DIFERENCIA ENTRE BORDER Y OUTLINE

Tanto outline como border en CSS se utilizan para definir los bordes de un elemento, pero tienen diferencias importantes:

* Border (borde): Es una línea sólida que rodea el contenido del elemento y se encuentra dentro del área total del elemento. Es decir, el ancho del borde se suma al tamaño total del elemento. Puedes definir el color, grosor y estilo del borde, como punteado, sólido o con bordes redondeados.
* Outline (contorno): También es una línea que rodea el contenido del elemento, pero se dibuja justo fuera del borde del elemento, es decir, no afecta al tamaño total del elemento. El contorno generalmente se muestra como una línea discontinua (un borde punteado) por defecto, y se utiliza para resaltar el elemento sin afectar su tamaño.

En resumen, la principal diferencia es que border afecta al tamaño total del elemento, mientras que outline no lo hace y se utiliza principalmente para resaltar el elemento sin cambiar su diseño. Además, outline suele ser más útil en casos como la navegación mediante teclado, donde se necesita un enfoque visual rápido en un elemento activo

MARGENES

-Predomina el margen que tiene más px

-**Cuando dos márgenes se tocan es porque están colapsando, y cuando eso pasa es que va a predominar el margen mayor.**

-Cuando nosotros ponemos una etiqueta dentro de otra están literalmente una dentro de la otra, no limitadas, lo que ocurre es un colpaso de márgenes. Lo que podemos hacer si quisiéramos separar una etiqueta de la otra en el caso del nav del header por ejemplo es poner un pequeño padding dr 0.1px esto genera más relleno y permite que el nav pueda bajar y estar más separado del header, no se pueden separar, están dentro una de otra. Si le sacamos el padding me baja todo el contenido porque están uno dentro del otro. Esto también es colpaso de márgenes

-Toda etiqueta o elemento en bloque que tenga un width definido se puede centrar con margin 0 auto, sino tengo el width definido no se puede centrar

El margen que predomina es el mayor, cuando tenemos dos elementos y su márgenes o bordes se están tocando también lo afecta al borde mayor y baja la etiqueta padre con las que tenga dentro. En ese caso le tenemos que dar a la etiqueta padre un padding para que pueda intervenir.

ANTES QUE NADA: ¿QUE ES EL FLUJO?

-El flujo de una web es el modo en el que se van disponiendo, de forma natural, los elementos en nuestra página

POSTIONS

-Cosas a tener en cuenta:

-Flujo renderizado: Por norma general los elementos se dibujan de izquierda a derecha y de arriba abajo. El punto 0,0 de los elementos, por norma general, es la esquina superior izquierda.

-Espacio reservado: Es el espacio que tiene un elemento asignado en el navegador.

-Elemento posicionado: Esto significa que el elemento tiene la propiedad position con un valor distinto a “static”, que es el valor que tiene esta propiedad por defecto.

-Stackin context: Contexto de apilamiento. Es el orden en el que se apilarán las cajas que se superponen, dentro del mismo contenedor.

Orden que se sigue:

-Tiene el orden de adelante a atrás

-Lo primero que se va a ver siempre son los elementos posicionados con z-index de 1 o más

-Elementos posicionados sin z-index declarado (o z-index: auto;)

-Elementos no posicionados.

-Elementos con z.index negativo.

Al posicionar un elemento se habilitan 5 propiedades que podemos utilizar para mover los elementos en los 3 ejes.

Top: El elemento se moverá desde la parte superior la distancia que le hayamos indicado.

Rigth: El elemento se moverá desde la parte derecha la distancia que le hayamos indicado.

Bottom: El elemento se moverá desde la parte inferior la distancia que le hayamos indicado.

Left: El elemento se moverá desde la parte izquierda la distancia que le hayamos indicado.

Z-index: Nos permite mover el elemento en el contexto de apilamiento (eje z). Manipula el orden de apilamiento, para usarla es importante que un elemento este posicionado. Modificamos su contexto de apilamiento, el valor por defecto es auto. Los órdenes de apilamiento por defecto es los que tiene HTML.

-**NORMAS DE Z-INDEX:**

-No usar números consecutivos si vamos a usar mas de un x-indez, porque si queremos agregar un elemento que se posicione entre dos elementos con z-indez definidos y estos son consecutivos vamos a tener que modificar todo, tratar de usar números separados de 10 en 10 o de 100 en 100. Siempre se tiene en cuenta ante todo lo que dice el HTML en estos casos

-El contenedor padre jamas se va a colocar por delante del hijo, lo que podemos hacer es poner un elemento hijo detrás de un elemento padre y para eso se usa z-index: -1; **pero el padre no tiene que tener ningún z-index declarado**, solo auto que es como no ponerle nada

**NOTA:** Si a un elemento le indicamos la propiedad top y/o left no tomara en cuenta las propiedades de bottom y right y estas no funcionarán.

