CSS

Antonio Espín Herranz

Contenidos

- Introducción.
- Selectores, propiedades, valores
- Atributo Style
- Orden de aplicación
- Pseudo Selectores
- Tipos de propiedades: longitud, url, color, de palabra clave
- Propiedades: display, white-space, fuentes, color, background, text
- Elementos Box: márgenes, bordes, padding, tamaño, posicionamiento.
- Enlaces.
- Organización de la página en capas.
- Capas: maquetación a 2 y 3 columnas, centrar contenedores.
- Visibilidad y opacidad en capas.

Introducción

- CSS: Proporciona estilos visuales a los elementos del documento, como son: tamaño, color, fondo y bordes, etc.
- Tipos de elementos:
  - **block**:
    - Se posicionan uno debajo de otro.
      - Todos los elementos estructurales: header, nav, section, aside, footer y otros como p, div, form: son de tipo block (por defecto).
  - **inline**:
    - Se posiciona uno al lado del otro en la misma línea horizontal. Cuando se llena la línea salta a la siguiente y así sucesivamente.

Introducción

- Colocación de los elementos css:
  - En el mismo elemento con el atributo style:
    - **Elementos en línea**:

```
<body>
    <p style="font-style: 20px">Mi párrafo</p>
</body>
```

    - **Estilos embebidos**:

```
<head>
    <style>
        p {font-style: 20px; }
    </style>
</head>
<body>
    <p>Mi párrafo</p>
</body>
```

Introducción

- Colocación de los elementos css:
  - **Ficheros externos**:
    - Se referencian dentro del head:

```
<head>
    <link rel="stylesheet" href="estilos.css">
</head>
<body>
    <p>Mi párrafo</p>
</body>
```

    - Esta forma es la aconsejada, no complica el código html y permite la reutilización de los estilos en varias páginas.

Atributo STYLE

- En casos muy raros, es posible que el usuario desee aplicar un estilo concreto, una sóla vez a un elemento sin modificar el fichero con extensión *css*, para esto se utiliza el atributo STYLE.

- El atributo **STYLE** posee mayor precedencia que el estilo definido en el fichero *css*.

HERENCIA

- No es necesario que todas las reglas especifiquen que estilo utilizar puesto que los elementos específicos heredan por defecto el estilo de los elementos más generales que los incluyen.

- En caso de que se aplique otro estilo en un elemento mas interno, siempre prevalece el estilo mas interno o mas especifico, sobre el de mayor jerarquía o mas específico.

ORDEN DE APLICACIÓN

- Las referencias circulares no están permitidas.
- El orden en el que se aplican las reglas es muy sencillo. Generalmente si existen más de una regla para un mismo elemento, ***se aplica la regla más específica***.
- Por ejemplo: En una regla con un atributo ID y CLASS tendría preferencia la aplicación del ID primero y después de la clase y en su defecto la regla del elemento general y sino las reglas que rigen el elemento padre y si no existen las reglas generales de estilo del navegador.

