

2.9. Cascading Style Sheet Nivel Intermedio

Hasta ahora hemos visto el comportamiento de los contenedores. Veamos ahora **cómo controlar el comportamiento del contenido del contenedor**.

2.9.1 Propiedad overflow

La propiedad **overflow** (desbordamiento o excedente) nos permite controlar el **comportamiento del contenido que se encuentra en una caja o contenedor**. Por lo tanto, mediante esta propiedad podremos especificar si queremos recortar un contenido, mostrar barras de desplazamiento o mostrar el contenido que excede de un elemento a nivel de bloque.

Solo funciona sobre elementos de tipo bloque con una altura definida.

Propiedad	Descripción	Valores
Overflow	Desbordamiento o excedente	visible hidden scroll auto
Overflow-x	Desbordamiento eje x horizontal	
overflow-y	Desbordamiento eje y vertical	

Ejemplo overflow

HTML	CSS
<pre> <html> <head> <meta charset="utf-8"> <title>Propiedad padding</title> <link rel="stylesheet" href="styles.css"> </head> <body> <h1>Propiedad overflow</h1> <h2>overflow: visible (default):</h2> <div class="visible">Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh eismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Lorem ipsum dolor sit amet, consectetuer adipiscing elit.</div>

 <h2>overflow: scroll</h2> <div class="scroll">Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh eismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh eismod tincidunt ut laoreet dolore magna aliquam erat volutpat.</div> <h2>overflow: hidden</h2> <div class="hidden">Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh eismod tincidunt ut laoreet dolore magna aliquam </pre>	<pre> .visible { background-color: lightyellow; width: 150px; height: 150px; overflow: visible; /*Contenido sale del elemento y es completamente visible.*/ } .scroll { background-color: lightgreen; width: 150px; height: 150px; overflow: scroll; /*Muestra una barra de scroll (horizontal y vertical) cuando los contenidos no caben en el contenedor.*/ } .hidden { background-color: lightblue; width: 150px; height: 150px; overflow: hidden; /*Contenido que sale del contenedor padre se oculta y no se muestran barras de scroll. */ } .auto { background-color: lightgrey; width: 150px; height: 150px; overflow: auto; } /*Ejemplo de overflow:hidden con floats*/ .auto-img { background-color: lightgrey; overflow: hidden; </pre>

```
erat volutpat. Lorem ipsum dolor sit amet,
consectetuer adipiscing elit, sed diam nonummy nibh
euismod tincidunt ut laoreet dolore magna aliquam
erat volutpat.</div>
```

```
<h2>overflow: auto con scroll</h2>
```

```
<div class="auto">Lorem ipsum dolor sit amet,
consectetuer adipiscing elit, sed diam nonummy nibh
euismod tincidunt ut laoreet dolore magna aliquam
erat volutpat. Lorem ipsum dolor sit amet,
consectetuer adipiscing elit, sed diam nonummy nibh
euismod tincidunt ut laoreet dolore magna aliquam
erat volutpat.</div>
```

```
<h2>overflow: hidden con floats</h2>
```

<p>En este caso la imagen excede el contenedor y tiene la propiedad float: left. Al establecer la propiedad "overflow: auto o hidden" conseguimos que el contenido no desborde el contenedor.</p>

```
<div class="auto-img">
```

```
Lorem ipsum dolor sit amet, consectetuer adipiscing
elit, sed diam nonummy nibh euismod tincidunt ut
laoreet dolore magna aliquam erat volutpat. Lorem
ipsum dolor sit amet, consectetuer adipiscing elit, sed
diam nonummy nibh euismod tincidunt ut laoreet
dolore magna aliquam erat volutpat.</div>
```

```
</body>
```

```
</html>
```

```
}
.auto-img img{
float: left;
max-height: 300px;
}
```

Propiedad overflow

overflow: visible (default):

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Lorem ipsum dolor sit amet, consectetuer adipiscing elit.

overflow: scroll

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

overflow: hidden

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

Ejemplo overflow-x y overflow-y

Cuando el contenido dentro del .container sea más ancho que 200 píxeles, en lugar de recortar el contenido o mostrarlo fuera del contenedor, se agregarán barras de desplazamiento horizontales que permitirán al usuario desplazarse y ver el contenido oculto.

Esto es útil cuando deseas **asegurarte de que el contenido desbordado sea accesible** para los usuarios que pueden no estar viendo el sitio en dispositivos con pantallas lo suficientemente anchas para mostrar todo el contenido a la vez.

```
.container {  
  width: 200px;  
  height: 100px;  
  overflow-x: scroll;  
}
```

Cuando el contenido dentro del `.container` sea más alto que 150 píxeles, el contenido que exceda este límite de altura no será visible. **En lugar de mostrar barras de desplazamiento verticales para permitir que el usuario vea el contenido oculto, el contenido se recortará y quedará oculto dentro del contenedor.**

```
.container {  
  width: 300px;  
  height: 150px;  
  overflow-y: hidden;  
}
```

Es importante tener en cuenta que **overflow-x y overflow-y pueden combinarse para controlar el desbordamiento en ambas dimensiones**. En el siguiente ejemplo, el contenedor tendrá barras de desplazamiento en el eje x si el contenido se desborda horizontalmente, y barras de desplazamiento en el eje y si el contenido se desborda verticalmente.

```
.container {  
  width: 400px;  
  height: 200px;  
  overflow-x: auto;  
  overflow-y: scroll;  
}
```

2.9.2 Funciones matemáticas

Algunas de las funciones matemáticas más comunes en CSS son las que se muestran a continuación.

Propiedad	Descripción	Valores Ejemplo
<code>abs()</code>	Devuelve el valor absoluto de un número	<code>abs(-10)</code> = 10
<code>sin()</code>	Devuelve el seno de un ángulo	<code>sin(45deg)</code> ≈ 0.71
<code>cos()</code>	Devuelve el coseno de un ángulo	<code>cos(60deg)</code> = 0.5
<code>tan()</code>	Devuelve la tangente de un ángulo	<code>tan(30deg)</code> ≈ 0.58
<code>sqrt()</code>	Devuelve la raíz cuadrada de un número	<code>sqrt(16)</code> = 4
<code>pow()</code>	Eleva un número a una potencia	<code>pow(2, 3)</code> = 8
<code>min()</code>	Devuelve el valor mínimo entre dos números	<code>min(10, 5)</code> = 5
<code>max()</code>	Devuelve el valor máximo entre dos números	<code>max(10, 5)</code> = 10
<code>random()</code>	Devuelve un número aleatorio entre 0 y 1	<code>random()</code> ≈ 0.35
<code>floor()</code>	Devuelve el valor entero menor o igual	<code>floor(3.8)</code> = 3
<code>ceil()</code>	Devuelve el valor entero mayor o igual	<code>ceil(3.2)</code> = 4
<code>round()</code>	Devuelve el valor redondeado al entero más cercano	<code>round(3.7)</code> = 4
<code>clamp()</code>	Limita un valor dentro de un rango	<code>clamp(10, 5, 8)</code> = 8
<code>calc()</code>	Realiza cálculos matemáticos en propiedades CSS	<code>width: calc(100% - 20px)</code>
<code>attr()</code>	Obtiene el valor de un atributo HTML y lo utiliza en CSS	<code>content: attr(data-text)</code>

calc()

La función matemática `calc()` en CSS se utiliza para realizar cálculos aritméticos y combinar diferentes unidades de medida. Algunos ejemplos de cómo se puede utilizar:

Ejemplo 1: Suma de valores. En este ejemplo, se realiza la suma de 100px y 20px, lo que da como resultado un ancho de 120px.

`width: calc(100px + 20px); /* Resultado: 120px */`

Ejemplo 2: Resta de valores. En este caso, se resta 10px al 80% del ancho disponible, lo que dará como resultado un ancho relativo.

