HTML5 Y CSS3

Introducción

¿Qué aprenderé en este tema?

- Qué es HTML y su evolución
- Qué es CSS y su evolución
- Partes de una página Web
- Añadir contenido a una Web
- Dotar de estilos visuales básicos a nuestra web usando CSS

Definiciones:

HTML, siglas en inglés de ***HyperText Markup Language*** ('lenguaje de marcas de hipertexto'), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros. Es un estándar a cargo del *World Wide Web Consortium* (W3C) o Consorcio WWW, organización dedicada a la estandarización de casi todas las tecnologías ligadas a la web, sobre todo en lo referente a su escritura e interpretación. HTML se considera el lenguaje web más importante siendo su invención crucial en la aparición, desarrollo y expansión de la World Wide Web (WWW). Es el estándar que se ha impuesto en la visualización de páginas web y es el que todos los navegadores actuales han adoptado. ***NO ES UN LENGUAJE DE PROGRAMACIÓN***

El lenguaje HTML basa su filosofía de desarrollo en la diferenciación. Para añadir un elemento externo a la página (imagen, vídeo, *script*, entre otros.), este no se incrusta directamente en el código de la página, sino que se hace una referencia a la ubicación de dicho elemento mediante texto. De este modo, la página web contiene solamente texto mientras que recae en el navegador web (interpretador del código) la tarea de unir todos los elementos y visualizar la página final. Al ser un estándar, HTML busca ser un lenguaje que permita que cualquier página web escrita en una determinada versión, pueda ser interpretada de la misma forma (estándar) por cualquier navegador web actualizado.

¿Cuáles son las características principales del HTML? HTML es un lenguaje de marcado que nos permite indicar la estructura de nuestro documento mediante etiquetas. Este lenguaje nos ofrece una gran adaptabilidad, una estructuración lógica y es fácil de interpretar tanto por humanos como por máquinas.

Sin embargo, a lo largo de sus diferentes versiones, se han incorporado y suprimido diversas características, con el fin de hacerlo más eficiente y facilitar el desarrollo de páginas web compatibles con distintos navegadores y plataformas (PC de escritorio, portátiles, teléfonos inteligentes, tabletas, etc.) No obstante, para interpretar correctamente una nueva versión de HTML, los desarrolladores de navegadores web deben incorporar estos cambios y el usuario debe ser capaz de usar la nueva versión del navegador con los cambios incorporados. Normalmente los cambios son aplicados mediante parches de actualización automática (Firefox, Chrome) u ofreciendo una nueva versión del navegador con todos los cambios incorporados, en un sitio web de descarga oficial (Internet Explorer). Por lo que un navegador desactualizado no será capaz de interpretar correctamente una página web escrita en una versión de HTML superior a la que pueda interpretar, lo que obliga muchas veces a los desarrolladores a aplicar técnicas y cambios que permitan corregir problemas de visualización e incluso de interpretación de código HTML. Así mismo, las páginas escritas en una versión anterior de HTML deberían ser actualizadas o reescritas, lo que no siempre se cumple. Es por ello que ciertos navegadores todavía mantienen la capacidad de interpretar páginas web de versiones HTML anteriores. Por estas razones, todavía existen diferencias entre distintos navegadores y versiones al interpretar una misma página web.

CSS (siglas en inglés de **C**ascading **S**tyle **S**heets), en español «Hojas de estilo en cascada», es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado.[2] Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en HTML o XHTML; el lenguaje puede ser aplicado a cualquier documento XML, incluyendo XHTML, SVG, XUL, RSS, etcétera. Te puede ayudar a crear tu propio sitio web. Junto con HTML y JavaScript, CSS es una tecnología usada por muchos sitios web para crear páginas visualmente atractivas, interfaces de usuario para aplicaciones web y GUIs para muchas aplicaciones móviles (como Firefox OS).[3]

CSS está diseñado principalmente para marcar la separación del contenido del documento y la forma de presentación de este, características tales como las capas o *layouts*, los colores y las fuentes.[4] Esta separación busca mejorar la accesibilidad del documento, proveer más flexibilidad y control en la especificación de características presentacionales, permitir que varios documentos HTML compartan un mismo estilo usando una sola hoja de estilos separada en un archivo `.css`, y reducir la complejidad y la repetición de código en la estructura del documento.

La separación del formato y el contenido hace posible presentar el mismo documento marcado en diferentes estilos para diferentes métodos de renderizado, como en pantalla, en impresión, en voz (mediante un navegador de voz o un lector de pantalla, y dispositivos táctiles basados en el sistema Braille. También se puede mostrar una página web de manera diferente dependiendo del tamaño de la pantalla o tipo de dispositivo. Los lectores pueden especificar una hoja de estilos diferente, como una hoja de estilos CSS guardado en su computadora, para sobreescribir la hoja de estilos del diseñador.

La especificación CSS describe un esquema prioritario para determinar qué reglas de estilo se aplican si más de una regla coincide para un elemento en particular. Estas reglas son aplicadas con un sistema llamado *de cascada*, de modo que las prioridades son calculadas y asignadas a las reglas, así que los resultados son predecibles.

La especificación CSS es mantenida por el World Wide Web Consortium (W3C). El MIME type `text/css` está registrado para su uso por CSS descrito en el RFC 2318[5]. El W3C proporciona una herramienta de validación de CSS gratuita para los documentos CSS.

Documentos HTML

1. Evolución de HTML hasta HTML5

HTML 1.0 (1991)

- Creado por Tim Berners Lee en el CERN para que la comunidad científica compartiera información
- Derivado de SGML
- 20 etiquetas / 13 aún perduran
- Primer navegador Nexus

HTML 2.0 (1994)

- Primera especificación oficial
- Incluía todo HTML 1.0
- Aparecen nuevos navegadores Lynx, Cello, Mosaic
- 19 nuevas etiquetas (imágenes, formularios etc..)
- <!DOCTYPE> que asocia un documento con un DTD para validarlo

HTML 3.2 (1997)

- Fue una recomendación
- 1996 Internet Explorer
- Se añaden tablas, mapas etc...
- Desaparecen etiquetas como marquee y blink (MS)
- Primera versión desarrollada enteramente por W3C
- Aparece CSS y los navegadores empiezan a adoptarlo

HTML 4.01 (1999)

- Fue una recomendación
- Ya incluye hojas de estilos
- Se quitan etiquetas de estilos de versiones anteriores
- Se mejora la presentación de fuentes, fondos y colores

XHTML 1.0 (2000)

- Nueva versión de HTML con la rigidez de XML
- Pocas etiquetas se dejan atrás y básicamente es un conjunto de reglas sobre cómo escribir adecuadamente
- Validadores

HTML 5.0 (2014)

- Fue una recomendación y las sucesivas versiones también (aún no es estándar)
- Comenzó a desarrollarse por las pocas posibilidades de aplicaciones complejas de XHTML
- Etiquetas Semánticas
- APIs
- Simplifica DOCTYPE, Link, Script
- Mejora formularios

2. Etiquetas, atributos y comentarios

Anteriormente hemos definido **HTML** como un lenguaje de marcas o etiquetas, pero, ¿qué es eso de una etiqueta? y ¿qué tipos de etiquetas tengo?

Básicamente podemos distinguir entre dos tipos de etiquetas:

- Etiquetas con contenido
- Etiquetas sin contenido

Etiquetas con contenido.

Son etiquetas que tiene tres partes (por este orden)

1. Apertura de la etiqueta
2. Contenido de la etiqueta
3. Cierre de la etiqueta

Algo así:

```
<etiqueta>
    Contenido de la etiqueta
</etiqueta>
```

HTML tiene un número limitado de etiquetas y no es necesario conocerlas todas. Con una lista más o menos pequeña podemos construir la gran mayoría de páginas web. En caso contrario, siempre podemos visitar las referencias.

Un ejemplo de etiqueta de HTML con contenido sería un párrafo:

```
<p>
    HOLA MUNDO
</p>
```

- `<p>` Sería la apertura
- **HOLA MUNDO** sería el contenido, en este caso sólo texto, aunque podríamos meter muchas cosas “dentro”.
- `</p>` Sería el cierre de la etiqueta

Etiquetas sin contenido

Son etiquetas que sólo tienen parte de apertura y carecen de contenido. Pueden estar cerradas o no, aunque yo recomiendo que se cierren. De esta manera:

- `<etiqueta >`
- `<etiqueta />`

Un ejemplo de etiqueta HTML sin contenido sería una imagen:

```
<img ...  />
```

Atributos

Las etiquetas pueden tener atributos de los que nos interesa saber lo siguiente:

- Proporcionan información adicional sobre la etiqueta.
- Las etiquetas pueden tener o no tener atributos e incluso tener más de uno.
- Siempre se añaden en la etiqueta de apertura.
- Hay atributos generales (para todas) o específicos (para algunas).
- Se representan nombre_atributo=”valor_atributo”.

Un ejemplo:

```
<img src="foto.png"  />
```

Tipos de etiquetas

Hay muchos tipos de etiquetas, pero a lo largo de este curso vamos a centrarnos en las siguientes:

- Etiquetas de cabecera
- Etiquetas de texto
- Etiquetas de imágenes, tablas, listas y enlaces
- Etiquetas multimedia
- Etiquetas semánticas
- Y alguna más pero no todas ;) .

Comentarios

Además de todas estas etiquetas nuestra página web podrá llevar comentarios que son, normalmente, texto descriptivo que no se van a mostrar en nuestra web.

Los comentarios van encerrados en esta estructura `<!-- -->` y un ejemplo sería:

```
<!-- Esto es un comentario en HTML -->
```

Guía de estilo-Buenas prácticas:

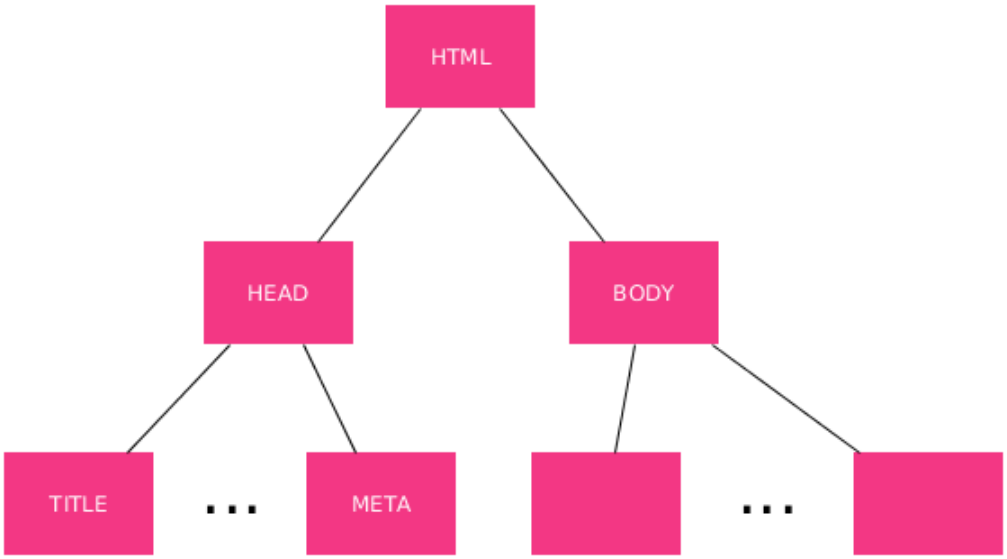
https://www.w3schools.com/html/html5_syntax.asp

3. Estructura de una página web

Una vez ya tenemos claro qué es eso de *etiqueta* con lo que yo voy a llenar mi página web el siguiente paso es tener muy claro qué estructura tiene que tener mi web.

