Diseño Web Responsive

Unidad 6: Nuevos elementos multimediales

Contacto: consultas@elearning-total.com
Web: www.elearning-total.com

Indice

Unidad 6: Nuevos elementos multimediales

Objetivos

Que el alumno logre:

- Conocer los nuevos elementos semánticos incorporados en la versión HTML5.

Elementos eliminados en HTML5

Algunos de los objetivos de HTML5 son evitar el código innecesario y facilitar la tarea a los desarrolladores de navegadores, a los fabricantes de editores de html y fundamentalmente a los desarrolladores de los sitios web.

Para lograr este fin, se eliminaron algunos elementos que se consideraban innecesarios.

Una recomendación importante para tener en cuenta es que todo lo que pueda ser resuelto empleando estilos, debe ser realizado mediante CSS. De esta forma, logramos la finalidad de abstraer la estructura del documento y los datos de la representación que nos proveen los estilos.

Etiquetas de representación:

- <basefont>
- <big>
-
- <font>
- <s>
- <strike>
- <tt>
- <u>

También se eliminan las etiquetas de diagramación mediante marcos, ya que afectan la usabilidad, la accesibilidad y la navegabilidad del sitio:

- <frameset>
- <frame>
- <noframes>

Además se eliminan:

- <acronym>
  - en su lugar se utiliza <abbr>
- <applet>
  - se utiliza <object>

- <dir>
    - su función se reemplaza con la listas (ul por ejemplo)

En relación a los atributos, dejan de ser soportados:

- <html> deja de soportar el atributo **versión**
- <img> deja de soportar los atributos **longdesc** y **name**
- <link> deja de soportar el atributo **target**
- <meta> deja de soportar el atributo **scheme**
- <object> deja de soportar los atributos **archive**, **classid**, **codebase**, **codetype**, **declare** y **standby**.
- <a> deja de soportar los atributos **rev** y **charset**.

Algunas recomendaciones generales en cuanto a etiquetas y atributos en HTML5:

- No se recomienda el uso del atributo **aling** (como atributo de html), ya que su función puede ser afectada mediante el uso de CSS.
- Para la etiqueta body, no es recomendable usar los atributos **alink**, **link**, **text**, **vlink**, **background** y **bgcolor**.
- Para elementos de tabla, no se recomienda emplear los atributos **bgcolor**, **border**, **cellpadding**, **valign**, **height**, **width**, **nowrap**, **frame** y **char**.
- En el caso de listas y menús, no se recomienda emplear los atributos **type** y **compact**.
- Tampoco es recomendable utilizar **hspace** y **vspace** en las etiquetas <img> y <object>

Elementos reinterpretados

Con el fin de aportar una nueva estructura de página con mayor valor semántico, alguno de los elementos ya utilizados en HTML se resignificaron:

- **<dl>**
- **<cite>**
- **<address>**
- ****
- ****
- **<small>**
- **<i>**
- **<b>**
- **
**
- **<hr>**

Nuevos elementos

Podemos encontrar una amplia variedad de nuevas etiquetas desarrolladas para HTML 5 (y algunas otras que aún están en desarrollo). Algunas de ellas son:

- <articule>
- <aside>
- <audio>
- <canvas>
- <command>
- <details>
- <dialog>
- <embed>
- <figure>
- <footer>
- <header>
- <hgroup>
- <keygen>
- <mark>
- <meter>
- <nav>
- <output>
- <progress>
- <rp>
- <rt>
- <ruby>
- <section>
- <source>
- <time>
- <video>

HTML 5 introduce nuevos elementos "inline" muy útiles para aumentar nuestro existente arsenal de "span, strong, em, abbr, etc". A partir de ahora a estos elementos ya no se les llamará "inline", sino “text-level semantics.”

Mark

Cuando revisamos un listado de una búsqueda en una web, usualmente vemos el término por el que hemos buscado iluminado o resaltado dentro de cada resultado. Por lo general se marca cada instancia del término de búsqueda con un elemento **span**, pero **span** desde un punto de vista semántico, es algo impreciso, ya que realmente sirve para poco más que aplicarle una clase específica a un elemento dentro de un estilo ya definido.

