CSS

Índice

CSS

CSS es un lenguaje de hojas de estilo que nos permite darle estilo a nuestros documentos HTML. CSS nos permite cambiar el color de los textos, el tamaño de las fuentes, el color de fondo, la posición de los elementos, etc. Se llama CSS por "Cascading Style Sheets" (hojas de estilo en cascada), ya que los estilos se aplican de arriba

hacia abajo, y si hay dos estilos que se intentan aplciar al mismo elemento, el que predomina es el que está más abajo.

Criterios para el estilo que se aplica

Si alguna regla tiene una especificidad mayor, se aplica ese estilo. Que una regla tenga una especificidad mayor significa que tiene un selector más específico, por ejemplo un selector de ID tiene más especificidad que un selector de clase y un selector de clase tiene más especificidad que un selector de elemento.
De no aplicar ninguna regla adicional, el estilo que se aplica es el que está más abajo en el documento (o documentos) CSS. El orden en que se importan los documentos CSS también es importante, ya que si se importa un documento CSS después de otro, el estilo que se aplica es el del documento CSS que se importó después.
`!important` es una palabra clave que se puede usar para forzar que se aplique un estilo, aunque tenga una especificidad menor que otro estilo o esté más arriba. Se usa así: `color: red !important;`. **Se recomienda no usar esta palabra clave**, ya que puede causar problemas de mantenimiento en el código.

Estas reglas no implican que un estilo descarta al otro completamente, es decir, las reglas que se pisan son las que sobreescriben las reglas con más especificidad, pero no se eliminan. Por ejemplo, si tenemos el siguiente código:

```
p {
    color: red;
}

.intro {
    color: blue;
}
```

Y en el HTML:

```
<p class="intro">Hola</p>
<p>Mundo</p>
```

Hola

Mundo

En este caso la regla `color` del selector de clase `.intro` sobreescribe la regla `color` del selector de elemento `h1`, pero no se elimina. Por lo tanto, el color del texto es azul en el párrafo con la clase `intro`, pero rojo en el párrafo sin clase.

Otro ejemplo:

```
p {
    color: red;
    font-size: 30px;
}

.intro {
    color: blue;
}
```

Y en el HTML:

```
<p class="intro">Hola</p>
<p>Mundo</p>
```

Hola

Mundo

En este caso la regla `color` del selector de clase `.intro` sobreescribe la regla `color` del selector de elemento `h1`. Por lo tanto, el color del texto es azul en el párrafo con la clase `intro`, pero rojo en el párrafo sin

clase. Sin embargo, la regla `font-size` del selector de elemento `h1` no se sobreescribe, por lo que el tamaño de la fuente es de 30px en ambos párrafos.

Tipos de selectores

Selectores de elemento

Estos selectores se utilizan para seleccionar elementos por su nombre de etiqueta. No utilizan ningún tipo de prefijo y se aplican a todos los elementos que coincidan con el nombre de la etiqueta. Por ejemplo, si queremos cambiar el color de todos los elementos `h1` en nuestra página, podemos usar el siguiente código CSS:

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

Selectores de clase

Los selectores de clase se utilizan para seleccionar elementos por su atributo `class`. Para usar un selector de clase, debemos anteponer un punto (`.`) al nombre de la clase. Por ejemplo, si queremos cambiar el color de todos los elementos que tengan la clase `intro`, podemos usar el siguiente código CSS:

```
.intro {
    color: red;
}
```

Y en el HTML:

```
<p class="intro">Lorem ipsum dolor sit amet.</p>
<h3 class="intro">Lorem ipsum dolor sit amet.</h3>
<p>Lorem ipsum dolor sit amet.</p>
```

El resultado sería:

Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet.

Selectores de ID

Los selectores de ID se utilizan para seleccionar elementos por su atributo `id`. Para usar un selector de ID, debemos anteponer un numeral (`#`) al nombre del ID. Notar que como los ID's deben ser únicos, estas reglas de estilos solo afectarán a un elemento. Por ejemplo, si queremos cambiar el color de un elemento que tenga el ID `intro`, podemos usar el siguiente código CSS:

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

Y en el HTML:

```
<p id="intro">Lorem ipsum dolor sit amet.</p>
<h3>Lorem ipsum dolor sit amet.</h3>
<p>Lorem ipsum dolor sit amet.</p>
```

El resultado sería:

Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet.

Lorem ipsum dolor sit amet.