NOTA: Recordad que **HTML define la estructura** que tiene que tener una web.

De manera general podemos representar toda página Web como un árbol *genealógico* cuya estructura común, para todas las webs, podemos decir que será la siguiente:



El Modelo de Objetos del Documento (**DOM**) es una interfaz de programación de aplicaciones (API) para documentos HTML y XML. Define la estructura lógica de los documentos y el modo en que se accede y manipula un documento. En la especificación del DOM, el término "documento" se utiliza en un sentido amplio

Esto significa lo siguiente:

- Hay una etiqueta *padre* de todo, la etiqueta **html** y entre la apertura y el cierre de esta etiqueta meteremos el resto de nuestra página.
- La etiqueta **html** tiene dos hijos, la etiqueta **head** que es la cabecera, que ya veremos qué elementos contiene, y la etiqueta **body** que es la que en realidad contiene los elementos de mi web.
- A su vez esas dos etiquetas anteriores pueden tener sus propios hijos y así sucesivamente.

Un ejemplo de esto sería:

estructura.html

```
<!DOCTYPE html>
<html lang="es">
<head>
    <!--Cabecera -->
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Mi primera página</title>
</head>
<body>
    <!-- Cuerpo -->
    <h1>HOLA MUNDO</h1>
</body>
</html>
```

4. Cabecera de una página web

La cabecera de mi página web es lo que está dentro de la etiqueta y en relación a ella es importante saber lo siguiente:

- Todo lo que va dentro de esa etiqueta no representa contenido alguno, es decir, nada de lo que yo pongo se va a ver en nuestro navegador.
- Contiene otras etiquetas que nos van a ayudar a dar una descripción de mi página.
- Puede contener enlaces a hojas de estilos y scripts.

De las etiquetas que puede contener la cabecera destacaremos las siguientes:

- **title:** que nos sirve para indicar el título de la página que es lo que se muestra en la parte de arriba del navegador.
- **style:** que es una etiqueta que me permite incluir estilos y que veremos con más detenimiento al final de este mismo curso.
- **meta:** que es una etiqueta que puede aparecer varias veces y con distintos atributos y que nos va a servir para dar una descripción de la página de diversas formas y maneras.
- **link:** para enlazar con archivos externos, normalmente hojas de estilos.
- **base:** para indicar la ruta base para construir el resto de las rutas a archivos en mi página web.
- **script:** para incluir normalmente ficheros javascript que doten de vida o animación a mi web.

La etiqueta meta

Es una etiqueta muy importante ya que, aunque no muestre nada, describe mi web y es, entre otras cosas, lo primero que leen tanto los navegadores como los buscadores.

Puede tener varias posibilidades, dependiendo de los atributos que lleve.

- `<meta charset="utf-8">` Es lo que pondremos siempre para no tener problemas con caracteres “extraños” (acentos, ñ etc…) u otros tipos de alfabetos.

  charset_example.html **(**comentar la etiqueta y verlo en Crome y Firefox)

- `<meta name="description" content="......">` Es lo que incluiremos para añadir una descripción sobre lo que es mi web. Ese texto descriptivo irá en el atributo *content*.

  ```
  <meta name="description" content="Web sobre diseño de interfaces web"/>
  ```

- `<meta name="keywords" content="...">` Es lo que incluiremos para categorizar el contenido de mi web. Por ejemplo: *content="programacion,html,meta"*

```
<meta name="keywords" content="html,css,javascript,bootstrap,jquery"/>
```

- `<meta name="author" content="yo">`

```
<meta name="author" content="Isabel Cayuela Pérez"/>
```

- `<meta name="viewport" content="width=device-width, initial-scale=1.0">` que es nuevo en HTML5 y que me asegura que mi web escalará para adaptarse a la pantalla del dispositivo. Esto, no obstante, no asegura que se vea bien.
- `<meta http-equiv="refresh" content="X">` que refrescará la web cada X segundos. Puede ser interesante para mostrar actualizaciones automáticamente en páginas que cambian, por ejemplo, las de resultados deportivos.

Añadiendo contenido. Etiquetas Básicas

1. Etiquetas relacionadas con texto

Ejemplo: texto.html
https://www.w3schools.com/html/default.asp

Vamos a empezar a añadir elementos a nuestra página web que, es lo mismo que decir, que vamos a empezar a añadir etiquetas dentro de la etiqueta.

Hay muchas etiquetas para añadir texto y nosotros vamos a presentarlas siguiendo la siguiente clasificación:

- Encabezados
- Etiquetas de formato
- Otras etiquetas interesantes

Encabezados

Son etiquetas que nos van a permitir añadir "títulos" o encabezados a distintas secciones de nuestra página. Estas etiquetas tienen el siguiente formato:

```
<hx>
   ...
   Contenido o texto
   ...
</hx>
```

x deberá ser sustituido por un número del 1 al 6, desde 1 que es el mayor tamaño hasta 6 que es el más pequeño. El texto se mostrará en negrita.

Veámos un ejemplo simple:

```
<h1>Hola</h1>
<h2>Hola</h2>
<h3>Hola</h3>
<h4>Hola</h4>
<h5>Hola</h5>
<h6>Hola</h6>
```

Etiquetas de formato

Hay multitud, todas le dan cierto estilo al texto que contienen y destacaremos las siguientes:

- **...** para poner un texto en negrita.
- **...** para poner un texto en cursiva.
- **~~...~~** para mostrar un texto tachado.
- **...** para poner un texto con énfasis (similar a cursiva).
- **^{...}** para poner un texto como superíndice de otro texto.
- **_{...}** para poner un texto como subíndice de otro texto.
- **<mark>...</mark>** para poner un texto subrayado (nuevo en html5).
- **<q>...</q>** para mostrar una pequeña cita.

- **<cite>...</cite>** para mostrar el título de una referencia bibliográfica.
- **<time>...</time>** para mostrar horas.
- **<address>...</address>** para mostrar direcciones.
- **<blockquote>...</blockquote>** para poner citas largas.

Otras etiquetas interesantes

- **
** Salto de línea.
- ** ... ** Párrafo.
- **<hr/>** Separación de Tema.

2. Imágenes, enlaces y rutas

Ej: imagenes_enlaces_rutas.html

Tras añadir elementos de texto a nuestra página, al cuerpo de la misma, vamos a continuar mejorando y haciendo nuestras webs más bonitas e interactivas.

Para ello, en este apartado vamos a añadir dos elementos básicos en toda web:

- Las imágenes
- Los enlaces

Y, además, explicaremos qué es una ruta y las clases que hay ya que, estos dos elementos anteriores, van a utilizar rutas en sus atributos.

Imágenes simples

La inclusión de imágenes simples se viene haciendo desde las primeras versiones de HTML con la etiqueta sin contendio ****

Los atributos más comunes que le podemos poner a esta etiqueta son:

- **src:** que indica la ruta en la que se encuentra el archivo de la imagen.
- **alt:** un texto alternativo para describir la imagen en caso de que no se cargue o para dispositivos especiales para usuarios con discapacidad visual (por ejemplo)
- **width:** para especificar la anchura de la imagen (px, % etc..). Si no se escogerá la anchura propia de la misma.
- **height:** para especificar la altura de la imagen (px, % etc..). Si no se escogerá la altura propia de la misma.

Es importante destacar que la imagen es un elemento en línea que se pone, si cabe, inmediatamente después de los elementos previamente añadidos.

Rutas

El concepto de ruta es un concepto muy importante ya que se utiliza es muchos temas relacionados con la informática y en concreto, en la creación de páginas WEB, se utiliza para referenciar archivos, recursos y/o partes de alguna web. De manera general podemos distinguir:

- **Relativas:** Toman como base el directorio en el que se encuentra nuestro fichero. Son las recomendadas.

- **Absolutas:** Toman como base el directorio raíz de mi equipo (/ en Linux y c:\ en Windows). Cuidado, sólo funcionarán en tu mismo equipo.

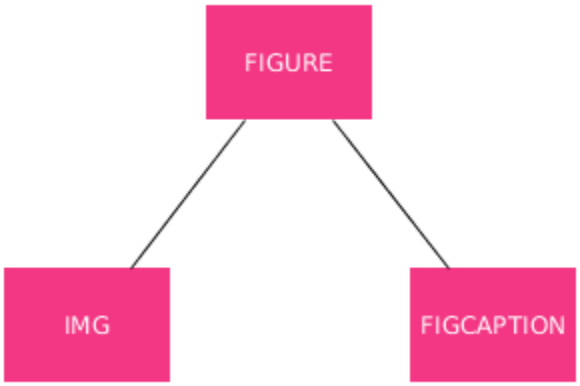
- **Url:** La dirección de Internet de un recurso (fichero, imagen etc..). Puede desaparecer y entonces dejará de mostrarse en nuestra web.

Figuras

Una novedad en HTML5 es la construcción de etiquetas, que nos va a permitir mostrar una imagen con un texto asociado.

El Modelo de Objetos del Documento (DOM) es una interfaz de programación de aplicaciones (API) para documentos HTML y XML. Define la estructura lógica de los documentos y el modo en que se accede y manipula un documento.

En este caso el árbol DOM será el siguiente:



Donde:

- **<figure>** es la etiqueta padre.
- **** es una etiqueta de imagen que hemos visto anteriormente.
- **<figcaption>** es una etiqueta que contendrá el texto asociado a la imagen.

Enlaces

Los enlaces, que se representan mediante el uso de la etiqueta **<a>** es una de las contrucciones más importantes en HTML. Esta etiqueta puede tener varios atributos, de lo cuales los más importante son:

- **href:** que es la dirección de Internet de destino (ya sea otra página web, un imagen, un fichero o lo que sea).
- **target:** que indica dónde voy a abrir ese enlace. Si no pongo nada se abrirá en la misma pantalla y si le doy el valor **target="_blank"** se abrirá en una nueva ventana de mi navegador.

