CSS

"Introducción a la Programación Web"

Introducción a CSS

CSS: Cascading Style Sheets, es decir, Hojas de Estilo en Cascada

Es un lenguaje creado para controlar el aspecto o presentación de los elementos de HTML.

Un archivo de tipo CSS no es más que un archivo simple de texto cuya extensión es .css.
Se pueden crear todos los archivos CSS que sean necesarios y cada página HTML puede enlazar tantos archivos CSS como necesite.

En términos simples, el lenguaje CSS es el que describe cómo nuestros elementos en HTML serán mostrados en una pantalla de computadora, celular u otro dispositivo multimedia.

Diferencia entre HTML y CSS

La diferencia crucial es:

- **que el HTML se utiliza para la creación de las páginas web**

- **y el CSS se utiliza para controlar el estilo y el diseño de las páginas web.**

Que es un Selector en CSS

Los selectores CSS son herramientas utilizadas para definir el estilo que quieres dar a tus elementos en CSS.

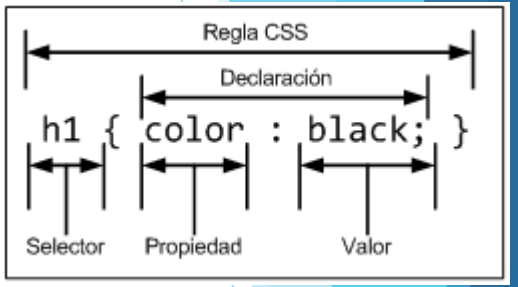
Estructura básica

Regla: Cada regla está compuesta de una parte llamada "selectores", un símbolo de "llave de apertura" ({), otra parte denominada "declaración" y por último, un símbolo de "llave de cierre" (}).

Selector: indica el elemento o elementos HTML a los que se aplica la regla CSS. El selector indica "a quién hay que hacérselo“

Declaración: especifica los estilos que se aplican a los elementos. Está compuesta por una o más propiedades CSS. La declaración indica "qué hay que hacer"

Propiedad: característica que se modifica en el elemento seleccionado, como por ejemplo su tamaño de letra, su color de fondo, etc.
Valor: establece el nuevo valor de la característica modificada en el elemento.



Como aplicar los selectores

Existen 3 formas para incluir selectores:

1-Inline Styles o Estilo de Línea:

```
<p style="color:green;">Texto en verde</p>
```

2-Etiqueta <style>:

```
<head>
    <style type="text/css">
        h1 {
        color: red;
        }
    </style>
</head>
```

3-Hoja de estilo externa:

```
<link rel="stylesheet" href="nombre-de-tu-archivo.css">
```

Tipos de selectores

- Selector universal
- Selector de tipo
- Selector de clase
- Selector de ID
- Selector de atributo
- Selector de pseudo-clase

Tipos de selectores - Universal

El asterisco (*) es el selector universal en CSS

Ejemplo de selector universal

Supongamos que queremos hacer que todos los elementos de la página sean de color naranja.

En este caso lo mejor es utilizar un selector universal

Tipos de selectores - Universal

Así es como luce un documento en HTML:

```
   <h1>All elements on the page, from the heading 1</h1>

<h2 class="pastoral">to the heading 2 with class=pastoral</h1>

<p>to the paragraph will be orange.</p>
```

Tipos de selectores - Universal

Este es el código en CSS con un selector universal para todos lo elementos:

```
    * {
color: orange;
}
```

Tipos de selectores - Universal

Este es el resultado que obtendremos al combinar los archivos:

Tipos de selectores - Tipo

La sintaxis del selector de tipo:

elemento { propiedades de estilo }

Ejemplo de selector de tipo:

Supongamos que el documento contiene párrafos (<p>) y líneas ().
Sin embargo, únicamente queremos darle una tonalidad naranja a las líneas.

Tipos de selectores - Tipo

Así es como luciría este documento en HTML:

```
    <span>One span. </span>
<p>No span. </p>
<span>Two span.</span>
<p>No span. </p>
```

Tipos de selectores - Tipo

Este es el código en CSS con un selector que define todas las líneas:

```
    span {
background-color: orange;
}
```

Tipos de selectores - Tipo

Este es el resultado que obtendremos al combinar los archivos:

One span.

No span.

Two span.

No span.

Tipos de selectores - Clase

La sintaxis del selector de clase es la siguiente

.nombre de clase { propiedades de estilo }

Ejemplo de selector de clase:

Supongamos que en esta ocasión queremos cambiar todos los elementos de la clase "pastoral" a un color naranja.

