

# DISEÑO DE INTERFACES WEB

Docente: Luis Úbeda

Autor: Marc Salom



# UNIDAD 3: USO DE ESTILOS. INTRODUCCIÓN.

- Sección 1: Introducción a CSS
  - Selectores
  - Cascada
  - Herencia
  - Especificidad
  - Modelo de Caja
  - Floats
  - Display



# S1: INTRODUCCIÓN A CSS

- CSS significa Cascading Style Sheets, y es el modo en que añadimos color, layout, estilo y diseño a nuestras páginas web.
- Podemos decir que:
  - HTML = describir el contenido dentro de una web.
  - CSS = describe la presentación.
- Para referenciar el documento CSS dentro de la página html, dentro del <head>:  
`<link rel="stylesheet" href=css/estilo.css">`



# S1: INTRODUCCIÓN A CSS

- Si quisiéramos un elemento `<h1>` dotarlo de color naranja indicariamos en el documento CSS la siguiente regla:

```
h1 {  
    color: orange;  
    text-align: center;  
}
```

- De este modo, indicamos que elemento html, en este caso, los elementos `<h1>` de la página html, se verían afectados por esta regla.
- Dentro de los corchetes, tenemos las propiedad color, text-align con un valor naranja y center, respectivamente.
- La combinación de una propiedad con un valor se le llama declaración.



# S1: INTRODUCCIÓN A CSS

- Selectores en CSS:

1. Selectores de tipo: afecta a todo elemento que se encuentra en la web.

2. Selectores descendientes:

```
header p {  
    color: green;  
}
```

```
header p span {  
    color: blue;  
}
```



# S1: INTRODUCCIÓN A CSS

## 3. Selectores de clase

```
.highlight {  
    background-color: yellow;  
}
```

Nota: También se puede aplicar el carácter descendiente utilizando un elemento html y una clase que lo contiene:

```
footer .highlight {  
    background-color: LightGray;  
}
```

(decide más el CSS en este ejemplo que solamente la clase)



# S1: INTRODUCCIÓN A CSS

## 4. Efecto cascada en CSS:

- Como desarrolladores web podemos ir variando con reglas CSS cómo afecta el código según se va aplicando de arriba abajo.
- El navegador es el primer elemento que impone reglas en base a un código HTML y tiene unos valores por defecto.
- Después, con CSS lo podemos modificar adecuando el estilo.



# S1: INTRODUCCIÓN A CSS

## 5. Herencia

- Se puede modificar el estilo de elementos hijos al especificar el estilo del elemento padre.

- Tomando el elemento body como referencia:

```
body {  
    color:green;  
}
```



# S1: INTRODUCCIÓN A CSS

## 5. Especificidad

La especificidad de una regla determina si se va a aplicar el estilo o no.

Ejemplo:

```
p {  
    color:green;  
}  
footer {  
    color:orange;  
}
```

(no se aplica si dentro del footer hay elementos <p>

Para que se aplique: footer p {color:orange;}

Por lo que apreciamos footer p es más específico que p.



# S1: INTRODUCCIÓN A CSS

## 5. Especificidad

Reglas básicas a tener en cuenta:



1. Body footer p es más específico que footer p.
2. Especificar una clase tiene más prioridad que la regla anterior.
3. Especificar un elemento añadiendo a continuación una clase es más específico que indicar solamente la clase.
4. Si dos reglas tienen la misma especificidad, se aplica la última aplicada en el documento CSS.



