Unidad Didáctica

HTML 5

Índice de contenido

1. Introducción 1
2. Estructura global 2
1.2. Tipo del documento 2
1.3. Etiqueta html 2
2. Metadatos 3
2.1. <head> 3
2.2. <tittle> 4
2.3. <style> 4
2.4. <base> 6
2.5. <link> 7
2.6. <meta> 8
2.7. <script> 8
2.8. <noscript> 8
3. Nuevas etiquetas semánticas 9
4. Formularios 14
4.1. Nuevos controles de tipo input: 14
4.2. Nuevos atributos: 15
Nuevas etiquetas de formulario: 16
4. Nuevos elementos multimedia 17
4.1 Video 17
4.2 Audio 19

1. Introducción

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 que es extremadamente significativo.

2. Estructura global

Los documentos HTML se encuentran estrictamente organizados. Cada parte del documento está diferenciada, declarada y determinada por etiquetas específicas. En esta parte del capítulo vamos a ver los nuevos elementos semánticos incorporados en HTML5.

1.2. Tipo del documento

<!DOCTYPE>

En primer lugar necesitamos indicar el tipo de documento que estamos creando.

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.

En XHTML y HTML 4.01 el <!DOCTYPE> hace referencia a un DTD ya que son lenguajes basados en SGML. El DTD especifica las reglas del documento, así los navegadores visualizan correctamente el contenido.

HTML 5 no se basa en SGML y por lo tanto no requiere una referencia a DTD, pero el <!DOCTYPE> sirve a los navegadores para conocer el tipo de documento a mostrar.

1.3. Etiqueta html

Sea:

<!DOCTYPE>

<html lang=”es”>

</html>

Este atributo define el idioma humano del contenido del documento que estamos creando, en este caso es por español.

Atributo	***Soportado en HTML 5***	***Valor***	***Descripción***
manifest	Nuevo!	URL	Especifica la dirección del caché del documento para navegación offline
xmlns	NO		

2. Metadatos

2.1. <head>

Es un contenedor de elementos de cabecera.

La etiqueta no cambió desde versiones anteriores y su propósito sigue siendo exactamente el mismo. 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 iconos, el resto de la información incorporada en el documento entre estas etiquetas es invisible para el usuario.

Elementos que se pueden usar dentro de <head>

- <title> (requerido)
- <style>
- <base>
- <link>
- <meta>
- <script>
- <noscript>

El atributo profile no está soportado en HTML5.

2.2. <tittle>

- Define un título en la barra del navegador.
- Proporciona un título a la página cuendo se añade a favoritos.
- visualiza un titulo para la página como resultado de motores de búsqueda.

Sólo existe un título por documento. Si se omite <tittle> el documento no valida.

Diferencias con HTML 4.01 y XHTML: ninguna.

2.3. <style>

Ejemplo:

```
<html>
 <head>
   <style type="text/css">
        h1 {color:red;}
        p {color:blue;}
  </style>
 </head>

<body>
          <h1>A heading</h1>
          <p>A paragraph.</p>
</body>
</html>
```

La etiqueta <style> se utiliza para incluir bloques de código CSS. Incrustar CSS.

Cada documento HTML puede contener tantas etiquetas <style> como se especifiquen.

Recuerda que: para enlazar archivos CSS externos se usa <link>

Atributo	Valor	Descripción
media	**all** por defecto **aural** sintoni zador de voz **braille** dispositivos braile **handheld** dispositivos de bolsillo **projection** proyectores **print** medio paginado **screen** monitores **tty** teletipos **tv** Television	Especifica el medio para el que está optimizado el estilo
Scoped (Nuevo en HTML 5)		Permite definir estilos a una sección específica del documento.
type	text/css	Indica que es un estilo.

Ejemplo de media:

```
<style type="text/css" media="print">
        h1 {color:#000000;}
        p {color:#000000;}
        body {background-color:#FFFFFF;}
    </style>
```

Ejemplo de scoped:

```
<!DOCTYPE html>
<html>
    <head>
        <style type="text/css">
            h1 {color:green;}
            p {color:black;}
        </style>
    </head>
<body>
```

```
        <div>
                <style type="text/css" scoped>
                        h1 {color:red;}
                        p {color:blue;}
                </style>
          <h1>This heading should be red</h1>
           <p>This paragraph should be blue.</p>
        </div>

<h1>This heading should be green</h1>
<p>This paragraph should be black.</p>
<p><b>Note:</b> The scoped attribute is currently supported only in
Firefox.</p>

</body>
</html>
```

2.4. <base>

Especifica una URL por defecto para los enlaces de la página y dónde y cómo se muestra.

