Guía práctica para CSS

1. Inclusión de estilos

Existen diversas maneras de incluir estilos a un archivo HTML. La primera, es escribir los estilos directamente en el archivo en una etiqueta `<style>`. En esta etiqueta, se aplican los estilos única y exclusivamente al archivo donde son escritos. **No podrán ser utilizados en ningún otro archivo**. Usualmente, esta etiqueta se coloca dentro de la etiqueta `<head>`.

```
<head>
    <meta charset="UTF-8">
    <title>Página</title>

    <style type = "text/css">
        body {
            margin: 0;
            padding: 0;
        }

        h1 {
            color: #FFC805;
            font-family: Times New Roman, serif;
        }
    </style>
</head>
```

Otra manera de incluir estilos, y **la más recomendable**, es escribirlos dentro de un archivo tipo CSS externo y luego incluirlos en el archivo HTML por medio de una etiqueta `<link>`. Se pueden manejar varios archivos CSS, usualmente guardados dentro de una misma carpeta de estilos, ya sea para dar estilos a diferentes archivos, o a diferentes elementos o conjuntos de elementos (acomodo de elementos, fuentes, animaciones, etc.) o a diferentes páginas (inicio, contacto, galería, productos, etc.), pero se recomienda **evitar crear demasiados archivos**, ya que esto puede afectar al rendimiento del sitio en general. De la misma manera, se puede estilizar todo el sitio utilizando un solo archivo, que contenga alguna forma de organización dentro de este para mantener un orden. La importación se hace utilizando una etiqueta `<link>` por cada archivo CSS, y se indica el tipo de archivo a importar y la ruta de este dentro de la misma etiqueta.

```
<head>
    <meta charset="UTF-8">
    <title>Página</title>
    <link rel="stylesheet" href="css/styles.css">
    <link rel="stylesheet" href="css/responsive.css">
    <link rel="stylesheet" href="css/animate.css">
</head>
```

La siguiente manera de dar estilos, es directamente darlos a cada elemento dentro del archivo HTML. Esto se hace escribiendo el atributo `style` dentro de cualquier elemento HTML.

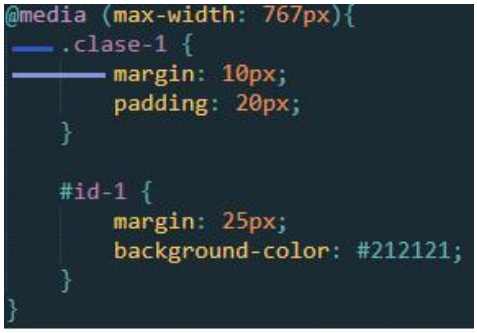
```
<p style = "color: blue;">Texto prueba</p>
```

Existen otras maneras de hacer importación de CSS, por ejemplo, por medio de llamadas a través de JavaScript o jQuery, pero **estas no se recomiendan**, debido a que requieren más tiempo para ejecutarse, lo cual resulta en un mayor tiempo de carga, y, por lo tanto, en una **página más lenta**.

2. Indentación

La indentación es y siempre será una de las herramientas más utilizadas por desarrolladores de todo tipo. Su principal utilidad es dar una **mejor estructura visual y organizacional** al código escrito. Facilita la navegación a lo largo de este a cualquier persona que lo lea.

La indentación más típica se realiza en capas, indentando una tabulación por capa. La indentación en CSS limpio, **no llega a más de 3 capas**, cuando se realiza la indentación con Media Queries. Existen diversos frameworks que implementan anidación más profunda de capas, tales como Sass o Less, que implementan variables, mixins, anidaciones internas, entre muchas otras cosas añadidas a la sintaxis original de CSS.