`width: calc(80% - 10px); /* Resultado: El 80% del ancho disponible menos 10px */`


Ejemplo 3: Multiplicación de valores. Aquí se realiza la multiplicación de 2 por el 50% del ancho disponible, obteniendo un ancho relativo igual al doble del 50%.

width: calc(2 * 50%); /* Resultado: El doble del 50% del ancho disponible */

Ejemplo 4: Combinación de operaciones. En este ejemplo, se realiza una combinación de operaciones donde se suma 100px con el 10% del ancho disponible y luego se divide el resultado por 3.

width: calc((100px + 10%) / 3); /* Resultado: El resultado de la operación combinada */

Ejemplo calc(): Realización de cálculos con calc() de CSS para conseguir un contenedor fijo y otro variable.

HTML	CSS
<pre> <html> <head> <meta charset="utf-8"> <title>Propiedad padding</title> <link rel="stylesheet" href="styles.css"> </head> <body> <div id="div1">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. </div> <div id="div2">Other text</div> <div id="div1">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. </div> <div id="div2"><button>Button</button></div> <div id="div1">Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat.</div> <div id="div2"> </div> </body> </html> </pre>	<pre> #div1 { float:left; width: calc(100% - 230px); background-color: yellow; padding: 15px; margin-bottom: 15px; clear: both; box-sizing: border-box; } #div2{ float:left; width: 230px; padding: 15px; box-sizing: border-box; } </pre> 

min()

Se utiliza para obtener el valor mínimo entre varios valores o dimensiones.

Ejemplo 1: Obtener el valor mínimo entre dos anchos. En este ejemplo, se obtiene el valor mínimo entre 200px y el 50% del ancho disponible, lo que dará como resultado el valor más pequeño entre ambos.

width: min(200px, 50%); /* Resultado: El valor mínimo entre 200px y el 50% del ancho disponible */

Ejemplo 2: Obtener el valor mínimo entre tres alturas. Aquí se obtiene el valor mínimo entre 100px, 150px y el 50% del alto disponible, resultando en la altura más pequeña entre los tres valores.

height: min(100px, 150px, 50%); /* Resultado: El valor mínimo entre 100px, 150px y el 50% del alto disponible */

Ejemplo 3: Combinar valores absolutos y relativos. En este caso, se obtiene el valor mínimo entre 16px y 1.2 veces el tamaño de fuente actual, lo que dará como resultado el tamaño de fuente más pequeño entre ambos.

font-size: min(16px, 1.2em); /* Resultado: El valor mínimo entre 16px y 1.2 veces el tamaño de fuente actual */

max()

Se utiliza para obtener el valor máximo entre varios valores o dimensiones.

Ejemplo 1: Obtener el valor máximo entre dos anchos. En este ejemplo, se obtiene el valor máximo entre 200px y el 50% del ancho disponible, lo que dará como resultado el valor más grande entre ambos.

width: max(200px, 50%); /* Resultado: El valor máximo entre 200px y el 50% del ancho disponible */

Ejemplo 2: Obtener el valor máximo entre tres alturas. Aquí se obtiene el valor máximo entre 100px, 150px y el 50% del alto disponible, resultando en la altura más grande entre los tres valores.

height: max(100px, 150px, 50%); /* Resultado: El valor máximo entre 100px, 150px y el 50% del alto disponible */

Ejemplo 3: Combinar valores absolutos y relativos. En este caso, se obtiene el valor máximo entre 16px y 1.2 veces el tamaño de fuente actual, lo que dará como resultado el tamaño de fuente más grande entre ambos.

font-size: max(16px, 1.2em); /* Resultado: El valor máximo entre 16px y 1.2 veces el tamaño de fuente actual */

round()

Se utiliza para redondear un número al entero más cercano.

Ejemplo 1: Redondear un número decimal. En este ejemplo el número decimal 3.7rem se redondea al entero más cercano, que es 4rem.

width: round(3.7rem); /* Resultado: 4rem */

Ejemplo 2: Redondear un número negativo. Aquí, el número negativo -2.3em se redondea al entero más cercano, que es -2em.

padding: round(-2.3em); /* Resultado: -2em */

Ejemplo 3: Redondear un número con decimales a un número específico de lugares decimales. En este caso, el número 1.2345 se redondea a 2 lugares decimales, resultando en 1.23.

font-size: round(1.2345, 2); /* Resultado: 1.23 */

attr()

Se utiliza para obtener el valor de un atributo HTML y utilizarlo en propiedades CSS.

Ejemplo 1: Obtener el valor de un atributo y utilizarlo como ancho. En este ejemplo, se obtiene el valor del atributo `data-width` y se utiliza como valor para la propiedad `width`. Esto puede ser útil cuando se necesita establecer dinámicamente el ancho de un elemento basado en un atributo específico.

width: attr(data-width);


Ejemplo 2: Obtener el valor de un atributo y utilizarlo en una operación matemática. Aquí, se obtiene el valor del atributo `data-height` y se utiliza en una operación matemática dentro de la función `calc()`. Esto permite realizar cálculos con el valor del atributo y aplicar el resultado como valor para la propiedad `height`.

height: calc(attr(data-height) * 2);

Ejemplo 3: Obtener el valor de un atributo y utilizarlo como contenido de un pseudo-elemento. En este caso, se obtiene el valor del atributo `data-content` y se utiliza como contenido para un pseudo-elemento. Esto permite mostrar dinámicamente el contenido de un atributo HTML en el estilo de un elemento.

content: attr(data-content);

Ejemplo attr(): utilizamos el atributo `data-author` para mostrarlo como contenido en el documento HTML.

HTML	CSS
<pre><html> <head> <meta charset="utf-8"> <title>Propiedad padding</title> <link rel="stylesheet" href="styles.css"> </head> <body> <p class="element" data-author="Soy yo">El autor es </p> </body> </html></pre>	<pre>.element::after { content: attr(data-author); font-weight: bold; color: green; }</pre> 

2.9.3 Cursores

En CSS, los cursores se utilizan para **cambiar la apariencia del cursor** del ratón cuando se encuentra sobre un elemento interactivo. Esto permite personalizar la forma del cursor y proporcionar retroalimentación visual al usuario.

Son los que se muestran a continuación.

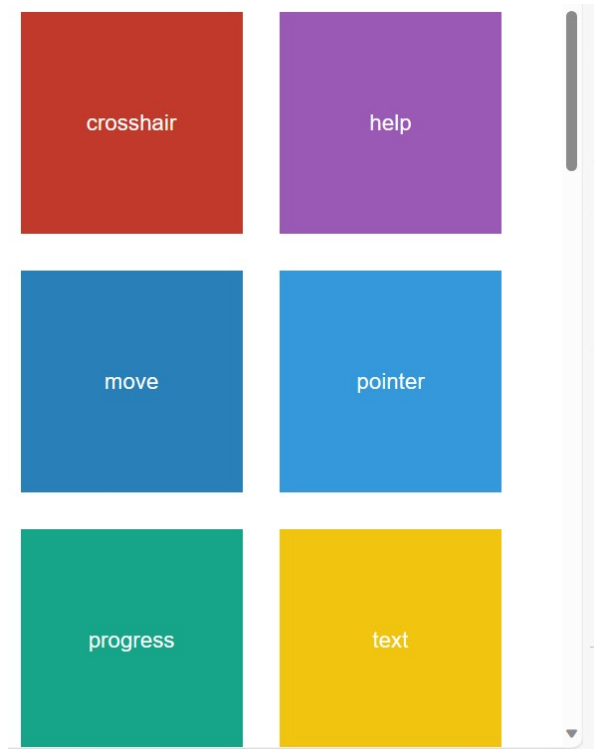
Valor	Descripción
crosshair	Cursor tipo cruz para tareas en las que se requiere precisión.
help	Interrogación. Cursor de ayuda.
move	Flechas hacia todos lados. Cursor para mover elementos.
pointer	Mano o apuntador. Cursor para hacer clic.
progress	Barra progreso o similar. Cursor que indica que se está trabajando en segundo plano.
text	Cursor que permite seleccionar texto.
wait	Cursor que indica que se debe esperar.
vertical-text	Flecha vertical, utilizado para texto en orientación vertical.
alias	Flecha diagonal, utilizado para indicar que se puede acceder a un enlace o recurso externo.
no-drop	Círculo con una línea diagonal, utilizado para indicar que no se permite soltar el elemento arrastrado.
grab	Mano cerrada, utilizado para indicar que el elemento se puede agarrar
grabbing	Mano abierta, utilizado para indicar que el elemento está siendo arrastrado o agarrado.
zoom-in	Símbolo de lupa con un signo de más, utilizado para indicar que se puede hacer zoom in.
zoom-out	Símbolo de lupa con un signo de menos, utilizado para indicar que se puede hacer zoom out.

