**Hojas de estilos CSS3**

|  |  |
| --- | --- |
| **Hojas de estilos** | Definición: mediante una hoja de estilos se pueden definir todas las propiedades visuales del documento en general y de sus elementos en particular y aplicarlos a un documento HTML. |
| **Formas de aplicar estilos** | * Hoja de estilos interna. * Hoja de estilos externa. * Estilos en línea (utilizar con moderación). |
| **Hoja de estilos externa** | La aplicamos con <link rel="stylesheet" type="text/css" href="HojaEstilos1.css">  Sirve para definir estilos y clases en un documento .css |
| **Hoja de estilos interna** | La aplicamos con <style></style> dentro de la sección <head> de la web  Sirve para definir estilos y clases en el mismo documento .html |
| **Estilos en línea** | Atención: utilizar con moderación.  Lo aplicamos con <h1 style="color:red;margin-left:15px;">Esto es un encabezado</h1>  Sirve para poder aplicar un estilo a un único elemento. |
| **Prioridades a la hora de aplicar estilos** | 1. Estilo en línea. 2. Hojas de estilos internas o externas (dependiendo de la que aparezca más tarde en el flujo del documento, en su declaración, que será la que tenga prioridad). 3. Lo que especifique por defecto el navegador.   Ampliaremos esto más adelante. |
| **Comentarios en CSS** | Se utilizan para comentar el código y hacerlo más fácil de entender.  Ejemplo:  /\*Esto es un comentario de una línea\*/  /\*Esto es un comentario  de varias líneas\*/ |
| **Selectores** | Sirven para localizar un elemento en el documento HTML. |
| **Selector de elemento** | Se utiliza para seleccionar un elemento basándonos en su nombre de elemento. Mediante esto podemos aplicar un estilo concreto a todos los <p>, por ejemplo.  Ejemplo:  p {  color:orange;  text-align:right;  }  En este caso aplicaríamos color naranja y alineación derecha a todos los párrafos del documento. |
| **Selector de id**  **(para estilos particulares)** | Se utiliza para seleccionar un elemento único mediante su id (identificador que no puede empezar por un número).  Ejemplo:  #texto1 {  color:yellow;  text-align:center;  }  En este caso aplicaríamos color amarillo y alineación centrada un elemento con el identificador texto1.  En <body> lo aplicaríamos de la siguiente manera:  <span id=”texto1”>Texto con estilo particular</span> |
| **Selector de clase** | Se utilizar para seleccionar elementos que cumplan con un atributo de clase específico.  Ejemplo:  .izquierda{  text-align:left;  }  En este caso aplicaríamos alineación izquierda a los elementos que cumplieran la clase “izquierda”.  En <body> tendríamos esto:  <p class=”izquierda”>Párrafo que cumple la clase izquierda</p> |
| **Selector de clase para un tipo de elemento específico** | Se utilizar para especificar unos determinados elementos que se vean afectados por una clase específica.  Ejemplo:  p.centrar {  text-align: center;  color: brown;  }  En este ejemplo únicamente los elementos <p> que cumplan la clase “centrar”, quedarían alienados de manera centrada. |
| **Aplicar varias clases a un elemento** | Es posible aplicar varias clases a un elemento:  Ejemplo:  <p class=”centrar textorojo”>Este párrafo cumplirá las clases centrar y textorojoz</p> |
| **Agrupar selectores** | Podemos agrupar selectores para ahorrar código.  Ejemplo:  h1, h2, p {  color:gray;  }  En este ejemplo agruparíamos los selectores para los tipos de elemento h1, h2 y p, que tendrían color gris. |
| **Colores** | Especifican un determinado nombre de color o un valor RGB, HEX, HSL, RGBA o HSLA  HTML soporta 140 nombres de colores: <https://www.w3schools.com/colors/colors_names.asp> |
| **color** | Uso: aplica un color a un texto.  Ejemplo:  <span style=”color:Bisque;”>Span con color bisque</span> |
| **background-color** | Uso: aplicar un color de fondo a un elemento HTML  Ejemplo:  <p style=”background-color:Tomato;”>Este párrafo tiene Tomato como color de fondo</p> |
| **valores de los colores** | Un color se puede especificar en diferentes formatos:   * RGB * HEX * HSL * RGBA * HSLA   Ejemplo: color en hexadecimal  <h1 style="background-color:#6a5acd;">#6a5acd</h1>  Selector de colores de w3schools: <https://www.w3schools.com/colors/colors_picker.asp> |
| **background-image** | Uso: especifica una imagen de fondo para un elemento.  Ejemplo:  <body style=”background-image:url(‘Img/Fondo1.jpg’)">  Con selectores:  body {  background-image:url(“Img/Halo1.jpg”);  }  Podemos utilizar estas propiedades adicionales para los fondos:   * background-repeat – se utiliza para indicar cómo debe repetirse la imagen de fondo   Ejemplos:  background-repeat:repeat-x; /\*Para repetir la imagen de fondo horizontalmente\*/  background-repeat: repeat-y; /\*Para repetir la imagen de fondo verticalmente\*/  background-repeat: no-repeat; /\*Para no repetir la imagen de fondo\*/   * background-position – se utiliza para indicar la posición que deseamos que tenga la imagen de fondo.   Ejemplos:  background-position:right top;  Posibles valores:  left top  left center  left bottom  right top  right center  right bottom  center top  center center  center bottom   * background-attachment – se utiliza para fija una imagen de fondo y que no se desplace con el scroll con el resto de la página. La opción predeterminada es scroll.   Ejemplos:  background-attachment: fixed;  Podemos ahorrar código de la siguiente manera:  body {  background: red url(“imagen.jpg”) no-repeat right top;  } |
| **border-style** | Uso: especifica un tipo de borde para un elemento en cuestión.  Ejemplo:  .claseborde1{  border-style:dashed;  }  Posibles valores:  none  hidden  dotted  dashed  solid  double  groove  ridge  inset  outset |
