CSS SEGUNDA PARTE

1. HERENCIA DE PROPIEDADES DE ESTILO.

Todas las etiquetas HTML que estén contenidas dentro de otras etiquetas HTML heredan sus estilos. Así, por ejemplo, si se define un estilo en la etiqueta **<body>,** todas las etiquetas que se incluyen dentro heredan el estilo.

Ejemplo. (herenciaEstilos)

```
<!DOCTYPE html>
<html>
<head>
        <title>Ejemplo de herencia de estilos</title>
        <style type="text/css">
                body { font-family:
                        Verdana;
                        color: blue; }
                p { color: red; }
</style>
</head>

<body>
  <h1>Titular de la página</h1>
  <h2>Encabezamiento 2</h2>
  <p>Párrafo.</p>
  <h3>Encabezamiento 3</h3>
</body>
</html>
```

Como se puede comprobar, en la etiqueta **<body>** se define el tipo de letra **Verdana** y color azul, y todas las etiquetas contenidas en **<body>** heredan este estilo. Ahora bien, si no se requiere heredar alguna característica, ésta deberá ser redefinida. En el ejemplo siguiente, el párrafo **<p>** redefine el color a rojo.

2. DEFINICIÓN DE UN ESTILO EN FUNCIÓN DEL CONTEXTO

Este otro recurso que posee las Css es la definición de un estilo para una marca HTML siempre y cuando la misma esté contenida por otra marca determinada. Sólo en ese caso el estilo para dicha marca se activará.

Ejemplo. (Ejemplo 20)

```
<!doctype html>
<html lang="es">
<!--Estilo en función del contexto-->
<head>
<style type ="text/css">
        p{color:red}
        p span{color:blue}
        h1 span{color:pink}
</style>
</head>
<body>
        <p>Curso <span>primero</span> de <span>ciclo</span>
        <h1>Ciclo formativo <span>ASIR y DAM</span></h1>
        <p>Lenguajes de marca</p>
</body>
</html>
```

3. DEFINICIÓN DE ESTILOS POR MEDIO DE CLASES

En muchas situaciones una regla de estilo puede ser igual para un conjunto de marcas HTML, en esos casos conviene plantear una regla de estilo con un nombre que posteriormente se puede aplicar a varias marcas de HTML. Para hacer esto se usan las clases.

Para definir una clase se debe empezar con el carácter **punto** y seguidamente un **nombre de clase** y dentro se definen las propiedades. El nombre de la clase no puede comenzar con un número. Para indicar que una marca sea afectada por esta regla se indica la palabra **class=** y el nombre de la clase entre comillas.

Ejemplo. (ejemplo22)

```
<!doctype html>
<html lang="es">
<head>
<style type="text/css">
 body{background-color:black}
 h1.clase1,h2.clase1{background-color:red;
                     color:white
                     }
 h1.clase2{text-decoration:underline;
           text-align:right;
          }
 h1.clase3{font-style:oblique;
           letter-spacing:15;
           color:blue;
          }
</style>
</head>
<body>
<h1 class="clase1">ciclo formativo</h1>
<h1 class="clase2">Lenguajes de marca</h1>
<h1 class="clase3">curso 19-20</h1>
<h2 class="clase1">Otra cabecera diferente con la misma clase</h2>
</body>
</html>
```

4. PROPIEDADES RELACIONADAS A LA DIMENSIÓN DE UN ELEMENTO

Existen de dos propiedades fundamentales que nos permiten fijar el ancho y el alto de una marca HTML:

- **width:** Permite definir la anchura del elemento.
- **height**: Permite definir la altura del elemento.

5. PROPIEDADES RELACIONADAS CON LAS LISTAS

Las listas se utilizan para enumerar una serie de elementos. Para las listas tenemos las siguientes propiedades CSS:

list-style-type

La propiedad ***list-style-type*** permite especificar el tipo de elemento de numeración de la lista. Normalmente los navegadores muestran un círculo relleno en las listas no numeradas (las que se realizan mediante la etiqueta **ul**) y números en las listas numeradas (etiqueta **ol**).

