DISEÑO DE INTERFACES WEB UD-2

MODELO CAJA

ÍNDICE

[BLOCK VS INLINE 3](#_Toc147867656)

[DISPLAY: NONE VS VISIBILITY: HIDDEN 3](#_Toc147867657)

[PARTES DE LA CAJA 4](#_Toc147867658)

[box-sizing 4](#_Toc147867659)

[width y height 4](#_Toc147867660)

[max/min width y height 5](#_Toc147867661)

[overflow 5](#_Toc147867662)

[margin 6](#_Toc147867663)

[A. MÁRGENES POR DEFECTO 6](#_Toc147867664)

[B. COLAPSO DE MÁRGENES 6](#_Toc147867665)

[C. MÁRGENES EN ELEMENTOS INLINE 6](#_Toc147867666)

[D. MÁRGENES NEGATIVOS 6](#_Toc147867667)

[padding 6](#_Toc147867668)

[EFECTOS EN CAJA 7](#_Toc147867669)

# BLOCK VS INLINE

Todos los elementos HTML disponen de una propiedad display que indica cómo los elementos van a ser mostrados, casi todos los elementos tienen la propiedad con valor block o inline.

* **block:** empieza en una nueva línea y ocupa el ancho disponible. En principio sólo pueden tener hijos inline (aunque algunos block admiten a otros block). Elementos de este tipo son <h1>, <p> o <div>.
* **inline:** no empieza en una nueva línea y ocupa sólo el ancho necesario. En principio solo puede tener hijos inline. Elementos de este tipo son <span><em><a>.

Podemos cambiar el comportamiento de los elementos con CSS.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

# DISPLAY: NONE VS VISIBILITY: HIDDEN

Ambos valores ocultan el elemento:

* **display: none:** El elemento desaparece de la estructura. No sólo no se ve, sino que no afecta al resto de elementos.

Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

* **visibility: hidden:** El elemento simplemente se oculta. El layout del documento no cambia: todos los elemento se mantienen ubicados igual que si el elemento fuera visible.

Interfaz de usuario gráfica, Aplicación, Teams

Descripción generada automáticamente

# PARTES DE LA CAJA

Imagen que contiene Texto

Descripción generada automáticamente

El ancho y el alto de un elemento hace referencia al contenido, pero con los padding y margin el tamaño cambia.

# box-sizing

Con el atributo box-sizing podemos hacer el tamaño haga referencia al contenido + padding.

El área total que ocupa el elemento (width + padding horizontal + margin horizontal) \* (height + padding vertical + margin vertical).

Es decir, el ancho y el alto sólo hacen referencia a contenido. A partir de CSS3 es posible indicar a que hace referencia o al contenido o al contenido más el padding.

Los valores que admite son content-box, border-box, initial e inherit.

# width y height

El ancho y el alto son propiedades no heredables, aunque podemos forzarlas.

Hay que tener en cuenta que no afectan a elementos inline.

Aceptan valores en longitud, porcentaje, auto e inherit. Auto hará que sea el propio navegador el que calcule el tamaño (tan ancho como el navegador o como el elemento que lo contenga).

# max/min width y height

Con esta propiedad vamos a controlar los tamaños máximos y mínimos. Están pensadas para definir los límites de los elementos. Sólo funcionan en los block y elementos que son reemplazos (como img).

Sólo afectan al contenido.

Prioridades:

* min-width > max-width > width.
* min-height > max-height > height.

# overflow

Lo más habitual es no definir la altura de los elementos y que esta sea calculada en función del tamaño del texto (así si el usuario cambia el tamaño, todo se debería adaptar).

Si definimos una altura hay que estudiar que pasa si el contenido no cabe, si se desborda.

Sólo funcionan en los block y elementos que son reemplazados (como img).

Tiene 4 valores posibles:

* visible

Texto, Escala de tiempo

Descripción generada automáticamente

* hidden

Texto

Descripción generada automáticamente

* scroll

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

* auto: El navegador decide.

# margin

Sirve para definir el tamaño del margen.

Admite 4 tipos de datos: longitud, auto, initial e inherit. El valor auto hará que el navegador decida la cantidad de margen necesario para adaptarse o rellenar el espacio disponible.

La longitud se puede expresar como % del ancho del elemento padre.

Admite además 4 formas de expresarlo:

* 4 valores (4px, 12px, 20px, 5px): Afecta a margin-top, right, bottom y left.
* 3 valores (4px, 12px, 5px): El primero afecta a top, el segundo a right y left y el último a bottom.
* 2 valores (2px, 20px): El primero afecta a top y bottom, el segundo a los laterales.
* 1 valor (20px): Afecta a todos por igual.

