Selectores

sábado, 1 de julio de 2023 18:57

Universal

```
* {
    color:black;
}
```

De tipo

```
h1 {
    color: grey;
}
```

Clases

```
.h2-class {
    color: blue;
}
```

ID - Identificador

```
#element {
    color: red;
}
```

Solo usar 1 ID para 1 elemento.

Por atributo

```
[atributo="palabra"] {
    color:skyblue;
}
```

Descendiente

```
h2 p {
    color:violet;
}
```

Pseudo-clases

```
button:hover {
    color:yellow;
}
```

Múltiple

```
h1, h2, h3 {
    color: grey;
}
```

Especificidad

domingo, 2 de julio de 2023 07:31

Metodología BEM

domingo, 2 de julio de 2023 07:31

Ejemplo:

PROPIEDAD ESPECIAL

```
<div class="form">
     <input type="text" class="form__input--active">
     <input type="text" class="form__input">
     <input type="text" class="form__input">
     <input type="text" class="form__input">
     <input type="text" class="form__input">
</div>
```

Elemento diferente

Elementos iguales

POR ORDEN

```
<div class="text">
     <p class="text__p">
          <h2 class="text__p-h2">
               <h3 class="text__p-h2-h3">
               </h3>
          </h2>
     <p>
</div>
```

Unidades

domingo, 2 de julio de 2023 07:31

Fijas:

- Px
- Pt
- Cm
- Mm
- Porcentaje

Relativas - Utilizadas en Responsive Design

- Em (por defecto: 16px, pero hereda la medida de su caja madre)
- Rem
- Vh = Viewport Height (% de la altura de la pantalla visible)
- Vw = Viewport Width (% de la anchura de la pantalla visible)

Texto

domingo, 2 de julio de 2023 07:31

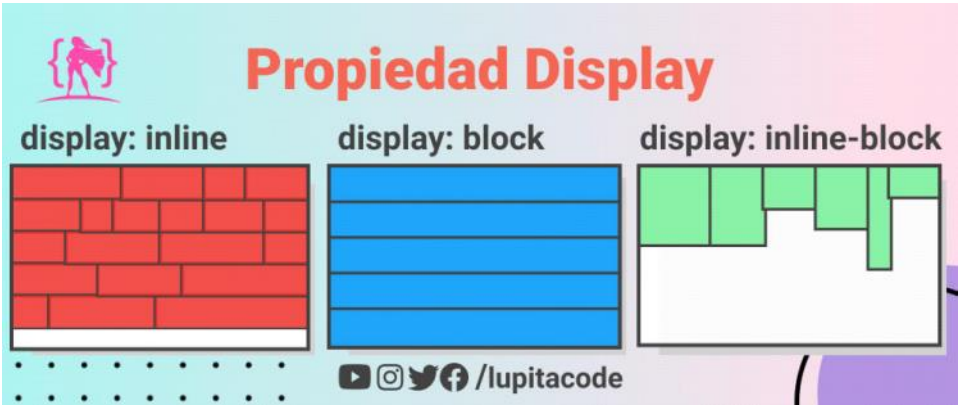
Propiedades del texto:

Font-size	2em, 16px, etc
Font-family	'Poppins', Sans-serif (si la primera no está disponible se usa la siguiente y así sucesivamente)
Line-height	1, 1.5, 2
Font-weight	Normal, bold, ultra-bold - 400, 500, 600, 700
Font-stretch	Normal, condensed
Font-variant	Small caps, etc
Color	red, blue, #000000

Cajas

domingo, 2 de julio de 2023 07:58

Display	Por defecto	
Inline	Ocupa el ancho y alto del contenido	Tamaño fijo (no se puede modificar height o width)
Block	Ocupan el ancho de la página y el alto del contenido	Tamaño moldeable



Propiedades de cajas

domingo, 2 de julio de 2023 08:12

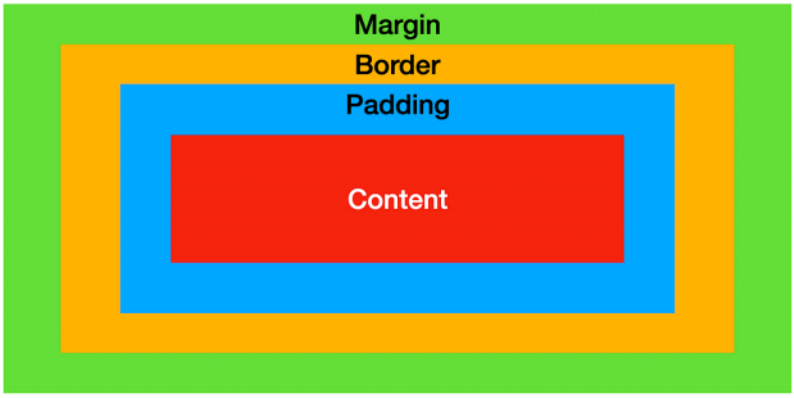
- **Padding:** Distancia entre el texto y la caja
- **Margin:** Distancia entre la caja y otros objetos.

Background-color	Red, #000000	
display	Inline-block - block - inline	
padding	10px, 20px, 30px, 40px (top, right, bottom, left) 10px, 20px (eje y, eje x)	Shorthand property (acortable)
margin	10px, 20px, 30px, 40px (top, right, bottom, left) 10px, 20px (eje y, eje x)	Shorthand property (acortable)
height	32px	
width	48px	
Box-sizing	Border-box - content-box	• **Border-box:** El tamaño total de la caja incluiria el tamaño del borde. • **Content-box**: Lo contrario, el borde se le añade al tamaño total, entre otras cosas.
Text-align	center	
border	4px solid blue (tamaño, estilo, color)	Shorthand property
Border-radius	10% - 50% etc	Esquinas redondeadas
Border-style	Solid - dashed - double - inset - outset - groove, ridge - etc	
Box-shadow	1px 2px 15px 0 black (eje x, eje y, desenfoque, borde, color)	
text-shadow	1px 2px 15px black (eje x, eje y, desenfoque, color)	
transform	rotate(90deg)	
opacity	0.5 = 50%	
transition	all 1s	

Box-model

domingo, 2 de julio de 2023 08:31

Modelo en el que se trabajan las cajas



Propiedades - Rango de jerarquía:

- Content -> line-height
- Padding -> padding
- Border -> border
- Margin -> margin

Outline

domingo, 2 de julio de 2023 08:42

No ocupa un espacio real en el DOM (Document object model), a diferencia del border

---

- El outline no hace que otros elementos se muevan, pero el border sí.
- El outline no forma parte de la caja
- Su función es enmarcar elementos

`Margin: auto`

Fórmula:
AP = Ancho total de la pantalla
AC = Ancho de la caja

Margin auto = (AP - AC) / 2

Las cajas no deben tener ningún espacio en HTML para que no muestren espacios entre ellas en pantalla

Positions

domingo, 2 de julio de 2023 09:12

Relative

El objeto adquiere 4 propiedades nuevas:
Top, right, bottom y left.

Cada una sirve para que el elemento se mueva determinada distancia hacia cierta dirección.
A partir del elemento anterior.

Las propiedades TOP y LEFT son las que tienen más importancia. Si están presentes, right y bottom se ignora.

Por ejemplo:

```
Elemento {
        Top: 20px
        Bottom: -30px
}
```

Es decir, le puedes especificar donde quieres que esté con más precisión

Absolute

Inicialmente cada caja tiene un espacio reservado. (para que sea visible)
Si ponemos una caja con position absolute, la caja que esté por delante de ella tomará su lugar y se sobrepondrá, aunque la caja siga estando allí.

Es decir, ese espacio propio de la caja, se elimina por completo.

Fixed

Igual que absolute, pero la caja queda fija con el scroll.

Sticky

Su función es definir a partir de qué punto la caja va a quedar fija.

Por ejemplo:

```
Caja {
        Position: sticky;
        Top: -50px;
}
```

(A partir de los 50px se estaría quedando fija en la pantalla)

Z Index

domingo, 2 de julio de 2023 12:00

Ordena los elementos uno por delante de otro.

- El elemento que tenga mayor valor, será el que esté posicionado más adelante y así sucesivamente.
- Por el flujo de html, inicialmente el objeto que es declarado al último es el que está posicionado más adelante, pero esto es modificable con Z-Index

Se recomienda colocar los z-index de 100 en 100 para dejar espacios en caso de que se quieran agregar más en otros elementos.

