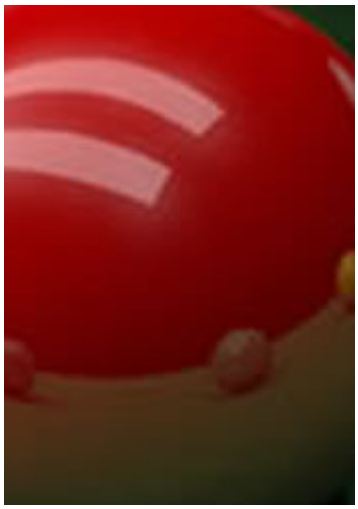
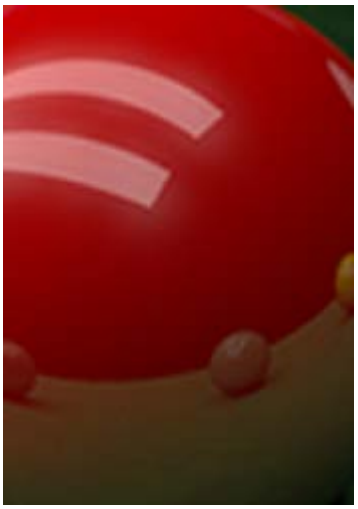
En el caso de una imagen simple (con pocos colores) vemos que el formato más adecuado es el .gif, ya que aunque la calidad de la imagen sea idéntica al png, el peso de la imagen es considerablemente inferior.

El caso de una imagen renderizada

JPG	GIF	PNG
		
Jpg con 20% de compresión. 6Kb	Gif con 256 colores. 13 Kb	Png 24 bits. 41 Kb

El formato png es el que más calidad nos ofrece, sin embargo, la opción Jpg parece la más adecuada por su relación calidad peso, será una decisión que debemos tomar sobre todo en función de la importancia de la imagen o el detalle que sea necesario mostrar.

Para ver mejor la diferencia en la calidad de las imágenes a continuación se muestra un detalle de ampliación realizada sobre las imágenes anteriores.

JPG	GIF	PNG
		
Jpg con 20% de compresión. 4 Kb	Gif con 256 colores. 10 Kb	Png 24 bits. 27 Kb

En la ampliación, podemos ver como el formato PNG mantiene mucha más calidad que el resto, esto sucederá siempre que trabajemos con imágenes con degradados muy suaves y líneas muy bien definidas.

2.2. Inserción de imágenes: etiqueta img y atributos.

Todas las páginas web acostumbran a tener un cierto número de imágenes, que permiten mejorar su apariencia, o dotarla de una mayor información visual.

Para insertar una imagen es necesario insertar la etiqueta **<img>**. Dicha etiqueta no necesita etiqueta de cierre.

El nombre de la imagen ha de especificarse a través del atributo **src**.

Por ejemplo, para insertar la siguiente imagen:

Habría que escribir:

```
<img src="images/logo_animales.jpg">
```

Teniendo en cuentra que la imagen se llama **logo_animales.jpg** y que está dentro de la carpeta **images**, que se encuentra en el mismo directorio que el documento actual (referencia relativa al documento).

Para trabajar de una forma más sencilla y ordenada, es recomendable que todos los documentos html se encuentren en un mismo directorio, y que dentro de este directorio existan diferentes carpetas para agrupar otros objetos, como puede ser una carpeta destinada a almacenar imágenes, o una carpeta destinada a almacenar archivos de audio, etc.

- **Texto alternativo**

Cuando una imagen no puede ser visualizada en el navegador, cosa que puede ocurrir al especificar mal el valor del atributo **src**, aparece un recuadro blanco con una **X** roja en su lugar, junto con el nombre de la imagen.

Podemos hacer que en lugar de mostrarse el nombre de la imagen aparezca el texto que nosotros deseemos, gracias al atributo **alt**.

Por ejemplo, imagina que deseamos mostrar una imagen llamada **gatito.gif**, con el texto alternativo **Imagen gato**, para ello insertamos el siguiente código:

```
<img src="/images/gatito.gif" alt="Imagen gato" >
```

El texto alternativo se muestra también al situar el puntero sobre la imagen. Si situamos el puntero sobre la imagen durante unos segundos, verás como aparece el texto **Imagen gato**.

El texto alternativo es muy útil a la hora de diseñar páginas más asequibles a los invidentes ya que los programas lectores son capaces de leer el texto alternativo.

- **Borde de una imagen**

En general, al visualizar una página en un navegador las imágenes aparecen sin ningún borde alrededor, pero es posible establecer uno a través del atributo **border**.

El atributo **border** puede tomar valores numéricos, que indican el grosor en píxeles del borde.

