

## 1.- Estructura de un documento HTML

HTML (*HyperText Markup Language*, Lenguaje de Marcado de HiperTexto) es un lenguaje estándar utilizado para la creación de páginas web desarrollado por W3C (*World Wide Web Consortium*).

Nos aporta una representación estructural del documento que permite la modificación de su contenido o presentación visual.

```
<body>
  <p>Esto es un párrafo que contiene un <a href="#">enlace</a> en
  medio.</p>
  <ul>
    <li>Primera entrada en la lista</li>
    <li>Segunda entrada en la lista</li>
  </ul>
</body>
```

# HTML5

Declaración. **<!DOCTYPE html>**

El elemento raíz

Contenido. **<HTML></HTML>** y dentro:

1.- Encabezado. **<HEAD></HEAD>**

Contiene información relativa a la página: título, icono de la pestaña, metadatos, estilos, etc.

2.- Cuerpo. **<BODY></BODY>**. Contiene lo que deseamos presentar por pantalla.

Tenemos un conjunto de marcas o etiquetas, con sus atributos/valores, limitados y ya definidos por la W3C para ser interpretados por los navegadores.

# HTML5

<!DOCTYPE html>

<HTML lang="es">

<HEAD>

<META charset="utf-8">

<META name="title" content="Mi primer HTML5">

<META name="description" content="Ejemplo de HTML5">

<META name="keywords" content="Tutorial, HTML5, CSS, Javascript">

<TITLE>Mi primer HTML5</TITLE>

</HEAD>

<BODY>

Cuerpo de la página

</BODY>

</HTML>

[https://www.w3schools.com/tags/ref\\_language\\_codes.asp](https://www.w3schools.com/tags/ref_language_codes.asp)

Par formado por atributo=valor

*html:5 o !*

emmet

<BODY>

HTML5

<header></header>

<nav></nav>

*header+nav+(main>section\*2>article\*2)+aside+footer*

<main>

<section>

<article></article>

<article></article>

</section>

<section>

<article></article>

<article></article>

</section>

</main>

<aside></aside>

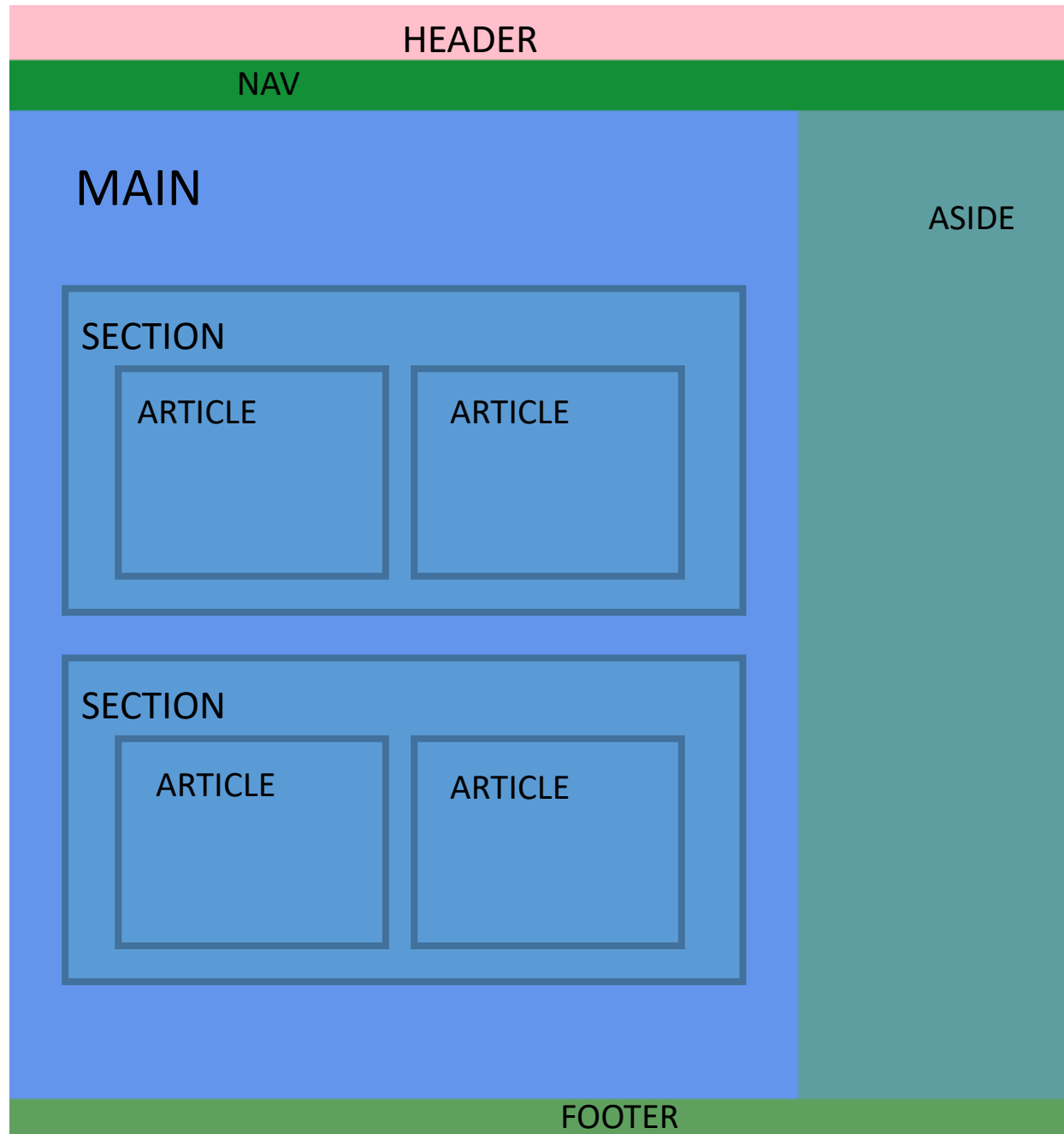
<footer></footer>

</BODY>

emmet



*00index.html*



<https://html5test.com/>

<https://validator.w3.org/>

<https://html5.validator.nu/>

## Semántica de una página HTML5

- Las propias etiquetas HTML deben dar información sobre el contenido y no su formato exclusivamente.
- HTML tiene etiquetas que añaden significado al código.
- Este significado es importante para los usuarios y para los rastreadores web.

*Por ejemplo `<H1></H1>` además de darle un tamaño de letra definido (que podríamos cambiar con CSS) está indicando que es un título, es decir, una información que describe el resto del contenido.*

*`<HEADER>`, `<FOOTER>`, `<MAIN>`, etc.*

## 2.- Identificación de etiquetas y atributos de HTML.

Una etiqueta (*tag*) es un texto que va encerrado entre el símbolo menor que (<) y el símbolo mayor que (>).

Existen dos tipos fundamentales de etiquetas:

- Las de inicio (como por ejemplo: <UL>)
- Las de fin (como puede ser: </UL>).

Las etiquetas siempre afectan al texto (y otras etiquetas) que se encuentren dentro de su apertura y cierre.

**Existen etiquetas que no requieren de apertura y cierre como <IMG>**

<https://developer.mozilla.org/es/docs/Web/HTML/Element>

# Caracteres especiales

Para mostrar correctamente los caracteres `<`, `>`, `"` y `&` en una página HTML se debe sustituir cada carácter especial por su entidad HTML.

`<p>`Los caracteres `&lt;`, `&gt;`, `&quot;` y `&amp;` pueden dar problemas con los textos en HTML si no se emplean caracteres de escape`</p>`

***01CaracteresEspeciales.html***



# Atributos

<A href="<https://www.w3schools.com/>"  
title="El mejor tutorial HTML">HTML5. Todos las etiquetas y atributos</A>

*02Atributos.html*

# Comentarios

`<!-- Esto es un comentario  
y no se verá en la página web -->`

Cuando se está diseñando un sitio web es conveniente, a medida que vamos terminando los distintos bloques, ir documentándolos indicando por qué se ha hecho de tal manera o por qué se usó tal etiqueta.

A veces, durante la etapa de desarrollo se usan los comentarios para anular temporalmente parte del código y ver cómo varía el resultado.

# Elementos de bloque y elementos en línea

Todos los elementos tienen un valor (una forma de mostrarse) predeterminada, *block* e *inline*.