| **border-width** | Uso: indica el ancho del borde.  Ejemplo:  Border-width: 5px;  Border-width: 2px 10px 4px 20px;  (Arriba, derecha, abajo, izquierda) |
| **border-color** | Uso: especifica el color de un borde.  Ejemplo:  border-color: red green blue yellow; |
| **Estilos individuales para bordes** | border-top-style: dotted; /\*Borde superior\*/  border-right-style: solid; /\*Borde derecho\*/  border-bottom-style: dotted; /\*Borde inferior\*/  border-left-style: solid; /\*Borde izquierdo\*/ |
| **Trabajo con bordes: formas de ahorrar código** | De esta forma podemos especificar el ancho, estilo y color del borde en una sola línea.  border: 4px solid red;  border-left: 2px dotted blue; |
| **border-radius** | Uso: sirve para crear bordes redondeados. Cuanto mayor sea el valor, más redondeado será el borde.  Ejemplo:  border-radius: 12px; |
| **box-sizing** | Uso: esta propiedad hace que el ancho del borde esté incluido en el ancho del elemento que tiene ese borde.  Ejemplo:  box-sizing: border-box  Realizar una práctica con 3 capas en paralelo: 1 con ancho 20%, otra con ancho 60% y otra con ancho 20%. |
| **margin** | Uso: permite especificar un margen en cada lado de los elementos.  Ejemplos:  #margenes1{  margin-top: 100px;  margin-bottom: 100px;  margin-right: 150px;  margin-left: 80px;  }  **Se puede ahorrar código aplicando valores a la propiedad margin.**  Ejemplo:  .margenes2{  margin: 25px 50px 75px 100px; /\* top, right, bottom, left\*/  }  .margenes3{  margin: 25px 50px; /\*top y bottom 25px, right y left 5opx\*/  .margenes4{  margin: 30px; /\*Todos los márgenes 30px\*/  }  **Valor auto:**  Centra horizontalmente un elemento dentro de su contenedor. Sólo centra los elementos que se visualizan como bloque (propiedad display:block;)  #capa1{  width: 200px;  margin: auto;  border: 3px dashed green;  }  **Valor heredado:**  Permite heredar una propiedad del elemento padre.  Ejemplo:  div {  border: 1px solid blue;  margin-left: 150px;  }  p.clasemargen1 {  margin-left: inherit;  }  <div>  <p class="clasemargen1">Este párrafo heredará la propiedad de margen de su elemento padre</p>  </div>  **Suma de márgenes (margin collapse):**  h1 {  margin: 0 0 50px 0;  }  h2 {  margin: 20px 0 0 0;  }  <h1>Encabezado 1</h1>  <h2>Encabezado 2</h2>  Entre los dos elementos tendremos 50px de separación: 50 bottom margin y 20 top margin. Es decir, tomaría el mayor de los dos valores. |
| **padding** | Uso: permite añadir espacio en el interior de un elemento con sus respectivos elementos hijos.  Ejemplo:  .clasecapa1{  padding-top: 50px;  padding-right: 30px;  padding-bottom: 50px;  padding-left: 80px;  }  <div class=”clasecapa1”>  Esta capa tendrá definidas varias propiedades padding  </div>  **Se puede ahorrar código aplicando valores a la propiedad padding.**  padding: 25px 50px 75px 100px; /\* top, right, bottom, left\*/  **Importante: el padding se suma al ancho de un elemento**  #capapadding1 {  width: 300px;  padding: 25px; /\*Izquierda y derecha\*/  }  Tendremos una capa de 350px de anchura  Ver demostración: capa de 300px de ancho vs capa de 300px de ancho con padding:25px. |
| **height** | Uso: permite establecer la altura de un elemento.  Ejemplo:  #capa1{  height: 200px;  width: 30%;  background-color: blue;  }  Establecer una altura máxima con max-height. Si el contenido del elemento supera este máximo, el elemento activará la propiedad overflow (ver ejemplo).  Por defecto tenemos max-height: none;  Ejemplo:  p.parr1{  max-height: 50px;  }  Establecer una altura mínima con min-height.  Ejemplo:  p.parr2{  min-height: 50px;  } |
| **width** | Uso: permite establecer la anchura de un elemento.  Ejemplo:  #capa2{  height: 300px;  width: 80%;  background-color: blue;  }  Establecer una anchura máxima con max-width. Si el contenido del elemento supera este máximo, automáticamente se cambiará la altura del elemento.  Por defecto tenemos max-width: none;  Ejemplo:  p.parr1{  max-width: 50px;  }  Establecer una anchura mínima con min-width.  Ejemplo:  p.parr2{  min-width: 50px;  } |
| **Modelo de cajas** | Imagen que contiene captura de pantalla  Descripción generada con confianza muy alta  Contenido: Contenido del elemento.  Padding: Espacio entre el límite superior del elemento y el contenido del elemento.  Borde: Borde que rodea al elemento.  Margen: Espacio entre el elemento y los elementos que lo rodean.  Calcular la altura y la anchura de un elemento:  div {  width: 320px;  padding: 10px;  border: 5px solid gray;  margin: 0;  }  Para calcular la anchura total del elemento:  320px (width) + 20px (left + right padding) + 10px (left + right border) + 0px (left + right margin) **= 350px** |
| **outline** | Imagen que contiene captura de pantalla  Descripción generada con confianza muy alta  La propiedad outline se encuentra aún más exterior que el borde. En este caso, no añade anchura al elemento.  Ejemplos:  p.dotted {outline-style: dotted;}  p.dashed {outline-style: dashed;}  p.solid {outline-style: solid;}  p.double {outline-style: double;}  p.groove {outline-style: groove;}  p.ridge {outline-style: ridge;}  p.inset {outline-style: inset;}  p.outset {outline-style: outset;}  **Propiedades individuales**: outline-style, outline-color, outline-offset y outline-width.  **Formas de ahorrar código**: Podemos especificar en una sola línea el estilo, anchura y color del outline.  p.ej1 {outline: dashed;}  p.ej2 {outline: dotted red;}  p.ej3 {outline: 5px solid yellow;}  p.ej4 {outline: thick ridge pink;}  Propiedad **offset**: añade un espacio entre border y outline.  p.clase1 {  margin: 30px;  border: 1px solid black;  outline: 1px solid red;  outline-offset: 15px;  } |
