

Agencia de  
Aprendizaje  
a lo largo  
de la vida

# Desarrollo Fullstack



# Les damos la bienvenida

Vamos a comenzar a grabar la clase

## Clase 05

### CSS Inicial

- ▶ Fuentes y Tipografías
- ▶ Colores
- ▶ Iconos

## Clase 06

### CSS Inicial

- ▶ Unidades de medida
- ▶ Modelo de Caja
- ▶ Positions

## Clase 07

### CSS Intermedio

- ▶ Transiciones
- ▶ Transformaciones
- ▶ Animaciones
- ▶ Pseudoselectores
- ▶ Pseudoclases

# CSS

Estructura en nuestros estilos



# Unidades de Medida

Existen **muchas** y cada una tiene una aplicación para cada caso particular.

Lo primero que debemos saber es que al igual que los enlaces en HTML existen de **2 tipos**, unidades **relativas** y **absolutas** cuya diferencia radica en si ese valor siempre va a tomar el mismo tamaño o si va a estar relacionado al tamaño de algo más.

# Absolutas

Son medidas **fijas** y no dependen de ningún otro factor.

**Ideales** en contextos donde las medidas no varían como en los **medios impresos** (documentos, impresiones, etc...), pero **poco adecuadas** para la web, ya que no se adaptan a diferentes resoluciones o pantallas, que es lo que tendemos a hacer hoy en día.

Si bien existen muchas como **cm** (centímetros), **mm** (milímetros), **in** (pulgadas), **pc** (picas), **pt** (puntos), etc...

La más conocida son los **Píxeles** por su fácil uso y aplicación práctica en pantallas.

```
img {  
  width: 300px;  
}
```

# Relativas

Mucho más **potente y flexible** en CSS. Al contrario de las unidades absolutas, dependen de algún otro factor (resolución, tamaño de letra, etc...). Tienen una curva de aprendizaje más compleja, pero **son ideales para trabajar en dispositivos con diferentes tamaños**, ya que son muy versátiles.

**em** -> 1em = tamaño de fuente relativo a la **herencia** o al *valor por defecto* del **navegador**.

**rem** -> 1rem = tamaño de fuente **relativo** al valor por defecto del **navegador**.

**vw** -> 100vw = total del **ancho** visible del navegador.

**vh** -> 100vh = total del **alto** visible del navegador

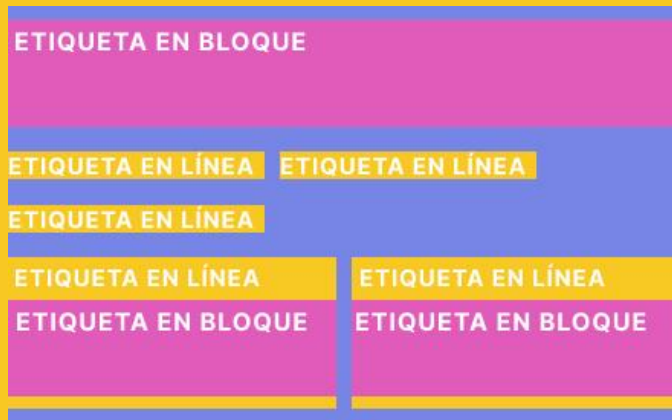
**%** Porcentaje Relativo al tamaño del elemento padre.

# Displays

Recordemos que, por defecto, **cada elemento HTML** tiene un tipo de representación concreta. Esos valores eran display **block** o **inline** y estaban relacionados de forma nativa a cada etiqueta.



Sin embargo, estos comportamientos  
nativos pueden ser *modificados*.



# Displays

Esta propiedad cambia el tipo de representación del elemento indicado y si bien **puede tomar muchos** valores diferentes, por ahora nos concentraremos en **4** de los cuales ya conocemos algunos.

```
display: block | inline | inline-block | none;
```

# Valores de display

## block

Ocupan el **100%** del ancho de su contenedor y comienzan en una nueva línea.

## inline

Ocupan el ancho de su contenido y **no aceptan** propiedades de **width**, **height** o **margins** y **padding**s superiores.

## inline-block

La **combinación de los anteriores**, ocupa el ancho de su contenido pero **sí acepta** que se modifique su tamaño o sus propiedades de caja.