Por ejemplo, para insertar la siguiente imagen con borde:

Habría que escribir:

```
<img src="imagenes/logo_animales.jpg" border="4" >
```

Hay que tener en cuenta que el borde de la imagen siempre será de color negro, (siempre que no utilicemos Css) a no ser que la imagen contenga un enlace, en cuyo caso el color del borde será el color establecido para los vínculos.

Por ejemplo, para insertar la siguiente imagen con borde y con un enlace:

Habría que escribir:

```
<a href="http://www.cecot.org" target ="_blank" >

<img src="imagenes/logo_animales.jpg" border="4" ></a>
```

Si se desea establecer un vínculo sobre una imagen y no se desea que se muestre el borde (que por su color indica que existe dicho vínculo), es necesario establecer border="0".

- **Tamaño de una imagen**

Cuando insertamos una imagen, esta se muestra en los navegadores con su tamaño original, pero por diversos motivos puede interesarnos modificar dicho tamaño.

A través de los atributos **width** (anchura) y **height** (altura) puede modificarse el tamaño de la imagen. Dicho cambio de tamaño no se aplica directamente sobre el archivo de imagen, sino que lo que varía es la visualización de la imagen en el navegador.

El valor que pueden tomar los atributos **width** y **height** ha de ser un número, acompañado de **%** cuando se desee que sea en porcentaje con respecto a la página.

Por ejemplo, para insertar la siguiente imagen (cuyo tamaño original era de 122 píxeles de anchura y 71 píxeles de altura) con 200 píxeles de anchura y 80 píxeles de altura:

Habría que escribir:

```
<img src="imagenes/logo_animales.jpg" width="200" height="80">
```

Importante:

Al modificar el tamaño de la imagen a través de estos atributos es muy probable que la imagen resultante no sea de buena calidad, en comparación de cómo podría quedar modificándola desde un editor externo, como Photoshop, Gimp...

- **alineación de una imagen**

```
PerrosGatos<img scr="imagenes/logo_animales.jpg" align="middle">Una web de animales
```

Indica cómo se alinea el texto que sigue a la imagen con respecto a esta. Indicará si la primera frase del texto se colocará en la parte alta de la imagen, **top**, en el punto medio de la imagen, **middle**, o en la parte de abajo de la imagen, **bottom**.

También se pueden utilizar alineaciones un poco más avanzada,

- **texttop** se alinea justo al comienzo del texto más alto de la línea.
- **top** se alinea al tamaño del primer carácter de la línea.
- **absmiddle**, es el centro real de la imagen.
- **middle** se coloca el texto a partir del punto medio.
- **absmiddle** el texto aparece centrado con la imagen.
- **absbottom** coloca el texto justo al final de la imagen.

Se recomienda que se usen estos últimos al ser más precisa la alineación, aunque solo son válidos para los navegadores más avanzados.

2.3. Mapas de imágenes.

- **<MAP> <AREA>**

Puede hacer que parte de la imagen sea un enlace a otra página, es decir, puede hacer un mapa sobre la imagen de manera que secciones de la imagen sean enlaces. Las tags usadas para esto son:

Las tags **<map>.....</map>**identifican que vamos a crear un mapa de imágenes. Generalmente, lleva asociado el atributo **name**= al que le sigue entre comillas el nombre del mapa.

La tag **<area>** define las áreas que vamos a poder activar en esa imagen. A esta tag le acompañan los siguientes atributos:

shape= Entre comillas estableceremos el tipo de área a definir. Puede tratarse de rect (rectangular), poly (poligonal) o circle (circular).

Coords= Entre comillas indicaremos los pares de coordenadas de cada punto del área a activar. Estas coordenadas las podemos averiguar utilizando un programade edición de imágenes. En las áreas rectangulares deben especificarse las coordenadas de la esquina superior izquierda y las de la esquina inferior derecha. En las poligonales especificaremos las coordenadas de todos los vértices del área. En las circulares indicaremos las coordenadas del centro del círculo y el valor del radio.

href= Como ya sabe, indica la dirección, entre comillas, de la página web a la que accede si pinchamos en un área determinada.

Finalmente, debemos saber que para que una imagen sea tratada como un mapa, además del código anteriormente descrito, debemos incluir en la tag de imagen correspondiente a la imagen a mapear el atributo **usemap**="#nombre del mapa".

Importante:

Video demostración crear un mapa de imágenes con adobe dreamweaver cs6

http://youtu.be/qB5jgkPhwCA

2.4. Inserción de elementos multimedia: audio, video y downloads.

2.4.1. Formatos de video en HTML5.

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 video codificado con los codificadores Theora, H.264, VP6 o VP8, respectivamente. Esta es la lista de los más usados:

- **OGG** codificador de video Theora y audio Vorbis.
- **MP4** codificador de video H.264 y audio AAC.
- **FLV** codificador de video VP6 y audio MP3. También soporta H.264 y AAC.
- **WEBM** codificador de video 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 video Theora y audio Vorbis.
- **Google Chrome** codificador de video Theora y audio Vorbis. También soporta codificador de video H.264 y audio AAC.
- **Opera** codificador de video Theora y audio Vorbis.
- **Safari** codificador de video H.264 y audio AAC.
- **Internet Explorer** codificador de video H.264 y audio AAC.