| **Trabajo con texto** | Tenemos varias formas de formatear el texto:  **Color del texto**: visto anteriormente.  #colortexto1 {  color: green;  }  **Alineación del texto**: izquierda, derecha y centrada.  h1.alineacion1 {  text-align: center;  }  h2.alineacion2 {  text-align: left;  }  h3.alineacion3 {  text-align: right;  }  p.alineacion4 {  text-align: justify;  }  **Decoración del texto**: sobrerrayado, tachado y subrayado.  p.sobre {  text-decoration: overline;  }  p.tachado {  text-decoration: line-through;  }  p.subrayado {  text-decoration: underline;  }  **Transformación del texto**: mayúsculas, minúsculas y capital.  p.may {  text-transform: uppercase;  }  p.min {  text-transform: lowercase;  }  p.cap {  text-transform: capitalize;  }  **Tabulado del texto**: lo especificamos con una unidad relativa o absoluta.  p.tabulado {  text-indent: 30px;  }  **Espaciado entre letras**: especifica el espacio que existirá entre los caracteres de un texto. Podrá ser un valor negativo, por ejemplo, aplicado a un h1.  h1 {  letter-spacing: -3px;  }  **Altura de línea**: especifica el espacio entre líneas, por ejemplo, para un párrafo. Si se utiliza un número, multiplica por el actual tamaño de fuente. En el siguiente ejemplo, aumentaría un 80% con respecto al tamaño de fuente.  p.muchoespacio {  line-height: 1.8;  }  Otros ejemplos (se toma como referencia el tamaño de fuente actual, muchas veces dado por el navegador):  div.a {  line-height: normal; /\*Opción por defecto\*/  }  div.b {  line-height: 1.6;  }  div.c {  line-height: 80%;  }  div.d {  line-height: 200%;  }  **Dirección del texto**: indica la dirección del texto. Por defecto es de izquierda a derecha.  Ejemplo (de derecha a izquierda):  p.alreves{  direction: rtl;  }  **Espaciado entre palabras**: especifica el espacio entre palabras. Puede tratarse de un valor negativo.  h1 {  word-spacing: 10px;  }  h2 {  word-spacing: -5px;  }  **Sombreado del texto**: establece un sombreado para el texto, tanto a nivel horizontal como vertical. También puede indicar el color del sombreado.  p {  text-shadow: 4px 2px blue;  } |
| **Fuentes** | A la hora de trabajar con fuentes, se pueden establecer una serie de propiedades.  **Familia de fuente**: establece una familia de fuente. Si el navegador no soporta la fuente especificada, lo intenta con el siguiente tipo que aparezca en la lista.  Se pueden utilizar dos grandes grupos de familias de fuente: familias de fuente genérica y familias de fuente específica. Normalmente las familias de fuente específica se indican mediante comillas dobles.  Google tiene un repositorio de fuentes muy interesante y extenso: <https://www.w3schools.com/howto/howto_google_fonts.asp>  Repositorio de Google Fonts: <https://fonts.google.com/>  Ejemplo:  Tendríamos que enlazar con la api de Google:  <link href='https://fonts.googleapis.com/css?family=ABeeZee' rel='stylesheet'>  #pruebafamiliafuente {  font-family: "Times New Roman", Times, serif;  }  **Trabajo con @font-face**  Uso: permite trabajar con fuentes instaladas por el propio desarrollador web.  Declaración del tipo de fuente:  @font-face {  font-family: miPrimeraFuente;  src: url(sansation\_light.woff);  }  Uso del tipo de fuente:  div {  font-family: miPrimeraFuente;  }  Formatos de fichero de fuentes soportados por los navegadores:    Repositorios de fuentes:  <https://www.dafont.com/es/>  <https://www.1001freefonts.com/es/>  **Estilo de fuente:** establece un estilo de fuente (normal, cursiva u oblicua).  Ejemplo:  p.normal {  font-style: normal;  }  p.italic {  font-style: italic;  }  p.oblique {  font-style: oblique;  }  **Tamaño de fuente:** esta propiedad determina el tamaño con el que una fuente se va a mostrar en el navegador.  Ejemplo:  **Tamaño con píxeles**  h1 {  font-size: 40px;  }  h2 {  font-size: 30px;  }  p {  font-size: 14px;  }  **Tamaño con em (recomendado por la W3C)**  1em es igual al tamaño actual de fuente del navegador, por lo general 16px. Permite que los usuarios puedan cambiar el tamaño de fuente en el menú del navegador (ver ejemplo con Firefox). Esta propiedad puede dar problemas en versiones antiguas de Internet Explorer.  h1 {  font-size: 2.5em; /\* 40px/16=2.5em \*/  }  h2 {  font-size: 1.875em; /\* 30px/16=1.875em \*/  }  p {  font-size: 0.875em; /\* 14px/16=0.875em \*/  }  **Solución para que se muestre el mismo tamaño de texto en todos los navegadores (debido a que em tiene problemas de compatibilidad con versiones antiguas):**  Establecemos la fuente por defecto en el body mediante porcentajes.  body {  font-size: 100%;  }  h1 {  font-size: 2.5em;  }  h2 {  font-size: 1.875em;  }  p {  font-size: 0.875em;  }  **Ancho de fuente:** permite establecer una anchura para la fuente que se utilice.  Ejemplo:  p.normal {  font-weight: normal;  }  p.negrita {  font-weight: bold;  }  **Tamaño de fuente responsivo**: este tamaño de fuente va a utilizar como referencia en tamaño de la ventana del navegador. 1vw es el 1% del tamaño de la ventana del navegador. Por ejemplo: si la ventana del navegador tiene 40cm de ancho, 1vw será equivalente a 0.4cm  Ejemplo:  <p style="font-size:10vw">Hola a todos</p>  **Propiedad font-variant:** todas las letras minúsculas son convertidas a letras mayúsculas. Normalmente los tamaños de fuente son variables para las letras mayúsculas.  Ejemplo:  p.normal {  font-variant: normal;  }  p.smallcaps {  font-variant: small-caps;  } |
| **Iconos** | Se pueden añadir iconos a una web mediante una librería de iconos como Font Awesome, por ejemplo. Lista de iconos: <https://www.w3schools.com/icons/icons_reference.asp>  Se debe añadir la siguiente línea de código en la sección <head> de la página web:  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">  Estos iconos se deben utilizar con elemento en línea, como puede ser un <span> o un <i> (Se suele utilizar para especificar un texto en lenguaje técnico).  Ejemplo de utilización de iconos:  <i class='fa fa-cc-paypal'></i>  <i class='fa fa-cc-paypal' style='font-size:24px'></i>  <i class='fa fa-cc-paypal' style='font-size:36px'></i>  <i class='fa fa-cc-paypal' style='font-size:48px;color:red'></i>  <br>  <p>Uso con un botón:</p>  <button style='font-size:24px'>Botón <i class='fa fa-cc-paypal'></i></button> |