Tipos de selectores - Clase

Así es como luciría este documento en HTML:

```
    <h1>Not orange</h1>

<h1 class="pastoral">Very orange</h1>
```

Tipos de selectores - Clase

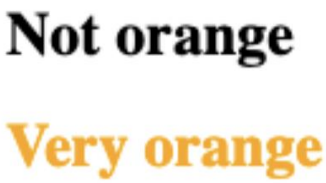
Este es el código en CSS con un selector que define todos los elementos pertenecientes a la clase "pastoral":

```
    .pastoral {
color: orange
}
```

Tipos de selectores - Clase

De acuerdo con estas reglas, la línea en h1 no debe cambiar de color, pero la segunda sí.

Este es el resultado:



Tipos de selectores - ID

La sintaxis del selector de ID es:

#nombre de ID { propiedades de estilo }

Ejemplo de selector de ID:

Cambiar el color y la alineación de un elemento nombrado con el ID "hubspot“.

Tipos de selectores - ID

Así es como luce el código en HTML:

```
<h1 id = "hubspot"> #id selector</h1>
```

Tipos de selectores - ID

Este es el código en CSS con un selector de ID que hace referencia a todos los elementos con el nombre "hubspot":

```
   #hubspot {
color:orange;
text-align:right;
}
```

Tipos de selectores - ID

Este es el resultado:

Tipos de selectores - Atributo

Los selectores de atributo están hechos para seleccionar todos los elementos que correspondan a un atributo específico o a un valor de atributo definido.

Algunas de las sintaxis disponibles para el selector de atributo son las siguientes:

- [attr] { propiedades de estilo }
- [attr=value] { propiedades de estilo }
- [attr~=value] { propiedades de estilo }
- [attr|=value] { propiedades de estilo }
- [attr^=value] { propiedades de estilo }
- [attr$=value] { propiedades de estilo }
- [attr*=value] { propiedades de estilo }

Tipos de selectores - Atributos

Selector de atributos:
Permiten seleccionar elementos HTML en función de sus atributos y/o valores de esos atributos.
[attr] : Este selector 'seleccionará' todos los elementos que contengan el atributo attr, sin importar el valor que tenga.

[attr=val] : Este seleccionará los elementos con el atributo attr, pero solo aquello cuyo valor coincida con val.

[attr^=val] : Seleccionará todos los elementos cuyo atributo attr comienza por el valor val.

[attr$=val] : Este selector elegirá todos los elementos cuyo atributo attr termina por el valor val.

Tipos de selectores - Atributos

Ejemplos de selectores de atributos:

```
/* Se muestran de color azul todos los enlaces que tengan un atributo "class",
independientemente de su valor */
a[class] { color: blue; }

/* Se muestran de color azul todos los enlaces que apunten al sitio
"http://www.ejemplo.com" */
a[href="http://www.ejemplo.com"] { color: blue; }

/* Selecciona todos los enlaces que empiezan con la palabra "mailto" */
a[href^="mailto:"] { ... }

/* Selecciona todos los enlaces que terminan en .html */
a[href$=".html"] { ... }
```

Tipos de selectores - Atributo

Ejemplo de selector de atributo:

Supongamos que queremos hacer que todos los links que contienen «**hubspot**» cambien el color de la URL a naranja.
En este caso puedes utilizar a[href*="hubspot"].

Así es como luce el código en HTML:

Tipos de selectores - Atributo

Así es como luce el código en HTML:

```
   <ul>
<li><a href="http://blog.com">blog.com</a></li>
<li><a href="http://hubspot.com">hubspot.com</a></li>
<li><a href="http://google.com">google.com</a></li>
<li><a href="http://blog.hubspot.com">blog.hubspot.com</a></li>
</ul>
```

Tipos de selectores - Atributo

Este es el código en CSS con un selector de atributo que modificará todos los elementos que contienen «hubspot» en su URL:

```
a[href*="hubspot"] {
color:orange;
}
```

Tipos de selectores - Atributo

Este es el resultado:

- blog.com
- hubspot.com
- google.com
- blog.hubspot.com

Tipos de selectores – pseudo-clase

La sintaxis del selector de pseudo clase es:

selector:puedo-clase { propiedades de estilo }

Ejemplo de selector de pseudo-clase:

En este ejemplo queremos cambiar el color de los enlaces a verde cuando el usuario haya visitado los sitios haciendo clic sobre el hipervínculo.

Los enlaces con los que no haya interactuado el usuario deberán permanecer en su color original, azul.

Además, queremos que los enlaces cambien a un color rosado cuando el usuario se desplace sobre ellos.

Tipos de selectores – pseudo-clase

Así es como luciría el código en HTML:

So you've already visited blog.hubspot.com. Why not check out our home site at hubspot.com?

Tipos de selectores – pseudo-clase

Este es el código en CSS con tres diferentes pseudo-clases:

- enlaces que no han sido visitados
- aquellos a los que se ha accedido
- aquellos sobre los que se está deslizando el usuario:

```
   a:link {
color: #0000FF;
}

a:visited {
color: #00FF00;
}

a:hover {
color: #FF00FF;
}
```

Tipos de selectores – pseudo-clase

Este es el resultado:

So you've already visited blog.hubspot.com. Why not check out our home site at hubspot.com?

