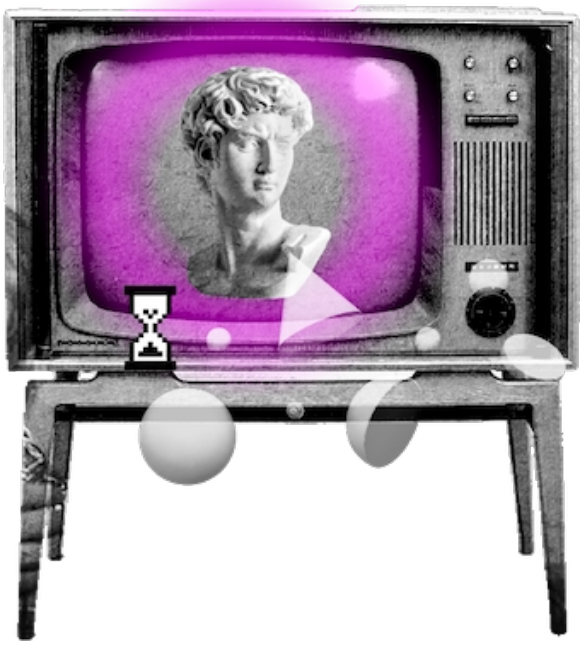
NUCBA

DISEÑO WEB

BOX MODEL + POSITION



APUNTES

CSS II - Box-model + Position

¿Que es el Box Model?

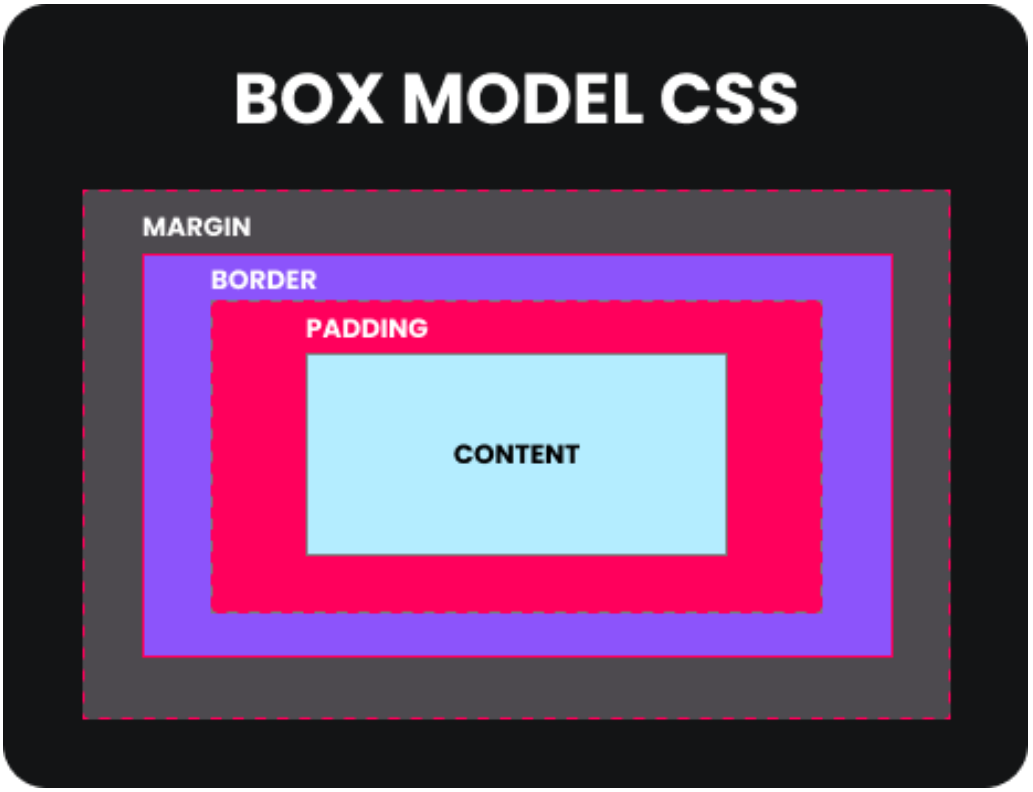
El **box model o modelo de caja** es un concepto clave en CSS que describe cómo se representan los elementos HTML en la pantalla. Cada elemento HTML se representa como una caja, que consta de cuatro partes principales:

Contenido: Este es el área central donde se coloca el contenido del elemento, como el texto o la imagen.

Padding: Este es el área de relleno alrededor del contenido. Se puede especificar un ancho de relleno en CSS para separar el contenido de los bordes o de otros elementos.