| **Favicon – No es un elemento CSS, pero se trata de una ampliación del temario** | Uso: el favicon es un icono de página. Es una imagen asociada a una web que aparece en la pestaña, barra de direcciones y en los marcadores del navegador.  Su tamaño normalmente es: 16×16, **32×32**, 48×48, 64×64 o 128×128 píxeles con una profundidad de color de 8, 24 o 32 bits.  El fichero con la imagen debe de tener una extensión ico, png, jpg o svg.  Utilidad del favicon:   * **Asociar la web a un elemento gráfico**: imagen corporativa, logotipo… * **Identificar fácilmente la web de entre las pestañas** que un usuario tenga abiertas en el navegador. * **Identificar la web fácilmente en los favoritos del navegador, historial de navegación, páginas recientes**…   Para ahorrar trabajo, existen herramientas automáticas de generación de favicon: <http://www.genfavicon.com/es/>  **Para establecer un favicon para cualquier navegador** (suele ser estándar, aunque existen algunos dispositivos que pueden necesitar adaptaciones extras):  <link rel="icon" href="/imagenfavicon.png" sizes="32x32" type="image/png"> |
| **Enlaces** | Mediante CSS3 se pueden aplicar diferentes estilos a los enlaces. Para ello se dispone de multitud de propiedades que ya se han visto con antelación y otras que se verán a continuación.  Existen varios estados en los que se pueden encontrar los enlaces:  a:link – un enlace normal, no visitado.  a:visited – un enlace que ha sido visitado por el usuario.  a:hover – cuando un usuario sitúa el ratón encima del enlace.  a:active – un enlace en el momento en el que se hace clic sobre él.  Ejemplo:  /\* enlace no visitado \*/  a:link {  color: red;  }  /\* enlace visitado \*/  a:visited {  color: green;  }  /\* ratón situado encima del enlace \*/  a:hover {  color: hotpink;  }  /\* enlace seleccionado, es decir, con clic sobre él\*/  a:active {  color: blue;  }  **Decoración del texto:** permite personalizar el enlace, de tal manera que no aparezca subrayado, por ejemplo.  Ejemplo:  a:link {  text-decoration: none;  }  a:visited {  text-decoration: none;  }  a:hover {  text-decoration: underline;  }  a:active {  text-decoration: underline;  }  **Color de fondo de los enlaces**: mediante la propiedad background-color se pueden establecer colores de fondo para los enlaces.  Ejemplo:  a:link {  background-color: yellow;  }  a:visited {  background-color: cyan;  }  a:hover {  background-color: lightgreen;  }  a:active {  background-color: hotpink;  }  **Ejemplo avanzado con enlaces:**  <!DOCTYPE html>  <html>  <head>  <style>  a:link, a:visited {  background-color: #f44336;  color: white;  padding: 14px 25px;  text-align: center;  text-decoration: none;  display: inline-block; /\*se puede aplicar ancho y alto a pesar de ser inline\*/  }  a:hover, a:active {  background-color: red;  }  </style>  </head>  <body>  <a href="default.asp" target="\_blank">Esto es un enlace</a>  </body>  </html> |
| **Listas** | Mediante CSS3 se pueden aplicar estilos a los elementos de tipo lista.  <ol> - Lista ordenada  <ul> - Lista desordenada  <li> - Elemento de una lista  **Estilos de lista:** mediante esta propiedad se pueden establecer diferentes tipos de lista, tanto para ordenadas como para desordenadas.  Ejemplo:  ul.a {  list-style-type: circle;  }  ul.b {  list-style-type: square;  }  ol.c {  list-style-type: upper-roman;  }  ol.d {  list-style-type: lower-alpha;  }  **Imagen como icono de lista:** se puede establecer una imagen como icono de lista.  Ejemplo:  ul {  list-style-image: url('imagen.gif');  }  <ul>  <li>Walter</li>  <li>Jesse</li>  <li>Saul</li>  <li>Mike</li>  </ul>  **Posición de los iconos de lista**: se puede modificar la posición de los iconos de la lista mediante la propiedad list-style-position  Ejemplo:  ul.a {  list-style-position: outside;  }  ul.b {  list-style-position: inside;  }  **Listas con colores**: se pueden establecer listas con diferentes colores utilizando las propiedades background o background-color.  Ejemplo:  ol {  background: #ff9999;  padding: 20px;  }  ul {  background: #3399ff;  padding: 20px;  }  ol li {  background: #ffe5e5;  padding: 5px;  margin-left: 35px;  }  ul li {  background: #cce5ff;  margin: 5px;  }  **En las siguientes sesiones trabajaremos con pseudoclases y selectores más avanzados (descendentes y ascendentes).** |
| **Combinadores- Selectores** | Un combinador establece una relación entre dos selectores.  Existen 4 tipos diferentes de combinadores en CSS:  **Selectores descendentes**  Seleccionan todos los elementos que son descendientes de un elemento específico.  Ejemplo: Todos los párrafos que estén dentro de elementos div  div p {  background-color: red;}  **Selectores hijos**  Selecciona todos los elementos que son hijos inmediatos de un elemento específico.  Ejemplo: todos los párrafos que son hijos inmediatos de un elemento div.  div > p {  background-color: purple;  }  **Selectores adyacentes hermanos**  Seleccionan todos los elementos que son adyacentes (inmediatamente posterior) a un elemento específico.  Ejemplo:  div + p {  background-color: blue;  }  **Selectores generales hermanos**  Seleccionan todos los elementos que son hermanos de un elemento específico, es decir, que se encuentren en el mismo nivel.  Ejemplo: (Virgulilla: Alt Gr +4)  div ~ p {  background-color: green;  } |
