



04-01 Introducción a CSS

Nombre		Curso	
Apellidos		Fecha	

1.- Introducción a CSS.

Las hojas de estilo en cascada (CSS - Cascading Style Sheets) son un estándar W3C que define la presentación de los documentos Web, es decir, el modo en el que se muestra un documento en pantalla o se suministra al usuario, ya sea por el monitor, en la pantalla del teléfono móvil o leído por un lector de pantalla. Lo más importante es que con CSS se mantienen las instrucciones de presentación separadas del contenido del documento HTML.

Las hojas de estilo en cascada como hoy las conocemos, comenzaron cuando Håkon Lie publicó su primer borrador de hojas de estilo HTML en cascada, al que pronto se le unió **Bert Bos**, gran impulsor de este estándar.

- ✓ **CSS Nivel 2, Revisión 1** que es simplemente una recomendación que realiza unos ajustes menores a CSS2 basándose en la experiencia de trabajo con él entre 1998 y 2004.
- ✓ **CSS Nivel 3**, basada en módulos, añade soporte para texto flotante vertical, mejor manejo de tablas, idiomas internacionales y una mejor integración con otras tecnologías como XML como SVG, MathML (*Mathematical Markup Language. Es un lenguaje de marcado basado en xml, cuyo objetivo es expresar notación matemática de forma que distintas máquinas puedan entenderla, para su uso en combinación con xhtml en páginas web, y para intercambio de información entre programas de tipo matemático en general*) y SMIL.
- ✓ **CSS Nivel 4** Parece que no existirá pronto
<https://www.silocreativo.com/css4-nueva-version-css-nunca-existira/>

Sin duda, no existe ninguna desventaja por utilizar CSS en la maquetación de páginas web, son todas ventajas y, entre ellas, podemos destacar las siguientes:

- ✓ **Mayor control** en el diseño de las páginas: Se puede llegar a diseños fuera del alcance de HTML.
- ✓ **Menos trabajo**: Se puede cambiar el estilo de todo un sitio con la modificación de un único archivo.
- ✓ **Documentos más pequeños**: por quitar los tags de formato.
- ✓ **Documentos mucho más estructurados**: Los documentos bien estructurados y accesibles.
- ✓ **El HTML de presentación está Obsoleto**

¿Si quieres definir la presentación de una página web utilizas...?

CSS

XHTML

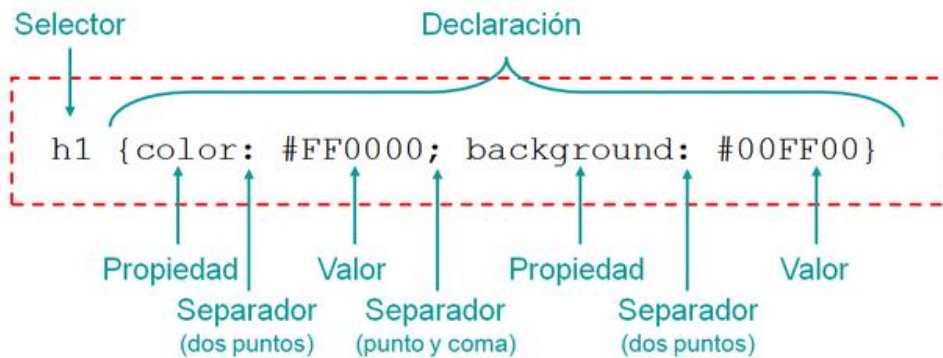
Javascript

PHP

con XHTML defines la estructura, con CSS la presentación y con los lenguajes de script de cliente y servidor creas webs más interactivas y dinámicas

Sintaxis

Los conceptos de css son los siguientes



1.1.- Añadir estilos a un documento con CSS.

A la hora de aplicar las reglas de estilo a un documento HTML, debes tener en cuenta que existen tres modos distintos:

- ✓ Estilos en línea.
- ✓ Hojas de estilos incrustados.
- ✓ Hojas de estilos externas: vinculadas (link) o importadas. (@import)

1.2.1.- CSS EN LÍNEA

El código CSS se escribe directamente en etiquetas HTML quiere formatear. Escribir este método no requiere escribir un selector en código CSS. Este método solo debe usarse si desea formatear un elemento una vez.