Border: Este es el borde que rodea al contenido y al relleno. Se puede especificar un ancho de borde, un color y un estilo en CSS.

Margin: Este es el margen alrededor de la caja que separa el elemento de otros elementos. Se puede especificar un ancho de margen en CSS para crear espacio alrededor del elemento.



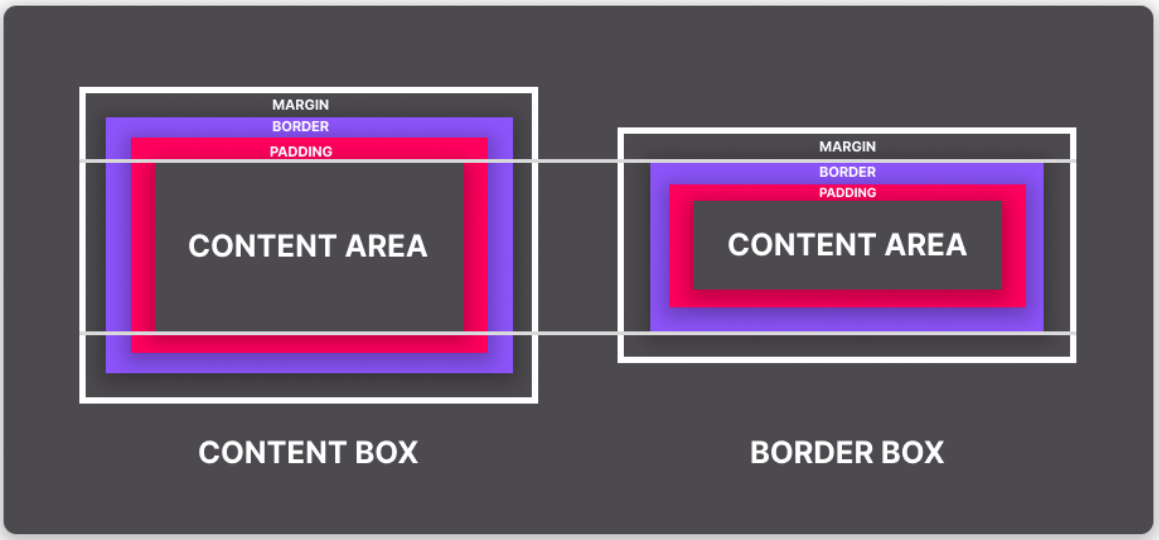
Entender este modelo es clave para poder manipular los elementos de nuestra web.

Box-sizing

Esta propiedad establece cómo se calcula el ancho y alto total de un elemento.
Puede tomar dos valores:

1. **content-box:** Es el valor por defecto de esta propiedad. Hace que se calcule el ancho o alto original del elemento sumando el **padding y el border (separación del borde con el contenido)** que se coloque al elemento.
   Por ejemplo: Si tengo un elemento con **200px** de ancho y de alto, y un padding de **10px** , el ancho y alto de nuestro elemento será de **220px** (**200px** de ancho/alto original más los **10px** de padding en la **derecha e izquierda** o **abajo y arriba**).

2. **border-box:** A diferencia de **content-box** este valor incluirá el **padding y el border** dentro del tamaño que especificamos a nuestro elemento.
   Por ejemplo: Si tengo un elemento con **200px** de ancho y de alto, y un padding de **10px** , el ancho y alto de nuestro elemento será de **200px** (El padding será "absorbido" por el tamaño original del elemento, incluyendo los **10px** de padding en cada costado de nuestra caja en esos **200px** de tamaño).



Padding

Establece el espacio de relleno desde el borde hacia el contenido de un elemento.

```css
div {
  padding: 10px; /*Todos los lados*/
  padding: 10px 20px; /*10px arriba/abajo 20px izq/der*/
  padding: 10px 11px 12px 13px; /*arriba/der/abajo/izq*/
}
```

También podría especificarse el de un solo costado usando **padding-left, padding-right, padding-top y padding-bottom**.

Border

La propiedad **border** en CSS se utiliza para especificar el **estilo, el ancho y el color** de los bordes de un elemento HTML.

```
div {
    /* ancho, estilo, color*/
  border: 1px solid red;

}
```

El valor del **“ancho”** especifica el ancho del borde. Podemos utilizar unidades absolutas o relativas.
El valor **“estilo”** define como se verá el borde. Por ejemplo, un valor **solid** implica que el borde sea una línea sin ningún detalle, mientras que, por ejemplo, **dotted** , hará que el borde sea una línea punteada.
El valor **“color”** define el color que tendrá nuestro borde.

