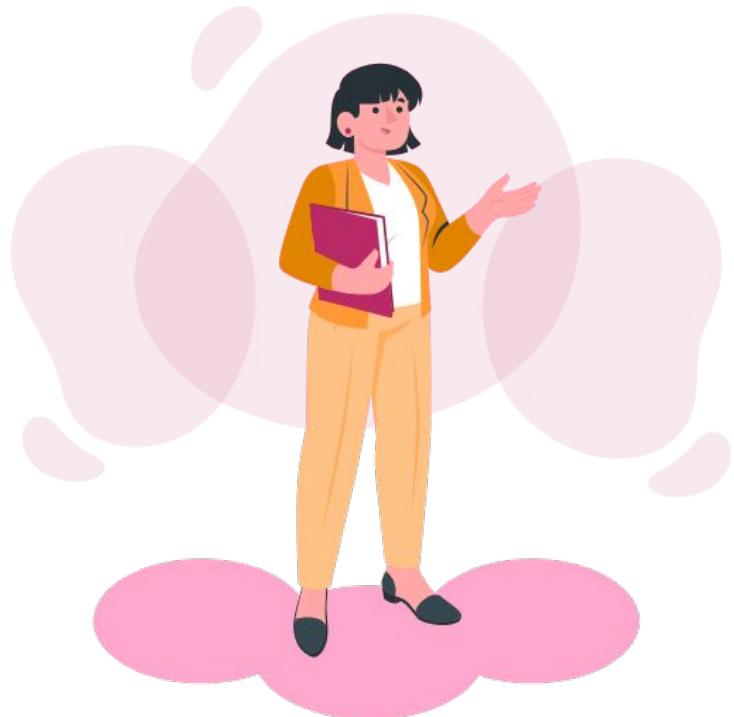


# Clase 4:

# Introducción a

# CSS (Parte 1)



# Introducción a CSS

# ¿Qué es CSS y para qué sirve?

- CSS significa Cascade Style Sheet o hoja de estilo en cascada y nos permite crear reglas que definen propiedades visuales para nuestros elementos.

```
span {  
    color: white;  
}
```

- En el código de ejemplo definimos que todos los elementos span de nuestro documento/s deben tener la tipografía de color blanco.
- De esta definición vemos que tenemos una forma de seleccionar elementos y que podemos aplicar una propiedad para modificar la forma en que se ve el contenido de la etiqueta seleccionada.

## ¿Qué es CSS y para qué sirve? - Continuación

```
selector {  
    nombrepropiedad: valor;  
}
```

- En CSS todo el tiempo vamos a estar utilizando estos dos conceptos:
  - \* Seleccionar elementos
  - \* Definir como queremos que se vean
- Según como utilizemos CSS podemos definir como se ve un elemento, un documento o todo nuestro sitio

### IMPORTANTE:

- Una herramienta muy útil a la hora de trabajar con CSS y JavaScript es la Devtools de Chrome

# ¿Cómo seleccionar elementos y definir el color de texto?

## Color de texto

- Por medio de la propiedad color podemos establecer el color de tipografía que tiene un elemento
- Esta propiedad color acepta un color como valor, dentro de las opciones puede ser:
  - \* Nombre de color en inglés (white, red, blue, yellow, gray, etc)
  - \* Color en formato RGB (`rgb(0,0,0)`) donde cada valor va de 0 a 255 siendo 0 el apagado y 255 prendido
  - \* Color Hexadecimal
- Para saber más pueden visitar el sitio de MDN

# Selector básico por tipo de elemento

- Utilizando el selector por tipo de elemento podemos agrupar la forma de aplicar estilos a los elementos según el nombre de su etiqueta
- Este es un selector general ya que va a seleccionar **TODOS** los elementos del mismo tipo

```
<p>Texto en color azul</p>
<p>Texto en color azul y <span>verde</span></p>
<div>Texto en color rojo</div>
<div>Texto en color rojo y <span>verde</span></div>
<span>Texto en color verde</span>
```

```
p {
    color: blue;
}

div {
    color: rgb(255, 0, 0);
}

span {
    color: #00FF00;
}
```