not-allowed	Círculo con una línea diagonal, utilizado para indicar que la interacción no está permitida.
cell	Cursor tipo cruz, utilizado en hojas de cálculo para ajustar celdas.
copy	Puntero con un símbolo de copia, utilizado para indicar que se puede realizar una copia de un elemento.
ew-resize	Flecha bidireccional horizontal, utilizado para indicar que se puede ajustar el tamaño horizontalmente.
ns-resize	Flecha bidireccional vertical, utilizado para indicar que se puede ajustar el tamaño verticalmente.
nwse-resize	Flecha bidireccional diagonal, utilizado para indicar que se puede ajustar el tamaño diagonalmente en dirección noroeste-sureste.
nesw-resize	Flecha bidireccional diagonal, utilizado para indicar que se puede ajustar el tamaño diagonalmente en dirección noreste-suroeste.
col-resize	Flecha bidireccional horizontal con una línea vertical, utilizado para indicar que se puede ajustar el tamaño de una columna.
row-resize	Flecha bidireccional vertical con una línea horizontal, utilizado para indicar que se puede ajustar el tamaño de una fila.
context-menu	Puntero con un símbolo de menú, utilizado para indicar que se puede abrir un menú contextual.
none	No se muestra ningún cursor.
Personalized	Imagen o valor personalizado en el cursor.

Estas explicaciones proporcionan una descripción breve de cada cursor, pero es posible encontrar más detalles y variaciones de estos cursores en la especificación CSS y en la documentación de los navegadores. Es importante **tener en cuenta que algunos valores de cursor pueden no ser compatibles en todos los navegadores o en todos los elementos.**

Ejemplo Cursores

HTML	CSS
<pre> <html> <head> <meta charset="utf-8"> <title>Propiedad padding</title> <link rel="stylesheet" href="styles.css"> </head> <body> <div class="container"> <div class="caja caja1">crosshair</div> <div class="caja caja2">help</div> <div class="caja caja3">move</div> <div class="caja caja4">pointer</div> <div class="caja caja5">progress</div> <div class="caja caja6">text</div> <div class="caja caja7">wait</div> <div class="caja caja8">vertical-text</div> <div class="caja caja9">alias</div> <div class="caja caja10">no-drop</div> <div class="caja caja11">grab</div> <div class="caja caja12">grabbing</div> <div class="caja caja13">zoom-in</div> <div class="caja caja14">zoom-out</div> <div class="caja caja15">not-allowed</div> <div class="caja caja16">no-drop</div> <div class="caja caja17">copy</div> <div class="caja caja18">ew-resize</div> <div class="caja caja19">ns-resize</div> <div class="caja caja20">nwse-resize</div> <div class="caja caja21">nesw-resize</div> <div class="caja caja22">col-resize</div> <div class="caja caja23">row-resize</div> <div class="caja caja24">context-menu</div> <div class="caja caja25">none</div> <div class="caja caja26">Personalized</div> </div> </body> </html> </pre>	<pre> .container{ position: relative; } .caja{ width: 180px; height: 180px; color: #FFF; text-align: center; line-height: 180px; margin: 15px; float: left; font-size: 18px; font-family: Arial; } .caja1{ background-color: #C0392B; cursor: crosshair; } .caja2{ background-color: #9B59B6; cursor: help; } .caja3{ background-color: #2980B9; cursor: move; } .caja4{ background-color: #3498DB; cursor: pointer; } .caja5{ background-color: #17A589; cursor: progress; } .caja6{ background-color: #F1C40F; cursor: text; } .caja7{ background-color: #E67E22; cursor: wait; } .caja8{ background-color: #95A5A6; cursor: vertical-text; } .caja9{ background-color: #2C3E50; cursor: alias; } .caja10{ background-color: grey; cursor: no-drop; } .caja11{ background-color: black; cursor: grab; } </pre>



```

}
.caja12{
  background-color: #C0392B;
  cursor: grabbing;
}
.caja13{
  background-color: #9B59B6;
  cursor: zoom-in;
}
.caja14{
  background-color: #2980B9;
  cursor: zoom-out
}
.caja15{
  background-color: #3498DB;
  cursor: not-allowed;
}
.caja16{
  background-color: #17A589;
  cursor: progress;
}
.caja17{
  background-color: #F1C40F;
  cursor: copy
}
.caja18{
  background-color: #E67E22;
  cursor: ew-resize;
}
.caja19{
  background-color: #95A5A6;
  cursor: ns-resize;
}
.caja20{
  background-color: #2C3E50;
  cursor: nwse-resize;
}
.caja21{
  background-color: grey;
  cursor: nesw-resize;
}
.caja22{
  background-color: #2980B9;
  cursor: col-resize;
}
.caja23{
  background-color: #3498DB;
  cursor: row-resize;
}
.caja24{
  background-color: #17A589;
  cursor: context-menu;
}
.caja25{
  background-color: #E67E22;
  cursor: none;
}
.caja26{
  background-color: #95A5A6;
  cursor:
url(https://cdn-icons-png.flaticon.com/128/2662/266
2839.png), pointer;}

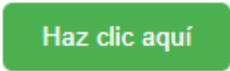
```

2.9.4 Botones


Ejemplo básico de creación de botón.

Ejemplo básico de creación de un botón.

En este ejemplo, hemos definido una **clase** **.my-button** que se aplica al botón en el HTML. Utilizamos propiedades CSS como padding, background-color, color, border, border-radius y cursor para estilizar el botón. También hemos añadido estilos para el estado :hover (cuando el cursor está sobre el botón) y :active (cuando el botón está siendo clicado).

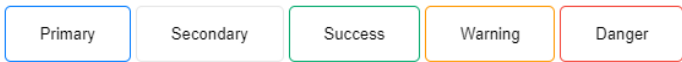
HTML	CSS
<pre><html> <head> <meta charset="utf-8"> <title>Propiedad padding</title> <link rel="stylesheet" href="styles.css"> </head> <body> <button class="my-button">Haz clic aquí</button> </body> </html></pre> 	<pre>.my-button { padding: 10px 20px; background-color: #4CAF50; color: white; border: none; border-radius: 5px; cursor: pointer; } .my-button:hover { background-color: #45a049; } .my-button:active { background-color: #3e8e41; }</pre>

Ejemplo botones tipo alert.


HTML	CSS
<pre><button class="btn primary">Primary</button> <button class="btn secondary">Secondary</button> <button class="btn success">Success</button> <button class="btn warning">Warning</button> <button class="btn danger">Danger</button></pre> 	<pre>.btn { border: none; color: white; padding: 14px 28px; cursor: pointer; border-radius: 5px;} .primary {background-color: #007bff;} .primary:hover {background: #0b7dda;} .secondary {background-color: #e7e7e7; color: black;} .secondary:hover {background: #ddd;} .success {background-color: #04AA6D;} .success:hover {background-color: #46a049;} .warning {background-color: #ff9800;} .warning:hover {background: #e68a00;} .danger {background-color: #f44336;}</pre>

```
.danger:hover {background: #da190b;}
```

Ejemplo botones tipo outline.