Sintaxis

```
style=" Propiedad:valor; "  
  
<h1 style="color: red;"> Daw Playamar CSS </h1>
```

En el ejemplo anterior, el elemento <h1> está en formato usando rojo, otros elementos, no usará rojo porque este formato solo se aplica a los elementos de párrafo que se especifican en el código CSS. Escribir CSS de esta manera comienza con la palabra "estilo = luego seguido de la sintaxis Propiedad: valor;"

No se usa mucho a día de hoy

1.2.2.- CSS incrustado

Podemos pegar el código CSS entre las etiquetas <head> y </head>. La escritura CSS de esta manera está precedida inicialmente por la etiqueta <style> </style>.

```
<head>  
  <style>  
    Etiqueta1 { Propiedad:valor; } </style>  
</head>
```

```
<head>  
  <style type="text/css">  
    h1 {color : red;}  
  </style>  
</head>
```

1.2.3- Hojas de estilo CSS externas con link

El código CSS externo está escrito en un archivo separado guardado con el sufijo ".css" o la extensión ".css" y enlazados desde el head con la sintaxis : <link rel="stylesheet" href="file.css">

Luego debe llamar al archivo CSS en todas las páginas web que cree. De esta manera, solo necesita tener un conjunto de código CSS que se utilice para todas sus páginas web. Entonces, hay dos pasos para implementar CSS de esta manera.

```
file.css
Etiqueta1 { Propiedad:valor; }
Etiqueta2 { Propiedad:valor; }

<head>
<link rel="stylesheet" href="file.css">
</head>
```

```
file.css
p {font-family: arial;}
h1 {color: red; }

<head>
<link rel="stylesheet" href="file.css">
</head>
```

En un archivo con el bloc de notas u otro editor de texto y asígnele un nombre, por ejemplo: style.css, luego escriba el código css en el archivo.

```
p {font-family: arial; font-size: small;} h1 {color: red; }
```

El segundo paso es llamar al archivo style.css de todas las páginas web. Para ello, ingrese el código a continuación, entre las etiquetas <head> y </head>

1.2.4- Hojas de estilo CSS externas con @import

@import está pensado para **incluirse dentro de las hojas de estilo**, su uso es poder enlazar hojas de estilo dentro de hojas de estilo. Para enlazar cualquier archivos externos dentro de html existe el link. (es mejor usar link)

```
file1.css
file.css
Etiqueta1 { Propiedad:valor; }
@import "file1.css"

<head>
<style > @import "file.css" </style >
</head>
```

```
file1.css
p {font-family: arial;}
file.css
@import "file1.css"
h1 {color: red; }

<head>
<style > @import "file.css" </style >
</head>
```

Comentarios

Al igual que en los documentos HTML, en las hojas de estilo se pueden incluir comentarios pero, en este caso, se escriben del siguiente modo: /* Este es un comentario */

1.3.- Conceptos clave de CSS.

Para que te puedas familiarizar con el comportamiento de CSS, es importante comprender una serie de conceptos clave.

1.3.1-Estructura y herencia.

Un documento HTML tiene una estructura determinada que es equivalente a un árbol genealógico cuando se hace referencia a la relación entre elementos:

- ✓ Se dice que un elemento es "hijo" de otro si está contenido directamente en él y este último pasa a ser su "padre". Por ejemplo: el elemento p es hijo del elemento body y el elemento body es padre del elemento p.
- ✓ Los elementos que tienen el mismo padre son "hermanos". Por ejemplo: un elemento p puede ser hermano de otro elemento p si ambos son hijos directos del elementobody.

Controlar la relación padre-hijo es fundamental para el funcionamiento de CSS. Un hijo puede "heredar" valores de propiedad de su padre. Con una buena planificación, la herencia puede emplearse para hacer más eficiente la especificación de los estilos.