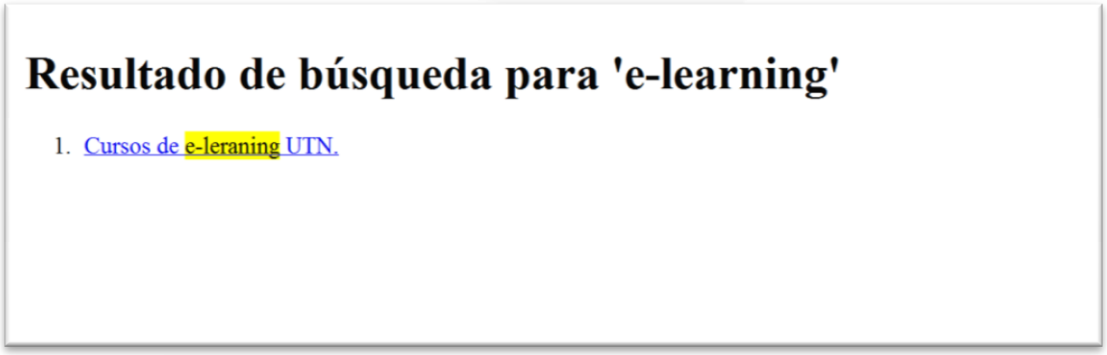
Se podría utilizar **em** o **strong** pero semánticamente no sería correcto ya que no querrías otorgarle una "importancia" al término de búsqueda, simplemente queremos que de alguna manera quede resaltado.

Para eso utilizamos el elemento **mark**:

```
<h1>Resultado de búsqueda para 'e-learning'</h1>
<ol>
<li>
<a href="http://www.sceu.frba.utn.edu.ar/">Cursos de  <mark>e-leraning</mark> UTN.</a>
</li>
</ol>
```

El elemento **mark** no lleva implícito ninguna importancia para el contenido salvo el mostrarlo como algo de interés en el contexto en que esté.

Según la especificación, **mark** denota fragmento de texto de un documento marcado o iluminado con fines de referencia debido a su importancia en otro contexto.



Meter

El elemento **meter** puede ser utilizado para marcar medidas, siempre que esas medidas sean parte de una escala con valores mínimos y máximos.

La función de la nueva etiqueta **meter** de la especificación HTML5 es la de **indicar una medida escalar dentro de un rango conocido**, o un valor fraccional.

Es importante aclarar que **meter** es usado para representar un rango, o sea que no sería correcto usarlo para mostrar nada más que un simple número, a no ser que ese número pueda ser acotado por un mínimo y un máximo.

```
<meter min="0" max="120" value="40">
    <span>1/3</span>
</meter>
```

Aquí también tenemos la posibilidad de usar contenido alternativo para aquellos navegadores que todavía no soportan dicho elemento.

Visualización de la etiqueta meter en Google Chrome

Progress

Mientras el elemento **meter** es muy bueno para describir algo que ya ha sido medido. El elemento **progress** te permite marcar un valor que está en proceso de cambio.

Este elemento puede ser usado (en conjunto con Javascript) para mostrar el progreso de una tarea o de un proceso que esté ocurriendo en la página web, como por ejemplo un archivo o recurso que se está subiendo o descargando. Su estructura es la siguiente:

```
Su perfil está <progress>60%</progress> completado.
```

Otra vez se pueden utilizar los atributos **min**, **max**, y **value**:

```
<progress min="0" max="100" value="60"></progress>
```

El elemento **progress** es más útil cuando se utiliza combinado con DOM Scripting. Se puede utilizar JavaScript para actualizaciones dinámicas del valor, permitiendo al navegador comunicar el cambio al usuario. Muy útil para cargar archivos con Ajax.

Un punto a notar es que **ambos elementos (meter y progress) son modificables en tamaño (tanto ancho como alto) mediante CSS**.

Imágenes

La directiva **<figure>** se trata de una nueva incorporación de HTML 5.

Es un elemento que engloba a la imagen y a la información adicional (título, descripción, referencias) que pueda acompañarla.

Se trata de un bloque como **<section>** o **<article>** que permite darle un tratamiento especial (incluso sacándolo del contenido principal y llevarlo, por ejemplo a una barra lateral).

<figure> se utiliza en conjunto con el elemento <figcaption>

Hay que tener en cuenta que una figura en HTML puede contener cualquier contenido que ayude a ilustrar el contenido principal, lo que incluye fragmentos de código, animaciones, objetos multimedia, o textos adicionales. No está restringido únicamente a imágenes.

El elemento **<figcaption>** representa un título o leyenda a una figura establecida por **<figure>.**

<figure> con <img>

```
<figure>
  <img src = "orang-utan.jpg"  alt = "Orangután bebe colgando de una cuerda"
>
</ figure>
```

<figure> con <img> y <figcaption>

El figcaption funciona como una leyenda de la imagen.

