Unidad 2: CSS

Introducción a la Creación de un Archivo CSS

Un archivo CSS (Cascading Style Sheets) es una herramienta fundamental para estilizar páginas web, separando el contenido HTML de su apariencia visual. Al crear un archivo CSS, es importante seguir ciertos pasos y buenas prácticas para asegurar que el proceso sea claro, eficiente y escalable.

Primeros pasos para crear un archivo CSS:

1. **Creación del archivo:**

    - Usa un editor de texto o un entorno de desarrollo integrado (IDE) como Visual Studio Code o Sublime Text.
    - Guarda el archivo con la extensión `.css` (por ejemplo, `estilos.css`).

2. **Estructura básica de un archivo CSS:**

    - El archivo CSS se compone de reglas. Cada regla incluye un **selector** que apunta a los elementos HTML que quieres estilizar, y un bloque de **declaraciones** encerrado entre llaves `{ }`.

    - Una declaración se forma con una **propiedad** (lo que deseas cambiar, como color o tamaño) y un **valor** (el ajuste específico que quieres aplicar).**Ejemplo:**

      ```
      h1 {
          color: blue;
          font-size: 24px;
      }
      ```

3. **Conexión al archivo HTML:**

    - Enlaza tu archivo CSS al documento HTML utilizando la etiqueta `<link>` dentro de la sección `<head>`:

```
<link rel="stylesheet" href="estilos.css">
```

Buenas prácticas iniciales:

- **Organización:** Mantén tu código CSS ordenado y usa comentarios (`/* texto aquí */`) para explicar secciones específicas.
- **Consistencia:** Sigue un formato uniforme, como usar siempre puntos y comas al final de cada declaración y tabulaciones o espacios consistentes para la sangría.
- **Uso de selectores claros:** Evita selectores excesivamente generales o complicados. Empieza con selectores simples como etiquetas (`h1`, `p`) o clases (`.mi-clase`).
- **Manejo de colores y tamaños:** Usa formatos estándar como códigos hexadecimales para colores (`#ff5733`) y unidades relativas como `em` o `rem` para tamaños, ya que ofrecen mayor flexibilidad y adaptabilidad.

Concepto de Padres e Hijos en CSS

Analogía con las Mamushkas

Imagina que los elementos HTML son como las famosas muñecas rusas conocidas como mamushkas. Cada muñeca puede contener otra más pequeña en su interior, y así sucesivamente. De manera similar, en HTML, los elementos pueden contener otros

elementos dentro de ellos, creando una relación de padres e hijos.

Elementos Padres e Hijos

- **Elemento Padre**: Es un elemento que contiene a uno o más elementos dentro de él.
- **Elemento Hijo**: Es un elemento que está contenido dentro de otro elemento.

Herencia de Estilos en CSS

En CSS, los estilos aplicados a un elemento padre pueden ser heredados por sus elementos hijos. Esto significa que si aplicas un estilo a un elemento padre, sus hijos también podrán recibir ese estilo, a menos que se indique lo contrario.

Ejemplo de Herencia

```
<section>

  <article>

    <h2>Título</h2>

    <p>Lorem ipsum dolor sit amet...</p>
```

```
    </article>

</section>
```

En este ejemplo, todos los textos dentro del `<section>` se mostrarán en color azul y con la fuente Arial, porque estos estilos se aplican al elemento padre y se heredan a los hijos.

Aplicar Estilos Específicos a los Elementos Hijos

Es posible aplicar estilos específicos a los elementos hijos sin alterar los estilos del elemento padre. Esto se logra utilizando selectores de elementos hijos en CSS.

Ejemplo de Estilos Específicos

HTML

```
<section>

        <article>

                <h2>Título</h2>

                <p>Lorem ipsum dolor sit amet...</p>

        </article>

</section>
```

CSS

```
/* Estilo aplicado al elemento padre <section> */

section {

color: blue;

font-family: Arial, sans-serif;

}

/* Estilo específico para el elemento hijo <h2> */

section h2 {

color: red;

font-size: 24px;
```

}

/* Estilo específico para el elemento hijo <p> */

section p {

font-size: 16px;

}

En este caso:

- El **<section>** y todos sus elementos hijos tendrán el texto en color azul y la fuente Arial.
- El **<h2>** dentro del **<section>** tendrá un color rojo y un tamaño de fuente de 24px, sobrescribiendo el color azul heredado.
- El **<p>** dentro del **<section>** tendrá un tamaño de fuente de 16px.