CONFLICTO PADRES E HIJOS:

Teniendo los dos (padre e hijo) la propiedad RELATIVE:
La única forma para que un contenedor hijo se pueda poner por detrás de un contenedor padre: (a pesar de tener un z-index inferior) es:

1. SACÁNDOLE la propiedad z-index al hijo (queda en undefined)
2. Dándole un z-index NEGATIVO al padre

Display

domingo, 2 de julio de 2023 15:18

block	Ocupan todo el ancho	Width y height modificable
inline	Ocupan el ancho del contenido	Width y height NO modificable
Inline-block	Ocupan el ancho del contenido	Width y height modificable
Table		EN DESUSO
Inline-table		EN DESUSO
List-item		EN DESUSO
Table-cell	Celda de una tabla	EN DESUSO
Table-row	Fila de una tabla	EN DESUSO
Table-column	Columna de una tabla	EN DESUSO
grid		
flex		
Inline-grid		
Inline-flex		
none	Elimina el objeto del DOM	

Overflow

domingo, 2 de julio de 2023 15:32

Su función es brindar distintas alternativas para gestionar la situación cuando un texto sobresale de una caja.

El overflow es lo que sobresale de una caja.

Además es una propiedad shorthand.

Propiedades:

- Inherit (por defecto) - Deja que el texto sobresalga
- Auto - Si es necesario, aparece la herramienta para scrollear. Ya sea eje X o eje Y.
- Scroll - Muestra forzadamente la herramienta scroll.
- Hidden - Oculta forzadamente la herramienta scroll.

Ejemplo:

```
.caja1 {
    overflow: auto
}

.caja2 {
    overflow-x: scroll
    overflow-y: hidden
}
```

Float

domingo, 2 de julio de 2023 15:58

Permite ubicar a un elemento a la izquierda o a la derecha

Ejemplo:

```
* {
      float: right
}
```

Su uso principal radica básicamente en hacer que el texto envuelva a una imagen.

La imagen debe ir arriba en el html.

```
.cont {
    margin: auto;
    margin-top: 50px;
    border: 4px solid red;
    background: grey;
    width: 50%;
    padding: 20px
}

.cont img {
    float: right;
    width: 300px;
    margin-right: 18px
}
```

Pseudoelementos

domingo, 2 de julio de 2023 16:54

- No forman parte del DOM (Document object model)
- No funcionan en cajas inline

Por ejemplo:

```
.texto::first-line {
    Color: blue;
}
```

(La primera línea se pinta de determinado color, independientemente del tamaño de la ventana)

Pseudoelementos más utilizados:

`::first-line`	No funciona en inline
`::first-letter`	No funciona en inline
`::placeholder`	El placeholder es el texto que aparece en un input (text, textarea).
`::selection`	El recuadro que aparece cuando seleccionas un texto
`::before`	
`::after`	

Before:

Se introduce un texto no seleccionable ANTES del texto original

```
.texto::before {
    content: "Hola ";
    color:  blue;
}

<h2 class="texto">
    cómo estás
 </h2>
```

En la página: Hola cómo estás

After:

Se introduce un texto no seleccionable DESPUÉS del texto original

```
.texto::after {
    content: "cómo estás";
    color:  blue;
}

<h2 class="texto">
    Hola
 </h2>
```

En la página: Hola cómo estás

Pseudoclases

domingo, 2 de julio de 2023 17:24

- Se aplica a un elemento.
- Escucha un evento.

Pseudoclases más utilizadas:

`:hover`	Cuando pasas el mouse por encima del elemento	Funciona en cualquier elemento
`:link`	Cuando un link NO ha sido visitado (etiqueta a)	
`:visited`	Cuando un link SI ha sido visitado (etiqueta a)	En espeficidad está entre las clases y los ids
`:active`	Cuando se mantiene el clic en un elemento	
`:focus`	Cuando se le da clic a un INPUT (mantienes seleccionado el elemento), cuando le das otro clic se revierten los cambios	
`:lang`	Muestra algo según el idioma determinado por el navegador	
`:first-child`	Selecciona el primer elemento de un grupo de elementos iguales.	
`:nth-child(n)`	Selecciona un elemento en específico de un grupo de elementos iguales.	`Ej: containers:nth-child(2) para seleccionar al segundo.`

Lang

```
:lang(en) {
    background: red;
}

:lang(es) {
    background: yellow;
}
```

HTML:

```
<p lang="en">
    Hello, how are you?
</p>

<p lang="es">
    Hola, ¿qué tal?
</p>
```

Object-fit

domingo, 2 de julio de 2023 18:52

Se aplica para modificar imágenes.

- Inicialmente, si le proporcionamos solo el width a una imagen cuadrada, esta se agrandará proporcionalmente, porque la caja se agranda.

Ejemplo:

```
* {
    object-fit: contain;
}
```

Propiedades:

`contain`	La imagen se ajusta a su resolución real pero deja un espacio en el contenedor si es necesario.
`cover`	La imagen se ajusta al contenedor sin perder la resolución.
none	La imagen va a mantener su tamaño exacto sin importar el tamaño del contenedor. No se va a agrandar ni achicar.
`scale-down`	Se queda con la mejor propiedad dependiendo de la situación

Object-position

Tira la imagen para determinado lado o distancia. (top, right, bottom, left)

Ejemplos:

```
* {
    object-position: left;
}
```

```
* {
    object-position: 2em;
}
```

Cursor

domingo, 2 de julio de 2023 19:09

Existen distintos tipos de cursor para indicar que se está realizando cierta acción en específico.

Cuando se le da clic o se pone el mouse encima.

Ejemplo:

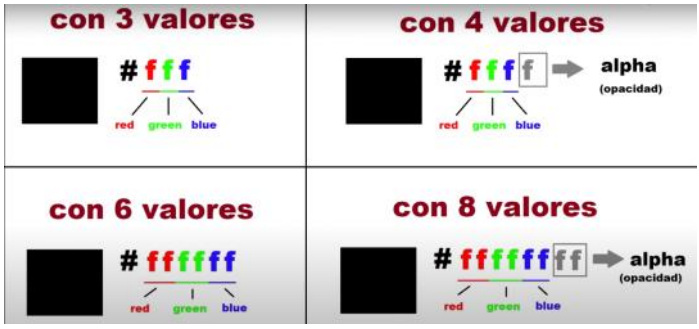
```
* {
    cursor: pointer;
}
```

```
.alias {cursor: alias;}
.all-scroll {cursor: all-scroll;}
.auto {cursor: auto;}
.cell {cursor: cell;}
.col-resize {cursor: col-resize;}
.context-menu {cursor: context-menu;}
.copy {cursor: copy;}
.crosshair {cursor: crosshair;}
.default {cursor: default;}
.e-resize {cursor: e-resize;}
.ew-resize {cursor: ew-resize;}
.grab {cursor: grab;}
.grabbing {cursor: grabbing;}
.help {cursor: help;}
.move {cursor: move;}
.n-resize {cursor: n-resize;}
.ne-resize {cursor: ne-resize;}
.nesw-resize {cursor: nesw-resize;}
.ns-resize {cursor: ns-resize;}
.nw-resize {cursor: nw-resize;}
.nwse-resize {cursor: nwse-resize;}
.no-drop {cursor: no-drop;}
.none {cursor: none;}
.not-allowed {cursor: not-allowed;}
.pointer {cursor: pointer;}
.progress {cursor: progress;}
.row-resize {cursor: row-resize;}
.s-resize {cursor: s-resize;}
.se-resize {cursor: se-resize;}
.sw-resize {cursor: sw-resize;}
.text {cursor: text;}
.url {cursor: url(myBall.cur),auto;}
.w-resize {cursor: w-resize;}
.wait {cursor: wait;}
.zoom-in {cursor: zoom-in;}
.zoom-out {cursor: zoom-out;}
```

Colorización

domingo, 2 de julio de 2023 19:15

- Usar colores en formatos específicos tiene una ventaja por sobre utilizar los colores predefinidos con nombres.
- Ya que, los distintos navegadores pueden interpretar dichos colores con una tonalidad diferente a la que queremos.
- Existen varios sistemas de colorización: Hexadecimal, RGB, RGBA, HSL, HSLA, etc.