Este principio por el que algunas reglas se ignoran y otras se heredan nos introducen un concepto muy importante: **"la cascada"**.

1.3.1- Prioridad de las hoja de estilo. *la "cascada"*.

La "cascada", de las hojas de estilo en cascada, se refiere a lo que ocurre si varias fuentes de información de estilo quieren dar formato al mismo elemento de una página. Cuando un navegador encuentra un elemento para el cual hay varias declaraciones de estilo, las ordena de acuerdo al origen de la hoja de estilo, la especificidad de los selectores y el orden de la regla para poder determinar cuál aplicar.

Los navegadores otorgan un peso distinto a las hojas de estilo que, ordenadas de menor a mayor peso, son:

- Prioridad

- ✓ Hojas de estilo del navegador.
- ✓ Hojas de estilo del lector.
- ✓ **Hojas de estilo de la persona que ha diseñado la página web.**
- ✓ Declaraciones de estilo !important del lector.

+ Prioridad

Además de este orden, existe otra jerarquía de pesos que se aplican a las hojas de estilo creadas por la persona que ha diseñado la página web. Es importante entender esta jerarquía y tener en cuenta que las reglas de estilo que están al final de la lista ignorarán a las primeras. La siguiente lista, que como la anterior está ordenada de menor a mayor peso, muestra esta otra jerarquía:

- Prioridad

1. ✓ Hojas de estilo externas vinculadas (con `link` en el Head).
2. ✓ Hojas de estilo externas importadas (con `@import` dentro del elemento `style` en la Head).
3. ✓ Hojas de estilo incrustadas (con el elemento `style` en la Head).
4. ✓ Estilos en línea (con el atributo `style` en la etiqueta del elemento).
5. ✓ Declaraciones de estilo marcadas como `!important`.

+ Prioridad

Especificidad del selector.

Hasta ahora se tuvieron en cuenta las distintas fuentes de la información del estilo. Pero aún puede existir algún conflicto a nivel de reglas. Por esa razón, "la cascada" continúa a nivel de reglas. Lo verás mejor con el siguiente ejemplo, que podría estar en una hoja de estilo externa o incrustada. En él se muestran dos reglas que hacen referencia al elemento `strong`.

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

En el ejemplo anterior, todo el texto del documento (X)HTML marcado con la etiqueta `strong` aparecerá en color rojo. Sin embargo, si el texto marcado con la etiqueta `strong` aparece dentro de una cabecera de primer nivel (`h1`), su color será azul.

Esto ocurre porque un elemento en un contexto determinado es más específico que en un contexto general y, por lo tanto, tiene más peso.

Cuanto más específico sea el selector se le dará más Prioridad .

Orden de las reglas.

Cuando una hoja de estilo contiene varias reglas en conflicto de igual peso, sólo se tendrá en cuenta la que está en último lugar. En el siguiente ejemplo, todas las cabeceras de primer nivel del documento serían rojas porque se impone siempre la última regla:

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

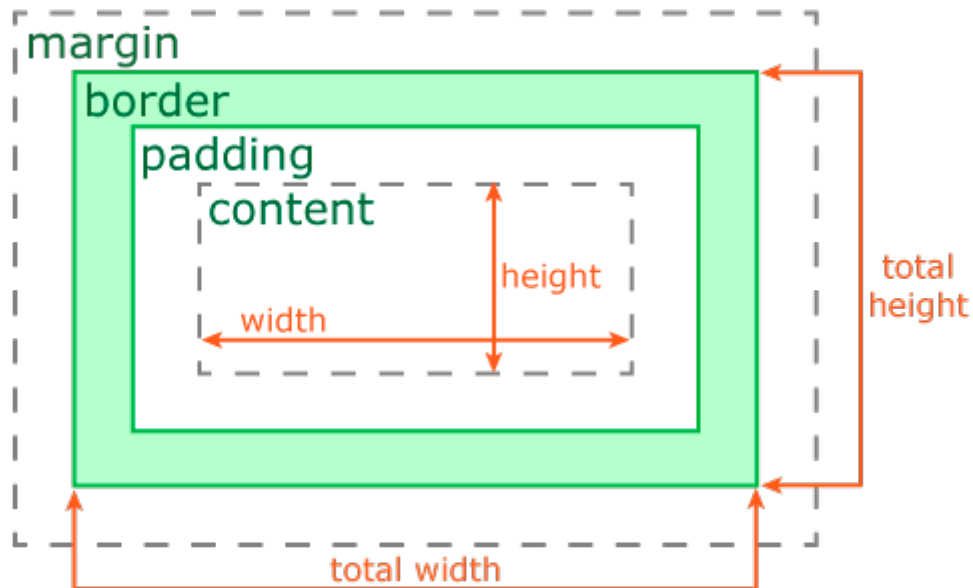
1.4.- El modelo de cajas de CSS.