2.4.2. Insertando video en HTML5.

Una de las características más mencionadas de HTML5 fue la capacidad de procesar video. Ahora que ya disponemos de soporte nativo para videos e incluso un estándar que nos permitirá crear aplicaciones de procesamiento de video compatibles con múltiples navegadores.

HTML5 ha introducido un elemento para insertar y reproducir video en un documento HTML.

- **<video>**

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:

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Reproductor de Video</title>
</head>
<body>
<section id="reproductor">
<video src="http://elprofedemicurso.es/videohtml5/trailer.mp4"
controls>
</video>
</section>
</body>
</html>
```

En teoría, el código anterior debería ser más que suficiente. Pero las cosas se vuelven un poco más complicadas en la vida real.

Primero 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.

El elemento <video> y sus atributos

Intentemos ignorar por un momento estas complicaciones y disfrutar de la simplicidad del 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, como en el siguiente ejemplo:

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Reproductor de Video</title>
</head>
<body>
<section id="reproductor">
<video id="medio" width="720" height="400" controls>
<source src=" http://elprofedemicurso.es/videohtml5/trailer.mp4">
<source src=" http://elprofedemicurso.es/videohtml5/trailer.ogg">
</video>
</section>
</body>
</html>
```

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.
- **loop** Si este atributo es especificado, el navegador comenzará a reproducir el video nuevamente cuando llega al final.
- **poster** Este atributo es utilizado para proveer una imagen que será mostrada mientras esperamos que el video comience a ser reproducido.
- **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.

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Reproductor de Video</title>
</head>
<body>
<section id="reproductor">
```

```
<video id="medio" width="720" height="400" preload controls
loop poster="http://elprofedemicurso.es/videohtml5/poster.jpg">
<source src="http://elprofedemicurso.es/videohtml5/trailer.mp4">
<source src="http://elprofedemicurso.es/videohtml5/trailer.ogg">
</video>
</section>
</body>
</html>
```

En el código anterior, el elemento **<video>** fue poblado con atributos. Debido a las diferencias en comportamiento entre un navegador y otro, algunos atributos estarán habilitados o deshabilitados por defecto, y algunos de ellos incluso no trabajarán en algunos navegadores o bajo determinadas circunstancias. Para obtener un control absoluto sobre el elemento **<video>** y el medio reproducido, deberemos programar nuestro propio reproductor de video en Javascript aprovechando los nuevos métodos, propiedades y eventos incorporados en HTML5.

2.4.3. Reproduciendo audio en HTML5.

Audio no es un medio tan popular como video en Internet.
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>**.

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Reproductor de Audio</title>
</head>
<body>
<section id="reproductor">
<audio src=" http://elprofedemicurso.es/audiohtml5/beach.mp3"
controls>
</audio>
</section>
</body>
</html>
```

- **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**. El primero indica que el audio no debería ser cacheado, normalmente con el propósito de minimizar tráfico innecesario. El segundo valor, **metadata**, recomendará al navegador obtener información sobre el medio (por ejemplo, la duración). El tercer valor, **auto**, es el valor configurado por defecto y le aconseja al navegador descargar el archivo tan pronto como sea posible.

Una vez más debemos hablar acerca de codificadores, y otra vez debemos decir que el código, mostrado en el ejemplo anterior, debería ser más que suficiente para reproducir audio en nuestro documento. Pero el formato MP3 está bajo licencia comercial, por lo que no es soportado por navegadores como Firefox u Opera. Vorbis (el codificador de audio del contenedor OGG) es soportado por esos navegadores, pero no por Safari e Internet Explorer. Por esta razón, nuevamente debemos aprovechar el elemento **<source>** para proveer al menos dos formatos entre los cuales el navegador pueda elegir:

```
<!DOCTYPE html>
<html lang="es">
<head>
<title>Reproductor de Audio</title>
</head>
<body>
<section id="reproductor">
<audio id="medio" controls>
<source src=" http://elprofedemicurso.es/audiohtml5/beach.mp3">
<source src=" http://elprofedemicurso.es/audiohtml5/beach.ogg">
</audio>
</section>
</body>
</html>
```

El código mostrado en el ejemplo anterior reproducirá música en todos los navegadores utilizando los controles por defecto. Aquellos que no puedan reproducir MP3 reproducirán OGG y viceversa.
Recordemos que MP3, al igual que MP4 para video, tienen uso restringido por licencias comerciales, por lo que solo podemos usarlos en circunstancias especiales, de acuerdo con lo determinado por cada licencia.
El soporte para los codificadores de audio libres y gratuitos (como Vorbis) se está expandiendo, pero llevará tiempo transformar este formato desconocido en un estándar.

