CSS
Cascading Style Sheets

HTML

HTML + CSS

Vs.

STRUCTURE

PRESENTATION

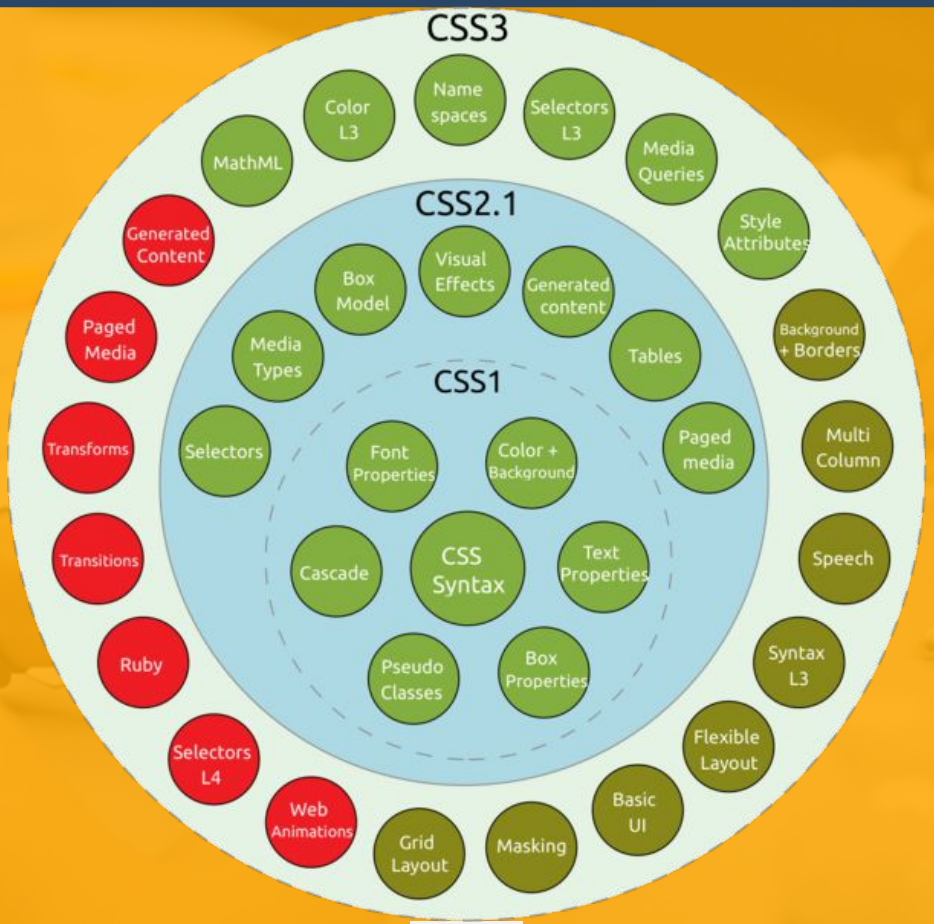
HTML y CSS

Hojas de Estilo en Cascada

- Permite controlar la apariencia de un sitio web.
- Permite definir un estilo visual para cada uno de los elementos de HTML.
- Separa los contenidos de su presentación:
  - Desarrollo paralelo.
  - Flexibilidad de apariencia.

Evolución

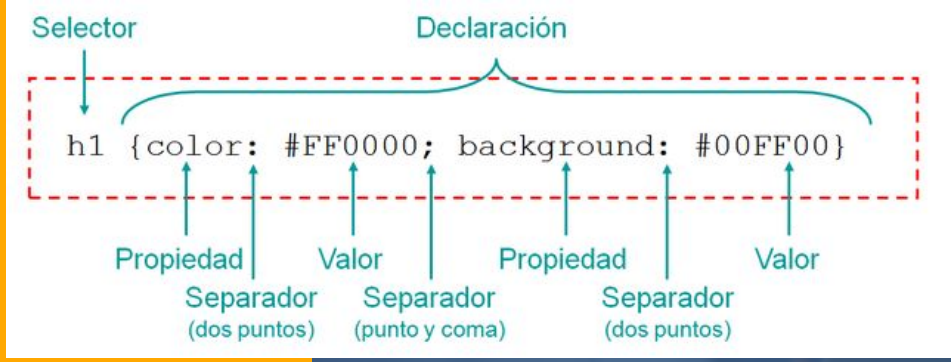
- Cada nivel de CSS se construye sobre el anterior, generalmente añadiendo funciones al previo.
- CSS 1.0: Publicada en diciembre 1996. Fuentes, colores, textos, propiedades de caja, id, listas.
- CSS 2.0 Mayo 1998. Posicionamiento absoluto/relativo, z-index, media types, nuevos selectores.
- CSS 3.0 Está dividido en documentos llamados “módulos”. Cada módulo añade nuevas funcionalidades. Nuevas propiedades para bordes, opacidad, sombras, etc.