| **Pseudoclases** | Una pseudoclase es usada para definir un estado de un elemento.  Por ejemplo, para los enlaces:  a:link {  background-color: yellow;  }  a:visited {  background-color: cyan;  }  a:hover {  background-color: lightgreen;  }  a:active {  background-color: hotpink;  }  **Las pseudoclases se pueden combinar con clases CSS.**  Ejemplo: sólo aplicaría esta propiedad en el estado hover de elementos que cumplan la clase resaltado.  a.resaltado:hover {  color: #ff0000;  }  **Estado hover en elementos div.**  Ejemplo:  div {  background-color: green;  color: white;  padding: 25px;  text-align: center;  }  div:hover {  background-color: blue;  }  **Caso interesante:** mostrar un párrafo oculto descendiente de un div al situar el ratón encima del div.  p {  display: none;  background-color: yellow;  padding: 20px;  }  div:hover p {  display: block;  }  <body>  <div>Sitúa el ratón encima para mostrar el párrafo  <p>¡Aparece!</p>  </div>  </body>  **Pseudoclase primer hijo:** selecciona los elementos que son primeros hijos de un elemento.  Ejemplo: **seleccionaría únicamente los párrafos que son primeros hijos de algún elemento.**  p:first-child {  color: blue;  }  Ejemplo: **seleccionaría el primer span que esté dentro de cada párrafo.**  p span:first-child{  color:yellow;  }  Ejemplo: **seleccionaría todos los <span> que se encuentren en los párrafos que son primeros hijos de algún elemento.**  p:first-child span {  color: pink;  } |
| **Pseudoelementos** | Se suelen utilizar para dar estilos a unas partes específicas de un elemento.  **Pseudoelemento first-line**: se utiliza para dar estilo a la primera línea de un elemento. Sólo se puede aplicar a elementos de bloque.  Ejemplo: la primera línea de los párrafos tendrá el siguiente estilo.  p::first-line {  color: #ff0000;  font-variant: small-caps;  }  **Pseudoelemento first-letter**: se utilizar para dar estilo a la primera letra de un elemento. Sólo se puede aplicar a elementos de bloque.  Ejemplo: la primera letra de los párrafos tendrá el siguiente estilo.  p::first-letter {  color: #ff0000;  font-size: xx-large;  }  **Los pseudoelementos se pueden combinar con clases CSS**  p.clase1::first-letter {  color: #ff0000;  font-size:200%;  }  **Se pueden combinar múltiples pseudoelementos:**  p::first-letter {  color: #ff0000;  font-size: xx-large;  }  p::first-line {  color: #0000ff;  font-variant: small-caps;  }  **Pseudolemento before**: se utiliza para insertar contenido antes de un elemento.  Ejemplo: inserta una imagen antes de los elementos h1  h1::before {  content: url(imagen1.png);  }  **Pseudoelemento after:** se utilizar para insertar contenido después de un elemento.  h2::after {  content: “ - Esto es un encabezado h2”;  }  **Pseudoelemento selection**: se utiliza para dar estilo a un elemento seleccionado por el usuario.  Ejemplo:  ::selection {  color: red;  background: yellow;  }  Código para Firefox:  ::-moz-selection { /\* Para Firefox \*/  color: red;  background: yellow;  } |
| **Display** | Display es una de las propiedades más importantes en CSS a la hora de controlar la distribución de los elementos en una página web.  Posibilidades:  p.ejemplo1 {display: none;}  p.ejemplo2 {display: inline;}  p.ejemplo3 {display: block;}  p.ejemplo4 {display: inline-block;}  Recordatorio:  **Elementos de bloque:**   * <div> * <h1> - <h6> * <p> * <form> * <header> * <footer> * <section>   **Elementos inline:**   * <span> * <a> * <img>   **Display: none** – Se utiliza para ocultar un elemento y que además no ocupe espacio en la página web.  Ejemplo:  div.clase1 {  width: 60%;  height: 250px;  background-color: red;  display: none;  }  **Modificar la propiedad display por defecto de un elemento**: gracias a CSS, esto es posible  Ejemplo: modifico la propiedad display por defecto de un elemento li (block)  li {  display: inline;  }  **Relativo a los elementos con display: inline**    Anotación respecto a los elementos inline-block (se añade un espacio de forma automática entre ellos):  <https://css-tricks.com/fighting-the-space-between-inline-block-elements/> |
| **Visibility** | Si utilizamos el valor hidden (oculto) funciona de manera similar a display: none, pero en este caso, el elemento sigue ocupando espacio en la página web. Los elementos suelen tener su propiedad visibilty con el valor visible.  Ejemplo:  h1.oculto1 {  visibility: hidden;  }  Posibilidades:  h2.clase1 {  visibility: visible;  }  h2.clase2 {  visibility: hidden;  } |
| **Position** | Es una propiedad de CSS que sirve para modificar el tipo de posicionamiento de un elemento.  Posibilidades:   * static * relative * fixed * absolute * sticky   **Position static:** se trata del valor por defecto en el posicionamiento de un elemento. Los elementos con este posicionamiento siguen el flujo normal de la página y no se ven afectados por las propiedades top, bottom, left y right.  Ejemplo:  div.estatico {  position: static;  border: 3px solid #73AD21;  }  **Position relative:** se posiciona de manera normal en el flujo del documento, sin embargo, sí se ve afectado por las propiedades top, bottom, left y right. Se posicionaría respecto a su posición normal, teniendo en cuenta los objetos que se encuentren por encima en el flujo del documento (tiene en cuenta la dimensión de esos elementos).  Ejemplo:  div.relativo {  position: relative;  left: 30px;  border: 3px solid #73AD21;  }  A lo largo de la sesión del 07-11-2019 realizamos una explicación y demostración en detalle de cómo funciona position:relative  .positionrelative1{  position: relative;/\*Se posiciona teniendo en cuenta su posición normal\*/  left:200px;  top:200px;  border:1px solid green;  }  .positionrelative2{  position: relative;  left:200px;  top:0px;  border:1px solid yellow;  }    **Position fixed**: se posiciona de manera relativa a la ventana del navegador y se adapta al scroll vertical que se aplique, es decir, el elemento se desplazará si hacemos scroll vertical. El objeto será posicionado con las propiedades top, bottom, left y right.  