APUNTES HTML5: HTML

José Juan Urrutia Milán

Reseñas

- Curso HTML5: https://www.youtube.com/playlist?list=PLU8oAlHdN5BnX63lyAeV0LzLnpGudgRrK
- w3c (página que decide los estándares CSS/HTML...): www.w3c.org

Siglas/Vocabulario

- **IDE:** Entorno de Desarrollo Integrado.
- **BBDD**: Bases de Datos.
- **API:** Conjunto de paquetes ya creados y predefinidos por otros usuarios.

Leyenda

Cualquier abreviatura o referencia será subrayada.
Cualquier ejemplo será escrito en **negrita.**
Cualquier palabra de la que se pueda prescindir irá escrita en cursiva.
Cualquier abreviatura viene explicada a continuación:

Abreviaturas/Referencias:

- 123: Hace referencia a cualquier número.
- nombre: Hace referencia a cualquier palabra/cadena de caracteres.
- a: Hace referencia a cualquier caracter.
- cosa: Hace referencia a cualquier número/palabra/cadena.
- Tipo_var o **…** : Hace referencia a cualquier tipo de variable primitiva o de tipo String.
- nombre_var: Hace referencia a cualquier nombre que se le puede dar a una variable.
- código: Hace referencia a cualquier instrucción. (Se usará para indicar dónde se podrá inscribir código.)
- variable: Hace referencia a cualquier variable.
- condición: Hace referencia a cualquier condición. Entiéndase por condición, una afirmación que devuelve un true o un false. Ej: (variable == 123). *Una variable del tipo boolean puede ser usada como una condición.

Índice

span

tildes

label

pre

Título III: Listas

Listas desordenadas / de puntos

Listas ordenadas / numeradas

Título IV: Figuras / Imágenes

Descripción a pie de imágen

Título V: Vídeos

src

controls

autoplay

loop

poster

preload

width

Cargar múltiples formatos

Crear reproductor de vídeo

Título VI: Audios

src

controls

autoplay

loop

preload

Cargar múltiples formatos

Título VII: Botones

type

Título VIII: Formularios

action

method

Elementos de formularios

text

password

email

search

url

tel

number

range

date

month

time

datetime

submit

button

file

Propiedades de elementos

autocomplete

autovalidate

maxlength

value

placeholder

required

multiple

autofocus

pattern

form

min max step

list

Propiedades especiales

Título IX: Datalist

Título X: Output

Título XI: Canvas

Mensaje

Título XII: Documentos HTML anidados

Título XIII: Vínculos

href

Vínculos de navegación

mailto

Estados

Título XIV: ComboBox

Componentes

Introducción

El objetivo de este curso es aprender HTML.
HTML es un lenguaje de etiquetas que conforma la estructura principal de todas las páginas web.
HTML puede (o debe) ser decorado con código CSS, para aplicar un diseño más vistoso.
La parte funcional de HTML se hace con JavaScript, un lenguaje de programación.
HTML, CSS y JavaScript conforman la estructura HTML5, donde HTML aporta estructura, CSS apariencia y JavaScript interactividad y animación.
HTML no es case sensitive, por lo que se puede indicar en mayúsculas o minúsculas, aunque por convención se suele hacer en minúsculas.
Los ficheros html se guardan con la extensión **.html** .

Capítulo I: Conceptos básicos

Comentarios

Para introducir comentarios en html (líneas que no se tendrán en cuenta), especificamos **<!--** al principio y **-->** al final del comentario:
<!--<h1>Titulo</h1>-->
La etiqueta superior no se mostrará.

