



Fundamentos del desarrollo web

1.3  
MANEJANDO  
HOJAS DE  
ESTILO

1.3

# MANEJANDO HOJAS DE ESTILO

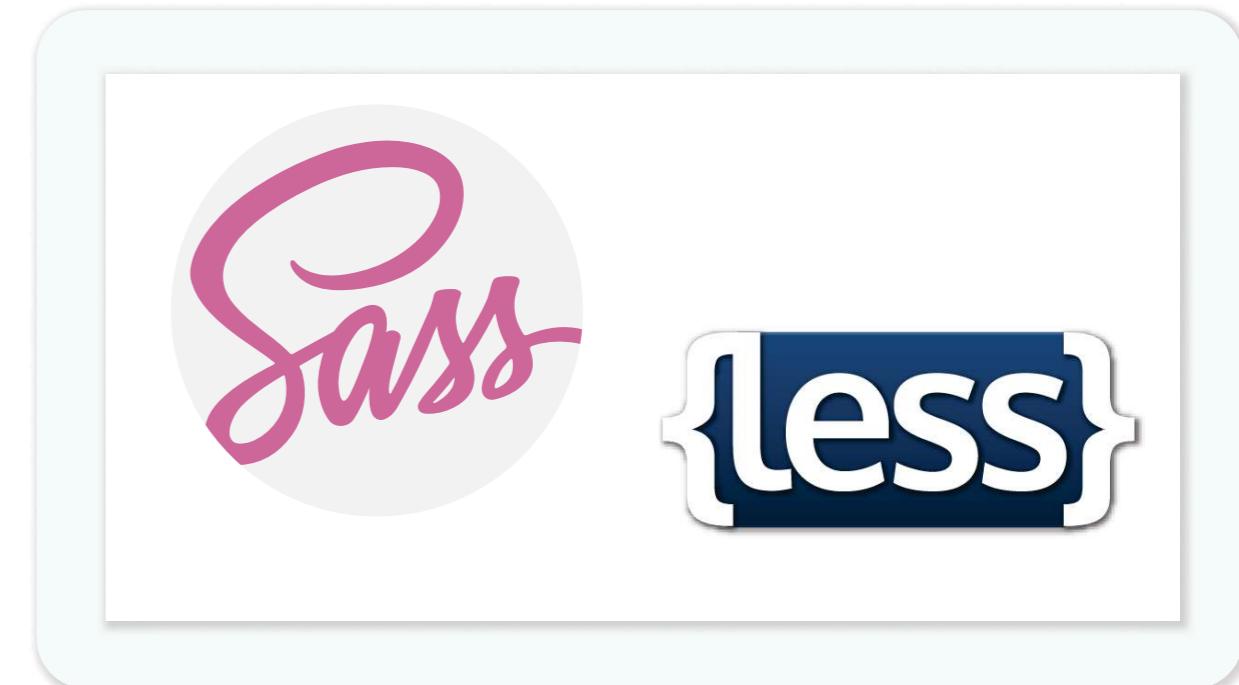
## Aprendizaje esperado:

Podrás modificar el estilo del contenido utilizando propiedades de CSS para la definición de aspectos visuales de la interfaz web y entenderás el posicionamiento de elementos utilizando CSS para la definición de aspectos visuales de la interfaz web. Comprendiendo lo anterior, serás capaz de describir el orden jerárquico de reglas de CSS para la definición de aspectos visuales de la interfaz web y finalmente tendrás la capacidad de aplicar y explicar el border-box-model reconociendo sus propiedades modificables para la definición de aspectos visuales de la interfaz web.