## MÁRGENES POR DEFECTO

El navegador puede definir alrededor de encabezados, párrafos, etc. Realizar un reinicio de los márgenes (por ejemplo \* {margin: 0px; padding: 0px;} es una buena práctica.

## COLAPSO DE MÁRGENES

Si los márgenes inferior y superior de dos elementos se solapan, sólo se usa el más grande.

## MÁRGENES EN ELEMENTOS INLINE

Los márgenes superior o inferior no afectan, pero el derecho e izquierdo sí, aunque eso implique crear varias líneas.

## MÁRGENES NEGATIVOS

Es posible definir márgenes negativos. El efecto que se crea es el de “atraer” a los elementos del lado correspondiente al que se haya aplicado el margin negativo.

# padding

Sirve para definir el tamaño del padding y permite 4 tipos de datos: longitud, %, initial e inherit..

La longitud se puede expresar como % del ancho del elemento padre.

Admite además las mismas 4 formas de expresarlo que margin.

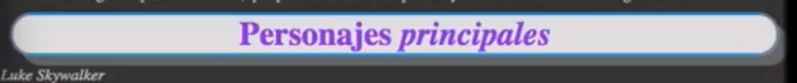
# EFECTOS EN CAJA

**border-radius**: 20px 🡪 indicamos el punto de partida del radio desde cada lado

border-radius: 50px/10px 🡪 aplica un radio distinto horizontal y verticalmente

border-radius: 50% 🡪 el punto de partida está a un porcentaje del tamaño de cada lado

**box-shadow**: 10px 10px  rgb(100,100,100) 🡪 sombra de la caja con un color determinado

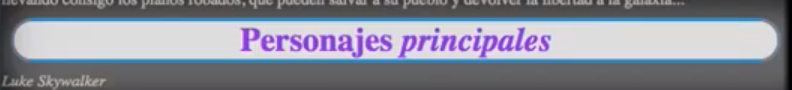


Con un tercer parámetro añadimos difuminado de la sombra:

box-shadow: 10px 10px **30px**  rgb(100,100,100)

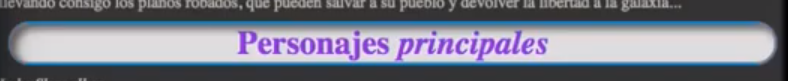
Con un cuarto parámetro esparcimos la sombra por todos los lados de la caja:

box-shadow: 10px 10px 30px **20px**  rgb(100,100,100)



con el parámetro **inset** la sombra se aplica por el interior de la caja, dando un efecto 3d.

box-shadow: 5px 5px 10px rgb(100,100,100) **inset**



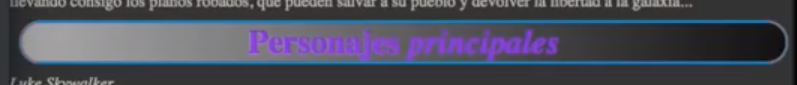
Uso de gradientes de color:

background: -webkit-linear-gradient(top,#111111, #aaaaaa);

cada navegador necesita un parámetro de renderizado para interpretar el gradiente,

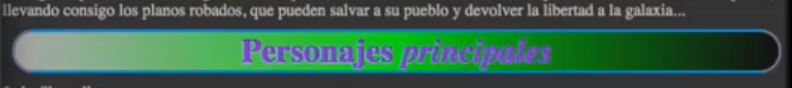


background: -webkit-linear-gradient(right,#111111, #aaaaaa);



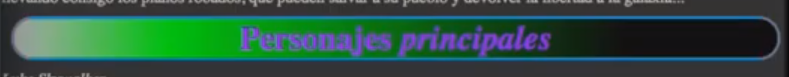
Podemos hacer que el gradiente pase por varios colores:

background: -webkit-linear-gradient(right,#111111, #11dd11,  #aaaaaa);



Con unos % podemos indicar la proporción de los colores en el gradiente:

background: -webkit-linear-gradient(right, #111111 20%, #11dd11 80%,  #aaaaaa);



Podemos indicar una inclinación del gradiente:

background: -webkit-linear-gradient(30deg, #111111 20%, #11dd11 80%,  #aaaaaa);

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

También tenemos los gradientes circulares o elípticos, además de los lineales que acabamos de ver.

background: radial-gradient(circle, #111111, #dddddd);

Una pantalla de computadora

Descripción generada automáticamente

background: radial-gradient(ellipse, #111111, #dddddd);

Texto

Descripción generada automáticamente con confianza baja

Cómo poner una imagen en el borde de la caja:

border-image-source: url(“imagen.png”);

border-image-slice: 10

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

con slice cortamos el perímetro de la imagen con un grosor de 10px (lo indicado), y ese borde se ajusta al de nuestra caja.