El modelo de cajas es un concepto fundamental para comprender el funcionamiento de las hojas de estilo. De acuerdo con este modelo, todos los elementos de una página web generan una caja rectangular alrededor llamada "caja del elemento".

En estas cajas se pueden configurar propiedades como bordes, márgenes y fondos (entre otras). Las cajas también se pueden emplear para posicionar los elementos y diseñar la página.

Las cajas de elementos, tal y como muestra la imagen, están hechas de cuatro componentes principales:

- ✓ **Contenido del elemento:** es lo que está en el núcleo de la caja está el.
- ✓ **Relleno (padding):** es el espacio que rodea al contenido.
- ✓ **Borde (border):** es la parte que perfila el relleno.
- ✓ **Margen (margin):** es el espacio que rodea al borde, la parte más externa del elemento.



Características fundamentales del modelo de cajas

- ✓ **El relleno, los bordes y los márgenes opcionales**, si ajustas a cero sus valores se eliminarán de la caja.
- ✓ **Rellenos extensibles** un color o imagen que apliques de fondo al elemento se extenderá por el relleno.
- ✓ Los bordes se generan con propiedades de estilo que especifican su estilo (por ejemplo: sólido), grosor y color. Cuando el borde tiene huecos, el color o imagen de fondo aparecerá a través de esos huecos.
- ✓ Los márgenes siempre son transparentes (el color del elemento padre se verá a través de ellos).
- ✓ Cuando definas el largo de un elemento estás definiendo el largo del área de contenido (los largos de relleno, de borde y de márgenes se sumarían a esta cantidad).
- ✓ Puedes cambiar el estilo de los lados superior, derecho, inferior e izquierdo de una caja de un elemento por separado.



Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

1.4.1.- Propiedades del Área de contenido y relleno.

Área de contenido. **width** y **height**

El área de contenido es la parte más interna de la caja

Propiedades

width : ancho

height : su altura

max-height, max-width, min-height, min-width. : establecen la altura/anchura máxima/minima de un elemento. Evita que el valor utilizado de la propiedad **height** o **width** sea mayor o menor que el valor especificado.

```
div {width:100px; height:200px; }
```

```
div {width:100px; height:200px; }
```

Relleno. (**padding**)

El relleno es una cantidad opcional de espacio existente entre el área de contenido de un elemento y su borde. Es conveniente que establezcas un valor de relleno cuando pones borde a un elemento.

Para establecer el relleno se emplea la propiedad **padding**. Esta propiedad, como muchas otras en CSS, obliga a configurar los valores en un orden determinado. Estos valores y su orden son: **top** (arriba), **right** derecha, **bottom** (debajo) y **left** (izquierda).

El ejemplo siguiente muestra una tabla que agrupa algunos ejemplos de la asignación de valores y su interpretación por CSS. En todos los ejemplos se ha empleado como selector el elemento **div**.