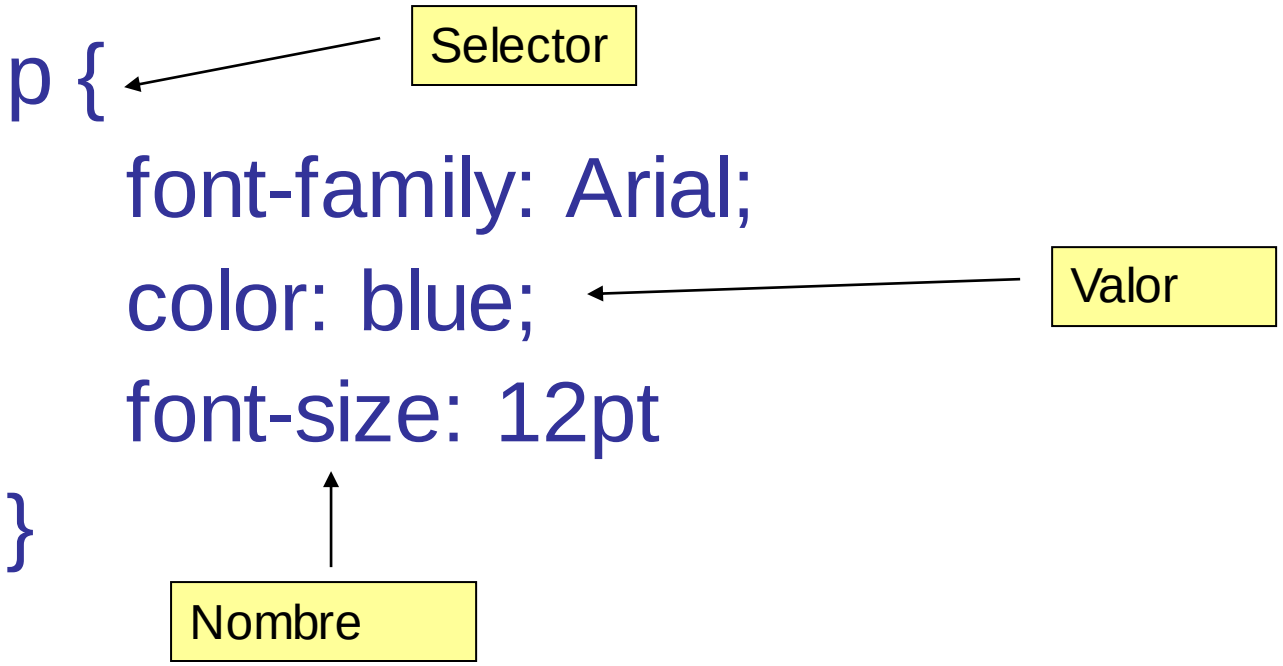
Componentes de las declaraciones

- **Selectores**:
  - Se utilizan para seleccionar a qué elementos de XHTML se aplican las propiedades que se especifican.
    - En el caso de XML, los selectores son los nombres de las etiquetas.
- **Nombre**:
  - Especifica la propiedad que se aplicará al elemento seleccionado.
- **Valor**:
  - Indica el valor que tomará la propiedad antes especificada.
- **Comentarios** → /* Esto es un comentario */

Unidades CSS

- Especificamos la lista de estilos a aplicar a un selector.
- La lista consta de un conjunto de pares de elementos separados por el carácter *";"* formados por el nombre de una propiedad y un valor y separados por el carácter *":"*.
- Sintaxis:
  - Selector {propiedad:valor; propiedad:valor; ... propiedad:valor}

Ejemplo



- Las hojas de estilos se pueden aplicar a varios documentos.

Selectores

- Se especifican de lo más genérico a lo más específico:
  - **Universal (asterisco)**:
    - Se aplica a todos los elementos del documento HTML.
    - ***** { font-family: arial}
      - Útil para aplicar ciertas reglas básicas a todas las etiquetas.
  - **De Tipo o de palabra clave**:
    - Se aplica únicamente al elemento especificado: p, h1, ul, etc.
    - **h2** { color: red }

Selectores

- **De Grupo**:
  - **h1, h2** { font-family: Arial; Color: blue }

- Basado en el atributo **CLASS** (con el punto):
  - /* Válido de las dos formas */
  - p**.**inicio { color: orange }
  - **.**inicio { color: orange }
    - En este caso p.inicio es más específico, sólo valdría para párrafos de la clase inicio. En el otro caso podría haber otras etiquetas distintas que pertenecieran a la misma clase.
  - esto es un párrafo

Selectores

- Basado en el atributo **ID**:
  - #a1 { color: green }
  - p#a1 { color:green }
  - Párrafo con ID

- **Selector de atributos (CSS3)**:
  - Seleccionamos etiquetas a partir de sus atributos:
  - p[name] { font-size: 20px }
    - Selecciona los párrafos que tienen el atributo name.
  - p[name="miTexto"] { font-size: 20px }
    - Párrafos cuyo atributo name sea "miTexto".

Selectores

- **Selector de atributos + expresiones regulares (CSS3):**
  - Los atributos también se pueden seleccionar utilizando expresiones regulares:
    - p[name**^=** “mi”] { font-size: 20px }
      - Selecciona los párrafos cuando el atributo name **comienza** por “mi”.
    - p[name**$=** “mi”] { font-size: 20px }
      - Selecciona los párrafos cuando el atributo name **termina** por “mi”.
    - p[name***=** “mi”] { font-size: 20px }
      - Selecciona los párrafos cuando el atributo name contiene “mi”, en cualquier parte del nombre.

Selectores

- **pseudo clases (CSS3)**:
  -
    - contenido 1
    - contenido 2
    - contenido 3
    - contenido 4
  -
  - Los elementos son hermanos considerando el elemento div.
    - **p:nth-child(2)** { background: #999999; }
      - Indicar el color de fondo para el segundo elemento.
    - **.mi_clase:nth-child(2)** { background: red }
      - Hace referencia al segundo hijo de un elemento que sea de la clase mi_clase.
    - Alternativo para cada hijo (fila impar / par):
      - **p:nth-child(odd**) { background: #999999; }
      - **p:nth-child(even**) { background: #333333; }

Selectores

- Otras pseudo-clases (CSS3):
  - **first-child**: hace referencia al primer hijo.
    - p:first-child { color: red }
  - **last-child**: hace referencia al último hijo.
    - p:last-child { color: blue }
  - **only-child**: sólo cuando haya un solo hijo.
    - p:only-child { color: red }
  - **:not(p)**
    - Representa una excepción aplica a todos los elementos excepto: los párrafos, dentro también se puede indicar una clase: **:not(.mi_clase)**

Selectores

- Nuevos selectores (CSS3): **> + ~**
  - **div > p.miTexto** { … }
    - Se selecciona cuando el **elemento de la derecha tiene al de la izquierda como padre**.
  - **p + p.miTexto { … }**
    - Se selecciona cuando el elemento de la derecha es precedido inmediatamente por el de la izquierda. **Ambos comparten el mismo padre**.
  - **p ~ p.miTexto { … }**
    - Es similar a +, pero el elemento afectado no necesita estar precedido de inmediato al elemento de la izquierda. Puede haber más de un elemento afectado:

Ejemplo

```
<div id="wrapper">
    <p class="mitexto1">My text1</p>
    <p class="mitexto2">My text2</p>
    <p class="mitexto3">My text3</p>
    <p class="mitexto4">My text4</p>
  </div>
   <p class="mitexto2">Texto  fuera</p>
```

- div > p.mitexto2 { color:red; } div es padre de **mitexto2**
- p.mitexto2+ p { color:red; } mitexto3 está después de mitexto2
- p.mitexto2 ~ p { color: red; } mitexto3,mitexto4 son hermanos y siguen a mitexto2.