HTML	CSS
<pre><button class="btn primary">Primary</button> <button class="btn secondary">Secondary</button> <button class="btn success">Success</button> <button class="btn warning">Warning</button> <button class="btn danger">Danger</button></pre> 	<pre>.btn { border: none; color: black; padding: 14px 28px; cursor: pointer; border-radius: 5px; } .primary {background-color: white; border: 2px solid #007bff;} .primary:hover {background: #0b7dda; color: white;} .secondary {background-color: white; border: 2px solid #e7e7e7;} .secondary:hover {background: #ddd; color: white;} .success {background-color: white; border: 2px solid #04AA6D;} .success:hover {background-color: #46a049; color: white;} .warning {background-color: white; border: 2px solid #ff9800;} .warning:hover {background: #e68a00; color: white;} .danger {background-color: white; border: 2px solid #f44336;} .danger:hover {background: #da190b; color: white;}</pre>

Ejemplo botones que ocupen el 100% del contenedor

HTML	CSS
<pre><button class="btn primary btn-block">Primary</button> <button class="btn secondary btn-block">Secondary</button> <button class="btn success btn-block">Success</button> <button class="btn warning btn-block">Warning</button> <button class="btn danger btn-block">Danger</button></pre>	<pre>.btn-block{ width: 100%; display: block; margin-bottom: 5px; }</pre> 

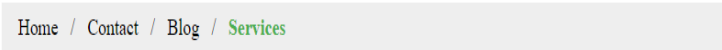
Ejemplo Crear un paginado en CSS

Crear un paginado en CSS implica utilizar **estilos para diseñar los elementos que formarán parte de la paginación, como los botones de navegación y los números de página.**

HTML	CSS
<pre><ul class="pagination"> « 1 2 3 4 5 6 7 » </pre> 	<pre>ul.pagination { display: inline-block; padding: 0; margin: 0; } ul.pagination li {display: inline;} ul.pagination li a { color: black; float: left; padding: 9px 18px; text-decoration: none; } ul.pagination li a.active { background-color: #4CAF50; color: white; } ul.pagination li a.active:hover {background-color: #397f3b;} ul.pagination li a:hover:not(.active) {background- color: #ddd;}</pre>

Ejemplo Crear migas de pan en CSS

Crear migas de pan o breadcrumbs.

HTML	CSS
<pre><ul class="breadcrumb"> Home Contact Blog Services </pre> 	<pre>ul.breadcrumb { padding: 8px 16px; list-style: none; background-color: #eee; } ul.breadcrumb li {display: inline;} ul.breadcrumb li+li::before { padding: 8px; color: black; content: "\00a0"; } ul.breadcrumb a:link, ul.breadcrumb a:visited { text-decoration:none; color: black; } ul.breadcrumb li a.active { color: #4CAF50; font-weight: bold; }</pre>

```
}
ul.breadcrumb li a.active:hover {
  color: #397f3b;
  font-weight: bold;
}
```

2.9.5 Formatos de las fuentes en CSS y uso de repositorios.

Actualmente no hay un formato de fuente único para todos los navegadores. Por este motivo, es necesario incluir las fuentes en todos los formatos para que sean visibles en el mayor número de navegadores. Los formatos más destacados son los siguientes:

Formato	Descripción
EOT	Embedded OpenType, Explorer
TTF	TrueType Font, IOS
WOFF	Web Open Font Format, Chrome
WOFF2	Mejora de WOFF
SVG	Scalar Vector Graphics

Regla @font-face en CSS

La regla @font-face **nos permite descargar una fuente o tipografía, cargarla en el navegador y utilizarla en nuestras páginas**. Así pues, tendríamos el siguiente código para incluir la fuente Open Sans:

```
@font-face {
  font-family: 'Open Sans';
  src: local('Open Sans'),
    url(opensans.woff2) format('woff2'),
    url(opensans.woff) format('woff'),
    url(opensans.ttf) format('truetype'),
    url(opensans.otf) format('opentype'),
    url(opensans.eot) format('embedded-opentype');
}
```

Repositorios de fuentes

Hoy en día es muy común utilizar un repositorio de fuentes que nos provea de los archivos necesarios para tener las fuentes disponibles en nuestra web.

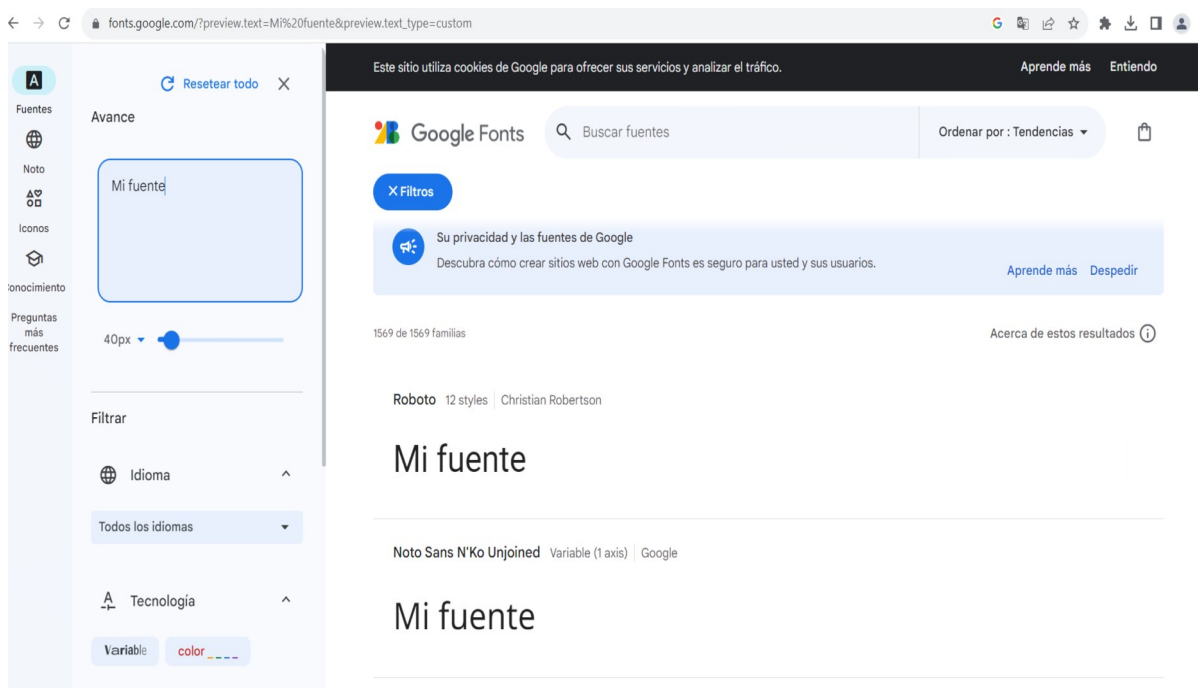
Uno de los repositorios proveedores de tipografías más utilizados es [Google Fonts](#). Los motivos de su uso son los siguientes:

- **Gratuito:** dispone de un amplio catálogo de fuentes gratuitas.
- **Cómodo:** su interfaz es de fácil uso.
- **Rápido:** utiliza un CDN que proporciona ventajas en cuanto a la velocidad.