Título I: Estructura básica

HTML consiste en etiquetas, las cuales deben abrirse y cerrarse (con /)
Todo el código html debe ir dentro de una etiqueta html.
Antes de esta primera etiqueta, hay que especificar la etiqueta **DOCTYPE** para especificar el formato de html.
Dentro de la etiqueta html, observamos dos principales: la etiqueta head y la etiqueta body.
Las etiquetas html pueden o no llevar atributos.
(cono indicar el idioma en la etiqueta html):
<html lang="es">

head

En la etiqueta head, se suelen incluir estructuras que no se verán en la página web: hojas de estilo, archivos de javascript externos o internos, iconos, el título de la página, etiquetas meta… (las cuales hacen por ejemplo que los buscadores categoricen nuestra página).

Etiquetas meta

-Una etiqueta meta puede ser: **<meta charset="utf-8"/>**
, la cual especifica si nuestra página tiene caracteres unicode, chinos, árabes…
-Otra etiqueta meta (**<meta name="description" content="txt"/>**) que sirve para categorizar las páginas web en navegadores.

-Otro ejemplo puede ser:
<meta name=”keywords” content=”html5, css3, Javascript, diseño web”>, la cual indica las palabras clave de la página web separados por comas que puede ser útil en buscadores.
Las etiquetas meta de búsqueda están empezando a quedarse obsoletas.

body

Dentro de body, incluimos todas las etiquetas visibles de la página web. Todo lo que introducimos aquí dentro, se verá en la página web.

```
<!DOCTYPE html>
<html>
    <head>
    </head>

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

propiedad id

Se puede establecer una propiedad con la que identificar a diferentes etiquetas:
<h1 id=”tituloPrincipal”>Titulo</h1>

Título II: Etiquetas title y link

Title

La etiqueta title , que se especifica en el body, nos permite darle un título a nuestra página web (cambiar el nombre de la ventana). Los buscadores también tienen en cuenta los títulos de las páginas web. (por encima de las palabras clave).
<title>texto</title>

Link

La etiqueta link (que se suele especificar después de la etiqueta title), sirve para enlazar o incluir documentos adicionales a nuestra página web, como hojas de estilo o documentos JavaScript.
La etiqueta link lleva dos atributos: rel (donde indicamos el tipo de documento) y href (donde indicamos la ruta del documento):
<link rel="tipo" href="ruta"/>

Añadir documentos CSS

<link rel="stylesheet" href="nombre.css"/>

Establecer imagen como icono de la página

<link rel="icon" href="nombre.ico"/>

Título III: Organización del body / Estructura

table

Para crear una tabla, creamos una etiqueta **table** la cual se abre y se cierra.
En su interior, creamos un etiqueta **tr** que hará referencia a las filas. Esta también se abre y se cierra.
Dentro de una fila o **tr**, creamos un **td** por cada celda de la tabla. Introducimos el componente y cerramos la celda y así hasta crear la tabla que queramos hacer.
Dentro de una tabla, también podemos indicar un título de la tabla con la etiqueta **caption**.
Podemos también dar cabeceras a filas y columnas, con etiquetas **th** indicando con la propiedad **scope** si son de columnas (**"col"**) o de filas (**"row"**):