**Bloques.** Siempre comienza en una nueva línea y ocupa todo el ancho disponible.

[\*03BloquesEnlinea.html\*](#)

**En línea.** No comienza en una nueva línea y sólo ocupa el ancho que sea necesario.

Puede modificarse esta situación empleando CSS o Javascript.

# Colores

En HTML, los colores se definen mediante tres números hexadecimales que representan los tonos rojo, azul y verde, usando la codificación RGB del color elegido. La sintaxis para codificar los colores sería, `color="#RRGGBB"`.

Cada dígito hexadecimal puede tomar hasta 16 valores distintos (0-9 y A-F).

RR, GG y BB representan, cada uno, un número hexadecimal entre 00 y FF para el rojo, el verde y el azul respectivamente. Con esta sintaxis, se pueden utilizar más de 16 millones de colores en las páginas. <https://htmlcolorcodes.com/es/>

También hay colores con nombres ya predefinidos.

Extensión para navegador *Colorzilla*.

# La barra separadora.

La etiqueta <HR>, como la mayoría de las marcas puede variar de aspecto dependiendo de una serie de parámetros que podemos predefinir en la hoja de estilos. No necesita cierre. Su formato se le debe dar con CSS.

```
hr {  
    background-color:red;  
    height:10px;  
    width: 50%;  
    border:10px solid blue;  
    border-radius:50px;  
}
```

## Etiquetas para textos.

Sin texto no tendríamos hipervínculos y no podríamos compartir y vincular sitios, no tendríamos una manera tangible de presentar información en páginas.

*Un mal uso sería el abuso del salto de línea BR para maquetar la presentación que deseamos, para separar trozos de texto, para separar títulos, etc.*

Existen etiquetas como <P> para la segmentación de párrafos y para los títulos debe usarse las etiquetas de cabecera desde la H1 hasta la H6.

# Etiquetas para textos.

En HTML no está permitido más de un elemento blanco (espacios, tabuladores, saltos de línea) separando cualquier elemento o texto, todos estos son convertidos a un único espacio blanco y el resto se omiten en la representación del documento. En el documento fuente podremos usar el espaciado que deseemos. La única excepción es `<PRE></PRE>`

*04Textos.html*

Muchas etiquetas aportan valor o significado semántico. Es decir que podemos dar un diseño determinado al texto para comunicar al visitante que clase de texto está leyendo.

*05Textos2.html*

Etiquetas semánticas VS

Etiquetas genéricas + hojas de estilo VS

Etiquetas no semánticas

*05textoSemantico.html*

1

<P>No olvides nunca  
<EM>separar contenido de  
formato</EM></P>



*Etiqueta <EM> además de poner  
en cursiva indica que es  
importante en el contenido.  
Puede modificarse con CSS*

2

<P>No olvides nunca  
<SPAN>separar contenido  
de formato</SPAN></P>



*Etiqueta <SPAN> genérica.  
Necesita CSS obligatoriamente  
para distinguirse  
span { font-style: italic; }*

3

<P>No olvides nunca  
<I>separar contenido de  
formato</I></P>



*Etiqueta <I> no sigue la  
filosofía de HTML5*



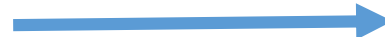
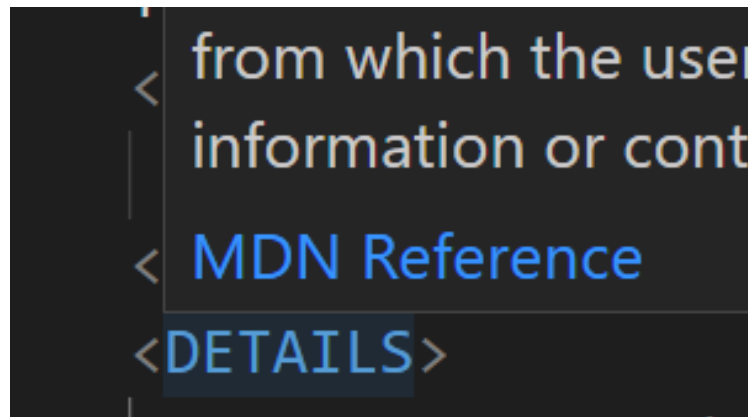
<DETAILS>

<SUMMARY>Usabilidad</SUMMARY>

*Cualidad de la página web o del programa informático que son sencillos de usar porque facilitan la lectura de los textos, descargan rápidamente la información y presentan funciones y menús sencillos, por lo que el usuario encuentra satisfechas sus consultas y cómodo su uso.*

</DETAILS>

*05TextoSemantico.html*



*Especificaciones W3C*

# Enlaces de hipertexto.

Un enlace se identifica fácilmente dentro de una página. **Al pasar por encima del texto o la imagen** el puntero del ratón y se podrá ver como el puntero cambia su forma original transformándose por regla general en una mano con un dedo señalador.

El atributo más importante de la etiqueta <A> es ***href*** que se utiliza para indicar la URL a la que apunta el enlace. Cuando el usuario pincha sobre un enlace, el navegador se dirige a la URL del recurso indicado mediante *href*.

Las URL de los enlaces pueden ser absolutas o relativas, internas o externas.

<A href="URL">Texto del enlace</A>

# Enlace dentro de la página (interno)

Habría que poner una marca en el punto de destino para que el enlace indique certeramente donde queremos lleve al visitante.

Para marcar el punto de destino de un enlace, se utiliza el atributo *id*.

```
<A id="marca"></A>
```

En la parte de texto donde queremos aparezca el enlace debemos incluir la marca <A> con los atributos adecuados donde el más importante es *href*, que contiene el URL o página donde ir.

```
<A href="#marca">Visita Página Ejemplo</A>
```

Es importante el detalle de la almohadilla (#).

## Enlace externo.

Un lugar posicionado en otro archivo de nuestro sitio web.

En este caso simplemente debemos indicar en el atributo *href* el camino donde se encuentra el archivo HTML al que queremos acceder.

```
<A href="informa.html">INFORMACIÓN</A>
```

```
<A href="productos.html">PRODUCTOS</A>
```

Es la forma que tenemos habitualmente de relacionar todos los archivos de un mismo sitio habitualmente distribuidos en menús horizontales o verticales

# Enlace externo.

Un lugar posicionado en otro servidor.

Este es el caso más popular que hemos visto en la definición de hiperenlaces, acceder con un clic a otro sitio web.

*[06Enlaces.html](#)*

Puede indicarse el dominio simplemente o algún archivo concreto e incluso una zona concreta de la web si se indica el símbolo # de una marca previamente puesta.

```
<A href="http://ctxt.es/">CONTEXTO Y ACCIÓN</A>
```

# Enlace absoluto.

La mayoría de enlaces de un sitio web apuntan a páginas del propio sitio web, por lo que se denominan "enlaces internos".

La otra característica que diferencia a los enlaces (y por tanto, también a las URL) es si el enlace es absoluto o relativo. Las URL absolutas incluyen todas las partes de la URL (protocolo, servidor y ruta) por lo que no se necesita más información para obtener el recurso enlazado.



# Enlace relativo.

Los **enlaces relativos** no se incluye la dirección completa del documento o archivo al que apuntan, sino que se omite la información correspondiente a servicio, el dominio y, probablemente, también la ruta.

Este tipo de enlaces se utilizan para direccionar a documentos o archivos que se encuentren en el mismo servidor que el documento en el que se localiza el enlace en cuestión.

*07Enlaces2.html*

Incluir un icono en la pestaña del navegador.  
DENTRO DEL <HEAD>

```
<HEAD>
```

```
<LINK rel="shortcut icon" href="imagenes/logo.png" type="image/ico">
```

```
</HEAD>
```





# Imágenes.

Formatos gráficos para páginas web		
GIF	JPG	PNG
<ul style="list-style-type: none"><li>- Compresión sin pérdida</li><li>- Comprime bien los dibujos</li><li>- Paleta de colores variable</li><li>- Hasta 256 colores</li><li>- Permite transparencia</li><li>- Permite animación</li><li>- Alta compatibilidad</li></ul>	<ul style="list-style-type: none"><li>- Compresión con pérdida</li><li>- Comprime bien las fotos</li><li>- Paleta de color real</li><li>- Hasta 16 Millones colores</li><li>- Sin transparencia</li><li>- Sin animación</li><li>- Alta compatibilidad</li></ul>	<ul style="list-style-type: none"><li>- Compresión sin pérdida</li><li>- Comprime bien los dibujos</li><li>- Paleta de colores variable</li><li>- Hasta millones de colores</li><li>- Permite transparencia</li><li>- Sin animación</li><li>- Menor compatibilidad</li></ul>
OPTIMIZACIÓN: <ul style="list-style-type: none"><li>- Reducir paleta de colores</li></ul>	OPTIMIZACIÓN: <ul style="list-style-type: none"><li>- Alterar calidad de la imagen</li></ul>	OPTIMIZACIÓN: <ul style="list-style-type: none"><li>- Reducir paleta y más</li></ul>

*08Imágenes.html*

# Listas

HTML incorpora unas listas con viñetas sencillas o también letras o números. Para dar más vistosidad a las páginas. Una de sus características principales es la de proporcionar la posibilidad de sintetizar información textual de lo que estemos mostrando en la web.