Un macaco descarada, Baja Kintaganban Río, Borneo. Original por Richard Clark

```
<figure>
  <img src = "macaque.jpg"  alt = "Macaco en los árboles" >
  <figcaption> Baja Kintaganban Río, Borneo. Original por <a href =
"http://www.flickr.com/photos/rclark/" > Richard Clark </ a> </ figcaption>
</ figure>
```

<figure> con varias imágenes

```
<figure>
  <img src = "/ kookaburra.jpg"  alt = "Kooaburra" >
  <img src = "/ pelican.jpg"  alt = "Pelican estaba en la playa" >
  <img src = "/ lorikeet.jpg"  alt = "Cheeky busca del arco iris Lorikeet" >
  <figcaption> Aves de Australia. De izquierda a derecha, Kookburra, Pelican y
Lorikeet Arco Iris. Originales de <a href =
"http://www.flickr.com/photos/rclark/" > Richard Clark </ a> </ figcaption>
</ figure>
```

Aves de Australia. De izquierda a derecha, Kookburra, Pelican y Lorikeet Arco Iris. Originales de Richard Clark

Multimedia en HTML 5

La nueva versión del estándar se preocupa sobre todo por la semántica de las páginas web. En el campo del multimedia, incorpora dos nuevas directivas que actúan como contenedores de video y audio: **<video>** y **<audio>**.

Con **<video>**, el navegador ya contaría con la capacidad de interpretar correctamente el contenido, sin requerir de los elementos externos.

<VIDEO>

Repasando la historia de Internet, podemos detenernos en el año 2004. No existía Youtube. Por lo mismo, si querías alojar y compartir en un navegador cualquier vídeo había que enfrentarse a los problemas de plataforma, de ancho de banda y de usabilidad para el usuario final.

Si nos encontrábamos con algún enlace que llevara a ver videos en la web seguramente empezarían a aparecer los logos de Real Player, de Windows Media Player o incluso de Quicktime.

Del lado del servidor los sysadmins tenían que luchar contra Real Media Server, Windows Media Server que era parte del IIS y otro montón de opciones, pero era complicado unificar. Adobe por su parte había logrado importantes avances para que desde archivos .swf pudieras incorporar videos, pero no había un solo canal.

En el año 2005, nace Youtube y muchos de esos problemas se solucionan (o al menos aparece una opción más simple de incorporar video a la web).

Paralelamente, Adobe demostró que su esfuerzo para crear un sistema para reproducir vídeo iba a ser valorado y aprovecharía el dominio que su plugin de flash tenía en todos los navegadores del planeta.

En abril del 2010 una carta de Steve Jobs (CEO de Apple) sobre sus pensamientos de Flash arranca uno de esos nuevos ciclos donde una plataforma disruptiva que cada día ganaba mercado le daba la espalda al famoso plugin de Adobe en sus dispositivos móviles.

Apple apoyaba el desarrollo de HTML5 porque todo se podía resolver en temas de vídeo con un tag apropiado para ese fin y aprobado por el estándar.

Hoy volvemos a sobre una batalla de grandes empresas por el codec luego de que están todos de acuerdo que el navegador lleva una opción amigable para incluir un elemento de video, porque este elemento es tan importante como las imágenes.

Y podemos usar perfectamente el ejemplo de las imágenes para entender lo que pasa con el video.

Todos los navegadores permiten que en un documento HTML incluyas una etiqueta <img> a la cual debemos indicarle un atributo con la ruta a una imagen. Y la imagen puede ser un .jpg, un .gif, un .bmp o un .png (entre otros formatos). Dependerá de la capacidad del navegador de cada usuario para reconocer todos los contenidos de esta imagen, interpretarla y mostrarla. Con el video pasa lo mismo.

En esta nueva versión de HTML, para incorporar un archivo de video solo debemos contar con una etiqueta destinada a este fin: **<video>**

Esta etiqueta se encarga de dar soporte a contenido audiovisual que se interpretaría directamente en el navegador. En este espacio, se puede incluir tanto un clip como un streaming.

Estos son sus principales atributos:

- **src**: permite especificar la URL del video.
- **audio**: se trata de un atributo que nos permite indicar el estado del audio del video correspondiente. Y, si está disponible la opción, se puede asignar el valor determinado **muted**.
- **autoplay**: indica que el video se ejecutara automáticamente tras su descarga.
- **loop**: indica que el video se volverá a ejecutar después de terminar.
- **height**, **width**: tamaño en pantalla (la altura y anchura respectivamente en pixels o bien en porcentaje) del video.
- **controls**: indica si hay controles embebido del video. La interpretación final de este atributo va a depender mucho de la implementación de los navegadores.
- **poster**: nos brinda la posibilidad de indicar la URL de una imagen que represente al video, que se mostrará cuando el video no está disponible o no sea accesible.
- **preload**: se utiliza para indicar que el video cargue junto con la página y que esté listo para iniciarse.