Herencia

Cuando se establece el valor de una propiedad CSS en un elemento, sus elementos descendientes heredan de forma automática el valor de esa propiedad.

Todos los elementos de la página tendrían color de letra azul.

```
body { color: blue; }
```

Cascada

El orden de las reglas de CSS importa: cuando dos reglas tienen la misma especificidad, se aplica la que aparece en último lugar en el CSS.

```
h1 {
  color: red;
}
h1 {
  color: blue;
}
```

El h1 va a ser de color azul, ya que es la última regla que se aplicó.

Especificidad

La especificidad consiste en un cálculo que hace el navegador para establecer su nivel de importancia.

Por ejemplo:

- Un selector de clase es más específico que un selector de etiqueta.

- Un selector de ID es más específico que un selector de clase.

Modelo de cajas

En CSS todo elemento está enmarcado en una caja.

Hay dos tipos de cajas:

- cajas en bloque
- cajas en línea.

Este comportamiento se puede modificar con la propiedad display de css.

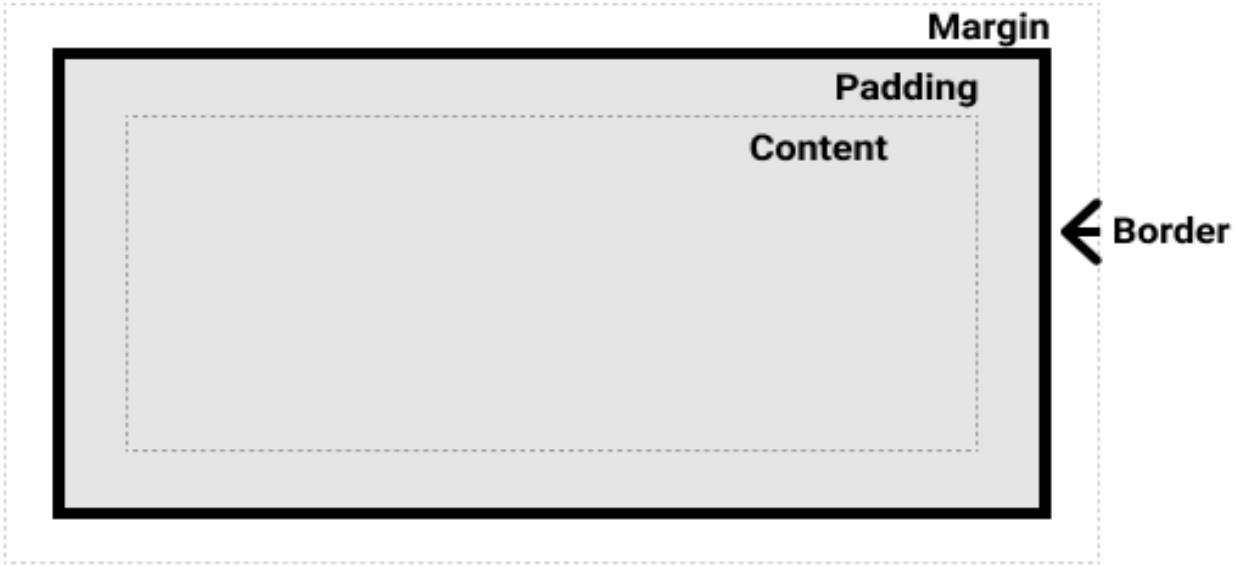
Modelo de cajas

Valores de la propiedad display :

- **block**
- **inline**
- **inline-block**
- **flex**
- **none**

Modelo de cajas

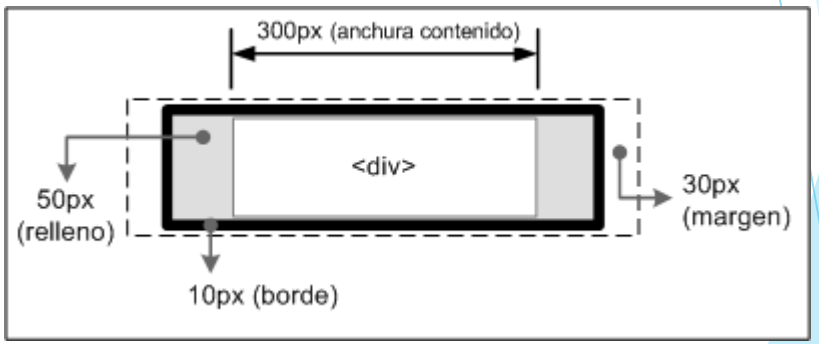
Una caja esta conformada de la siguiente manera:



Modelo de cajas

Por defecto, el ancho y el alto de un elemento no solamente se calculan teniendo en cuenta sus propiedades width y height. El margin, el padding y border también suman al cálculo final.

```
div {
  width: 300px;
  padding-left:  50px;
  padding-right: 50px;
  margin-left:   30px;
  margin-right:  30px;
  border: 10px solid black;
}
```



Ancho total de la caja: 30px + 10px + 50px + 300px + 50px + 10px + 30px = 480 píxel

Modelo de cajas

Esto se puede modificar a través de la propiedad **box-sizing** que acepta los siguientes 2 valores:

content-box: comportamiento por defecto para el tamaño de la caja.

border-box: el padding y border de ese elemento no incrementan su ancho. Si se define un elemento con un ancho de 100 pixeles, estos incluirán cualquier border o padding que se añadan, y la caja de contenido se encogerá para absorber ese ancho extra.

Propiedades de texto

font-family
Establece la familia tipográfica. Hay dos grandes grupos, sans-serif y serif

Ejemplo de serif

```
font-family: "Times New Roman", Times, serif;
```

Ejemplo de sans serif

```
font-family: Arial, Helvetica, sans-serif;
```

font-size
Establece el tamaño de fuente (px, em o %)

font-weight
Establece el peso de la fuente
Valores: normal | bold | light
Valor por defecto: normal

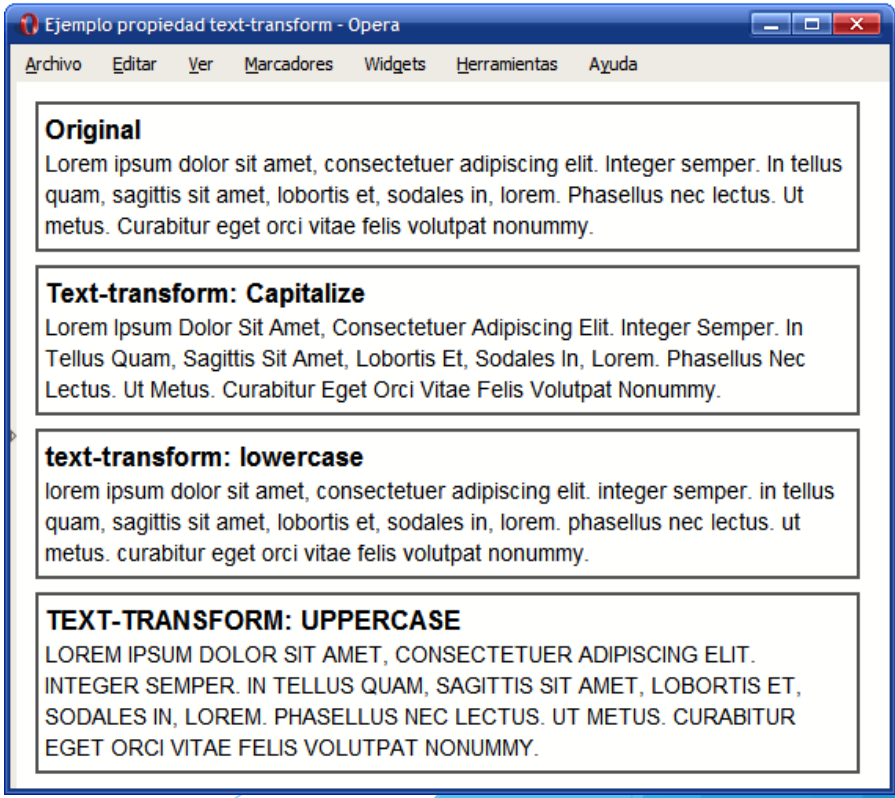
font-style
Se utiliza para especificar si la fuente es normal o italica.
Valores: normal | italic

Propiedades de texto

text-decoration
Establece la decoración del texto (subrayado, tachado).
Valores: none | underline | line-through | overline
Valor por defecto: none

text-transform
Transforma el texto original, a mayúsculas, minúsculas, etc.
Valores: none | capitalize | uppercase | lowercase
Valor por defecto: none

letter-spacing
Permite establecer el espacio entre las letras que forman las palabras del texto (interletrado)
Valores: normal | unidad de medida | inherit
Valor por defecto: normal



Propiedades color y background

color
Establece el color del texto

background-color
Establece el color de fondo
Valores: color | transparent
Valor por defecto: transparent

background-image
Establece la imagen de fondo
Valores: url | none
Valor por defecto: none

background-repeat
Establece la imagen de fondo
Valores: repeat | repeat-x (horizontalmente) | repeat-y (verticalmente) | no-repeat
Valor por defecto: repeat

background-attachment
Establece la fijación del fondo
Valores: scroll | fixed
Valor por defecto: scroll