RGB

RGB = Red Green Blue

Se declaran 3 valores del 0 al 255 (intensidad del color).

RGBA

Igual que RGB, pero se le añade el valor de la opacidad (0 al 1)

Ejemplo: 255, 0, 255, 0.5

A = alpha

HEX

Sistema de 16 unidades

Del 0 al 10: 0-10
Del 11 al 16: A-F

Ejemplo:

#FF00FF

https://htmlcolorcodes.com

Linear-gradient

Parámetros:

- Grados, color1 (%), color2 (%)

Keywords grados:

- `to top`
- `to bottom`
- `to left`
- `to right`

Ejemplos:

```
* {
    background: linear-gradient(to left, red, blue)
}
```

```
* {
    background: linear-gradient(45deg, #fff 50%, #000 75%)
}
```

Radial-gradient

Parámetros:

- Tipo dirección, color1 (%), color2 (%)

```
* {
    background: radial-gradient(circle to left, red, blue)
}
```

Responsive Design

domingo, 2 de julio de 2023 20:27

La función del responsive design es adaptar el diseño de la página web para todas las resoluciones.

- Mobile first = Empezar a crear la web especialmente para resoluciones pequeñas (celulares, tablets, etc), y posteriormente adaptarla para escritorio. Google recomienda las páginas con este concepto.

- Se trabaja con media-query: Cambiar el estilo de los elementos a partir de cierta resolución

Añadir etiqueta al head del HTML: (OBLIGATORIO)

```
<meta name="viewport"content="width=device-width, initial-
scale=1.0"/>
```

Ejemplo responsive design:

```
.div1, .div2 {
     Width: 45%;
     Display: inline-block;
}

@media only screen and (max-width: 800px) {

     .div1, .div2 {
          display: block;
          width: 100%;
     }

}
```

Es decir, cuando el ancho de la resolución sea menor a 800px, los divs se colocarán uno debajo de otro.

Menú Responsive

lunes, 3 de julio de 2023 17:57

- Menú que se despliega a partir de determinada resolución.
- Se oculta el nav normal y se muestra otro adaptado para celulares o tablets.

Paso 1

Se crearon 2 navs, uno para resolución de escritorio y el otro responsive.

En el responsive se agregan 2 divs:

- Botón
- Contenedor del menú

```
<header>
    <nav class="nav">
        <ul class="nav__ul">
            <li class="nav__li"><a href="#"> Inicio </a></li>
            <li class="nav__li"><a href="#"> Sobre nosotros </a></li>
            <li class="nav__li"><a href="#"> Gana dinero </a></li>
            <li class="nav__li"><a href="#"> Ayuda </a></li>
        </ul>/.nav__ul

        <ul class="nav__responsive-ul">
            <div class="nav__responsive-button"></div>
            <div class="nav__li-container">
                <li class="nav__responsive-li"><a href="#"> Inicio </a></li>
                <li class="nav__responsive-li"><a href="#"> Sobre nosotros </a></li>
                <li class="nav__responsive-li"><a href="#"> Gana dinero </a></li>
                <li class="nav__responsive-li"><a href="#"> Ayuda </a></li>
            </div>/.nav__li-container

        </ul>/.nav__responsive-ul
    </nav>/.nav
</header>
```

Paso 2

Pegar en el head del HTML el siguiente código:

```
<script src="https://kit.fontawesome.com/62ea397d3a.js" crossorigin="anonymous"></script>
```

Con este script se tendrá acceso a íconos transparentes y modificables en color.

Paso 3

Añadir los íconos mediante las clases, por ejemplo:

```
<li class="nav__li"><i class="fas fa-home"></i><a href="#"> Inicio </a></li>
<li class="nav__li"><i class="fas fa-user-friends"></i><a href="#"> Sobre nosotros </a></li>
<li class="nav__li"><i class="fas fa-dollar-sign"></i><a href="#"> Gana dinero </a></li>
<li class="nav__li"><i class="fas fa-question-circle"></i><a href="#"> Ayuda </a></li>
```

FAS = Font Awesome Solid

Paso 4

En CSS:

```
.nav {
    background-color: darkslateblue;
```

```
.nav__li:hover { /*Modifica solo el ícono*/
    color: #ccc
```

Paso 4

En CSS:

```css
.nav__responsive-ul {
    display: none;
}
```

Se oculta el nav responsive

```css
.nav {
    background-color: darkslateblue;
}

.nav__li {
    display: inline-block;
    padding: 12px;
    color: #fff;
}

.nav__li a {
    text-decoration: none;
    color: #fff;
}

.nav__li i {
    width: 30px;
    box-sizing: border-box;
    text-align: center;
    margin-right: 5px
}
```

Se definen las propiedades del nav

```css
.nav__li:hover { /*Modifica solo el icono*/
    color: #ccc
}

.nav__li:hover > a { /*Modifica solo la letra*/
    color: #ccc
}
```

Se añaden hovers

Paso 5

Se declara el evento en el cual, si el ancho de la resolución es menor a 500px:

- Se oculta el nav de escritorio

- Aparece el nav responsive

```css
@media only screen and (max-width: 500px) {
    .nav__ul {
        display: none;
    }
    .nav__responsive-ul {
        display: block;
    }
}
```

Flexbox

lunes, 3 de julio de 2023 19:39

Flexbox = Cajas flexibles

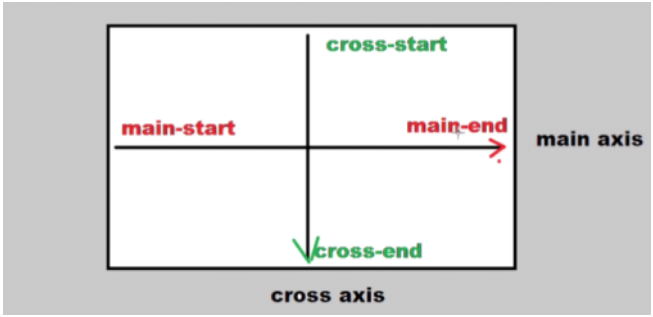
Su función es maquetar sitios web.

Nació como una alternativa simplificada de las tablas.

- Flex requiere de dos elementos: Flex container y Flex items
- De por sí, el contenedor se comporta como un block.
- Los hijos de un flex container son flex items, pero no sus nietos
- Si hay un desbalance en la cantidad de texto en cada ítem: No cambian las alturas ni los widths, se ajusta el contenido

Ejemplo:

```
.flex-container {
    display: flex;
}
```



- Main axis = Eje X - De izquierda a derecha
- Cross axis = Eje Y - De arriba a abajo

(Tiene una dirección en la que apunta.)

Flex-direction

Nos permite cambiar la dirección del main-axis

La propiedad se le aplica al contenedor. El valor por defecto es "row".

Valores:

`row`	Ordena los flex-items en FILAS	De izquierda a derecha
`row-reverse`	Ordena los flex-items en FILAS (al revés)	De derecha a izquierda
`column`	Ordena los flex-items en COLUMNAS	De arriba a abajo
`column-reverse`	Ordena los flex-items en COLUMNAS (al revés)	De abajo a arriba

Flex-wrap

- Mantiene el width o height de las cajas en una resolución menor.
- Pone las cajas una por debajo de otra.

Valores:

- Nowrap (por defecto)
- Wrap
- Wrap-reverse (al reducir la resolución, la última caja se pone arriba y así sucesivamente)

Flex-flow

Shorthand de flex-direction y flex-wrap

Ejemplo:

```
.flex-container {
    flex-flow: column wrap;
}
```

Justify-content

Valores:

`center`	Las cajas se centran al medio de la pantalla
`space-around`	Márgenes iguales --> Lo mismo que margin: auto
`space-between`	Calcula un margen automático para todas las cajas. Quiere que todas las cajas tengan la mayor y la misma distancia entre sí.
`space-evenly`	Calcula que las distancias entre las cajas sean exactamente iguales

Align-items

Se aplica cuando solo hay 1 línea de flex-items

Valores:

`center`	Las cajas se centran verticalmente en la pantalla visible.

`flex-start`	Las cajas se alínean al comienzo de la pantalla visible.
`flex-end`	Las cajas se alínean al final de la pantalla visible.
`stretch`	Por defecto (si el vh o vw es 100, las cajas se estiran hasta el final de la pantalla visible)
`baseline`	Todas las cajas se van para abajo A medida que se reduce la resolución algunas cajas se van yendo gradualmente hacia arriba.