También podemos discriminar el borde al cual queremos darle estilos, utilizando **border-left, border-right, border-top y border-bottom.**

Border-radius

La propiedad **border-radius** en CSS se utiliza para especificar la forma en que los bordes de un elemento HTML deben ser **redondeados**. Con esta propiedad, podemos crear bordes suaves y redondos en tus elementos, en lugar de bordes rectos y angulares.

```
div {
 border-radius: 10px;
 border-radius: 10px 5%; /* arriba-izq y abajo-der | arriba-der y abajo-izq */
 border-radius: 10px 11px 12px 13px; /* abajo-izq | abajo-der | abajo-der | abajo-izq */

}
```

También se puede asignar un border radius individual a cada borde del elemento:

```
div {
border-top-left-radius: 10px;
border-top-right-radius: 5%;
border-bottom-left-radius: 2rem;
border-bottom-right-radius: 1em;
}
```

Margin

La propiedad **margin** se utiliza para establecer un **margen alrededor de un elemento HTML**. El **margen es el espacio vacío alrededor** del elemento y se utiliza para separar un elemento de los demás elementos en la página.

Podemos definirla utilizando la propiedad **margin** o especificando el valor para cada costado como propiedad individual:

```
div {
margin: 1px; /*Todos los elementos*/
margin: 1px 10px; /*arriba/abajo | izq/der*/
margin: 10px 11px 12px 13px; /*arriba | der | abajo | izq*/
margin-top: 10px;
margin-right: 10px;
margin-bottom: 10px;
margin-left: 10px;
}
```

Utilizaremos la propiedad **margin** para separar un elemento de otros elementos en la página, mientras que utilizaremos la propiedad **padding** para añadir un espacio adicional dentro de un elemento para mejorar su aspecto o funcionalidad.

Reseteando estilos en css

El **reset de estilos** en CSS se realiza para establecer una base común de estilos para todos los elementos en una página web. Esto es útil porque los navegadores tienen diferentes valores predeterminados para los estilos de los elementos HTML, lo que puede hacer que una página web tenga un aspecto diferente en diferentes navegadores.

Vamos a hacer este reset aplicando los siguientes estilos al selector universal **“*”**.

```
*{
  margin:0;
  padding:0;
  box-sizing:border-box;
}
```

Recomendamos que sea lo primero que hagan a la hora de comenzar un proyecto.
Si lo consideran necesario, también podrían agregar **list-style:none;** para resetear los estilos de las listas y **text-decoration: none;** para quitar los estilos por defecto de los enlaces.

CSS Layout

El **CSS Layout** es una técnica de diseño de páginas web que permite definir cómo se deben organizar los elementos de una página en la pantalla. Se pueden usar diferentes tipos de layout en CSS, como el **layout de flujo, el layout de caja y el layout flexible**, entre otros.

El **layout de flujo** es el tipo de layout que se utiliza de forma predeterminada en la mayoría de los navegadores. En este tipo de layout, los elementos se organizan uno detrás de otro en una secuencia lineal, desde la parte superior hasta la parte inferior de la página.

El **layout de caja**, por otro lado, permite organizar los elementos en una serie de cajas que se colocan una al lado de la otra. Este tipo de layout es útil para crear estructuras más complejas, como menús de navegación y columnas laterales.

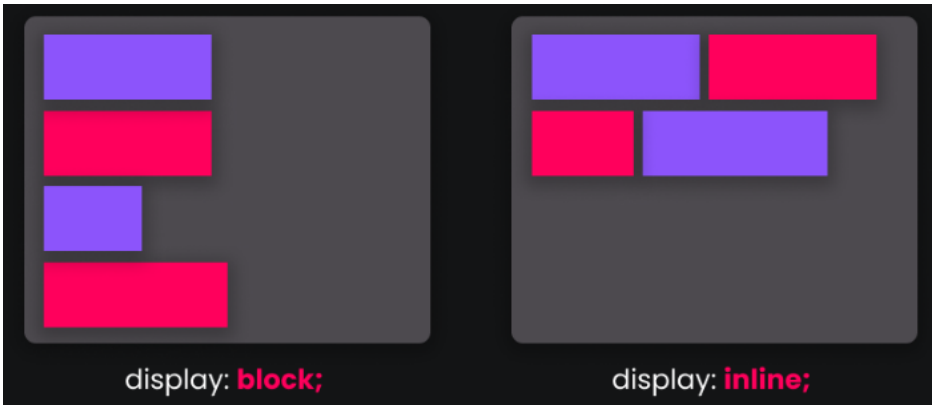
El **layout flexible** es un tipo de layout que permite que los elementos se ajusten automáticamente a diferentes tamaños de pantalla y resoluciones. Este tipo de layout es especialmente útil en la era de los dispositivos móviles, ya que permite que la página se vea bien en diferentes tipos de dispositivos.