Selectores

- **Pseudo selectores** de primera línea y letra:
  - Elemento: first-line { declaraciones}
  - Elemento: first-letter { declaraciones }
  - Ejemplo:
    - p:first-letter { font-size: large }

- **Selector de elementos descendientes**:
  - Se aplica al elemento especificado siempre y cuando esté contenido en el elemento contenedor o padre.

  - Sintaxis:
    - elemento_contenedor elemento_descendiente { declaraciones }

Ejemplo

```
<div id="lista">
    <ul>
        <li>Elemento 1</li>
        <li>Elemento 2</li>
    </ul>
</div>

#lista ul {
    color: red;
    margin: 0;
}
```

Selectores

- **Pseudo selectores para hipervínculos**:
  - a:link
    - No visitados.
  - a:actived
    - Cuando el usuario hace click sobre el enlace.
  - a:visited
    - Cuando el enlace ya ha sido visitado.
  - a:hover
    - Cuando pasa el ratón por encima.

Ejemplo

- En la CSS tendríamos algo así:

```
a:link  {
    color: red
}

a:actived {
    color: blue
}

a:visited {
    color: green
}

a:hover {
    color: maroon
}
```

VALORES DE LAS PROPIEDADES

- **Longitud**: Cuando estamos trabajando con el tamaño de algo, por ejemplo, de las fuentes.
- **URL**: Cuando necesitamos un objeto externo, por ejemplo, una imagen.
- **Color**: Lo podemos aplicar mediante los códigos de los colores → #AA0022, (RRGGBB), RGB.
- **Palabra clave**: Hay propiedades que sólo pueden tomar determinados valores

LONGITUD

- **Absoluta**:
  - Inch(pulgadas): in (1inch=2.53cm)
  - Centimetros: cm
  - Milimetros: mm
  - Puntos: pt (1pt=1/72inch)
  - Picas: pc (1pc=12pt)

```
p{ font-size: 1in }
h1{ margin: 1.5cm }
h3 { margin: 1000mm }
```

- **Relativa**:
  - EME; em (tamaño de la letra ‘M’ que se está usando)
  - Equis; ex (tamaño de la letra 'x' en la fuente que se está usando)
  - Pixel; px

```
p{ font-size: 12px }
h1{ margin: 0.5em }
h3 { margin: 1ex }
```

- **Porcentaje**:
  - Se puede expresar un tamaño relativo como un porcentaje del valor actual de la propiedad.

```
p{ font-size: 120% }
```

URL

- Este valor se puede aplicar a las propiedades que indican la imagen del fondo, y las imagenes de una lista, *background-image*, *list-style-image* y *list-style*.

```
body  {background-image:  url(fondo-azul.jpg)  }
p { font-size: 20pt; font-family: sans-serif}
```

- El formato para especificar una URL es mediante la funcion *url()*. Como argumento recibe una dirección relativa o absoluta que puede o no ir con comillas simples o dobles.

COLOR

- Por nombre: red, blue, etc.
- Componentes hexadecimales: #FF0012.
- Componentes enteras: rgb(220, 0 , 255).
- Porcentajes: rbg(100%, 66%, 23%).

PALABRA CLAVE

- Estos son los valores más variables de los cuatro.

- Su valor depende de la propiedad que estemos aplicando, habrá propiedades que solo puedan tomar una serie de valores.

PROPIEDADES

- Display: Como se va a mostrar un elemento.
- White-space: como tratar los espacios, tabuladores y retornos de carro del fichero.
- Fuentes: estilo, tipo de letra, etc.
- Color: color del texto.
- Background: color de fondo, imagen de fondo.
- Text: apariencia del texto.
- Box: ancho, alto, bordes, margen.

DISPLAY

- La propiedad display determina como va a ser mostrado un elemento, qué espacio le va a ser asignado para su aparición en pantalla.
  - **block**: Estos elementos se separan unos de otros generalmente por un salto de línea.
  - **inline**: El elemento forma parte de la línea.
  - **list-item**: El elemento es parte de una lista. En este caso existen tres propiedades adicionales que son:
    - list-style-type: Que puede tomar los valores *disc*, (propiedad por defecto), *circle*, *square*, *decimal*, *lower-roman*, *upper-roman*, *lower-alpha*, *upper-alpha* y *none*.
    - list-style-image: Toma la URL de la imagen a mostrar.
    - list-style-position: Determina si los indicativos de la lista van a ser puestos fuera *outside* o dentro *inside* de la lista.
  - **none**: Esta propiedad será para los elementos invisibles, puesto que no se les asignará espacio para ser mostrados.