CSS3
Name spaces
Selectors L3
Media Queries
Style Attributes
Background + Borders
Multi Column
Speech
Syntax L3
Flexible Layout
Basic UI
Masking
Grid Layout
Web Animations
Selectors L4
Ruby
Transitions
Transforms
Paged Media
Generated Content
MathML
Color L3
CSS2.1
Visual Effects
Generated content
Tables
Paged media
Text Properties
Box Properties
Pseudo Classes
Cascade
Selectors
Media Types
Box Model
CSS1
Font Properties
Color + Background
CSS Syntax

Ventajas

- Reutilización: se define la estética visual de los elementos una sola vez y se aplica a todas las páginas del sitio.
- Diseño unificado: todas las páginas del sitio web hacen referencia a un único estilo visual.
- Optimización de los tiempos de carga y de tráfico en el servidor: al dividir contenido y apariencia obtenemos archivos más ligeros, y esto reporta menores tiempos de carga y volumen de tráfico del servidor.

Sintaxis



- Cada conjunto de reglas CSS consta de dos partes: el selector y un bloque de declaraciones.

- El **selector** apunta al elemento HTML al que desea aplicar estilo.
- El **bloque de declaración** contiene una o más declaraciones separadas por punto y coma.
- Cada **declaración** incluye un nombre de propiedad CSS y un valor, separados por dos puntos.

Agregar CSS a un documento

- Cuando un navegador lee una hoja de estilo, dará formato al documento HTML de acuerdo con la información de la hoja de estilo cargada.
- Existen tres (3) formas de incluir una hoja de estilo dentro de un documento HTML:
  - En línea (inline)
  - Interno
  - Externo

CSS en línea

- Se puede usar un estilo en línea para aplicar un estilo único a un solo elemento.
- Para usar estilos en línea, se agrega el atributo de style (estilo) al elemento relevante.
- El atributo de estilo puede contener cualquier propiedad CSS.

```
<h1 style="color:blue;text-align:center;">Titulo</h1>
<p style="color:red;">Texto del parrafo</p>
```

DEMO

CSS Interno

- Se puede utilizar una hoja de estilo interna si una sola página HTML tiene un estilo único.
- El estilo interno se define dentro de la etiqueta <style>, dentro de la sección de cabecera <head>.

```
<head>
  <style>
    body { background-color: linen; }
    h1 { color: maroon; margin-left: 40px; }
  </style>
</head>
```

DEMO

CSS Externo

- Con una hoja de estilo externa, puede cambiar el aspecto de un sitio web completo cambiando solo un archivo.
- Cada página HTML debe incluir una referencia al archivo de hoja de estilo externo dentro de la etiqueta <link>, dentro de la sección del encabezado.
- El archivo externo no debe contener etiquetas HTML. Y debe guardarse con la extensión .css

```
<link rel="stylesheet" type="text/css" href="estilo.css">
```

DEMO

Selectores

- Los selectores CSS se utilizan para "buscar" (o seleccionar) los elementos HTML a los que se desea aplicar estilo.
- Podemos dividir los selectores de CSS en cinco categorías:
  - Simples (elemento, id y class)
  - Combinación (basados en relaciones)
  - Pseudo-clase (por estado)
  - Pseudo-elementos (parte de un elemento)
  - Por atributos (basados en un atributo o valor de atributo)

Juego (CSS Dinner)

Selectores

- Elemento

```
p {
  color: red;
}
```

- Id (#)

```
#parrafo1 {
  text-align: center;
  color: blue;
}
```

- Clase (.)

```
.centrado {
  text-align: center;
}
```

- Universal (*)

```
* {
  margin: 0;
}
```

- Agrupamiento (,)

```
h1, h2, h3 {
  text-decoration: underline;
}
```

- Pseudo-elementos (::)

```
p::first-line {
  color: #ff0000;
  font-variant: small-caps;
}
```