Es posible especificar varios formatos alternativos para un mismo video para asegurarse de que el navegador es capaz de interpretar alguno de ellos. Para ello no hay que incluir el atributo @src y usar un elemento nuevo: **<sourse>**.

Este elemento acepta varios atributos: @src, para indicar la dirección de cada fichero; @type contiene el tipo MIME del fichero; y @codec contiene información adicional sobre la forma de interpretarlo.

Ejemplo de video básico (sin elementos de control)

```
<video src="tu_video.ogv">Tu navegador no soporta HTML5 </video>
```

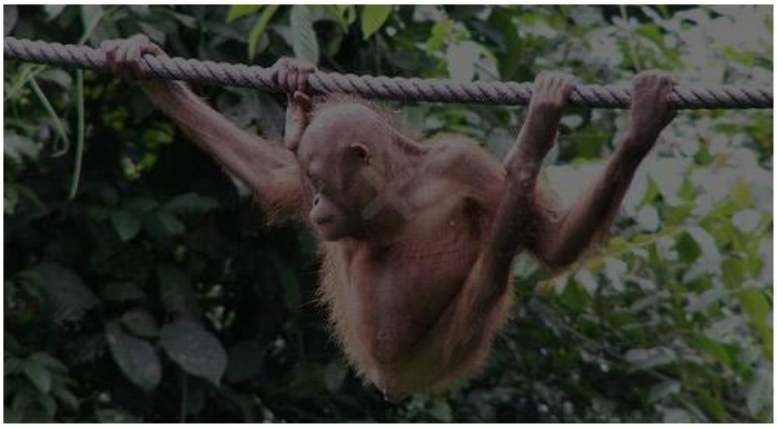
Ejemplo de video con controles

```
<video controls src="http://devfiles.myopera.com/articles/2642/sintel-trailer.ogv">Tu navegador no soporta HTML5 </video>
```

Ejemplo de video con atributo @poster:

```
<video controls src="tu_video.ogv" poster="orang-utan.jpg" >Tu navegador no soporta HTML5 </video>
```

Esta imagen se mostrará cuando el navegador no tenga soporte para la etiqueta o cuando no reconozca la ruta o el formato del video.



No todos los navegadores soportaran el video en formato .ovg para ello podemos agregar una variación a nuestro código. Esto puede solucionarse agregando un segundo **<source>** para que si nuestro formato .ogv no es aceptado se use otro formato.

```
<video autoplay controls>
<source src="tu_video.ogv" type="video/ogg" />
<source src="tu_video.mp4" type="video/mp4" />
<p>Tu navegador no soporta HTML5</p>
 </video>
```

Y para los navegadores que no soportan html5 agregaremos el video en flash con .flv.

```
<video autoplay controls>
<source src="tu_video.ogv" type="video/ogg" />
<source src="tu_video.mp4" type="video/mp4" />
<object width="160" height="90" data="video.flv">
      <param name="movie" value="tu_video.flv">
      <embed src="tu_video.flv" width="160" height="90">
</object>
<p>Tu navegador no soporta HTML5 ni Flash</p>
 </video>
```

Video y Compatibilidad de Navegadores

Se deben abordar dos cuestiones muy importantes cuando se trata el video web con HTML5 y la compatibilidad. Una de ellas es, la de que navegador funciona con que formato de video.

En este punto, ir más allá de lo que se puede probar, es un poco arriesgado. Sin embargo, podemos examinar cuatro tipos diferentes de contendores de archivos y codecs y utilizarlos.

El formato del contenedor 3GP está relacionado con MPEG-4 pero, en realidad, es un formato H.263 y su principal aplicación es para dispositivos móviles como iphone.

Podemos ver la matriz de compatibilidad de los principales navegadores en los que se han llevado a cabo las pruebas de video.

http://caniuse.com/#feat=video

<AUDIO>

De la misma forma, la ejecución de sonidos se canalizaría a través de esta directica en lugar de requerir la inclusión de elementos externos.

El objetivo de esta etiqueta es permitir la carga y ejecución de archivos de audio sin requerir un plug-in de Flash, Silverlignt o Java.