WHITE-SPACE

- Esta propiedad indica si los espacios, tabuladores o retornos de carro del fichero fuente son significativos o no.
  - **normal**: El funcionamiento es exactamente igual que en HTML y es la opción por defecto.
  - **nowrap**: Respeta los saltos de línea pero dentro de cada línea actúa de forma normal.
  - **pre**: Actúa como la etiqueta PRE en HTML. Mantiene los espacios en blanco.

FUENTES

- Se centra en el tipo y tamaño de la letra:
  - **font-family**: El valor que puede tomar es una lista separadas por comas de nombres de familias de fuentes que son (por ejemplo): *Serif*, *sans-serif*, *Monospace*, *Cursive* ,*Fantasy*, etc.. La lista proporciona simplemente un orden de preferencia a la hora de mostrar el texto. También es posible indicar el nombre concreto de la fuente detrás del nombre de su familia. El valor de esta propiedad se hereda por los elementos anidados o hijos.
  - **font-style**: Los valores disponibles son *normal*, *italic* y *oblique*(igual que italic).
  - **font-variant**: Puede tomar dos valores *normal* y *small-caps*(todas mayúsculas y las mayúsculas más grandes) .
  - **font-weight**: Los valores que puede tomar son *normal*, *bold*, *bolder*, *lighter*, *100 ... 900*.
  - **font-size**: El tamaño puede ser especificado como un tamaño normal (14pt por ejemplo) o mediante una palabra clave: xx-small, x-small, small, medium (opción por defecto), large, x-large, xx-large.

COLOR

- Esta propiedad se puede expresar mediante una palabra clave, números decimales, porcentajes o un número hexadecimal.
- Esta propiedad es heredada por los elementos hijos.

BACKGROUND

- Esta propiedad expresa el color del fondo y puede tomar valores de color o valores de la dirección de una imagen y engloba a varias propiedades individuales:
  - **background-color**: Los valores posibles son cualquier expresión de color. Tambien admite el valor *transparent* que hace que el elemento tome el color del elemento superior, es el valor por defecto.
  - **background-image**: Como valor puede tomar cualquier dirección relativa o absoluta de una imagen con la función *url()*. Si para un elemento se especifica esta propiedad y la anterior, el orden de precedencia hace que aparezca como fondo la imagen. Si esta propiedad se aplica al elemento principal, la imagen se toma como la propiedad *background* del elemento *BODY* de HTML. en cualquier otro caso se aplica sólo al espacio del elemento.
  - **background-repeat**: Esta propiedad indica cómo se utiliza la imagen de fondo asignada para rellenar el fondo. Los valores que puede tomar son: *repeat*, *repeat-x*, *repeat-y* y *no-repeat*.
  - **background-attachment**: Los posibles valores son ***scroll*** y ***fixed***. *Cuando tenemos una imagen de fondo. Con fixed la imagen permanece fija y con scroll la imagen se desplaza según movemos el scroll de la página.*
  - ***background-size*: *cover***. *Para que la imagen se reparta por todo el body.*
  - **background-position**: Indica la posición de la imagen con respecto a la ventana del navegador. Los valores que puede tomar son:
    - Palabras clave: top, center, bottom, left, rigth.
    - Porcentajes: Se puede indicar la posición indicando un porcentaje con respecto al elemento padre. Se deben dar dos porcentajes, x e y respectivamente.
    - Posiciones absolutas: utilizando dos medidas para la x y la y respectivamente.

Ejemplo

```
/* Imagen de relleno fija en el body que ocupa toda la
   pantalla */
body {
   background-position: center;
   background-image:
   url("../imagen/fotos/galeria/Desert.jpg");
   background-repeat: no-repeat;
   background-attachment: fixed;
   background-size: cover;
}
```