Selectores de atributo

Los selectores de atributo se utilizan para seleccionar elementos que tengan un atributo con un valor específico. Para usar un selector de atributo, debemos especificar el nombre del atributo entre corchetes (`[]`). Si queremos seleccionar los inputs que tengan un atributo `type` con el valor `number`, podemos usar el siguiente código CSS:

```css
input[type="number"] {
    border: 3px solid red;
}
```

Y en el HTML:

```html
<input type="text" placeholder="Text" />
<input type="number" placeholder="Number" />
<input type="text" placeholder="Text" />
```

El resultado sería:

Text Number Text

Selectores de pseudo-clase

Los selectores de pseudo-clase se utilizan para seleccionar elementos basados en su estado. Por ejemplo, si queremos cambiar el color de un elemento cuando el usuario pasa el mouse por encima (*hover*), podemos usar el siguiente código CSS:

```css
a:hover {
    color: red;
}
```

Y en el HTML:

```html
<a href="#">Link 1</a>
<a href="#">Link 2</a>
<a href="#">Link 3</a>
```

El resultado sería:

Link 1 Link 2 Link 3

Tipos de unidades

Las unidades de CSS se utilizan para especificar el tamaño de las propiedades CSS. Existen diferentes tipos de unidades, pero las más comunes son:

- `px`: Un pixel.
- `em`: Un múltiplo del tamaño de la fuente del elemento.
- `rem`: Un múltiplo del tamaño de la fuente del elemento raíz.
- `%`: Un porcentaje del tamaño del elemento padre.
- `vh`: Un porcentaje del tamaño de la ventana del navegador.
- `vw`: Un porcentaje del ancho de la ventana del navegador.

Disposición de elementos

Elementos de bloque (display: block)

Los elementos de bloque se utilizan para agrupar elementos en bloques verticales. Por ejemplo, los elementos `div` y `p` son elementos de bloque por defecto. Los elementos de bloque siempre comienzan en una nueva línea y ocupan todo el ancho disponible. Es posible cambiar el comportamiento de cualquier elemento a un elemento de bloque usando la propiedad `display: block`.

Por ejemplo:

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

Lorem ipsum dolor sit amet, consectetur adipisicing elit.

Elementos de línea (display: inline)

Los elementos de línea se utilizan para agrupar elementos en líneas horizontales. Por ejemplo, los elementos `span` y `a` son elementos de línea por defecto. Los elementos de línea no comienzan en una nueva línea y solo ocupan el espacio necesario para mostrar su contenido. Es posible cambiar el comportamiento de cualquier elemento a un elemento de línea usando la propiedad `display: inline`.

Por ejemplo:

Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet

Elementos de línea en bloque (display: inline-block)

Los elementos de línea en bloque se utilizan para agrupar elementos en bloques verticales. Por ejemplo, los elementos `button` y `input` son elementos de línea en bloque por defecto. Los elementos de línea en bloque no comienzan en una nueva línea y solo ocupan el espacio necesario para mostrar su contenido. Es posible cambiar el comportamiento de cualquier elemento a un elemento de línea en bloque usando la propiedad `display: inline-block`.

Por ejemplo:

Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet
Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet Lorem ipsum dolor sit amet
Lorem ipsum dolor sit amet

Flexbox (display: flex)

El modelo flexbox es un modelo de diseño unidimensional que define la relación entre los elementos de un contenedor y sus elementos hijos. Es importante esta última parte, ya que será necesario tener en cuenta que cuando definimos un elemento como flexbox, son sus elementos hijos los que se comportarán de forma diferente y no dicho elemento. Para definir un elemento como flexbox, debemos usar la propiedad `display: flex`.

Este modelo de diseño se basa en dos ejes: el eje principal y el eje transversal. El eje principal se define con la propiedad `flex-direction` siendo el valor por defecto `row` (horizontal). Los elementos hijos se distribuyen en el eje principal de izquierda a derecha y se alinearán en dicho eje dependiendo de la propiedad `justify-content`. Para alinearlos en el eje transversal, debemos usar la propiedad `align-items`.

Este es un ejemplo de cómo funciona el modelo de diseño flexbox:

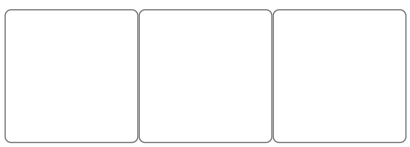
Justify-content

Alinea los elementos hijos en el eje principal. Los valores posibles son:

Flex-start (justify-content)

Este valor alinea los elementos hijos al principio del eje principal. El primer elemento se alinea al principio del eje principal y el último elemento se alinea al final del eje principal.

```
div {
    justify-content: flex-start;
}
```



Flex-end (justify-content)