Varios ejemplos de enlaces:

Enlaces dentro de la misma página

Puedo enlazar enlaces dentro de la misma página con construcciones como la siguiente:

```
...
<a href="#contacto">Contacto</a>
...
<section id="contacto">
    .....
</section>
....
```

3. Práctica: Imágenes y enlaces.

HTML5yCSS3_Practica1a

4. Listas

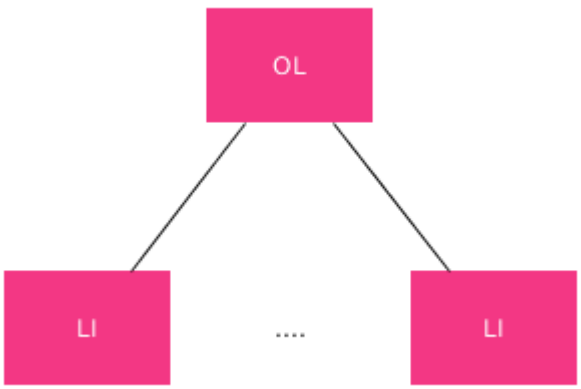
Ej: listas.html

Las listas son una de las construcciones más usadas a la hora de elaborar textos, no sólo en HTML. No obstant e, en HTML están también presentes y pueden ser de 3 tipos:

- ****Listas Numeradas:**** que son aquellas que expresan un orden entre los diferentes elementos de la lista. Est e orden podrá ser numérico, alfabético etc..
- ****Listas no numeradas:**** que simplemente muestra los elementos de la lista uno tras otro.
- ****Listas de definición:**** que muestran diversos términos junto con su definición.

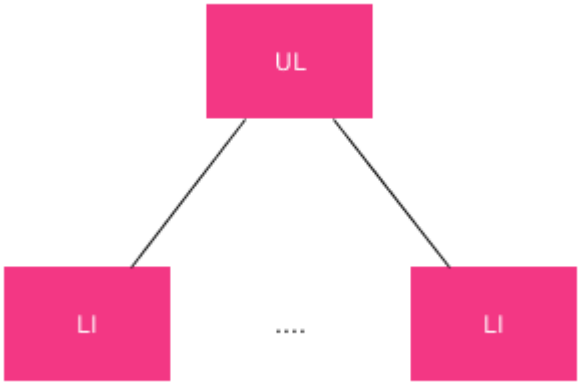
A continuación, vamos a ver el árbol DOM para cada una de estas variedades:

Lista Numeradas



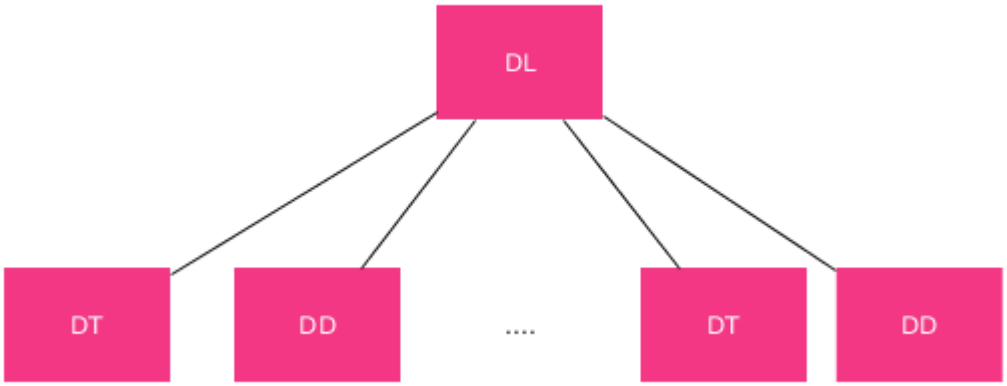
****ol**** sería la etiqueta padre y cada uno de los elemento de la lista iría en una etiqueta ****li****.

Listas no Numeradas



ul sería la etiqueta padre y cada uno de los elementos de la lista iría en una etiqueta **li**.

Listas de definición



Dl sería la etiqueta padre y cada término se define mostrando consecutivamente las etiquetas **dt**, que se corresponde con el término que vamos a definir, y **dd** que es la definición del término anterior.

5. Práctica: Listas

HTML5yCSS3_Practica1b

6. Tablas

Ej: tablas.html

Otras de las construcciones básicas en HTML son las tablas.

Además de para la presentación de elementos se usaron históricamente para dar estructura a las páginas. No obstante, ya no se maqueta con tablas.

A pesar de eso siguen siendo un elemento importante y a continuación vamos a presentar varias formas de hacer tablas.

Tablas Simples

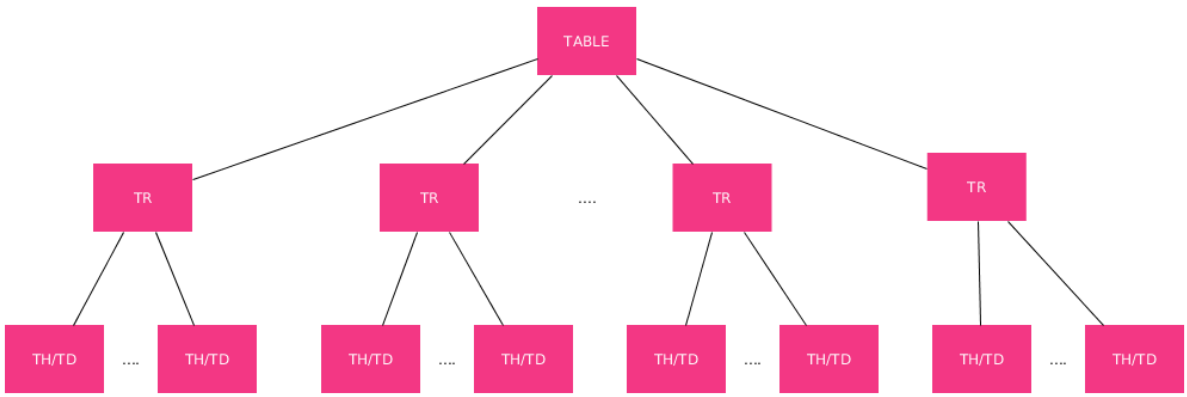
La estructura del árbol DOM más simple para una tabla sería similar a la siguiente:

- Una etiqueta **<table>** que contiene toda la tabla.
- Como hijos directos, tantas etiquetas **<tr>** como fijas queramos que tenga nuestra tabla.
- Dentro de cada fila tantas celdas **<th>/<td>** como queramos que tengan nuestras filas. La única diferencia entre estas dos es que el contenido en la segunda se presenta centrado y el texto en negrita.

NOTA: Si no coinciden el número de celdas en todas las filas veremos que suceden cosas "extrañas".

NOTA: La anchura de las celdas de una misma columna será la anchura de la más ancha de la columna.

NOTA: La altura de las celdas de una misma fija será la altura de la más alta de la fila.




```
<table>
    <tr>
        <th>Nombre</th>
        <th>Apellidos</th>
        <th>Dirección</th>
    </tr>
    <tr>
        <td>Pepe</td>
        <td>Pérez</td>
        <td>Aquí</td>
    </tr>
    <tr>
        <td>Manuel</td>
        <td>López</td>
        <td>Allí</td>
    </tr>
    <tr>
        <td>María</td>
        <td>Fernández</td>
        <td>Mas allá</td>
    </tr>
    <tr>
```

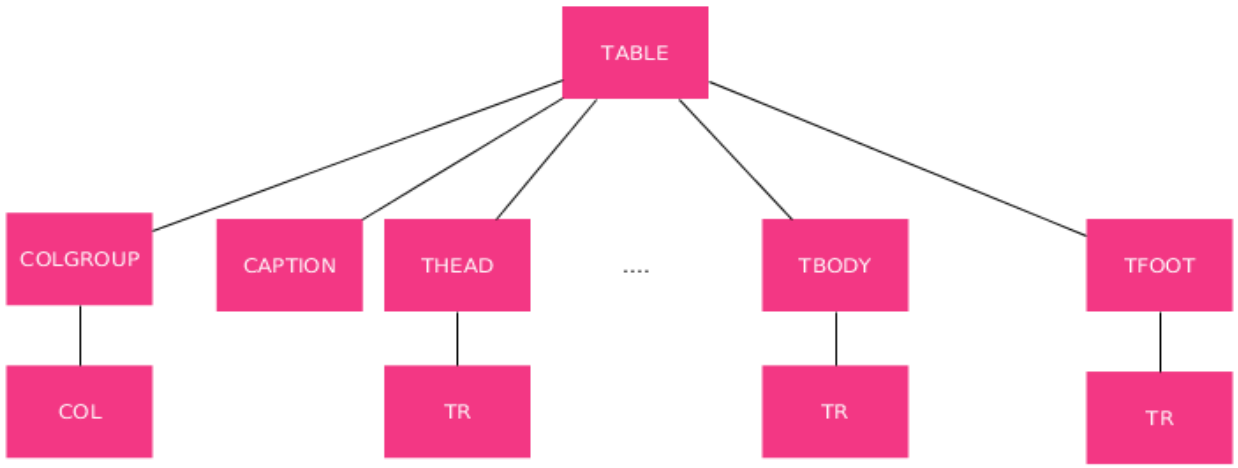
```
        <td>Sara</td>
        <td>Gallardo</td>
        <td>Mas aquí</td>
    </tr>

</table>
```

Tablas Completas

Existe además una forma más precisa y completa de construir tablas. Una forma que puede contener (aunque no es obligatorio) otras etiquetas dentro de la etiqueta raíz **<table>**. Estas nuevas etiquetas pueden ser:

- **<colgroup>** que nos permite agrupar las columnas para darles estilos. Cada uno de esos grupos lo definiremos usando una etiqueta con un atributo **span** para definir el número de columnas de cada grupo.
- **<caption>** para añadir un título o leyenda a la tabla en la parte superior.
- **<thead>** que contendrá la fila (usualmente) o filas que sean la cabecera de una tabla.
- **<tbody>** que es donde pondremos las filas que son el contenido propiamente dicho de la tabla, el cuerpo.
- **<tfoot>** que es donde pondremos las filas que son el pie de nuestra tabla.