Esta propiedad permite cambiar esos símbolos para elegir el que deseemos. La cuestión es que (aunque se podría) no es recomendable hacer que la etiqueta **ol** muestre

símbolos no numéricos y tampoco que la etiqueta **ul** muestre símbolos numéricos, para mantener la coherencia semántica en el lenguaje HTML.
En todo caso los posibles valores de esta propiedad son:

- **armenian**. La lista quedará encabezada por números del alfabeto armenio.
- **circle.** La lista quedará encabezada por círculos sin rellenar.
- **decimal**. Números decimales.
- **decimal-leading-zero**. Números decimales empezando por cero.
- **square.** Cuadrados rellenos.
- **disc**. La lista quedará encabezada por círculos rellenos. Es la viñeta por defecto
- **upper-roman.** Números romanos en mayúsculas.
- **lower-roman.** Números romanos en minúsculas.
- **lower-alpha**. Letras minúsculas del alfabeto actual.
- **upper-alpha.** Letras mayúsculas del alfabeto actual.
- **none.** Sin números.
- **georgian**. Números del alfabeto georgiano.
- **hebrew**. Números del alfabeto hebreo.
- **hiragana.** Números del alfabeto Hiragana japonés en uso actual.
- **hiragana-iroha.** Números del alfabeto japonés Hiragana clásico.
- **katakana.** Números del alfabeto Katakana japonés en uso.
- **katakana-iroha.** Números del alfabeto Katakana japonés clásico.
- **lower-greek.** Letras griegas minúsculas
- **lower-latin.** Letras minúsculas latinas.
- **upper-latin.** Letras mayúsculas latinas.
- **cjk-ideographic**. Se usan símbolos numéricos ideográficos chinos.

Ejemplo. listas

```
ol.armenian {list-style-type:armenian;
                color:red;}

<ol class="armenian">Lista 1
    <li>Brasil</li>
    <li>Uruguay</li>
</ol>
```

list-style-image

En lugar de utilizar uno de los símbolos anteriores, se puede indicar una imagen con la que se rellenará la lista. La imagen puede tener cualquiera de los formatos habituales en las páginas web **(*gif, jpg, png*).** Lógicamente el tamaño debe de ser apropiado; si es muy grande la lista quedará totalmente descuadrada.

Ejemplo. listas

```
ul.imagen {list-style-image:url(anim_021.gif)
            }

<ul class="imagen"> Lista 7
        <li>Brasil</li>
        <li>Uruguay</li>
        <li>Argentina</li>
</ul>
```

list-style-position

Solo tiene dos posibles valores referidos a la posición del texto respecto de la imagen.

- **inside**. El símbolo de numeración es interior a los márgenes del elemento en el que se coloca.
- **outside**. El símbolo de numeración es exterior.

Ejemplo. listas

```
ol.letrasmayusculas {list-style-type:upper-alpha;
                     list-style-position:inside
                    }
<ol class="letrasmayusculas"> Lista 4
    <li>Brasil</li>
    <li>BrasilBrasilBrasilBrasilBrasilBrasilBrasil</li>
    <li>Uruguay</li>
    <li>Argentina</li>
</ol>
```

list-style

La propiedad **list-style** permite en un solo golpe configurar las propiedades anteriores. Su sintaxis es:

list-style: list-style-type, list-style-position, list-style-image;

No es necesario seguir ese orden.

Ejemplo:
list–style: square url("cuadrado.gif") inside;

6. VISUALIZACIÓN DE ELEMENTOS

Propiedad display

Los distintos elementos de HTML tienen un modo por defecto de presentarse por pantalla. Así, por ejemplo, los elementos **p** o **h1**, utilizan un área rectangular, mientras que los elementos **strong** o **em** se muestran seguidos en la pantalla.
La propiedad que se encarga de estos es ***display***, esta propiedad permite establecer la forma en la que se muestra por pantalla un elemento.
Por ejemplo, en HTML los elementos de tipo ***p*** (párrafos) se entiende que se mostrarán como párrafos independientes, es decir, con cada etiqueta **p** comienza un nuevo párrafo o bloque. No irán seguidos dos elementos de párrafo consecutivos.
Gracias a **display** el comportamiento de un elemento se puede modificar.

Principales valores de display.

- **none**. El elemento queda oculto (también sus elementos hijos). Es usado en JavaScript para hacer aparecer y desaparecer un elemento.
- **inline**. Los elementos se muestran sin romper la línea actual, es decir, seguidos. Se muestran dentro de la línea. Por lo tanto, podemos colocar dos elementos inline seguidos.

  Los elementos ***inline*** ignoran los márgenes superior e inferior, pero sí reconocen los rellenos y los márgenes izquierdo y derecho, así como los bordes y los colores e imágenes de relleno. No hacen caso de las propiedades **width**, **height.**

  Ejemplos de elementos que son por defecto de este tipo serían: **a, span**, **small**, **strong**, **em**, **mark**, **abbr**, **kbd**, etc.