HTML ofrece a los autores varios mecanismos para especificar listas de información. Todas las listas deben contener uno o más objetos de lista. Las listas pueden contener:

- Información no ordenada.
- Información ordenada.
- Definiciones

# Listas no ordenadas.

Las listas no ordenadas van dentro de la etiqueta <UL> y de su cierre </UL>. Cada punto que queramos añadir a la lista, lo haremos dentro de la etiqueta <LI> y su cierre.

Si no le indicamos nada a la etiqueta <LI>, ésta se generará de forma automática. El atributo *type* nos permite elegir entre varios tipos de iconos o viñetas.

```
<UL>
```

```
<LI type="circle">Esto es un tipo de punto.</LI>
```

```
<LI type="square">Este es otro.</LI>
```

```
<LI type="disc">Y este es otro diferente.</LI>
```

```
</UL>
```

# Listas ordenadas

Las listas ordenadas van enmarcadas dentro de las etiquetas `<ol>` `</ol>`. Cada punto de la lista se escribe con `<li>` pero al ser listas ordenadas los símbolos serán números y éstos se irán generando automáticamente por orden, conforme escribamos nuevos puntos.

`<OL>`

`<LI value="20">Este será el número 20. </LI>`

`<LI>Este será el 21. </LI>`

`<LI> Este será el 22. Y así sucesivamente. </LI>`

`</OL>`

# Listas ordenadas.

- 1: numeración estándar: 1, 2, 3...
- a: numeración alfabética en minúsculas: a, b, c...
- A: numeración alfabética en mayúsculas: A, B, C...
- i: numeración en números romanos y en minúsculas: i, ii, iii...
- I: numeración en números romanos y en mayúsculas: I, II, III...

<OL><LI type="I">Número romano  
mayúsculas. </LI>

<LI type="A">Las letras en mayúsculas </LI>

<LI>Tomates </LI>

<LI>Lechugas </LI></OL>

# Anidar listas.

Las listas se pueden anidar unas dentro de otra aunque sean de tipos distintos. Una lista sólo puede contener elementos LI que a su vez pueden contener otros elementos HTML.

```
<OL>  
  <LI>Primer plato</LI>  
    <UL>  
      <LI>Macarrones</LI>  
      <LI>Gazpacho</LI>  
      <LI>Judías pintas</LI>  
    </UL>  
</OL>
```

# Listas de definición.

Las listas de definición están constituidas por términos y su subsiguiente definición. La forma de implementar las listas de definición es con la marca <DL>.

<DL>

<DT> Hormiga</DT>

<DD> Insecto himenóptero</DD>

<DT> Mosquito</DT>

<DD> Insecto díptero</DD>

</DL>

[09Listas.html](#)

[10Listas2.html](#)

Cada término de la lista de definición se indica con la marca <DT> y no creará sangría, mientras que la descripción del término se indica con la marca <DD> y sangrará hacia la derecha para destacar el término.

# Tablas.

Una tabla HTML puede ser considerada, de manera simple, como un grupo de filas donde cada una de ellas contiene un grupo de celdas (y no al revés).

Las tablas en HTML tienen tres etiquetas que permite crearlas de forma ágil: `<TABLE>` para crear la tabla, `<TR>` para crear cada fila y `<TD>` para crear cada columna.

La etiqueta `<TABLE>` encierra todas las filas y columnas de la tabla. Las etiquetas `<TR>` definen cada fila de la tabla y encierran todas las columnas. La etiqueta `<TD>` definirá entonces a cada una de las celdas de datos.



# Tablas.

```
<TABLE>
```

```
<TR>
```

```
<TH>Origen</TH>
```

```
<TH>Destino</TH>
```

```
<TH>Compañía</TH>
```

```
</TR>
```

```
<TR>
```

```
<TD>Málaga</TD>
```

```
<TD>Madrid</TD>
```

```
<TD>ALSA</TD>
```

```
</TR>
```

```
<TR>
```

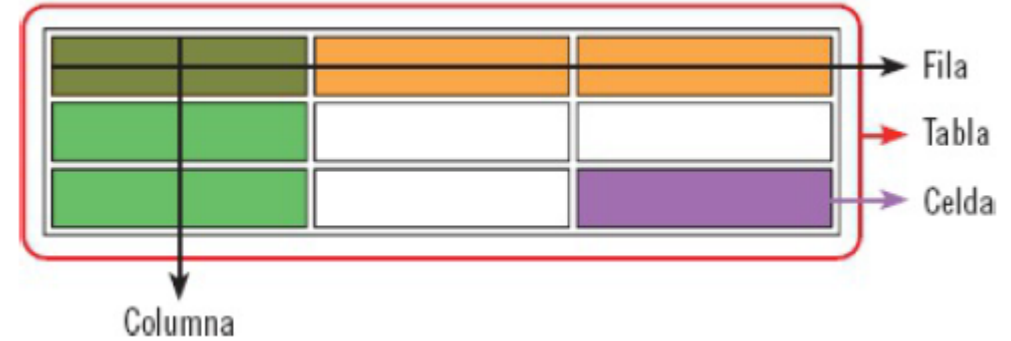
```
<TD>Cádiz</TD>
```

```
<TD>Almería</TD>
```

```
<TD>GadirBUS</TD>
```

```
</TR>
```

La primera fila es TH  
para destacarla



```
<TR>
```

```
<TD>Sevilla</TD>
```

```
<TD>Málaga</TD>
```

```
<TD>ALSA</TD>
```

```
</TR>
```

```
<TR>
```

```
<TD colspan="3">Todos los autobuses equipados con  
WIFI</TD>
```

```
</TR>
```

```
</TABLE>
```

[11Tablas.html](#)

[11Tablas2.html](#)

También existe *rowspan*

Uso de *summary*

# Los iframes.

IFRAME nos permite crear una zona dentro de la página donde se puede incrustar otra web.

Es un rectángulo del tamaño que consideremos oportuno que tiene asociada una página web que se carga en dicho espacio.

Será independiente y autónoma respecto al contenido de la página madre. Puede configurarse para que muestre *scroll* (barras de desplazamiento) o el tamaño del borde.

[\*12Iframe.html\*](#)

```
<iframe src="https://www.w3schools.com"></iframe>
```

# Formularios.

Un formulario puede insertarse en un documento HTML a través de la etiqueta FORM que actuará como contenedor para todos los elementos de entrada. La información introducida debe ser enviada a un servidor (agente procesador) utilizando el atributo *action*. Habitualmente se indica el archivo que recogerá la información para procesarla.

Cómo serán enviados los valores de los campos debemos especificarlo con el atributo *method*:

- *post*, los datos del formulario son adjuntados al cuerpo del mismo.
- *get*, los datos del formulario son adjuntados a la URL. (NO RECOMENDABLE)

La recogida de la información se hace mediante programación con lenguajes del lado del servidor.

## Formularios.

Si los datos son pocos y no nos importa su confidencialidad o manipulación, utilizamos GET. Si son largos o privados o importantes, no lo dudaremos, usaremos POST.

### <INPUT>

color	range
date	search
datetime	tel
email	time
month	url
number	week
text	file
password	radio
image	checkbox

- Elementos enfocables (enlaces e inputs)

- Empleo de tabindex

- Etiqueta LABEL

<BUTTON type="button">

<TEXTAREA rows='5' cols='45'>

</TEXTAREA>

<SELECT> → <OPTION>

# Formularios.

```
<FORM method="post" action="destino.html">
```

AQUÍ VAN LOS CONTROLES O INPUTS

```
</FORM>
```

```
<input type="text">
```

```
<input type="submit" value="¡DALE!">
```

```
<input type="reset" value="BORRAR!">
```

*13Formularios.html*

*14Formularios2.html*

*15Formularios3.html*

*16Formularios4.html*

## Formularios.

**id** se utiliza para identificar el elemento **HTML** a través del Modelo de Objeto del Documento (a través de JavaScript o con estilo CSS), se espera que sea único dentro de la página.