Un ejemplo sería:

```
<table>
    <colgroup>
        <col span="3" style="background-color: grey">
        <col style="background-color: yellow">
        <col style="background-color: green">
    </colgroup>
    <caption>Alumnos matriculados</caption>
    <thead>
        <tr>
            <th>Nombre</th>
            <th>Apellidos</th>
            <th>Dirección</th>
            <th>Teléfono</th>
            <th>Email</th>
        </tr>
    </thead>
    <tbody>
```

```
        <tr>
            <td>Pepe</td>
            <td>Pérez</td>
            <td>Aquí</td>
            <td>11111111</td>
            <td>yosoy@pepe.es</td>
        </tr>
        <tr>
            <td>Manuel</td>
            <td>López</td>
            <td>Allí</td>
            <td>222222222</td>
            <td>yosoy@manuel.es</td>
        </tr>
        <tr>
            <td>María</td>
            <td>Fernández</td>
            <td>Mas allá</td>
            <td>33333333</td>
            <td>yosoy@maria.es</td>
        </tr>
        <tr>
            <td>Sara</td>
            <td>Gallardo</td>
            <td>Mas aquí</td>
            <td>44444444</td>
            <td>yosoy@sara.es</td>
        </tr>
    </tbody>
    <tfoot>
        <tr>
            <td>Pie de la tabla donde pongo más texto para que se vea como crecen</td>
        </tr>
    </tfoot>
</table>
```

Con todo lo explicado anteriormente podemos ver que las tablas que conseguimos son relativamente sencillas. En la vida real, nos encontraremos con estructuras tabulares más complejas. Éstas también se pueden construir en HTML utilizando los siguientes atributos en las etiquetas y, es decir, en las celdas.

- **rowspan:** me va a permitir que una celda ocupe más de una fila.
- **colspan:** me va a permitir que una celda ocupe más de una columna.

NOTA: Antes de escribir el código HTML de una tabla compleja es recomendable estudiar su estructura previamente. Puede parecer broma, pero yo sigo usando papel y lápiz para eso ;) (tú mismo)

Un ejemplo sería:

```
<table>
    <caption>HORARIO DE CLASE CURSO 2018-2019</caption>
    <thead>
        <tr>
            <th>HORAS</th>
            <th>Lunes</th>
            <th>Martes</th>
            <th>Miércoles</th>
            <th>Jueves</th>
            <th>Viernes</th>
```

```
        </tr>
    </thead>
    <tbody>
        <tr>
            <td>8:00</td>
            <td rowspan="2">Matemáticas</td>
            <td>Lengua</td>
            <td>Inglés</td>
            <td colspan="2">Ciencias</td>
        </tr>
        <tr>
            <td>9:00</td>
            <td>Historia</td>
            <td>Ciencias</td>
            <td>Matemáticas</td>
            <td>Lengua</td>
        </tr>
        <tr>
            <td>10:00</td>
            <th colspan="5">RECREO</th>
        </tr>
        <tr>
            <td>10:30</td>
            <td>Inglés</td>
            <td rowspan="2">Ciencias</td>
            <td>Matemáticas</td>
            <td>Lengua</td>
            <td>Historia</td>
        </tr>
        <tr>
            <td>11:30</td>
            <td>Historia</td>
            <td>Ciencias</td>
            <td>Matemáticas</td>
            <td>Inglés</td>
        </tr>
    </tbody>
</table>
```

7. Bordes de tablas

Ej:borde_tablas.html

En el apartado anterior, apartado en el que hemos presentado las tablas, no hemos hablado para nada de los bordes que normalmente pueden presentar estas estructuras.

De hecho, habéis podido ver que, por defecto, los navegadores no les ponen borde a las tablas.

Para ponerles bordes nos tenemos que adelantar un poquito en el temario y hablar de estilos, de CSS.

No es totalmente correcto lo que vamos a hacer aquí, pero nos va a servir para entender los tipos de bordes más usados (hay muchos más).

En principio lo que vamos a hacer es lo siguiente:

```
<head>
    ...
    <!--Etiqueta para definir estilos-->
```

```
    <!-- Aquí definiremos los bordes-->
    <style>
        ...
        /*Estilos para los bordes*/
        ...
    </style>

</head>
```

Y vamos a presentar tres tipos de bordes, los que considero más usados.

- **Bordes Simple**
- **Bordes sin colapsar**
- **Bordes inferior o superior**

Bordes Simples

Pondremos dentro la etiqueta **<style>** lo siguiente:

```
table {
  border-collapse: collapse;
}

td,
th {
  border: 1px solid black;
}
```

Bordes sin Colapsar

Pondremos dentro la etiqueta **<style>** lo siguiente:

```
table,
td,
th {
  border: 1px solid black;
}
```

Bordes Inferior/Superior

Pondremos dentro la etiqueta **<style>** lo siguiente:

```
table {
  border-collapse: collapse;
}

td,
th {
  border-bottom: 1px solid black;
}
```

Si lo quisiéramos superior debemos cambiar *border-bottom* por *border-top*

8. Práctica: Tablas.

HTML5yCSS3_Practica1c

Formularios en HTML

1. Formularios HTML

Ej: forms.html

https://www.w3schools.com/html/html_forms.asp

Los formularios son otros de los elementos que se han asociado desde siempre a las páginas webs. Los hemos usado tanto que es difícil calcular cuántos formularios habremos rellenado a lo largo de nuestra larga vida (al menos la mía) como usuario de la web.

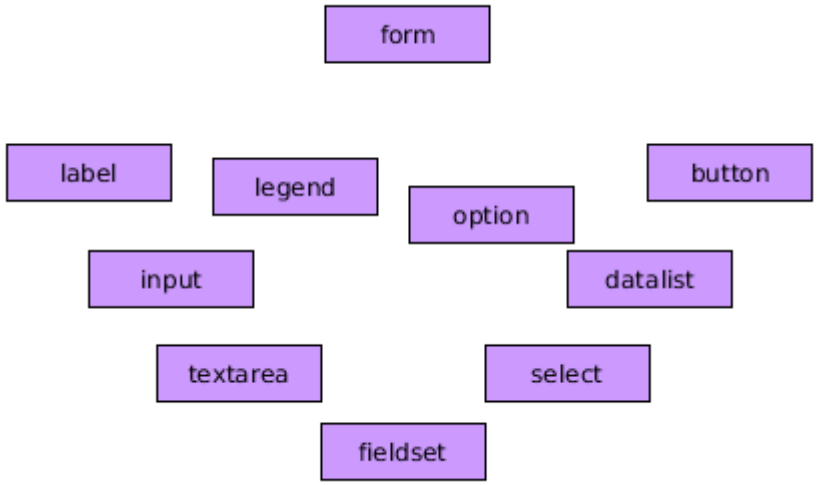
Formularios de registro, usuario y contraseña, solicitudes etc… son contextos en los que los podemos encontrar.

De un modo más formal podemos decir lo siguiente:

- Los formularios son elementos, que, como los enlaces, permiten una interacción del usuario con la página web.
- Su tarea principal es recoger información. El usuario, por tanto, debe introducir esa información en los campos del formulario.
- Una vez se ha recogido esa información el formulario la enviará al servidor, por correo o donde decida el programador para ser tratada, mostrada y/o almacenada.

Estructura del formulario

En un formulario nos podemos encontrar con muchos elementos.



El más importante de todo es el elemento padre, la etiqueta **<form>** que va a contener en su interior todos los elementos que conformen nuestros formularios.

Esta etiqueta puede tener varios atributos de entre los que destacamos:

- **method** que indica cómo se va a pasar la información al destino. Puede ser por GET (se ve la información en la barra del navegador) y por POST (no se ve y es la opción por defecto).
- **action** que indica el destino de nuestros datos. Normalmente será una URL o dirección de Internet. Existen más opciones.

A continuación, vamos a ver algunos de los elementos más frecuentes y a describir su estructura y funcionamiento.

Labels e Input

Normalmente para la recogida de información los formularios usando etiquetas **<input>**. Pero, para poder saber qué campos estamos rellenando una buena práctica (se puede de otras maneras menos elegantes) es poner una etiqueta (etiqueta)delante de cada input para dar nombre y asociar ambas.

Un ejemplo sería:

```
<label for="nombre">Nombre:</label>
<input type="text" name="nombre" />
```

Aquí estamos asociando la etiqueta al campo usando los atributos ***for*** y ***name***.

El campo que hemos usado para la recogida de información es un input. Existen muchos tipos que veremos con más detalle en el próximo apartado.

Listas desplegables

Son elementos que también hemos visto muchas veces. Al hacer click sobre ellos nos muestran varias opciones de las que podremos escoger una o varias.

Para crear listas desplegables usaremos las etiquetas **<select>** como etiqueta padre y una etiqueta **<option>** para cada una de las opciones que tengamos en la lista desplegable. Si queremos agrupar esas opciones las meteremos a su vez dentro de una etiqueta **<optgroup>**.

Un ejemplo sería:

```
<select name="provincia">
   <optgroup label="Aragon">
      <option value="Z">Zaragoza</option>
      <option>Huesca</option>
      <option selected>Teruel</option>
   </optgroup>

</select>
```

Fijaros los nuevos atributos que he metido.

Áreas de Texto

Son campos de formulario para meter textos largos. Tienen dos atributos interesantes.

- **rows:** para indicar el número de filas que tiene el campo.
- **cols:** para indicar el número de columnas que tiene el campo.

Un ejemplo sería:

```
<textarea cols="10" rows="10">
   Lorem ipsum dolor sit amet consectetur adipisicing elit. Cumque voluptate corrupti ut earum, aliquam sint amet repellendus, vitae placeat repellat quis, ipsa qui. Velit placeat reiciendis tempore autem nostrum eaque?
</textarea>
```

Botones

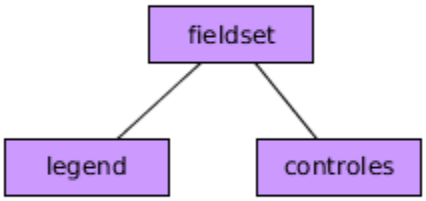
La etiqueta **<button>** no creo que se necesario explicar para que sirve…Haz click!!!

El atributo más importante que tienen es type que puede tomar tres valores:

- **button:** se comporta como un botón que puede estar presionado o no. Nada especial.
- **reset:** al hacer click borra todos los datos introducidos previamente en el formulario y pone los valores por defecto.
- **submit:** al hacer click envía los datos introducidos al destino que hemos puesto en el action del formulario.

Agrupando atributos

Hay situaciones en las que los campos de los formularios por significado, por importancia o por cualquier razón, queremos que se muestren agrupados. Para estos casos tenemos la etiqueta **<fieldset>** que la usaremos siguiente el siguiente esquema:



Siendo:

- **<legend>** una etiqueta que contiene el nombre del grupo y que se mostrará en la parte de arriba.
- **controles** todos los campos que queramos meter.