3. Técnicas de Accesibilidad y Usabilidad.

3.1. Usabilidad web. Importancia de la Accesibilidad.

3.1.1. Concepto de Accesibilidad.

- **¿Qué es la accesibilidad?**

Cualidad de accesible.

- **¿Qué significa accesible?**

1. adj. Que tiene acceso.

2. adj. De fácil acceso o trato. 3. adj. De fácil comprensión, inteligible.

- **¿Qué es la accesiblidad web?**

Cualidad que tiene o no una web de ser accesible.

La **accesibilidad** es la parte de la **usabilidad** que implica la necesidad de facilitar el acceso a la web. Y una web para ser usable, debe ser accesible. No olvidemos que la usabilidad parte de los principios del diseño universal o diseño para todos.
Existen varias normas o estándares acerca de la accesibilidad: en España encontramos las normas UNE 139801-EX y UNE 139802-EX (que recogen más de cien requisitos de accesibilidad), y a nivel internacional podemos guiarnos por las pautas diseñadas por la "Iniciativa de Accesibilidad a la Web" (WAI) del World Wide Web Consortium (W3C), que contienen catorce pautas de diseño. En la WAI se han determinado tres niveles de profundidad al dotar de **accesibilidad** a los sitios web, por los que un proveedor de contenidos de páginas web:

- **Nivel A**: *debe* de satisfacer este punto de verificación.
- **Nivel AA**: *debería* satisfacer este punto de verificación.
- **Nivel AAA**: *puede* satisfacer este punto de verificación.

Entre las pautas están:

- Proporcionar alternativas equivalentes de contenido visual y auditivo.
- Asegurar que los textos y gráficos son comprensibles cuando se vean sin color.
- Utilizar marcadores y hojas de estilo apropiadamente.
- Identificar el lenguaje natural usado.

- Crear tablas que se transformen correctamente.
- Asegurar que las páginas que incorporan nuevas tecnologías se transformen correctamente.
- Proporcionar al usuario el control sobre los cambios de los contenidos "tiempo dependientes".
- Asegurar que el interfaz de usuario sigue los principios de un diseño accesible: funcionalidad de acceso independiente del dispositivo, teclado operable, voz automática, etc.
- Diseñar con independencia del dispositivo.
- Utilizar soluciones de accesibilidad provisionales de forma que las ayudas técnicas y los antiguos navegadores operen correctamente.
- Utilizar tecnologías W3C (de acuerdo con las especificaciones) y seguir las pautas de accesibilidad.
- Proporcionar información de contexto y orientación para ayudar a los usuarios a entender páginas o elementos complejos.
- Implementar mecanismos de navegación claros y consistentes (información orientativa, barras de navegación, un mapa del sitio, etc.) para incrementar la probabilidad de que una persona encuentre lo que está buscando en un sitio.
- Asegurar que los documentos son claros y simples para que puedan ser más fácilmente comprendidos.

3.1.2. Personas con discapacidad discapacidades atendidas por los estándares.

Personas con discapacidad y discapacidades atendidas por los estándares	
• No todos lo usuarios usan navegadores web • Existen dispositivos especiales para discapacitados • Existe software específico para discapacitados	• Usuarios Ciegos • Usuarios con Ceguera al Color • Usuarios con visión débil que no pueden leer texto pequeño • Usuarios con deficiencia auditiva o sordos • Usuarios que no usan ratón • Usuarios con discapacidades debidas a la artritis u otros problemas de control motor • Epilepsia fotosensible

¿La accesiblidad únicamente dirigida a discapacitados?

No. El concepto, como bien se ha definido inicialmente, es más general.

Podemos hablar de usuarios sin discapacidad, o incluso con cierta discapacidad, pero no significativa en cuanto al uso de la web se refiere, a los que se deba tener en cuenta, como por ejemplo:

- Usuarios con acceso a internet vía modem (56Kbits)
- Usuarios con resoluciones inferiores a 1024x768px y 16/24/32bits color. Usuarios con dispositivos de representación distintos a los comunes monitores, tanto CRT como TFT/Plasma. Por ejemplo, PDAs, GPS de los coches, etc...
- Usuarios que carecen de plugins específicos (Flash, SVG, MathXML, etc...) Usuarios con un hardware un poco obsoleto (p.e. grandes cantidades de javascript, mucho consumo de cpu y memoria).

¿Por qué tu sitio web debe ser accesible?