**name** corresponde al elemento del **formulario** e identifica lo que se publica de nuevo en el servidor .

```
<label for="clave">Contraseña </label>
```

```
<input type="password" id="clave" name="clave">
```

```
<label for="html">HTML</label>
```

```
<input type="radio" id="html" name="lenguaje" value="HTML">
```

```
<label for="css">CSS</label>
```

```
<input type="radio" id="css" name="lenguaje" value="CSS">
```

# Formularios.

*Login*      *18formularioLogin.html*

*Registro*      *18formularioRegistro.html*

*pattern*      *19pattern.html*

*pattern*      *19destino.html*

# Expresiones regulares

$[a-zA-Z]+([.][a-zA-Z0-9_-]+)^*@[a-zA-Z]+([.][a-zA-Z0-9_-]+)^*.[a-zA-Z]\{2,4\}$

*Por ejemplo, para controlar que los correos electrónicos cumplan un patrón*

<https://lenguajehtml.com/html/formularios/validaciones-html5/#patrones-de-validaci%C3%B3n-html5>

?= obliga a que la siguiente expresión aparezca en el “texto”

$(?=.[0-9])(?=.[a-z])(?=.[A-Z]).\{8,\}$

Debe incluirse al menos un número, una mayúscula y una minúscula en 8 caracteres o más.

<https://www.adictosaltrabajo.com/2015/01/29/regexsam/>



# Atributos globales

[https://www.w3schools.com/tags/ref\\_standardattributes.asp](https://www.w3schools.com/tags/ref_standardattributes.asp)

<https://www.htmlquick.com/es/reference/attributes.html>

- *tabindex / accesskey*
- *class / id*
- *contenteditable*
- *dir*
- *hidden*
- *translate*

*20atributosGlobales.html*

## Audio y vídeo.

`<audio controls>`

`<source src="recursos/yuju.mp3">`

`<source src="recursos/yuju.wav">`

`<source src="recursos/yuju.ogg">`

Tu navegador no soporta el elemento audio

`</audio>`

*[17Multimedia.html](#)*

`<video controls width="600" height="400" poster="imagenes/fotoplaya.png">`

`<source src="recursos/big_buck_bunny.mp4" type="video/mp4">`

`<source src="recursos/big_buck_bunny.ogg" type="video/ogg">`

`<source src="recursos/big_buck_bunny.webm" type="video/webm">`

Tu navegador no soporta el elemento video

`</video>`

- Con `<audio>` indicamos que vamos a incrustar un audio.
- Con `<source>` especificamos la ruta del audio a reproducir.
- Con `<video>` indicamos incrustar un vídeo, podemos asignar un alto y ancho al reproductor.

### 3.- Hojas de estilo.

Son un conjunto de normas o reglas mediante las que indicamos el modo en que aparecerá un documento en pantalla controlando temas de formato y posicionamiento como son el color, el fondo, el tipo de fuente, la apariencia de los bordes, los márgenes, la alineación y el espacio entre caracteres, entre otras muchas.



# Selectores

El **selector** es el elemento sobre el que se aplicarán las propiedades, puede ser desde una simple etiqueta, clases, clases aplicadas a etiquetas concretas, sucesión lógica de etiquetas, etc.

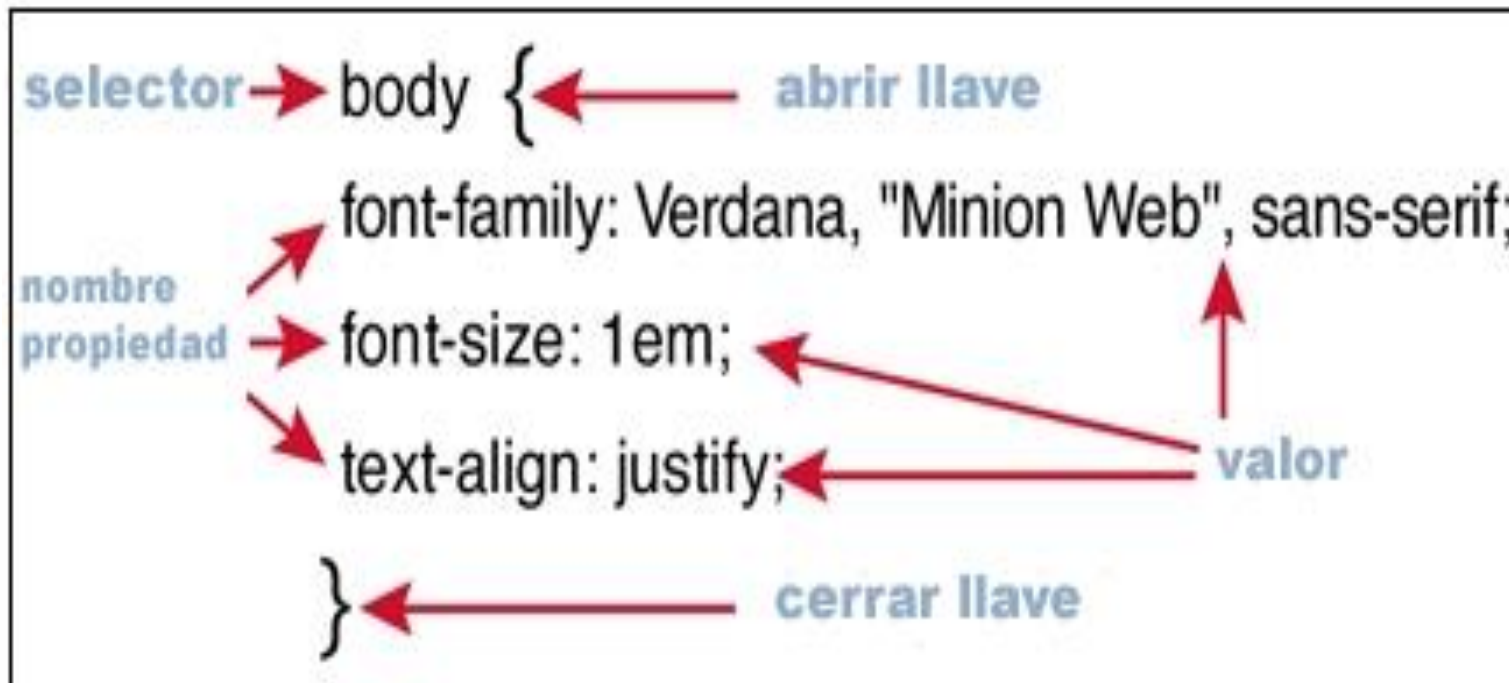
Las **propiedades** irán siempre seguidas de dos puntos(:) y el valor de las propiedades.

**\* Es el selector universal**

```
* {  
    padding:0;  
}
```

# Selectores

La sintaxis tiene una estructura como la que se muestra en la imagen siguiente.



## Elementos y estructura de una hoja de estilo.

Cuando creamos un estilo, las especificaciones que se realizan para un determinado elemento serán aplicables a todos aquellos elementos que se encuentren por debajo de él atendiendo a un criterio de **herencia**.

*Por ejemplo, si especificamos un tipo de letra para el elemento <BODY> todos aquellos elementos que puedan heredar las características se presentarán con el mismo tipo de fuente. Así, el elemento <P> heredará el tipo de letra salvo que especifiquemos lo contrario.*

```
<HTML><HEAD><STYLE>  
p {  
color:red;  
}  
</STYLE></HEAD>
```

```
<BODY>  
Hola  
<HR>  
<CODE>hola</CODE>  
<P>Hola <CODE>hola</CODE></P>  
</BODY></HTML>
```

CSS01.html

## Elementos y estructura de una hoja de estilo.

Prefijo	Familia de navegadores a los que aplica	
-webkit-	Chrome, Safari, Android, iOS	
-moz-	Firefox	
-o-	Opera	

Prefijos que se antepone a una regla CSS para que sea aplicada exclusivamente por un navegador concreto pero no por el resto de navegadores.

El uso de prefijos suele aplicarse a propiedades que se encuentran en fase experimental o que aún no se han convertido en un estándar.

