

CSS

¿Qué es? ¿De dónde viene? ¿Cuál es su utilidad?

Origen del lenguaje CSS

- En los orígenes de la Web, los estilos estaban muy acoplados al propio documento HTML
- Existían elementos como `<center>` o ``, y atributos como ***background***, ***bghcolor*** o ***border***, que se aplicaban directamente en el propio documento HTML
- En webs pequeñas, esto era un problema relativamente pequeño y manejable, pero a medida que Internet crecía se convirtió en un problema acuciante, puesto que cambios de estilo en webs grandes suponía tener que modificar archivo HTML por archivo HTML, lo cual es ineficiente, laborioso y proclive a errores.

Origen del lenguaje CSS

- El **W3C** decidió que era hora ya de desacoplar la estructura de la Web, el HTML, de los estilos visuales que se le aplicasen, y para ello propuso la creación de un lenguaje de hojas de estilos específico para el lenguaje HTML.
- Se presentaron nueve propuestas, y finalmente se tuvieron en consideración dos: **CHSS** (***C**ascading **H**TML **S**tyle **S**heets*) de Håkon Wium Lie y la **SSP** (***S**tream-based **S**tyle **S**heet **P**roposal*) de Bert Bos.

Origen del lenguaje CSS

- Entre 1994 y 1995 ambos creadores de las dos propuestas **CHSS** y **SSP**, unieron fuerzas escogiendo lo mejor de cada propuesta y crearon el lenguaje llamado **CSS**.
- En 1995 el **W3C** decidió apostar por el desarrollo y estandarización de esta tecnología y la añadió al grupo de trabajo de **HTML**, publicando a finales de 1996 la primera recomendación oficial, conocida como **CSS Nivel 1**.
- En Mayo de 1998 se publicó la segunda recomendación oficial, conocida como **CSS Nivel 2**, llegando a refinamientos que condujeron a la recomendación **CSS 2.1** cuya última versión de esa recomendación data de Junio de 2011.
- Los primeros borradores de **CSS 3** datan de Junio de 1999 y su primera recomendación data de 2011.

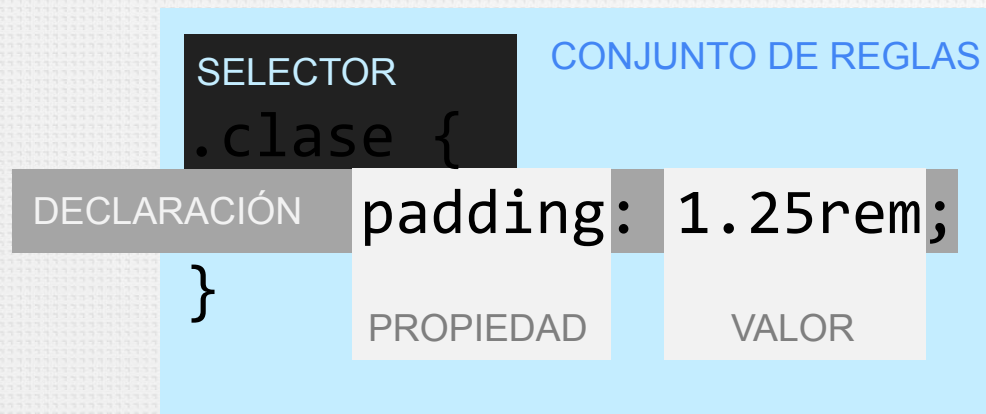
¿Qué es CSS?

- **CSS** es el acrónimo de **Cascade Style Sheet** (Hoja de Estilos en Cascada)
- Son hojas de estilo que definen mediante su sintáxis cómo se van a disponer en la pantalla los elementos **HTML** y qué aspecto visual van a tener.
- Se interpretan en el modo top-down, de principio a fin.
- **NO** es un lenguaje de programación, pues carece de estructuras de control. Podría decirse que es un lenguaje declarativo y no imperativo; decimos qué tiene que hacerse, pero no cómo hacerlo.
- Su objetivo es el de separar de una forma clara y definitiva la estructura de un documento **HTML** y sus estilos visuales, desacoplando completamente estos dos aspectos. De esta forma podemos estilar con una misma hoja de estilos diferentes documentos **HTML**, facilitando enormemente la tarea de desarrollo y su posterior mantenimiento.