Ejemplo:

```
<head>
   <base href="http://www.w3schools.com/images/" target="_blank">
</head>

<body>
  <img src="stickman.gif" width="24" height="39" alt="Stickman">
   <a href="http://www.w3schools.com">W3Schools</a>
</body>
```

Las imágenes se buscan en **http://www.w3schools.com/images** y los enlaces se abren en una nueva ventana.

Solo puede existir una etiqueta <base> y además en <head>

Diferencias con HTML 4.01: ninguna.

Diferencias con XHTML : en HTML 5.0 no necesita etiqueta fin.

Atributos:

Atributo	Valor	Descripción
href	URL	Especifica la URL por defecto que usa la página
target	**_blank:** abre el enlace en una nueva ventana o pestaña. **_parent:** Por defecto. Abre el link en el mismo sitio donde se pulsó o hizo clic. **_self,_top,*framename: (uso con iframes)***	Especifica la presentación (ventana y cómo) para los enlaces y formularios de la página

HTML 5 no soporta marcos, pero sí iframes.

2.5. <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 .

Es un elemento vacío, sólo contiene atributos. Aparece sólo en la sección de cabecera, pero puede aparecer varias veces.

- Diferencias con HTML 4.01: Algunos atributos en 4.01 no son soportados en HTML 5. El atributo sizes en HTML 5.
- Diferencias con XHTML : en HTML 5.0 no necesita etiqueta fin.

En HTML5 ya no se necesita el atributo charset que especificaba el sistema de codificación de carateres.

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 (en el próximo capítulo estudiaremos cómo utilizar estilos CSS).

Ejemplo:

```
<!DOCTYPE html>
<html lang="es">
 <head>
    <meta charset="utf-8">
    <meta name="description" content="Ejemplo de HTML5">
    < meta name="keywords" content="HTML5, CSS3, JavaScript">
   <title>Este texto es el título del documento</title>
   <link rel=”stylesheet” href=”misestilos.css”>
 </head>
<body>
</body>
</html>
```

2.6. <meta>

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 (UTF-8) y agregar otras etiquetas <meta> como description o keywords para definir otros aspectos de la página web, como es mostrado en el ejemplo anterior.

2.7. <script>

Sin variaciones con respecto a XHTML y HTML 4.01

2.8. <noscript>

Sin variaciones con respecto a XHTML. y HTML 4.01

3. Nuevas etiquetas semánticas

La razón para crear nuevas etiquetas estructurales es dividir las páginas Web en partes lógicas que describan el tipo de contenido que incluyen. Conceptualmente, se piensa en la página Web como un documento. Los documentos tienen encabezados, pies de página, capítulos y otras convenciones diferentes que dividen el documento en partes lógicas.

```
<header>
<hgroup>
<nav>
<article>
<section>
<aside>
<time>
<mark>
<footer>
```

<header>

Es el equivalente a la **cabecera** de la página web. Contiene el título o nombre de la empresa/titular de la página, logo e información relacionada. La norma define a la cabecera como un grupo de ayudas a la navegación.

No es una etiqueta obligatoria, aunque debemos usarla si realmente nuestras páginas tienen algún bloque de código que se pueda entender como "cabecera".

Además de los elementos citados, un header podría contener un menú de ayuda a la navegación, una tabla de contenidos, un formulario de búsqueda, etc. Su contenido no es tan extricto.

Al tratarse de etiquetas semánticas, nos da igual la posición que ocupen dentro de nuestra página.

<hgroup>

Gracias a la etiqueta <hgroup> podemos encerrar etiquetas <h1>, <h2>, etc, para que el navegador simplemente no las tenga en cuenta, para que no las contabilice como creadoras de nuevas secciones sino solamente como titulares.

Ejemplo:

```
<section>
  <hgroup>
    <h1>Primera sección de nivel 1</h1>
    <h2>Breve descripción de la sección de nivel 1</h2>
  </hgroup>
<p>En esta nueva sección hablaremos de gggg...</p>
</section>
```

<nav>

Formada por enlaces a las zonas principales de la web, o por un menú de navegación, desplegable de enlaces, etc.