NOTA: la etiqueta tiene un borde por defecto

Un ejemplo sería:

```
<fieldset>
   <legend>Datos Personales</legend>
   <p>
      <label for="nombre">Nombre:</label>
      <input type="text" name="nombre" placeholder="Inserta Nombre" required />
   </p>
   <p>
      <label for="apellidos">Apellidos:</label>
      <input type="text" name="apellidos" value="Pérez" readonly />
   </p>

   <p>
      <label for="provincia">Provincia</label>
      <select name="provincia">
         <optgroup label="Aragon">
            <option value="Z">Zaragoza</option>
            <option>Huesca</option>
            <option selected>Teruel</option>
         </optgroup>

      </select>
   </p>
```

```
  <p>
    <input type="submit" value="Enviar" />
  </p>
</fieldset>
```

Atributos interesantes de los elementos de los formularios

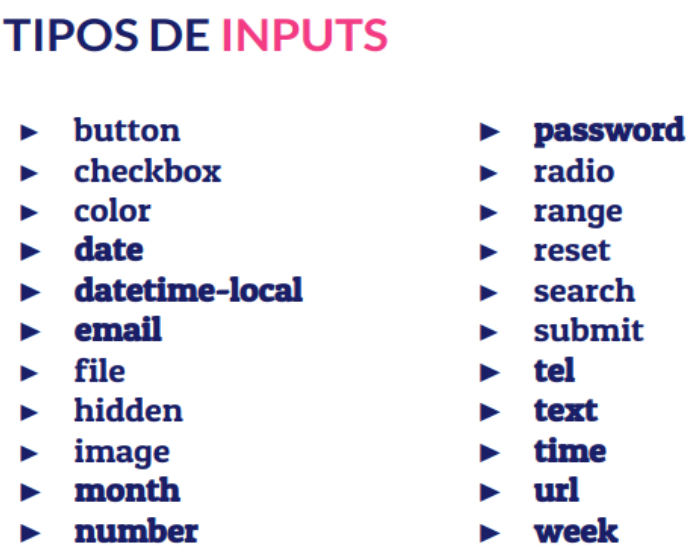
Además, existen una serie de atributos interesantes que podemos añadir a los controles o elementos de los formularios:

- **required** que indica que un control es obligatorio de rellenar.
- **disabled** que mostrará el elemento como deshabilitado.
- **readonly** que nos dice que el campo es de sólo lectura.
- **placeholder** que muestra un mensaje en los inputs dando pistas de lo que debemos poner. Al empezar a rellenarlo desaparece.
- **selected** indica la opción seleccionada.
- **value** el valor que tiene el elemento.
- **tabindex** para asegurarnos que podemos navegar por los elementos usando el tabulador podemos establecer un orden para cada uno.
- **multiple** para seleccionar varias opciones en una lista desplegable.
- Y muchos más…

2. Tipos de inputs en formularios

Ej: inputs.html

La lista de tipos (type) de Inputs que podemos tener es muy larga. La podemos ver en la siguiente imagen:



Muchos de ellos son nuevos en HTML5 por lo que es importante comprobar cómo se ven en los distintos navegadores o comprobarlo de manera previa usando la página HTML5 Test.

Tienen especial relevancia los siguientes:

- Radio Groups
- Checkboxes
- Datalist

Radio Group

Es una agrupación de inputs que presenta opciones que queremos que sean excluyentes:

Un ejemplo sería:

```
<label for="genero">Sexo</label>
<input type="radio" name="genero" value="masculino" />Hombre<br />
<input type="radio" name="genero" value="femenino" checked />Mujer<br />
```

Para que sean excluyentes deben de tener el mismo valor para el atributo ***name*** y el ***type="radio"***

CheckBoxes

Es una agrupación de opciones que presenta opciones de las cuáles podemos elegir una o varias.

Un ejemplo sería:

```
<label for="dispositivos">Dispositivos electrónicos</label><br />
<input type="checkbox" name="dispositivos" value="pc" />PC<br />
<input type="checkbox" name="dispositivos" value="table" />Tableta<br />
<input type="checkbox" name="dispositivos" value="movil" />Móvil
```

Fijaros que para agruparlos deben de tener el mismo valor para **name** y el ***type="checkbox"***

Data Lista

Es una nueva forma en HTML5 de hacer una lista de valores posibles para un input.

Un ejemplo sería:

```
<input list="editor">

<datalist id="editor">
    <option value="Atom">
    <option value="NotePad++">
    <option value="VsCode">
    <option value="Sublime">
    <option value="Brackets">
</datalist>
```

Con el atributo **list** indicamos la lista de opciones que vamos a tener y en la etiqueta metemos las que queramos.

Es importante destacar que podríamos meter otras opciones en el Input pero al usar esta estructura se nos ayuda a rellenarlo y a elegir la opción correcta.

3. Práctica: Formularios

HTML5yCSS3_Practica2a

Mas etiquetas accesibilidad

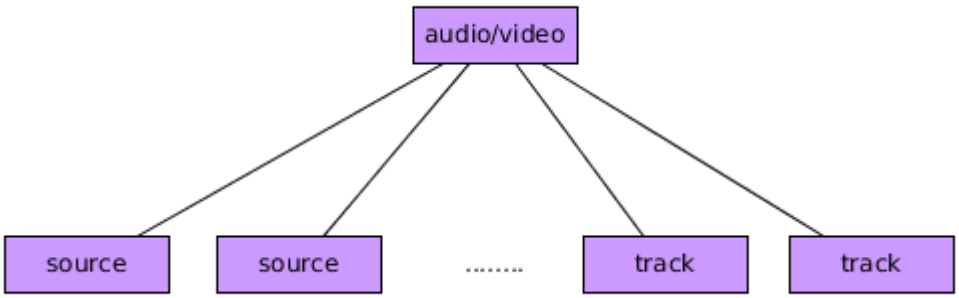
1. Etiquetas multimedia

Ej.: multimedia.html, audio.html, video.html

https://www.w3schools.com/html/html_media.asp

Una de las novedades más esperadas en HTML5 es la aparición de etiquetas específicas para multimedia.

Las más importantes son las etiquetas `<audio>` y `<video>` y para representarlas vamos optar por usar, de manera general estructuras de árbol similares a la siguientes (hay otros árboles válidos, pero este es más fácil y general):



- La etiqueta raíz será `<audio>` o `<video>` según lo que queramos mostrar en nuestro navegador.
- Dentro de esta etiqueta raíz tendremos tantas etiquetas `<source>` como formatos distintos de ese mismo elemento ofrezcamos. Es importante asegurarnos de que el formato del fichero multimedia es soportado por los navegadores. Si ofertamos varias fuentes el navegador decidirá cuál reproduce.
- Opcionalmente podremos añadir descripción de audio, subtítulos o similares usando una o varias etiquetas `<track>`.

Las etiquetas `<audio>` y `<video>` pueden tener los siguientes atributos.

- **controls** para mostrar los controles gráficos de reproducción (play, stop, pause, volumen).
- **autoplay** para que se empiece a reproducir al cargar la página de manera automática.
- **loop** si queremos que se reproduzca en bucle de manera infinita.
- **muted** si queremos que se reproduzca en silencio.

Adicionalmente la etiqueta `<video>` puede tener más atributos:

- `width` y `height` para indicar la anchura y altura del vídeo.
- **poster** para mostrar una imagen mientras el vídeo se carga.

Etiqueta <source>

Tiene dos atributos fundamentales:

- **src** para indicar dónde está el fichero multimedia (una ruta).
- **type** para indicar el tipo de fichero, por ejemplo: `_type="audio/mp3"_ o _type="video/webm"_`.

Etiqueta <track>

Tiene 5 atributos fundamentales:

- **src** para indicar dónde está el fichero de descripción (una ruta).
- **kind** para indicar si es subtítulo, descripción etc.. Tiene diversos valores que se pueden consultar en la documentación.
- **scrlang** que indica el código del idioma (es, en, fr…) de la descripción. El navegador elegirá uno u otro dependiendo de la configuración el usuario.
- **label** la etiqueta que describe el idioma (español, inglés etc..).
- **type** el tipo de fichero. Hay diversos, aunque lo más normal es *type="text/vtt"*.

2. Práctica multimedia

HTML5yCSS3_Practica2b

3. Etiquetas semánticas

Ej.: elpais.com, as.com – inspeccionar

semanticas2.html

https://www.w3schools.com/html/html5_semantic_elements.asp

Las etiquetas semánticas son nuevas etiquetas, que aparecen con la versión HTML5, y que sirven para que desarrolladores, navegadores y buscadores entiendan mejor qué tipo de contenido tienen ciertas secciones o partes de nuestra web.

Las etiquetas semánticas que existen son las siguientes (su significado es autoexplicativo si traduces del inglés):

- **header**
- **footer**
- **main**
- **section**
- **article**
- **aside**
- **details**
- **dialog**
- **summary**
- **nav**

Cuando usamos alguna de estas etiquetas sucede lo siguiente:

- A nivel visual se comportan como una etiqueta `<div>`.
- Aportan significado, podemos conocer el qué es lo que hay dentro (no como un div). Esto ayuda a buscadores y desarrolladores tal y como se dijo antes.
- Se utilizan para maquetar, como los divs.
- Tienen reglas de anidación y éstas son todos distintas. Por ejemplo: *Sólo puede tener un main y no puede ser hijo de article, aside, footer, header o nav*

Este tipo de etiquetas que componen la web semántica nos sirven para que cualquier mecanismo automático (un navegador, un motor de búsqueda, un lector de *feeds*...) que lea un sitio web sepa con exactitud qué partes de su contenido corresponden a cada una de las partes típicas de un sitio.

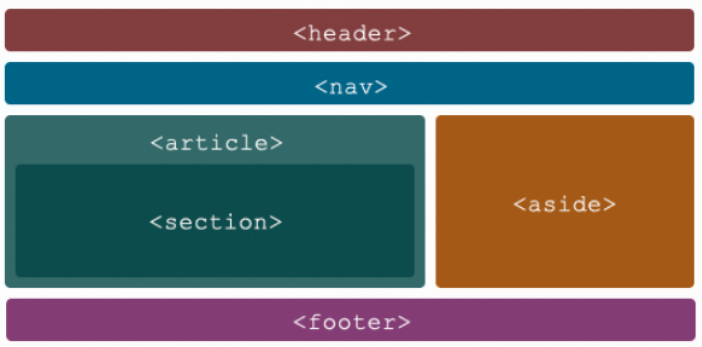
Observando esas etiquetas semánticas estructurales, cualquier sistema podrá procesar la página y saber cómo está estructurada. Veamos algunas de estas etiquetas que introduce HTML5 en este sentido.