Los posibles valores que podemos pasarles son:

-STATIC: Es el valor que tiene por defecto el elemento, con este valor el elemento NO ESTA POSICIONADO y por lo cual no podemos moverlo.

-RELATIVE: El elemento se mantendrá en su posición y medidas en el flujo renderizado y mantendrá su espacio reservado. Si lo movemos lo hara usando su posición en el html como punto de referencia. Acepta valores negativos y positivos.

-ABSOLUTE: El elemento perderá sus medidas y su espacio reservado. Si lo movemos usará el elemento padre contenedor posicionado como referencia. Si no tiene ninguno, usará el elemento HTML como referencia (parte superior izquierda) pero si le pasamos una propiedad ´por ej: bottom: 0; este se va a mover debajo de todo pero siempre moviéndose respecto al elemento HTML, porque si yo le paso una propiedad de right se va a mover pero tomando en cuenta el body o HTML. En cambio si toma como referencia un contenedor padre se va a mover respecto a ese contendor**. TENER EN CUENTA QUE SI UN ELEMENTO NO TIENE NINGUNA POSITION Y OTRO SI COMO EN ESTE CASO, VA A PERDER SU ESPACIO RESERVADO Y EL ELEMENTO QUE NO TENGA POSITION VA A TOMAR ESE ESPACIO RESERVADO. Cuando a un elemento elemento en línea le ponemos position absolute le estamos dando contexto de elemento en bloque para poder ponerle medidas, margenes y que lo tome.**

-FIXED: El elemento perderá sus medidas y su espacio reservado. Si lo movemos usará el elemento HTML de referencia, y además se quedará fijo en esa posición aunque hagamos scroll.

-STICKY: Es una mezcla de position relative y fixed. Con este tipo de posicionamiento los valores top, left, bottom y rigth no sirven para mover el elemento, si no para indicarle en que punto pasará a tener un comportamiento de posicionamiento fixed , hasta llegar a ese punto se comportará como si tuviera relative

**AHORA SI:**

La propiedad position es la que se encarga de decirle a todos los elementos de nuestra pagina web que tipo de referencia tienen que tomar para posicionarse

-Todo elemento creado desde HTML se crea desde el margen superior izquierdo.

-CSS Los que nos permite con las posiciones es: cambiar el flujo con el los elementos que vienen predeterminado

-Las posiciones se definen con la palabra position.

-Las posiciones que existen son:

* Static (valor, flujo por defecto) no tiene relevancia es por defecto
* Relative (El elemento se coloca relativo al flujo normal)

Cuando usamos la propiedad de relative le estaremos diciendo que tome la posición relativamente a su posición natural. Su espacio natural se sigue conservando

Se va moviendo según su posición de origen. Es por eso que los contenedores padres siempre tienen relative porque si tenemos un header con alto y ancho no se va a mover pero si le ponemos relative, le damos la posibilidad a los elementos que tenga dentro de moverse y poder posicionarse como queramos con la propiedad de absolute.

Se mueve desde su posición original y además conserva su flujo

Cuando un elemento tiene la propiedad "position: relative" en CSS, eso significa que su posición será relativa a su ubicación original en el flujo normal del documento, antes de aplicar cualquier cambio de posicionamiento.

Imagina que tienes un documento HTML con varios elementos posicionados de forma regular, uno después del otro en el flujo normal. Si aplicas "position: relative" a uno de esos elementos, este se moverá desde su posición original utilizando las propiedades "top", "right", "bottom" y "left". Sin embargo, a pesar de ese desplazamiento visual, en el flujo normal del documento, los demás elementos seguirán ocupando el espacio que el elemento con "position: relative" ocuparía si no se hubiera movido.

Es como si el elemento con "position: relative" tuviera una "caja invisible" en su posición original que ocupa espacio en el flujo del documento, pero luego se mueve visualmente dentro de esa caja sin afectar a otros elementos.