El comité de estandarización W3C deja abierto a cada empresa que desarrolla navegadores los formatos que quieran soportar (así tenemos que algunos soportan mp3, wav, ogg, au)

Un ejemplo de disponer el elemento audio dentro de una página sería:

```
<audio src="sonido.ogg" autoplay controls loop></audio>
```

Estos son sus principales atributos:

- **src:** es un atributo único con la dirección del fichero de audio.

- **autoplay**: indica que el video se ejecutara automáticamente tras su descarga.
- **loop**: indica que el audio se volverá a ejecutar tras terminar, y el número de veces que va a hacerse.
- **controls**: indica si hay controles embebido del audio. Como en el caso del video, la interpretación va a depender mucho de la implementación de los navegadores.
- **preload**: permite indicar si el audio se cargará con la página. Puede tomar los valores: **auto**, **metadata** y **none**.

Como no hay un formato de audio universalmente adoptado por todos los navegadores el elemento audio nos permite agregarle distintas fuentes (igual que el video):

```
<audio controls autoplay loop>
        <source src="sonido.ogg">
        <source src="sonido.mp3">
        <source src="sonido.wav">
        <source src="sonido.au">
Tu navegador no soporta esta etiqueta
</audio>
```

El elemento source indica a través de la propiedad src la ubicación del archivo de audio respectivo.

El orden que disponemos estas fuentes es importante. Primero el navegador busca la primera fuente y verifica que puede reproducir dicho archivo, en caso negativo pasa a la siguiente fuente. Como vemos si queremos que cualquier navegador reciba un audio podemos inclusive hacer el aplique de Flash respectivo por si el navegador no implementa el elemento **AUDIO** o no soporta el formato de archivo.

Por el momento no hay un formato con soporte para todos los navegadores, dependerá del sitio que implementemos para ver si tiene sentido duplicar nuestros archivos con distintos formatos de audio.

<CANVAS>

Con el uso de la directiva <CANVAS> se pueden definir áreas de presentación independientes sin requerir argucias de programación en Javascript.

Cada “canvas” o “lienzo” definido es un área bidimensional en el que se puede dibujar figuras o reproducir imágenes. Pueden definirse varios canvas en una misma página y hacer que se alternen para estar en primer plano lo que permite obtener efectos visuales muy atractivos.

Con HTML pueden definirse varios canvas dentro de un documento con parámetros que permiten identificarlos (@name) y asignarles unas dimensiones (@width y @height).

A esta etiqueta le dedicaremos la unidad 3 de este módulo debido a las posibilidades de desarrollo con las que cuenta.

Elementos Interactivos HTML 5

Uno de los grandes cambios de HTML 5 con respecto a versiones anteriores es que ha simplificado la forma de introducir contenidos multimedia que antes requerían de extensiones específicas como Flash.

<DETAILS>

Un elemento que recoge información adicional que puede mostrarse u ocultarse. Hay una cierta divergencia en la forma de entender este elemento, por lo que debe usarse con precaución: para el W3C contendrá un atributo <LEGEND> que por algunos navegadores, contiene elementos <DT> y <DD> como si fuera una lista.

Ante un elemento <DETAILS>, el navegador incorpora un símbolo que permite ampliar/reducir detalles.

```
<details>
<summary>Copyright 1999-2011.</summary>
<p>All pages and graphics on this web site are the property of the company.</p>
</details>
```

<MENU> y <COMMAND>

<MENU> define un menú con sus opciones. Cada elemento del menú se define con <COMMAND> que contiene los atributos @OnClick, con la acción que debe desencadenarse al pulsar sobre el elemento del menú, y @label e @icon con la etiqueta y la imagen asociados a ese elemento.

<MENU> cuenta con el atributo @label para dar un nombre al menú, y @type que permite indicar el tipo de menú de comandos. Hay definidos 3 valores: “context menú” para un menú contextual; “toolbar” para una barra de herramientas; y “list” para una lista de comandos.

Dentro de <MENU> puede utilizarse <LI> para separar elementos, como si fuera una lista.

Cada elemento del menú define con directivas ya conocidas como <BUTTON>m indicando el tipo al que pertenecen (con @type).