- Pseudo-clases (:)

```
p:hover {
  color: red;
}
```

- Combinaciones

```
p.azul {
  color: darkblue;
}
```

Selectores combinados

- Descendente ( ). **e** **f**. Selecciona todos los elementos de tipo **f** que están dentro de **e** sin importar la profundidad. No confundir con el agrupamiento (,).

```
p a {
  color: red;
}
```

```
<body>
 <p>
     Ingrese a <a href=”...”>enlace</a>. O también a
     <strong><a href=”...”>otro enlace</a></strong>
 </p> …
</body>
```

Selectores

Combinaciones

DEMO

/* Todos los elementos de tipo "p" con atributo class="aviso" */

p.aviso { ... }

```
<p class=”aviso”>SI</p>
<p class=”otra”>NO</p>
```

/* Todos los elementos "p" y todos los elementos con atributo class="aviso" de la página */

p, .aviso { ... }

```
<h1 class=”aviso”>SI</h1>
<h2>NO</h2>
<p><b class=”aviso”>SI</b></p>
<p>SI</p>
```

/* Todos los elementos con atributo class="aviso" que estén dentro de cualquier elemento de tipo "p" */

p .aviso { ... }

```
<p>
 <span class=”aviso”>
  SI
 </span>
</p>
<p>
 NO
 <span class=”otra”>
  NO
 </span>
</p>
```

¡Cuidado con las similitudes!

Selectores combinados

- Hijo (>). **e** > **f**. Selecciona todos los elementos de tipo **f** que son hijos directos de **e**.

```
p > a {
  color: red;
}
```

```
<body>
 <p>
     Ingrese a <a href=”...”>enlace</a>. O también a
     <strong><a href=”...”>otro enlace</a></strong>
 </p> …
</body>
```

Selectores

Combinaciones

DEMO

Selectores combinados

- Adyacente (+). **e** + **f**. Selecciona todos los elementos de tipo **f** que cumplan las dos siguientes condiciones:
    - **e** y **f** deben ser hermanos
    - **f** debe aparecer inmediatamente después de **e** en el código HTML

```
h1 + h2 {
  color: red;
  text-decoration: underline;
}
```

```
<body>
 <h1>Titulo1</h1>
 <h2>Subtítulo</h2> …
 <h2>Otro subtítulo</h2> …
</body>
```

Selectores

Combinaciones

DEMO

Selectores combinados

- Hermano general (~). **e** ~ **f**. Selecciona todos los elementos de tipo **f** que cumplan las dos siguientes condiciones:
    - **e** y **f** deben ser hermanos
    - **f** debe aparecer ~~inmediatamente~~ después de **e** en el código HTML

```
h1 ~ h2 {
  color: red;
  text-decoration: underline;
}
```

```
<body>
 <h1>Titulo1</h1>
 <h2>Subtítulo</h2> …
 <h2>Otro subtítulo</h2> …
</body>
```