## Elementos y estructura de una hoja de estilo.

Hay que tener en cuenta que se establece un orden de prioridad para la aplicación de las normas en caso de que resulten contradictorias.

El orden de ejecución de los estilos es el siguiente:

- Estilo especificado en la cabecera del documento. **DEPENDE DEL ORDEN EN QUE ESTÉN dentro de HEAD**  
*Entre <style> </style> en <HEAD>*
- Estilo definido en un documento independiente al que se enlaza nuestra página. **RECOMENDADO**  
*En un fichero externo (etiqueta <link> en <HEAD>)*
- Estilo especificado dentro de la etiqueta.  
*En <body>, dentro de una etiqueta <p style="text-align:center;">*



Elementos y estructura de una hoja de estilo.  
Vinculación a una hoja de estilo externa.

Dentro de <HEAD> insertamos una etiqueta <LINK>.

***<LINK href="estilos03.css" rel="stylesheet" type="text/css">***

También puede incluir un atributo media para especificar los medios a los que se aplican las reglas de la hoja de estilo. Al leer el valor del atributo media en la etiqueta <LINK> el navegador puede descargar selectivamente archivos de hojas de estilo aplicables únicamente a los medios utilizados por el explorador.

***<LINK href="estilos03print.css" rel="stylesheet" type="text/css" media="print" >***

CSS03.html

estilos03.css

estilos03print.css

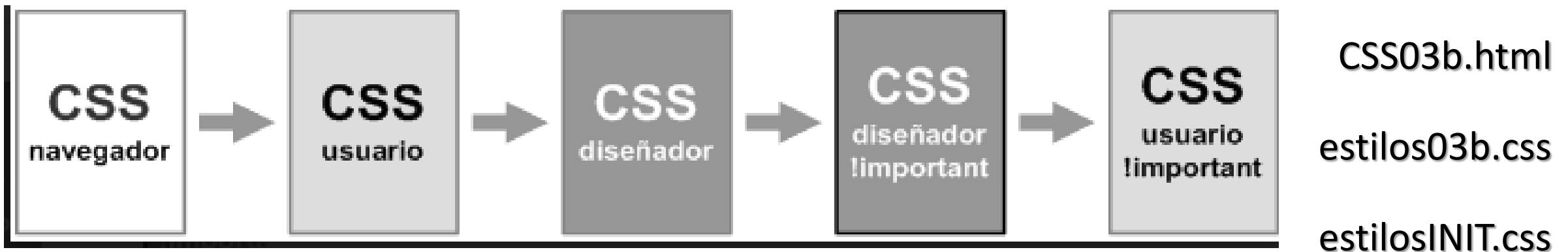
## Estructura de las hojas de estilo

Cuando hablamos de "hojas de estilo", normalmente hacemos referencia a una **hoja de estilo de autor**. Es la hoja de estilos que ha creado o enlazado el autor del documento, el diseñador de la web.

¡Son las que hacemos nosotros!

Las **declaraciones importantes** rompen el orden lógico. Van seguidas de la directiva *!important*.

**\* { font-family: "Comic Sans MS" !important; }**



# Selectores. Técnicas de aplicación de estilos.

- **Selectores.** Podemos redefinir cualquier elemento o etiqueta HTML.

```
h1 { color:lime ; }
```

- **Selectores de clase.**

Existen dos tipos de selectores de clase: generales (afectan a todas las marcas donde se posicionen en el HTML con el atributo *class*) y específicos (se especifica a que selector afectan).

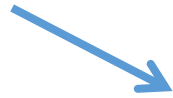
.peque { font-size: 6px;} → <p class="peque">HOLA</p>  
p.mediano {font-size: 12px;} → <div class="peque">ADIOS</div>

NO LE AFECTA p.mediano → <p class="mediano">HOLA</p>  
NO LE AFECTA p.mediano → <div class="mediano">HOLA</p>

## Selectores. Técnicas de aplicación de estilos.

- **Selectores ID.** Son similares a los selectores de clase pero están pensados para que el elemento que especifica sea único..

*#cabecera { font-size: 16px; }*



*<DIV id="cabecera">Contenido con formato único</DIV>*

css04.html

estilos04.css

## Selectores. Técnicas de aplicación de estilos.

- **Selectores contextuales** (descendientes).

Podemos definir unas propiedades para un selector cuando se encuentre en unas condiciones concretas, por ejemplo, cuando esté dentro de otro selector aunque no sea hijo directo.

```
div .peque{ color:orange; }
```

CSS05.html

-----

```
<DIV><EM>Hola</EM></DIV>
```

estilos05.css

```
<P><EM>Hola</EM></P>
```

```
<DIV>Hola <P><EM>Hola</EM></P></DIV>
```

# Selectores. Técnicas de aplicación de estilos.

## **Grupo de selectores.**

Para agrupar propiedades y crear un código más compacto y fácil de visualizar se pueden aplicar reglas comunes a diferentes selectores.

*Por ejemplo,*

*h1, h2, h3 {font-family: Arial, sans-serif;}*

*.seccion1, #tipo1, h4 { font-style: italic; }*

# Selectores. Técnicas de aplicación de estilos.

## Selectores adyacentes.

El selector adyacente se emplea para seleccionar elementos que en el código HTML de la página se encuentran justo a continuación de otros elementos.

***h1 + h2 { color: red ;}***

## Selector de hijo directo.

Se utiliza para seleccionar un elemento que es *hijo directo* de otro elemento y se indica mediante el signo "*mayor que*" (>) CSS05b.html

***P>span { color:blue;}***

## Selectores. Técnicas de aplicación de estilos.

- **Pseudoclases.** Permiten referirnos a los selectores según su estado o situación. Para indicarlás en la hoja de estilo se escriben detrás del selector separadas por dos puntos.

Por ejemplo, *:first-child* es una pseudo-clase que va orientada a personalizar el primer elemento hijo de cualquier otro elemento

```
div p:first-child {  
    font-size:20px;  
    font-style:italic; /* cursiva */  
}
```

CSS06a.html

estilos06a.css

CSS06b.html

estilos6b.css



## Selectores. Técnicas de aplicación de estilos.

- **Pseudoelementos.** Emplean :: en lugar de :

Son elementos de una página que no están contenidos en una etiqueta específica y por tanto no podemos asignarles atributos directamente.

p::first-line { color: blue; }	.caso::before { content : "OJO: "; color : #d14; /*#dd1144;*/ }	CSS07a.html estilos07a.css
p::first-letter { font-family:"palo seco"; font-size: 200%; color: red; }	.caso::after { content : url("imagenes/exclama.jpg"); }	CSS07b.html estilos07b.css  CSS07c.html estilos07c.css

<https://front.id/es/articles/pseudo-clases-y-pseudo-elementos>

Listado de pseudoclases y pseudoelementos

## Selectores. Herencia de estilos.

La herencia permite declarar propiedades en elementos de nivel alto y que estas propiedades se transmitan a todos los elementos descendientes.

Sólo algunas propiedades se heredan por defecto, pero la herencia puede forzarse mediante la palabra clave *inherit*.

*No todas las propiedades CSS son heredadas, porque algunas de ellas no tendría sentido que lo fueran. Por ejemplo, los márgenes o bordes no se heredan porque es poco probable que un elemento hijo necesite los mismos márgenes que su padre.*

El color de fondo no se hereda, pero el valor inicial de background-color (color de fondo) es transparente, lo cual significa que el fondo del padre se verá a través de él.

## Selectores. Herencia de estilos.

El color de fondo no se hereda, pero el valor inicial de *background-color* (color de fondo) es *transparent*, lo cual significa que el fondo del padre se verá a través de él.

Valores iniciales y herencia de cada propiedad

<https://www.w3.org/TR/css-2010/#properties>

CSS08a.html

estilos08a.css

<https://lenguajecss.com/css/cascada-css/herencia-css/#valores-especiales-de-herencia>

## Selectores. El selector de atributo

*selector [atributo] {*

*-----*

*-----*

*}*

CSS08b.html

estilos08b.css





En el ejemplo puede verse también el uso de *::before* y de selector hijo directo. Todavía hay especificaciones no implementadas como es el caso de

*::details-marker*

*::-webkit-details-marker*

# Selectores. La especificidad en caso de conflicto.

La especificidad de un selector se calcula contando por separado el número de tipos de selectores siguiendo esta prioridad:

- #identificadores  Máxima prioridad siempre *!important* styles en las etiquetas. 
- .clases, :pseudoclases y [selectores de atributos]
- Tipos de elementos (etiquetas) y ::pseudoelementos.