```
<table>
<caption>Tabla</caption>
```

<th scope=”col”>Fila 1</th><th scope=”col”>Fila 2</th>
<tr> <td> 0,0 </td> <td> 0,1 </td> </tr>
<tr> <td> 1,0 </td> <td> 1,1 </td> </tr>
</table>

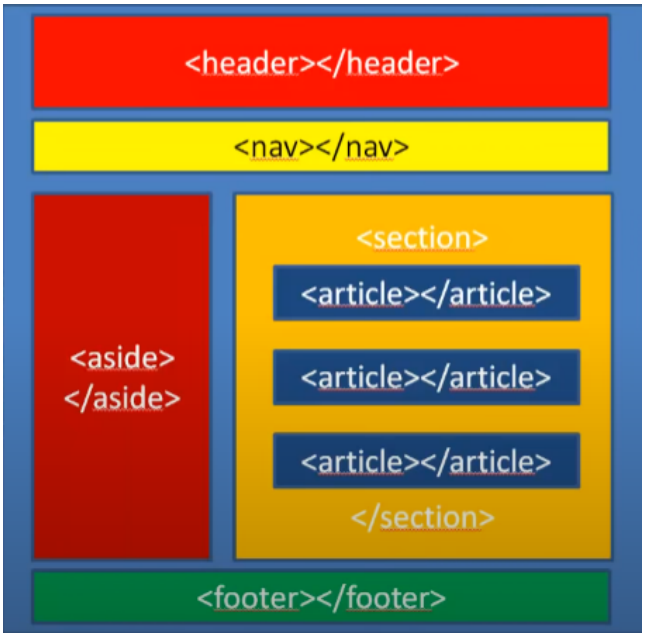
Indicar las propiedades sobre td:
rowspan: indicar cuantas filas abarcará esta casilla de alta.
colspan: indicar cuantas columnas abarcará esta casilla de ancha.

div

El body se dividía en **div** (** **), los cuales son cajas que contienen diferentes componentes, aunque estos se quedaron obsoletos debido a los nuevos dispositivos como tablets o móviles. (A pesar de esto, todavía se siguen usando a modo de contenedores o de rectángulos, por lo que no están obsoletos).
*Los div son de tipo block.
**Los div no aportan información visual.

Forma actual

La actual forma de dividir un body, es la siguiente:



header, donde incluir la cabecera del sitio.

nav, donde incluir una barra de navegación o reproductor de vídeo.
section/artice, donde aparece la información principal (se puede crear una section sin un article).
aside, donde puede aparecer una barra con vínculos, por ejemplo.
footer, donde especificar información adicional.

Todas las etiquetas deben abrirse y cerrarse.
Se pueden incluir tantas etiquetas de cada tipo como se desee (imagen orientativa).
Dentro de estas etiquetas estructurales, podemos indicar nuevas etiquetas como títulos… o nuevas etiquetas estructurales (como un header o footer dentro de un article de un section, ...).
**Estas etiquetas no son obligatorias, pero son recomendables sobre todo en dispositivos como tablets o teléfonos móviles con resoluciones menores y diferentes renderizados.
***El orden de estas estructuras se mostrará de arriba a abajo según se muestre de arriba a abajo en código html, aunque los diferentes componentes se pueden reordenar con CSS.

Capítulo II: Etiquetas principales

Etiquetas block

Son aquellas etiquetas que se separan de las demás con un salto de página por encima y por debajo y además ocupan todo el ancho del documento.
La mayoría de etiquetas son etiquetas block.

Etiquetas inline

No generan salto de página ni por encima ni por debajo y además no ocupan todo el ancho del documento, sólo la parte esencial para poder mostrar su contenido.

Añadir espacio en blanco

Para añadir un espacio en blanco entre dos etiquetas inline, escribir entre ellas: ** ** :
**   **

Añadir salto de línea

Para añadir un salto de línea, indicamos la etiqueta **
** .
*No es necesario cerrarla.
**Puede ser de gran utilidad en formularios.

Título I: Títulos

Existen ciertas etiquetas que nos permiten crear títulos. Estas etiquetas son conocidas como etiquetas h y hay seis tipos: desde h1 hasta h6 (el tamaño de letra va disminuyendo).
Entre la etiqueta h de apertura y cierre, indicamos el texto:
<h1>txt</h1>
<h5>txt</h5>

*Por compatibilidad, es conveniente que sólo haya una etiqueta h1 y que las demás se vayan incluyendo por jerarquía (h2 > h3 > … > h6). Aunque parece ser que es posible incluir varias de estas, siempre y cuando estén en diferentes bloques estructurales (diferentes **article** , por ejemplo).

hgroup

hgroup nos permite agrupar varios títulos que están relacionados. No cambiará la forma en la que se visualiza la página pero será de ayuda para el navegador.

Título II: Párrafos

Los párrafos son varías líneas de texto que nos muestran información, similares a los títulos h pero de menos importancia:
** txt **

Etiquetas de marcado

Podemos indicar etiquetas de marcado o de propiedades dentro de un párrafo o título (o dentro de cualquier estructura de contenga un texto), de la siguiente forma:
<estructura><marcado>texto</marcado></estructura>

*Estas etiquetas, aparte de mostrar un efecto visual, aportan sobre todo un sentido semántico al texto (ya que el efecto visual se puede dar fácilmente con CSS).

Tipos de etiquetas de marcado

Etiqueta	Uso visual y semántico
<mark>	Marcar un texto (resalto en amarillo).
****	Cursiva (sustituye a la obsoleta <i>)
****	Marcar texto importante. (sustituye a la obsoleta <b>)

<small>	Presentar textos legales. (coyright, derechos reservados...)
<big>	Para texto grande.
<sub>	Para subíndice.
<sup>	Para superíndice.
<ins>	Para subrayar texto.
<del>	Para tachar texto.
<cite>	Títulos de libros, películas...
<address>	Información del contacto (debe ir en footer).
<time> (*)	Representar fechas y horas (no tiene uso visual).

***<time datetime="2014/01/15" pubdate>Noticia publicada el día 15/01/2014</time>**

span

Las etiquetas span nos permiten contener trozos de texto de otras etiquetas como p, h1, il… Estas etiquetas no aportan ningún formato visual, sólo nos permiten identificar a porciones de texto para añadirles un estilo desde CSS especificando un atributo **id**:
(html)
** Hola, soy Juan **

(css)
#azul{color:#00F}

tildes

A veces no se mostrarán las tildes, para hacer que estas se muestren siempre, indicar **á** donde aparezca una, sustituyendo **a** por la vocal que posea la tilde. De esta forma:
** Vivo en Almería <!--Vivo en Almería-->**

*Se pueden indicar tantas tildes en el mismo texto como queramos:
** El huracán derribó esa casa **

label

Las etiquetas **label** son idénticas a las **p** , salvo que estas se utilizan para incluir pocas palabras, ya que estas etiquetas son etiquetas **inline**, a diferencia de las **p** que son **block** .

pre

Las etiquetas **pre** son idénticas a las **p** , salvo que cualquier espacio o salto de línea que se incluya dentro de código de una etiqueta **pre** se verán reflejados en la página web.

Título III: Listas

Las listas nos permiten enumerar diferentes componentes.

Listas desordenadas / de puntos

Las listas desordenadas (o listas de puntos) nos permiten mostrar diferentes elementos.
Para ello, creamos una etiqueta **
** que contendrá diferentes etiquetas **
texto
**, donde cada componente li es un elemento de la lista.

Listas ordenadas / numeradas

Las listas ordenadas (o listas numeradas) nos permiten mostrar diferentes elementos.
Para ello, creamos una etiqueta **
** que contendrá diferentes etiquetas **
texto
**, donde cada componente li es un elemento de la lista.

Título IV: Figuras / Imágenes

Podemos usar la etiqueta **figure** para insertar figuras o imágenes (u otros componentes).
Elementos independientes a la página web pero que tienen algo que ver con esta.
(Importante seleccionar imágenes sin copyright).
(Importante que las imágenes sean formatos: .jpg , .gif , .png).

Para insertar imágenes, creamos una etiqueta **img** (dentro de la etiqueta **figure)** a la que le indicamos el atributo **src="ruta"**. (la ruta puede ser absoluta (https:...) o relativa (C:/...)), con el que podemos seleccionar la imagen a mostrar.
*Si la imagen se encuentra en la misma carpeta que nuestro archivo html, no es necesario especificar directorios, ya que podemos usar este directorio como padre.