Ejemplo:  div.fixed {  position: fixed;  bottom: 0;  right: 0;  width: 300px;  border: 3px solid #73AD21;  }  **Position absolute**: el elemento se posicionará en relación a la ventana del navegador o su padre más cercano (al contrario que fixed, que siempre lo hará en relación a la ventana del navegador).  Ejemplo:  div.relativa {  position: relative;  width: 400px;  height: 200px;  border: 3px solid #73AD21;  }  div.absoluta {  position: absolute;  top: 80px;  right: 0;  width: 200px;  height: 100px;  border: 3px solid #73AD21;  }  **Position sticky**: se trata de una mezcla entre position relative y position fixed. Se posiciona de manera relativa hasta que su posición es alcanzada por la parte superior del navegador y es “arrastrada” por éste.  Versiones antiguas de Edge no soportan este tipo de posicionado. Safari requiere un código personalizado mediante webkit, tal como se muestra en el siguiente ejemplo.  Ejemplo:  div.sticky {  position: -webkit-sticky; /\* Safari \*/  position: sticky;  top: 0;  background-color: green;  border: 2px solid #4CAF50;  }  **Posicionamiento de elementos por niveles: z-index**  Mediante la propiedad z-index, se puede variar el orden en el que aparecerán unos elementos que se encuentran encima de otros. Cuanto mayor sea el valor de z-index, más prioridad le dará al elemento sobre el que se ha aplicado para aparecer sobre los demás.  Ejemplo:  div.capa1 {  position: absolute;  left: 0px;  top: 0px;  z-index: -1;  }  div.capa1 {  position: absolute;  left:30px;  top: 50px;  z-index: 1;  }  Ver ejemplo con imágenes y texto:  <!DOCTYPE html>  <html>  <head>  <style>  .contenedor {  position: relative;  }  .centrado {  position: absolute;  left: 0;  top: 50%;  width: 100%;  text-align: center;  font-size: 18px;  }  img {  width: 100%;  height: auto;  opacity: 0.3;  }  </style>  </head>  <body>  <h2>Texto sobre imagen</h2>  <p>Vamos a añadir texto sobre la imagen:</p>  <div class="contenedor">  <img src="img\_5terre\_wide.jpg" alt="Imagen1" width="1000" height="300">  <div class="centrado">Centrado</div>  </div>  </body>  </html> |
| **Overflow** | Mediante la propiedad overflow se puede especificar que el contenido del elemento aparezca cortado o se creen barras de desplazamiento para que se pueda visualizar en su totalidad.  Existen diferentes valores que puede adquirir esta propiedad:   * visible – Opción por defecto. El contenido no será cortado y sobrepasará el tamaño de su contenedor. * hidden – El contenido será cortado y no se verá en su totalidad. * scroll – El contenido será cortado, pero se añadirán barras de desplazamiento para poder visualizar el resto. * auto - El contenido será cortado, pero se añadirán barras de desplazamiento (si es necesario) para poder visualizar el resto.   Ejemplo de visible:  div {  width: 200px;  height: 50px;  background-color: #eee;  overflow: visible;  }  Ejemplo de hidden:  div {  overflow: hidden;  }  Ejemplo de scroll:  div {  overflow: scroll;  }  Ejemplo de auto:  div {  overflow: auto;  }  **Se pueden ajustar las barras de desplazamiento a nivel horizontal y a nivel vertical:**  div {  overflow-x: hidden; /\* A nivel horizontal, se ocultan las barras \*/  overflow-y: scroll; /\* A nivel vertical, se muestran las barras \*/  }  **Overflow:auto es un muy buen recurso cuando tenemos un elemento flotante dentro de un contenedor y el elemento flotante le supera en altura. Mediante esta propiedad el contenedor se adaptará a la altura del contenido.**  div {  border: 3px solid #4CAF50;  padding: 5px;  }  .clearfix {  overflow: auto;  }  .img2 {  float: right;  }  <div class="clearfix">  <img class="img2" src="Imagen.jpg" alt="Imagen1" width="170" height="170">  Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum...  </div> |
| **Float** | La propiedad float especifica como un elemento debe flotar en una página web. Esta propiedad es muy importante a la hora de trabajar en la distribución de un documento web. Se suele utilizar para maquetar textos e imágenes de manera simple.  Puede tomar los siguientes valores:   * left – El elemento flotará a la izquierda, dentro de su contenedor. * right - El elemento flotará a la derecha, dentro de su contenedor. * none – El elemento no flotará a ningún lado. Este es el comportamiento por defecto de los elementos * inherit – El elemento heredará el valor float de su padre.   Ejemplo de float right (ver ejemplo con texto e imagen):  img {  float: right;  }  Ejemplo de float left:  img {  float: left;  }  Ejemplo de float none (la imagen aparece donde esté situada en el texto):  img {  float: none;  } |
| **Clear** | La propiedad clear dictamina si los elementos pueden flotar o no.  Puede tomar varios valores:   * none – Permite que los elementos floten a ambos lados. Esta es la propiedad por defecto. * left – No permite que los elementos floten a la izquierda. * right- No permite que los elementos floten a la derecha. * both – No permite que los elementos floten a ambos lados. * inherit – Hereda el valor clear de su padre.   Ejemplo:  <head>  <style>  .div1 {  float: left;  width: 100px;  height: 50px;  margin: 10px;  border: 3px solid #73AD21;  }  .div2 {  border: 1px solid red;  }  .div3 {  float: left;  width: 100px;  height: 50px;  margin: 10px;  border: 3px solid #73AD21;  }  .div4 {  border: 1px solid red;  clear: left;  }  </style>  </head>  <body>  <h2>Sin clear</h2>  <div class="div1">div1</div>  <div class="div2">Este ejemplo no utiliza clear</div>  <br><br>  <h2>Con clear</h2>  <div class="div3">div3</div>  <div class="div4">Este ejemplo utiliza clear left</ div>  </body> |