- Para acceder de forma más cómoda a la audiencia (usuarios potenciales) de tu sitio web.
- Para tomar una posición de liderazgo en tu sector particular de negocio.
- Para ganarte la fidelidad de tus usuarios.
- Para no realizar ningún tipo de discriminación ante posibles usuarios
- Es simplemente una cuestión de buen sentido comercial, al incrementar el número potencial de tus clientes.
- Para los sitios web institucionales la justificación es muy sencilla, debes facilitar la información a la sociedad.
- Cuestiones legales y normativas.

3.2. Usabilidad web. Importancia de la usabilidad.

3.2.1. Concepto de Usabilidad

- La usabilidad de un producto será lo fácil de utilizar que es.
- El grado en el que un producto puede ser usado por determinados usuarios para conseguir objetivos específicos con eficacia, eficiencia y satisfacción en un contexto específico de uso.
- Capacidad de un producto software de ser entendido, aprendido, usado y atractivo para el usuario, cuando es usado bajo unas condiciones específicas.

La Usabilidad no es un concepto nuevo:

- Marketing (Que espera el usuario de la herramienta)
- Interfaces Gráficas de Usuario (Como usa el usuario la herramienta)
- Control de Calidad (Fiabilidad y Eficacia de la herramienta)

3.2.2. La Usabilidad según el Perfil

- **Usuario:** Percibe la diferencia entre hacer una tarea de forma precisa y completa o no y que el proceso sea ameno o frustrante.
- **Desarrollador:** De él depende la diferencia entre el éxito o el fallo de un sistema.
- **Gestor:** Determina la productividad de la herramienta.

3.2.3. ¿Por qué invertir en Usabilidad?

- La usabilidad de un producto determina la decisión de una compra más que el precio del producto.
- **Menor tiempo de aprendizaje = Beneficios de Mantenimiento y Entrenamiento.**
- La satisfacción del usuario disminuye la incomodidad y el estrés.
- Diseños usables suelen implicar diseños más simples. Diseños más simples son más baratos de construir.
- Diseños centrados en el usuario tienen éxito en las aplicaciones de mercado.

Usabilidad = Simplicidad = Satisfacción del Usuario = Mayores Beneficios

3.2.4. Metodología para la Usabilidad

Ejemplos que mejoran la Usabilidad en la Web

- Resumen de una línea del contenido al principio de un portal
- Inclusión de una Caja de Búsquedas Internas
- Navegación Rápida por el Portal. Accesos rápidos. Rama actual de Navegación.
- Formato Simple en la Navegación
- Uso de imágenes, sonidos o animaciones no simplemente para decorar la página.

Ejemplos que NO mejoran la Usabilidad en la Web

- Descripción de la actividad del portal confusa o resumen ambiguo.
- Imágenes muy pequeñas de grandes imágenes.
- Formularios excesivamente restrictivos.

3.3. Aplicaciones para verificar la accesibilidad de sitios web estándares

Existen diferentes sistemas para comprobar si una página Web es accesible, entre ellos cabe reseñar una serie de herramientas automatizadas que ayudan a evaluar, a través de la verificación de los estándares de facto, la accesibilidad global que presentan los contenidos de un portal Web y determinar cuáles son los puntos fuertes y débiles susceptibles de mejora. A continuación, se recogen las herramientas de validación automática más extendidas y los principales programas para facilitar la accesibilidad y navegadores utilizados por personas que presentan alguna limitación ya sea de índole física o técnica.

- **Validadores de código del w3c**

http://www.codexexempla.org/traducciones/pautas-accesibilidad-contenido-web-2.0.htm

- **TAW3**

Programa de evaluación automática, de los niveles WAI, de accesibilidad desarrollado por la Fundación CTIC3, que sirve de ayuda a los desarrolladores de páginas Web y consigue que su trabajo cumpla las normas de accesibilidad.

Se puede obtener de forma gratuita en la siguiente dirección:

http://www.tawdis.net/

- **HiSoftware Cynthia Says**

Se trata de un conocido analizador on-line para el análisis de la accesibilidad de páginas Web.
Cynthia, se puede obtener de forma gratuita en la siguiente dirección:

http://www.contentquality.com/

3.4. Ejemplos sitios web usables y malos ejemplos

Primero empezaremos este apartado mencionando una muy buena página para profundizar sobre usabilidad:
http://webusable.com/wuExcelent.htm

- **Buen ejemplo:**

Un muy buen ejemplo de cómo orientar una web de ecommerce
www.regalador.com

Para empezar, los menús de navegación están perfectamente orientados al cliente, pues ayudan a este a encontrar los productos a través de sus gustos o intereses. Si avanzamos a la ficha de producto, veremos que tanto la colocación de los elementos como las descripciones y colores permiten identificar las distintas áreas perfectamente.