```
<figure>
    <img src="ruta"/>
</figure>
```

*Se puede modificar el tamaño de la imagen con CSS.

Descripción a pie de imágen

Para crear una descripción a pie de imágen, debemos indicar una etiqueta **figcaption** después de nuestra etiqueta **img** (todo dentro de **figure**).

```
<figure>
    <img src="ruta"/>
    <figcaption>txt</figcaption>
</figure>
```

Título V: Vídeos

Para añadir un vídeo, usamos la etiqueta **video**, a la que le podemos aplicar diferentes atributos.

*Los principales formatos que soportan los navegadores son .mp4 y .ogg , así que intentar usar sólo esos.

src

Especifica la ruta del vídeo (absoluta o relativa).

controls

Especifica si el vídeo tendrá botones (play, pause, pantalla completa, volúmen, ...).
*No hace falta indicarle ningún valor, sólo nombrarlo o no.

autoplay

Establece si el vídeo se empezará a reproducir cuando se cargue la página o no.
*No hace falta indicarle ningún valor, sólo nombrarlo o no.
**Situar esta propiedad en último lugar, ya que si no se empezará a reproducir el vídeo sin haber cargado una propiedad importante que se especifica después del autoplay.

loop

Especifica si el vídeo se reproducirá una y otra vez o sólo una vez.
*No hace falta indicarle ningún valor, sólo nombrarlo o no.

poster

Incluye una imagen jpg a modo de portada de vídeo.
Indicar como valor la ruta de la imagen.
*Tener en cuenta que la imagen tenga la misma resolución que el vídeo.

preload

Especifica algunas características importantes del vídeo antes de cargarlo (fps, duración...).

width

Establece el ancho del vídeo (Al especificar sólo width, el navegador automáticamente define el height de manera proporcional).
*Ocurre lo mismo con **heigth**.
**No se recomienda especificar ambos (o uno u otro).

Cargar múltiples formatos

Es posible que un formato no se pueda cargar en un navegador específico, por lo que para solucionar esto, podemos indicar múltiples versiones del vídeo con diferentes extensiones para asegurarnos de que alguna sea compatible con el navegador.
Esto lo hacemos con la etiqueta **source**:

```
<video controls>
<source src="video.mp4">
<source src="video.ogg">
</video>
```

Crear reproductor de vídeo

Utilizando CSS y HTML podemos crear nuestro propio reproductor de vídeo. Para ello, tenemos que crear botones para el play/pause y un div que usaremos como barra de desplazamiento.

Título VI: Audios

Para añadir un audio, usamos la etiqueta **audio**, a la que le podemos aplicar diferentes atributos.
*Los principales formatos que soportan los navegadores son .mp3 y .ogg , así que intentar usar sólo esos.
**Intentar que los audios no tengan copyright, ya que los usaremos en una página web.

src

Especifica la ruta del audio(absoluta o relativa).

controls

Especifica si el audio tendrá botones (play, pause, volúmen, ...).
*No hace falta indicarle ningún valor, sólo nombrarlo o no.
**Si este no se especifica, no se mostrará nada en la página web aunque se podrán crear unos controles propios con la ayuda de JavaScript.

autoplay

Establece si el audio se empezará a reproducir cuando se cargue la página o no.
*No hace falta indicarle ningún valor, sólo nombrarlo o no.
**Situar esta propiedad en último lugar, ya que si no se empezará a reproducir el audio sin haber cargado una propiedad importante que se especifica después del autoplay.

loop

Especifica si el audio se reproducirá una y otra vez o sólo una vez.
*No hace falta indicarle ningún valor, sólo nombrarlo o no.

preload

Especifica algunas características importantes del audio antes de cargarlo (duración...).

Cargar múltiples formatos

Es posible que un formato no se pueda cargar en un navegador específico, por lo que para solucionar esto, podemos indicar múltiples versiones del audio con diferentes extensiones para asegurarnos de que alguna sea compatible con el navegador.
Esto lo hacemos con la etiqueta **source**:

```
<audio controls>
<source src="audio.mp3">
<source src="audio.ogg">
```

</audio >

Título VII: Botones

Para ello, usaremos la etiqueta <button>, a la que se le pueden especificar diferentes atributos html así como dar un aspecto con CSS.
Es recomendable especificar un **id** a todos los botones para luego identificarlos desde código JavaScript.
El nombre del botón lo especificamos entre las etiquetas de apertura y cierre.
<button id=”b” type=”button”>Púlsame</button>

type

Especifica qué tipo de botón estamos añadiendo.
Especificar “button” para un botón normal.

Título VIII: Formularios

Para crear un formulario, creamos una etiqueta **form** a la que le damos diferentes propiedades como un **name**, un **id**, un **method** (**method=”get”**)
<form name=”formulario” id=”formulario” method=”get”>
</form>

*Es conveniente que todo el texto de un formulario esté dentro de etiquetas **<label>** en vez de etiquetas **<p>** .

action

Establece un archivo (normalmente .php) que procesa la información del formulario. La información se enviará con la tecnología especificada en **method** y dentro de un JSON en formato clave:valor, donde la clave será el atributo **name** de cada campo.

method

Establece de qué forma se enviará el formulario.
Si indicamos **method=”get”** , la información viajará a través de la barra de ulr. Si especificamos **“post”** , la información viajará de forma transparente.

Elementos de formularios

Dentro de un formulario, podemos crear varios elementos diferentes: un input de texto, un input de contraseña, un botón submit…
Todos los elementos que incluyamos en el formulario, se establecerán uno al lado de otro, en la misma fila (a no ser que introduzcamos un salto de línea).

La propiedad type hace que los campos se autovaliden de forma automática sin escribir nada de código JavaScript.
Las etiquetas **input** no hace falta cerrarlas (no indicar /).
Los diferentes input se diferenciarán entre sí por la propiedad **type**.
Dependiendo del buscador, puede que los campos no funcionen bien.
Los valores que puede adoptar **type** son:

*Es recomendable meter a los elementos de los formularios dentro de tablas **table** .

text

Crea un input de texto. No presenta validación.

password

Idéntico a **text** pero sustituye todos los caracteres por puntos.

email

Crea un cuadro de texto que obliga al usuario a introducir al menos una arroba.

search

Crea un input de texto. No presenta validación.

url

Crea un input de texto. Presenta validación.

tel

Crea un input de texto. Presenta validación.

number

Crea un input de texto que sólo deja introducir números. Crea a su vez dos flechas de aumento y decremento que se pueden personalizar con **min max** y **step**.

range

Crea un deslizador seleccionador de números. Se puede personalizar con **min max** y **step**.
*Cuando se mueve, emite un evento de tipo **“change”.**

date

Crea un input de texto en el que introducir día mes y año en el formato dd/mm/aaaa, presenta un calendario que se despliega.

month

Crea un input de texto en el que introducir mes y año. Presenta un calendario desplegable.

time

Crea un input de texto en el que introducir hora y minuto.

datetime

Crea dos input de texto (un date y un time) además de devolver la zona horaria de la región.

submit

Botón para enviar el formulario dependiendo del **method** especificado.
*Se puede cambiar el texto del botón (que por default es 'Enviar consulta') con la propiedad **value="texto" .**

button

Botón normal.

file

Genera un botón que nos permite abrir un asistente para seleccionar un archivo de nuestro dispositivo.
*Cuando selecciona un archivo, emite un evento de tipo **"change".**
Presenta una propiedad JavaScript llamada **files que devuelve un array con todos los archivos seleccionados.

Propiedades de elementos

autocomplete

Establece si se mostrarán sugerencias a modo de menú desplegable (**"on"**) en los input de texto o no (**"off"**).
*Por defecto, se encuentra on.

autovalidate

Establece si los campos se autovalidarán automáticamente dependiendo de su **type** o no lo harán.
*Por defecto, se valida.

maxlength

Establece la longitud máxima de caracteres que se pueden incluir.

value

Establece el texto del campo de texto (o el valor del input range) que se muestra al cargar la página.

placeholder

Establece un texto en el input (a modo de marca de agua) que se borra cuando se introduzca texto en él. (el placeholder es igual al hint de AndroidStudio).

<input type="text" placeholder="Nombre">

required

Establece que un **input** debe tener algún valor cuando se pulse en el botón submit. De lo contrario, no se podrá enviar nada.