Align-content

Se aplica cuando se trabaja con varias líneas de flex-items.

- Tiene los mismos valores que align-items.

Propiedades Flex-items

martes, 4 de julio de 2023 20:32

Al igual que el flex container, los flex items también cuentan con distintas propiedades reservadas.

Entre ellas se encuentran:

Align-self

Determina la alineación en el cross-axis de determinada caja de manera independiente.

`center`	La caja se centra verticalmente en la pantalla visible.
`flex-start`	La caja se alínea al comienzo de la pantalla visible.
`flex-end`	La caja se alínea al final de la pantalla visible.
`stretch`	Por defecto (si el vh o vw es 100, la caja se estiran hasta el final de la pantalla visible)
`baseline`	La caja se va para abajo Si se reduce la resolución, en determinado punto la caja se iría para arriba de las demás.

Margin

El funcionamiento del margin en las cajas flexibles es particularmente diferente al de las cajas normales.

Por ejemplo, si le proporcionamos un margin-left: auto, la caja se va completamente hacia la derecha, y viceversa.

Flex-grow

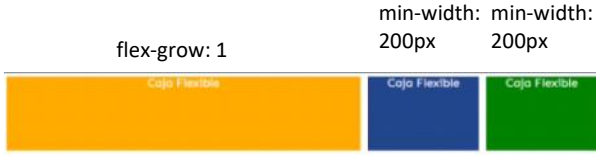
Reparte el espacio sobrante entre determinada caja o cajas. Es decir, los flex-items elegidos se van a estirar proporcionalmente hasta ocupar todo el ancho de la pantalla en partes iguales, y si se reduce la resolución, se quedan hasta el min-width.

Es recomendable aplicar un flex-wrap: wrap, para que las cajas se coloquen una debajo se otra cuando se llegue a determinada resolución. (min-widths sumados de cada caja + margins)

Se trabaja con proporciones

Ejemplo:

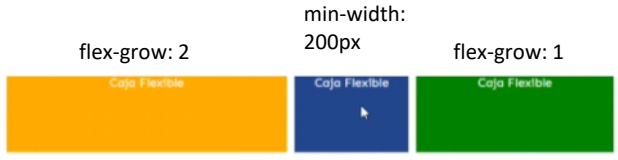
```
.flex-item {
    flex-grow: 1;
}
```



También se puede aplicar la propiedad a uno o varios flex-items individualmente.

En el caso de la imagen, el ancho de la página se repartiría en:

- Las 2 cajas con min-width
- La caja con ancho ilimitado (flex-grow: 1)



Flex-basis

Similar al width. En términos de especificidad, el flex-basis está por encima del width.

Al tener más relevancia en los rangos jerárquicos del entorno de flex, se recomienda usarlo por sobre width, ya que cumplen la misma función.

Flex-shrink

Determina qué cantidad de espacio va a ceder cierto flex-item cuando tenemos una resolución inferior a la deseada, en términos relativos (proporciones).

Por ejemplo:

- Hay 2 cajas con flex-shrink: 2, y hay 1 caja con flex-shrink: 1
- Pero queremos que todas las cajas midan 300px en algún punto

Entonces, lo que pasaría sería lo siguiente:

La caja con shrink 2 cedería el doble de espacio que las otras dos hasta que TODAS lleguen a 300px de ancho.

El valor por defecto es 1.

0 = No cede espacio.

Flex (shorthand)

Agrupa las tres propiedades anteriores: flex-grow, flex-shrink y flex-basis.

El primer parámetro es obligatorio, es decir, flex-grow.

Por ejemplo:

```
.flex-item {
    flex: 1, 2, 300px;
}
```

Order

Similar al z-index. Es decir, el flex-item que tenga mayor valor será el que esté posicionado más hacia la dirección en la que apunta el main axis. (Por defecto hacia la derecha, pero también podría ser hacia la izquierda, arriba o abajo, en caso de que esté estipulado así).

Al igual que en el z-index, se recomienda aumentar los valores por cada ítem de 50 en 50 o de 100 en 100 para dejar espacios en el aire, en el caso de que se quieran agregar más en otros elementos.

Grid

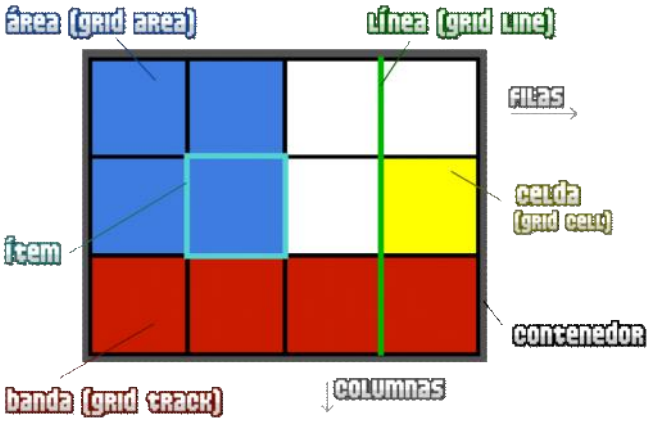
miércoles, 5 de julio de 2023 19:46

Grid = grilla

Al igual que Flexbox, es una propiedad de display. Es un estilo de layout en el cual se trabaja con grillas.

Una caja grid funciona a través de celdas, rows y columns (tracks), areas, etc.

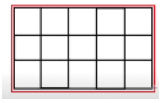
- Requiere de una serie de elementos: Grid-container y Grid-items
- El contenedor se comporta como block
- Los hijos directos del Grid-container son Grid-items, sus nietos no.



Ejemplo:

```
.grid-container {
    display: grid;
}
```

Grid container

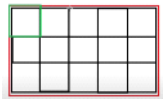


Es la totalidad del contenedor.

Grid item

Los hijos directos de un grid-container. No son necesariamente celdas (grid-cell).

Grid cell



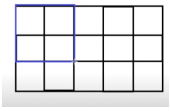
Cada una de las celdas, es decir, las divisiones que se forman dentro del container.

Grid track

Existen dos tipos: column y row.

Columns + Rows = Grid Tracks

Grid area



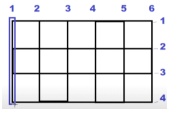
Determinadas áreas dentro del contenedor que seleccionamos para darles propiedades en específico.

No están definidas por defecto, es decir, se definen manualmente.

Las áreas deben estar obligatoriamente conformadas por un grupo de celdas consecutivas, tanto vertical como horizontalmente.

Por lo tanto, no se pueden agrupar celdas en diagonal ni de ninguna otra manera que no forme un cuadrilátero.

Grid line



Existen dos tipos: column-line y row-line

Column-lines = Columns + 1
Row-lines = Rows + 1

Propiedades de Grid

miércoles, 5 de julio de 2023 20:26

Grid-template-rows

Determina la cantidad de filas. Como parámetros se colocan las medidas para cada fila. (ya sean medidas exactas o relativas)

Medidas exactas (estáticas):

```
.grid-container {
    grid-template-rows: 150px 150px 150px;
}
```

Medidas relativas (proporcionales):

```
1fr = 1/total

.grid-container {
    grid-template-rows: 1fr 2fr 1fr;
}
```

La segunda columna tendría el doble de alto que las otras filas.

Grid-template-columns

Comparte las mismas características que Grid-template-rows.

Grid-gap

Propiedad shorthand. Determina la distancia entre una celda y la otras respectivamente.

(Fusión de las propiedades grid-row-gap y grid-column-gap)

1. Distancia entre la celda y las demás celdas.

```
.grid-container {
    grid-gap: 10px;
}
```

Grid-column-gap

Determina la distancia entre las columnas.

Grid-row-gap

Determina la distancia entre las filas.

Grid-column

Define la cantidad de celdas que va a ocupar un grid-item.

Ejemplo:

```
.grid-container:first-child {
    grid-column: 1 / 3;
}
```

(El grid-item va a ocupar el espacio desde la línea vertical 1 hasta la 3).

Esto genera por "inercia", que las demás celdas se empujen sucesivamente, y si fuese el caso se crearían nuevas filas para incluir a los elementos desplazados.