Conclusión

Entender la relación de padres e hijos en CSS es esencial para aplicar estilos de manera eficiente y organizada en un documento HTML. La herencia de estilos permite mantener la consistencia visual, mientras que los selectores específicos para elementos hijos ofrecen flexibilidad para personalizar la apariencia de cada parte del contenido. Al igual que las mamushkas, cada elemento HTML puede contener otros elementos, y CSS te permite controlar cómo se ven y se comportan estos elementos anidados.

Diferentes Formas de Insertar CSS en un documento HTML

CSS (Cascading Style Sheets) se puede aplicar a un documento HTML de varias formas: mediante una vinculación externa, una inserción interna con la etiqueta `<style>`, y el uso de estilos en línea. A continuación, se describen estos métodos, sus ventajas y desventajas, y se muestran ejemplos de cómo aplicarlos en diferentes contextos.

1. Vinculación Externa

¿Qué es?

La vinculación externa consiste en enlazar un archivo CSS separado al documento HTML mediante la etiqueta `<link>` dentro del `<head>`.

Ejemplo

```
<head>

   <link rel="stylesheet" href="styles.css">

</head>
```

Ventajas

- **Separación de Contenidos**: Mantiene el HTML limpio y separado del CSS.
- **Reutilización**: Un solo archivo CSS puede ser reutilizado en múltiples páginas web.
- **Mantenimiento**: Facilita la actualización de estilos, ya que los cambios en el archivo CSS afectan a todas las páginas que lo utilizan.

Desventajas

- **Carga Adicional**: Requiere una solicitud HTTP adicional para cargar el archivo CSS, lo que puede afectar el tiempo de carga de la página.

2. Inserción Interna con la Etiqueta `<style>`

¿Qué es?

La inserción interna coloca el CSS directamente dentro del documento HTML utilizando la etiqueta `<style>` dentro del `<head>`.

Ejemplo

```
<head>

<style>

body {

background-color: lightblue;

}

h1 {

color: navy;

margin-left: 20px;

}
```

</style>

</head>

Ventajas

- **Menos Solicitudes HTTP**: No requiere una solicitud adicional para cargar los estilos.
- **Conveniencia**: Útil para estilos específicos de una sola página.

Desventajas

- **No Reutilizable**: Los estilos definidos de esta manera no se pueden reutilizar en otras páginas.
- **Mantención Compleja**: Si se utilizan muchas reglas de estilo, el documento HTML puede volverse difícil de manejar y mantener.

3. Estilos en Línea

¿Qué es?

Los estilos en línea se aplican directamente a los elementos HTML mediante el atributo `style`.

Ejemplo

<body>

<h1 style="color: navy; margin-left: 20px;">Título</h1>

Este es un párrafo.

</body>

Ventajas

- **Específico y Rápido**: Ideal para aplicar estilos rápidos y específicos a un solo elemento.
- **No Requiere `<head>`**: Puede ser usado directamente en el contenido del documento.

Desventajas

- **No Reutilizable**: Los estilos deben ser escritos repetidamente para cada elemento.
- **Difícil de Mantener**: Los estilos en línea dispersos pueden hacer que el HTML sea difícil de leer y mantener.
- **Especificidad Alta**: Los estilos en línea tienen una especificidad alta, lo que puede complicar el uso de CSS externo o interno.

Comparación de Métodos

Vinculación Externa

- **Pros**: Reutilización, mantenimiento sencillo, separación de contenido.
- **Contras**: Requiere solicitudes HTTP adicionales.

Inserción Interna

- **Pros**: Menos solicitudes HTTP, conveniente para estilos específicos de una página.
- **Contras**: No reutilizable, difícil de mantener en grandes cantidades.

Estilos en Línea

- **Pros**: Aplicación rápida y específica, no requiere `<head>`.
- **Contras**: No reutilizable, difícil de mantener, alta especificidad.

Conclusión

Elegir la forma adecuada de insertar CSS en un documento HTML depende del contexto y las necesidades del proyecto. La vinculación externa es ideal para proyectos grandes con múltiples páginas, la inserción interna es útil para estilos específicos de una sola página, y los estilos en línea son mejores para ajustes rápidos y específicos. Comprender las ventajas y desventajas de cada método permite aplicar CSS de manera eficiente y organizada.

Resumen de los Conceptos Clave de CSS

Sintaxis Básica de CSS

¿Qué es CSS?