* + Por más que lo movamos a los elementos siguen manteniendo su espacio en blancos, los espacios que ocupan se conservan, su flujo sigue estando. El espacio donde está el elemento sigue estando.
  + Se mueve desde la posición que le indiquemos (izquierda, derecha.etc) desde su posición inicial, es relativo a su posición de origen, O sea va desde la izquierda por ejemplo y desde su posición inicial. Está en su posición inicial, le indico left y se mueve desde su posición hacia la izquierda y mantiene su flujo
  + Es relativo a su posición original
  + el 0 va a ser el punto inicial, vuelve al punto en el que nació
* Absolute (El elemento se coloca respecto a su contenedor posicionado más cercano)
  + Se va a posicionar siempre al navegador, al body.
  + Cuando aparece el position absolute como primera medida está fuera del flujo. Se posiciona a su ancestro más cercano posicionado, si su ancestro más cercano no está posicionado(es decir que no le dimos una position) se queda en el body o navegador, pero si yo posiciono a su ancestro más cercano(es decir le ponemos una position) absolute se va a posicionar dentro del ancestro siempre obedece a su ancestro mientras este esté posicionado, con un position relative o un position fixed y le podemos poner la dirección con top, bottom, left, right
  + El relative y el absolute siempre van a ir de la mano donde el padre siempre va a ser relative y los hijos van a ser absolute
  + Osea lo que hace es moverse tomando el elemento ancestro como referencia y las direcciones que le indiquemos
  + Si no esta posicionado su ancestro mas cercano se va al body o al navegador.

Cuando se aplica **position: absolute** a un elemento, se eliminará del flujo normal del documento y se colocará en una posición específica según las siguientes reglas:

1. **Elemento contenedor**: El elemento con **position: relative**, **position: absolute**, **position: fixed** o **position: sticky** actuará como el "elemento contenedor" del elemento posicionado. Si no hay un elemento contenedor, se utilizará el elemento raíz del documento (**<html>**).
2. **Coordenadas**: Para posicionar el elemento, se utilizan las propiedades **top**, **right**, **bottom** y **left**. Estas propiedades definen la distancia desde el borde superior, derecho, inferior e izquierdo del elemento contenedor, respectivamente. Puedes establecer estos valores en píxeles, porcentajes o con otras unidades CSS válidas.
3. **Posicionamiento relativo al contenedor**: El elemento posicionado se colocará en función de las coordenadas definidas en relación con su elemento contenedor más cercano. Por ejemplo, si estableces **top: 10px** y **left: 20px**, el elemento se ubicará a 10 píxeles desde la parte superior y 20 píxeles desde la izquierda del contenedor.
4. **Desplazamiento**: Los elementos posicionados no afectan el diseño de otros elementos en el flujo del documento. Otros elementos se comportarán como si el elemento posicionado no estuviera presente, lo que puede resultar en superposiciones o espacios vacíos.

Es importante tener en cuenta que cuando se usa **position: absolute**, el elemento no ocupará espacio en el flujo normal de la página, lo que significa que otros elementos se comportarán como si el elemento posicionado no estuviera presente. Esto puede causar que otros elementos se superpongan con el elemento posicionado. Para evitar superposiciones no deseadas, es posible que necesites ajustar la posición de los elementos utilizando las propiedades de posición y las coordenadas adecuadas.

* Fixed (El elemento se coloca respecto al viewport)
* Es relativo al navegador, independientemente del ancestro que tenga

-Se salta todo el flujo de la web pero va a tomar como referencia los marcos del navegador lo que conseguimos con esto es que cada vez que hagamos scroll, el elemento no se mueva, podemos hacer un pop up (Pop up o ventana emergente

Se refiere a cualquier ventana que se abre de forma automática, es decir, sin que el usuario interactúe haciendo clic en la interfaz) controlamos las direcciones con top, right, bottom, left.

* + Tienen similitudes con el absolute pero la diferencia es que el absolute depende de que su ancestro este posicionado, el fixed es relativo al navegador con esto podemos hacer botones que no se muevan aunque hagamos scroll
* Sticky

-Es una mezcla entre position relative y fixed. Los valores right, bottom, top, left no sirven para mover el elemento, lo utlizamos como punto de referencia donde queremos que el elemento pase a tener un comportamiento similar al que tiene fixed. Es decir que el elemento quede estático aunque hagamos scroll

La propiedad **position: sticky** en CSS es otro valor de la propiedad de posicionamiento que combina características de **position: relative** y **position: fixed**. Un elemento con **position: sticky** se comporta como un elemento con posicionamiento relativo (**position: relative**) hasta que se desplaza fuera del área visible del navegador, momento en el cual se fijará en su posición y se comportará como un elemento con posicionamiento fijo (**position: fixed**).

La magia de **position: sticky** radica en que el elemento permanecerá "pegado" en su posición cuando el usuario haga scroll y el elemento llegue al límite superior de su contenedor (el límite inferior si has establecido **position: sticky** en la parte inferior). Una vez que se desplaza fuera de la pantalla en esa dirección, el elemento dejará de ser "sticky" y volverá a comportarse como un elemento con posicionamiento relativo hasta que alcance nuevamente el límite.

-Cuando nosotros le agregamos a sticky los valores de top, bottom, right,left lo que le decimos que cuando el header por ejemplo este a 0px del top por ej, el elemento se empiece a comportar de manera similar a un elemento fixed.