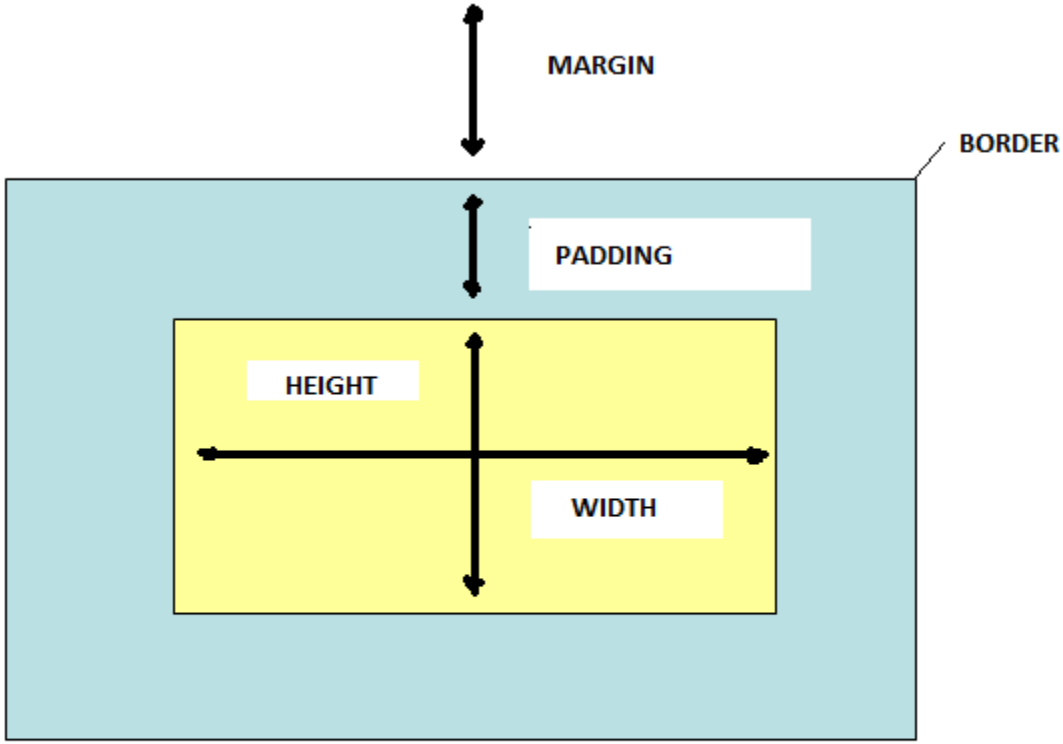
TEXT

- Este conjunto de propiedades resumen la apariencia del texto.
  - **word-spacing**: Permite expandir o contraer el espacio entre palabras. Puede tomar cualquier valor de longitud y el valor normal es 1em.
  - **letter-spacing**: Permite expandir o contraer el espacio entre letras. Puede tomar cualquien valor de longitud y el valor normal es 0.3em.
  - **text-decoration**: Puede tomar uno de estos cinco valores: *none*, *underline*, *overline*, *line-through* y *blink*. Y ninguno son autoexcluyentes excepto *blink (IE → No, Mozilla → SI)*
  - **vertical-align**: Especifica la posición relativa de un elemento respecto a la línea de escritura. Los valores que puede tomar son *baseline*, *sub*, *super*, *top*, *text-top*, *middle*, *bottom* y *text-bottom*.
  - **text-transform**: Permite indicar que tipo de letra utilizar. Los valores permitidos son *capitalize*, *uppercase*, *lowercase* y *none*.
  - **text-align**: Indica la alineación del texto *left*, *right*, *center* y *justify*. Se aplica a elementos de bloque.
  - **text-indent**: Indica la indentación de la primera línea de los elementos de bloque. Los posibles valores es cualquier medida de longitud.
  - **line-height**: Indica la distancia entre una línea de escritura y la siguiente. Puede tomar cualquier valor absoluto de longitud o un porcentaje que indica la altura con respecto al alto de la letra.

BOX

- Este conjunto de propiedades permite especificar las propiedades de ancho, alto, márgenes, bordes de los espacios para mostrar los elementos. Incluyen:
  - Propiedades de márgenes.
  - Propiedades de Bordes.
  - Prop. De ajuste.
  - Prop. De tamaño.
  - Prop. De posicionamiento.

ESQUEMA DE BOX



Box-sizing

- Tener en cuenta las siguientes medidas para calcular el ancho / alto final de un elemento.
  - Tamaño + márgenes + márgenes internos + bordes.
- En el caso de los márgenes y los bordes cuentan por doble (cada lado).
- Por ejemplo, con estas propiedades:
  - width: 100px, margin: 20, padding: 10, border: 1px
  - Total sería: 100 + 20*2 + 10*2 + 1*2 = 162 px
  - ***Se puede forzar al navegador que incluya el tamaño del padding y border. En este caso la fórmula sería: tamaño + márgenes.***

Box-sizing

- Esta propiedad es nueva en **CSS3**:
- Dos posibles valores:
  - **content-box**:
    - Los navegadores agregarán valores de padding y border al tamaño especificado por width y height.
  - **border-box**:
    - En este modo el padding y el border son incluidos dentro del elemento.

Box-sizing

```
div {
    width: 100px;
    margin: 20px;
    padding: 10px;
    border: 1px solid #000000;

    -moz-box-sizing: border-box;
    -webkit-box-sizing: border-box;
    box-sizing: border-box;
}
```

- Los prefijos moz y webkit hacen referencia a los navegadores mozilla, safari y chrome, puede que esta propiedad no esté disponible en los navegadores.
- En este caso se utilizan una serie de prefijos:
  - **-moz: Mozilla**
  - **-webkit: para Safari y Chrome**
  - **-o: Opera.**
  - **-khtml: para Konqueror.**
  - **-ms: para Internet Explorer.**
  - **-chrome: Específico para Google Chrome.**

BOX: MARGENES

- Especifican la cantidad de espacio que se añade a un bloque fuera de su borde. Se especifican mediante:
  - margin-top
  - margin-bottom
  - margin-left
  - margin-right

- La propiedad margin incluye a las otras cuatro recibiendo cuatro valores top, bottom right y left.

- La consecuencia de estas propiedades depende de si el elemento es de bloque o es un elemento in-line.

BOX: BORDES

- Los bordes se gestionan con estas 3 propiedades:
  - **border-style**: que puede tomar los valores
    - dotted .....
    - dashed - - - - -
    - solid ----------
    - double ==========
    - froove (no soportado)
    - ridge (con volumen)
    - inset (borde 3D hacia dentro)
    - outset(borde 3D hacia fuera)
  - **border-width**: Que engloba *border-bottom-width*, *border-top-width*, *border-right-width* y *border-left-width*. Cada uno toma un valor de longitud.
  - **border-color**: Especifica los colores de los bordes, y admite, uno, dos, tres o cuatro valores de colores.