Ejemplos

```
div {padding: 3px 20px 3px 20px; }
```

Establece un relleno para todos los elementos **div** de 3 píxeles por encima del área de contenido, 20 píxeles a su derecha, 3 píxeles por debajo y 20 píxeles a su izquierda.

```
div {padding: 3px 20px 3px; }
```

Al omitir un valor, asume que el valor del relleno a la izquierda es el mismo que el de la derecha.

```
div {padding: 3px 20px; }
```

Al omitir dos valores, asume que el primer valor corresponde al relleno por encima y por debajo del área de contenido y, el segundo valor corresponde al relleno a la derecha y a la izquierda.

```
div {padding: 3px; }
```

Al omitir tres valores, asume que ese valor es el mismo para todos.

Otras características interesantes del relleno son:

- ✓ El valor del relleno se sumará al de **width** ya definido en el elemento.
- ✓ Su color es el mismo al del área de contenido.
- ✓ El relleno nunca se "colapsa". por los márgenes que sí se colapsan.

En el siguiente ejemplo se muestra cómo configuramos el relleno y el color de fondo del área de contenido de los elementos **h1** del documento

El color del área de relleno será el mismo que el del elemento.

```
h1 {padding: 4px 10px; background: #ccc; }
```


1.4.2.- Bordes.

Un borde es una línea dibujada alrededor del área de contenido de un elemento y de su relleno (**padding**), aunque ya vimos que éste último era opcional.

Los bordes funcionan, a la hora de establecer su valores, de la misma manera que el relleno visto anteriormente, siguiendo un orden: superior, derecho, inferior, izquierdo, siempre en el sentido de las agujas del reloj y comenzando en las 12. Es fácil de recordar.

Se pueden establecer valores distintos para cada uno de los bordes y omitir valores, al igual que hacíamos con el relleno. Podemos configurar el estilo del borde, su grosor y su color. Las propiedades que nos permiten hacerlo son:

✓ **Border-style:** con esta propiedad configuramos el estilo del borde. Esta propiedad es, sin duda, la más importante del borde, ya que, si no está presente el borde no existirá.

La propiedad **border-style** puede tener los valores: **none**, **hidden**, **dotted**, **dashed**, **solid**, **double**, **groove**, **ridge**, **inset**, **outset** e **inherit**.



En el siguiente ejemplo configuramos cada uno de los lados de la caja con un borde distinto.

```
div {border-style: solid dashed dotted double; }  
div {border-style: solid dashed dotted double; }
```

✓ **Border-width:** con esta propiedad configuramos el grosor del borde.

Los valores que puede tomar esta propiedad son: **thin**, **medium**, **thick**, **inherit** o un valor concreto en píxeles. Si no se especifica esta propiedad tomará medium como valor por defecto. En el siguiente ejemplo configuramos un grosor distinto en cada uno de los lados del borde.

```
div {border-style: solid; border-width: thin medium thick 12px; }
```

✓ **Border-color:** con esta propiedad configuramos el color del borde.

Si no especificamos el color el elemento coge el del "primer plano", es decir que si, por ejemplo, tenemos una caja en cuyo interior hay texto, el color del borde será el color del texto. Existe el color **transparent**.

En el siguiente ejemplo especificamos un estilo y un ancho igual para todos los bordes y un color distinto para cada borde.

```
div {border-style: solid; border-width: 4px; border-color: #333 #red rgb(0,0,255) #0044AC; }
```

La propiedad **border** une todas las propiedades "border" vistas hasta ahora. En esta propiedad, a diferencia de las anteriores, no hay que colocar los valores en ningún orden concreto. La propiedad **border** se emplea cuando se quieren configurar los cuatro lados por igual.

También tenemos las propiedades: **border-top**, **border-right**, **border-bottom** y **border-left**.

```
h1 {border: .5em solid blue; }  
h2 {border-left: solid blue .5em; }  
h3 {border-right: solid .5em; }
```

Ejemplo con varios bordes

```
<!DOCTYPE html>  
<html>
```



Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web

IES Playamar

```
<head>
<meta charset="utf-8">
<title>Page title</title>
<style>
  p.one {
    border-style: solid;
    border-width: 5px;
  }
  p.two {
    border-style: solid;
    border-width: medium;
    border-color: red;
  }
  p.three {
    border: 3px dashed #ffffff;
    background-color: #0000FF;
    color: #ffffff;
  }
</style>
</head>

<body>
  <p class="one">First paragraph.</p>
  <p class="two">Second paragraph.</p>
  <p class="three">Third paragraph.</p>
</body>
</html>
```

1.4.3.- Márgenes.