1 tabulación (2° capa)

2 tabulaciones (3° capa)

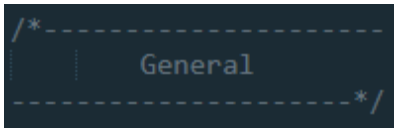
1 tabulación (2° capa)

2 tabulaciones (3° capa)

3 tabulaciones (4° capa)

3. Organización y estructura de escritura de estilos

Al escribir CSS, lo más recomendable es **contar con una estructura base comprensible** para facilitar la navegación a lo largo del archivo. La estructura básica recomendada es, realizar las divisiones principales de acuerdo con las secciones de la página que se está estilando, además de algunas secciones más generales, tales como fuentes, propiedades base de elementos generales, tales como elementos *div, section, a*, clases utilizadas a lo largo del documento, entre otros. Posteriormente, se pueden realizar subdivisiones dentro de estas si se quiere ser más específico. Para marcar una sección dentro del documento, usualmente se utilizan, además del título de la sección, caracteres adicionales, para dar más visibilidad a la división. Ej.:



```
/*------------------------------
        Tabla de contenido

    1. General
    2. Header
    3. Inicio
    4. Acerca de
    5. Galeria
    6. Contacto
    7. Footer
------------------------------*/
```

Una de las acciones **altamente recomendadas** es crear una tabla de contenidos al inicio del documento a modo de comentario. Esto para facilitar las modificaciones o buscar estilos específicos con más facilidad dentro del documento. La tabla de contenido cumple la función de índice dentro del archivo CSS. Si el archivo de CSS es muy corto, se puede omitir esta.

Dentro de cada sección, es recomendable ir dando a estilos conforme a su orden de aparición en el archivo a estilar, así se logra mantener el mismo flujo en ambos documentos.

```
<body>
    <section class="contenedor">
        <div class="sect-1">
            <h1>Titulo 1</h1>
            <p>Lorem ipsum dolor sit amet,
            consectetur adipisicing elit. Maiores
            accusantium iure praesentium in,
            impedit.</p>
        </div>
        <div class="sect-2">
            <h1>Titulo 2</h1>
            <p>Lorem ipsum dolor sit amet,
            consectetur adipisicing elit. In,
            porro tenetur aperiam illum debitis
            sequi.</p>
        </div>
        <div class="sect-3">
            <h1>Titulo 3</h1>
            <p>Lorem ipsum dolor sit amet,
            consectetur adipisicing elit.
            Molestiae totam dignissimos aliquid
            commodi.</p>
        </div>
    </section>
</body>
```

```
/*--------------
    General
--------------*/

* {
    margin: 0;
    padding: 0;
    box-sizing: border-box;
    font-family: Montserrat, sans-serif;
}

div {
    width: 100%;
    padding: 6rem 20%;
}

/*--------------
    Sección 1
--------------*/

.sect-1 {
    background-color: #6ACE89;
    color: #fff;
}

.sect-1 h1 {
    text-align: center;
    color: #F8CCE0;
}

.sect-1 p {
    text-align: right;
}
```

```
/*--------------
    Sección 2
--------------*/

.sect-2 {
    background-color: #62A6F3;
}

.sect-2 h1 {
    text-align: right;
    color: #FFAF84;
}

.sect-2 p {
    text-align: center;
    line-height: 1.8rem;
}

/*--------------
    Sección 3
--------------*/

.sect-3 {
    background-color: #F36265;
}

.sect-3 h1 {
    color: #99D7D5;
}

.sect-3 p {
    font-size: .8rem;
}
```

El orden que se recomienda seguir para estilar elementos va desde el elemento exterior, siguiendo con los elementos inmediatamente dentro de este, junto con todos los que se encuentran en ese mismo nivel, así hasta haber recorrido todos los elementos a estilar hasta el elemento que ya no cuente con otro elemento dentro de este. Después de esto, se sigue con el siguiente elemento inferior al inicial en el mismo nivel.