## Contenidos del submódulo:

- 1.3.1 ¿Qué es una hoja de estilo y para qué sirve?
- 1.3.2 Sintaxis básica de una hoja estilo
- 1.3.3 Selectores y estructura avanzada css
- 1.3.4 Manejo de assets e imágenes
- 1.3.5 Rutas absolutas y relativas en css
- 1.3.6 Buenas prácticas, orden jerárquico y pesos de las reglas
- 1.3.7 El inspector de elementos

Contenido adicional al contenido formativo propuesto por SENCE



Aprender CSS es lo básico para poder presentar nuestros documentos con el formato y diseño adecuado para el usuario. Hoy en día los desarrolladores front end y los diseñadores UI manejan frameworks que hacen más fácil la mantenimiento de las hojas de estilo. Los más populares son SASS <https://sass-lang.com> y LESS <http://lesscss.org>

## 1.3.1

# ¿Qué es una hoja de estilo y para qué sirve?

## ¿Qué es css?

Las hojas de estilo en cascada o Cascade Style Sheets (CSS) son un lenguaje de maquetación web que nos permite estilizar los componentes html para que se vean armónicos y también para que sean más fáciles de usar. Los sitios web partieron como un instrumento de divulgación de contenidos, es por eso que los componentes web html no tienen una apariencia pulida y acorde a nuestros tiempos. Fueron pensados como elementos clickeables y para pc de escritorio. La diversificación de dispositivos que acceden a internet y también la popularización de la red, trajo consigo distintos objetivos a los sitios web y css vino a ayudar a estilizar debidamente estos cambios.

La intención principal detrás de las hojas de estilo en cascada CSS (siglas en inglés de Cascading Stylesheets) es separar el código a través del cual se definen los contenidos de una página web, del que se escribe para su presentación o aspecto visual. Tal como hemos visto en secciones anteriores, el lenguaje HTML se utiliza para estructurar el contenido de nuestra página web desde el punto de vista semántico (títulos, subtítulos, texto, etc.). CSS es la tecnología que usamos para la estética del mismo o, más concretamente, cómo deben ser desplegados cada uno de los elementos de un documento HTML. Este principio es fundamental por muchos motivos. Un ejemplo claro es el “diseño adaptativo” (responsive design): poder adaptar el mismo contenido a diferentes dispositivos. Es decir, que una misma página web se puede visualizar de una manera

en un PC que un móvil, optimizada para cada caso[1].

Desde los inicios del W3C, el concepto y la importancia de separar contenidos de estilos fue determinando la confección de CSS. Éste ha pasado por etapas o versiones que han sufrido actualizaciones como en el caso de otros estándares.

La última versión anterior a CSS3, que es el que actualmente utilizamos, fue CSS2.1, una revisión de la especificación CSS2 original de 1997. Muchos se sorprenden al saber que CSS2 sólo se transformó en una recomendación oficial del W3C en 2011. Además, CSS3 comenzó a discutirse en 1998, un año luego de la aparición de CSS2. CSS2.1 fue una revisión que W3C realizó en base a la forma real en cómo había sido implementado en realidad en el mercado, que había evolucionado a un estado de muchas inconsistencias entre navegadores en términos de la implementación de CSS2. La creciente carrera de competencia entre navegadores de internet en la última década a favorecido la estandarización de CSS3 que anteriormente, cuando Internet Explorer dominaba el mercado con su implementación CSS2.1 y se resistía a la introducción de CSS3 está estandarizado en documentos individuales que siguen su propio camino y han dado una mejor dinámica al estandar que un solo documento centralizado como fue el caso de CSS2. En la actualidad, la especificación CSS3 es un estandar de la industria en todos los navegadores, posee una gran comunidad de desarrolladores construyendo aplicaciones que éste[2].



1.3.1

# ¿Qué es una hoja de estilo y para qué sirve?

Las principales características que presenta CSS3 como mejoras de las versiones anteriores son[3]:

- Soporte de funciones orientadas a diseño responsivo.
- Código modular.
- Bordes redondeados.
- Bordes para imágenes.
- Sombras de texto.
- Imágenes de fondo múltiples.
- Mayor velocidad de carga.
- Transformaciones 2D/3D para animaciones y transiciones nativas sin necesidad de Flash o Javascript.
- Nuevos colores y efectos de imagen.