BOX: BORDES

- Las propiedades de los bordes se agrupan tambien como:
  - border-top
  - border-right
  - border-bottom
  - border-left
  - border
- Cada una de las cuatro primeras puede admitir valores de ancho, estilo y color para cada uno de los bordes.
- La ultima propiedad *border* admite valores de ancho, estilo y color para los cuatro bordes.

BOX: AJUSTE

- Propiedades de ajuste: Especifican el espacio entre el bloque y el borde del bloque. Se especifica mediante:
  - padding-bottom
  - padding-top
  - padding-left
  - padding-right
- Todas admiten un valor de longitud. Y se agrupan las cuatro como *padding*.

BOX: TAMAÑO

- Propiedades de tamaño: Es posible obligar a una caja a tener unas dimensiones determinadas. Las propiedades que te permiten esto son ***width*** y ***height***.
- Los valores por defecto de estas propiedades es *auto*. Se aplican a elementos de bloque.

BOX: POSICIONAMIENTO

- Su utilidad es para cambiar la posición por defecto de cada caja y son ***float*** y ***clear***.
  - **float**: Puede ser inicializada a *none*, *left* o *right*. Hace flotar un elemento a la derecha o izquierda de su posición dentro del texto. Se utiliza mucho con imágenes.
  - **clear**: Especifica si un elemento puede o no tener y donde elementos flotantes. Los valores posibles son : *none*, *left*, *both* y *right*. Si se especifica que no puede tener elementos flotantes a la derecha, *clear:right* por ejemplo y los tiene el texto se moverá hacia abajo hasta que el elemento flotante desaparezca.

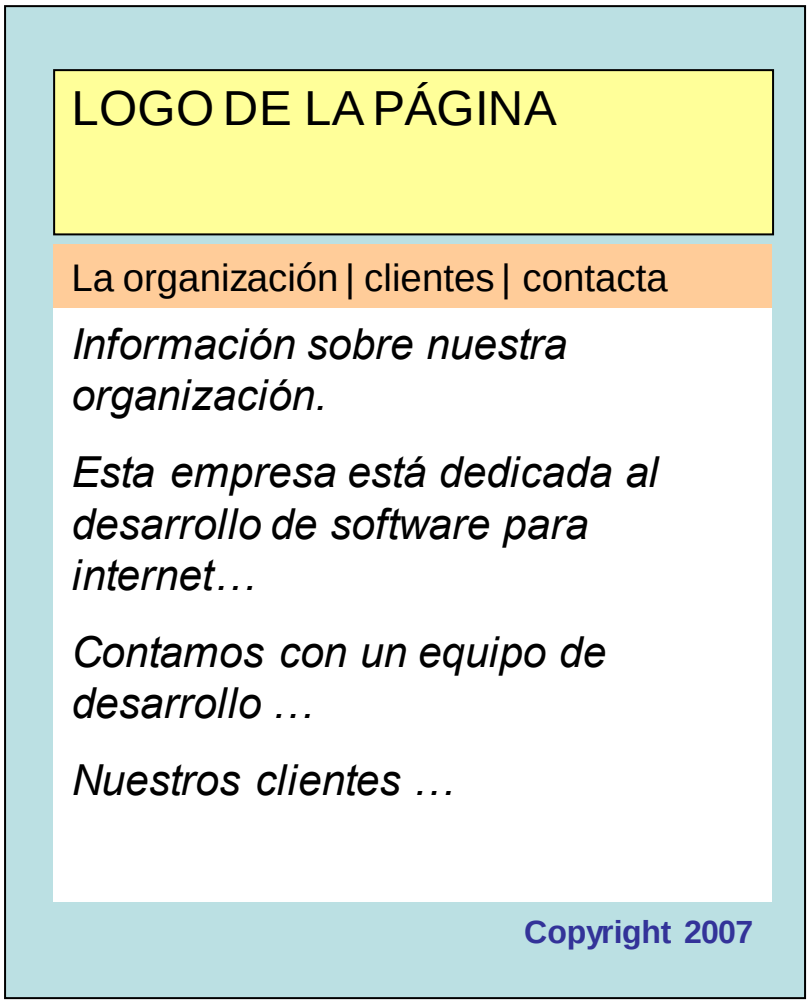
ENLACES

- Podemos aplicar estilos a los 4 estados de los enlaces:
  - a:link → Hipervínculos no visitados.
  - a:actived → Cuando el usuario hace click sobre el enlace.
  - a:visited → Cuando el enlace ya ha sido visitado.
  - a:hover → Cuando el ratón pasa por encima del enlace.

MAQUETACIÓN

- La maquetación de las páginas la vamos a realizar con capas → elementos DIV.
- No vamos a utilizar tablas a no ser que queramos expresamente una tabla.
- Normalmente tendremos una capa contenedor y cada área de nuestra página estará repartida a su vez en capas.