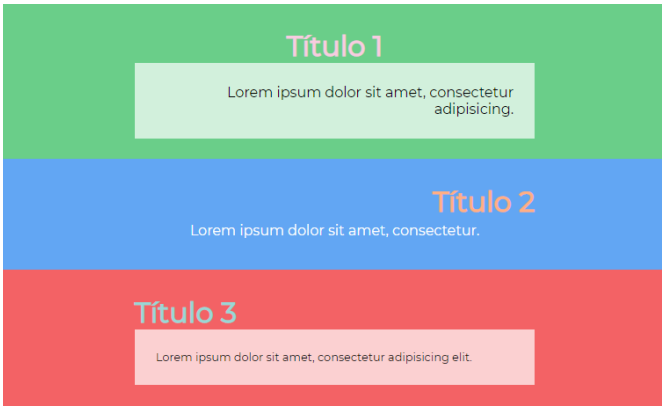
4. Clases generales

Las clases generales sirven para dar estilos a lo largo de uno o varios documentos. Usualmente se utilizan con colores de texto o de fondo, para dar un margen o padding específicos o modificar cualquier otra propiedad de manera constante. Esto para **evitar repetitividad** de las mismas propiedades. En el ejemplo,

se utiliza la clase `white-background` para dar un fondo blanco con transparencia y un padding específico a todos los elementos que la contengan.

```html
<body>
    <div class="contenedor-1">
        <div class="sect-1">
            <h1>Titulo 1</h1>
            <p class = "white-background">Lorem
            ipsum dolor sit amet, consectetur
            adipisicing.</p>
        </div>
        <div class="sect-2">
            <h1>Titulo 2</h1>
            <p>Lorem ipsum dolor sit amet,
            consectetur.</p>
        </div>
        <div class="sect-3">
            <h1>Titulo 3</h1>
            <p class = "white-background">Lorem
            ipsum dolor sit amet, consectetur
            adipisicing elit.</p>
        </div>
    </div>
    <div class="contenedor-2">
        <div class="sect-1">
            <h1>Titulo 1</h1>
            <p>Lorem ipsum dolor sit amet,
            consectetur.</p>
        </div>
        <div class="sect-2">
            <h1>Titulo 2</h1>
            <p class = "white-background">Lorem
            ipsum dolor sit amet, consectetur
            adipisicing elit.</p>
        </div>
    </div>
</body>
```

```css
.white-background {
    background-color: rgba(255,255,255,.7);
    padding: 1.5rem;
}
```



5. Propiedades con capacidad de agrupación

Existen ciertas propiedades que permiten desglosarse o, en su defecto, agruparse, tales como las relacionadas con `background`, `border` o `flex`, o incluso propiedades más básicas como `margin` y `padding`. Estas propiedades, sirven como atajos, cada uno a sus respectivos desgloses. Por ejemplo, con `margin`, se tiene acceso a `margin-top`, `margin-right`, `margin-bottom` y `margin-left` en una sola línea. La diferencia entre el uso de los atajos y las propiedades directas yace en dos puntos. Uno de ellos, es el hecho de que se realiza la asignación de la propiedad en **una sola línea**, por lo cual, el tiempo de carga se optimiza un poco. El segundo punto, es que, al utilizar el atajo, y no asignar todas las propiedades que le corresponden, **las no asignadas se resetean**; esto ocurre de manera diferente dependiendo de la propiedad. Así que, si se quiere asignar una sola propiedad sin modificar las demás, se tendrá que recurrir a la propiedad directa.

En cuanto a optimización y estructura se refiere, no implica mucha diferencia el utilizar una o la otra, así que, **puede quedar a comodidad de cada quien**.

6. Duplicidad de propiedades

Hay casos en los que se quieren asignar una o más propiedades iguales a más de un elemento. Esto se puede lograr utilizando las clases generales mencionadas previamente, pero, si se quiere asignar propiedades a una cantidad no muy grande de elementos, crear una clase únicamente para estos, quedaría un poco holgada. Es en estos casos donde se puede recurrir a escribirlos directo en las propiedades de cada elemento.