# S1: INTRODUCCIÓN A CSS

## 6. Modelo de Caja

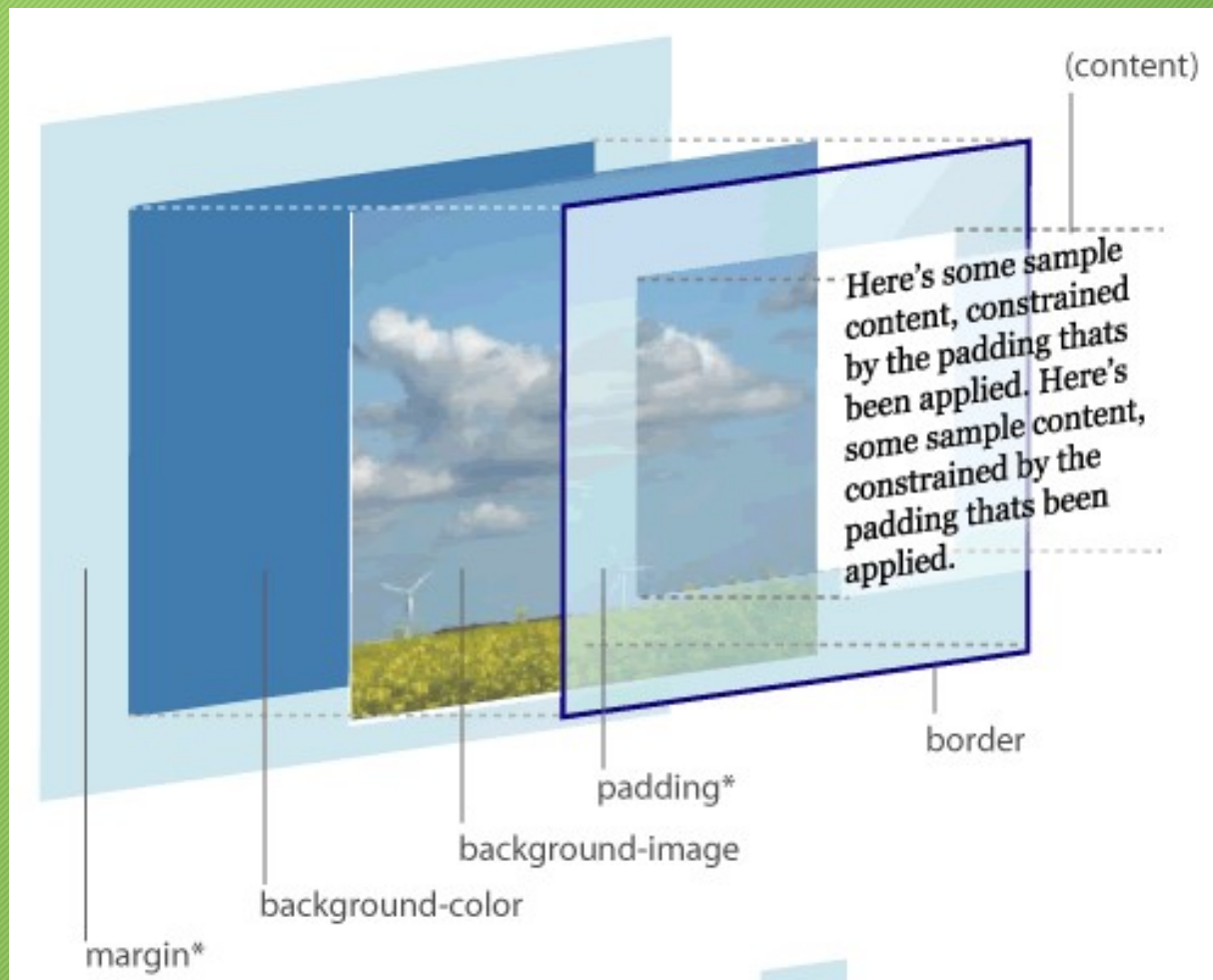
- En HTML todo está formado por cajas
- Una caja es un área que toma un cierto espacio de la página web.
- Elementos que usamos en CSS para crear espacio en una caja:
  - Padding: espacio dentro de la caja
  - Border: crear un marco en la caja
  - Margin: crear espacio fuera de la caja

```
.caja-a {  
  background-color: pink;  
  padding: 30px;  
  border: 3px solid black;  
  margin-bottom: 10px;  
}
```



# S1: INTRODUCCIÓN A CSS

## 6. Modelo de caja







# S1: INTRODUCCIÓN A CSS

## 6. Modelo de Caja

- Si además, queremos darle una anchura y altura a la caja, tenemos los elementos `width` y `height` en CSS.

```
.caja-a {  
    background-color:  pink;  
    padding: 30px;  
    border: 3px solid  black;  
    margin-bottom: 10px;  
    height: 150px;  
    width: 200px;  
}
```



# S1: INTRODUCCIÓN A CSS

## 6. Modelo de Caja

- Tenemos una propiedad en CSS que nos permite aunque añadamos margin y border a una caja, si aplicamos en CSS:

`box-sizing: border-box;`

*“No aumentará el tamaño de la caja a más de lo indicado por la anchura (width) y altura (height)”*