Ejemplo 2:

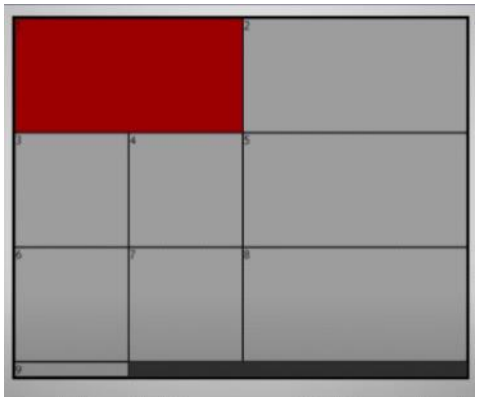
```
.grid-container:first-child {
    grid-column: 1 / span 2;
}
```

Span = indica la cantidad de columnas a ocupar.

(Va a ocupar 2 columnas desde la línea vertical 1)

Grid-row

Aplica lo mismo que para la propiedad grid-column.



Repeat()

Función que indica la cantidad de columnas o filas a colocar, y sus respectivas medidas.

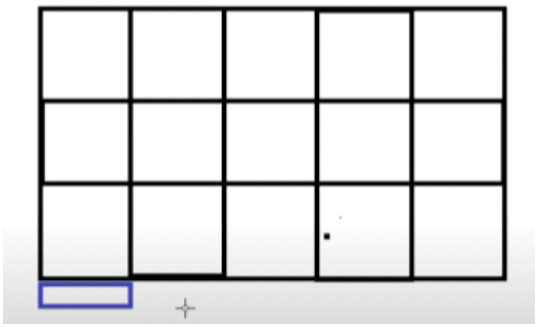
Ejemplo:

```
grid-template-columns: repeat(3, 120px)
```

3 columnas de 120 px

Grid implícito y explícito

miércoles, 5 de julio de 2023 21:50



Se entiende como Grid Implícito a aquellas celdas "sobrantes", que no han sido declaradas expresamente en el código.

Ya sea porque eventualmente ciertas celdas fueron desplazadas forzadamente generando esta situación, o porque simplemente no fueron declaradas.

Es decir, no forman parte del Grid Explícito.

Además, los elementos que formen parte del Grid Implícito cuentan con algunas propiedades particulares.

Propiedades:

Grid-auto-rows

Su función es gestionar y definir el tamaño de las celdas que eventualmente vayan a formar parte del Grid Implícito.

En este caso, le decimos al container que en caso de existir de celdas sobrantes, estas pasarán a formar nuevas FILAS y no columnas.

Ejemplo:

```
.grid-container {
    grid-auto-rows: 150px;
}
```

Grid-auto-columns

Mismo funcionamiento que Grid-auto-rows, pero en caso de existir celdas sobrantes, estas pasarán a formar nuevas COLUMNAS y no filas.

Grid-auto-flow

Gestiona los espacios vacíos en la grilla en alguna situación en particular que genere esto.

Valores:

`dense`	Ocupa los espacios libres de manera forzada, adapta determinadas celdas para que ocupen esos lugares.

Grid dinámico

viernes, 7 de julio de 2023 13:21

Son propiedades que le añaden ciertas funcionalidades con respecto a la distribución de las celdas en la grilla.

- Por defecto, el ancho mínimo de una celda radica en el ancho de la palabra más larga. (En cualquier unidad de medida, ya sea exacta o relativa)

Propiedades:

Min-content

El ancho de la celda se adapta a la cantidad mínima de contenido (sin cortar las palabras), es decir, hasta el final de la palabra más larga.

Ejemplo:

```
.grid-container {
    grid-template-rows: repeat(3, min-content);
}
```

Max-content

El ancho de la celda se adapta a la cantidad máxima de contenido. (Todo el texto de corrido en 1 línea)

Minmax()

Determina la medida mínima y máxima de ciertas filas o columnas.

Ejemplos:

```
.grid-container {
    grid-template-rows: repeat(3, minmax(100px, 1fr));
}

.grid-container {
    grid-template-rows: repeat(3, minmax(min-content, max-content));
}
```

Auto-fill

Calcula la cantidad máxima de columnas o filas posibles con determinadas propiedades.

Ejemplo:

```
.grid-container {
    grid-template-rows: repeat(auto-fill, minmax(100px, 1fr));
}
```

Es decir, la cantidad máxima de columnas posibles con mínimo 100px y 1fr máximo. (Puede dejar espacios vacíos)

Auto-fit

Ajusta las columnas para ocupar todo el espacio disponible, sin dejar espacios vacíos.

Alineación y Control de Flujo

viernes, 7 de julio de 2023 16:04

- Existen diversos tipos de alineación en Grid:

Alineación general	Todos los elementos
Alineación parcial	Columnas o filas
Alineación particular	Elemento individual

Propiedades:

_____ITEMS_____

Justify-items

Similar al justify-content de Flexbox.

Alínea los ÍTEMS horizontalmente

`center`	Centra las cajas y el texto
`start`	Alínea las cajas para la IZQUIERDA del contenedor y centra el texto
`end`	Alínea las cajas para la DERECHA del contenedor y centra el texto

Ejemplo:

```
.grid-container {
    justify-items: center;
}
```

Align-items

Alínea los ÍTEMS verticalmente

`center`	Alínea las cajas en el centro del contenedor.
`start`	Alínea las cajas para ARRIBA en el contenedor
`end`	Alínea las cajas para ABAJO en el contenedor

_____CONTENT_____

Justify-content

Alínea las COLUMNAS o FILAS horizontalmente

`center`	Centra las filas y columnas
`start`	Alínea las filas y columnas hacia la IZQUIERDA
`end`	Alínea las filas y columnas hacia la DERECHA
`space-around`	Márgenes iguales --> Lo mismo que margin: auto
`space-between`	Calcula un margen automático para todas las cajas. Quiere que todas las cajas tengan la mayor y la misma distancia entre sí.
`space-evenly`	Calcula que las distancias entre las cajas sean exactamente iguales

Align-content

Alínea las COLUMNAS o FILAS verticalmente

`center`	Centra las filas y columnas
`start`	Alínea las filas y columnas hacia ARRIBA
`end`	Alínea las filas y columnas hacia ABAJO
`space-around`	Márgenes iguales --> Lo mismo que margin: auto
`space-between`	Calcula un margen automático para todas las cajas. Quiere que todas las cajas tengan la mayor y la misma distancia entre sí.
`space-evenly`	Calcula que las distancias entre las cajas sean exactamente iguales

Aplicado al Grid-item

viernes, 7 de julio de 2023 16:50

Propiedades reservadas que se aplican individualmente para gestionar determinado Grid-item.

Align-self

Alínea individualmente un Grid-item en el EJE Y (verticalmente):

Propiedades:

`start`	Alínea las cajas lo máximo posible para ARRIBA (hasta que se tope con el margin de otra caja)
end	Alínea las cajas lo máximo posible para ABAJO (hasta que se tope con el margin de otra caja)

Ejemplo:

```
.grid-item:nth-child(8) {
    align-self: start;
}
```

Justify-self

Alínea individualmente un Grid-item en el EJE X (horizontalmente):

Propiedades:

`start`	Alínea las cajas lo máximo posible para la IZQUIERDA (hasta que se tope con el margin de otra caja)
end	Alínea las cajas lo máximo posible para la DERECHA (hasta que se tope con el margin de otra caja)

Place-self

Shorthand de Align-self y Justify-self

Parámetros:

1. Align-self
2. Justify-self

Ejemplo:

```
.grid-item:nth-child(8) {
    place-self: start end;
}
```

Stretch

(default)

Order

Reemplaza la posición de determinado Grid-item por otro en específico.

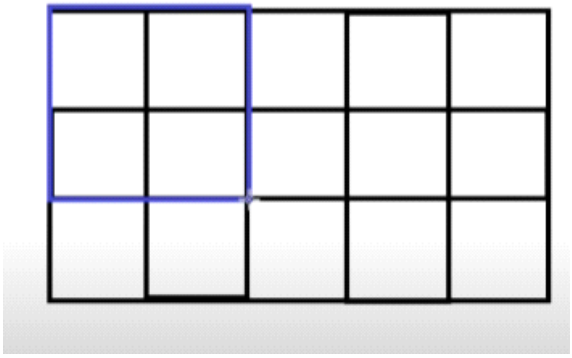
Ejemplo:

```
.grid-item:nth-child(2) {
    order: 3;
}
```

(El Grid-item 2 se desplazará hasta la posición del 3 y viceversa)

Grid Area

viernes, 7 de julio de 2023 19:45



Un Grid Area es un conjunto consecutivo de Grid-cells.