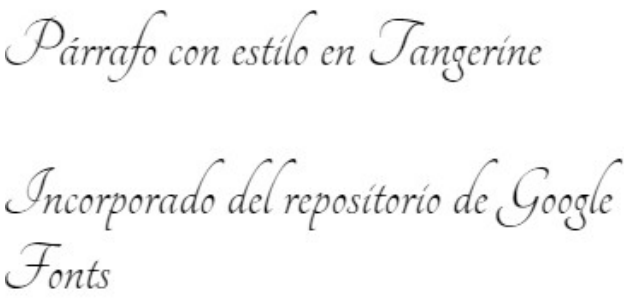
Cómo se usa Google Fonts

Google Fonts es un **servicio de Google que proporciona una amplia variedad de fuentes tipográficas de calidad para uso gratuito** en sitios web. Para usar el repositorio de fuentes de Google Fonts, sigue estos pasos:

1. **Accede al sitio web de Google Fonts:** Abre tu navegador web y visita fonts.google.com
2. **Explora las fuentes:** Puedes navegar por las diferentes fuentes disponibles utilizando las opciones de filtrado, como las categorías de fuentes, el grosor (weight), el estilo, etc. También puedes buscar una fuente específica usando el cuadro de búsqueda.
3. **Selecciona las fuentes:** Haz clic en las fuentes que desees usar. Una vez seleccionadas, se agregarán al panel de selección, ubicado en la parte inferior derecha de la página.
4. **Personaliza las opciones:** En el panel de selección, puedes personalizar las opciones de las fuentes, como el grosor, el estilo (normal, itálico), el conjunto de caracteres y más.
5. **Obtén el código de incrustación:** Una vez que hayas seleccionado todas las fuentes y ajustado las opciones, haz clic en el botón «Usar en Google Fonts» en la parte inferior del panel de selección.
6. **Incrusta las fuentes en tu sitio web:** En el panel lateral se pueden ver dos tipos de incrustación, mediante `<link>` o mediante `@import`.



Ejemplo Fuente

HTML	CSS
<pre><html> <head> <meta charset="utf-8"> <title>Propiedad font-size</title> <link rel="stylesheet" href="style.css"> <link rel="stylesheet" type="text/css" href="http://fonts.googleapis.com/css? family=Tangerine"> </head> <body> <p> Párrafo con estilo en Tangerine</p> <p>Incorporado del repositorio de Google Fonts</p> </body> </html></pre>	<pre>p{ font-family: 'Tangerine', cursive; font-size: 38px; }</pre> 

2.9.6 Sombras en texto y contenedores o cajas.

Propiedad text-shadow

El formato es prácticamente el mismo para las dos propiedades.

- **Text-shadow:** posx posy desenfoque color | none;
- **box-shadow:** posx posy desenfoque tamaño color inset | none;

Significado de las propiedades:

- **posx:** sombra que se aplica en el eje horizontal.
- **posy:** sombra que se proyecta en el eje vertical.
- **desenfoque o blur:** cantidad de sombra borrosa.
- **tamaño:** tamaño de la sombra hacia los lados.
- **color:** color de la sombra.
- **inset:** posiciona la sombra dentro del marco.

Importante:

- Para aplicar sombras arriba y a la izquierda se deben utilizar valores negativos.
- Para aplicar doble sombra se pueden poner dos valores separados con coma.
- Recuerda que para este tipo de propiedades debemos incluir los prefijos para navegadores.

Ejemplo

Veamos también un ejemplo de «box-shadow»:

box-shadow: 10px 12px 14px 2px rgba(0,0,0,0.47);

- 10px. La sombra estará desplazada 10 píxeles hacia la derecha desde el borde del elemento.
- 12px. La sombra estará desplazada 12 píxeles hacia abajo desde el borde del elemento.
- 14px. La sombra tendrá un desenfoque de 14 píxeles, lo que hará que la sombra tenga un aspecto difuso y suave.
- 2px. La sombra se extenderá 2 píxeles desde el tamaño definido por el blur-radius.

- Color: `rgba(0,0,0,0.47)`. El color de la sombra será un gris oscuro con una opacidad de 0.47, lo que significa que la sombra será algo transparente.

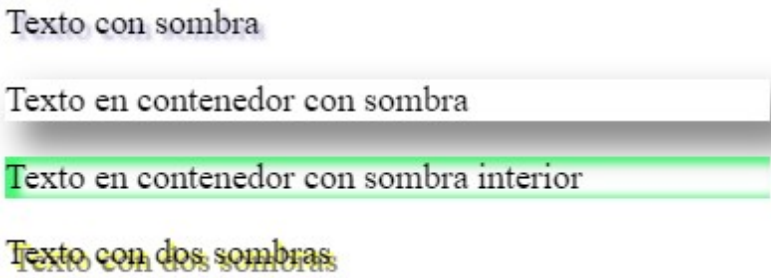
Ejemplo

Para aplicar varias sombras se puede añadir una lista de sombras separadas por comas. En el siguiente ejemplo se aplicarían dos sombras.

text-shadow: 3px 4px 0px #fff, 2px 2px 1px #ccc;

Recuerda que para este tipo de propiedades debemos incluir los **prefijos para navegadores** correspondientes. (**-moz-** para Firefox, **-ms-** para Microsoft Edge e Internet Explorer y **-webkit-** para Chrome)

Ejemplo

HTML	CSS
<pre><p class="shadow1">Texto con sombra</p> <p class="shadow2">Texto en contenedor con sombra</p> <p class="shadow3">Texto en contenedor con sombra interior</p> <p class="shadow4">Texto con dos sombras</p></pre>	<pre>.shadow1 {text-shadow: 3px 3px 2px rgba(152, 150, 207, 0.7);} .shadow2 {box-shadow: 10px 10px 14px 2px rgba(0,0,0,0.47);} .shadow3 {box-shadow: 5px 1px 4px 2px #45f875 inset;} .shadow4 {text-shadow: 3px 4px 0px grey, 2px 2px 1px yellow;}</pre> 

Generadores de sombras

Para facilitarnos la tarea de la creación de sombras podemos utilizar un generador de sombras tanto para los textos como para las cajas o contenedores. Algunas herramientas online útiles son las siguientes:

- Generador de sombras en textos y cajas: <https://cssgenerator.org/box-shadow-css-generator.html>

2.9.7 Gradientes.

Los gradientes nos permiten añadir efectos de colores degradados en nuestros diseños. Los gradientes están disponibles desde CSS3 y se configuran como fondos, por lo que tendremos que usar la propiedad "background". Disponemos de dos tipos de gradientes:

Gradiente lineal

Se define mediante la función **linear-gradient()** y permite crear fondos degradados en una dirección específica. El formato es el siguiente:

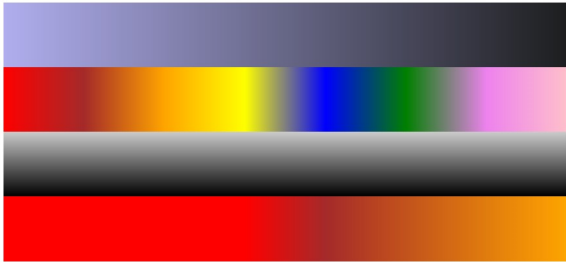
```
background: linear-gradient(dirección|ángulo, color, color, color...);
background: linear-gradient(dirección|ángulo, color posición, color posición, color posición...);
background: linear-gradient(color, color, color...);
```

Significado de las **propiedades**:

- **Dirección | ángulo**: indica la dirección o ángulo del gradiente. Puede ser especificado usando las palabras clave **to top**, **to bottom**, **to left** y **to right**. También puede ser indicado por un ángulo para declarar una dirección específica del gradiente.
- **Color**: permite crear una lista con varias paradas para ir cambiando el color del gradiente. Se pueden definir tantos colores como se necesiten. Se puede utilizar el nombre de color (en inglés), valor hexadecimal, color en RGB o HSL.
- **Posición**: altura a la que comienza a cambiar el color del gradiente.