| **Clearfix** | Es un truco que permite que, si un elemento flotante es más grande que el elemento que lo contiene obligue, al elemento contenedor a adaptarse a esas dimensiones.  La clase clearfix la aplicamos sobre el contenedor.  Ejemplo:  <head>  <style>  div {  border: 3px solid #4CAF50;  padding: 5px;  }  .img1 {  float: right;  }  .clearfix {  overflow: auto;  }  .img2 {  float: right;  }  </style>  </head>  <body>  <p>En este ejemplo, la imagen es más alta que el elemento que la contiene y además flota a la derecha, por lo que desborda sobre su contenedor</p>  <div>  <img class="img1" src="Imagen1.jpg" alt="Imagen1" width="170" height="170">  Texto aleatorio</div>  <p style="clear:right">Para solucionar este problema, añadimos una clase clearfix con overflow auto:</p>  <div class="clearfix">  <img class="img2" src="pineapple.jpg" alt="Pineapple" width="170" height="170">  Otro texto aleatorio</div>  </body>  **Existe una clase clearfix aún más avanzada y que suele evitar cualquier posible error (se utiliza mucho en la actualidad aunque no es necesaria al utilizar Flex y Grid):**  .clearfix::after {  content: "";  clear: both;  display: table;  }  **Otra versión moderna de clearfix:**  .clearfix{  display:flow-root;  }  **Actualmente este problema se intenta solventar usando las tendencias Flexbox y Grid para la maquetación de páginas web.** |
| **Alinear elementos** | **Alineación centrada de elementos a nivel horizontal:** para centrar horizontalmente elementos de tipo bloque, se utiliza la propiedad margin:auto  Ejemplo:  .center {  margin: auto;  width: 50%;  border: 3px solid green;  padding: 10px;  }  **Alineación centrada de texto a nivel horizontal:** se utiliza la propiedad text-align:center  Ejemplo:  .center {  text-align: center;  border: 3px solid green;  }  **Centrado de imágenes a nivel horizontal**: lo ideal es poner las propiedades margin-left y margin-right a auto y además convertir en elemento en bloque.  Ejemplo:  img {  display: block;  margin-left: auto;  margin-right: auto;  width: 40%;  }  **Alineación izquierda y derecha utilizando position:absolute (Cuidado – Si posicionamos de esta manera dentro de un contenedor, hay que recordar que el contenedor debe estar posicionado a su vez, por ejemplo, con position:relative)**  Hay que recordar que los elementos con position absolute puede situarse independientemente del flujo del documento web y se pueden superponer sobre otros.  .right {  position: absolute;  right: 0px;  width: 300px;  border: 3px solid #73AD21;  padding: 10px;  }  **Alineación izquierda y derecha utilizando float**  .right {  float: right;  width: 300px;  border: 3px solid #73AD21;  padding: 10px;  }  **Centrado vertical utilizando padding:** una solución muy rápida y sencilla es utilizar padding top y bottom:  Ejemplo con un texto dentro de un contenedor:  .center {  padding: 70px 0;  border: 3px solid green;  }  <div class="center">  <p>Este texto está centrado verticalmente mediante padding</p>  </div>  Se le puede añadir text-align:center para también centrarlo a nivel horizontal.  **Centrado vertical utilizando line-height**  <head>  <style>  .center {  line-height: 200px;  height: 200px;  border: 3px solid green;  text-align: center;  }  .center p {  line-height: 1.5;  display: inline-block;  vertical-align: middle;  }  </style>  </head>  <body>  <h2>Centrado a nivel vertical y horizontal</h2>  <p>En este ejemplo se va a realizar el centrado vertical y horizontal de un elemento:</p>  <div class="center">  <p>Este párrafo está centrado a nivel vertical y horizontal.</p>  </div>  </body>  **Centrado vertical utilizando position y transform**  <style>  .center {  height: 200px;  position: relative;  border: 3px solid green;  }  .center p {  margin: 0;  position: absolute;  top: 50%;  left: 50%;  -ms-transform: translate(-50%, -50%);  transform: translate(-50%, -50%);  }  </style>  <h2>Centrado a nivel vertical y horizontal:</h2>  <p>Centrado de un elemento a nivel vertical y horizontal mediante position y transform:</p>  <div class="center">  <p>Párrafo centrado a nivel vertical y horizontal</p>  </div>  <p>Cuidado. En versiones anteriores a IE8 puede haber problemas</p> |
| **Opacidad** | Esta propiedad especifica la transparencia/opacidad de un elemento. Puede tomar un valor entre 0 y 1.  **Transparencia de una imagen**  img {  opacity: 0.5;  filter: alpha(opacity=50); /\* Para IE8 y anteriores \*/  }  **Transparencia de una imagen cuando situamos el ratón encima de ella**  img {  opacity: 0.5;  filter: alpha(opacity=50); /\*IE8 y anteriores\*/  }  img:hover {  opacity: 1.0;  filter: alpha(opacity=100); /\*IE8 y anteriores\*/  }  **Transparencia en capas**  div {  background-color: #4CAF50;  padding: 10px;  }  div.capa1 {  opacity: 0.1;  filter: alpha(opacity=10); /\* IE8 y anteriores \*/  }  div.cap2 {  opacity: 0.3;  filter: alpha(opacity=30); /\* IE8 y anteriores \*/  }  div.capa3 {  opacity: 0.6;  filter: alpha(opacity=60); /\* IE8 y anteriores \*/  }  **Texto en capas transparentes**  <style>  div.capaconfondo {  background: url(Imagen1.jpg) repeat;  border: 2px solid black;  }  div.cajatransparente {  margin: 30px;  background-color: #ffffff;  border: 1px solid black;  opacity: 0.6;  filter: alpha(opacity=60); /\* IE8 y anteriores\*/  }  div.cajatransparente p {  margin: 5%;  font-weight: bold;  color: #000000;  }  </style>  </head>  <body>  <div class="capaconfondo">  <div class="cajatransparente">  <p>Texto sobre una capa transparente</p>  </div>  </div> |