## Las clases

las clases son un atributo de las etiquetas html que nos permite

las clases son un atributo de las etiquetas html que nos permite modificar la apariencia original del componente a través de la hoja de estilos css. Acá vemos un ejemplo de como estilizar un componente button añadiendo la clase "mi\_boton" como atributo y luego en la hoja de estilos o archivo .css añadimos propiedades de diseño seguido de un valor y separadas por ;

Código HTML

```
<button class="mi_boton">
  hazme click!
</button>
```

Resultado en pantalla

Código CSS

```
.mi_boton {
  width: 100%;
  color: #2F72E7;
  background-color: #32D1D5;
}
```



Fundamentos del desarrollo web

1.3  
MANEJANDO  
HOJAS DE  
ESTILO

 AWAKELAB

### 1.3.2

## Sintaxis básica de una hoja de estilo

### Formato básico de una hoja de estilos:

```

1. mi_boton {
    width: 100%;
    color: #2F72E7;
    background-color: #32D1D5;
}

.mi_boton > .mi_icono {
    color: pink;
}

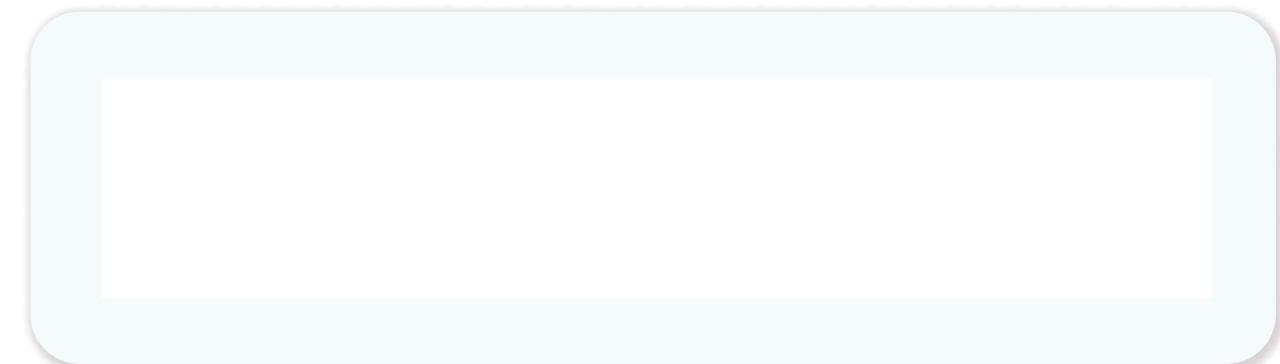
```

1. SELECTOR DE CLASE ( . Y NOMBRE)  
2. CURLY BRACE { (ABRE LAS PROPIEDADES DE CLASE)  
3. NOMBRE DE LA PROPIEDAD (DEBE LLEVAR NOMBRE Y : )  
4. VALOR DE LA PROPIEDAD (SE CIERRA CON ; )  
5. SELECTOR DE HERENCIA (>)  
6. CURLY BRACE } (CIERRA LAS PROPIEDADES DE CLASE)

### Los elementos básicos CSS: reglas, selectores y propiedades

Las hojas de estilo constan, normalmente, de una serie de instrucciones de estilo que definen cómo se han de representar determinados elementos de la página. Los elementos de página se identifican con las etiquetas HTML, identificadores, clases y otros selectores.

Una instrucción o regla CSS tiene la siguiente estructura:



Estas instrucciones o reglas nos sirven para definir son en el fondo las definiciones de visualización estética de nuestro documento. Estamos familiarizados a realizar este tipo de definiciones en los típicos procesadores de texto, como Libre Office Write o Microsoft Word, a través de sólo algunos clics. La diferencia es que en este caso lo hacemos de manera literal y en un lenguaje especial (CSS) con el que es posible dar instrucciones similares a la máquina.

A través de estas reglas podemos definir cosas como las siguientes:

- “El título principal de cada página (

# ) debe estar centrado, ser de color gris oscuro y usar la fuente Montserrat con un tamaño de 40 pixeles”.



## 1.3.2

## Sintaxis básica de una hoja de estilo

- “Los subtítulos de cada página de nivel dos (`<h2>`) deben estar alineados a la izquierda, color negro ligeramente más claro y usar también el tipo de letra Montserrat, pero con un tamaño de 32 pixeles”.
- “Las imágenes dentro de la columna principal deben expandirse siempre al ancho máximo de dicha columna.”

Estos son ejemplos a modo ilustrativo y pasamos a continuación a traducir las instrucciones anteriores a lenguaje CSS, las que tendrían el siguiente contenido[1]:

### Reglas

Este extracto de hoja de estilos se divide en 3 bloques diferenciados, agrupados en llaves “`{...}`”. Cada uno de estos grupos es una regla e implementa una directiva concreta de la pequeña lista que expusimos de una manera redactada un poco más arriba.