No todos los enlaces de una página web han de pertenecer obligatoriamente a un nav. Los enlaces de políticas, de datos de contacto, mapas del sitio, copyrights, enlaces a secciones secundarias etc, que aparecen normalmente en un pie de página no es necesario que los rodeemos con la etiqueta <nav>, pues la etiqueta <footer> ya recoge esos aspectos. Ésta etiqueta es más bien para enlaces que favorezcan la navegación por la web, que nos dirijan a zonas importantes.

El porqué HTML5 añade esta etiqueta es sencillo. Cada vez es más habitual que nuestras páginas aparezcan no solo en los navegadores que usamos en el ordenador, sino también en dispositivos móviles, teléfonos, iPads, etc. Al usar esta etiqueta el agente de usuario (el navegador o como quiera llamarse según el dispositivo usado) podría dar la opción al usuario de esconder el menú de navegación para no molestar, para tener más espacio disponible para ver el contenido importante. También podría dar la opción de hacerlo aparecer cuando el usuario quiera cambiar de sección. Esto solo sería posible si se sabe dónde empieza y dónde termina ese menú de navegación y eso es precisamente lo que conseguimos con la etiqueta nav.

<section>

Este gran apartado puede agrupar diferentes subapartados (de tipo '**article**') de diferentes temas, o bien puede definir un gran apartado de contenido unitario.

La etiqueta <section> y su correspondiente etiqueta de cierre </section> se utilizan para encerrar el código correspondiente a una sección genérica dentro de un documento o aplicación. Normalmente, un bloque de texto al que perfectamente le podríamos colocar un título o encabezado. Además, todo el contenido que engloba ha de guardar cierta relación entre sí.

Recuerda que no se debe usar para englobar un bloque cualquiera de código simplemente porque necesitas encerrarlo dentro de algo para aplicarle así estilos CSS o algún Script. Si el único motivo para encerrar un bloque de código es el de poder aplicarle lo anterior, no tiene sentido aplicarle una de estas nuevas etiquetas. En su lugar se puede y debe usar un DIV de toda la vida.

Otra regla que podemos usar para saber si un bloque de código o un elemento de una página web debe encerrarse entre esta etiqueta es el preguntarse.... si la web fuera un libro.... estaría ese elemento en el indice inicial del mismo? Si la respuesta es SI, es buen candidato para section.

Dentro de una página web por tanto, pueden existir varios section, cómo no en un libro suelen existir varias secciones en el índice. Es más, dentro de un section pueden haber otros section secundarios y así sucesivamente.

<article>

Es una pieza independiente de contenido, que puede estar contenida (o no) dentro de un apartado de tipo '**section**'.

También con la etiqueta <article> podemos separar cada uno de esos elementos independientes que pudieran componer ese section concreto. Un article podría considerarse por tanto como cada uno de los elementos en que podemos dividir un section. Al igual que el resto de estas nuevas etiquetas, posee etiqueta de cierre, <article>.

En una página web pueden existir varios articles, dentro de un section o incluso independientes del mismo. Pueden además contener en su interior títulos con h1 y párrafos, además de otros articles en su interior.

Un ejemplo real de article podría ser un mensaje de un foro, el artículo de una revista o periódico, un comentario de un usuario a una entrada de blog o incluso una entrada en un blog. La estructura podría ser la siguiente:

```
<article>
  <h1>La etiqueta Article</h1>
  <p>La etiqueta <b>article</b> suele usarse para fragmentos independientes de contenido...</p>
  ..
</article>
```

Deciamos que un article podría ser tan grande como para poder contener incluso header, footer, etc. Ejemplo:

```
<article>
 <header>
   <h1>La etiqueta Article</h1>
```

```
 </header>
  <p>La etiqueta <b>Article</b> suele usarse para fragmento independientes de contenido...</p>
  ...
  <footer>
     <p><small>Contenido publicado por Juanito</small></p>
 </footer>
</article>
```

Las diferencias entre article y section, cuándo usar una etiqueta article y cuándo utilizar la etiqueta section, no termina de quedar demasiado clara tras leer la norma que rige el Html5. Como regla general podríamos establecer lo siguiente:

Si el fragmento de código posee significado por si mismo, si en caso de escribirlo en un papel separado del resto de la web el fragmento continua teniendo su sentido, podemos usar un <article>.