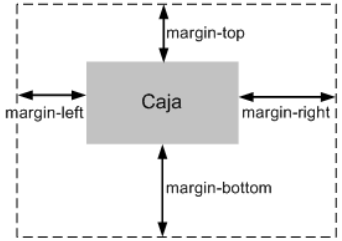
background-position
Establece la imagen de fondo
Valores: valor | top | center | bottom || left | center | right
Valor por defecto: 0% 0%

Otras propiedades

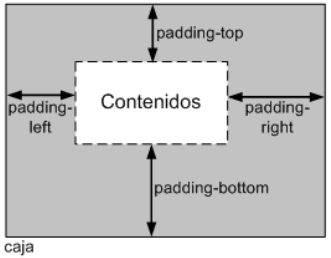
width
Establece el ancho del elemento
Valores: medida | inherit
Valor inicial: 0

height
Establece el alto del elemento
Valores: medida | inherit
Valor inicial: 0

margin
Permite especifica
entre los elemento
Valores: medida | a



padding
Establece cada uno de los rellenos
horizontales y vert
elemento
Valores: medida | i
Valor inicial: 0



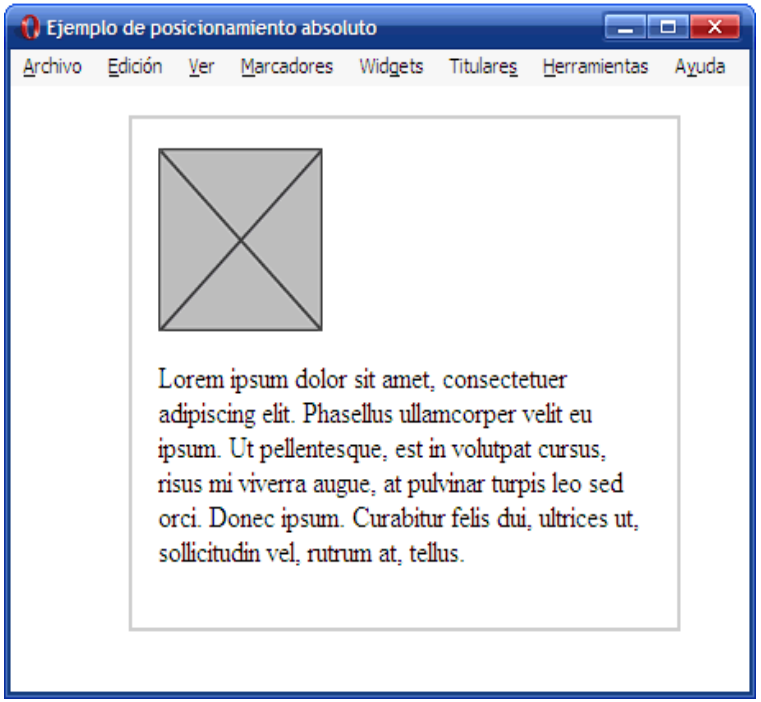
Propiedad position

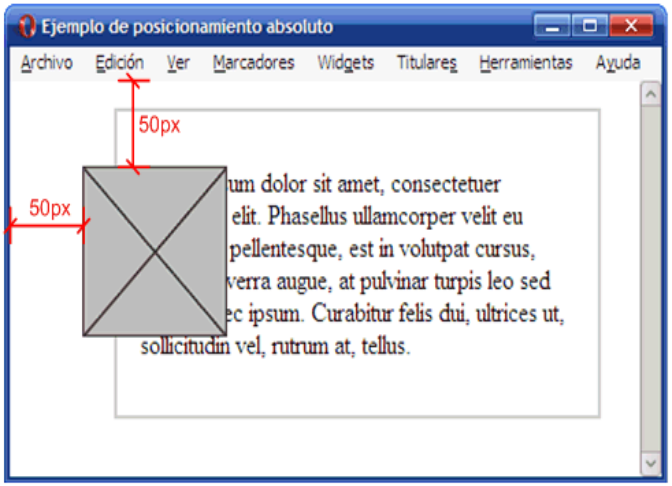
Sirve para posicionar un elemento dentro de la página.

Los posibles valores que puede adoptar la propiedad son:

- **Static**
- **Relative**
- **Absolute**
- **Fixed**

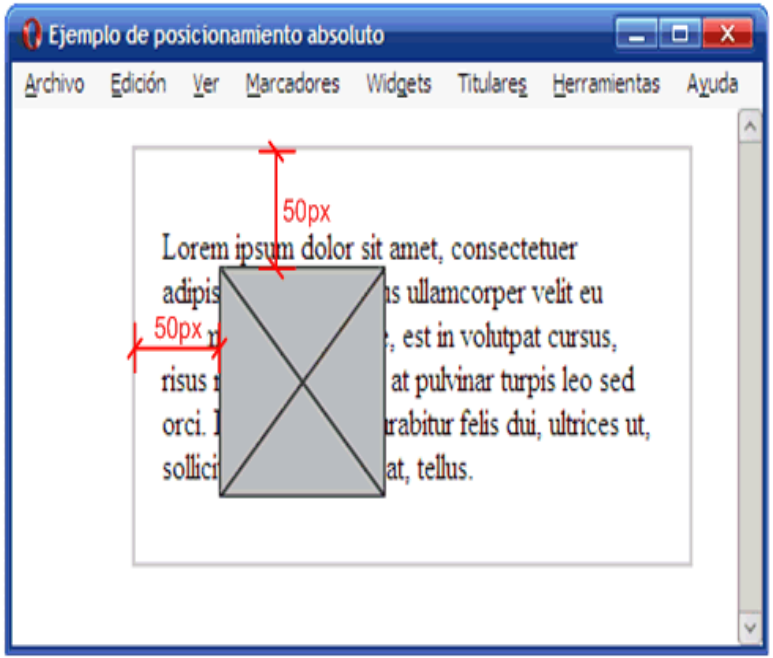
Propiedad position





En este caso la imagen tiene position absolute y toma como referencia la ventana del navegador.

Propiedad position



Al ponerle al div contenedor position relative, la imagen lo toma como referencia para su posicionamiento.

Propiedad z-index

Cuando varios elementos se superponen, z-index determina cual está por encima del otro.

Un elemento con mayor z-index se ubica por encima de uno con z-index menor.

Esta propiedad se puede aplicar solamente en los elementos que poseen algún tipo de posición que no sea static.

Referencias

- https://code.tutsplus.com/es/tutorials/the-30-css-selectors-you-must-memorize--net-16048
- https://cssreference.io/
- https://css-tricks.com/almanac/properties/
- https://www.quackit.com/css/properties/
- https://developer.mozilla.org/es/docs/Web/CSS/position#Examples
- https://www.w3schools.com/css/exercise.asp

Flexbox

¿Qué es flexbox?

Es un sistema de elementos flexibles que se caracterizan por su habilidad de alterar su ancho y alto para ajustarse lo mejor posible al espacio disponible en cualquier dispositivo.

Este tipo de modelo flexbox, nos permite controlar parámetros tales como:

- la alineación
- y la dirección de los elementos (horizontal/vertical)

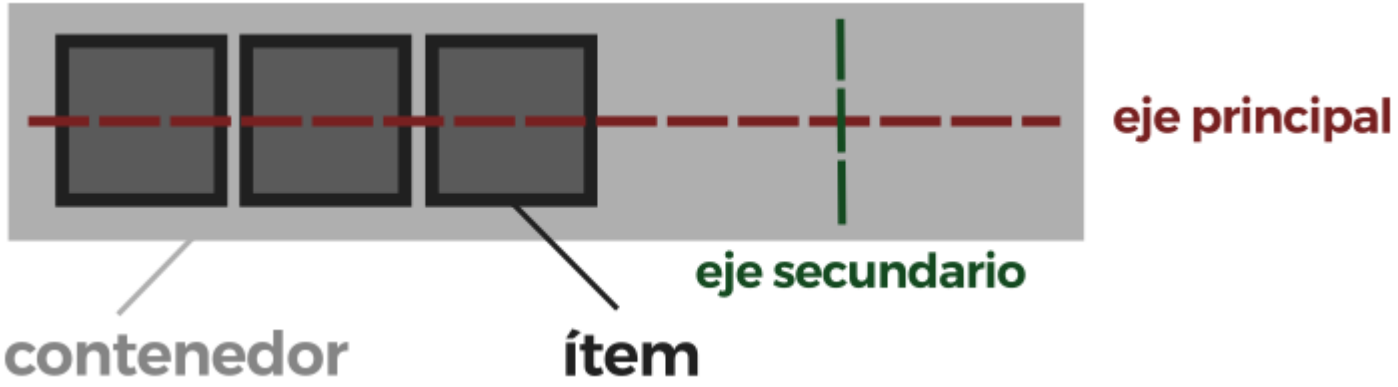
Conceptos básicos

Contenedor flexible (Flex container)
El elemento "padre" que contiene cada uno de los ítems flexibles.

- **Eje principal:** Los contenedores flexibles tendrán una orientación principal específica. Por defecto, es en horizontal (en fila).

- **Eje secundario:** De la misma forma, los contenedores flexibles tendrán una orientación secundaria, perpendicular a la principal. Si la principal es en horizontal, la secundaria será en vertical, y viceversa.

