LEMONCODE

HTML

Cascada CSS

La cascada en CSS es el proceso utilizado por el navegador para decidir qué estilos aplicar a un elemento. A veces, puede haber conflictos entre diferentes reglas CSS, y la cascada ayuda a resolverlos.

Ejemplo práctico

Vamos a ver un ejemplo práctico para comprender cómo funciona la cascada. Supongamos que tenemos el siguiente HTML:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <link rel="stylesheet" href="style.css" />
        <title>Cascada CSS</title>
    </head>
    <body>
        <h1 class="title" id="pageTitle">Título de ejemplo</h1>
    </body>
</html>
```

Y vamos a aplicar diferentes estilos a este elemento utilizando reglas CSS.

```
// style.css

/* Regla 1 */
h1 {
    color: red;
}

/* Regla 2 */
#pageTitle {
    color: blue;
}

/* Regla 3 */
.title {
    color: purple;
}
```

En este caso, hay tres reglas que afectan al mismo elemento `<h1>`. Ahora veamos qué estilo se aplicará.

El primer selector `h1` establece el color en rojo. Luego, el segundo selector `#pageTitle` establece el color en azul. Por último, el tercer selector `.title` establece el color en púrpura.

El estilo que prevalece y se aplica al elemento es el que tiene más especificidad y se declara en último lugar. En este caso, el color azul del selector `#pageTitle` es el que se mostrará.

Criterios de la cascada

La cascada utiliza varios criterios para resolver conflictos de estilos:

1. **Posición y orden de aparición**: El orden en que se declaran las reglas CSS influye en cuál prevalece.
2. **Especificidad de los selectores**: Algunos selectores tienen más prioridad que otros. En el ejemplo anterior, el selector `#pageTitle` que es un ID tiene más prioridad que el selector `.title` que es una clases.
3. **Origen de la hoja de estilos**: El lugar de donde proviene el CSS puede afectar su importancia (hoja de estilos de la página, hoja de estilos del navegador).
4. **Importancia**: Algunas reglas CSS tienen más peso que otras, especialmente cuando se utiliza la regla `!important`.

Ejemplo con posición y orden de aparición

En este ejemplo, veremos cómo afecta el orden de aparición de las reglas CSS:

```
// style.css

/* Regla 1 */
h1 {
    color: red;
}

/* Regla 2 */
h1 {
    color: blue;
}
```

En este caso, la última regla declarada `color: blue` es la que prevalecerá y se aplicará al elemento.

Ejemplo con especificidad de los selectores

En este ejemplo, veremos cómo la especificidad de los selectores afecta la cascada:

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <link rel="stylesheet" href="style.css" />
```

```
        <title>Cascada CSS</title>
    </head>
    <body>
        <p class="red blue">Soy un párrafo</p>
        <p class="blue red">Soy un párrafo</p>
    </body>
</html>
```

```
// style.css

/* Regla 1 */
p.red {
    color: red;
}

/* Regla 2 */
p.blue {
    color: blue;
}
```

Si tenemos un párrafo con las clases `.red` y `.blue`, el color aplicado será el azul, ya que la regla `.blue` tiene mayor especificidad. Indistintamente de cómo se añadan las clases al elemento HTML, el color aplicado será el azul.

El orden de las declaraciones en las reglas CSS también es importante. Vamos a ver un ejemplo donde ponemos que el color sea verde.

```
// style.css

/* Regla 1 */
p.red {
    color: red;
}

/* Regla 2 */
p.blue {
    color: blue;
    color: green;
}
```

En este caso, el navegador ignora la declaración con el color azul porque fue declarada antes que la de color verde.

Escribir dos valores para la misma característica puede ayudar a crear opciones alternativas para los navegadores que no aceptan un valor en particular. Este método de declarar la misma propiedad dos veces es válido porque los navegadores ignoran los valores que no entienden. A diferencia de otros lenguajes de programación, CSS no mostrará un error o romperá el código cuando encuentre una línea que no pueda

analizar. En su lugar, el valor que el navegador no entiende y no puede analizar es totalmente ignorado, y continúa procesando el resto del CSS sin alterar lo que ya comprende.

Conclusión

La cascada en CSS es fundamental para entender cómo se aplican los estilos a los elementos. A través de los criterios de posición y orden de aparición, especificidad de los selectores, origen de la hoja de estilos y la importancia de las reglas, el navegador determina qué estilo tiene prioridad en caso de conflictos.