Si no tiene tanto sentido, pero tiene relación con lo comentado en el resto de esa página, podemos usar un <section>.

Si el algo que no tiene mucho que ver con el tema de esa página en concreto ya dijimos que se trataría de un <aside>.

Si no cumple con ninguna de las condiciones anteriores, pero necesitamos encerrarlo entre dos etiquetas para poder aplicarle estilos o scripts, entonces lo que debemos usar es un Div.

<aside>

Define un bloque de contenido (tangencial) relacionado con el contenido principal, pero que no es esencial para la compresión del mismo.

Con las etiquetas <aside> y </aside>, se nos invita a rodear todo aquel contenido que no es directamente contenido principal del que estamos hablando o del que estamos tratando en esa página en concreto.

Lo usaremos por tanto para todos aquellos elementos secundarios, como podrian ser los bloques publicitarios, enlaces externos, citas, un calendario de eventos, etc, siempre claro que no encontremos otra etiqueta más acorde de entre las ya comentadas.

<footer>

Equivale al pie de página de un apartado concreto ('**section**' o '**article**') o de la página web en general. Contendrán por tanto enlaces a otras webs relacionadas, al mapa de la web, a una página de comentarios sobre el copyright, una política del portal, quizás algo de publicidad, etc.No tiene que estar en la parte más baja de una página web para poder ser un footer. Tened en cuenta que todas etiquetas intentan informar a los navegadores sobre lo que pueden contener esos bloques, no más.

<time>

El elemento <time> nos permite declarar un texto comprensible para humanos y navegadores que representa fecha y hora .

<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.

Por defecto CSS asume que este tipo de elementos tienen la propiedad display:inline, así que si queremos poner alturas o anchuras, tendremos que cambiarlo a display:block;

```
header, footer, aside, nav, article, section {
     display: block;
}
```

<figure>

Es un nuevo tipo de bloque de HTML5 con contenido semántico propio y está orientado a contener imágenes. De esta manera el navegador y cualquier robot que acceda a nuestra página sepa el significado del contenido que contiene este tipo de bloque (que estará destinado a imágenes).

<**figcaption**>. Nos permite añadir un texto como pie de foto a una imagen.

<**figure**>

```
        <img src="img/imagen.png">
        <figcaption>
                Este texto es el <b>figcaption</b> de esta imagen.
        </figcaption>
</figure>
```

4. Formularios

4.1. Nuevos controles de tipo input:

A continuación mencionamos cada uno de los doce nuevos INPUT que están presentes en la quinta especificación del lenguaje HTML, con una breve explicación de cada uno.

INPUT tel:

Este tipo de input viene predispuesto con un formato para escribir números telefónicos. En realidad no hace ninguna validación, pero sí se puede implementar una con la nueva API de validación de JavaScript.

INPUT number pre formateado:

Sirve para escribir solo números. En algunos navegadores, cuando se ejecuta el evento onSubmit no se hace el envío en caso que el campo number esté lleno de caracteres que no sean numéricos.

INPUT search:

Además de proporcionar un campo de entrada, se le agrega un icono de búsqueda para distinguirlo de un campo de navegación.

INPUT color:

Este input nos brinda una paleta de colores donde el usuario puede escoger un color de forma dinámica. Es lo que llamamos un colorpicker, con la particularidad que nos lo ofrece el propio navegador.

INPUT range:

Proporciona un control que se desliza, cambiando automáticamente el valor del campo.

INPUT URL:

Este tipo de entrada viene con un formato para URL absoluta.

INPUT email:

Tiene la capacidad de aceptar únicamente direcciones de correo electrónico. Además, se pueden enviar varios email separados por comas, si tiene especificado el atributo multiple.

INPUT date:

Para introducir una fecha que no haga parte de del conjunto horario.

INPUT month:

Para introducir meses del año.

INPUT week:

Ofrece una utilidad para escribir y captar información de semanas.

INPUT time:

Obtiene información con horas, minutos y segundos.

Tipo datetime-local:

Recibe la hora local del dispositivo.