Selectores

Combinaciones

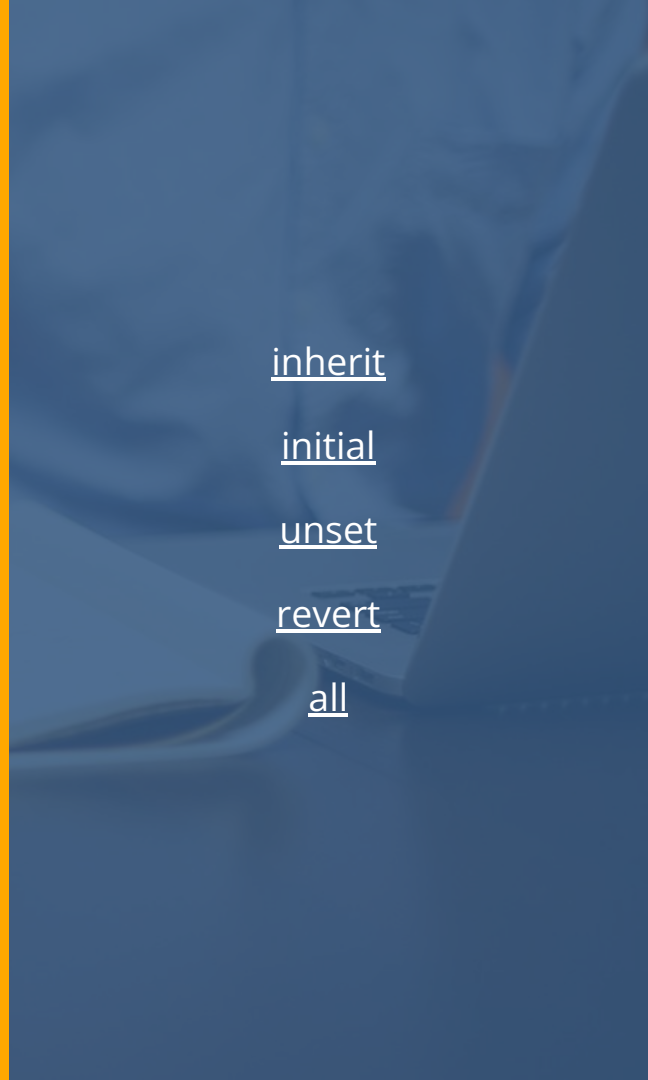
DEMO

Conflictos de reglas

- Cuando más de una regla se puede aplicar a un elemento hay un conflicto a resolver.
- Cascada: cuando dos reglas tienen la misma especificidad, se aplica la última.
- Especificidad: se aplica la que tenga el selector más específico.
- Herencia: algunos valores de propiedades se heredan del elemento padre. Otros no.
- Origen: es posible que el usuario configure un CSS para anular el estilo del desarrollador. Aunque esto está en desuso.

Herencia

- Existen 4 valores universales de propiedades para el control de la herencia. Todas las propiedades CSS aceptan estos valores.
- **inherit**: hereda el valor del padre.
- **initial**: en propiedades no heredadas, este valor refuerza el comportamiento por defecto y es necesario sólo para sobrescribir otra regla.
- **unset**: es inherit para propiedades heredadas e initial para no heredadas. Uso especial con “all:”.
- **revert**: revierte el valor al que habría tenido si no se hubiesen hecho cambios.



inherit

initial

unset

revert

all

Especificidad

- Mide cuán específico es un selector. Más específico, más prioridad.
- Se calcula de la siguiente manera:
  - a: 1 si la declaración de estilo es inline si no, 0
  - b: cantidad de atributos ID en el selector
  - c: cantidad de clases, pseudo-clases y otros atributos en el selector
  - d: cantidad de elementos y pseudo-elementos en el selector
- Luego se ordena un número con la forma: a,b,c,d.

```
*               {}  /* a=0 b=0 c=0 d=0 -> specificity = 0,0,0,0 */
li              {}  /* a=0 b=0 c=0 d=1 -> specificity = 0,0,0,1 */
li:first-line {}  /* a=0 b=0 c=0 d=2 -> specificity = 0,0,0,2 */
ul li           {}  /* a=0 b=0 c=0 d=2 -> specificity = 0,0,0,2 */
ul ol+li        {}  /* a=0 b=0 c=0 d=3 -> specificity = 0,0,0,3 */
h1 + *[rel=up]{}  /* a=0 b=0 c=1 d=1 -> specificity = 0,0,1,1 */
ul ol li.red  {}  /* a=0 b=0 c=1 d=3 -> specificity = 0,0,1,3 */
li.red.level  {}  /* a=0 b=0 c=2 d=1 -> specificity = 0,0,2,1 */
#x34y         {}  /* a=0 b=1 c=0 d=0 -> specificity = 0,1,0,0 */
style=""          /* a=1 b=0 c=0 d=0 -> specificity = 1,0,0,0 */
```

Cálculo de especificidades

Modificador !important

- Podemos declarar reglas de estilos como **!important** para que tomen precedencia sobre otras reglas de estilos.
- No usar a menos que sea absolutamente necesario ya que cambia el orden de aplicación de los estilos.

```
body{
 font-family: verdana, arial  !important;
}
```

Aplicación del CSS

- CSS del navegador.
- CSS del usuario.
- CSS del autor.
- Declaraciones **!important** en CSS del autor.
- Declaraciones **!important** en CSS del usuario.

- Si hay conflicto se resuelve por especificidad.
- Si persiste el conflicto, la última regla gana.

Fuente

- Hay 5 familias genéricas: serif, sans-serif, monospace, cursive y fantasy.
- **font-family** se usa para establecer una lista ordenada de fuentes que se usarán para mostrar un elemento. Usar al final una genérica como respaldo.
- **font-style** usado para hacer cursiva.
- **font-weight** indica la intensidad (negrita).
- **font-variant** permite pequeñas mayúsculas.
- **font-size** define el tamaño de la fuente.
- **font** es la propiedad abreviada para setear **style**, **variant**, **weight**, **size** y **family**.