# Vinculación de HTML y CSS

# ¿Cómo agregar CSS al HTML?

Existen diferentes formas de aplicar CSS:

- Se puede aplicar a nivel elemento utilizando el atributo style de HTML
- En un documento por medio de la etiqueta style en el head
- Compartido entre varios documentos utilizando la etiqueta link en el head de nuestros documentos que queremos vincular

# Agregar hoja de estilo en un elemento (atributo style)

- La forma más particular e individual que tenemos de utilizar css es utilizando el atributo style que tienen los elementos HTML
- Dentro del atributo style escribimos las propiedades visuales separadas por punto y coma

```
<span style="color: white;">Texto en color blanco!!</span>
```

- De esta forma establecemos que un elemento span en particular va a tener el color de texto blanco

## Agregar hoja de estilo en un elemento (atributo style) - Continuación

- Si quiero que dos o más elementos tengan el mismo color lo hacemos de la siguiente manera:

```
<span style="color: white;">Texto en color blanco!!</span>
<span style="color: white;">Otro texto en color blanco!!</span>
```

- Genial, ya tenemos 2 span con texto en blanco, ahora me pregunto, si tengo 50 span en el documento, los tengo que modificar uno a uno?, La respuesta es NO y ya vamos a ver cómo lo hacemos
- Cuando definimos estilos a nivel elementos lo estamos haciendo de forma individual
- Esta es una buena opción cuando necesitamos que si o si un elemento se vea de una determinada forma
- Utilizando esta definición de css se hace más difícil mantener o modificar nuestros elementos ya que para cambiar el color de una tipografía (como en el ejemplo) vamos a tener que modificar uno a uno todos los elementos (mucho trabajo)
- Si tenemos tan solo un documento puede ser una tarea relativamente simple pero si tenemos 300 documentos se hace más complicado
- Para evitar este problema necesitamos otra alternativa que nos permita definir los atributos visuales de nuestros elementos a nivel documento

# Agregar hoja de estilo en un documento (style)

- Por medio de la etiqueta style podemos definir los estilos que queremos para nuestros elementos a nivel documento
- Utilizando selectores podemos cambiar la forma que se ve uno o muchos elementos del mismo tipo
- Para definir que el texto que estamos escribiendo dentro de la etiqueta style es css, utilizamos el atributo type con el valor "text/css"
- Si bien podemos omitir este atributo ya que HTML5 no lo require dejamos en manos del browser como interpretar el contenido de esta etiqueta
- Para estar más seguro y lograr mejor compatibilidad con browsers anteriores definimos el atributo type del elemento style

```
<!DOCTYPE html>
<html>
  <head>
    <title>Usando CSS</title>
    <style type="text/css">
      span {
        color: white;
      }
    </style>
  </head>
  <body>
    <span>Texto en color Blanco</span>
    <span>Texto en color Blanco</span>
    <span>Texto en color Blanco</span>
    <span>Texto en color Blanco</span>
    <span>Texto en color Blanco</span>
  </body>
</html>
```

# Agregar hoja de estilo en un documento (style) - Continuación

- Con tan solo un cambio podemos establecer que todos nuestros elementos span se vean de otro color
- Esta forma de utilizar los estilos está buena cuando necesitamos definir estilos para un documento
- Si bien esta forma es más general que definir los estilos en cada elemento, sigue siendo una individual a nivel documento
- Un sitio va a tener más de un documento y si queremos mantenerlo necesitamos una forma de poder unificar todos los estilos para todos nuestros documentos

```
<!DOCTYPE html>
<html>
  <head>
    <title>Usando CSS</title>
    <style type="text/css">
      span {
        color: red;
      }
    </style>
  </head>
  <body>
    <span>Texto en color Rojo</span>
    <span>Texto en color Rojo</span>
    <span>Texto en color Rojo</span>
    <span>Texto en color Rojo</span>
    <span>Texto en color Rojo</span>
  </body>
</html>
```