*Por ejemplo, el selector **ul.menu > li > a** tiene especificidad (0, 1, 3), ya que tiene cero identificadores, una clase y tres etiquetas.*

*El selector **a** tiene especificidad (0, 0, 1)*

*Se comparan por tipo y en el momento que uno es superior gana la prioridad. En el ejemplo anterior,  $0=0, 1>0$  ya se para de mirar, **ul.menu > li > a** tiene más prioridad. En caso de empate: miramos el orden de aparición, la última en llegar es la que gana.*

# Propiedades tipo shorthand

font-family

font-style → italic, oblique  
(cursiva)

font-weight → bold  
(negrita)

font-size

font-variant → small-caps  
(versalitas)

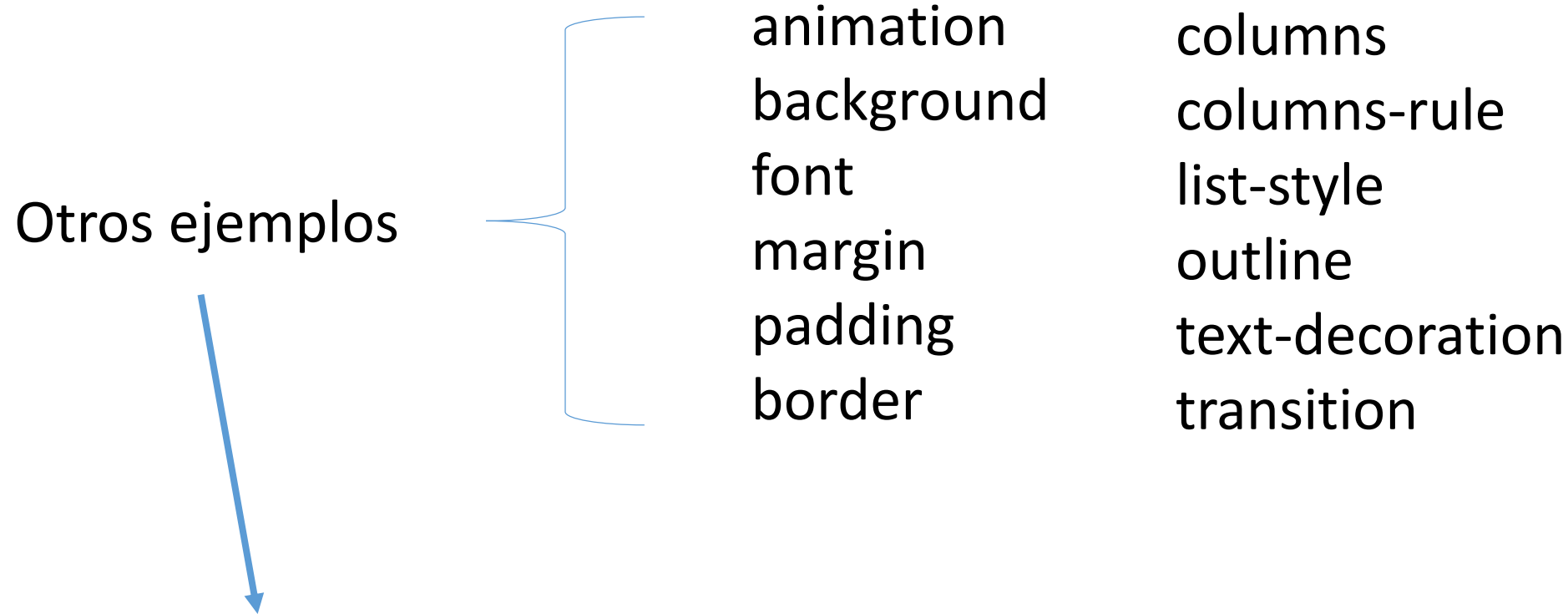
**Propiedad tipo *shorthand* permite agrupar en una sola propiedad varias características.**

***font***



Define todo lo relacionado con los recursos tipográficos, sus propiedades

# Propiedades tipo shorthand

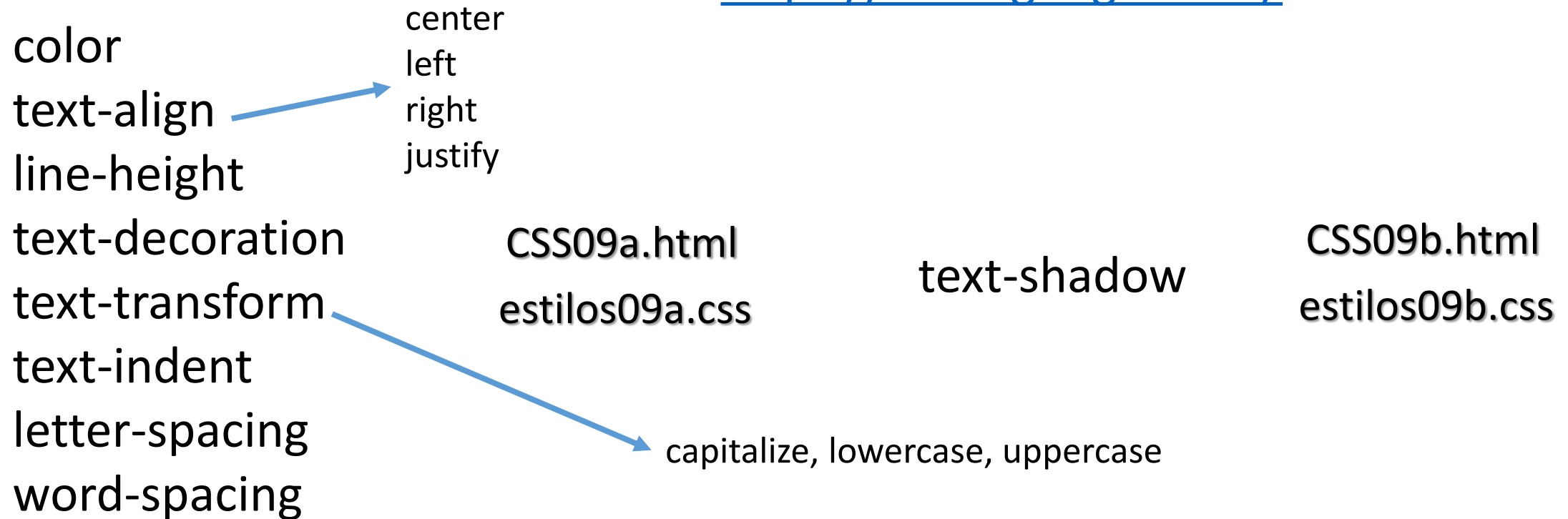


border, border-bottom, border-left, border-right, border-top, border-radius, border-style, border-width, border-color

# Textos y fuentes

Fuentes externas

<https://fonts.google.com/>



Las unidades de medida

<https://lenguajecss.com/css/modelo-de-cajas/unidades-css/>



# Alinear textos y elementos en línea en una caja

```
section {
  text-align: center;
}
```

Los hijos heredarán esta propiedad y por eso P también se centra

```
<SECTION>
<P>Hola mundo cruel</P>
<IMG src='imágenes/santiago1.jpg'>
</SECTION>
```