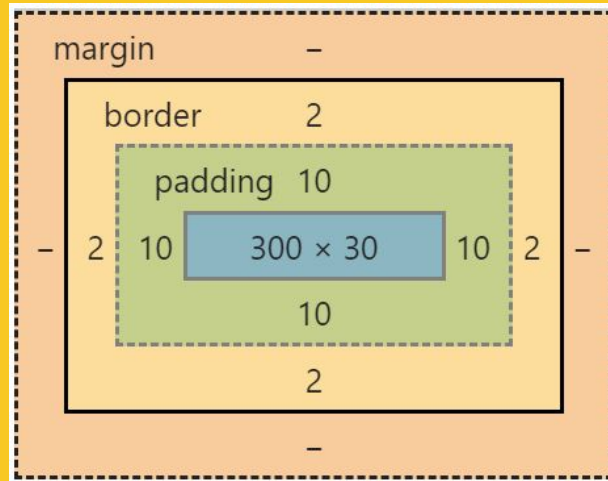
## none

Este valor resulta en que **el elemento seleccionado no sea mostrado** ni ocupe espacio en el lugar donde debería estar.



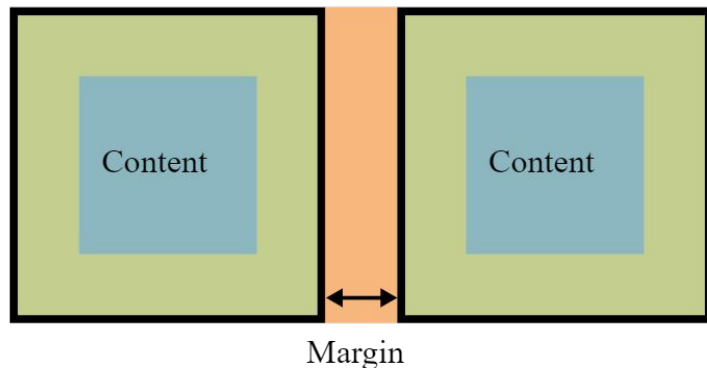
# Modelo de Caja

Es un **sistema** que tiene el navegador para interpretar las diferentes partes de lo que solemos denominar **cajas**, es decir, un elemento HTML con ciertos **límites y dimensiones**.



# Propiedad *margin*

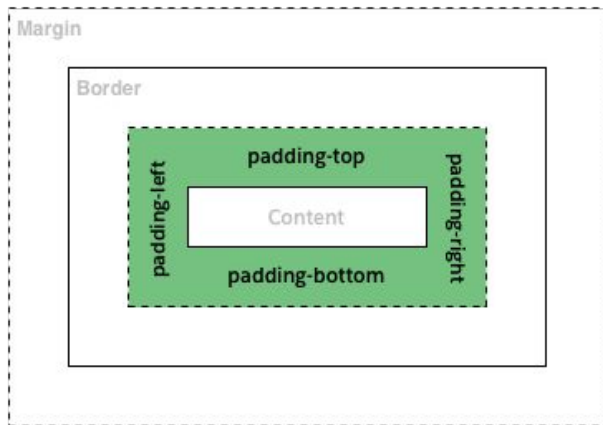
Se utiliza para **crear espacio** alrededor de los elementos, **FUERA** de los **bordes definidos**.



```
/* top right bottom left */  
margin: 10px 20px 10px 20px;  
/* top right/left bottom */  
margin: 10px 20px 10px;  
/* top/bottom right/left */  
margin: 10px 20px;  
/* top/right/bottom/left */  
margin: 10px;
```

# Propiedad *padding*

Se utiliza para **crear espacio** alrededor de los elementos, **DENTRO** de los **bordes definidos**.



```
/* top right bottom left */  
padding: 10px 20px 10px 20px;  
/* top right/left bottom */  
padding: 10px 20px 10px;  
/* top/bottom right/left */  
padding: 10px 20px;  
/* top/right/bottom/left */  
padding: 10px;
```

# Propiedad **border**

Permiten especificar el **estilo**, el **ancho** y el **color** del borde de un elemento.



**solid**



**dotted**



**dashed**



**double**



**groove**




**ridge**



**inset**



**outset**

```
/* ancho estilo color*/  
border: 2px solid  #f32872;
```

# Overflow

Sucede cuando **superamos** los límites de tamaño de **nuestros contenedores**.