Recuerda utilizar la extensión que te facilita la tarea de crear los prefijos para navegadores. En estas propiedades hay diferencias entre las funciones estándar y las especificadas por los navegadores.

Ejemplo

HTML	CSS
<pre><div></div> <div></div> <div></div> <div></div></pre> 	<pre>div:nth-child(1){ background: linear-gradient(90deg, rgba(176,174,238,1) 0%, rgba(29,30,31,1));} div:nth-child(2){ background: linear-gradient(to right,red,brown,orange,yellow,blue,green,violet,pink);} div:nth-child(3){ background: linear-gradient(to top, #000, #ccc);} div{padding:40px;} div:nth-child(4){ background: linear-gradient(to right,red 300px,brown 400px,orange 700px);}</pre>

Gradiente radial

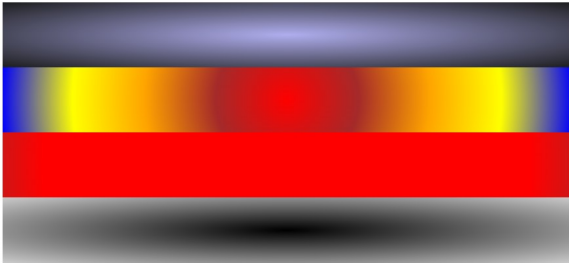
Permite crear degradados con formas circulares. Se define mediante la función **radial-gradient()** y un nuevo atributo para la forma. El formato es el siguiente:

```
background: radial-gradient(forma, color, color, color...);
background: radial-gradient(forma posicion, color, color, color...);
background: radial-gradient(forma, color, posicion, color posicion, color posicion...);
background: radial-gradient(color,color, color...);
```

Significado de las **propiedades**:

- **Forma:** existen dos tipos de forma: **circle y ellipse** (valor por defecto). De forma optativa se puede indicar el tamaño de la forma.
- **Color:** permite crear una lista con varias paradas para ir cambiando el color del gradiente. Se pueden definir tantos colores como se necesiten.
- **Posición:** para cada color se puede indicar la posición en la que comienzan las transiciones.

Ejemplo

HTML	CSS
<pre><div></div> <div></div> <div></div> <div></div></pre> 	<pre>div:nth-child(1){ background: radial-gradient(ellipse, rgba(176,174,238,1) 0%, rgba(29,30,31,1));} div:nth-child(2){ background: radial- gradient(circle,red,brown,orange,yellow,blue);} div:nth-child(3){ background:radial-gradient(circle,red 300px,brown 400px,orange 700px);} div:nth-child(4){ background: radial-gradient(#000, #ccc);} div{padding:40px;}</pre>

Generadores de gradientes

Para facilitarnos la tarea de la creación de gradientes podemos utilizar un generador online como: cssgradient.io

Patrones de gradientes

Podemos aplicar diseños muy trabajados con las plantillas que se pueden encontrar en: leaverou.github.io/css3patterns

2.9.8 Transiciones en CSS.

Las transiciones en CSS nos permiten aplicar un **cambio de estilo gradual a los elementos** del documento HTML. Además, nos ofrecen la ventaja de poder **especificar el tiempo** para que se produzca la transformación entre estilos, de esta forma podríamos utilizarlas para dar un **efecto de animación**.

Todos los parámetros para aplicar las transiciones se pueden establecer en una sola línea y también mediante propiedades por separado.

transition:[propiedad a modificar] [Duración] [Tipo de animación] [Retardo];

Propiedades:

all
background-color
border
border-radius
color
top
bottom
left
right
box-shadow
width
height
line-height
margin
opacity
word-spacing
letter-spacing
fill
padding
stroke
text-shadow
vertical-align
visibility
z-index

[Duración en segundos]: se debe especificar el número de segundos que va a durar la animación. Por ejemplo: 3s (3 segundos).

[Tipo de animación]: esta propiedad es opcional y sirve para indicar la velocidad de la animación. Algunas de las posibilidades son las siguientes:

- linear: la velocidad de la animación es uniforme en todo el recorrido.
- ease: la velocidad se acelera al inicio, luego se retarda un poco y se acelera al final de nuevo.
- ease-in: la animación empieza lentamente y va aumentando progresivamente.
- ease-out: la animación empieza muy rápida y va descendiendo progresivamente.
- ease-in-out: la animación empieza y acaba lentamente, y es en la parte central del recorrido donde la velocidad es más rápida.

[Retardo]: tiempo (en segundos) que el navegador esperará antes de poner en marcha la animación. Se especifica el número de segundos a esperar seguido de la «s» (ejemplo: 1s).

NOTA: No olvides utilizar los prefijos para los navegadores**Ejemplo**

Crea un contenedor de 180 píxeles de ancho y de alto, incluye un texto en su interior y aplícale un color de fondo. Realiza las siguientes transiciones cuando el usuario pase por encima del contenedor el puntero del ratón:

- Cambia el color de fondo del contenedor.
- Cambia el tamaño de las letras del contenedor.
- Cambia el color de las letras del contenedor.
- Cambia el valor del interlineado.
- Cambia el valor de la propiedad border-radius del contenedor.
- Cambia el valor de la sombra del contenedor.
- Cambia la opacidad del contenedor.
- Cambia tres propiedades más a tu elección.

HTML	CSS
<pre><div class="container"> <div class="caja caja1">Background</div> <div class="caja caja2">Font-size</div> <div class="caja caja3">Color</div> <div class="caja caja4">Line- height</div> <div class="caja caja5">Border- radius</div> <div class="caja caja6">Box- shadow</div> <div class="caja caja7">Opacity</div> <div class="caja caja8">Border</div> <div class="caja caja9">Margin</div> <div class="caja caja10">Padding</div> </div></pre>	<pre>.container{ position: relative;} .caja{ width: 180px; height: 180px; background-color: #30B037; color: #FFF; text-align: center; line-height: 180px; margin: 15px; float: left; font-size: 18px; font-family: Arial;} .caja1{ background-color: #C0392B; -webkit-transition: background-color 1s linear; transition: background-color 1s linear;} .caja1:hover{ background-color: #3F51B5;} .caja2{ background-color: #9B59B6; -webkit-transition: font-size 1s ease; transition: font-size 1s ease;} .caja2:hover{ font-size: 28px;} .caja3{ background-color: #2980B9; -webkit-transition: color 1s linear; transition: color 1s linear;} .caja3:hover{ color: yellow;} .caja4{ background-color: #3498DB; -webkit-transition: 1s linear; transition: 1s linear;} .caja4:hover{ line-height: 300px;} .caja5{ border-radius: 0px; background-color: #17A589;</pre>

Background	Font-size	Color	-webkit-transition: 1s linear; transition: 1s linear}
Line-height	Border-radius	Box-shadow	.caja5:hover{ border-radius: 20px;}
Opacity	Border	Margin	.caja6{ -webkit-transition: 1s linear; transition: 1s linear; background-color: #F1C40F;}
			.caja6:hover{ -webkit-box-shadow: 10px 10px 14px 2px rgba(0,0,0,0.47); box-shadow: 10px 10px 14px 2px rgba(0,0,0,0.47);}
			.caja7{ -webkit-transition: 1s linear; transition: 1s linear; background-color: #E67E22;}
			.caja7:hover{ opacity: 0.3;}
		Padding	.caja8{ -webkit-transition: 1s linear; transition: 1s linear; background-color: #95A5A6;}
			.caja8:hover{ border-style: solid; border-color: #95A5A6; border-color: #000;}
			.caja9{ -webkit-transition: 1s linear; transition: 1s linear; background-color: #2C3E50;}
			.caja9:hover{ margin: 80px;}
			.caja10{ -webkit-transition: 1s linear; transition: 1s linear; background-color: black;}
			.caja10:hover{ padding: 30px;}

Ejemplo

Crea un menú con transiciones de color.