Sintaxis CSS

Cómo escribir CSS y qué errores evitar

Sintaxis CSS



Un conjunto de reglas consiste en un selector y una serie de declaraciones (propiedad - valor) encerradas entre llaves.

Sintaxis CSS

- **Manejo de los selectores.**

- Los selectores en **CSS** se leen de derecha a izquierda, siendo el elemento seleccionado el que aparece al final de la línea.

.principal ul li

- Aquí estamos seleccionando un elemento `` que es hijo de un elemento `` y a su vez es hijo de un elemento con la clase **principal**.

Sintaxis CSS

- Todos los conjuntos de reglas que le queramos aplicar a un selector debemos encerrarlas entre llaves.

BIEN

```
.clase {  
    padding: 1.25rem;  
}
```

MAL

```
.clase {  
    padding: 1.25rem;  
}
```

MAL

```
.clase  
    padding: 1.25rem;  
}
```

MAL

```
.clase  
    padding: 1.25rem;
```

Sintaxis CSS

- **Manejo de las llaves en los conjuntos de reglas.**
 - Si olvidamos la llave del principio, CSS seguirá leyendo la siguiente porción de texto como si fuese parte del selector. Después probablemente encontrará un carácter ';' que es inválido en el nombre de un selector, y se romperá dejando de funcionar correctamente. Hará que este conjunto de reglas y el inmediatamente siguiente dejen de funcionar y de afectar a nuestros estilos, recuperándose después.
 - Si olvidamos la llave de cierre es aun peor ya que probablemente acabe arruinando el resto del **CSS** a partir de ese conjunto de reglas mal cerrado, a no ser que casualmente se encuentre una doble llave de cierre.
 - Es importante abrir y cerrar correctamente las llaves en los conjuntos de reglas que vayamos especificando, para no romper nuestros estilos.

Sintaxis CSS

- Manejo de los espacios.

- Hay que tener cuidado con las declaraciones, porque **poner o quitar espacios puede hacer que signifiquen cosas diferentes.**

≠ .principal ul li
.principalulli

- La primera línea selecciona el `` dentro de un `` y dentro de un elemento con la clase **principal**.
- La segunda línea selecciona un elemento con la clase **principalulli**.

≠ .cabecera .titulo
.cabecera.titulo

- La primera línea selecciona un elemento con la clase **titulo** dentro de un elemento con la clase **cabecera**.
- La segunda línea selecciona un elemento con la clase **cabecera** que además lleva la clase **titulo**.

Sintaxis CSS

- Manejo de los espacios.

- No se pueden poner espacios en las propiedades o en cualquier lugar donde se nombren cosas. Agregar espacios en esas situaciones cambia efectivamente los nombres, haciendo que dejen de funcionar.

`background-image: url(imagen.jpg);`

`background   image: url(imagen.jpg);`

- Aparte de estos dos supuestos, poner espacios en **CSS** no rompe las cosas, pero conviene ser pulcro y ordenado al escribirlo, tabulándolo convenientemente para facilitar su legibilidad.

`header{padding: 20px;} header { padding: 20px; }`

Sintaxis CSS

- Manejo de los puntos y coma.

- Cada declaración dentro de un conjunto de reglas **CSS** debe terminar en un punto y coma ';'
- Ese punto y coma es obligatorio, de lo contrario CSS seguirá leyendo la siguiente propiedad como si fuera parte del valor de la declaración anterior.

BIEN

```
.header {  
    padding: 20px;  
    max-width: 600px;  
    background: black;  
    color: white;  
}
```

MAL

```
.header {  
    padding: 20px;  
    max-width: 600px  
    background: black;  
    color: white;  
}
```

Sintaxis CSS

- **Manejo de los caracteres perdidos.**

- Esto es importante en cualquier lenguaje de programación. Cada carácter en el código importa, por lo que si dejamos caracteres sin control dentro del código, acabarán por romper las cosas.

BIEN

```
.header {  
    padding: 20px;  
}
```


MAL

```
.header {  
    /padding: 20px;  
}
```

Sintaxis CSS

- **Manejo de las rupturas de línea.**

- Podemos romper las líneas en CSS siempre que queramos mientras no rompamos las normas anteriormente descritas aunque, una vez más, hay que procurar ser ordenado y limpio.

<pre>.header { padding: 20px; }</pre>		<pre>.header { padding: 20px; }</pre>
---	---	---