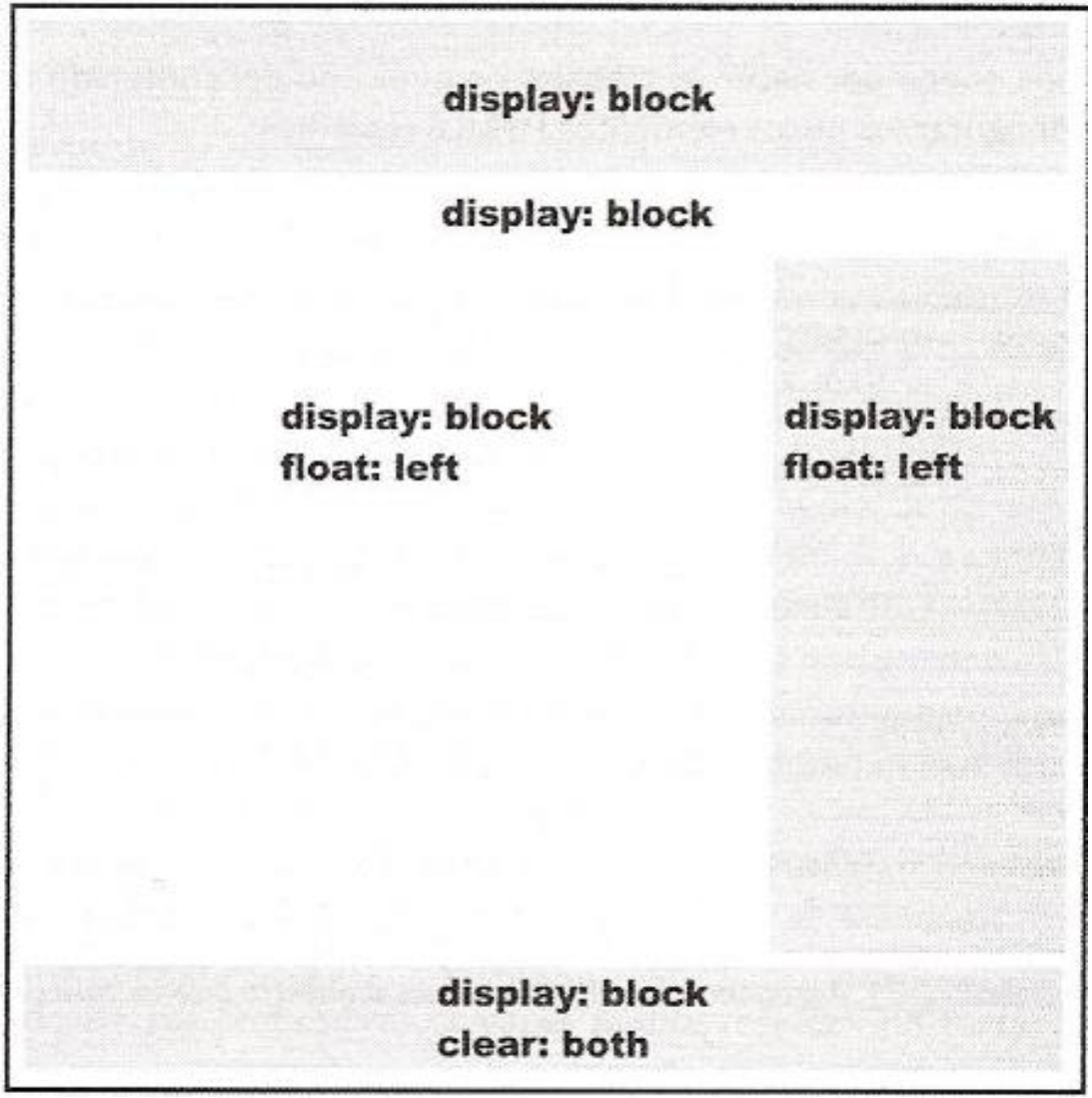
EJEMPLO




```
<body>
    <div id="contenedor">
        <div id="cab"></div>
        <div id="menu_nav"></div>
        <div id="cuerpo"></div>
        <div id="pie"></div>
        </div>
</body>
```

Aplicamos estilos a cada capa { tamaño, color, letra, etc. }

Con la estructura HTML5



Centrar contenedores

- Hay varias formas de centrar contenedores:

```
#contenedor {
    margin: 0 auto;
    width: 500px;
    text-aling:center;
}
```