Los elementos de <COMMAND>, tan solo se muestran si están contenidos dentro de un <MENU>.

```
<menu>
<li><input type="checkbox" />Red</li>
<li><input type="checkbox" />blue</li>
</menu>
```

Atributos de la etiqueta command

type

- Indica el tipo de comando y el valor por defecto en el mando. **valores: checkbox, radio, command**

checked

- Se utiliza sólo cuando el tipo es una radio o una casilla de verificación y define si el botón de comando respectivo se comprueba o no. **valor: checked**

disabled

- Este es un atributo booleano que indica que el comando no está disponible en la forma actual.**valor: disabled**

label

- Da el nombre del comando, ya que sería muestra al usuario. Debe mencionarse explícitamente y no debe estar vacío. **valores: command, name**

icon

- Da una imagen o una imagen que representa el comando. Debe contener una dirección válida. **valor: url**

radiogroup

- Puede ser utilizado solamente si el comando es una radio. **valor: radiogroup**

Formularios en HTML 5

Los formularios son elementos que han tenido una gran evolución a lo largo del desarrollo de la web. Su misión primordial siempre ha sido permitir que el usuario pueda ingresar información para que sea procesada por un servidor.

Esta operación puede ir desde el envío de un mail hasta la actualización en una base de datos, entre muchas otras posibilidades.

Los formularios pueden potenciarse con el uso de Javascript, ya que es posible utilizar esta tecnología para validar los datos antes de que lleguen al servidor.

Etiqueta form

El elemento **form** es el que agrupa cada uno de los formularios.

Actúa como contenedor y a la vez permite que se ejecuten las entradas de datos.

Tiene algunos atributos que permiten que el usuario interactúe de formas distintas:

- ***action***: Es uno de los más importantes, ya que define la dirección URL a la que se enviará la información del formulario (habitualmente una página en PHP o ASP que procesará el requerimiento del formulario).
- ***accept-charset***: es posible indicar el juego de caracteres que puede manejar el servidor que recibe unformacipon del formulario. Por ejemplo, se puede utilizar el varlo UTF-8.
- ***autocomplete***: Permite sólo *on* y *off* (por defecto *on*) y permite que el navegador rellene aquellos campos *input* de forma automática.
- ***enctype***: Indica la codificación de la información que se enviará al servidor.
    - Tiene 3 posibles estados:
        - *application/x-www-form-urlencoded*
            - por defecto si no se indica.
        - *multipart/form-data*
            - no se codifican los caracteres, debe emplearse cuando se utiliza en el formulario algún campo con opción de upload)
        - *text/plain*

- los espacios se convierten en el símbolo +, y el resto de los caracteres especiales se mantienen sin codificar.

- ***method***: Indica el tipo de cabecera por el que se enviará la información.

    - Permite 2 estados:
        - *GET*
            - Con get pasamos la información (nombre=valor) por medio de la URL y será visible en la barra del navegador. No es recomendable utilizar este método para información privada, pero puede ser de utilidad en los casos en los que se quiera que el usuario guarde la dirección en el historial o en favoritos.
        - *POST*
            - Es recomendable para las transacciones con el servidor donde no deseamos que los datos pasen por la URL, y no pone un límite de caracteres para el traspaso de datos.

- ***name***: Es el nombre del formulario. Si se indica no puede estar vacío, pero puede no ponerse (y el sistema lo creará automáticamente de forma interna).

- ***novalidate***: Un atributo booleano que indica si la información se validará o no.

- ***target***: Al igual que los enlaces, permite los valores:
    - *_blank*,
    - *_self* (por defecto),
    - *_parent*
    - *_top*.

Etiqueta input

El principal elemento para la introducción de datos sigue siendo el **input**, y en esta ocasión llega con muchísimos atributos y con muchos tipos de entradas.

El principal atributo del sistema de entrada de datos es el ***type***.

Hasta ahora los valores más habituales eran *hidden* o *text* que básicamente eran textos alfanuméricos, pero ahora eso ha cambiado, y la lista de tipos es bastante grande:

- ***hidden***: Es un contenido oculto que suele corresponder a una serie de caracteres alfanuméricos. Este valor no puede ser manipulado por el usuario.

- ***text***: Campo de texto que no permite saltos de línea. Si se añade el atributo ***value*** será el contenido original del campo.

- ***search***: Campo de texto que no permite saltos de línea y que indica que será usado como contenido de una búsqueda. Si se añade el atributo ***value*** será el contenido original del campo.

- ***tel***: Campo de texto que indica un número de teléfono. No tiene ningún formato en particular, por lo que se pueden incluir espacios o guiones, por ejemplo.

- ***url***: Campo de texto en el que se pueden incluir direcciones URL absolutas. Se puede hacer una especie de *autosuggest* junto a otros elementos.

- ***email***: Campo de texto que permite incluir correo electrónico. Principalmente si se incluye el atributo *value* permite una única dirección, y si es el *múltiple* puede incluir varias.

- ***password***: Campo de texto, pero que aparece oculto para el usuario y así proteger la información.