### Selectores

Para especificar a qué partes de la página HTML se aplica cada regla en cuestión, le precede un selector. Este selector especifica el ámbito de aplicación de la regla. Recordemos que un documento HTML está organizado de manera jerárquica a modo de árbol. Este ámbito de aplicación van a ser una o varias ramas de ese árbol.

En las dos primeras reglas (selectores “`h1`” y “`h2`”) el ámbito de aplicación es muy amplio porque los selectores son etiquetas HTML. Es decir, estamos diciendo que la primera regla se aplique a todos los elementos que utilicen la etiqueta `<h1>` y que la segunda se aplique a todos elementos `<h2>`.

El último selector es más específico y nos dice que la tercera regla se aplica solamente a elementos `<img>` que sean hijos de un elemento `<div>`.



## 1.3.2

# Sintaxis básica de una hoja de estilo

es más específico y nos dice que la tercera regla se aplica solamente a elementos `<img>` que sean hijos de un elemento `<div>`. Esto se hace con la posición de los elementos, es decir, al poner “img” a la derecha de “div” estamos diciendo que img debe tener un `<div>` como padre. Pero, además, con el sufijo de “.contenido” estamos definiendo además que el “div” no puede ser cualquiera sino que debe pertenecer a la clase “contenido”. Esto es útil para diferenciar distintos bloques `<div>`, por ejemplo usarlo para diferenciar la columna principal de contenido de otra columna para una barra lateral.

Podríamos haber hecho lo mismo con cualquier otro elemento HTML. En general, las clases (atributo “class” en HTML) son muy útiles para tratar los mismos elementos HTML de una manera diferenciada según su lugar y función en la página. Otro atributo comúnmente utilizado es “id”, con el cuál identificamos el elemento en particular, por ejemplo si tenemos un `<div id="contenido_1">`, el “id” nos serviría para referenciar específicamente ese div en particular. En este caso en lugar de utilizar “.contenido” en el selector se usa “#contenido\_1”, donde “#” hace referencia a que estamos refiriéndonos a un “id”.

## Propiedades

Tal como ya hemos mencionado, una regla puede tener una serie de

declaraciones internas en una sintaxis `propiedad:valor`, por ejemplo, `font-size: 40px;` que establece que a los elementos que caen dentro del ámbito de la regla, según el selector, deben poseer un tamaño de fuente de 40 píxeles. Los elementos deben ser por tanto elementos tipo texto, es decir, elementos `<h1>`, `<h2>`, `<p>`, u otro que podemos encontrar en la sección donde anteriormente hemos tratado las etiquetas disponibles para texto en HTML. De todas formas, en caso que exista algún elemento dentro del ámbito de la regla que no sea tipo texto, la propiedad será ignorada.

## Formas de implementar css

Como hemos dicho, el uso de CSS nos permite aplicar estilos a páginas web para que se vean exactamente con el estilo visual que deseamos. Esto es posible porque CSS está conectado al DOM para que pueda rediseñar rápida y fácilmente cualquier elemento. Por ejemplo, si no deseamos utilizar el aspecto predeterminado de `<h1>`, `<h2>` y otras etiquetas de encabezado, podemos asignar nuevos estilos para anular la configuración predeterminada de la familia de fuentes y el tamaño utilizado, o en negrita o cursiva debe establecerse, además de muchas otras propiedades. Hemos creado archivos HTML en secciones anteriores de este módulo, donde incluso hemos de manera básica el uso de CSS con un elemento `<style>` dentro del elemento `<head>` del documento HTML. En la práctica, existen diversas formas de incluir el código CSS en nuestro desarrollo y a continuación los revisaremos, partiendo por la forma que ya hemos utilizado[4].

