2. Selectores

Los selectores nos **ayudan a indicar el elemento sobre el que se van a aplicar los estilos**. Los selectores pueden apuntar a elementos específicos, clases, identificadores o incluso atributos de un elemento.

Existen muchos tipos de selectores y algunos de los más destacados son los que se detallan a continuación.

2.1. Selector universal

Sintaxis: `* { atributo:valor; }`

Ejemplo: `* { color: grey; } /* El estilo se aplicará a todos los elementos de la página*/`

El **selector universal (*)** es útil cuando deseas **aplicar un estilo a todos los elementos en una página web sin excepción**. Es una forma rápida y sencilla de establecer reglas generales que afectarán a todos los elementos en el documento. Por ejemplo, si deseas eliminar todos los márgenes y rellenos predeterminados de los elementos en tu página para empezar desde cero, puedes usar el selector universal para restablecerlos.

2.2. Selector etiqueta

Sintaxis: `etiqueta { atributo:valor }`

Ejemplo: `p {color: green;} /* El estilo se aplicará a todos los elementos <p>.*/`

2.3. Selector clase

Sintaxis: `.clase { atributo:valor }`

Ejemplo: `.blend{color: red;} /* El estilo se aplicará a cualquier elemento que tenga la clase .blend */`

2.4. Selector identificador

El **selector identificador** utiliza el atributo id para seleccionar un elemento. Solo puede haber un elemento con un id dado en un documento.

Sintaxis: `#id { atributo:valor }`

Ejemplo: `#cent {color: blue;} /* El estilo se aplicará al elemento que tenga el id #cent */`

2.5. Selector descendiente

Un **elemento es descendiente de otro cuando se encuentra entre las etiquetas de apertura y de cierre del elemento padre.**

Sintaxis: `selector1 selector2 selectorN {atributo: valor;} /* El estilo se aplica sobre el selector N */`

Ejemplo: `div p { color: black;} /* El estilo se aplica a todos los párrafos que se encuentren dentro de una etiqueta div */`

2.6. Combinación de selectores

La combinación de selectores nos permite dar un **estilo a todos los selectores indicados.**

Sintaxis: `selector1, selector2, selector3{atributo: valor;} /* El estilo se aplica sobre los selectores indicados */`

Ejemplo: `div, p { color: orange;} /* El estilo se aplica a todos los divs y párrafos */`

2.7. Selector de hijos

Se usa para **seleccionar un elemento que es *hijo* de otro elemento.**

Sintaxis: `selector1 > selector2 {atributo: valor;} /* El estilo se aplica sobre el selector 2 */`

Ejemplo: `div > p { color: white;} /* El estilo se aplica a todos los párrafos que sean hijos de un div */`

2.8. Selector adyacente

Se usa para **seleccionar elementos que son *hermanos*, es decir, su elemento padre es el mismo y están seguidos en el código HTML**.

Sintaxis: `selector1 + selector2{ atributo: valor; } /* El estilo se aplica al selector 2 */`

Ejemplo: `div + p { color: black;} /* El estilo se aplica a todos los párrafos que sean hermanos de un div */`

2.9. Resumen de selectores básicos

Selector	Descripción
*	Selecciona todos los elementos del DOM
etiqueta	Selecciona todas las etiquetas indicadas
.class	Selección de los elementos con la clase .class
#id	Selección del elemento con id #id
sel1 sel2	Selección de los selectores sel2 que se encuentren dentro de los selectores sel1
.class1.class2	Selección de los elementos con las dos clases: class1 y class2
sel1.class1	Selección de todos los selectores sel1 con clase class1
sel1, sel2	Selección de todos los selectores separados por comas
sel1 > sel2	Selección de los selectores sel2 cuando son hijos de sel1
sel1 + sel2	Selección del selector sel2 cuando es hermano de sel1 (su elemento padre es el mismo)

2.10. Ejemplos

styles.css

```
p { background-color: grey; } /* Selector etiqueta */
.clase { color: red; } /* Selector clase */
#ident { color: green; } /* Selector Identificador */
* { font-style: italic; } /* Selector universal */
p a { background-color: orange; } /* Selector descendiente */
h3, small { color: blue;} /* Combinación de selectores */
div>span { color: pink; } /* Selector de hijos */
span+small { background-color: yellow; } /* Selector adyacente*/
```

inicio.html

```
<html>
<head>
    <meta charset="utf-8">
    <title>Selectores</title>
    <link rel="stylesheet" href="styles.css">
</head>
<body>
    <h1 class="clase">Texto de color rojo</h1>
    <p id="ident">Texto de color verde</p>
  <h2>Selector descendiente</h2>
  <p>
      <a href="#">Enlace</a><br>
      <span>
            <a href="#">Enlace</a>
      </span>
  </p>
  <h3>Texto h3</h3>
  <div>
    <span>Texto con span</span>
    <small> Texto con small</small>
    <span>Texto con span</span>
    <small> Texto con small</small>
    <small> Texto con small</small>
  </div>
</body>
</html>
```

2.11. AGRUPACIÓN DE REGLAS.

Cuando se crean archivos CSS complejos con decenas o cientos de reglas, es habitual que los estilos que se aplican a un mismo selector se definan en diferentes reglas:

```
h1 { color: red; }
...
h1 { font-size: 2em; }
...
h1 { font-family: Verdana; }
```

Las tres reglas anteriores establecen el valor de tres propiedades diferentes de los elementos `<h1>`. Antes de que el navegador muestre la página, procesa todas las reglas CSS de la página para tener en cuenta todos los estilos definidos para cada elemento.