HTML	CSS
<pre> Servicios Proyectos Recursos </pre>	<pre>/* Menú horizontal con transiciones */ /* Formato general de cada item */ .menu { display: block; position: relative; height: 76px; width: 120px; margin: 0; overflow: hidden;} /* Bordes superiores */ .menu.servicios { border-top: #ffc000 0.3em solid;} .menu.recursos { border-top: #b81800 0.3em solid;} .menu.proyectos {</pre>

```

</span>
</a></li>
<li><a class="menu contacto"
href="#" data-title="Contacto">
<span>
<span>Contacto</span>
</span>
</a></li>
</ul>

```



```

border-top: #00a53c 0.3em solid;}

.menu.contacto {
border-top: #9600b4 0.3em solid;}

/* Atributos y transición */
.menu > span {
display: block;
position: absolute;
overflow: hidden;
left: 0;
top: 0;
width: 100%;
height: 0%;
transition: 0.5s ease-in-out;
-webkit-transition: 0.5s ease-in-out;}

.menu:after, .menu > span > span {
display: block;
text-align: center;
border-radius: 0em;
padding: 2em 0 1.5em;}

.menu:after {
content: attr(data-title);
width: 100%;
background: #000;
color: #fff; }

.menu > span > span {
width: 120px;
color: #fff;}

.menu.servicios > span > span {
background: #ffc000;}

.menu.proyectos > span > span {
background: #00a53c;}

.menu.recursos > span > span {
background: #b81800;}

.menu.contacto > span > span {
background: #9600b4;}

/* Lo que pasa con hover */
.menu:hover > span {
height: 100%; }

/* Dando formato a lista */
ul {
list-style-type: none;
margin: 0;
padding: 0;}

li a {
display: inline;
float: left;
text-decoration: none;
font-family: sans;}

```


2.9.9 Transformaciones en CSS: rotar, torcer, escalar o desplazar.

Nos permiten **rotar, torcer, escalar o desplazar** los elementos de nuestra página web. Este tipo de propiedades de CSS3 son muy interesantes para convertir el lenguaje de hojas de estilo en un sistema capaz de realizar todo tipo de **efectos visuales**. Las dos propiedades que nos sirven para definir las transformaciones son **transform** y **transform-origin**.

- **Transform-origin:** La posición de origen que se utilizará para la transformación es por defecto el lado superior izquierdo del elemento.
- **Transform:** La posición de origen para realizar la transformación es el eje central del elemento.

Tipos de transformaciones

Las transformaciones que podemos aplicar son las siguientes:

- **Scale:** modifica el tamaño de los elementos. La función `scale()` se establece con uno o dos valores, que representan la **cantidad de escala que se aplica en cada dirección: `scale(x)` o `scale(x,y)`**. Se define mediante un **valor numérico** de manera que cuando un valor de coordenadas está fuera del rango `[-1, 1]`, el elemento crece a lo largo de esa dimensión. Cuando está dentro del rango el elemento se encoge.

transform: scale(0.5); /* Escala el elemento a la mitad */

- **Translate:** cambia la posición del elemento hacia la izquierda, derecha, arriba o abajo. La función `translate()` se establece con uno o dos valores: **`translate(x)` o `translate(x,y)`**. Los valores `x` e `y` son los vectores de translación en las coordenadas `x` e `y`. Sus valores pueden estar definidos en píxeles, porcentajes,...

transform: translate(10px); /* Traslada el elemento 10px hacia la derecha */

- **Rotate:** gira o rota los elementos en grados: **`rotate(v)`**.

transform: rotate(45deg); /* Rota el elemento 45 grados */

- **Skew:** distorsiona los elementos según el **ángulo en grados**. La función `skew()` se establece con uno o dos valores: **`skew(x)` o `skew(x,y)`**.

transform: skew(45deg); /* Distorsiona el elemento 45 grados en el eje x */

- **Matrix:** mueve o transforma los elementos con precisión de píxel. La función `matrix()` se establece con seis valores numéricos: `matrix(a,b,c,d,x,y)`. Los dos últimos valores representan la translación y los primeros la transformación lineal.

transform: matrix(0.5, 0.1, 0.5, 1, 10, -2);

La propiedad `transform` se usa junto con la propiedad `transition` vista anteriormente para que la transformación pueda tener una transición espaciada en el tiempo:

```
.caja1{
  -webkit-transition: 1s linear;
  transition: 1s linear;
```

```

}
.caja1:hover{
  -webkit-transform: scale(.5);
  transform: scale(.5);
}

```

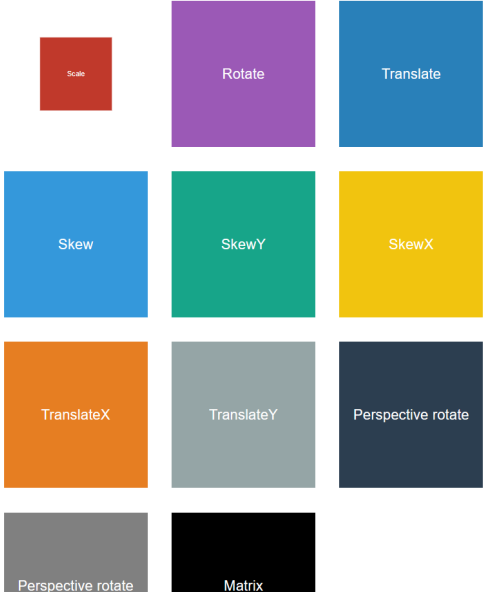
NOTA: No olvides utilizar los prefijos para los navegadores

Ejemplo

Crea un contenedor de 180 píxeles de ancho y de alto, incluye un texto en su interior y aplícale un color de fondo. Realiza las siguientes transformaciones cuando el usuario pase por encima del contenedor el puntero del ratón:

- Modifica la escala del contenedor.
- Modifica la rotación del contenedor.
- Modifica la posición del elemento mediante “translate”.
- Distorsiona el contenedor mediante “skew”.

HTML	CSS
<pre> <div class="container"> <div class="caja caja1">Scale</div> <div class="caja caja2">Rotate</div> <div class="caja caja3">Translate</div> <div class="caja caja4">Skew</div> <div class="caja caja5">SkewY</div> <div class="caja caja6">SkewX</div> <div class="caja caja7">TranslateX</div> <div class="caja caja8">TranslateY</div> <div class="caja caja9">Perspective rotate</div> <div class="caja caja10">Perspective rotate</div> <div class="caja caja11">Matrix</div> </div> </pre>	<pre> .container{ position: relative;} .caja{ width: 180px; height: 180px; color: #FFF; text-align: center; line-height: 180px; margin: 15px; float: left; font-size: 18px; font-family: Arial;} .caja1{ background-color: #C0392B; -webkit-transition: 1s linear; transition: 1s linear;} .caja1:hover{ -webkit-transform: scale(.5); transform: scale(.5);} .caja2{ background-color: #9B59B6; -webkit-transition: 1s linear; transition: 1s linear;} .caja2:hover{ -webkit-transform: rotate(360deg); transform: rotate(360deg);} .caja3{ -webkit-transition: 1s linear; transition: 1s linear; background- color: #2980B9;} .caja3:hover{ -webkit-transform: translate(10%); transform: translate(10%);} .caja4{ background-color: #3498DB; -webkit-transition: 1s linear; transition: 1s linear;} .caja4:hover{ -webkit-transform: skew(10deg); transform: skew(10deg);} .caja5{ background-color: #17A589; -webkit-transition: 1s linear; transition: 1s linear;} .caja5:hover{ -webkit-transform: skewY(10deg); transform: skewY(10deg);} .caja6{ background-color: #F1C40F; -webkit-transition: 1s linear; </pre>