- **block**. Son elementos que se muestran dentro de una caja rectangular. Rompen la línea actual y, por defecto, no admiten otro elemento a su lado (independientemente de su tamaño o alineación).
  Elementos por defecto, de este tipo son: **p**, **h1**, **div**, **section**, **article**, **nav**, **figure**, **ul**, etc.
- **inline-block**. El elemento forma un bloque rectangular, pero es interior a su contenedor (no rompe la línea actual). Sí admiten elementos a su lado. Admiten manipular las propiedades **width** y **height** lo que demuestra su parecido con el valor **block**.
  Ejemplos de elementos de este tipo son: **button**, **image**, **input**, **textarea**, **select**, etc.

Ejemplo. (display.html)

```
<!doctype html>
<html lang="es">
<head>
<style type="text/css">
p{background-color:red;
  color:white;
  display:inline; /*No admite ancho ni alto. Además, no agrega salto al final*/
 }
  /*El color de fondo ocupa sólo el espacio del texto, es decir se convierte en un elemento de línea*/

h3 { display:inline-block;/*Es un elemento de bloque, pero va seguido*/
      height:50px;
      width:30%;
      padding-top:20px;
      background-color:olive;
  }
a {background-color:red;
    display:block; /*Ahora tenemos un elemento de línea que lo convertimos en un elemento de bloque*/
    width:20%;
    margin:2px;
  color:white;}
</style>
</head>
<body>
          <p>hola mundo</p>
          <p>adiós mundo</p>
          <br>
          <h3>Ciclo formativo</h3>
          <h3>Lenguajes de marcas</h3>
          <br>
          <a href="http://www.google.es">Buscador</a>
          <a href="http://www.terra.es">Noticias</a>
</body>
</html>
```

7. DIVISIONES (DIV) O CAPAS

La etiqueta ** ** define una división. Esta etiqueta permite agrupar varios elementos de bloque (párrafos, encabezados, listas, tablas, divisiones, etc). Son zonas de la página independientes que se pueden posicionar. En principio, los navegadores no muestran nada especial cuando se crea una división, salvo que se dé formato a la división con la hoja de estilo.
En los ejemplos siguientes hay dos divisiones que incluyen párrafos.

Ejemplo. ejemploCapa.html

```
<!doctype html>
<html lang="es">
<head>
        <style>
                div{background-color:olive;
                        border:dotted 2px;
                        color:white;
                        padding-left:10px}
        </style>
</head>
<body>
<div>
  <p>Hola mundo</p>
  <p>adiós mundo</p>
</div>
<br>
<div>
  <p>¿Qué es el hardware? - El que recibe los golpes cuando falla el software.</p>
  <p>¿Qué es una gamba tirando piedras?<br>
          es una gamberra</p>
  <p>¿Qué diferencia hay entre un árbol y un borracho?<br>
        Que el árbol comienza en el suelo y termina en la copa;
        y el borracho empieza por la copa y acaba en el suelo.</p>
</div>
</body>
</html>
```

8. PSEUDOCLASES

Las pseudoclases son unas clases especiales, que se refieren a algunos estados especiales del elemento HTML. Las pseudoclases se definen añadiendo **dos puntos** antes de la pseudoclase concreta. La sintaxis es la siguiente:

```
etiqueta:pseudoclase {propiedad}
```

Las más usadas son las que se utilizan con la marca <a> (ancla).

Pseudoclases más importantes:

- **:link** :Enlace sin visitar
- **:visited** :Enlace visitado
- **:hover** :Enlace que tiene la flecha del mouse encima
- **:active** :Es la que tiene foco en ese momento.
- **:first-line:** primera línea de texto.
- **:first-letter:** primera letra de texto.
- **:first-child** hace referencia al primer elemento de un tipo contenido dentro de otro. Es necesario que el tipo del primer elemento hijo sea el indicado en la hoja de estilo. (Es decir, debe ser el primer hijo para que se aplique la propiedad).