Fuente

Style - Weight - Variant

Size

font

Google Fonts

Familia Genérica	Ejemplo Concreto
Serif	Times New Roman Georgia Droid Serif
Sans-serif	Arial Verdana Droid Sans
Monospace	Courier New Consolas Roboto Mono
Cursive	Caveat Pacifico
Fantasy	Special Elite Press Start 2P

Ejemplos de Fuentes

Texto

- **text-align** para alineación horizontal.
- **text-decoration** permite decorar el texto (enlaces).
- **text-transformation** transforma el texto a mayúsculas, minúsculas y “capitalize”.
- **text-indent** regula la sangría de la 1ª línea.
- **letter-spacing** y **word-spacing** definen el espacio entre letras y palabras respectivamente.
- **line-height** define la altura del renglón.
- **white-space** dice como tratar los espacios.
- **text-shadow** establece el efecto de sombra.

text-align

text-decoration

text-transformation

Espaciado

text-shadow

Colores

- Los colores pueden especificarse por nombre o su valor RGB, HEX, HSL, RGBA, HSLA.
- Algunos nombres de colores son red, green, blue, gold.
- RGB utiliza 3 valores para armar el color: red, green y blue. Con HEX se usa su valor hexadecimal. HSL permite establecer tono, saturación y luz. Todos son equivalentes.
- **color** es usado para colorear el texto de un elemento.
- **background-color** permite establecer el color de fondo de un elemento.

Colores

Listado de Colores

Fondo

- Además del color, se puede establecer una imagen o un degradado como fondo.
- **opacity** establece la opacidad. El canal alfa del color puede lograr efectos similares pero no es lo mismo.
- **background-image** permite establecer una imagen como fondo.
- **background-repeat** dice como se repite esa imagen.
- **background-position** ubica la imagen.
- **background-attachment** dice si la imagen queda fija o se mueve con el scroll.

background

background-image

background-repeat

background-attachment

background (Shorthand)

Pseudo clases y elementos

- Pueden usarse en CSS como selectores, pero no existen en el código fuente HTML.
- Son "insertados" bajo ciertas condiciones para que el CSS pueda referenciarlos.
- Son abstracciones que permiten referirse a elementos que de otro modo resulta imposible.
- Las pseudo clases (:) pueden asociarse a estados, como “hover” y “visited”, entre otros.
- Los pseudo elementos (::) pueden ver como partes de un elemento, como “first-letter” y “first-line”.

Pseudo Clases

Pseudo Elementos

Unidades de Medida

- CSS divide las unidades de medidas en 2 grupos.
- **Relativas:** definen su longitud en relación con otra medida (em, ex, px*, porcentaje).
- **Absolutas:** establecen de forma completa el valor de una medida (cm, mm, pt, in).
- Una medida se indica como un valor numérico seguido de una unidad de medida (10px, 5%).
- Unidades como cm o mm son útiles para imprimir.

Unidades

* Los píxeles (px) son relativos al dispositivo de visualización. Para dispositivos de bajo dpi, 1px es un píxel del dispositivo (punto) de la pantalla . Para impresoras y pantallas de alta resolución, 1px implica varios píxeles de dispositivo.

Unidad	Relativa a
em	Tamaño de letra del elemento padre, en el caso de propiedades tipográficas como font-size, y tamaño de la fuente del propio elemento en el caso de otras propiedades, como width.
ex	Altura x de la fuente del elemento.
ch	La medida de avance (ancho) del glifo "0" de la letra del elemento.
rem	Tamaño de la letra del elemento raíz.
lh	Altura de la línea del elemento.
vw	1% del ancho de la ventana gráfica.
vh	1% de la altura de la ventana gráfica.
vmin	1% de la dimensión más pequeña de la ventana gráfica.
vmax	1% de la dimensión más grande de la ventana gráfica.

Unidades Relativas

Las medidas relativas no se heredan tal cual por los elementos descendientes, sino que se propagan los valores calculados.

Frameworks

- Conjunto de librerías css que permiten facilitar las tareas comunes que realizan los desarrolladores css.
- Se componen de creación de grillas para facilitar la maquetación, conjunto de clases para tipografías, reseteadores de css, facilitan el diseño responsivo, etc.

Bootstrap

* Los reseteadores o normalizadores, proponen resetear o normalizar estilos que varían entre navegadores.

¡Gracias!

¿Preguntas?