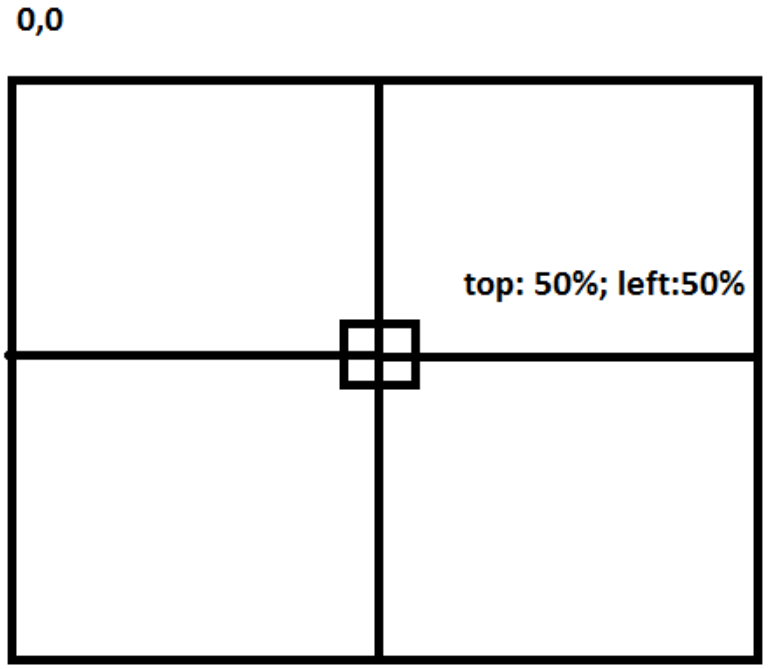
- Centrar en horizontal:

```
#contenedor {
    position: absolute;
    width: 200px;
    left: 50%;
    margin-left: -100px;
}
```

Centrar contenedores II

- Centrado en Horizontal y Vertical:

```
#contenedor {
    position: absolute;
    width: 200px;
    left: 50%;
    margin-left: -100px;
    height: 200px;
   top: 50%;
    margin-top:-100px
}
```



Centrar tabla con CSS

- En el CSS:

```
table {
    margin-left: auto;
    margin-right: auto;
}
```

Ejemplo

```
<div id="contenedor">
        <div id ="contenedor_hijo">
                <div id="contenedor_hijo2">
                        CSS es un lenguaje …
                </div>
        </div>
</div>

#contenedor { margin:0 auto; width: 50%; border:1px solid
    red }

#contenedor_hijo {margin: 20px auto; width:50%;
    border:1px solid black}

#contenedor_hijo2 {margin: 40px auto; width:50%;
    border:1px solid blue}
```

Maquetación a 2 columnas (a)

```
<div id="contenedor">
    <div class="contenedor2">
        <p class="p_left">continuación se ...</p>
        <p class="p_right">continua …</p>
    </div>
    <div class="clear"></div>

    <div class="contenedor2">
        <div class="imagen"><img src="2_columnas.PNG" /></div>
        <p class="p_right">continuación se ... </p>
    </div>
    <div class="clear"></div>

    <div class="contenedor2">
        <div class="imagen"><img src="2_columnas.PNG" /></div>
        <p>continuación se </p>
    </div>
    <div class="clear"></div>

    </div>
```

Maquetación a 2 columnas (b)

```
#contenedor { margin:50px auto 0 auto; width:80%; border:1px solid black; }

.contenedor2 { width:75%; margin: 10px auto 10px }

.p_left { float:left; width: 45%; background-color: gray; color: white}

.p_right { float: right; width: 45%; background-color: orange; color: white }

.imagen { float: left }

p { padding: 10px; text-align: justify }

.clear {clear:both}
```

Maquetación a 3 columnas

```
<div id="contenedor">
        <div class="columna"><p>continuación se indica … </p></div>
        <div class="columna"><p>continuación se indica … </p></div>
        <div class="columna"><p>continuación se indica … </p></div>
        <div class="clear"></div>
        <div id="pie">Este es el pie de la página</div>
</div>
```

```
#contenedor { margin:50px auto 0 auto; width:80%; border:1px solid black }

.columna { width:33%; float:left; }

p { padding: 10px }

.clear {clear:both}

#pie { border: 1px solid red; text-align:center; }
```

Visibilidad en Capas: display

- Permite ocultar un elemento haciendo que desaparezca de la página.
- Los otros elementos ocupan el lugar de este.
- (inline | block | none):
  - inline / block: nos permiten cambiar el estilo de visualización de los elementos.
  - none: Oculta el elemento y no se muestra.

Visibilidad en Capas: visibility

- Hace invisible un elemento creando su caja pero no muestra el contenido.
- Los otros elementos mantienen su posición como si fuera invisible.
- (visible | hidden | collapse | inherit):
  visible: por defecto, hidden: oculta.
  collapse: para tablas ocultar filas, columnas.

z-index

- Cuando tenemos varias capas solapadas podemos elegir cual es la que queremos ver.
- (auto | numero | inherit):Indicando 0 será el valor mas pequeño.
- Cuanto mas valor tiene el número mas cerca está (para que sea visible para el usuario).

Opacidad en capas

- Nos permite hacer capas sean semitransparentes.
- Esta propiedad difiere en los navegadores.
- En IE 7: opacity: entre 0 y 1.

```
opacity = (opacity == 100)?99.999:opacity;

// IE | Win
filter = "alpha(opacity:"+opacity+")";

// Safari<1.2, Konqueror
KHTMLOpacity = opacity/100;

// Older Mozilla and Firefox
MozOpacity = opacity/100;

// Safari 1.2, newer Firefox and Mozilla, CSS3
opacity = opacity/100;
```

Consejos

- A veces dejar una línea en blanco en nuestro código xHTML puede dar lugar a separaciones en el aspecto de la página:
  - Por ejemplo en IE una capa que contiene a una imagen.

- ***EJEMPLOS****: Crear menús verticales y horizontales con listas.*

Enlaces

- Validador CSS:
  - http://jigsaw.w3.org/css-validator/