-Cuando usamos sticky lo que hacemos es recorrer el alto del elemento que lo contiene (que es el elemento padre) por eso:

**Sticky lo que toma como referencia es el alto del elemento padre, si no le ponemos alto al elemento que lo contiene por mas que hagamos scroll no va a pasar nada aunque este la propiedad de sticky. Porque con sticky**  **lo que hacemos es recorrer el alto del elemento que lo contiene.**

-Por eso mantiene pegado el elemento en la dirección que le hemos puesto (top, right, etc) mientras hagamos scroll durante los px que coloquemos.

Ejemplo: hago scroll durante 300px y mi nav con posición top y 300px se va a mostrar arriba hasta que recorra esos 300px algo asi seria mas o menos

En el caso de que tenga contenido como un p por ejemplo la propiedad va a seguir estando mientras tenga contenido porque esa es la dimensión (alto, ancho) que le damos

* Index z

-El eje z se mueve superponiéndose, solo funciona con los elementos posicionados sino lo están no va a funcionar. Esto nos puede servir para hacer hero en el que queremos que el texto quede delante de nuestra imagen o video por ejemplo

* + Index z: auto;

-Esas posiciones tienen valores o propiedades que se activan que son:

* top (de arriba hacia abajo)
* right
* left
* right
* bottom(de abajo hacia arriba)
* index z

-Esto va a deifnir en que dirección me va a mover el elemento

-display none desparece al elemento del flujo

EFECTOS PARA NUESTRO HERO

-Primero podemos hacer scroll sin br y es de esta manera

En HTML:

<section class="element"></section><!-- *aca hacemos un scroll* -->

En CSS:

.element{

    height: 1200px;

}/\* *aca hacemos un scroll* \*/

Y luego podemos darle un estilo a nuestro hero para que tenga formas con las siguientes paginas:

<https://smooth.ie/blogs/news/svg-wavey-transitions-between-sections>

1. En esta pagina nosotros hacemos la forma que queramos y luego copiamos el documento HTML en nuestro index.html en el header
2. En donde dice HTML OUTPUT
3. **Esto lo hacemos antes del scroll**
4. Cuando hayamos pegado el código en HTML y vayamos al navegador nos va a aparecer encima de nuestro hero entonces para modificarlo necesitamos las positions
5. Vemos si tiene un clase y si no se la agregamos para poder llamarla desde CSS
6. Luego de ponerle la clase lo llamamos en CSS y le vamos a poner una position
7. Vamos a ir al elemento padre de nuestra clase que acabamos de crear en este caso header y le vamos a poner la position relative
8. Luego en nuestra clase le vamos a poner position absolute, la dirección en este caso bottom: 0; y le vamos a dar una dimensión con width y con height7
9. Es muy importante que en nuestro código html el color de la ola sea blanco, lo modificamos poniendo #fff
10. .hero{/\* *es para que tenga una cabecera* \*/
11. width: 100%;
12. height: 600px;
13. background-size: cover;/\* *para que ocupe toda la dimension que puse* \*/
14. background-position: left;
15. background-repeat: no-repeat;
16. background-image: linear-gradient(37deg, rgba(148, 177, 230, 0.068), rgba(219, 107, 223, 0.356)), url(./guardaropas.jpg);
17. background-attachment: fixed;
18. position: relative;
20. }

Aca le damos la position a su ancestro mas cercano

.wave{

    width: 100%;

    height: 20px;

    position: absolute;

    bottom:0;

}

Aca le damos la position a la clase con la dirección y la posicon absolute porque relative es e lpadre y absolute los hijos

La propiedad "bottom: 0" en CSS se utiliza en conjunción con la propiedad "position: absolute" para posicionar un elemento de manera absoluta dentro de su elemento padre que tenga una posición relativa o absoluta.

Cuando se define "position: absolute" en un elemento, este se retira del flujo normal del documento y se posiciona de manera absoluta en relación con su primer ancestro que tenga una posición diferente a "static" (que puede ser "relative", "absolute", "fixed" o "sticky"). Si no hay un ancestro con una posición diferente a "static", el elemento se posicionará en relación con el cuerpo (body) del documento.

Ahora bien, al agregar "bottom: 0", se está estableciendo la distancia del elemento posicionado desde el borde inferior del elemento padre. Es decir, el elemento se pegará al borde inferior del contenedor padre que tenga una posición establecida (distinta de "static").

El uso común de "bottom: 0" con "position: absolute" es para asegurarse de que un elemento esté siempre anclado en la parte inferior de su contenedor, independientemente del tamaño del contenedor o del contenido dentro de él. Esto puede ser útil, por ejemplo, para fijar un elemento en la parte inferior de un contenedor y garantizar que permanezca visible aunque el contenido del contenedor cambie o se desplace.