La interacción, o en este caso, la compra online se ve muy favorecida por los múltiples elementos que componen la ficha de producto: títulos y H1, fotos e imágenes, precio, llamada a la acción y gastos de envío, textos de influencia, vídeo en la propia ficha y opciones de venta cruzada.

Al final, diseñar una web tiene que cumplir un único objetivo. VENDER. Todo lo que no implique esto, significa que es susceptible de mejora.

Malos ejemplos de usabilidad.

http://www.web_4_all.republika.pl/
http://arngren.net/
http://www.web_4_all.republika.pl/

4. Herramientas de edición web.

No hay modo de obviarlo: el diseño efectivo de sitios web puede ser una tarea compleja, difícil y larga. Afortunadamente, las herramientas de software de diseño de sitios web pueden hacer el proceso mucho más fácil. Cuando se trata de diseño de sitios web, incluso los expertos utilizan software de diseño web, en lugar de diseñar desde cero. Es más fácil, uno se ahorra tiempo y lo típico es que así se logre un mejor resultado.

En todo caso, hay una variedad de herramientas disponibles, que han sido diseñadas para adaptarse a varios niveles diferentes niveles de habilidad. Echemos un vistazo a algunas de las mejores herramientas de software para el diseño de sitios web.

4.1. Adobe Dreamweaver

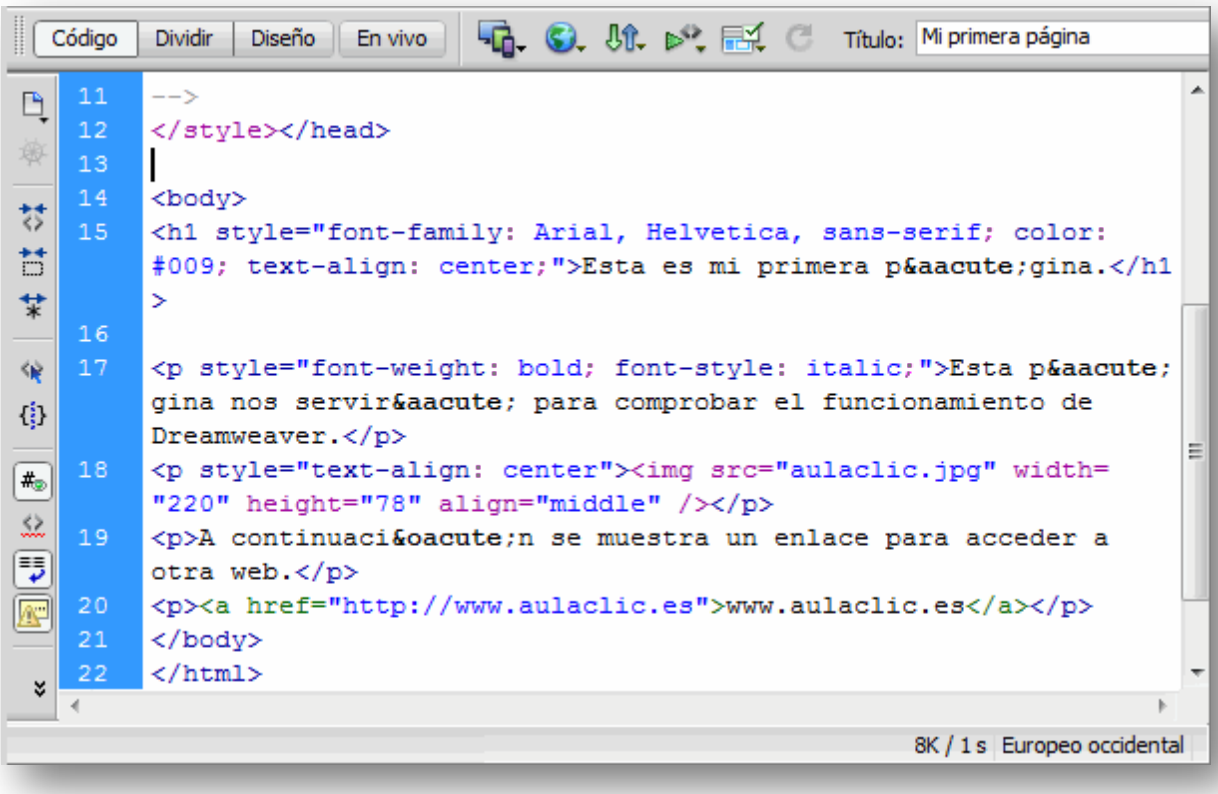
Dreamweaver tiende a ser el software de diseño de sitios web elegido por muchos diseñadores con experiencia moderada y también por expertos. Definitivamente, es mucho más que un editor de HTML, Dreamweaver logra integrar completamente el diseño visual con las herramientas de codificación. También es compatible con una variedad de diferentes lenguajes de scripting, incluyendo PHP, ASP y CSS.