El margen es la cantidad de espacio que se puede añadir alrededor del borde de un elemento.

Esta propiedad se configura con la propiedad **margin**. Al configurar esta propiedad debemos tener en cuenta que, a la hora de establecer los valores para los márgenes, hay que emplear la misma filosofía que con la propiedad **padding**.

Los márgenes **top** y **bottom** de dos elementos que van seguidos se "colapsan". Es decir, se asume como margen entre ambos elementos el mayor de ellos. El siguiente ejemplo muestra lo que ocurre cuando tenemos dos elementos un **h1** y un **h2** colocados uno a continuación del otro.

```
h1 {margin: 10px 20px 10px 20px; }  
h2 {margin: 20px; }
```

En el primer caso el margen superior e inferior es de 10px. En el segundo caso es de 20px. El espacio resultante entre los dos elementos será de 20px.

Por el contrario, si fuesen dos elementos colocados "uno al lado del otro" (como dos elementos **span**), deberíamos tener en cuenta que los márgenes **right** y **left** no se colapsan, sino que se suman.

¿En qué orden se dan las medidas de los márgenes en CSS?

1. **top bottom right left**
2. **left right top bottom**
3. **top left right bottom**
4. **top right bottom left**

@media

CSS introduce la posibilidad de orientar las hojas de estilo a medios de presentación específicos. Para ello se emplea el atributo **media** del elemento **link** del cual ya viste un ejemplo en la presentación del apartado anterior.

La siguiente tabla muestra los valores que puede tomar el atributo **media**:

- **all** Todos los medios definidos.
- **braille** Dispositivos táctiles que emplean el sistema Braille.
- **embossed** Impresoras que emplean el sistema Braille.
- **handheld** Dispositivos de mano: móviles, [PDA](#), etcétera.
- **print** Impresoras y navegadores en el modo "vista previa para imprimir".
- **projection** Proyector y dispositivos para presentaciones.
- **screen** Pantallas de ordenador.
- **speech** Sintetizadores para navegadores de voz empleados por personas discapacitadas.
- **tty** Dispositivos textuales limitados, como teletipos y terminales de texto.
- **tv** Televisores y dispositivos con resolución baja.

En el ejemplo siguiente se muestra cómo se pueden emplear en la función **@media** de la misma forma que hacíamos con la función **@import**.

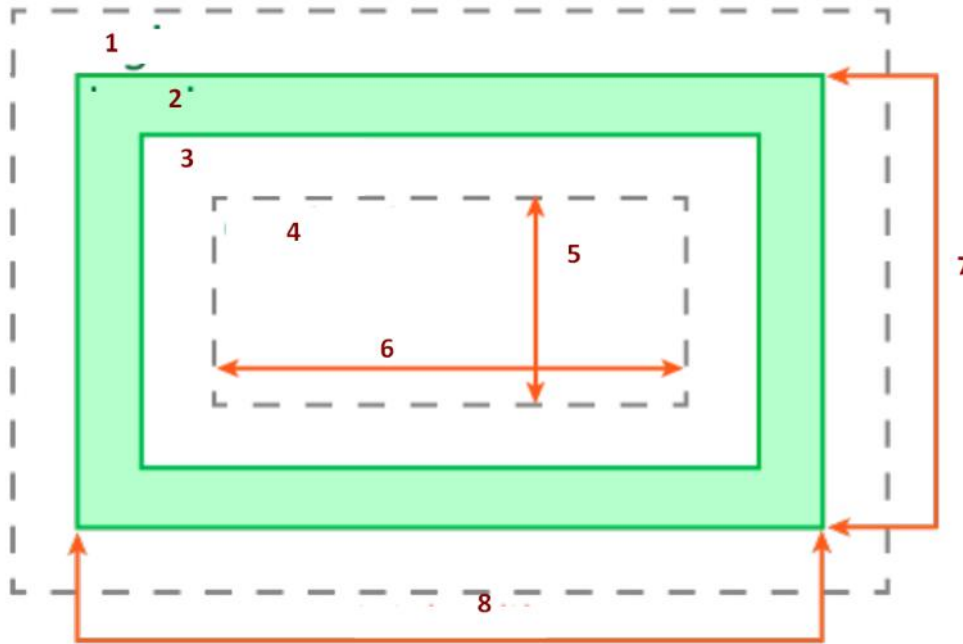
```
<style type="text/css">  
<!--  
@import url(http://estilos/miestilo.css);  
@media print {  
body { font-size: 10pt; } /* Establece el tamaño de fuente para impresión */  
}  
@media screen {  
body { font-size: 13px; } /* Establece el tamaño de fuente para visualización */  
}  
p {font-face: Verdana;}  
-->  
</style>
```