Elementos en bloque

Los elementos en bloque son elementos que **por defecto ocupan todo el ancho disponible** de la página y **crean un nuevo bloque de contenido**. Por lo general, estos elementos tienen un margen por defecto y un salto de línea antes y después de ellos, lo que significa que no se pueden colocar otros elementos a su lado. Algunos ejemplos de elementos en bloque son los elementos **div, h1, p y ul**.

Elementos en línea

Los elementos en línea son elementos que **por defecto solo ocupan el ancho necesario para mostrar su contenido** y **no crean un nuevo bloque de contenido**. Estos elementos se colocan **uno al lado del otro en la misma línea**, y **no tienen margen ni salto de línea antes o después de ellos**. Algunos ejemplos de elementos en línea son el elemento **span, a y strong**.



Overflow

La propiedad **overflow** es una propiedad CSS que determina qué hacer con el contenido que sobrepasa los límites de un elemento contenedor. Es especialmente útil cuando se trata de elementos con un **tamaño fijo.**

Esta propiedad puede tomar los siguientes valores:

```
div{
  overflow: visible;
  overflow: hidden;
  overflow: scroll;
  overflow: auto;
}
```

1. **visible**: Muestra el contenido que sobrepasa los límites del elemento contenedor, sin hacer ninguna acción adicional.
2. **hidden**: Oculta el contenido que sobrepasa los límites del elemento contenedor, sin mostrar barras de desplazamiento o hacer cualquier otra acción.
3. **scroll**: Muestra barras de desplazamiento para permitir al usuario desplazarse por el contenido que sobrepasa los límites del elemento contenedor.
4. **auto**: Muestra barras de desplazamiento solo si es necesario, es decir, si el contenido sobrepasa los límites del elemento contenedor.

Position

La propiedad **position** en CSS es una propiedad que **determina cómo se posiciona un elemento en relación a su elemento contenedor o a la página**.

Esta propiedad puede tomar los siguientes valores:

```
div{
  position: static;
  position: relative;
  position: absolute;
  position: fixed;
}
```

Position static

Es el valor predeterminado que tienen los elementos. Posiciona al elemento de manera normal, **siguiendo el flujo del documento HTML**.

Position relative

Posiciona el elemento en r**elación a su posición normal** en el flujo del documento, pero permite moverlo mediante las propiedades **top, right, bottom y left**.

```
div{
  position: relative;
  top: 10px;
  left: 10%;
  right: 15px;
  bottom: 12px;
}
```

No es necesario utilizar todas las direcciones para posicionar el elemento.

Position absolute

Posiciona el elemento en relación a su **elemento contenedor más cercano** que tenga una **posición diferente de static**. Si no hay ningún elemento contenedor con una posición diferente de static, se posiciona en relación a la **página**.

```
div{
  position: relative;
}

div > span {
  position:absolute;
  top:10px;
  left:30px;
}
```

En este ejemplo, suponiendo que hay un **único** span que es hijo directo del div, este se posicionará según el **div** a 10px del **borde superior** del div y a 30px del **borde izquierdo** de dicho **div**.

Position fixed

Posiciona el elemento en **relación a la ventana del navegador** y lo fija en una **posición específica, incluso cuando se desplaza la página**.

```
div{
  position: fixed;
  top: 0px;
  left: 0px;
}
```

En el ejemplo podemos ver que estamos posicionando a **nuestro div** en la **parte superior izquierda de la página**, y se mantendrá ahí incluso aunque hagamos scroll.

IMPORTANTE: Queda un valor de position más por ver que es el position **sticky**, pero lo vamos a estar viendo dentro de algunas clases. Por otro lado, es importante que entendamos que no vamos a estar usando los distintos tipos de **position** para mover todos los elementos que tiene nuestra página o intentar centrarlos. Lo usaremos específicamente cuando se trate de algo que necesariamente debe colocarse en un punto determinado de la página o en un punto específico relacionado a otro elemento de la misma.

Nucba tip: *Sigan investigando sobre los temas aquí expuestos. Practiquen, apoyense en la documentación oficial y sobre todo mantengan la constancia y la curiosidad a medida que vayan aprendiendo cosas nuevas.*

#HappyCoding 🚀