4.2. Nuevos atributos:

- `Autofocus` : al cargarse la página activa o toma el foco el control asociado. Valor el mismo "autofocus". Es correcto: autofocus="autofocus" o autofocus
- `autocomplete:` valores:"on", "off". Opión de activar o no el autocompletar.
- `Formnovalidate`: especifica que los elementos del formulario no serán validados antes de ser enviados al servidor. Valor :"formnovalidate"
- `formtarget`: especifica cómo se va a mostrar la respuesta del servidor cuando los datos son enviados. Para type="submit" y type="image"
  Valores posibles:
  _blank
  _self
  _parent
  _top
  framename
- `min` : especifica un valor mínimo para un control. Valor: un nº o una fecha.
- `Max` : especifica un valor máximo para un control. Valor: un nº o una fecha.
- `height:` especifica el alto en pixeles de un control de tipo image.
- `Width`: especifica el ancho en pixeles de un control de tipo image.
- `pattern` : Expresión regular o patrón por el que validar el contenido de un control
- `placeholder` : descripción corta del control.
- `Required` : control obligatorio. Valor: "required"
- `step` : especifica un número que será el incremento o decremento del valor de un control

- `multiple`: especifica que el control admite varios valores. Valor:"multipple"

El navegador Opera es el único que casi proporciona un soporte completo para los nuevos controles de formularios y atributos. El W3C Markup Validator nos advierte sobre esta situación: The date input type is so far supported properly only by Opera. Please be sure to test your page in Opera.

Nuevas etiquetas de formulario:

En el caso del HTML5 y los formularios tenemos que destacar que no solamente se han creado nuevas etiquetas, sino que se ha añadido soporte a las existentes anteriormente. De momento estas son las nuevas etiquetas que nos ofrece.

Etiqueta METER:

Esta nueva etiqueta se usa para representar escalas de medidas conocidas, de ahí su nombre: Meter, el cual tiene relación directa con medida. Se puede emplear para representar escalas de medición conocidas como longitud, masa, peso, entre otras.

Etiqueta PROGRESS:

Se usa para crear barras de progreso. Podemos emplearlas o usarlas en procesos de larga duración, como la descarga de archivos, para indicar a cualquier usuario de nuestra aplicación el progreso de la operación que se está realizando.

Elemento DATALIST:

Es una extensión que sirve para crear campos de autocompletado. Sirve para especificar una lista de datos u opciones que se pueden utilizar para sugerir el autocompletado de elementos como INPUT. Por tanto, para utilizar un DATALIST, tenemos que combinar ese elemento con otros elementos de formulario al que le colocamos atributos nuevos para asociar el DATALIST para hacer el autocompletado.

Etiqueta KEYGEN:

Usada para generar pares de claves, clave pública y privada. Al enviar el formulario al servidor por cualquiera de los métodos HTTP, en el cliente se guarda una clave privada la clave pública se empaqueta y se envía al servidor.

Etiqueta o elemento OUTPUT:
Muestra el resultado de un cálculo matemático, su uso básico puede ser tan básico como el de mostrar una simple suma de dos números.

4. Nuevos elementos multimedia

- **AUDIO**: Para insertar sonido dentro de una web.
- **VIDEO**: Para insertar clips de vídeo.
- **EMBED**: Para embeber contenido externo de otro tipo, como el traído de diversos plugins que se comercializan actualmente o se comercializarán en el futuro.
- **SOURCE**: Permite especificar varias fuentes diferentes cuando se insertan elementos en AUDIO y VIDEO.

4.1 Video

HTML5 finalmente introdujo un elemento para insertar y reproducir vídeo en un documento HTML. El elemento <video> usa etiquetas de apertura y cierre y sólo unos pocos parámetros para lograr su función. La sintaxis es extremadamente sencilla y sólo el atributo src o source es obligatorio:

```
<!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://minkbooks.com/content/trailer.mp4">
              <source src="http://minkbooks.com/content/trailer.ogg">
          </video>
      </section>
 </body>
</html>
```

Posible problema con los navegadores: 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 navegadores como Safari e Internet Explorer) se encuentra bajo licencia comercial).

Los formatos OGG y MP4 son contenedores de video y 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.

Atributos para <video>

- **src** : 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.

- **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.

- **controls** Este atributo muestra controles de vídeo 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.

- **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://minkbooks.com/content/poster.jpg">
                  <source src="http://minkbooks.com/content/trailer.mp4">
                  <source src="http://minkbooks.com/content/trailer.ogg">
             </video>
       </section>
    </body>
</html>
```

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.

4.2 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.

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://minkbooks.com/content/beach.mp3">
				<source src="http://minkbooks.com/content/beach.ogg">
			</audio>
		</section>
	</body>
</html>
```