| **Barras de navegación (menús)** | **Ampliación de enlaces:**  **Concepto de ancla (Permite saltar a una sección de la misma página, es decir, proporciona una manera de realizar enlaces internos).**  <p><a href="#aqui">Saltar hasta el ancla que hemos fijado</a></p>  <h2 id="aqui">Destino del enlace</h2>  <p>En enlace nos llevará hasta aquí</p>  **Lista de enlaces**  <ul>  <li><a href="http://www.google.com/ ">Google</a></li>  <li><a href="http://www.bing.com/ ">Bing</a></li>  <li><a href="http://www.search.yahoo.com/">Yahoo</a></li>  <li><a href="http://www.yandex.ru/ ">Yandex</a></li>  </ul>  **Barra de navegación vertical (indicando la página en la que estamos)**  <head>  <style>  ul {  list-style-type: none;  margin: 0;  padding: 0;  width: 200px;  background-color: #f1f1f1;  }  li a {  display: block;  color: #000;  padding: 8px 16px;  text-decoration: none;  }  li a.activo {  background-color: #4CAF50;  color: white;  }  li a:hover:not(.activo) { /\*Cuando situamos el ratón encima, siempre que no sea sobre un elemento con la clase activo\*/  background-color: #555;  color: white;  }  </style>  </head>  <body>  <h2>Barra de navegación vertical</h2>  <p>En este ejemplo tenemos un elemento con una clase activo que va a actuar de manera diferente al resto de elementos de la lista</p>  <ul>  <li><a class="activo" href="#inicio">Inicio</a></li>  <li><a href="#lanzamientos">Lanzamientos</a></li>  <li><a href="#analisis">Análisis</a></li>  <li><a href="#nosotros">Acerca de</a></li>  </ul>  </body>  **Barra de navegación horizontal (utilizando elementos en línea)**  <head>  <style>  ul {  list-style-type: none;  margin: 0;  padding: 0;  }  li {  display: inline;  }  </style>  </head>  <body>  <ul>  <li><a href="#elemento1">Elemento 1</a></li>  <li><a href="#elemento2">Elemento 2</a></li>  <li><a href="#elemento3">Elemento 3</a></li>  <li><a href="#elemento4">Elemento 4</a></li>  </ul>  </body>  **Barra de navegación horizontal (utilizando elementos flotantes)**  <head>  <style>  ul {  list-style-type: none;  margin: 0;  padding: 0;  overflow: hidden;  }  li {  float: left;  }  li a {  display: block; /\*Para que el padding haga efecto. ¡Ojo! ¡No cambia el tipo de elemento que es!\*/  padding: 8px;  background-color: #dddddd;  }  </style>  </head>  <body>  <ul>  <li><a href="#elemento1">Elemento 1</a></li>  <li><a href="#elemento2">Elemento 2</a></li>  <li><a href="#elemento3">Elemento 3</a></li>  <li><a href="#elemento4">Elemento 4</a></li>  </ul>  <p>Es importante especificar el DOCTYPE para que los elementos flotantes funcionen de manera correcta</p>  <p>El elemento ul lleva el overflow activado para impedir que los elementos que contiene desborden y destrocen la armonía del menú horizontal</p>  </body>  **Menú sticky**  <head>  <style>  body {  font-size: 28px;  }  ul {  list-style-type: none;  margin: 0;  padding: 0;  overflow: hidden;  background-color: #333;  position: -webkit-sticky; /\* Safari y otros \*/  position: sticky;  top: 0;  }  li {  float: left;  }  li a {  display: block;  color: white;  text-align: center;  padding: 14px 16px;  text-decoration: none;  }  li a:hover {  background-color: #111;  }  .active {  background-color: #4CAF50;  }  </style>  </head>  <body>  <div class="header">  <h2>Haz scroll vertical</h2>  <p>Hay que aplicar scroll vertical para poder comprobar el efecto sticky.</p>  </div>  <ul>  <li><a class="active" href="#elemento1">Elemento 1</a></li>  <li><a href="#elemento2">Elemento 2</a></li>  <li><a href="#elemento3">Elemento 3</a></li>  </ul>  <h3>Ejemplo de menú sticky/h3>  <p>La barra de navegación <strong>se pegará</strong> al parte superior de la ventana del navegador cuando hagamos scroll vertical hacia abajo</p>  <p><strong>Recordar:</strong> Es importante utilizar -webkit- para asegurarnos la compatibilidad con todos los navegadores</p>  <p>Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>  <p>Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>  <p>Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>  <p>Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>  <p>Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>  <p>Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>  <p>Lorem ipsum dolor sit amet, illum definitiones no quo, maluisset concludaturque et eum, altera fabulas ut quo. Atqui causae gloriatur ius te, id agam omnis evertitur eum. Affert laboramus repudiandae nec et. Inciderint efficiantur his ad. Eum no molestiae voluptatibus.</p>  </body>  **Veremos más ejemplos cuando apliquemos media queries.**  **Una de los pilares a partir de ahora será trabajar con Grid y Flexbox, bases de la maquetación web en la actualidad y nos ahorrarán muchos problemas al evitar elementos flotantes innecesarios.** |
| **Trabajo con medidas relativas** | **En construcción**  **Unidad em**  Se trata de una unidad relativa que toma como referencia el tamaño de fuente base de su contenedor. Si no existe un contenedor, tomará como tamaño base el configurado en el navegador, normalmente 16px.  Esto nos puede dar quebraderos de cabeza si utilizamos múltiples contenedores.  **Unidad rem**  Se trata de una unidad de una unidad relativa que toma como referencia el tamaño de fuente base configurado en el navegador (raíz, de ahí rem – la r es de root).  Para trabajar con texto puede ser mucho más útil usar rem que em porque puede evitarnos muchos quebraderos de cabeza.  Hay que tener en cuenta que rem no funciona en IE 8 y anteriores, por lo que tenemos que utilizar retrocompatibilidad, es decir, asociar también un código legible por los navegadores antiguos:  font­size: 16px;  font­size: 1rem;  **Más adelante veremos otras unidades relativas: vw y vh.** |