CSS09c.html  
estilos09c.css

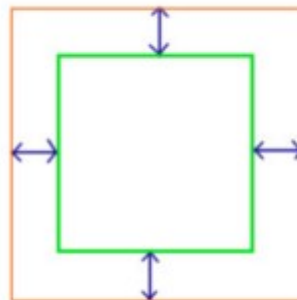
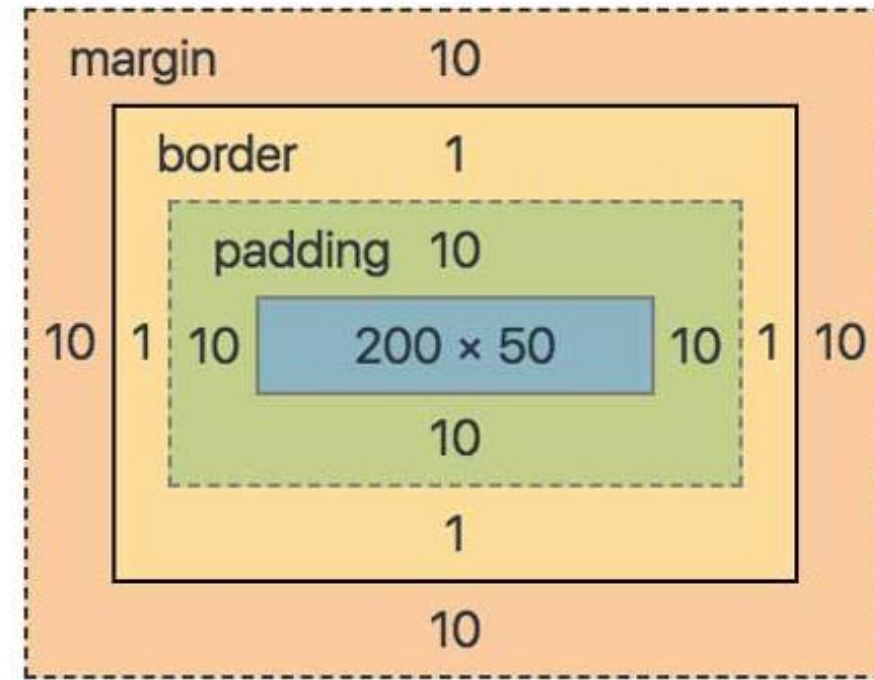
## Alinear un bloque dentro de otro bloque (*centrar una caja dentro de otra*)

Si queremos centrar P dentro de SECTION

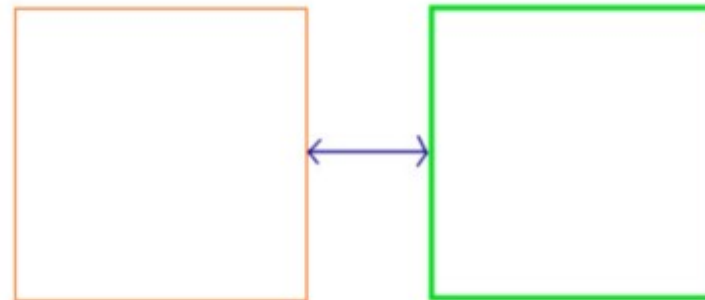
```
p {
  margin:0 auto;
}
```

# Modelo de cajas

width  
height  
margin  
padding  
border



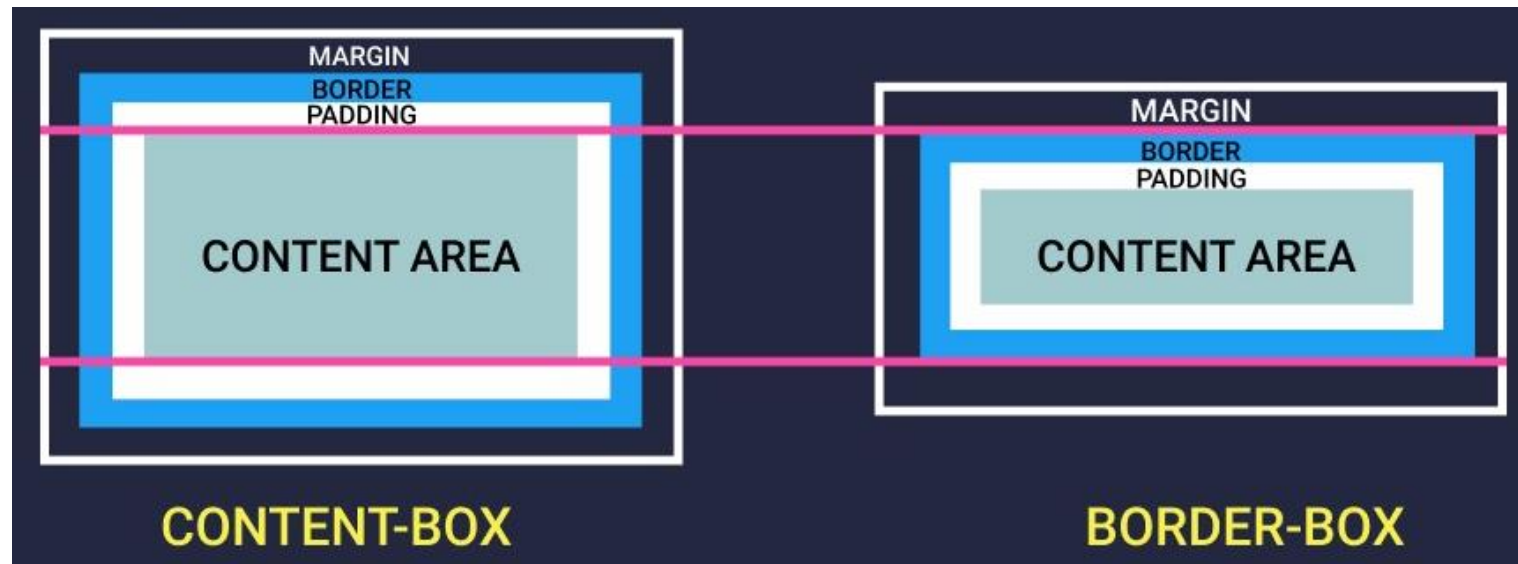
**Padding**



**Margin**

[http://librosweb.es/libro/css/capitulo\\_4.html](http://librosweb.es/libro/css/capitulo_4.html)

El valor *border-box* en el **box-sizing** hace que el *padding* y el *border* pasen a formar parte del cálculo del ancho de la caja y no lo suman posteriormente.



CSS10a.html  
estilos10a.css

*display* { *inline*  
*block*  
*inline-block* → Los elementos *inline-block* fluyen con el texto y demás elementos como si fueran elementos en línea pero respetan el ancho, el alto y los márgenes verticales.