Dependiendo el caso, puede **generar scroll vertical u horizontal**, ocultar el contenido sobrante o **dejarlo simplemente que fluya**.

CSS  
IS  
AWESOME



# Valores de overflow

## auto

Se colocan **barras de desplazamiento** (sólo las necesarias).

## hidden

Se **oculta** el contenido que sobresale.

## visible

Se **muestra el contenido que sobresale** (comportamiento por defecto)

## scroll

Se colocan **barras de desplazamiento** (horizontales y verticales).

Lorem ipsum dolor sit amet,  
consectetur adipisicing elit.  
Magnam fuga odio sequi  
delectus rem? Tempora  
accusantium pariatur quaerat  
repudiandae explicabo  
laudantium itaque sapiente  
delectus labore eaque dicta  
consectetur dignissimos  
corporis!

Lorem ipsum dolor sit amet,  
consectetur adipisicing elit.  
Itaque qui consectetur quaerat  
quo hic quidem illum ipsum  
dolore modi. Numquam repellat  
cum iure quisquam veniam  
praesentium rerum nobis  
voluntatem tempora

Lorem ipsum dolor sit amet,  
consectetur adipisicing elit.  
Odit labore laudantium nisi  
aliquid nulla qui quisquam  
recusandae quis corporis  
expedita ipsum debitis mollitia  
ducimus ex enim deleniti  
resum quos Tempora

Lorem ipsum dolor sit amet,  
consectetur adipisicing elit.  
Animi minus voluptate mollitia  
optio maiores harum expedita  
numquam accusantium ut  
eligendi rem cupiditate illum  
aspernatur doloremque  
possimus vitae voluptates

# POSITIONS

No te dejes vencer por un diseño

# Positions

Hasta el momento aprendimos a manejar y posicionar los elementos de una web en base a un **flujo estático** y continuo donde **las cajas** se iban creando en el orden en el cual **fueron escritas** en el HTML.

Gracias a los **positions**, vamos a poder modificar el flujo estático de nuestros elementos, permitiendo **superposiciones** o cambios referencia **sobre** los que **las cajas** están dispuestas.

# Position y sus valores

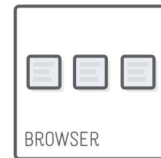
La propiedad `position:` cuenta con los siguientes valores:

`static` | `relative` | `absolute` | `fixed` | `sticky`

`static`

**Valor por defecto.** Este valor indica que el elemento debe adoptar el flujo natural del sitio.

STATIC



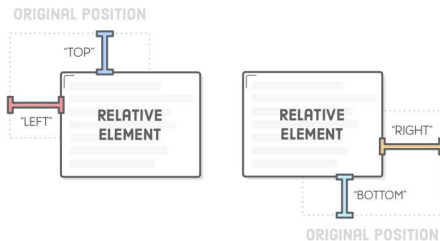
# Position y sus valores

## relative

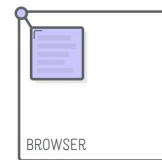
Se comporta igual que static a menos que le agreguemos las propiedades: `top` | `bottom` | `right` | `left` causando un **reajuste en su posición** y *sin modificar el espacio que ocuparía originalmente.*



RELATIVE POSITIONING



# Position y sus valores



ABSOLUTE POSITIONING

**absolute**

La posición de una caja se establece de forma **absoluta** respecto de su elemento contenedor relative, o el body por defecto.

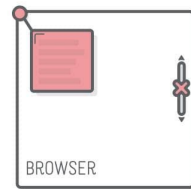
***El resto de elementos de la página ignoran la nueva posición del elemento.***



# Position y sus valores

## fixed

Hace que la caja esté **posicionada con respecto a la ventana del navegador**, lo que significa que **se mantendrá en el mismo lugar** incluso al hacer scroll en la página.



FIXED POSITIONING



# Position y sus valores

## sticky

La caja **se mantiene static** *hasta que el scroll del navegador llega a ella* y *se comporta como fixed*. Una vez que el tamaño de su contenedor llega a su fin, **vuelve a comportarse como static**.





# z-index

En los momentos que **nuestras cajas con position se superpongan**, podemos utilizar la propiedad **z-index** para **manejar el orden de las capas.**

# No te olvides de dar el presente

## **Recordá:**

- **Revisar la Cartelera de Novedades.**
- **Hacer tus consultas en el Foro.**

**Todo en el Aula Virtual.**

# Gracias