- ***datetime***: Permite una fecha y hora válida en un formato como *0037-12-13T00:00Z*, *1979-10-14T12:00:00.001-04:00* o *8592-01-01T02:09+02:09*. Si se indica un número, serán los milisegundos transcurridos desde el 01/01/1970.

- ***date***: Permite indicar una fecha. El atributo *value* indica la fecha en cuestión y el *min* y *max* los límites del rango de fecha. El atributo *step* marca los milisegundos (por defecto 1 día) entre un valor y otro. El formato sería *2010-03-09*.

- ***month***: Permite indicar un mes. Tiene los mismos atributos que *date*. El formato sería *2010-03*.

- ***week***: Permite indicar la semana del año. Tiene los mismos atributos que *date*. El formato sería *2010-W03*(tercera semana).
- ***time***: Permite indicar una hora. Tiene los mismos atributos que *date*. El formato sería *17:23* y opcionalmente los segundos, *17:23:59*.

- ***datetime-local***: Es igual que el *datetime* pero no permite indicar ningún tipo de desfase horario.

- ***number***: Representa un valor numérico. Este puede ser negativo (si empieza con un guión -), seguido de un número y con decimales (si se le indica un punto .) y para acabar, un exponente, empezando con una letra e, seguida de un símbolo positivo + o negativo – y la cantidad exponencial. Un ejemplo completo

sería: *-12345.678e+90*.

- ***range***: El tipo *range* permite un *slide* en donde el número (valor) no es lo 100% importante, sino que es el formato en el que se muestra. Un poco más abajo, un ejemplo.

- ***color***: Este elemento representa un color. El formato es del estilo *#A0F59E* (o sea, el RGB).

- ***checkbox***: Este elemento ya existía, y es el que permite elegir unas cuantas opciones de un grupo de ellas.

- ***radio***: Al igual que el anterior, permite elegir de un grupo, pero sólo una de las opciones.

- ***file***: Éste permite la opción de subir archivos. Con él se pueden seleccionar archivos que luego serán enviados al sistema. Si se le indica el atributo *multiple* se podrán seleccionar varios ficheros a la vez. Además, si se indica el atributo *accept* se puede limitar ficheros según su MIME-type, por ejemplo con *audio/**, *video/** o*image/**.

- ***submit***: Este es el botón que permite el envío de información al sistema. Puede ir acompañado de otros atributos que hacen que "ese botón" haga cosas que no son las generales del formulario. Los parámetros son: *formaction* (que hace ese botón), *formenctype* (la codificación), *formmethod* (la forma de envío),*formnovalidate* (la validación) y *formtarget* (en qué "ventana" se abre)

- ***image***: Es igual que el *submit* pero permite ser una imagen en vez de un botón en sí. Para ello ha de ir acompañado del atributo *src* que indica la dirección URL de la imagen.
- ***reset***: Otro botón que si pulsas deja los valores del formulario en su valor inicial.

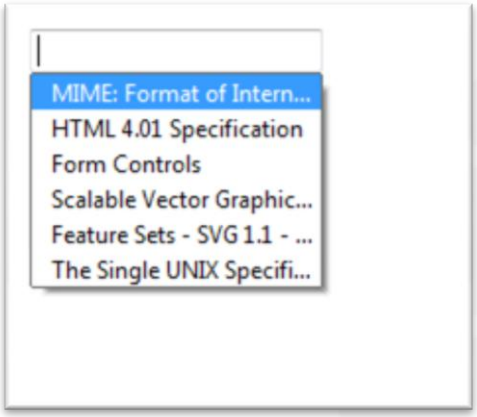
- ***button***: Un botón que "no hace nada". Suele utilizarse habitualmente con funcione JavaScript que son los que realizan la acción.

Ejemplo de *autosuggest* gracias al *type="url"*

```
<input type="url" name="location" list="urls">
<datalist id="urls">
  <option label="MIME: Format of Internet Message Bodies"
value="http://www.ietf.org/rfc/rfc2045">
  <option label="HTML 4.01 Specification" value="http://www.w3.org/TR/html4/">
  <option label="Form Controls" value="http://www.w3.org/TR/xforms/slice8.html#ui-
commonelems-hint">
  <option label="Scalable Vector Graphics (SVG) 1.1 Specification"
value="http://www.w3.org/TR/SVG/">
```

```
  <option label="Feature Sets - SVG 1.1 - 20030114"
value="http://www.w3.org/TR/SVG/feature.html">
  <option label="The Single UNIX Specification, Version 3" value="http://www.unix-
systems.org/version3/">
</datalist>
```