Practica 04-01

Ejercicio 1. Crea un documento htm en el que pongas los tres tipos de insertar css (línea, incrustado, link y @import) para poner los h1 en rojo , h2 en verde y p en azul

Ejercicio 2. Incrusta esta imagen en la pagina y Crea esta tabla en css con los conceptos

1	2	3	4	5	6	7	8



Ejercicio 3. Copia esta tabla en el html. Señala verdadero o falso

Afirmacion	Verdadero / Falso
El content va fuera del padding	
El padding va dentro del margin	
El border va entre el margin y el padding	
el margin va entre el padding y el content	
El border va fuera del padding	
width total = width + paddings left right + borders r l	
height total = height + paddings + borders + margins	

Ejercicio 4. Usa el anterior ejemplo Crea un CSS que muestre la Estructura y herencia y Especificidad del selector y Orden de las reglas. Se debe ver estas prioridades

- Hojas de estilo externas vinculadas (con link en el Head).
- Hojas de estilo externas importadas (con @import dentro del elemento style en la Head).
- Hojas de estilo incrustadas (con el elemento style en la Head).
- Estilos en línea (con el atributo style en la etiqueta del elemento).
- Declaraciones de estilo marcadas como !important.
- Especificidad del selector (ejemplo div h1)
- Orden de las reglas (h3 h3 h3)

Ejercicio 5. Inserta Comentarios en el anterior para ver que tiene mas o menos prioridad.

Ejercicio 6. Crea los siguientes bordes usando css puedes usar esta estructura

```
div.nueve {
  border: 7px ;
  border-style: inset;
  border-color: #FFFF00 #FF00FF #00FFFF forestgreen;
  background-color: #ffffff;
  color: #0000FF ;
}
<div class="nueve"> <br><h3>Bordes 9. 7px inset .</h3><br></div><br>
```

Bordes 1.

Bordes 2.

Bordes 3. dashed

Bordes 4. ridge

Bordes 6. dotted

Bordes 7. dashed

Bordes 8. outset

Bordes 9. 7px inset .

Bordes 10. 10px; groove;

Ejercicio 7. Modifica el anterior ejercicio para que los ejercicios 1 a 4 queden con estos márgenes (usa auto)

- Borde1 : alineado al centro
- Borde2 : alineado a la derecha
- Borde3 : alineado a la izquierda
- Borde4 : a 100 px de la izquierda

Ejercicio 8. Modifica el anterior ejercicio para que los ejercicios 1 a 4 queden con estos paddings

- Borde5 : padding todos 20px
- Borde6 : padding verticales 40 px horizontales 100 px
- Borde7 : paddings 10 20 30 40 px



Diseño de Interfaces Web

Ciclo de Grado Superior Desarrollo de Aplicaciones Web
IES Playamar

Ejercicio 9. Modifica el ejemplo de @media para crear un estilo diferente cuando sea móvil, por pantalla y para imprimir.

En pantalla el p debe verse verde móvil el p debe verse rojo y en print debe verse azul.