CSS (Cascading Style Sheets) es un lenguaje utilizado para describir la presentación de un documento HTML. Permite controlar el diseño y el formato de las páginas web, como colores, fuentes, espaciados y disposición de los elementos.

Sintaxis Básica

La sintaxis de CSS se compone de selectores y declaraciones. Un selector define a qué elementos se aplican los estilos, y una declaración incluye una propiedad y un valor. Las declaraciones se agrupan dentro de bloques de declaración.

```
selector {

    propiedad: valor;

}
```

Ejemplo

```
p {

    color: blue;

    font-size: 16px;

}
```

En este ejemplo, el selector `p` aplica estilos a todos los elementos `<p>` en el documento HTML, cambiando su color a azul y el tamaño de fuente a 16px.

Herencia en CSS

¿Qué es la Herencia?

La herencia en CSS permite que ciertos estilos aplicados a un elemento padre se transmitan a sus elementos hijos. No todas las propiedades CSS son heredables, pero aquellas que afectan el texto, como `color` y `font-family`, sí lo son.

Ejemplo de Herencia

```
body {

    color: black;

    font-family: Arial, sans-serif;

}
```

En este caso, todos los elementos dentro del `<body>` heredarán el color negro y la fuente Arial.

La Cascada en CSS

¿Qué es la Cascada?

La cascada es el principio que determina qué estilos se aplican a un elemento cuando hay conflictos. CSS asigna un peso a cada regla según su origen (inline, interno, externo), especificidad y orden de aparición.

Especificidad

La especificidad se calcula con base en el tipo de selectores usados. Los estilos inline tienen mayor especificidad que los selectores de ID, clases y elementos.

Orden de Aparición

Si dos reglas tienen la misma especificidad, la última que aparece en el código tendrá prioridad.

Selectores en CSS

Tipos de Selectores

- **Selector de Elemento**: Selecciona todos los elementos de un tipo específico.

  ```
  p {
      color: blue;
  }
  ```

- **Selector de Clase**: Selecciona elementos con una clase específica.

  ```
  .clase {
      color: green;
  }
  ```

- **Selector de ID**: Selecciona un elemento con un ID específico.

  ```
  #id {
      color: red;
  }
  ```

- **Selectores de Atributo**: Selecciona elementos basados en un atributo. Por ejemplo, para estilizar todos los campos de entrada de texto:

  ```
  input[type="text"] {
      background-color: yellow;
      border: 1px solid #ccc;
  }
  ```

  Buenas Prácticas con Selectores de Atributo

- Evita hacer selectores de atributo demasiado generales, como `input[type]`, ya que pueden afectar múltiples elementos de forma no intencionada.

- Combina selectores de atributo con clases o IDs para mayor precisión:

  ```
  .formulario input[type="text"] {
    padding: 10px;
    font-size: 14px;
  }
  ```

  En este ejemplo, solo los campos de texto dentro de un formulario con la clase `formulario` serán estilizados. Esto mejora la especificidad y la claridad del código.

Recomendaciones y Buenas Prácticas

1. **Mantén el Código CSS Organizado**

   - Usa comentarios para explicar secciones del CSS.
   - Utiliza nombres de clases e IDs descriptivos y coherentes con el contenido o función de los elementos.

2. **Usa una Hoja de Estilo Externa**

   Esto facilita el mantenimiento y la reutilización de los estilos en múltiples páginas.

3. **Minimiza el Uso de Estilos en Línea**

   Prefiere usar hojas de estilo externas o internas para mantener el HTML limpio y estructurado.

4. **Evita la Sobrespecificidad**

   No hagas los selectores más específicos de lo necesario, ya que puede dificultar la sobrescritura de estilos y el mantenimiento del código.

5. **Usa Variables CSS (Custom Properties)**

   Las variables facilitan la gestión de valores repetitivos y permiten actualizaciones más sencillas.

   ```
   :root {
   ```

```
    --main-color: blue;
}
p {
    color: var(--main-color);
}
```

Conclusión

Dominar la sintaxis básica de CSS, comprender la herencia y la cascada, y saber utilizar selectores de manera eficiente son habilidades esenciales para cualquier desarrollador web. Incorporar buenas prácticas y aprovechar selectores como los de atributo mejorará la organización, claridad y efectividad del código CSS, optimizando el mantenimiento y la escalabilidad de tus proyectos.