*Al ser booleano, no es necesario indicar parámetros, simplemente indicarlo:

<input type="text" required>

**Por defecto, se encuentra en false.

multiple

Establece que un input puede enviar múltiples valores a la vez. El usuario debe separar estos valores por comas.

*Al ser booleano, no es necesario indicar parámetros, simplemente indicarlo.

**Por defecto, se encuentra en false.

autofocus

Establece que el elemento recibe el foco al cargar la página.

*Al ser booleano, no es necesario indicar parámetros, simplemente indicarlo.

**Por defecto, se encuentra en false.

pattern

Sirve para crear validaciones especiales.

<input pattern="[0-9]{5}">

Este input sólo permite que se incluyan 5 dígitos que deben estar cada uno entre el 0 y el 9.
*No se puede usar sobre un objeto con la propiedad **type**.
**Especificar “[a-z]” para de la “a” a la “z” minúsculas, “[A-Z]” de la “A” a la “Z” mayúsculas y “[A-Za-z]” de la “a” a la “z” sin importar mayúsculas y minúsculas.

form

Crea un objeto de formulario (input) fuera de un formulario pero que funciona como si estuviese dentro de él. Indicar name del formulario padre:
<form><input type=”text”></form>

<form name=”formulario”></form>
<input type=”text” form=”formulario”>

No existe diferencia entre estos dos códigos.

min max step

Hace que el campo tenga dos flechas de aumento y decremento, que hagan que el número establecido incremente o decremente el **step** indicado step también restringe los números que no se pueden alcanzar con este step.
También podemos indicar un valor mínimo o máximo al campo.
<input type=”number” min=”10” max=”100” step=”2”>
*Por defecto, no existe min o max y step = 1.
Propiedad de **type number o **range**.

list

Específica a un elemento input que debe coger la información de un **Datalist** con el id especificado.
Al aplicar esta propiedad, el input se convierte en una especie de ComboBox (como el JComboBox de Java o el ComboBox de python)

(ver Título IX) (dentro de un form):
<input type=”tel” list=”telefonos”>

Propiedades especiales

Los formularios tienen unas propiedades especiales que dan a elementos desde código JavaScript para personalizarlos aún más. (Consultar apuntes JavaScript I, Capítulo I, Título VI).

Título IX: Datalist

Los datalist son unas listas de datos que podemos almacenar a modo de “clave valor” (entrecomillado ya que no es del todo así).
Estas listas no aportan nada visual a nuestra página web, sólo funcionalidad.

Para crear un datalist, creamos una etiqueta **datalist** con una id. Dentro de esta etiqueta, creamos tantas etiquetas **option** como queramos (una por cada valor a almacenar). Estas etiquetas tendrán dos propiedades:

- **value** la cual almacena el valor de la etiqueta.
- **label** la cual da nombre a la etiqueta (recomendable mostrar algunos espacios en blanco antes).

<datalist id=”telefonos”>
<option value=”999999999” label=“ Teléfono 1”>
<option value=”999999998” label=“ Teléfono 2”>
<option value=”999999997” label=“ Teléfono 3”>
</datalist>

Título X: Output

Elemento output que nos sirve de “display” donde poder reflejar un valor. Similar a una etiqueta **p** .
<output id=”texto”>Texto</output>

Título XI: Canvas

La API Canvas nos permite mostrar gráficos y animaciones a través del elemento principal de esta API: Canvas.

Un Canvas es una especie de lámina (JPanel de Java, Frame de python...) que se coloca sobre nuestra página web.
Los Canvas adquieren el mismo color de fondo que la página web (aunque se puede cambiar desde CSS).
Para crear un Canvas, crearemos una etiqueta **canvas** a la que le podemos especificar una serie de propiedades como **id** , **width** , **height** …
<canvas id=”lienzo” width=”500”, height=”300”>
</canvas>

Los elementos Canvas tienen una serie de métodos JavaScript que podemos utilizar para dibujar, borrar en el Canvas, crear un degradado en objeto Canvas...
(Consultar Apuntes JavaScrip It, Capítulo II).

Mensaje

Por convención, cuando se crea un canvas se suele poner dentro un mensaje por si el navegador que ejecuta el canvas no lo soporta, que no haya un espacio en blanco en la página web.
<canvas id=”lienzo” width=”500”, height=”300”>
Su navegador no soporta el elemento html5 Canvas.
</canvas>
*Si nuestro navegador soporta canvas, este mensaje no se mostrará.

Título XII: Documentos HTML anidados

Podemos incluir un fichero HTML dentro de otro, con ayuda del componente **<iframe>** . Para ello, será necesario indicarle la ruta del archivo html , así como un width y un height:
(Dentro de un documento HTML):