- <section></section>: se utiliza para representar una sección "general" dentro de un documento o aplicación, como un capítulo de un libro. Puede contener subsecciones y si lo acompañamos de h1-h6 podemos estructurar mejor toda la página creando jerarquías del contenido, algo muy favorable para el buen posicionamiento web.
- <article></article>: representa un componente de una página que consiste en una composición autónoma en un documento, página, aplicación, o sitio web con la intención de que pueda ser reutilizado y repetido. Podría utilizarse en los artículos de los foros, una revista o el artículo de periódico, una entrada de un blog, un comentario escrito por un usuario, un widget interactivo o cualquier otro artículo independiente de contenido. Cuando los elementos de <article> son anidados, los elementos interiores representan los artículos que en principio son relacionados con el contenido del artículo externo. Por ejemplo, un artículo de un blog que permite comentarios de usuario, dichos comentarios se podrían representar con <article>.
- <aside></aside>: representa una sección de la página que abarca un contenido relacionado con el contenido que lo rodea, por lo que se le puede considerar un contenido independiente. Este elemento puede utilizarse para efectos tipográficos, barras laterales, elementos publicitarios, para grupos de elementos de la navegación, u otro contenido que se considere separado del contenido principal de la página.
- <header></header>: representa un grupo de artículos introductorios o de navegación. Está destinado a contener por lo general la cabecera de la sección (un elemento h1-h6 o un elemento hgroup), pero no es necesario.
- <nav></nav>: representa una sección de una página que enlaza a otras páginas o a otras partes dentro de la página. No todos los grupos de enlaces en una página necesita estar en un elemento nav, sólo las secciones que constan de bloques de navegación principales son apropiados para el elemento de navegación.
- <footer></footer>: representa el pie de una sección, con información acerca de la página/sección que poco tiene que ver con el contenido de la página, como el autor, el copyright o el año.
- <hgroup></hgroup>: representa el encabezado de una sección. El elemento se utiliza para agrupar un conjunto de elementos h1-h6 cuando el título tiene varios niveles, tales como subtítulos o títulos alternativos.
- <time>: representa o bien una hora (en formato de 24 horas), o una fecha precisa en el calendario gregoriano (en formato ISO), opcionalmente con un tiempo y un desplazamiento de zona horaria.

ESTRUCTURA TRADICIONAL DE UN DOCUMENTO EN HTML 4

tradicional.html

ESTRUCTURA TRADICIONAL DE UN DOCUMENTO EN HTML5



semanticas.html

4. Otras etiquetas

Ej.: otras.html

Estamos llegando al final de lo que queríamos saber sobre HTML. Recordaros que este es un tema de **INTRODUCCIÓN** que pretende ser directo y práctico.

HTML5 tiene muchas más cosas y muchas más etiquetas que no hemos visto. Algunas se escapan al contenido por su complejidad y otras no se van a incluir por mantener las cosas simples.

Hay que conocer la base y saber usar las referencias, esto es fundamental en la informática de hoy en día. Conforme vayamos practicando iremos abandonando las referencias las cuáles sólo visitaremos para consultas muy puntuales.

No obstante, antes de *"cerrar"* la lista de etiquetas vamos a comentar de manera rápida algunas más que puede ser interesantes. No las vamos a ver en detalle, pero pondremos un ejemplo para cada una de ellas.

- `<iframe>` nos permite mostrar páginas web dentro de otras páginas.

```
<!-- Etiqueta iFrame -->
<iframe src="http://www.as.com" height="300px" width="600px"></iframe>
```

- `<canvas>` nos permite dibujar, animar y un mundo infinito de posibilidades. Es la base de todos los juegos HTML5, desde el más sencillo al más complejo.
- `<div>` es una etiqueta para separar las secciones de los documentos y que se utiliza para maquetar. Es el elemento en **bloque** por excelencia.

```
<div>
  Some text inside a div
</div>
```

- `<span>` es un elemento en **línea** que no aporta nada visualmente pero que a nivel lógico me sirve para, por ejemplo, poder distinguir ciertas partes dentro de un texto.

```
and some text with <span>span</span>
```

- `<script>` nos sirve para introducir código escrito en otros lenguajes, típicamente en javascript.

```
<script>
    var nombre = prompt("Hola como te llamas?");
    alert("Encantado " + nombre);
</script>
```

- **<object>** y **<embed>** sirven para mostrar documentos externos o de terceros en la pantalla del navegador. Su uso y significado es similar pero el segundo ha quedado *deprecated* lo que significa que puede llegar algún momento en el que los navegadores no lo entiendan.

```
<!-- Adding external objects with object -->
    <!-- docs, videos, svg, depende de plugins de terceros-->
    <object type="application/pdf" data="https://www.um.es/atica/documentos/html.pdf" width="300"
height="200">
    </object>

    <embed  type="application/pdf"  src="https://www.um.es/atica/documentos/html.pdf"  width="300"
height="200" />
```

5. Accesibilidad en HTML

"Hacer nuestra web accesible es construirla teniendo siempre presente que puede ser visitada por todo tipo de aplicaciones y usuarios, incluso por aquellos con alguna discapacidad."

Algunas reglas generales

- Utiliza **etiquetas semánticas** en vez de que no aportan significado.
- Usa el atributo ***for*** para las etiquetas en los formularios (los pone en relación con el campo correcto)
- Añade atributos ***alt*** a las imágenes
- Especifica el **idioma** de la página
- Utiliza las **cabeceras** correctamente para que los lectores "conozcan" la estructura de tu página

Atributos ARIA

https://developers.google.com/web/fundamentals/accessibility/semantics-aria/aria-labels-and-relationships?hl=es

"Los atributos especiales **ARIA (**Accesible Rich Internet Application**)** son atributos que se añade a las etiquetas HTML para que éstas sean **entendidas mejor** por los lectores que usan aquellas personas con alguna discapacidad."

ALGUNAS REGLAS GENERALES

- role (alert,tab,document, navigation...)
- aria-owns (expresa relación jerárquica mediante el id de otro elemento)
- aria-hidden (oculta partes innecesarias)
- aria-live
- aria-checked
- aria-label
- aria-disabled
- aria-valuenow, aria-valuemax,aria-valuemin

CSS

1. Evolución de CSS hasta CSS3

CSS es un lenguaje de diseño gráfico para definir y crear la presentación de un documento estructurado escrito en un lenguaje de marcado. Es muy usado para establecer el diseño visual de los documentos web, e interfaces de usuario escritas en HTML...

LA VERSIÓN 1.0 (1996)

- Por Hakon Wium Lie y Bert Bos
- Para poner orden al desorden que había a la hora de dar estilos en los diferentes navegadores.
- Propiedades para el tipo de letra
- Colores de texto y de fondo
- Alineación de texto, imágenes y tablas
- Margen, borde, padding y posicionamiento de los elementos

LA VERSIÓN 2.0 (1998)

- Mejoras en el posicionamiento de elementos
- Capas con z-index
- Sombras
- Dirección del texto
- ….

LA VERSIÓN 2.1 (2011)

- Muchos borradores y versiones desde el 2004
- Corrige errores de la versión anterior
- Suprime elementos que no fueron tenido en cuenta por los navegadores
- Añade las extensiones de los navegadores
- ….

LA VERSIÓN 3

- Es modular y ha evolucionado desde 1999
- Cada módulo tiene un grado de madurez o estandarización propio. No habrá CSS4 única.
- Esquinas redondeadas
- Gradientes
- Transiciones y animaciones
- Nuevos layouts (Flex y Grid)
- Media-Queries (diseño responsive)

2. Añadir hojas de estilo a nuestro HTML

Ej.:estilos, ejemplo2

En apartados anteriores de este mismo curso hemos ido, de manera esporádica, añadiendo estilos en algunos ejercicios.

Ahora vamos a ver existen otras formas de añadir estos estilos y vamos a explicar qué formas son más recomendables y porqué.

De manera general podremos añadir CSS a nuestro HTML de cuatro formas:

- **Estilos In-line**
- **Etiqueta Style**
- **Directiva `@import`**
- **Etiqueta link**

Estilos In-line

La inclusión de estilos In-line consiste en la inclusión de estilos visuales usando el atributo **style** de las etiquetas.

Este atributo es una atributo general que está presente en todas aquellas etiquetas que sirven para describir el contenido de la página.

Un ejemplo de estilos In-Line sería:

```
<h1 style="color:red">
...
<h1>
...
<h2 style="color:red">
...
<h2>
```

Es directo, pero si quieres realizar un cambio en todos los elementos que tienen ese color de letra en éste y en todos los demás ficheros de mi sitio tendrías un montón de trabajo.

Etiqueta Style

La etiqueta `<style>` me permite definir todos los estilos de una misma página web en un único sitio que, normalmente, suele estar en la cabecera.

Un ejemplo del uso de esta etiqueta sería:

```
<style>
 h2 {
     color: green;
 }
</style>
```

Si hubiera algún cambio en el estilo de mi sitio sólo tendría que hacerlo una vez por página. Sigue siendo mucho trabajo si mi sitio web está compuesto, por ejemplo, por miles de páginas.

Directiva `@import`

Nos permite importar un fichero css directamente en nuestro HTML.

Se utiliza dentro de la etiqueta **style** de esta manera:

```
<style>
    @import 'css/import.css';
</style>
```

import.css sería el nombre de nuestro fichero de estilos pudiendo ser el que nosotros queramos.

Utilizar esta técnica me permite realizar cambios en el estilo de toda mi web únicamente tocando un fichero.

Sin embargo, hay un inconveniente. El rendimiento, ya que es bloqueante en algunos navegadores, y se para la descarga del resto de la página hasta que no se ha descargado la hoja de estilos.

Por lo tanto, hay que usar esta opción con cuidado.

Etiqueta Link

Es la opción que os recomiendo. Tendremos un único punto de cambio y a la vez no es bloqueante.

Un ejemplo de uso de esta etiqueta, que se pone dentro de la cabecera, es:

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

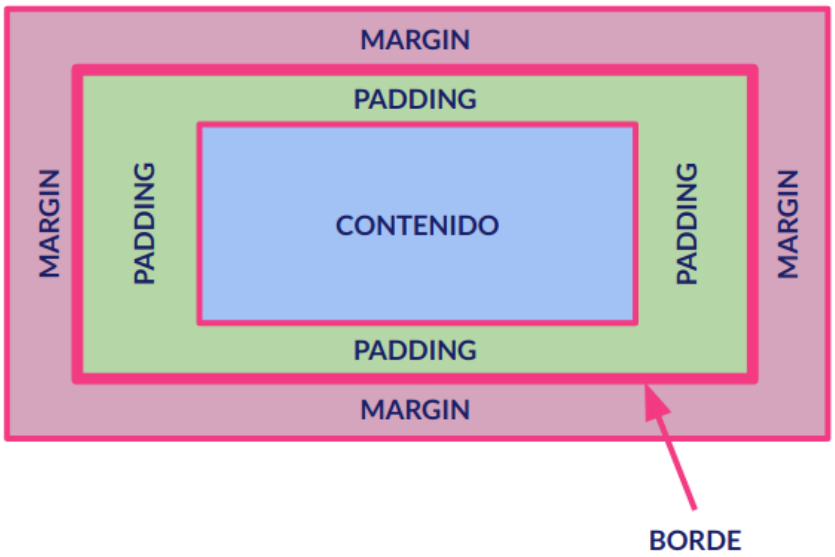
Tiene tres atributos en los que indicamos qué fichero quiero (*href*), que tipo de fichero es (*type*) y relación que existe entre el documento y el documento enlazado (*rel*)

3. El modelo de caja. Etiquetas en línea y en bloque

Ej.:modelo_de_caja

Una de los conceptos más importantes que debemos recordar a la hora de elaborar CSS es que todos y cada uno de los elementos de mi página web son cajas.