- Son determinadas áreas dentro del contenedor que seleccionamos para darles propiedades en específico.
- No están definidas por defecto, es decir, se definen manualmente.
- Las áreas deben estar obligatoriamente conformadas por un grupo de celdas consecutivas, tanto vertical como horizontalmente.
- Por lo tanto, no se pueden agrupar celdas en diagonal ni de ninguna otra manera que no forme un cuadrilátero.

Grid-template-areas

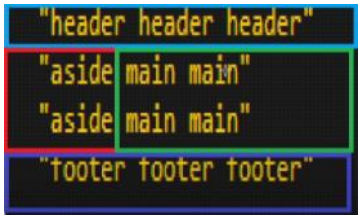
Se asigna la distribución de las áreas en relación a lo que se estipule en el texto.

Es decir:

En la misma línea, si se repite 1 vez un área "aside" y 2 veces un área "main":

El área "aside" tendrá 1/3 del ancho y el área "main" tendrá 2/3.

Ejemplo:




```
.grid-container {
    grid-template-areas:

    "header header header"
    "aside main main"
    "aside main main"
    "footer footer footer"
    ;
}
```

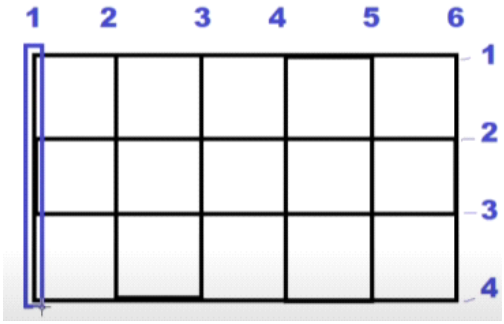
(Se crearon 4 Grid-areas: header, aside, main y footer)

Asignar áreas a los Grid-items

```
.grid-header {
    grid-area: header;
}

.grid-main {
    grid-area: main;
}

.grid-aside {
    grid-area: aside;
}

.grid-footer {
    grid-area: footer;
}
```

Grid Lines

viernes, 7 de julio de 2023 20:22



Las Grid Lines son aquellas líneas divisoras que se ubican entre las columnas y filas.

Existen dos tipos: column-line y row-line

- Column-lines = Columns + 1
- Row-lines = Rows + 1

Nombrar Grid Lines

Ejemplo:

```
.grid-container {
     grid-template-rows:
          [f-line]
          150px
          [s-line]
          150px
          [t-line]
          150px
          [h-line]
     ;
     grid-template-columns: repeat(3, 150px);
}

.grid-item:nth-child(2) {
     grid-row: s-line / h-line;
}
```

(El segundo Grid-item va a ocupar los espacios desde la segunda hasta la cuarta línea divisora)

Grid Shorthands

viernes, 7 de julio de 2023 21:26

Grid-template

Shorthand de Grid-template-columns y Grid-template-rows.

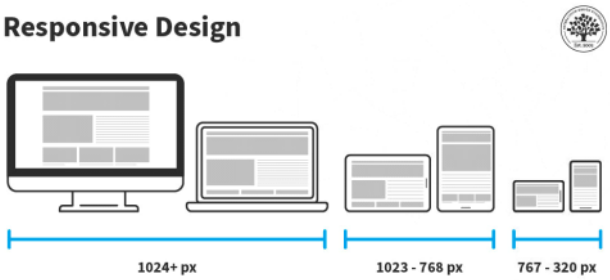
Ejemplo:

```
.grid-container {
    grid-template: repeat(3, 150px) / repeat(4, 200px);
}
```

(3 columnas de 150px y 4 filas de 200px)

Responsive Design

viernes, 7 de julio de 2023 21:36



El Responsive Design es una metodología / técnica utilizada para adaptar todo el contenido de una página web a distintas resoluciones, con el objetivo de optimizar la experiencia de usuario (UX).

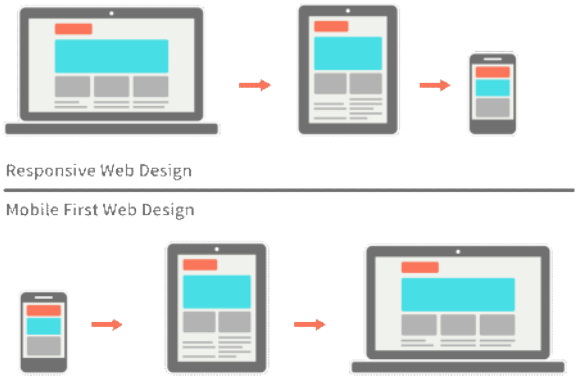
- Se trabaja con:
    - Estructuras flexibles (contenedores, imágenes y videos flexibles)
    - Media-queries
- @Media = Condicional que consulta y valida las características del dispositivo que accede a la página:
    - Ancho y alto de la ventana gráfica
    - Ancho y alto del dispositivo
    - Orientación (tablets, en modo horizontal o vertical)
    - Resolución (FHD, QHD, etc.)
- Tipos de Media Query:

`all`	Apto para todos los dispositivos
`print`	Destinado a material impreso / Modo vista previa de impresión
`screen`	Destinado a las pantallas
`speech`	Destinado a sintetizadores de voz

- Operadores: AND, OR
- Orientation: landscape (horizontal), portrait (vertical)

Metodologías

sábado, 8 de julio de 2023 09:48



Mobile First (recomendado)

Empezar a crear la web especialmente para resoluciones pequeñas (celulares, tablets, etc), y posteriormente adaptarla para escritorio o resoluciones mayores.

- Google recomienda las páginas con este concepto.
- Otorga ventajas en el ámbito del SEO (posicionamiento en navegadores)
- Es una buena práctica ya que facilita distribuir los elementos.

Desktop First

Empezar la web orientada a resoluciones grandes, por ejemplo monitores normales (16:9) o monitores ultra-wide (21:9), y posteriormente adaptarla a resoluciones menores.

Content First

Estructurar la web en base al contenido, independientemente de si va a estar orientada para resoluciones grandes o pequeñas.

Sintaxis Media Queries

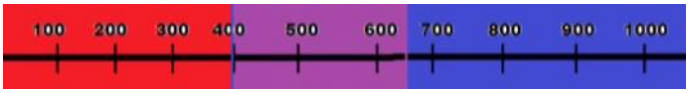
sábado, 8 de julio de 2023 09:57

1. Regla
2. Tipo
3. Condicional
4. (Parámetro : valor)

Ejemplos:

```
@media screen and (max-width: 400px) {
    display: none;
}
```

```
@media screen and (min-width: 400px) and (max-width: 650px) {
    display: block;
    background-color: violet;
}
```



Práctica Flex

sábado, 8 de julio de 2023 20:14

```
* {
    font-family: sans-serif;
}
.content {
    display: flex;
    flex-direction: column;
    height: 100vh;
}
.header, .main, .aside, .footer {
    padding: 20px;
}
.header {
    background-color: lightblue;
    flex-basis: 60px;
}
.main {
    background-color: slateblue;
    flex-basis: 300px;
    flex-grow: 2;
    flex-shrink: 0;
}
.aside {
    background-color: darkslateblue;
    flex-basis: 200px;
    flex-grow: 1;
    flex-shrink: 0;
}
.footer {
    background-color: violet;
    flex-grow: 1;
}
```

```
<body>
    <div class="content">
        <header class="header">Header</header>
        <article class="main">Main</article>
        <aside class="aside">Aside</aside>
        <footer class="footer">Footer</footer>
    </div>
</body>
```

```
@media screen and (min-width: 700px) {
    .content {
        flex-direction: row;
        flex-wrap: wrap;
    }
    .main, .aside {
        height: 100%;
    }
    .main {
        flex-grow: 4;
    }
    .aside {
        flex-grow: 1;
    }
    .header {
        flex-basis: 100%;
        max-height: 60px;
    }
    .footer {
        flex-basis: 100%;
    }
}
```