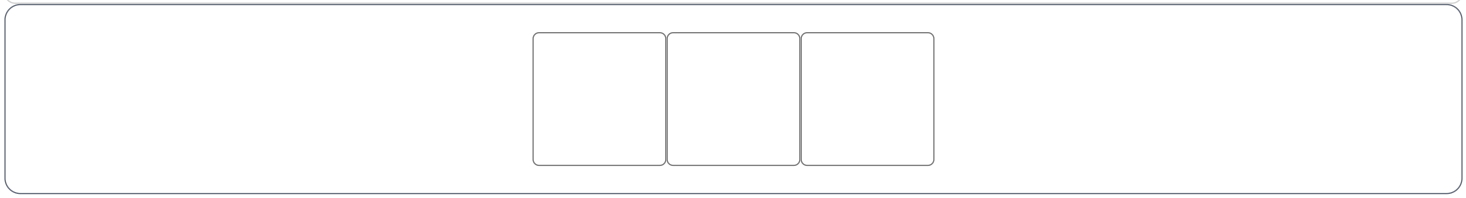
Este valor alinea los elementos hijos al final del eje principal. El primer elemento se alinea al principio del eje principal y el último elemento se alinea al final del eje principal.

```
div {
    justify-content: flex-end;
}
```

Center (justify-content)

Este valor alinea los elementos hijos al centro del eje principal. El primer elemento se alinea al principio del eje principal y el último elemento se alinea al final del eje principal.

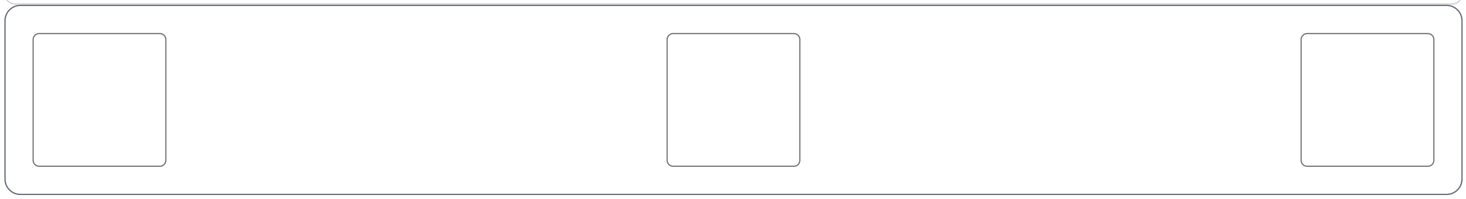
```
div {
    justify-content: center;
}
```



Space-between (justify-content)

Este valor distribuye los elementos hijos de forma equitativa en el eje principal, dejando el máximo espacio posible entre ellos. El primer elemento se alinea al principio del eje principal y el último elemento se alinea al final del eje principal sin dejar ningún espacio con el borde del contenedor.

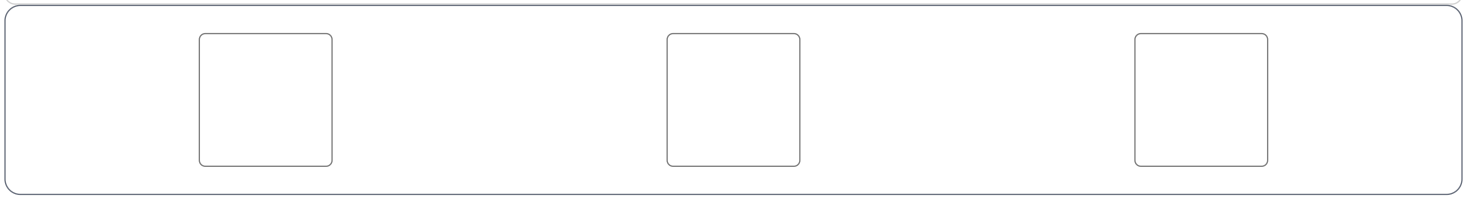
```
div {
    justify-content: space-between;
}
```



Space-around (justify-content)

Este valor distribuye los elementos hijos de forma equitativa en el eje principal, dejando el máximo espacio entre ellos y agregando medio espacio entre ellos y los bordes. El primer elemento se alinea al principio del eje principal y el último elemento se alinea al final del eje principal dejando un espacio con el borde del contenedor.