Dreamweaver proporciona a los usuarios un alto nivel de control a lo largo de todo el proceso de diseño. El nivel de personalización que ofrece el software es prácticamente inigualable. Dreamweaver te permite crear funciones interactivas avanzadas, como menús desplegables, imágenes de sustitución, diseños de la red de fluidos, aplicaciones sencillas para dispositivos móviles y paneles plegables.

Gracias a la avanzada capacidad de WYSIWYG, la creación de un sitio web sencillo con Dreamweaver es, en realidad, bastante fácil. Dreamwaver desarrollará código HTML por ti; también otros programas de software WYSIWYG lo harán por ti, por lo que no tienes que ser un genio de la codificación para crear un sitio web. El Design View de Dreamweaver te proporcionará el editor de WYSIWYG más avanzado disponible en el mercado hoy.

Debido a que Dreamweaver ofrece edición híbrida, si tienes conocimientos de codificación, puedes trabajar en un entorno de doble panel y así aprovechar la

codificación de WYSIWYG y la codificación manual al mismo tiempo. Y si ya eres un experto en codificación, puedes trabajar exclusivamente en código, sólo haciendo el cambio a modo visual para ver el producto final. Para terminar, Dreamweaver es compatible con docenas de plug-ins, por lo que el diseño web resulta aún más fácil.



Manual oficial en castellano dreamweaver cc

Curso de dreamweaver cs6 aulaclic

4.2. Microsoft Expressions Web

Igual que Dreamweaver, el 'Microsoft Expressions Web program' tiene funciones de edición híbrida en una configuración de doble panel, lo que permite a los usuarios trabajar en WYSIWYG y con el código manual al mismo tiempo. 'Microsoft Web Expressions' viene con plantillas web integradas que hacen el diseño web más fácil. Pero esto no significa necesariamente que el programa es

para principiantes, ya que no existen herramientas de dibujo, ni colocación a través de la herramienta de "arrastrar y soltar". Es el heredero de Microsoft FrontPage 2003.

Microsoft Expressions Web es también notable por sus funciones de SEO, cosa que resulta de gran valor cuando se trata de crear un sitio web efectivo. El programa ofrece consejos e ideas sobre cómo optimizar tu sitio para obtener mejor rastreo y mejor posición en el ranking de los motores de búsqueda.

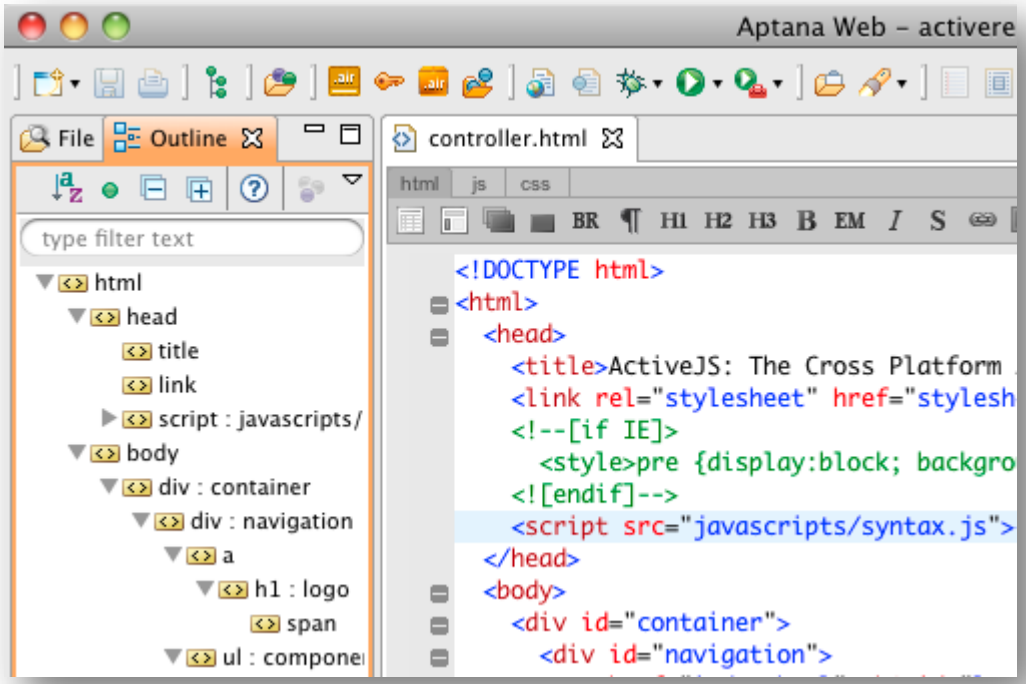
4.3. Editores web de software libre

- **Aptana Studio**

Aptana Studio es un entorno de desarrollo integrado de software libre basado en eclipse y desarrollado por Aptana, Inc., que puede funcionar bajo Windows, Mac y Linux y provee soporte para lenguajes como: Php, Python, Ruby, CSS, Ajax, HTML y Adobe AIR. Tiene la posibilidad de incluir complementos para nuevos lenguajes y funcionalidades.