## 1.3.2

# Sintaxis básica de una hoja de estilo

## Como elemento <style> dentro del mismo archivo HTML (Estilos Incrustados)

La primera forma que revisaremos para aplicar estilos a nuestra página web, es agregar el código CSS requerido dentro de el elemento `<head>` de nuestro código HTML. Por lo tanto, como hemos aplicado en códigos previos del curso, si quisiéramos cambiar el estilo de todos los elementos de etiqueta `<h1>` que estuvieran presentes en la página, utilizaríamos el siguiente código dentro de `<head>`:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <meta name='robots' content='index, follow'>
    <title>The style tag</title>
    <style>
      h1 {
        color:olive;
        font-size :36pt;
        font-family:'Times New Roman';
        font-style :italic;
      }
    </style>
  </head>
  <body>
    <h1>Este es un Encabezado de Tipo 1</h1>
  </body>
</html>
```

## Importando una hoja de estilos

Cuando deseamos aplicar estilos a un sitio web completo y no solamente a una página web individual, una mejor forma de administrar hojas de estilo es extraerlas completamente del código HTML y colocarlas en archivos separados para luego importar los que se requieren en las diversas páginas. Esto permite usar diferentes hojas de estilo para distintos layouts (Como web o impresión) sin necesidad de alterar el código HTML.

### 1. @import

Una forma de llevar a cabo lo descrito es utilizando la directiva `@import` de CSS como se muestra a continuación:

```
<style>
  @import url('styles.css');
</style>
```

De esta forma instruimos al navegador a obtener la hoja de estilo de nombre `styles.css`. El comando `@import` entrega la flexibilidad de crear hojas de estilo que a su vez importan otras hojas de estilos



Fundamentos del desarrollo web

### 1.3 MANEJANDO HOJAS DE ESTILO

#### 1.3.2

## Sintaxis básica de una hoja de estilo

Es importante enfatizar que las etiquetas `<style>` no deben estar incluidas dentro de ningún archivo CSS.

### 2. Elemento `<link>`

También podemos incluir una hoja de estilos CSS con la etiqueta `<link>` de HTML como sigue:

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

Este método tiene el mismo efecto que la directiva `@import`, con la excepción que `<link>` es una etiqueta sólo utilizable en HTML, por lo que no puede ser utilizada desde dentro de un archivo CSS. Nótese además que `<link>` no debe ir incluido dentro de etiquetas `<style>...</style>`.

### Como atributo `style` de un elemento HTML (Estilo Inline)

Es posible definir o sobreescribir ciertos estilos para la página web de manera caso a caso, insertando atributos de estilo directamente en un elemento HTML, tal como se muestra a continuación:

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

Sin embargo, este método debe ser excepcional dado que quiebra la separación entre contenido y presentación, que es una de las ventajas principales de la definición de estilos.

## 1

Fundamentos del desarrollo web

1.3  
MANEJANDO  
HOJAS DE  
ESTILO

 AWAKELAB

1.3.3

## Selectores y estructura avanzada CSS

Como vimos más arriba, el manejo de estilos se realiza a través de un listado de instrucciones o reglas, compuestas por selectores, declaraciones y pares propiedad:valor.

Anteriormente vimos algunos ejemplos ilustrativos de reglas CSS de acuerdo a algunos objetivos básicos de diseño. En los mencionados utilizamos algunos selectores simples del tipo etiqueta HTML, tales como `<h1>`, `<h2>` o `<div>`. De la misma forma incluimos un conjunto muy limitado de propiedades. En la práctica existen múltiples formas de especificar un selector, así como también una infinidad de propiedades CSS.

A continuación haremos un resumen de herramientas imprescindibles que serán suficientes para llevar a cabo una infinidad de personalizaciones de presentación de contenidos en páginas web que desarrollemos[4].

### Modificar apariencia a través de selectores

Existen una gran variedad de selectores. Tal como hemos visto hasta el momento, es posible seleccionar por tipo de elemento HTML, tal como `<p>`, `<div>`, `<img>`, etc. Sin embargo, además podemos seleccionar elementos bajo otros criterios, tales como: Elementos que estén contenidos dentro de otros (Descendientes), elementos que sean hijos directos y por el método muy usado de atributo de elementos HTML: "id", "class", "target" y otros. CSS está

tan bien pensado que para la selección de lo que queramos personalizar, y su relación con cualquier otro elemento, podemos encontrar los selectores para los resultados deseados.

**Nota:** No nos referiremos mayormente a las propiedades mostradas en estos ejemplos de selectores.

#### 1. Selector por tipo de elemento HTML.

Hemos visto varios ejemplos de este tipo, y consiste específicamente en indicar el tipo de elemento HTML según su etiqueta. Por ejemplo:

```
p { text-align:justify; }  
img { border:1px solid #444; }  
h1 { font-size:26pt; }
```

#### 2. Selector por descendencia.

Los selectores por descendencia permiten aplicar estilos a elementos que estén contenidos dentro de otro elemento. Por ejemplo, las siguientes reglas establecen que: 1) El color de todos los textos dentro de los elementos `<b>` sean de color rojo, pero

## 1

Fundamentos del desarrollo web

### 1.3 MANEJANDO HOJAS DE ESTILO

## 1.3.3

## Selectores y estructura avanzada CSS

solamente si la aparición de esos textos ocurren dentro de un elemento **<p>** (Por ejemplo: **<p><b>Hola</b>Mundo</p>**), y 2) Que el color del texto será azul cuando esté en negrita, siempre que esté dentro de un elemento de lista, y siempre que además esté dentro de una lista sin orden:

```
p b { color:red; }  
ul li b { color:blue; }
```

### 3. Selector por calidad de hijo.

Especifica elementos que sólo sean hijos directos de otro elemento. Si consideramos el ejemplo anterior, de texto rojo en negrita, podemos acotarlo especificando que sólo aplicará la regla si **<b>** es hijo directo del elemento **<p>**. Por lo tanto, la regla aplicará en este caso: **<p><b>Hola</b>Mundo</p>**, pero no en **<p><i><b>Hola</b>Mundo</i></p>**: La sintaxis para esto es la siguiente:

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

### 4. Selector por ID.

En caso de existir un elemento HTML con un id determinado, como por ejemplo: **<div id = 'mydiv'>**, éste puede ser seleccionado directamente desde CSS a través del símbolo **#** de la siguiente forma, para cambiar el texto a itálico:

```
#mydiv { font-style:italic; }
```

Las ids son para trabajar con secciones de un documento que solo aparecerán una sola vez. Esto dado que corresponde a un atributo único dentro del DOM.

Es posible acotar el alcance del selector por id, por ejemplo, para que aplique sólo si el elemento es del tipo **<p>**, con **p#mydiv**. Esto podría ser confuso a primera vista, dado que id es un atributo único. Sin embargo, en sitios dinámicos esto puede ser útil, cuando el id puede eventualmente ser asignado a distintos tipos de elementos dependiendo de alguna condición al momento de construir la página en el servidor.

## 1.3.3

# Selectores y estructura avanzada CSS

## 5. Selector por Clase.

Cuando existe un número de elementos en una página que se desean personalizar con los mismos estilos, puede asignárseles un nombre de clase, por ejemplo: `<span class="myclass">`. Luego puede crearse una única regla que modifique todos estos elementos a la vez. La siguiente regla crea un margen izquierdo de 10 pixeles para todos los elementos de la clase "myclass" que es precedida de ":" como sintaxis que alude a una clase. El igual que el selector por id, podemos acotar el alcance de la regla a un tipo de elemento, por ejemplo sólo a elementos del tipo `<p>` que pertenezcan a esa clase, con `p.myclass`.

```
.myclass { margin-left:10px; }
```

## 6. Selector por Atributo.

El selector por atributo nos permite seleccionar los elementos que contengan cierto valor asignado a alguno de sus atributos en el DOM. Por ejemplo, todos los elementos `<input type="submit">` del documento serían seleccionados con:

```
[type='submit'] { width:100px; }
```

También es posible acotar la selección, por ejemplo a elementos como este, pero que sólo estén dentro de un formulario, a través de: `form [type='submit']`.

Este tipo de selector también funciona con los atributos `id` y `class`, sin embargo el uso más común es a través de `#` y `:`, respectivamente, según se vio más arriba.

## 7. Selector Universal.

Existe el selector comodín `*` que significa seleccionar cualquier elemento. Su uso aislado provoca resultados que no son de mucha utilidad, como aplicar los estilos a todos los elementos de un documento. Sin embargo, combinándolo con otros selectores puede entregar opciones interesantes. Por ejemplo, seleccionar todos los elementos `<p>` dentro de elementos con `id="boxout"`, pero sólo si no son descendientes directos de `<p>`. Esto lo logramos con la siguiente instrucción o regla:



## 1.3.3

# Selectores y estructura avanzada CSS