Sin embargo, siempre se tiene que buscar hacer la asignación de propiedades de la manera **menos repetitiva posible**. Es aquí donde se suele recurrir a asignación de las propiedades comunes entre elementos del mismo grupo en una sola instancia y las propiedades en las que difieren, de manera individual. Esto para facilitar cambios posteriores y optimizar el archivo en general. Ej.:

```css
.sect-1 {
    padding: 3rem 2rem;
    border: 1.5px solid black;
    background-color: #6ACE89;
    font-size: 2.5rem;
    line-height: 2.65rem;
    font-weight: bold;
}

.sect-2 {
    padding: 3rem 2rem;
    border: 1.5px solid black;
    font-size: 2.5rem;
    line-height: 2.65rem;
    font-weight: bold;
    background-color: #62A6F3;
    color: #fff;
}

.sect-3 {
    padding: 3rem 2rem;
    border: 1.5px solid black;
    font-size: 2.5rem;
    line-height: 2.65rem;
    font-weight: bold;
    background-color: #F36265;
    margin-bottom: 2.5rem;
}
```

```css
.sect-1,
.sect-2,
.sect-3 {
    padding: 3rem 2rem;
    border: 1.5px solid black;
    font-size: 2.5rem;
    line-height: 2.65rem;
    font-weight: bold;
}

.sect-1 {
    background-color: #6ACE89;
}

.sect-2 {
    background-color: #62A6F3;
    color: #fff;
}

.sect-3 {
    background-color: #F36265;
    margin-bottom: 2.5rem;
}
```

7. Media Queries (Diseño responsivo)

Al momento de crear los Media Queries, existen dos maneras principales que se recomienda utilizar. La primera consiste en realizarlos dentro del mismo archivo donde se generaron los estilos de la página. En este caso, se recomienda, para facilidad de navegación, implementar cualquiera de las dos opciones siguientes:

a) Agregarlos **al final de cada sección**, conteniendo únicamente las propiedades correspondientes a esa sección.

b) **Asignarles su sección propia** dentro del archivo CSS, usualmente una de las secciones finales, y anexarla a la tabla de contenido.

La otra alternativa que se recomienda está más dirigida para cuando se tiene un archivo o varios muy extensos de CSS. Es aquí cuando se puede recurrir a generar todos los Media Queries dentro de un solo archivo individual. En este caso, se agregan todas las propiedades dentro de los Media Queries siguiendo el mismo orden que se siguió en los archivos adicionales de estilos, seccionando las propiedades dentro de los Media Queries por archivo de estilos adicional, con sus respectivas secciones internas.

8. Nombramiento de archivos, clases e identificadores

Los nombres de los archivos, las clases y los identificadores en CSS pueden contener prácticamente cualquier texto. Dos cosas que nunca se deben incluir en un nombre de archivo o clase son: espacios y caracteres especiales (á, í, ó, ñ, etc.). No es **para nada recomendable** asignar nombres completamente arbitrarios a ninguno de estos, ya que complicaría demasiado su localización dentro de la carpeta o el archivo para alguien que no sea la misma persona que escribió el código. Se recomienda elegir un nombre que sea **lo más descriptivo posible** referente a lo que se está nombrando. Por ejemplo, en la imagen se muestra una clase con el nombre white-background, la cual, cambia el color de fondo del elemento al que se le aplique a blanco.

```css
.white-background {
    background-color: rgba(255,255,255,.7);
    padding: 1.5rem;
}
```

9. Pseudo-clases y pseudo-selectores

Al momento de editar pseudo-clases o pseudo-selectores, lo más recomendable es asignar las propiedades a estos inmediatamente después de haber asignado estilos al elemento en su estado natural. Algunos selectores como `:focus`, `:active` y `:hover` suelen estilarse juntos, unificando las propiedades a una sola selección, aunque, por supuesto, este no siempre es el caso.