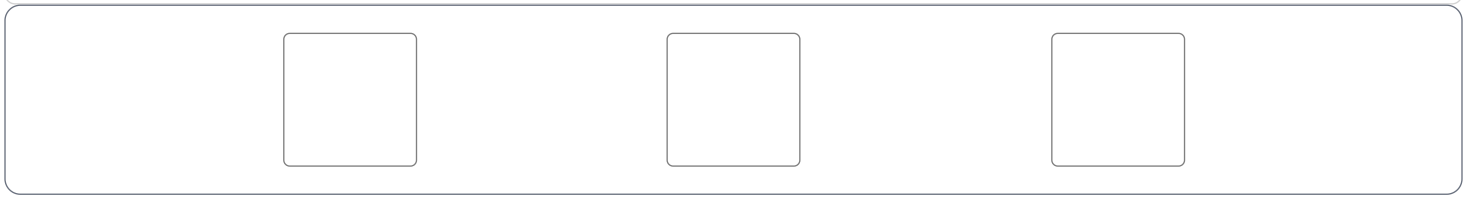
```
div {
    justify-content: space-around;
}
```



Space-evenly (justify-content)

Este valor distribuye los elementos hijos de forma equitativa en el eje principal, dejando el máximo espacio entre ellos y agregando el mismo espacio entre ellos y los bordes. El primer elemento se alinea al principio del eje principal y el último elemento se alinea al final del eje principal dejando un espacio con el borde del contenedor.

```
div {
    justify-content: space-evenly;
}
```



Stretch (justify-content)

Este valor estira los elementos hijos para que ocupen todo el espacio disponible en el eje principal. El primer elemento se alinea al principio del eje principal y el último elemento se alinea al final del eje principal.

```
div {
    justify-content: stretch;
}
```

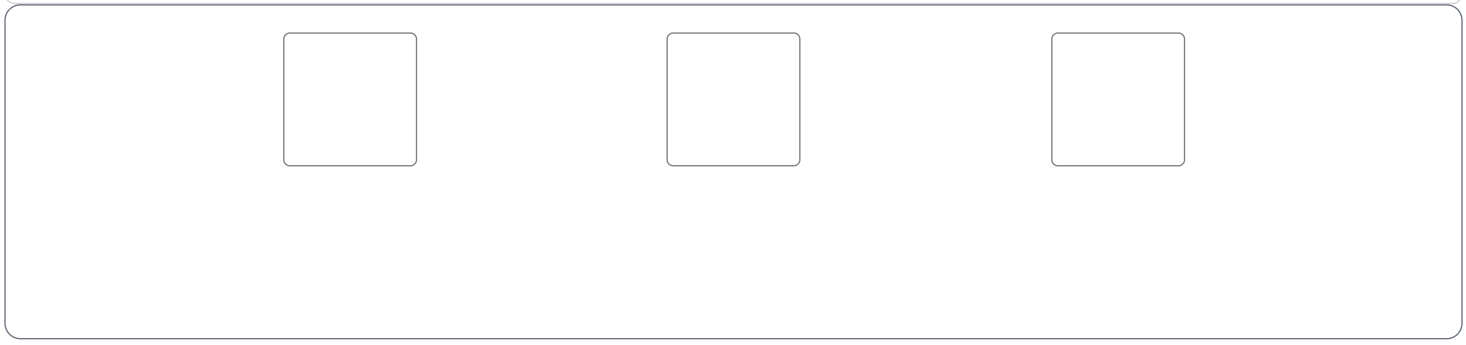
Align-items

Alinea los elementos hijos en el eje transversal. Los valores posibles son:

Flex-start (align-items)

Este valor alinea los elementos hijos al principio del eje transversal. El primer elemento se alinea al principio del eje transversal y el último elemento se alinea al final del eje transversal.

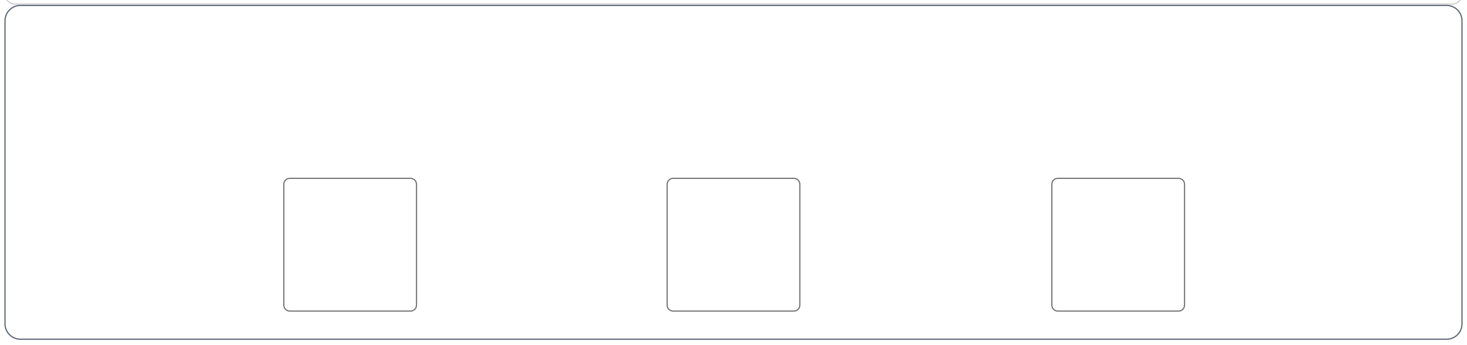
```
div {
    align-items: flex-start;
}
```



Flex-end (align-items)

Este valor alinea los elementos hijos al final del eje transversal. El primer elemento se alinea al principio del eje transversal y el último elemento se alinea al final del eje transversal.