Es importante hacer notar que el orden en que se definen las pseudoclases es fundamental para su funcionamiento (**debe respetarse el orden: link-visited-hover-active)**

Ejemplo

```
<!doctype html>
<html lang="es">
<head>
<title>Pseudoclases</title>

<style TYPE="text/css">

a{text-decoration:none;
  background-color:olive;
  color:white;
  font-weight:bold;
  padding:5px;
  margin-bottom:5px;
  border-bottom:3px solid black;
  border-top:3px solid black;
  }

a:hover{background-color:white;
          color:black;
         }
a:active{background-color:green;
         }
p:first-letter{font:250% Verdana;
               color:red;}
h2 span:first-child{color:red;}//Se aplica al primer hijo (span) que esté dentro de un h2
</style>
</head>
<body>
         <a href="https://www.google.ES">enlace google</a>
         <a href="http://www.terra.es"> enlace a terra</a>
         <p>La primera letra de este párrafo es una letra capital, es decir, una letra más grande. Para ello, en la hoja de estilo tienes que aumentar su tamaño. En este caso también hemos cambiado el color y el tipo de letra. Podíamos haber puesto más propiedades.
         </p>

         <h2>Seguimos <em>con</em> las <span>pseudoclases</span></h2>
         <!--No se aplica  la propiedad al span porque no es el primer hijo-->
         <h2>En este caso <span>vamos a cambiar</span> las propiedades <span>del hijo</span></h2>
         <h2>Podemos ver que <span>sólo</span> cambia la primera de <span>ellas</span></h2>
</body>
</html>
```

En el caso de la pseudoclase nth-child es importante tener en cuenta una serie de valores:

- **odd:** representa elementos cuya posición numérica es impar: 1, 3, 5, etc.

  :nth-child(odd)

- **even:** representa elementos cuya posición numérica es par: 2, 4, 6, etc.

  :nth-child(5n)

- **<An+B>:** representa elementos cuya posición numérica coincide con el patrón An+B, para cada entero positivo.

  Ejemplos:

:nth-child(7)
Representa el séptimo elemento.

:nth-child(5n)
Representa los elementos 5, 10, 15, etc.

:nth-child(3n+4)
Representa los elementos 4, 7, 10, 13, etc.

Ejemplo. pseudoclasesFirst.html

```
<!doctype html>
<html lang="es">
<head>
<style TYPE="text/css">
      p:first-child{background-color:tomato;}
      p:last-child{color:blue;}
      p:nth-child(odd){letter-spacing:15px;color:yellow;}

</style>
</head>
<body>
<div>
     <p>Hola mundo</p>
     <p>Hola mundo</p>
     <p>Hola mundo</p>
     <p>Hola mundo</p>
</div>
     <h2>Adios mundo</h2>
     <h2>Adios mundo</h2>
</body>
</html>
```

9. PROPIEDADES RELACIONADAS CON LAS TABLAS

La mayoría de las propiedades de uso habitual en CSS (bordes, colores, tipos de letra, alineaciones...) son perfectamente aplicables a las tablas. Sin embargo, hay propiedades CSS que, explícitamente, sirven para modificar la forma de visualizar los elementos de una tabla.

La propiedades para tablas son:

border-collapse

Permite indicar cómo manejar los bordes adyacentes entre elementos. Posibles valores:

- **collapse**. Se unen los bordes adyacentes para formar un único borde. Es decir, no hay espacio entre bordes adyacentes.
- **separate**. Los bordes adyacentes se mantienen separados. Es el valor por defecto.

Es una propiedad de la etiqueta **<table>.**

border-spacing

Esta propiedad determina el espacio entre celdas contiguas en la tabla.
Si pasamos una sola unidad de medida, todos los lados de cada celda la usaran para su separación. Si pasamos dos unidades de medida el primero se usa para la separación horizontal entre celdas y el segundo para la separación vertical.

caption-side

Indica la posición del título respecto a la tabla. Los valores que puede tomar son: top, bottom.

empty-cells

Indica si deseamos mostrar las celdas vacías de la tabla. Los valores que admite son:

- show :se dibujan los bordes y fondos como en las celdas normales.
- hide: bordes o fondos no se dibujan alrededor de las celdas vacías. No funciona con los bordes colapsados.