Eso exactamente, ¿qué significa? Significa que todos los elementos que representan algo en HTML tiene la siguiente estructura visual:



Con ese modelo debemos por lo tanto distinguir las siguientes zonas:

- El **contenido** que es lo “importante”, el texto en las etiquetas de texto, la imagen en las etiquetas de imagen etc.
- El **padding** que es la distancia que existe entre el contenido y el borde de la caja.

- El **borde** que es el elemento que marca la división entre el elemento y el resto de los elementos de la página.
- El **margen** que es la distancia entre el elemento y el resto de los elementos de la página.

Podemos comprobar estas propiedades, y el hecho de que por defecto los navegadores les dan valor, utilizando las herramientas para desarrolladores de los mismos.

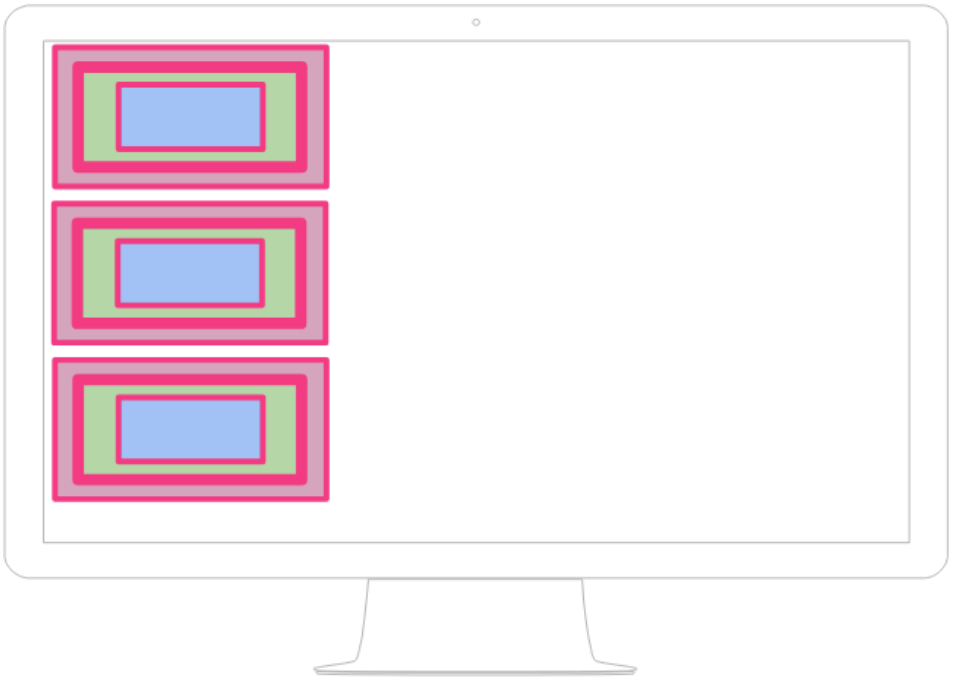
Comportamiento de las cajas

Además de conocer la estructura de las cajas CSS debemos conocer cómo se comportan estas cajas cuando las representamos en nuestro navegador.

De manera general podemos distinguir entre elementos en bloque y los elementos en línea.

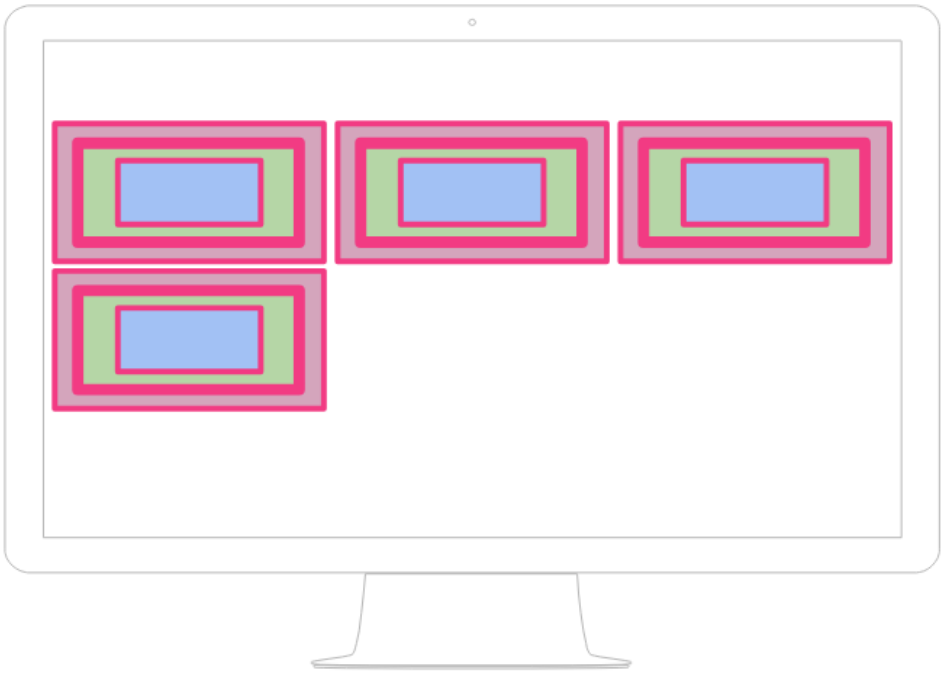
Los **elementos en bloque** son elementos que, independientemente de la anchura que tengan, se separan verticalmente de los elementos anteriores y posteriores. Es como si *"provocaran"* un salto de línea antes y uno después (figuradamente hablando).

Representado esto gráficamente sería algo así:



En cambio, los **elemento en línea** se van sucediendo a lo largo de la misma línea, mientras caben, uno detrás de otro y de izquierda a derecha (al menos en nuestro idioma). Cuando no caben pasan a la línea siguiente. Digamos que *"fluyen"* dependiendo de la anchura de la pantalla de nuestro navegador. Este fluir es precisamente la clave a la hora de maquetar páginas web que es algo que veremos más adelante.

Representado este comportamiento gráficamente sería algo así:



NOTA: Es muy importante saber qué tipo de elemento es cada etiqueta que usemos.

4. Selectores CSS

Ej.: selectores.html

https://www.w3schools.com/css

Ya sabemos incluir estilos en nuestro HTML, ya sabemos que todos los elementos de nuestra página web son cajas y ahora vamos a dar el siguiente paso, seleccionar los elementos que queremos, de entre todos los que hay, para posteriormente darles estilos.

Para ello utilizaremos selectores CSS.

Los **selectores CSS** son **reglas** o **patrones** que nos van a permitir seleccionar los distintos elementos de mi página web para poder modificar sus propiedades o estilos.

Desde mi punto de vista dominar los selectores CSS es lo más importante. Un dominio correcto de los mismos te ahorrará trabajo.

Estos selectores es lo que vamos a incluir en nuestros archivos CSS y su sintaxis de manera general es la siguiente:

```
selector {
   prop1: valor1;
   prop2: valor2;
....
   propn: valorn;
}
```

- **selector** hace referencia a la regla o patrón mediante cuya aplicación elegiremos uno o varios elementos de mi página web.
- **propX** son las propiedades que queremos modificar en los elementos seleccionados.

- **valorX** es el valor que daremos a cada una de las propiedades modificadas.

Todo con la sintaxis que tenemos expresada en la imagen.

Algunos selectores

https://www.w3schools.com/css/css_selectors.asp

https://www.w3schools.com/css/css_combinators.asp

- ***** es el selector universal. Selecciona todos los elementos.
- **#id** selecciona el elemento que tenga ese valor (id en este ejemplo) en el atributo id.
- **.class** selecciona los elementos que tengan ese valor (class en este ejemplo) en el atributo class.
- **etiqueta** selecciona esas etiquetas concretas.
- **selector1,selector2** sirve para cambiar las propiedades de los elementos seleccionados por ambos selectores
- **selector1 selector2** sirve para cambiar las propiedades de los elementos seleccionados por selector2 que se encuentran dentro de aquellos seleccionados por selector1.
- **selector1>selector2** sirve para cambiar las propiedades de los elementos seleccionados por selector2 que son hijos directos (en el árbol DOM) de aquellos que se seleccionen con selector1.
- **selector1+selector2** sirve para cambiar las propiedades de los elementos seleccionados por selector2 que están justo después de aquellos que se seleccionan mediante selector1.
- **selector1~selector2** igual que el anterior pero justo antes.
- **[atribut expre valor]** siendo expre (=,~=,|=,$=,*..) para seleccionar elementos atendiendo al valor de sus atributos.

Un ejemplo de todos ellos podemos verlo en el siguiente HTML:

```
* {
  font-family: "Courier New", Courier, monospace;
}

#main {
  background-color: grey;
}

h1.especial {
  color: blue;
}

h2,
h3 {
  border: 1px solid orange;
}

li {
  color: red;
}

li li {
  color: green;
}

p > img {
  border: 5px solid black;
}

img[alt="segunda"] {
  border: 5px solid red;
}
```

```
p ~ h3.antes {
  background-color: pink;
}
p + h3 {
  background-color: blue;
}
```

Además, pueden combinarse de todas las maneras que se nos ocurran para realizar selecciones más complejas.

Hoja de estilos ¿en “Cascada”?

Ya desde la misma definición de CSS dijimos que la *C* hacía referencia a “en Cascada”. Pero exactamente, ¿qué significa eso?

Básicamente dos cosas:

a) Los estilos se van **propagando** hacia abajo o lo que es lo mismo si especificamos una propiedad en un elemento padre los hijos tienen el mismo valor para esas propiedades.

b) Si hay más de una regla que se puede aplicar al mismo elemento y hay **conflicto**, entonces se aplica la **regla más específica**.

5. Propiedades interesantes

Ej.: propiedades, ejemplo3, ejemplo4, ejemplo5

https://www.w3schools.com/css

Ya sabemos incluir estilos, ya sabemos sobre cajas y ya sabemos cómo seleccionar los elementos de mi página web para poder aplicarles estilos.

Ahora, en este apartado, vamos a ver que propiedades podemos modificar, pero antes, vamos a recordar la sintaxis general de los archivos CSS:

```
selector {
   prop1: valor1;
   prop2: valor2;
....
   propn: valorn;
}
```

Hay muchísimas propiedades que podemos modificar, algunas son comunes a todas las etiquetas y otros sólo se pueden modificar especificamente.