Conceptos básicos



Conceptos básicos

Ejemplo en código:

```html
<div class=”contenedor”>
    <div class="elemento">1</div>
    <div class="elemento">2</div>
    <div class="elemento">3</div>
    <div class="elemento">4</div>
    <div class="elemento">5</div>
    <div class="elemento">6</div>
    <div class="elemento">7</div>
</div>
```

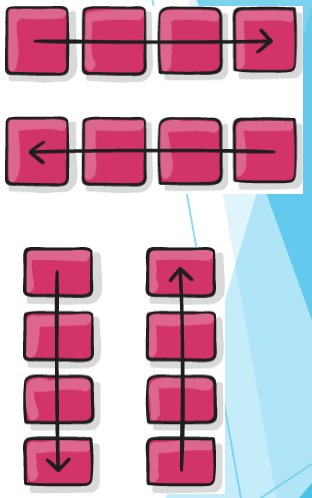
```css
.contenedor{
    display:flex;
}
.elemento{
    width:25%;
}
```

Al no haber definido aún el comportamiento de dirección y tamaño que tendrán los elementos de nuestro contenedor, aunque hayamos definido un ancho de 25%, éstos se adaptan a su padre ocupando el 100% de ancho entre la suma de todos.

Dirección de los ejes

Propiedad flex-direction

Permite modificar la dirección del eje principal del contenedor para que se oriente en horizontal (por defecto) o en vertical. Además, también podemos incluir el sufijo -reverse para indicar que coloque los ítems en orden inverso.

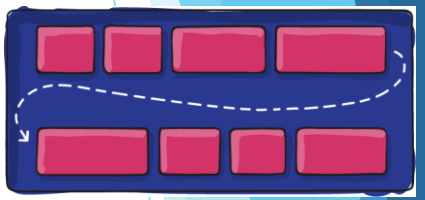


- **flex-direction: row;** Los elementos se visualizan de izquierda a derecha (valor por defecto)
- **flex-direction: row-reverse;** Los elementos se visualizan de derecha a izquierda.
- **flex-direction: column;** Los elementos se visualizan de arriba hacia abajo.
- **flex-direction: column-reverse;** Los elementos se visualizan de abajo hacia arriba.

Dirección de los ejes

Propiedad flex-wrap

Permite especificar el comportamiento del contenedor respecto a evitar que se desborde (nowrap, valor por defecto) o permitir que lo haga, en cuyo caso, estaríamos hablando de un contenedor flexbox multilínea, es decir los elementos se distribuyen en varias filas.



- **flex-wrap: nowrap;** Establece los elementos en una sola línea (no permite que se desborde el contenedor).
- **flex-wrap: wrap;** Los elementos se muestran en una sola línea, pero si su ancho supera la del contenedor, se distribuyen en varias filas.
- **flex-wrap: wrap-reverse;** Su comportamiento es igual al flex-wrap: wrap pero en dirección inversa.

Dirección de los ejes

Atajo / short hand

Existe una propiedad llamada **flex-flow** que permite resumir los valores de las propiedades flex-direction y flex-wrap en una sola propiedad:

```
.container {
  /* flex-flow: <flex-direction> <flex-wrap>; */
  flex-flow: row wrap;
}
```

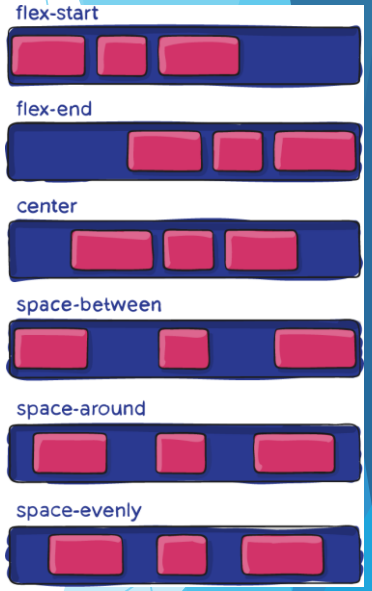
Propiedades de alineación

Sobre el eje principal

Existe distintas propiedades dentro de flexbox para disponer los ítems dependiendo de nuestro objetivo.

justify-content: Se utiliza para alinear los ítems del eje principal (por defecto, el horizontal). Y puede tomar los siguiente valores:

- **flex-start:** Agrupa los ítems al principio del eje principal.
- **flex-end:** Agrupa los ítems al final del eje principal.
- **center:** Agrupa los ítems al centro del eje principal.
- **space-between:** Distribuye los ítems dejando el máximo espacio para separarlos.
- **space-around:** Distribuye los ítems dejando el mismo espacio alrededor de ellos (izq/dcha).
- **space-evenly:** Distribuye los ítems dejando el mismo espacio (solapado) a izquierda y derecha.