CSS10b.html  
estilos10b.css

*border-radius*

*background*  
*border*

CSS10c.html  
estilos10c.css

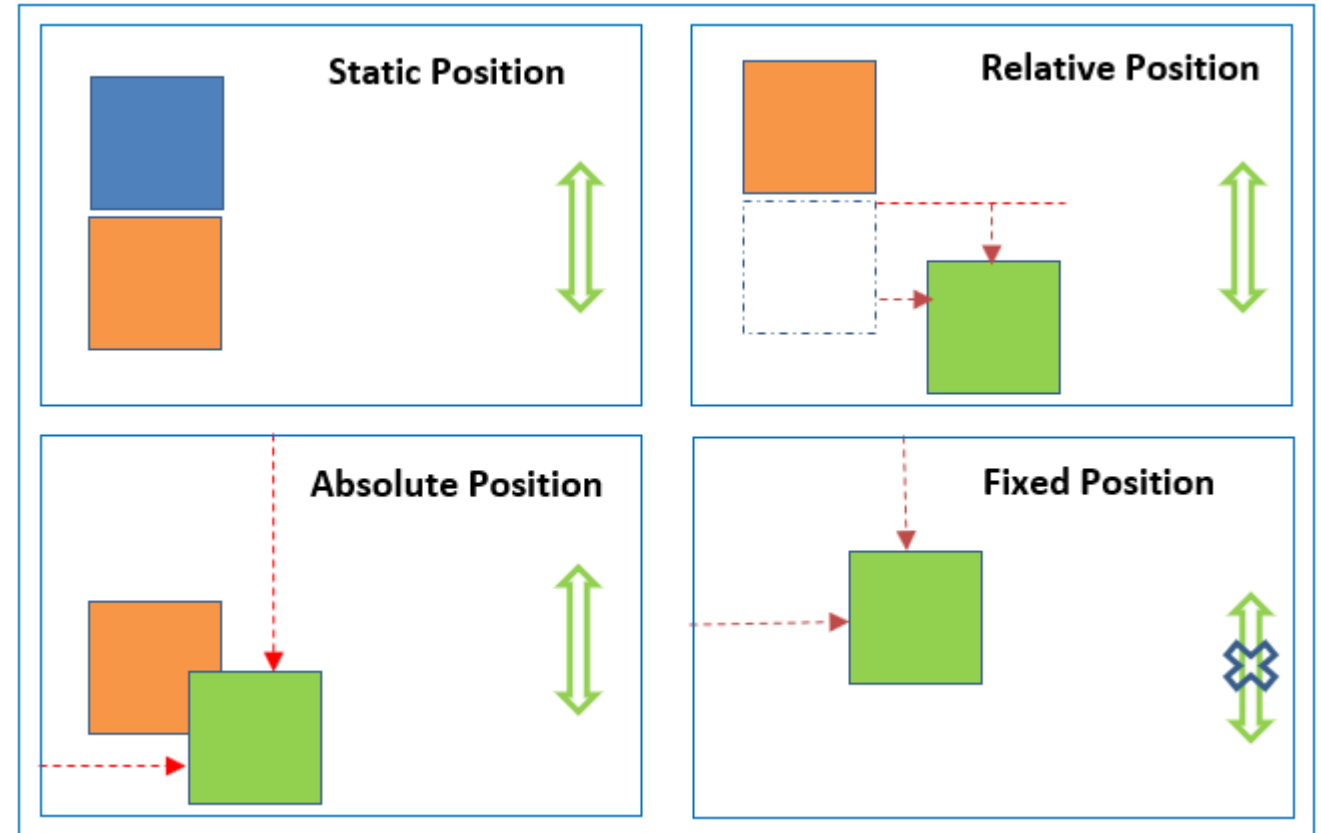
Diferencia entre *border* y *outline*

<https://developer.mozilla.org/es/docs/Web/CSS/outline>

# Posicionamientos

*static* (por defecto)  
*absolute*  
*relative*  
*fixed*

CSS11a.html  
 estilos11a.css




Método tradicional de  
 distribución de bloques



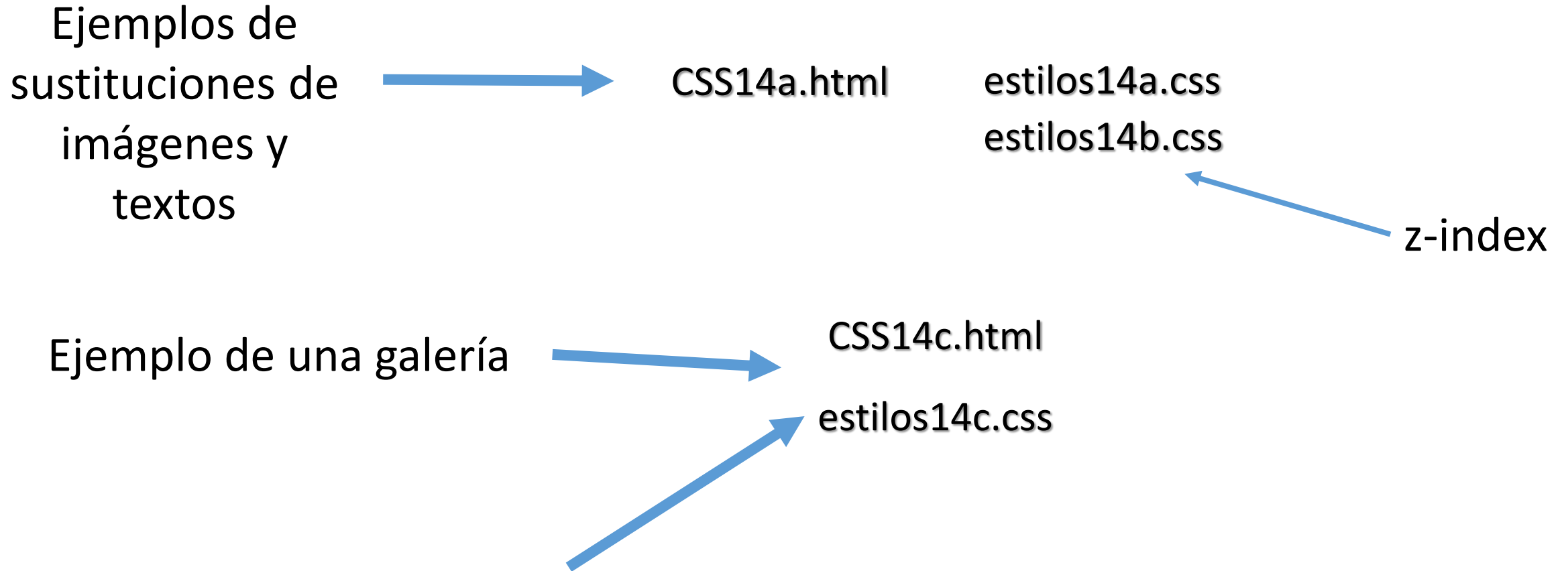
*float:left;*  
*float:right;*  
*clear:both;*

CSS11b.html  
 estilos11b.html  
*column*

Listas  CSS12.html estilos12.css

Visualización.  
Imágenes de sustitución  CSS13.html  
estilos13.css

- Sustituyendo *background-image*
- *visibility:visible / hidden* *display: none / block / inline*  
No se ve pero reserva espacio No se ve y pierde su espacio
- *z-index*



Si tenemos un elemento con `position: absolute` y ese elemento está dentro de un contenedor que también está posicionado con los valores `relative`, `absolute` o `fixed`, el elemento se va a colocar respecto a ese contenedor no respecto al *viewport* (área visible del navegador).

# FORMULARIOS

CSS15.html  
estilos15.css

Enlaces  
como botón

CSS16.html  
estilos16.css

*transition*

<https://cybmeta.com/animaciones-basicas-con-css-transition>

<https://joanleon.dev/transiciones-css>

# TABLAS

*border-collapse*  
*border-spacing*  
*caption-side*

*empty-cells*  
*table-layout*

CSS17.html  
estilos17.css

CSS17b.html  
estilos17b.css

Ejemplo de  
*display:flex;*

Formatos  
apariciencia





# Distribución básica de bloques usando *display:flex;*

## Propiedades del bloque padre.

CSSFlex.html

`display:flex;`

estilosFlex.css

`flex-wrap`

`flex-direction`

CSSFlex2.html

`justify-content`

estilosFlex2.css

`align-items`

`align-content`

## Propiedades de los bloques hijos.

`order`

`flex-grow`

`flex-basis`

*Puede haber elementos que sean padres e hijos a la vez*

# Hojas de estilos alternativas

En una página web, las hojas de estilo se enlazan en la cabecera <HEAD> mediante una etiqueta <LINK>. El atributo *rel* con el valor *stylesheet* indica que se trata de un enlace a una hoja de estilo y el atributo *href* indica la ubicación y nombre de la hoja de estilo enlazada.

```
<link rel="stylesheet" href="estilo18a.css">
```

Para que el usuario pueda elegir en el navegador qué hoja de estilo quiere aplicar, es suficiente con que los enlaces a las hojas de estilo alternativas tengan el atributo *title* y el valor *alternate* en el atributo *rel*.

```
<link rel="stylesheet" href="estilo18a.css" title="Estilo 1">
```

```
<link rel="alternate stylesheet" href="estilo18b.css" title="Estilo 2">
```

```
<link rel="alternate stylesheet" href="estilo18c.css" title="Estilo 3">
```

CSS18.html

estilos18a.css

estilos18b.css

estilos18ccss

## Buenas prácticas en el uso de hojas de estilo.

- Todos los documentos tienen que incluir una directiva DOCTYPE.
- No usar el atributo *style* dentro de las etiquetas.
- Es recomendable emplear los selectores justos y necesarios.
- Utilizar minúsculas tanto para los nombres, propiedades y valores de una regla CSS.
- Los nombres de los selectores tienen que tener un significado relativo al contenido y no a su disposición o estructura.

## Buenas prácticas en el uso de hojas de estilo y HTML5.

- Ordenar las propiedades de una regla siempre del mismo modo.
- Elegir un solo idioma para el nombre de los selectores.
- Agrupar clases en la medida que sea posible.
- Primero se diseña la estructura en HTML y después se introduce formato en CSS, no al revés.
- Usar propiedades agrupadas (*shorthands*).

## Buenas prácticas en el uso de hojas de estilo y HTML5

- Comentar siempre el código.
- Si queremos centrar cajas, *margin: 0 auto*; es una buena solución.
- Para centrar contenido, *text-align:center*;
- Intentar utilizar las etiquetas HTML para lo que fueron diseñadas originalmente.
- Cuando usemos Hx no puede haber salto de un nivel a otro no consecutivos.

*Si mi página tiene un H3, tiene que haber también un H2 y un H1.*

## La importancia de los preprocesadores y el futuro de CSS

Un preprocesador es un programa que cuenta con su propia sintaxis única. Después de escribir el código, lo *compilará* a CSS puro.

Con un preprocesador CSS es posible añadir **funcionalidades adicionales** que de otro modo CSS no tendría (por ejemplo, uso de variables o permitir anidamiento de selectores).



<https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/less-css-tutorial/>