```
#boxout * p {border:1px solid green; }
```

## 8. Selector por Grupo.

Los selectores pueden agruparse utilizando “” de la siguiente forma:

```
p, #idname, .classname {  
    border-bottom:1px dotted orange;  
}
```

En este caso todos los elementos del tipo **<p>**, los que tengan **id="idname"**, y los que pertenezcan a la clase **class="classname"** serán seleccionados para aplicarles los estilos especificados en la regla.

## Modificar apariencia a través de selectores

Es difícil recordar y tener en mente permanentemente el gran número de propiedades de **CSS3**. Hay referencias online muy útiles para consultar este tipo de información, tal como la Referencia CSS provista por Mozilla Developers Network (MDN)

[https://developer.mozilla.org/es/docs/Web/CSS/Referencia\\_CSS](https://developer.mozilla.org/es/docs/Web/CSS/Referencia_CSS). La buena noticia es que no hace falta usar permanentemente una gran parte de estas propiedades. En general se cumple la Ley de Pareto del **20/80**. Sabiendo una parte menor de ellas se puede lograr una habilidad suficiente para desempeñarse en el mundo del diseño Web. La documentación oficial, libros, tutoriales online y comunidades de desarrolladores siempre estarán disponibles para consultas sobre aplicaciones más específicas. De esta forma, la experiencia de uso logrará la expansión del conocimiento en la materia.

A continuación revisamos un listado de algunas propiedades que son de uso habitual y será de utilidad familiarizarse con ellas:

### 1. Textos.

- **font-family**: tipo de letra (nombre del tipo => “Montserrat”, “Open Sans”, etc.).
- **font-size**: tamaño de letra.
- **font-weight**: peso (normal, negrita, ...).

## 1.3.3

# Selectores y estructura avanzada CSS

```
#boxout * p {border:1px solid green; }
```

## 8. Selector por Grupo.

Los selectores pueden agruparse utilizando “” de la siguiente forma:

```
p, #idname, .classname {  
    border-bottom:1px dotted orange;  
}
```

En este caso todos los elementos del tipo **<p>**, los que tengan **id="idname"**, y los que pertenezcan a la clase **class="classname"** serán seleccionados para aplicarles los estilos especificados en la regla.

## Modificar apariencia a través de selectores

Es difícil recordar y tener en mente permanentemente el gran número de propiedades de **CSS3**. Hay referencias online muy útiles para consultar este tipo de información, tal como la Referencia CSS provista por Mozilla Developers Network (MDN)

[https://developer.mozilla.org/es/docs/Web/CSS/Referencia\\_CSS](https://developer.mozilla.org/es/docs/Web/CSS/Referencia_CSS). La buena noticia es que no hace falta usar permanentemente una gran parte de estas propiedades. En general se cumple la Ley de Pareto del **20/80**. Sabiendo una parte menor de ellas se puede lograr una habilidad suficiente para desempeñarse en el mundo del diseño Web. La documentación oficial, libros, tutoriales online y comunidades de desarrolladores siempre estarán disponibles para consultas sobre aplicaciones más específicas. De esta forma, la experiencia de uso logrará la expansión del conocimiento en la materia. A continuación revisamos un listado de algunas propiedades que son de uso habitual y será de utilidad familiarizarse con ellas:

### 1. Textos.

- **font-family**: tipo de letra (nombre del tipo => “Montserrat”, “Open Sans”, etc.).
- **font-size**: tamaño de letra.
- **font-weight**: peso (normal, negrita...).
- **font-style**: estilo (normal, cursiva, ...)



## 1.3.3

# Selectores y estructura avanzada CSS

- **letter-spacing:** espacio entre letras.
- **line-height:** espacio entre líneas / altura de la línea.
- **text-decoration:** cosas como subrayados, tachados, etc.
- **text-align:** alineación del texto (left, center, right).

En relación a tamaño de fuentes, éstos pueden estar especificados como dimensiones absolutas o relativas al contexto de la página. Las siguientes tablas son de utilidad para saber en qué casos recurrir a uno u otro modo. Generalmente en diseños responsivos se utilizan tamaños relativos de fuentes en los textos de la página.

## 2. Unidades

### Unidades absolutas para tamaño de fuentes

UNIDAD	DESCRIPCIÓN
in	Pulgadas - 1in equivale a 2.54cm.
cm	Centímetros
mm	Milímetros

UNIDAD	DESCRIPCIÓN
pt	Puntos - En CSS, un punto está definido como 1/72 pulgadas (0.353mm).
pc	picas - 1pc equivale a 12pt.
px	pixel - 1px equivale a 0.75pt.

### Unidades relativas para tamaño de fuentes

UNIDAD	DESCRIPCIÓN
em	Relativa al tamaño de la fuente del elemento (2em significa 2 veces el tamaño de la fuente actual)
ex	Relativa a la altura de la fuente actual
ch	Relativa al ancho del "0" (Cero)
rem	Relativa al tamaño de la fuente del elemento raíz
vw	Relativa al 1% del ancho del viewport
vh	Relativa al 1% del alto del viewport



## 1.3.3

# Selectores y estructura avanzada CSS

## 3. Colores y Fondos.

- **color:** color del elemento. Es importante notar que existen diferentes formatos para especificar el color como palabras predefinidas ("red", "green", etc.), formato RGB o valor hexadecimales.
- **background-color:** color del fondo del elemento.
- **background-image:** usar una imagen de fondo.
- **background-repeat:** usar una imagen de fondo como mosaico. Permite diferentes modos de organización de la imagen (ver detalles en material de referencia).
- **opacity:** opacidad del elemento. Va desde 0 (completamente transparente) hasta 1 (sólido). Un valor de 0.5 sería, por tanto, un nivel de transparencia del 50%.

## 4. Bordes

- **border:** añade un borde a un elemento y especifica sus propiedades (grosor, estilo de línea, etc.). Ver también el modelo de caja de arriba.

- **border-color:** color del borde.

- **border-style:** diferentes estilos (sólido, rayitas, puntos, etc.).
- **border-radius:** redondear las esquinas de un elemento.

## 4. Otras

- **float:** propiedad avanzada que permite una maquetación más sofisticada. Posicionar los elementos de manera "flotante". Ver este tutorial básico sobre posicionamiento flotante.
- **clear:** controla el comportamiento de los elementos adyacentes a elementos posicionados de forma flotante. Ver también el tutorial anterior sobre posicionamiento flotante.
- **overflow:** controla el comportamiento de los contenidos que no caben en su elemento contenedor (valores: visible, hidden, scroll, auto, inherit)
- **display:** controla diferentes aspectos de la visualización de un elemento, permite incluso ocultarlo con el valor "none".
- **list-style-image:** URL con una imagen que se debe usar como viñeta.



Fundamentos del desarrollo web

1.3  
MANEJANDO  
HOJAS DE  
ESTILO AWAKELAB

---

**1.3.3**

## Selectores y estructura avanzada CSS

---

- **list-style-type:** diferentes estilos de viñetas y numeración para los elementos de la lista.
- **box-shadow:** aplicar un efecto de sombra a un elemento.

---

**1.3.4**

## Manejo de assets e imágenes

---

Se le conoce como assets a los elementos que acompañan nuestros desarrollos pero que no son parte del código fuente sino que son elementos que se enlazan al código para que puedan ser mostradas en pantalla o ser disponibilizadas para su descarga. Dentro de los assets contamos con elementos como los archivos multimedia (videos, música), los documentos como los pdf, las webfonts y finalmente las imágenes. Si bien las imágenes son parte de la multimedia las separamos porque su uso y recomendaciones son más extensas que el uso de videos y música.

### Multimedia y documentos

Los elementos multimedia y documentos no pueden ser incluidos desde las hojas de estilo, sino que deben ser implementados dentro del html o la hoja de scripts (.js) es por eso que para este apartado es importante comprender que son assets, pero no son assets que sean llamados por css. Los documentos son enlazables vía el tag `<a>` y los videos y la música pueden ser enlazados pero también se pueden disponibilizar para su stream através del tag `<video>` y `<audio>`