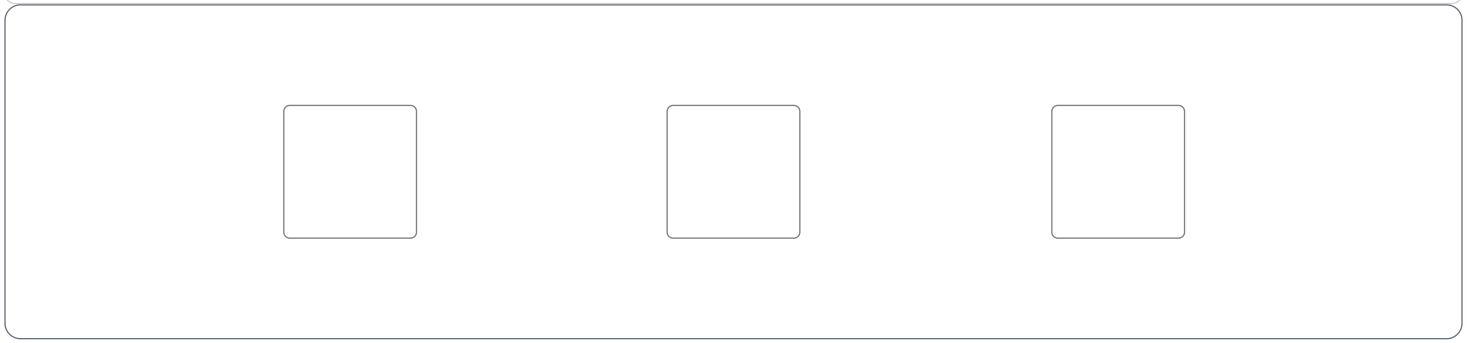
```
div {
    align-items: flex-end;
}
```



Center (align-items)

Este valor alinea los elementos hijos al centro del eje transversal. El primer elemento se alinea al principio del eje transversal y el último elemento se alinea al final del eje transversal.

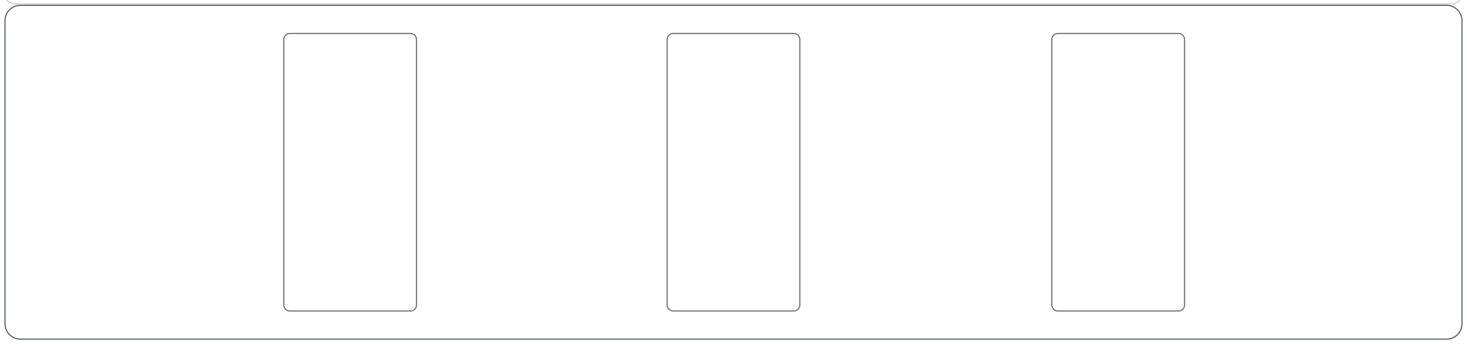
```
div {
    align-items: center;
}
```



Stretch (align-items)

Este valor estira los elementos hijos para que ocupen todo el espacio disponible en el eje transversal. El primer elemento se alinea al principio del eje transversal y el último elemento se alinea al final del eje transversal.

```
div {
    align-items: stretch;
}
```



Flex-direction

Establece la dirección del eje principal. Los valores posibles son:

Row (flex-direction)

Este valor establece el eje principal horizontalmente, de izquierda a derecha.

```
div {
    flex-direction: row;
}
```