Práctica Grid

sábado, 8 de julio de 2023 20:15

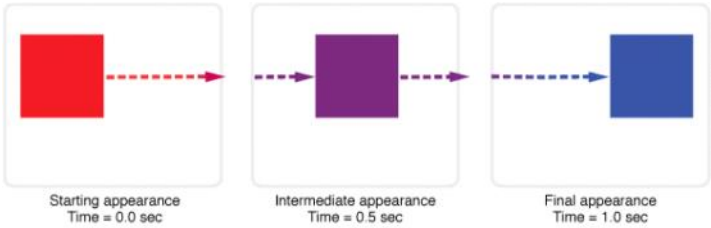
```
* {
    font-family: sans-serif;
}
.content {
    display: flex;
    flex-direction: column;
    height: 100vh;
}
.header, .main, .aside, .footer {
    padding: 20px;
}
.header {
    background-color: lightblue;
    flex-basis: 60px;
}
.main {
    background-color: slateblue;
    flex-basis: 300px;
    flex-grow: 2;
    flex-shrink: 0;
}
.aside {
    background-color: darkslateblue;
    flex-basis: 200px;
    flex-grow: 1;
    flex-shrink: 0;
}
.footer {
    background-color: violet;
    flex-grow: 1;
}
```

```
<body>
    <div class="content">
        <header class="header">Header</header>
        <article class="main">Main</article>
        <aside class="aside">Aside</aside>
        <footer class="footer">Footer</footer>
    </div>
</body>
```

```
@media screen and (min-width: 700px) {
    .content {
        display: grid;
        grid-template-rows: 60px 1fr 1fr 1fr 1fr 80px;
        grid-template-columns: repeat(3, 1fr);
    }
    .main {
        grid-row: 2 / 6;
        grid-column: 1 / span 2;
    }
    .aside {
        grid-row: 2 / 6;
    }
    .header {
        grid-column: 1 / span 3;
    }
    .footer {
        grid-row: 6 / 7;
        grid-column: 1 / span 3;
    }
}
```

Transition

viernes, 7 de julio de 2023 21:25



Propiedad que permite realizar cambios graduales en los estilos de los elementos, en determinado lapso de tiempo y con ciertas características.

- Para que una transición se ejecute es necesario disparar un evento (hover, onclick, focus, etc.)

Transition-property

Determina la propiedad en la cual se aplicará la transición.

Sintaxis:

- transition-property: propiedad (background, color, opacity, height, width, etc.)

Ejemplo:

```
.caja {
    background: red;
    transition-property: background;
}

.caja:hover {
    background: slateblue;
}
```

Transition-duration

Define la duración de la transición.

Ejemplo:

```
.caja {
    background: red;
    transition-duration: 1.0s;
}
```

Transition-delay

Determina la cantidad de tiempo que va a transcurrir antes de que se ejecute la transición.

```
.caja {
    background: red;
    transition-delay: 2.5s;
}
```

Transition-timing-function

Curva del tiempo en la que va a ejecutarse la transición. (Cambios en la aceleración y desaceleración)

Cabe resaltar que independientemente de la propiedad que elijamos, todas se terminarán de ejecutar en el mismo tiempo exacto.

Inherit = Heredar el color o propiedad de la caja que lo contiene.

Ease = Retraso

Valores:

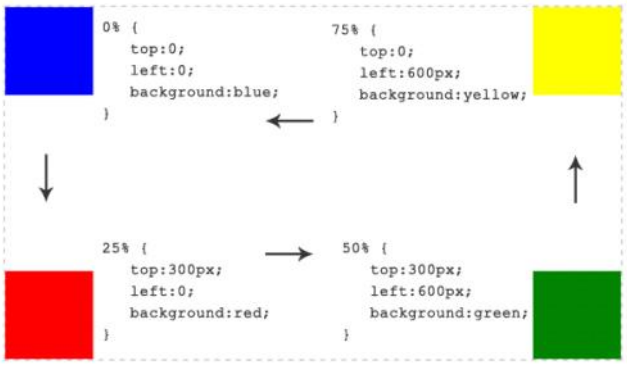
`linear`	Velocidad constante
`ease`	(Por defecto) Leve retraso al final
`ease-in`	Retraso al inicio
`ease-out`	Retraso al final
`ease-in-out`	Retraso al inicio y al final
`steps`	int, start | end
`initial`	-
`inherit`	Hereda la propiedad de la caja contenedora

Asignar valores simultáneamente

```css
.caja {
    transition-property: left, background, color;

    transition-duration: 1s, 2.5s, 5s;

    transition-delay: 0.3s, 0.5s, 0.7s;

    transition-timing-function: linear, ease-in, ease-out;
}
```

Animaciones

sábado, 8 de julio de 2023 21:18



Las animaciones permiten realizar efectos visuales en base a una secuencia de estilos.

Keyframe = Punto clave

- Requiere de la propiedad @keyframes
- Para declarar los keyframes de la secuencia, se trabaja con from / to (2 keyframes) o con porcentajes (múltiples keyframes).

Ejemplos:

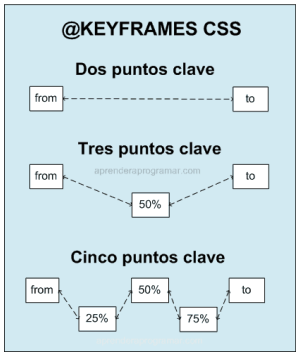
From / to

```
.caja {
     animation-name: desplazarse;
     animation-duration: 1s;
}

@keyframes desplazarse {
     from {
          position: relative;
          left: 0;
     }
     to {

          left: 80%;
     }
}
```

Porcentajes

```
.caja {
     animation-name: desplazarse;
     animation-duration: 1s;
}

@keyframes desplazarse {
     0% {
          position: relative;
          left: 0;
     }
     50% {
          left: 30%;
     }

     100% {
          left: 80%;
     }
}
```

```
@keyframes nombre {
     from {
          propiedad: valor;
     }
     to {
          propiedad: valor-nuevo;
     }

}

.elemento {
     animation-name: nombre;
     animation-duration: 1s;
}
```



Animation-timing-function

Curva del tiempo en la que va a ejecutarse la animación. (Cambios en la aceleración y desaceleración)

linear	Velocidad constante
ease	(Por defecto) Leve retraso al final
ease-in	Retraso al inicio
ease-out	Retraso al final
ease-in-out	Retraso al inicio y al final
steps	int, start | end
initial	-
inherit	Hereda la propiedad de la caja contenedora

Animation-iteration-count

Determina la cantidad de veces que se va a repetir la animación.

Ejemplos:

1)

```
.caja {
     animation-iteration-count: 3;
}
```

2)

```
.caja {
     animation-iteration-count: infinite;
}
```

Animation-direction

Cambia el orden de la secuencia de la animación.

Valores:

- Normal (por defecto)
- Reverse
- Alternate
- Alternate-reverse

Ejemplos:

```
.caja {
     animation-direction: normal;
}
```

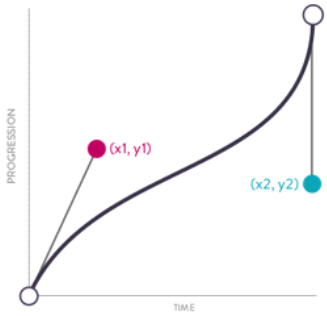
Animation-fill-mode

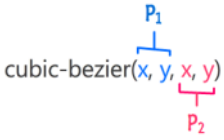
Define cuál va a ser el modo final de la animación.

none	-
backwards	Se queda con los valores del PRIMER keyframe
forward (recomendado)	Se queda con los valores del ÚLTIMO keyframe
both	Si existe un delay, antes de iniciar la animación adopta los valores estáticos del primer keyframe (background)

Cubic Bezier

sábado, 8 de julio de 2023 21:49





Función que se aplica en la propiedad animation-timing-function.

Permite modificar con máxima precisión la curva de tiempo de determinada animación. (aceleración y desaceleración)

- Se le otorgan 4 parámetros. (coordenadas en términos relativos)
- Recurso: https://cubic-bezier.com

Ejemplo:

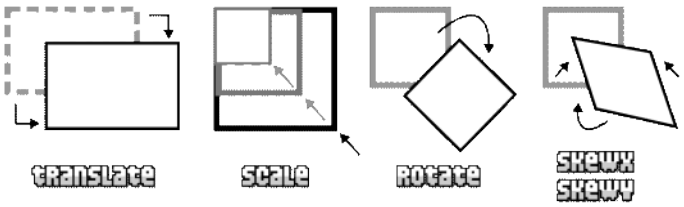
```
.caja {
    animation-timing-function: cubic-bezier(0.4, 0.2, 0.3, 1.2);
}
```

Transform

sábado, 8 de julio de 2023 22:06

Ejemplo:

```
.caja {
    transform:
        translate(50px, -70%);
        rotate(30deg);
}
```



TRANSLATE

Translate()

Desplaza a los elementos hacia cierta dirección.