Características:

- Asistente de código para HTML y Javascript.
- Librerías ajax (jQuery, prototype, scriptaculous, Ext JS, dojo, YUI y Spry entre otras).
- Conexión vía FTP, SFTP, FTPS y Aptana Cloud.
- Herramientas para trabajo con base de datos.
- Marcado de sintaxis mediante colores.
- Compatible con extensiones para Eclipse (existen más de 1000).



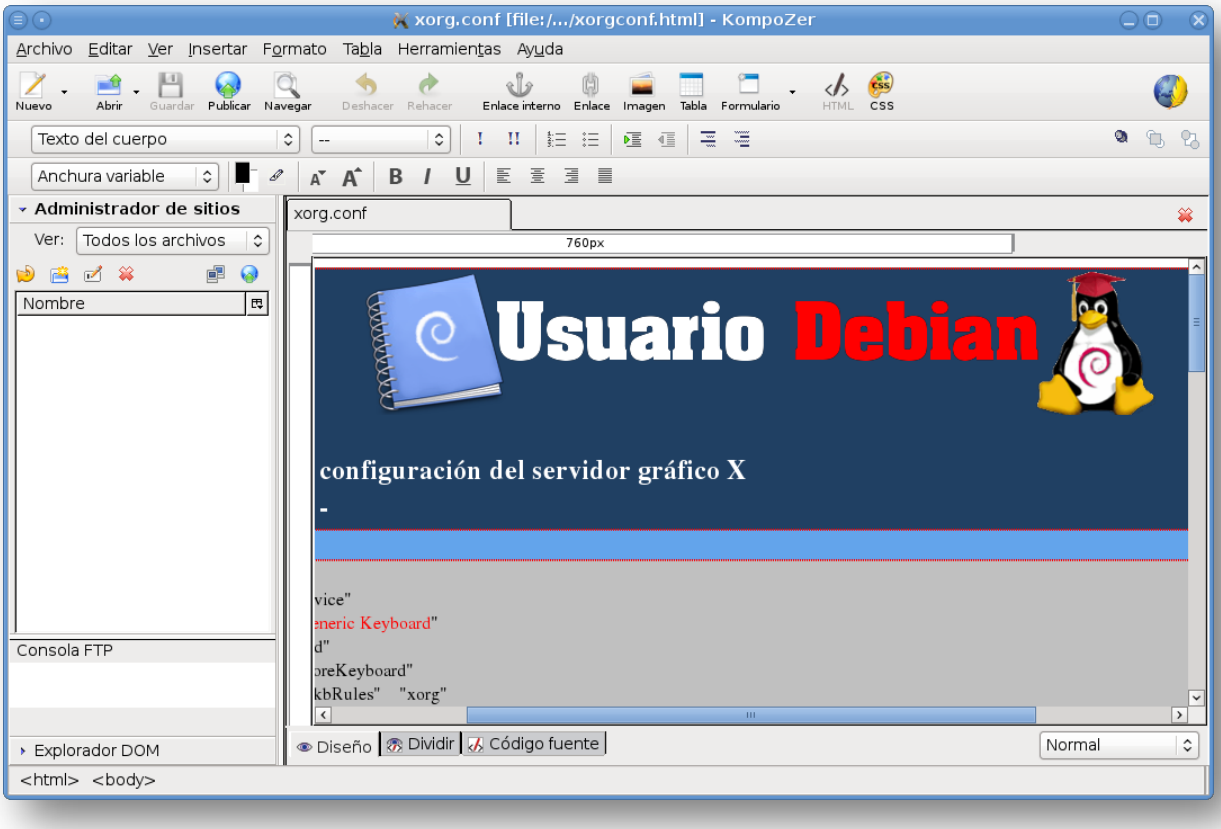
Pagina oficial

- **Kompozer**

Es un completo sistema de Web que combina archivos Web manejables y de fácil uso del editor de páginas WYSIWYG.

Hay binarios disponibles para GNU/Linux, Windows, MacOSX y OS/2.

Es considerado una de las mejores alternativas libres a Adobe CS6, y comparado favorablemente con Adobe Dreamweaver.



- Web oficial de Kompozer
- Manual en Español

5. Glosarios y diccionarios HTML5

https://developer.mozilla.org/es/docs/HTML/HTML5/HTML5_lista_elementos

http://www.imaginanet.com/diccionario-html5.html?

http://www.desarrolloweb.com/manuales/manual-lenguaje-html5.html