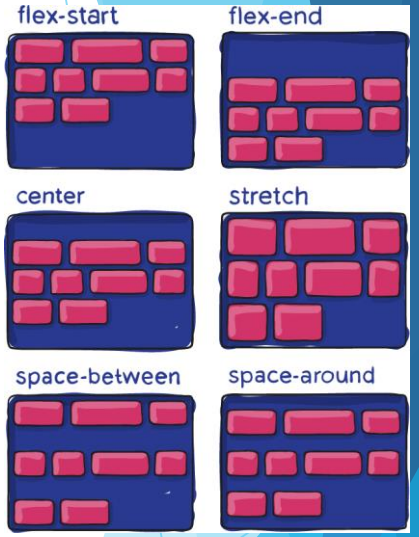


Propiedades de alineación

Sobre el eje principal

La propiedad **align-content** actúa sobre cada una de las líneas de un contenedor multilínea (no tiene efecto sobre contenedores de una sola línea). Los valores que puede tomar son los siguientes:

- **flex-start:** Agrupa los ítems al principio del eje principal.
- **flex-end:** Agrupa los ítems al final del eje principal.
- **center:** Agrupa los ítems al centro del eje principal.
- **stretch:** Alinea los ítems estirándolos de modo que cubran desde el inicio hasta el final del contenedor.
- **space-between:** Distribuye los ítems desde el inicio hasta el final.
- **space-around:** Distribuye los ítems dejando el mismo espacio a los lados de cada uno.

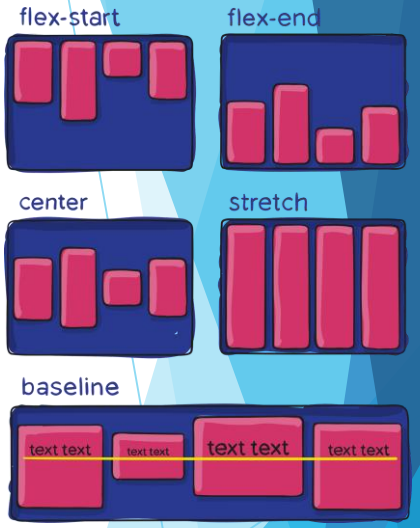


Propiedades de alineación

Sobre el eje secundario

La propiedad **align-items**, se encarga de alinear los ítems en el eje secundario del contenedor. Los valores que puede tomar son los siguientes:

- **flex-start:** Alinea los ítems al principio del eje secundario.
- **flex-end:** Alinea los ítems al final del eje secundario.
- **center:** Alinea los ítems al centro del eje secundario.
- **stretch:** Alinea los ítems estirándolos de modo que cubran desde el inicio hasta el final del contenedor.
- **baseline:** Alinea los ítems en el contenedor según la base del contenido de los ítems del contenedor.

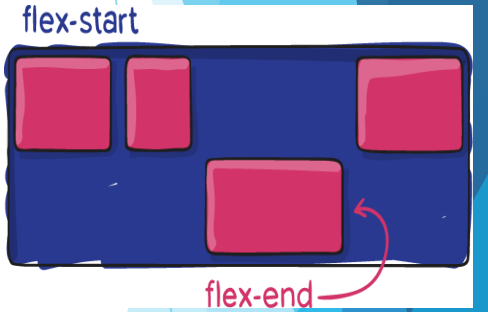


Propiedades de alineación

Sobre el eje secundario

La propiedad **align-self** actúa exactamente igual que align-items, sin embargo se utiliza sobre un ítem hijo específico y no sobre el elemento contenedor. Los valores que puede tomar son los mismos que align-items:

- **flex-start:** Alinea los ítems al principio del eje secundario.
- **flex-end:** Alinea los ítems al final del eje secundario.
- **center:** Alinea los ítems al centro del eje secundario.
- **stretch:** Alinea los ítems estirándolos de modo que cubran desde el inicio hasta el final del contenedor.
- **baseline:** Alinea los ítems en el contenedor según la base del contenido de los ítems del contenedor.
- **auto:** Hereda el valor de align-items del padre (si no se ha definido, es stretch).



Propiedades de hijos

Todas las propiedades vistas se aplican sobre el elemento contenedor. Las siguientes propiedades, se aplican sobre los ítems hijos.

- **flex-grow:** 0 Número que indica el factor de crecimiento del ítem respecto al resto.
- **flex-shrink:** 1 Número que indica el factor de decrecimiento del ítem respecto al resto.
- **flex-basis:** size - content tamaño por defecto que tendrán los ítems antes de aplicarle la distribución de espacio. Generalmente, se aplica un tamaño (unidades, porcentajes, etc...), pero también se puede aplicar la palabra clave content que ajusta automáticamente el tamaño al contenido del ítem, que es su valor por defecto.
- **order:** 0 Número que indica el orden de aparición de los ítems.

Resumen

HTML siempre representará contenido y CSS siempre representará la apariencia de ese contenido.

Referencias

- https://medium.com/@alexcamachogz/una-gu%C3%ADa-completa-de-flexbox-768b038de5e9
- https://developer.mozilla.org/es/docs/Web/CSS/CSS_Flexible_Box_Layout/Conceptos_Basicos_de_Flexbox
- https://flexboxfroggy.com/#es
- https://developer.mozilla.org/es/docs/Web/CSS/position#Examples

Gracias!