Row-reverse (flex-direction)

Este valor establece el eje principal horizontalmente, de derecha a izquierda.

```
div {
    flex-direction: row-reverse;
}
```

Column (flex-direction)

Este valor establece el eje principal verticalmente, de arriba a abajo.

```
div {
    flex-direction: column;
}
```

Column-reverse (flex-direction)

Este valor establece el eje principal verticalmente, de abajo a arriba.

```
div {
    flex-direction: column-reverse;
}
```

Flex-wrap

Establece si los elementos hijos deben ajustarse a una sola línea o pueden ajustarse a múltiples líneas. Los valores posibles son:

No-wrap (flex-wrap)

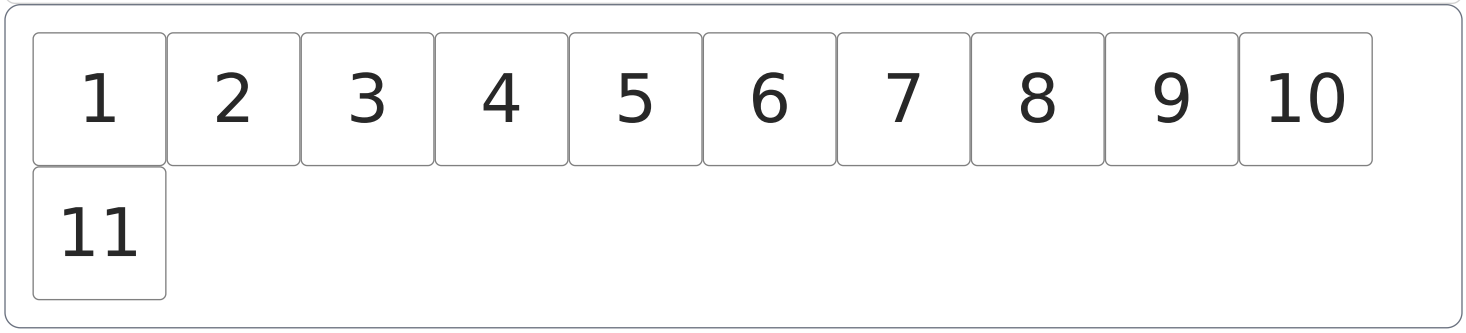
Este valor establece que los elementos hijos no deben ajustarse a múltiples líneas.

```
div {
    flex-wrap: nowrap;
}
```

Wrap (flex-wrap)

Este valor establece que los elementos hijos deben ajustarse a múltiples líneas.

```
div {
    flex-wrap: wrap;
}
```



Wrap-reverse (flex-wrap)

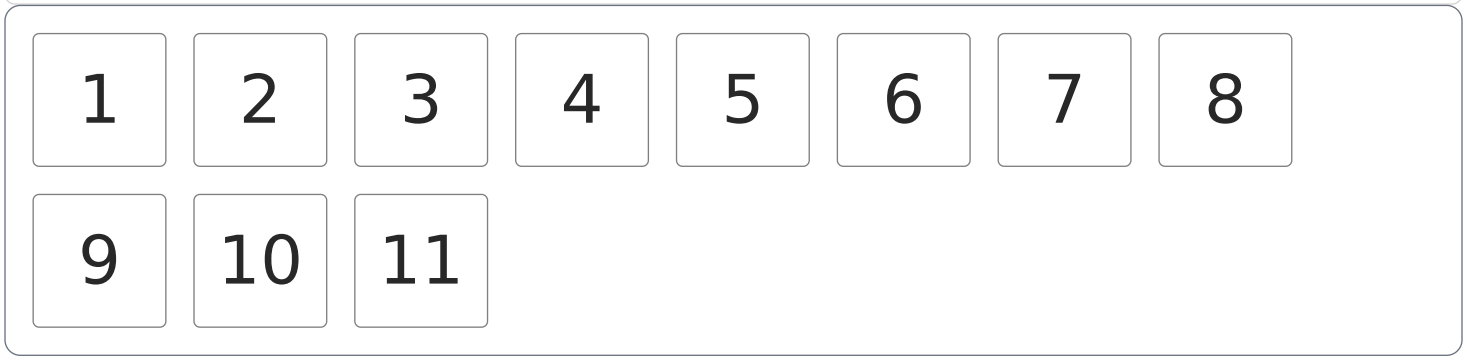
Este valor establece que los elementos hijos deben ajustarse a múltiples líneas, pero en orden inverso.

```
div {
    flex-wrap: wrap-reverse;
}
```

Gap

Establece el espacio entre los elementos hijos. Este espacio se distribuye uniformemente entre los elementos hijos.

```
div {
    gap: 20px;
}
```