## Agregar hoja de estilo en un documento externo (link) - Continuación

- CSS se llama Hoja de estilo en cascada ya que los elementos pueden heredar propiedades visuales de padre a hijo
- Si establecemos estilos al body todos los elementos que esten en el documento van a heredar esos atributos
- No todos los elementos son heredables
- Una buena forma de ver esto es utilizando la barra de developer tools
- Los estilos que definimos en un archivo externos pueden ser sobreescritos utilizando la etiqueta styles
- Los estilos definidos en style pueden ser sobreescritos utilizando el atributo style de los elementos
- De esta forma podemos ver que utilizamos conceptos generales y son pisados por los más individuales
- Este mismo concepto se da entre elementos
- Si definimos atributos visuales para el body podemos sobreescribir como se ven los párrafos por ejemplo dejando que el resto de los elementos hereden las propiedades declaradas en el body

# Agregar hoja de estilo en un documento externo (link) - Continuación

- Como podemos ver en este ejemplo el estilo que establecemos en el elemento va a ser el que quede definido finalmente
- El estilo definido en el documento (etiqueta style) pisa el que esta definido en el archivo general. En este caso importa el orden en que fueron llamados los estilos
- Como regla podemos decir que siempre ponemos los estilos más generales primero y después sobrescribimos lo que necesitamos

```
styles.css

p {
    color: red;
}
```

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

<style type="text/css">
    p {
        color: blue;
    }
</style>

<p style="color: black;">Texto en color negro!!</p>
```

# Selectores

# Selectores

- Un selector es la forma en la que elegimos a que elementos queremos asignarle atributos visuales con CSS
- Los selectores pueden ser generales o particulares, es decir que puedo seleccionar varios elementos o tan solo uno y de forma muy particular
- Podemos utilizar un solo selector:

```
selector {  
    // código CSS  
    // código CSS  
    // código CSS  
}
```

# Selectores - Continuación

- Podemos utilizar más de un selector separados por coma:

```
selector1, selector2 {  
    // código CSS  
    // código CSS  
    // código CSS  
}
```

# Tipo de etiqueta

- Es el selector que venimos utilizando en todos los ejemplos anteriores
- El selector por tipo de etiqueta nos permite elegir elementos por el nombre de la etiqueta
- Este tipo de selector retorna una colección de elementos ya que selecciona todos los elementos con el mismo nombre de etiqueta
- Por ejemplo podemos seleccionar los elementos p y h1 y definir que el color de texto sea blanco

```
p {  
    color: white;  
}  
  
h1 {  
    color: white;  
}
```

# Tipo de etiqueta – Continuación

- Dado que ambos elementos se van a ver de la misma forma podemos escribir los selectores de la siguiente forma:

```
p, h1 {  
    color: white;  
}
```

# Selector universal

- Por medio del selector asterisco (\*) podemos seleccionar todos los elementos
- Este selector se utiliza muchas veces para borrar estilos que vienen predefinidos en el browser

```
* {  
    color: white;  
}
```

# Selector por clase

- Los elementos HTML tiene un atributo class que nos permite asignarles un nombre de clase
- En CSS podemos seleccionar los elementos por nombre de clase utilizando un punto y el nombre de la clase
- Este selector afecta a todos los elementos que tengan el nombre de clase seleccionado

En CSS:

```
.rojo {  
    color: red;  
}
```

En HTML:

```
<p class="rojo">Este texto es ROJO</p>
```

# Selector por clase - Continuación

- Podemos hacer un selector más particular si queremos seleccionar sólo algunos elementos que tengan una determinada clase
- Si queremos seleccionar sólo los títulos H1 que tienen una clase rojo podemos hacerlo de la siguiente forma:

En CSS:

```
h1.rojo {  
    color: red;  
}
```

En HTML:

```
<h1 class="rojo">Este texto es ROJO</h1>  
<p class="rojo">Este texto es NEGRO</p>
```

- Este último selector sólo selecciona los elementos del tipo h1 que tienen la clase rojo

# Selector por ID

- Los elementos HTML tienen un atributo ID que nos permite seleccionarlos de forma única
- Con CSS podemos seleccionar los elementos por ID utilizando el símbolo de numeral (#) y el nombre del ID del elemento
- Este es un selector individual ya que solo modifica un solo elemento
- Si queremos seleccionar el elemento que tiene el ID principal y establecer el texto en azul lo podemos hacer de la siguiente manera:

En CSS:

```
#principal {  
    color: blue;  
}
```

En HTML:

```
<div id="principal">Contenido de mi div en AZUL</div>
```

# Selector descendiente

- El selector descendiente permite seleccionar cualquier elemento
- Ahora que sabemos varios selectores podemos ver más propiedades de CSS
- Existen más selectores y los vamos a seguir viendo más adelante

[Juego para aprender mas sobre selectores](#)

# Colores

# Colores

Los colores en las computadoras están compuestos por 3 colores básicos llamados RGB que vienen de Red, Green y Blue, es decir Rojo, Verde y Azul

Los monitores están compuestos de píxeles que son como pequeñas luces que pueden mostrar estos colores

Cuando la luz está apagada obtenemos el color negro

Cuando la luz está en su máxima intensidad en todos los colores obtenemos el color blanco

Existen diferentes sistemas para especificar un color:

- Colores por nombre
- RGB
- HEXA
- HUE

Tabla de colores web HTML, Colores Hexadecimales, Colores RGB:  
<https://www.kreatibu.com/app/tabla-de-colores-web-html-colores-hexadecimales-colores-rgb/>

## Nombre de color

- Existen nombres de colores que podemos utilizar directamente y el browser sabe como mostrarlos
- La lista de colores que podemos utilizar no es muy extensa por lo cual nos limita a la hora de elegir un color
- El nombre del color es en inglés (ejemplo: white/blanco)
- Son fáciles de utilizar
- En cada nueva versión de CSS se agregan más colores
- Ejemplos de colores: white, silver, gray, black, red, blue, green, orange, pink, brown and yellow

```
body { color: black; }

p { color: gray; }

a { color: orange; }
```

[Lista de colores según versión de CSS](#)

# RGB

- Para este color utilizamos la función de CSS llamada rgb
- Podemos asignar un valor entre 0 y 255 a cada color
- Si todos los valores son 0 obtenemos el color negro
- Si todos los valores son 255 obtenemos el color blanco
- El orden de los colores es el especificado en el nombre, es decir que el primer valor es para rojo, el segundo para verde y el tercero para azul

## Ejemplos:

- **rojo**: `rgb(255, 0, 0)`
- **verde**: `rgb(0, 255, 0)`
- **azul**: `rgb(0, 0, 255)`

```
body {  
    /* color negro */  
    color: rgb(0, 0, 0);  
}  
  
p {  
    /* color verde */  
    color: rgb(0, 255, 0);  
}
```

# RGB – Continuación

- Existe otra forma de utilizar rgb en CSS y es por medio de la función rgba() que agrega un canal más de transparencia
- Podemos asignar un valor entre 0 y 1 para el nivel de transparencia
- Se asigna como cuarto parámetro de la función

```
body {  
    /* color rojo */  
    color: rgba(255, 0, 0, 1);  
}  
  
p {  
    /* color rojo más transparente */  
    color: rgba(255, 0, 0, 0.4);  
}
```

# HEXA

- El sistema de HEXA representa los colores en Hexadecimal
- El color arranca con un numeral (#)
- Tenemos valores del 0 a la letra F
- Los números van del 0 al 9
- Continúan desde la letra A hasta la F
- Asignamos un par de letras para cada color del RGB
- Los dos primeros caracteres son para el rojo
- Los dos del medio son para el verde
- Los dos últimos son para el azul
- Si todos los valores son 0 tenemos el color negro (todo apagado)
- Si todos los valores son F tenemos el color blanco (todo prendido)

## Ejemplos:

- blanco: #FFFFFF
- negro: #000000
- rojo: #FF0000
- verde: #00FF00
- azul: #0000FF