Shorthand de TranslateX y TranslateY.

- Se recomienda utilizarlo por sobre las funciones top, left, right, bottom, etc, de position: relative, ya que posee una gran ventaja en cuanto a rendimiento y optimización.

Ejemplo:

```
.caja {
    transform: translate(50px, -70%);
}
```

TranslateX()

Desplaza a los elementos en el eje X (horizontal)

Ejemplo:

```
.caja {
    transform: translateX(50px);
}
```

100% = width del elemento

TranslateY()

Desplaza a los elementos en el eje Y (vertical)

Ejemplo:

```
.caja {
    transform: translateY(-70%);
}
```

100% = height del elemento

SCALE

Scale()

Shorthand de ScaleX y ScaleY.

Estira determinado elemento (a través de vectores)

- Se le otorga un valor proporcional

Ejemplo:

```
.caja {
    transform: scale(1.8, 2.5);
}
```

ScaleX()

Estira determinado elemento en el eje X (horizontal)

- Se le otorga un valor proporcional

Ejemplo:

```
.caja {
    transform: scaleX(1.5);
}
```

ScaleY()

Estira determinado elemento en el eje Y (vertical)

- Se le otorga un valor proporcional

Ejemplo:

```
.caja {
    transform: scaleY(2);
}
```

SKEW

Skew()

Deforma determinado elemento.

- Se le otorga un valor en grados sexagesimales (deg), centesimales (grad) o radianes (rad).

Ejemplos:

1)

```
.caja {
    transform: skew(40deg);
}
```

2)

```
.caja {
    transform: skew(-250deg);
}
```

ROTATE

Rotate()

Rota determinado elemento.

- Se le otorga un valor en grados sexagesimales (deg), centesimales (grad) o radianes (rad).

Ejemplos:

```
.caja {
    transform: rotate(30deg);
}
```

Clip Path

sábado, 8 de julio de 2023 22:41

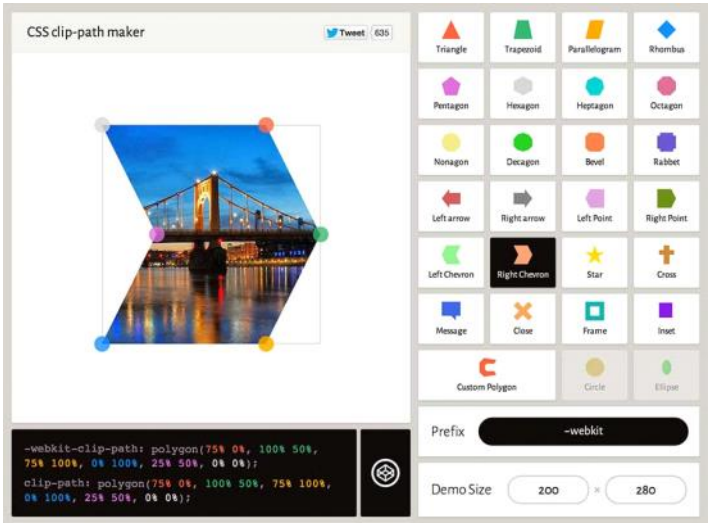
Clip-path()

Permite recortar un elemento a diversas formas geométricas.

- Recurso: https://bennettfeely.com/clippy/

Ejemplo:

```
.caja {
    clip-path: polygon(50% 0%, 0% 100%, 100% 100%);
}
```



Background

sábado, 8 de julio de 2023 22:52

Background-color

Define el color de fondo.

```
.bg {
    background-color: slateblue;
}
```

Background-img

Coloca una imagen de fondo.

```
.bg {
    background-img: url(https://url-imagen);
}
```

Background-size

Determina el tamaño del fondo.

Valores:

`contain`	La imagen se ajusta a su resolución real pero deja un espacio en el contenedor si es necesario.
`cover`	La imagen se ajusta al contenedor sin perder la resolución.
none	La imagen va a mantener su tamaño exacto sin importar el tamaño del contenedor. No se va a agrandar ni achicar.
`scale-down`	Se queda con la mejor propiedad dependiendo de la situación

```
.bg {
    background-size: cover;
}
```

Background-repeat

Determina si la imagen de fondo se va a repetir o no.

- En el caso de que la imagen no se repita y quede un espacio vacío, este se rellenará con el color de fondo definido.

Valores:

- repeat
- repeat-x
- repeat-y
- no-repeat

```
.bg {
    background-repeat: no-repeat;
```

```
}
```

Background-clip

Determina a partir de qué punto se va a mostrar la imagen de fondo. (Aplicando un recorte)

Valores:

- content-box
- padding-box
- border-box

```
.bg {
    background-clip: border-box;
}
```

Background-origin

Determina desde qué punto se crea la imagen de fondo, SIN RECORTARSE.

Valores:

- content-box
- padding-box
- border-box

```
.bg {
    background-origin: border-box;
}
```

Background-position

Determina la posición de la imagen de fondo.

Valores:

- left
- right
- top
- bottom
- center

```
.bg {
    background-position: left top;
}
```

Background-attachment

Valores:

- scroll (por defecto)
- fixed (imagen fija)
- inherit (heredado del contenedor)

```
.bg {
```

```
    background-attachment: fixed;
}
```

Variables

sábado, 8 de julio de 2023 23:13

Una variable es un espacio que se almacena en memoria y guarda un valor.

Existen dos tipos de variables: globales y locales.

- Las GLOBALES son aquellas que pueden invocarse desde cualquier elemento.
- Las LOCALES son aquellas que se declaran solo para ser invocadas por determinado tipo de elemento, clase, id, etc.

Declarar una variable en CSS

Global:

```
:root {
     --color-rojo: #f43;
}

.caja {
     background: var(--color-rojo);
}
```

Local:

```
div {
     --color-rojo: #f43;
}

div div {
     background: var(--color-rojo);
}
```

Filtros

sábado, 8 de julio de 2023 23:25

Propiedad que permite aplicar y ajustar efectos gráficos en los elementos. (especialmente imágenes)

Filtro	Unit	Función
none	px	Por defecto
blur	px	Desenfoque gaussiano
brightness()	number or %	Brillo
contrast()	number or %	Contraste
drop-shadow()	px - px - px - color	Sombra (img transparentes)
grayscale()	number or %	Escala de grises
hue-rotate()	deg - rad - grad	Rotar la gama de colores
invert()	number or %	Invertir la gama de colores
opacity()	number or %	Opacidad
saturate()	number or %	Saturación
sepia()	number or %	Efecto sepia
url()	("filters.svg#filter.id")	Personalizado

Ejemplo:

```
.img {
    filter:
        brightness(2.3)
        grayscale(40%)
        contrast(1.5)
        drop-shadow(10px 10px 5px #000)
    ;
}
```

Otras propiedades

domingo, 9 de julio de 2023 11:34

Direction

Ordena el texto de determinada forma.

`ltr`	De izquierda a derecha
`rtl`	De derecha a izquierda
`initial`	Por defecto
`inherit`	Valor heredado

Letter-spacing

Separación entre letras.

```
.caja {
    letter-spacing: 1px;
}
```

Scroll-behavior

Sensibilidad del scroll al redirigirte de una parte de la página a otra.

Valores:

- Smooth (progresivo)
- Auto (brusco)

User-selected

Definir si determinado texto va a poder ser seleccionado, o no.

Su función es principalmente que cierto texto no pueda ser copiado.

Valores:

- None (no se puede seleccionar)
- Auto (sí se puede seleccionar)

Text-shadow

Le otorga sombra a determinado texto.

Ejemplo:

```
.caja {
    text-shadow: (2px 2px 1px #000);
}
```

Text-decoration

Le otorga determinado estilo al texto.

Valores:

- underline
- dotted
- wavy
- overline
- none

Ejemplo:

```
.caja {
    text-decoration: underline;
}
```

Más selectores

Recurso: https://www.w3schools.com/cssref/css_selectors.php