Grid (display: grid)

Información sobre el módulo Grid: Grid

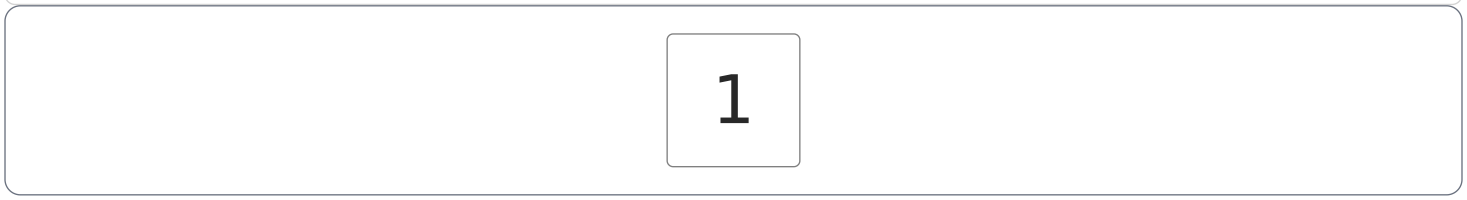
Position

Esta propiedad establece el tipo de posicionamiento de un elemento. Los valores posibles son:

Static

Este valor establece que el elemento no se posiciona de ninguna manera. Es el valor por defecto.

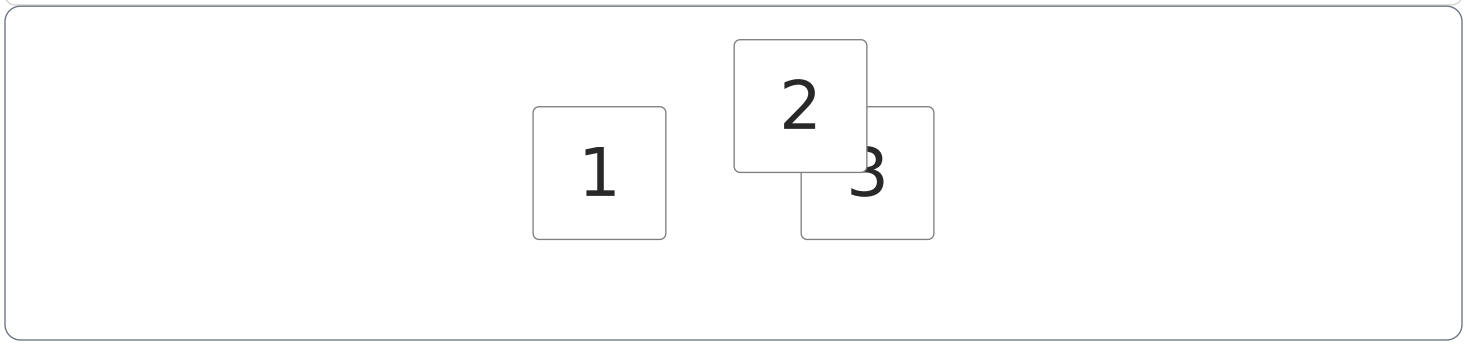
```
div {
    position: static;
}
```



Relative

Este valor establece que el elemento se posiciona de forma relativa a su posición normal. Es decir, se desplaza respecto a la posición que tendría si no se aplicara ningún tipo de posicionamiento. Su desplazamiento se realiza mediante las propiedades top, right, bottom y left, y no afecta a los elementos que se encuentren a su alrededor. Por ejemplo:

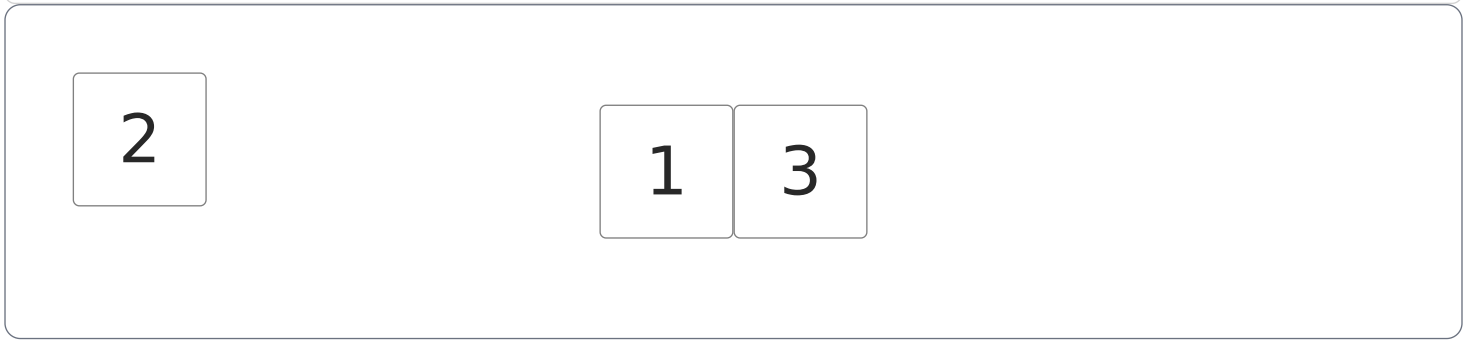
```
div {
    position: relative;
}
```



Absolute

Este valor establece que el elemento se posiciona de forma absoluta respecto a su elemento padre más cercano que tenga un posicionamiento distinto de static. Si no existe ningún elemento padre con un posicionamiento distinto de static, el elemento se posiciona respecto a la ventana del navegador. Su desplazamiento se realiza mediante las propiedades top, right, bottom y left, y no afecta a los elementos que se encuentren a su alrededor de ninguna manera ya que virtualmente desaparece de la página a la hora de desplazar el resto de elementos. Por ejemplo:

```
div {
    position: absolute;
}
```

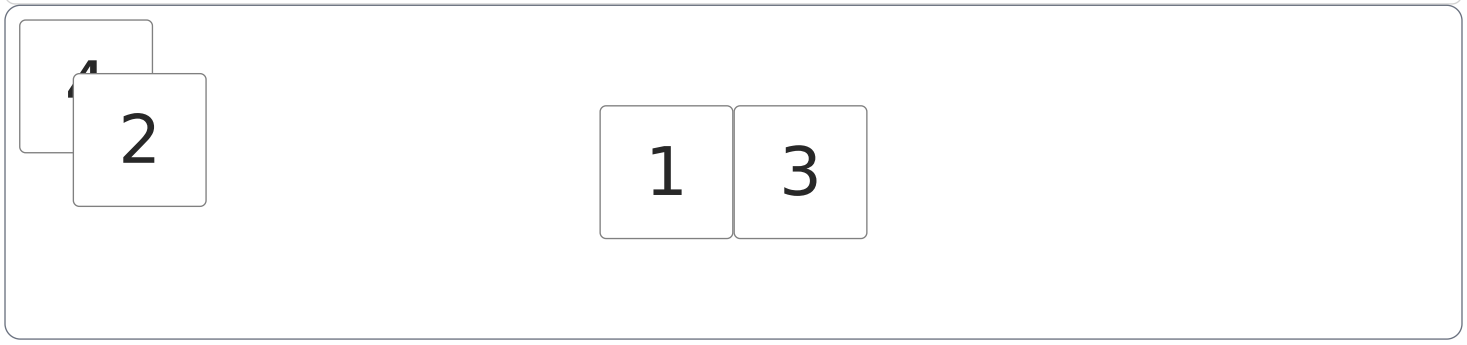


En este ejemplo el elemento "2" se posiciona de forma absoluta respecto al elemento padre "result-15" que tiene un posicionamiento relativo.

z-index

Esta propiedad establece el orden de los elementos que se encuentran en una misma posición. El valor por defecto es 0, y cuanto mayor sea el valor, más adelante se mostrará el elemento. Solo se aplicará a los elementos que tengan un posicionamiento distinto de static. Por ejemplo:

```
div {
    position: absolute;
    z-index: 1;
}
```



En este ejemplo el elemento "4" se posiciona de forma absoluta respecto al elemento padre "result-16" que tiene un posicionamiento relativo, y tiene un z-index de 2, por lo que se mostrará por encima del elemento "2" que tiene un z-index de 3.

Fixed

Este valor establece que el elemento se posiciona de forma absoluta respecto a la ventana del navegador. Su desplazamiento se realiza mediante las propiedades top, right, bottom y left, y no afecta a los elementos que se encuentren a su alrededor de ninguna manera ya que virtualmente desaparece de la página a la hora de desplazar el resto de elementos. A diferencia del valor absolute, este valor se desplaza cuando se hace scroll en la página junto con ella. Por ejemplo el header de la página en la que estás leyendo este cheatsheet.

Box model

El box model es el modelo de caja que se utiliza para representar los elementos HTML en la pantalla. Cada elemento HTML se representa como una caja rectangular que contiene tanto el contenido como el relleno, el borde y el margen. El box model se compone de los siguientes elementos:

- Content: es el contenido del elemento HTML.
- Padding: es el espacio que hay entre el contenido y el borde.
- Border: es el borde que rodea al contenido y al padding.
- Margin: es el espacio que hay entre el borde y los elementos vecinos.

Diagrama del box model:


Margin
Border
Padding
Content