```
<section>
    <iframe src="archivo.html" width="500" height="400">
    </iframe>
</section>
```

Título XIII: Vínculos

Podemos incluir vínculos que nos llevaran a otras páginas web gracias a etiquetas **<a>.**
Al pinchar en estos vínculos, la página html actual se convertirá en la dirección indicada en la misma pestaña.
Para usar una etiqueta **a** , debemos definir un texto sobre el que pinchará el usuario (el cual aparecerá en azul y subrayado) y un enlace que podemos especificar de distintas formas:

href

Establece el enlace de la página web o documento que abre al pinchar en el vínculo:

```
<a href="https://www.google.com">Google</a>
<a href="documento.pdf">Google</a>
```

Vínculos de navegación

Podemos establecer que un vínculo nos lleve a un sitio específico de la página actual. Esto lo hacemos indicando a un contenedor como un ** ** un **id** específico y creando un vínculo con un **href** igual a una almohadilla y a ese valor de **id**.
De esta forma, cuando pulsemos en el vínculo, la página nos llevará a la posición en la que se encuentra el **div** con el **id**.

** código **
código
<a href=”#contenedor”>Ir al contenedor<a>

*Si sólo especificamos la almohadilla, iremos a la parte superior de la página web:
Ir al inicio

mailto

Abre el gestor de correos electrónicos preferido del usuario y abre la pestaña de enviar un correo al correo especificado:
Enviar

Estados

Un vínculo puede tener 4 estados diferentes.
Estos estados sirven para modificar el estilo del vínculo desde CSS con selectores de pseudoclases a los que se le indican los diferentes valores (después de los dos puntos):

- **reposo :** Estado default. Valor: **link** .
- **visitado :** Cuando la página está en tu historial (adopta color marrón por defecto). Valor: **visited** .
- **rollover :** Cuando el ratón se sitúa encima del vínculo. Valor: **hover** .
- **click :** Cuando el ratón está pulsando el vínculo. Valor: **active** .

Título XIV: ComboBox

Podemos crear un comboBox con la etiqueta **<select>**, especificando entre estas dos todos los componentes del comboBox.

Componentes

Cada componente estará dentro de una etiqueta **<option>** , especificando una propiedad **value** para acceder desde JavaScript y un nombre entre las etiquetas de inicio y cierre.

```
<select id=”comboBox”>
<option value=”null”>Seleccione una</option>
<option value=”h”>Hombre</option>
<option value=”m”>Mujer</option>
<option value=”wtf”>Helicóptero Apache Nuclear</option>
</select>
```