```
body {  
    color: #FFFFFF;  
}  
  
p {  
    color: #FF0000;  
}
```

# HEXA – Continuación

- En caso de que las dos letras del mismo color se repitan podemos utilizar una sola, por ejemplo para blanco puede ser #FFF o negro #000

```
body {  
    color: #FFF;  
}
```

# HSL

- Este sistema hace referencia a Hue, Saturation and Intensity, es decir Matiz, Saturación e Intensidad
- Podemos establecer un valor entre 0 y 360 para el HUE según el color que queremos HUE
- Tanto saturación como intensidad aceptan un valor que va de 0 a 100%

## Ejemplo:

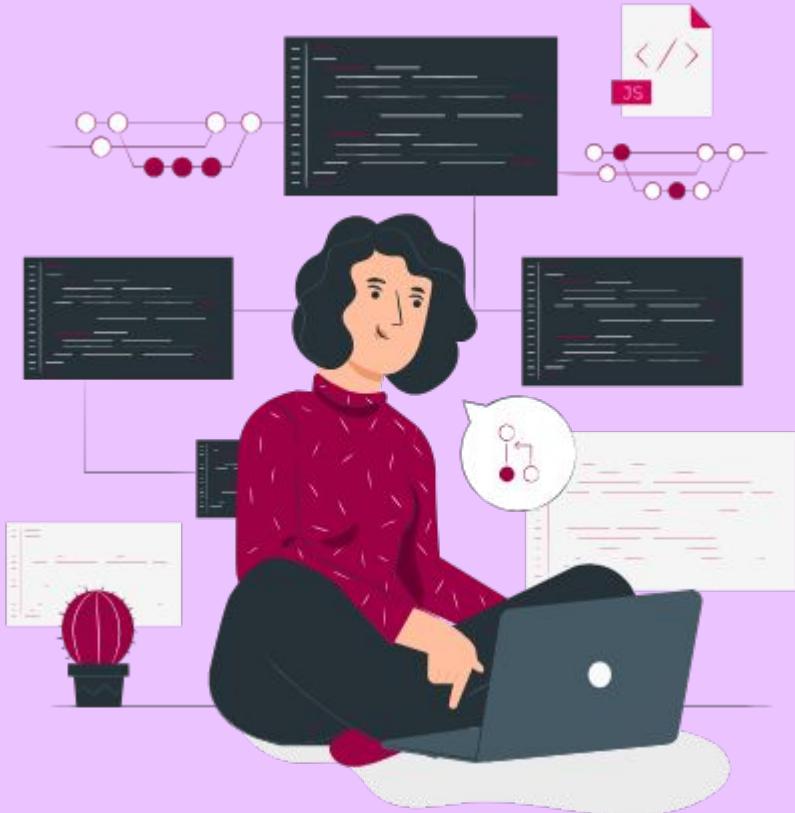
- **negro**: `hsl(0, 0%, 0%)`
- **blanco**: `hsl(0, 0%, 100%)`
- **rojo**: `hsl(0, 100%, 50%)`

# HSL

- En este sistema también contamos la función con alpha que funciona de la misma manera que el rgba.
- El cuarto valor acepta un número entre 0 y 1 para el nivel de transparencia que queremos
- Ejemplo: gris con transparencia: hsla(0, 100%, 100%, 0.5);}

[Calculador de HSL](#) [Más sobre HUE](#) [Más sobre HSL](#) [Teoría del color](#)

# Veamos un ejemplo de este tema



Ahora es  
momento  
de aplicar lo  
aprendido en  
clase



# Ejercitación

- <https://github.com/Ada-IT/ejercicios-frontend/blob/master/modulo-1/ejercicios/05-introduccion-a-css.md>