Visualización desde Firefox



Ejemplo de *range*

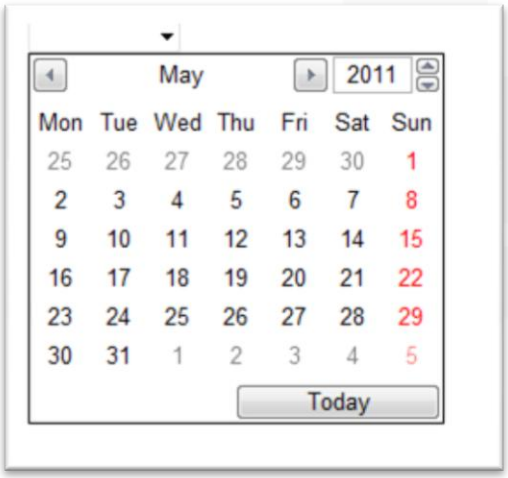
```
<input type="range" min="-100" max="100" value="0" step="10" name="power"
list="powers">
<datalist id="powers">
  <option value="0">
  <option value="-30">
  <option value="30">
  <option value="+60">
</datalist>
```

Visualización en Google Chrome

Ejemplo de date:

```
<input name="birthday" type="date">
```

Visualización en Opera:

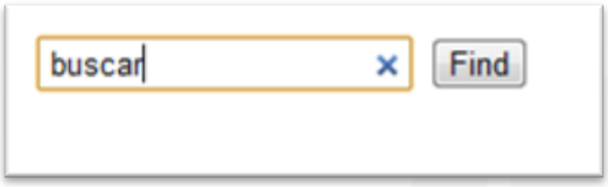


Ejemplo de caja de búsqueda:

```
<form>
 <input name="q" type="search">
```

```
 <input type="submit" value="Find">
</form>
```

Visualización en Chrome:



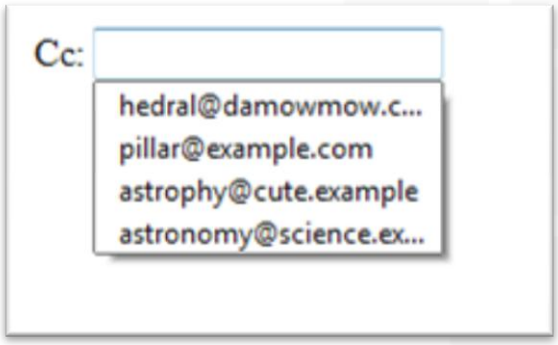
Además de los distintos tipos que puede haber, existen distintos atributos:

- ***autocomplete***: Permite el estado *on* y *off* y hace que el campo se rellene o no automáticamente si el sistema es capaz de identificar la información.
- ***list***: Es el valor de una lista identificada por el *datalist*. Con esto se pueden crear sistema de "autosugerencias".
- ***readonly***: Es un valor booleano, y permite o no que el usuario pueda modificar el contenido.
- ***size***: Es el número de caracteres que se muestran por pantalla.
- ***required***: Otro atributo booleano que indica que ese elemento ha de incluirse obligatoriamente.
- ***multiple***: Permite que algunos elementos seleccionen más de un valor.
- ***maxlength***: Es la cantidad máxima de caracteres que se permite.
- ***pattern***: Indica el formato de contenido que tiene que tener… Por ejemplo: *[0-9][A-Z]{3}* sería un número seguido de 3 letras en mayúsculas.
- ***min*** y ***max***: Indican los valores mínimo y máximo que se permiten.
- ***step***: Indica el valor de paso cuando hay elementos que permiten saltar.
- ***placeholder***: Es un "ejemplo" del valor que se puede incluir. Desaparece automáticamente al hacer clic sobre él.

Ejemplo de selección múltiple de cuentas de correo:

```
<label>Cc: <input type=email multiple name=cc list=contacts></label>
<datalist id="contacts">
  <option value="hedral@damowmow.com">
  <option value="pillar@example.com">
  <option value="astrophy@cute.example">
  <option value="astronomy@science.example.org">
</datalist>
```

Visualización en Firefox:



Actividades de Aprendizaje

Actividad 1

1. Tomando como base la maquetación creada en la Unidad 5, incorporar al menos 2 elementos multimediales en HTML5

Te invitamos a compartir los resultados en la sección "Actividades" de la unidad 6.

Resumen

En esta Unidad…

Trabajamos con los nuevos elementos multimediales de HTML5.

En la próxima Unidad…

Comenzaremos a trabajar con las nuevas incorporaciones en el nivel 3 de CSS.