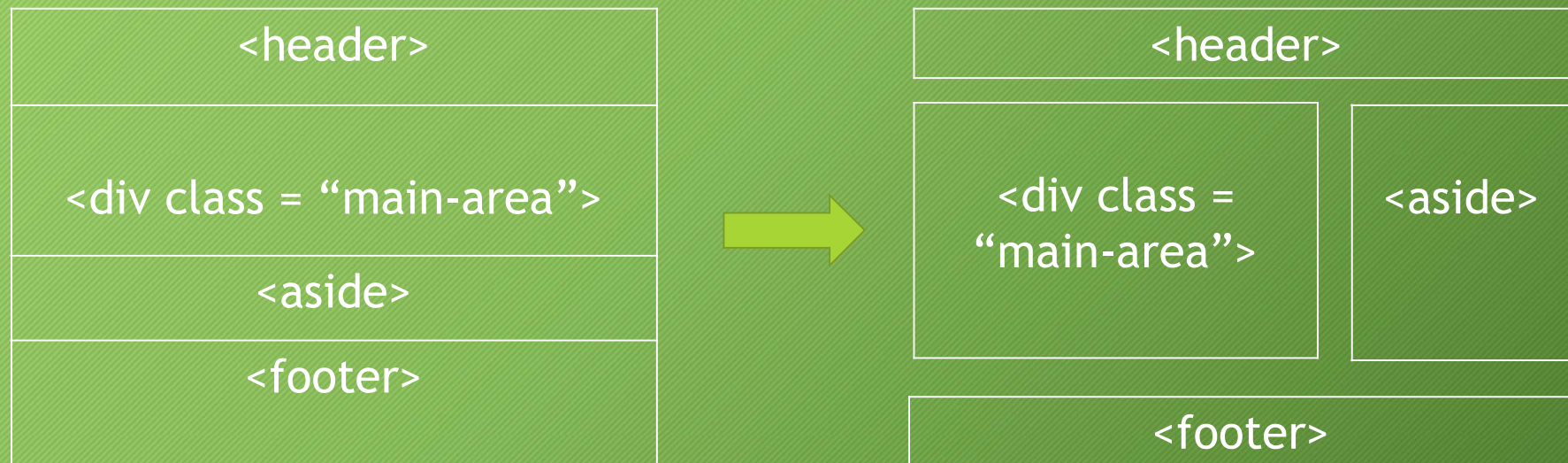
- *Atención. Si no aplicas ese comando, el width y el height será lo que ocupará el elemento contenido del modelo de caja.*



# S1: INTRODUCCIÓN A CSS

## 7. Organizar cajas usando Floats en CSS

- Hasta ahora hemos visto elementos situados uno encima del otro, tomando el espacio disponible que nos permite el navegador.



- `.main-area { float: left;}`



# S1: INTRODUCCIÓN A CSS

## 7. Organizar cajas usando Floats en CSS

Si queremos que el efecto float que hace que desaparezca un elemento del orden normal de ejecución no se aplique, se puede usar esta clase que le llamo grupo (como se podría llamar de otro modo) y le aplico la siguiente regla en CSS:

```
.grupo:before,  
.grupo:after{  
    content: "";  
    display: table;  
}  
.grupo:after {  
    clear: both;  
}  
.grupo {  
    zoom: 1;  
}
```



# S1: INTRODUCCIÓN A CSS

## 8. Organizar cajas usando la propiedad display en CSS

La propiedad CSS display especifica si un elemento es tratado como none, block or inline, o inline-block.

- Block: un elemento block se muestra como si fuera un elemento block, valga la redundancia y siempre tiene algo de espacio por encima y por debajo, que lo separa de los demás.
- Inline: se comporta como una caja independiente y no tiene en cuenta los elementos que tiene alrededor. Es decir, no crea espacios sino que se superpone si así lo impone su diseño.
- Inline-block: elemento aparece en línea pero se comporta como un elemento en block (en cuanto a los espacios).
- None: se usa para ocultar un elemento.



# S1: INTRODUCCIÓN A CSS

Referencia de w3schools a la propiedad DISPLAY:

[https://www.w3schools.com/cssref/tryit.php?filename=trycss\\_display](https://www.w3schools.com/cssref/tryit.php?filename=trycss_display)

## The display Property

### display: none:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

### display: inline:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. **HELLO WORLD!** Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

### display: block:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit.

**HELLO WORLD!**

Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

### display: inline-block:

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Etiam semper diam at erat pulvinar, at pulvinar felis blandit. **HELLO WORLD!** Vestibulum volutpat tellus diam, consequat gravida libero rhoncus ut.

He añadido un border para que veáis como trata los espacios el display inline (se superpone) y el display block (crea espacios).