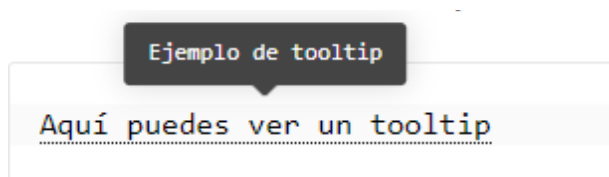
	<pre> transition: 1s linear;} .caja6:hover{ -webkit-transform: skewX(-5deg); transform: skewX(-5deg);} .caja7{ -webkit-transition: 1s linear; transition: 1s linear; background- color: #E67E22;} .caja7:hover{ -webkit-transform: translateX(20px); transform: translateX(20px);} .caja8{ -webkit-transition: 1s linear; transition: 1s linear; background- color: #95A5A6;} .caja8:hover{ -webkit-transform: translateY(20px); transform: translateY(20px);} .caja9{ -webkit-transition: 1s linear; transition: 1s linear; background- color: #2C3E50;} .caja9:hover{ -webkit-transform: perspective(150px) rotateX(45deg); transform: perspective(150px) rotateX(45deg); } .caja10{ -webkit-transition: 1s linear; transition: 1s linear; background- color: grey;} .caja10:hover{ -webkit-transform: perspective(150px) rotateY(45deg); transform: perspective(150px) rotateY(45deg);} .caja11{ -webkit-transition: 1s linear; transition: 1s linear; background- color: black;} .caja11:hover{ -webkit-transform: matrix(0.5, 0.1, 0.5,1,10, -2); transform: matrix(0.5, 0.1, 0.5, 1, 10, -2);} </pre>
--	---

2.9.10 Tooltip en CSS: Pasos para crear tooltips.

Un *tooltip* es una **pequeña ventana emergente** que se muestra cuando el usuario coloca el cursor sobre un elemento. Proporciona información adicional o aclaratoria sobre el elemento al que está asociado, sin ocupar mucho espacio en la página. El tooltip suele aparecer cerca del elemento objetivo y desaparece cuando el cursor se mueve fuera de él.

El **contenido de un tooltip** puede ser texto, imágenes u otros elementos HTML. Se utiliza comúnmente para proporcionar descripciones breves, etiquetas o indicaciones sobre la función o el propósito de un elemento en particular, como enlaces, botones o imágenes. **Los tooltips son una forma efectiva de mejorar la usabilidad y la experiencia del usuario** al proporcionar **información adicional de manera contextual y discreta**.

Ejemplo:



Pasos para crear tooltips

1. Crea la estructura HTML para el elemento que tendrá el *tooltip*. Por ejemplo, puedes usar un elemento `` dentro de un elemento con una clase específica.

```
<div class="tooltip-container">
  <span class="tooltip-text">Este es un tooltip</span>
  Mi elemento
</div>
```

2. Estiliza el contenedor del *tooltip* y el texto del *tooltip* en tu archivo CSS. Puedes ajustar los estilos según tus preferencias.

```
.tooltip-container {
position: relative;
display: inline-block;
}
.tooltip-text {
visibility: hidden;
width: 120px;
background-color: #000;
color: #fff;
text-align: center;
border-radius: 6px;
padding: 5px;
position: absolute;
z-index: 1;
bottom: 125%; /* Posiciona el tooltip encima del elemento */
left: 50%;
transform: translateX(-50%);
opacity: 0;
transition: opacity 0.3s;
}
```

3. Agrega una regla CSS para mostrar el *tooltip* cuando se pasa el cursor sobre el elemento. Puedes utilizar el selector `:hover` para lograrlo.

```
.tooltip-container:hover .tooltip-text {
visibility: visible;
opacity: 1;
}
```

Con estos pasos, has creado un *tooltip* básico con CSS. Puedes personalizar los estilos y añadir efectos adicionales según tus necesidades. Además, también puedes utilizar JavaScript para crear *tooltips* más avanzados con funcionalidades adicionales.

Ejemplo

HTML	CSS
<pre><div class="tooltip-container"> Mensaje del tooltip Este elemento tiene un tooltip </div></pre>	<pre>.tooltip-container { position: relative;</pre>

<p>Este elemento tiene un tooltip</p> <div data-bbox="142 228 438 331">Mensaje del tooltip</div>	<pre>display: inline-block;} .tooltip-text { visibility: hidden; width: 120px; background-color: #000; color: #fff; text-align: center; border-radius: 6px; padding: 5px; position: absolute; z-index: 1; top: 125%; /* Posiciona el tooltip bajo del elemento */ left: 50%; transform: translateX(-50%); opacity: 0; transition: opacity 0.3s;} .tooltip-container:hover .tooltip-text { visibility: visible; opacity: 1;}</pre>
--	---

Crear un tooltip en diferentes posiciones

Para crear un tooltip que se muestre en diferentes posiciones, como encima, a la derecha, abajo y a la izquierda, puedes ajustar las propiedades de posicionamiento en CSS. Veamos los pasos a seguir:

1. Asegúrate de tener la estructura HTML básica del *tooltip*, como se explicó anteriormente.
2. Agrega las siguientes reglas CSS para cada posición del *tooltip*:

- Para el **tooltip encima del elemento**:

```
.tooltip-text {
  bottom: 100%; /* Cambia el valor según sea necesario */
  left: 50%;
  transform: translateX(-50%);
}
```

- Para el **tooltip a la derecha** del elemento:

```
.tooltip-text {
  top: 50%;
  left: 100%; /* Cambia el valor según sea necesario */
  transform: translateY(-50%);
}
```

- Para el **tooltip abajo** del elemento:

```
.tooltip-text {
  top: 100%; /* Cambia el valor según sea necesario */
  left: 50%;
  transform: translateX(-50%);
}
```

- Para el **tooltip a la izquierda** del elemento:

```
.tooltip-text {
top: 50%;
right: 100%; /* Cambia el valor según sea necesario */
transform: translateY(-50%);
}
```

3. Asegúrate de ajustar las propiedades `width`, `background-color`, `color`, `border-radius`, `padding` y cualquier otro estilo que desees para que coincidan con el diseño de tu *tooltip*.

Con estos ajustes, el tooltip se posicionará correctamente en la dirección deseada según las reglas CSS proporcionadas.

2.9.11 Propiedades personalizadas – Variables CSS.

Nos permiten definir **nombres para valores determinados de las propiedades**. De esta forma, se puede utilizar ese nombre **como si fuese una variable**. La gran ventaja es que al modificar el valor de esas «variables» se cambia automáticamente en todos los lugares del documento CSS en los que se haya usado ese nombre.

Esta característica es utilizada recientemente ya que no estaba presente en las primeras versiones de CSS ni soportada en todos los navegadores en las siguientes versiones.

Para definir una **propiedad personalizada** pondremos dos guiones - - delante del nombre que queramos utilizar. Para definir la propiedad personalizada para todo el documento HTML la incluiremos en el **pseudoelemento :root**.

Ejemplo:

```
:root {
--principal-color: black; /* Valor personalizado */
--font: "Arial";
}
```

Para utilizar una **propiedad personalizada** hay que insertar su nombre dentro de la expresión `var ()`:

```
.clase {
background-color: var(--principal-color);
font-family: var(--font);
}
```

Ejemplo

HTML	CSS
<pre><h1>CSS Propiedades Personalizadas</h1> <p>Este texto se ve de color marrón</p> <div>Este fondo se ve verde azul y el texto en Arial</div></pre>	<pre>:root { --principal-color: brown; --secondary-color: #e1f4fc; --font: "Arial"; } p{ color: var(--principal-color);} div { background-color: var(--secondary-color); font-family: var(--font);}</pre>

**Este texto no tiene ningún
estilo definido**

CSS Propiedades personalizadas

Este texto se ve de color marrón

Este fondo se ve verde azul y el texto en Arial

Este texto no tiene ningún estilo definido