Ejemplo.

```
<!doctype html>
<html lang="es">
<head>
<title>Tablas</title>
<style type="text/css">
table caption {caption-side:bottom;
                background-color:lightblue;
                text-align:left;
                }

table {border:red 5px solid;
       width:50%;
       empty-cells:hide;
       margin-left:auto;
       margin-right:auto; /*Estas dos sentencias sirven para centrar un elemento de bloque*/
      border-collapse:collapse;
      height:200px;
      background-color:lightgreen;
      }

table td{border:red 5px solid
          }
table th{background-color:black;
      color:white;
          }
</style>
</head>
<body>
<table>
        <caption>Título de la tabla</caption>
        <tr>
                <th>Nombre</th>
                <th>Apellidos</th>
                <th>DNI</th>
        </tr>
        <tr>
                <td>José Antonio Eduardo </td>
                <td>Sánchez Sánchez</td>
                <td></td>
        </tr>
        <tr>
                <td>Ana</td>
                <td>Redondo Gutiérrez</td>
        <td>31.255.871</td>
        </tr>
</table>
</body>
</html>
```

10. FLOAT

La propiedad **float** posiciona un elemento moviéndolo todo lo posible a la izquierda o derecha de su posición original. Los tres valores de esta propiedad se interpretan de la siguiente manera:

- **Left:** el elemento se desplaza todo lo posible a la izquierda de la posición en la que se encontraba. El resto de los elementos de la página se adaptan para mostrarse a su derecha, siempre y cuando quepa (a menos que se indique lo contrario mediante la propiedad clear).
- **Right:** el elemento se desplaza todo lo posible a la derecha de la posición en la que se encontraba. El resto de los elementos de la página se adaptan para mostrarse a su izquierda, siempre y cuando quepa (a menos que se indique lo contrario mediante la propiedad clear).
- **None:** el elemento no se desplaza respecto de su posición original. Además de ser el valor por defecto de esta propiedad, el valor none se puede utilizar para eliminar el float aplicado a un elemento mediante otros selectores.

La principal característica del posicionamiento **float** es que el resto de los elementos de la página se adaptan para fluir alrededor de la caja flotante.

Hay que tener en cuenta que esta propiedad está pensada para elementos de bloque, que son aquellos que tengan un display con valor **block** o **inline-block**.

Cuando se utiliza la propiedad float suele ser útil aplicar también la propiedad **clear**, que permite controlar el comportamiento de los elementos cuando a su alrededor tienen elementos flotantes.

Ejemplo Float (Float.html)

```
<!doctype html>
<html>
<head>
	<title>Float</title>
<style>
	div {padding:10px;
	    height:105px;
	    border:solid blue;}
	div.izquierda{float:left;
	              background-color:grey;}
	div.medio{float:left;}
	div.derecha{float:left;
	            background-color:lightgrey;}
	p.final{clear:both;
	        height:30px;
	        border:solid;
	        width:250px;} /* para evitar que se meta en los huecos que hay entre las imágenes flotadas*/
	img{width:100px;}
	img.abeja{float:left; }
	img.sol{float:right;}
</style>
</head>
<body>
	<div class="izquierda">
	<p><B> CAJA 1</B> </p>
	<p>CICLO INFORMÁTICA </p>
	<p>CICLO INFORMÁTICA </p>
	</div>
	<a href="https://www.google.es">prueba</a>
	<div class="medio">
	<p><B> CAJA 2</B> </p>
	<p>CICLO ASIR </p>
	<p>CICLO DAM </p>
	</div>
	<div class="derecha">
	<p><B>CAJA 3</B> </p>
	<p>Lenguaje html y css</p>
	</div>
	<p class="final"><B>Final sin flotar</B></p>
	<hr>
	<img class="abeja" src="abeja.png" class="imagen1"></img>

	<img class="sol" src="sol.jpg"></img>
	<p>El ordenador, también denominada computador, es una máquina digital que ejecuta comandos para convertirlos en datos convenientes y útiles que posteriormente se envían a las unidades de salida.
	</p>
</body>
</html>
```

La propiedad **clear** indica el lado o lados de una caja que no debe ser adyacente a un elemento posicionado de forma flotante con la propiedad **float**. Los cuatro valores permitidos para esta propiedad tienen el siguiente significado:

- **Left:** hace que la caja sobre la que se aplica baje hasta que su borde superior esté por debajo del borde inferior de cualquier elemento flotado a la izquierda. Es decir, “no permite elementos flotados a su izquierda”.
- **Right:** hace que la caja sobre la que se aplica baje hasta que su borde superior esté por debajo del borde inferior de cualquier elemento flotado a la derecha. Es decir, “no permite elementos flotados a su derecha”.
- **Both:** hace que la caja sobre la que se aplica baje hasta que su borde superior esté por debajo del borde inferior de cualquier elemento flotado a la izquierda o a la derecha. Es decir, “no permite elementos flotados a la izquierda o a la derecha”.
- **None:** es el valor por defecto que se aplica a todos los elementos y no tiene efecto sobre la posición de la caja.