Podéis acceder a lista completa aquí:

Propiedades CSS - Lista completa

Es casi imposible conocerla entera, pero es una referencia que debemos de tener siempre presente en nuestra barra de favoritos.

Si queremos reducirla la lista a aquellas propiedades que son consideradas fundamentales o esenciales para poder empezar a aplicar estilos es mejor que nos fijemos en este enlace:

Propiedades CSS básicas

No obstante, voy a reducir aún más la lista a una serie de propiedades referentes a los siguientes aspectos:

- **Color**
- **Fondos**
- **Dimensiones y Unidades**
- **Márgenes, Bordes y Padding**
- **Texto**

Colores

Para establecer el **color del texto** de nuestra web lo podemos establecer usando la propiedad *color*. Por ejemplo:

```
/* Notación mediante colores */
p {
  color: red;
}

/* Notación hexadecimal */
h1 {
  color: #cccccc;
}

/* Notación RGB */
h3 {
   color: rgb(214, 122, 127);
}
```

Fondo

Usando CSS podemos también establecer el **fondo** de nuestros elementos. Hay diversas propiedades, las más destacadas:

- **background-color** para establecer el color de fondo.
- **background-image** para establecer una imagen de fondo.
- **background-repeat** para especificar cómo se repite la imagen de fondo. Puede tomar diversos valores.
- **background-origin** desde donde queremos que se repita la imagen.

Un ejemplo básico:

```
body {
  background: grey;
}
```

Dimensiones y unidades

Las dimensiones de los elementos de nuestra página se establecen usado las siguientes propiedades:

- **width** para la anchura de nuestro elemento.
- **height** para altura de nuestro elemento.

Y ambas podemos determinar usar varios tipos de unidades:

- **px:** En píxeles
- **%:** En relación a lo que ocupe el padre del elemento dentro del árbol DOM.
- **em:** En relación al tamaño por defecto de la letra del navegador en ese instante (normalmente 16px..)
- **rem:** En relación al tamaño por defecto de la letra que tiene la etiqueta HTML.

Un ejemplo básico:

```
#first img {
  width: 50%;
}

#second {
  width: 600px;
  ...;
}
```

Márgenes, Bordes y Paddings

Son propiedades para establecer las dimensiones de los elementos de la caja.

Para márgenes y paddings tenemos varias formas de hacerlo. Si tenemos en cuenta que A(Arriba)-D (Derecha) - AB (Abajo) - IZQ (Izquierda) (sentido de las agujas del reloj):

```
selector {
  /* A -D -AB-IZQ */
  margin: 20px 50px 20px 50px;

  /* A y AB - DCHA e IZQDA */
  margin: 20px 50px;

  /* Todos */
  margin: 50px;

  /* O de manera individual */
  margin-left: 10px;
  margin-top: 10px;
  margin-bottom: 10px;
  margin-right: 10px;
}
```

Para el padding sería exactamente lo mismo. Sólo tenemos que sustituir *margin* por *padding*.

En relación al borde de un elemento tenemos también varias posibilidades. De igual manera lo vamos a ilustrar mediante un ejemplo:

```
/* De manera general*/
border: 1px solid black;

/* Sólo el borde */
border-color: black;

/*Sólo la anchura del borde */
border-width: 1px;

/* Sólo el estilo de la línea del borde */
/* posibles valores solid, dashed, dotted */
border-style: solid;
```

Estilos para el texto

Hay multitud de propiedades para establecer los estilos del texto de mi página web. Algunas de las más destacables son:

```css
/* Para establecer el tipo de fuente */
font-family: "Times New Roman", Times, serif;

/* Para establecer el tamaño de la fuente */
font-size: 2em;

/* Para establecer el grosor del tipo de letra */
/* Posibles valores: bold, bolder, lighter */
font-weight: bold;

/* Para establecer la alineación texto */
/* Posibles valores: center, left, right, justify */
text-align: center;

/* Para establecer la decoración de texto */
/* Posibles valores: underline, overline, none, line-through */
text-decoration: underline;

/* Para establecer tabulaciones */
text-indent: 10px;

/* Para transformar un texto todo a mayúscula o minúsculas */
/* Posibles valores: uppercase, lowercase, capitalize */
text-transform: uppercase;
```

https://www.w3schools.com/css/css_font.asp

https://www.w3schools.com/css/css_icons.asp

6. Práctica: Selectores

HTML5yCSS3_Practica03

7. Pseudoselectores

Ej.: pseudoselectores_pseudoelementos

https://www.w3schools.com/css/css_pseudo_classes.asp

Además de los selectores tradicionales que hemos presentado anteriormente CSS nos proporciona unos “selectores” especiales, los llamados ***pseudoselectores*** y los ***pseudoelementos***.

Los **pseudoselectores** son palabras clave que se añaden a los selectores y que nos indican un **ESTADO** determinado de los elementos seleccionados.

Los **pseudoelementos** son palabras clave que se añaden a los selectores y que nos indican una **PARTE** de un elemento y nos permiten añadir contenido.

Pseudoselectores

Tienen la siguiente síntaxis:

```
selector:pseudoselector {
   prop1: valor1;
   prop2: valor2;
....
   propn: valorn;
}
```

Y podemos dividirlos en los de estado y los de posición

De estado

- **:link**
- **:visited**
- **:enabled**
- **:disabled**
- **:checked**
- **:required**
- **:optional**
- **:focus**
- **:hover**
- **:empty**

De posición

- **:first-child**
- **:first**
- **:first-of-type**
- **:last**
- **:last-child**
- **:last-of-type**
- **:nth-child(n)**
- **:nth-last-of-type()**

Pseudoelementos

https://www.w3schools.com/css/css_pseudo_elements.asp

Tienen la siguiente síntaxis:

```
selector::pseudoelemento {
   prop1: valor1;
   prop2: valor2;
....
   propn: valorn;
}
```

Los más destacados son:

- **::first-line**
- **::first-letter**
- **::after**
- **::before**
- **::selected**

8. Práctica: Pseudoselectores

HTML5yCSS3_Practica04

9. Estilos por defecto y reseteo de propiedades

Ej.: por_defecto.html

Una cosa que parece obvia pero que con frecuencia se olvida cuando empezamos a trabajar con HTML y CSS es el hecho de que los navegadores tienen sus propias hojas de estilos que se aplican a los distintos elementos con el objetivo de hacer las páginas sin estilos más visibles.

Estos estilos no tienen por qué coincidir de un navegador a otro y si queremos consistencia a través de los distintos navegadores se suelen usar las llamadas **hojas de RESETEO CSS**

Estas hojas sirven para eliminar ciertas características que imponen los navegadores.

Hay ciertos aspectos sobre este tema que debemos considerar:

- Este tipo de hojas de estilos para Resetear son usadas por Frameworks CSS (como BootStrap) que buscan que todas las páginas siempre luzcan igual independientemente del navegador.
- Si las usas deben ser adaptadas a cada proyecto.
- Hay que usarlas con cuidado porque podemos eliminar estilos que dábamos por sentado.
- El trabajo posterior tras usarlas es mayor pero el resultado es más personalizado.

Hay dos hojas de reseteo que son muy usadas:

Hoja de Reseteo de ERIC Meyer

Hoja de Reseteo de HTML5 Doctor

10. Prefijos específicos para navegadores

En uno de los apartados anteriores de este mismo curso hablamos de CSS3. Dijimos que era una especificación modular, extensa y que se iba implantándose poco a poco.

Los distintos navegadores conforme dan soporte a propiedades experimentales lo que hacen es añadirles un prefijo para indicar que ellos sí les han dado soporte antes de que sean soportadas por los demás o antes de que sean parte del estándar.

Estos son los llamados **prefijos de navegadores** y tienen una sintaxis similar a la siguiente:

```
a {
  -webkit-transition: -webkit-transform 1s;
  transition: -ms-transform 1s;
  transition: transform 1s;
}
```

Y los valores más comunes de esos prefijos son:

- **-webkit** para Chrome, Safari, nuevas versiones de Opera y para Firefox para iOS
- **-moz** para navegadores Firefox que no sean para iOS
- **-o** para versiones antiguas de Opera
- **-ms** para Internet Explorer y Microsoft Edge

El problema que surge es que nosotros, como desarrolladores no sabemos cuándo usarlo. CSS3 es una especificación larga con muchos módulos y es prácticamente imposible saber cuándo hay que añadir estos prefijos o cuándo las propiedades han dejado de ser experimentales.

Tenemos la suerte, no obstante, de tener varias herramientas para ello.

- ShoudIPrefix
- Autoprefixer

11. Optimización de CSS

Conforme vayamos avanzando es nuestro conocimiento y empecemos a trabajar en proyecto reales y en proyectos grandes oiremos hablar de un término como **optimización de CSS**.

Pero, ¿qué es eso realmente? Si tenemos que resumirlo de manera concisa diremos que optimizar nuestro CSS es:

- Minimizar el CSS para subirlo a producción (quitar cualquier espacio en blanco).
- Borrar reglas innecesarias y duplicadas.
- Ordenar las propiedades de nuestras reglas o selectores por orden alfabético para que todo sea más legible y fácil de mantener.
- Poner el CSS en la cabecera (aunque es eso algo sobre lo que ya hemos insistido).
- Si el CSS es muy grande lo partiremos en varios (varias etiquetas **link**).
- Organizar las reglas poniendo las que tengan relación juntas.
- Documentar tus ficheros CSS.

Para ayudaros con estas tareas existen muchas herramientas, pero yo os recomiendo estas:

- CSS Minifier
- CSS Optimizer
- Unused CSS

12. Herramientas relacionadas con CSS

De igual manera, veremos que al trabajar en proyectos grandes con CSS hay tareas que son tediosas y que favorecen la aparición de errores. Para evitar esto han ido surgiendo muchas herramientas, pero aquellas que han tenido más éxito y que deberías conocer son los preprocesdores CSS.

Un **preprocesador CSS** es un programa que te permite generar CSS…haciendo que la estructura sea más legible y más fácil de mantener.

Añaden al CSS tradicional características como:

- Las **variables**
- Los **selectores anidados**
- Los **bucles**
- Los **mixins**
- Etc.

Y los preprocesadores más conocidos son: **LESS**, **SASS** y **POSTCSS** que no es exactamente un preprocesador pero nos va a facilitar mucho el trabajo con CSS y es fácilmente extensible.

Un ejemplo sería:

```
$font-stack: Helvetica, sans-serif;
$primary-color: #333;

body {
  font: 100% $font-stack;
  color: $primary-color;
}
```

Donde estamos usando variables, la variable **$font-stack**

13. Autoría del documento

Este manual es una recopilación de documentos del profesor Juan Diego Pérez, adaptados por la profesora Isabel Cayuela Pérez.