Cuando el selector de dos o más reglas CSS es idéntico, se pueden **agrupar las declaraciones de las reglas para hacer las hojas de estilos más eficientes**:

```
h1 {
  color: red;
  font-size: 2em;
  font-family: Verdana;
}
```

El ejemplo anterior tiene el mismo efecto que las tres reglas anteriores, pero es más eficiente y es más fácil de modificar y mantener por parte de los diseñadores. **Como CSS ignora los espacios en blanco y las nuevas líneas, también se pueden agrupar las reglas de la siguiente forma:**

```
h1 { color: red; font-size: 2em; font-family: Verdana; }
```

Si se quiere **reducir al máximo el tamaño del archivo CSS** para mejorar ligeramente el tiempo de carga de la página web, también es posible indicar la regla anterior de la siguiente forma:

```
h1 {color:red;font-size:2em;font-family:Verdana;}
```

2.12. HERENCIA.

Una de las características principales de CSS es la **herencia de los estilos definidos para los elementos.** Cuando se establece el valor de una propiedad CSS en un elemento, sus elementos descendientes heredan de forma automática el valor de esa propiedad. Si se considera el siguiente ejemplo:

```
<!DOCTYPE html >
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Ejemplo de herencia de estilos</title>
<style type="text/css">
  body { color: blue; }
</style>
</head>
<body>
        <h1>Titular de la página</h1>
        <p>Un párrafo de texto no muy largo.</p>
</body>
</html>
```

En el ejemplo anterior, el selector **body** solamente establece el color de la letra para el elemento <body>. No obstante, la propiedad **color** es una de las que se heredan de forma automática, por lo que todos los elementos descendientes de <body> muestran ese mismo color de letra. Por tanto, establecer el color de la letra en el elemento <body> de la página implica cambiar el color de letra de todos los elementos de la página.

Aunque l**a herencia de estilos se aplica automáticamente, se puede anular su efecto estableciendo de forma explícita otro valor para la propiedad que se hereda**, como se muestra en el siguiente ejemplo:

```
<!DOCTYPE html>
<head>
      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
      <title>Ejemplo de herencia de estilos</title>
      <style type="text/css">
              body { font-family: Arial; color: black; }
              h1 { font-family: Verdana; }
              p { color: red; }
      </style>
</head>
<body>
        <h1>Titular de la página</h1>
        <p>Un párrafo de texto no muy largo.</p>
</body>
</html>
```

En el ejemplo anterior, se establece en primer lugar el color y tipo de letra del elemento <body>, por lo que todos los elementos de la página se mostrarían con ese mismo color y tipo de letra. No obstante, las otras reglas CSS modifican alguno de los estilos heredados.

De esta forma, los elementos <h1> de la página se muestran con el tipo de letra Verdana establecido por el selector h1 y se muestran de color negro que es el valor heredado del elemento <body>. Igualmente, los elementos <p> de la página se muestran del color rojo establecido por el selector p y con un tipo de letra Arial heredado del elemento <body>.

La mayoría de propiedades CSS aplican la herencia de estilos de forma automática. Además, para aquellas propiedades que no se heredan automáticamente, CSS incluye un mecanismo para forzar a que se hereden sus valores, tal y como se verá más adelante.

Por último, aunque la herencia automática de estilos puede parecer complicada, simplifica en gran medida la creación de hojas de estilos complejas. Como se ha visto en los ejemplos anteriores, si se quiere establecer por ejemplo la tipografía base de la página, simplemente se debe establecer en el elemento <body> de la página y el resto de elementos la heredarán de forma automática.

2.13. COLISIONES DE ESTILOS

En las hojas de estilos complejas, es habitual **que varias reglas CSS se apliquen a un mismo elemento HTML.** El problema de estas reglas múltiples es que se pueden dar colisiones como la del siguiente ejemplo:

```
p { color: red; }
p { color: blue; }
<p>...</p>
```

¿De qué color se muestra el párrafo anterior? **CSS tiene un mecanismo de resolución de colisiones muy complejo y que tiene en cuenta el tipo de hoja de estilo que se trate (de navegador, de usuario o de diseñador), la importancia de cada regla y lo específico que sea el selector.**

El **método** seguido por CSS para **resolver las colisiones de estilos** se muestra a continuación:

1. Determinar todas las declaraciones que se aplican al elemento para el medio CSS seleccionado.
2. **Ordenar** las declaraciones **según su origen** (CSS de navegador, de usuario o de

diseñador) y su prioridad (palabra clave !important).

3. **Ordenar** las declaraciones **según lo específico que sea el selector**. Cuanto más genérico es un selector, menos importancia tienen sus declaraciones.
4. Si después de aplicar las normas anteriores existen dos o más reglas con la misma prioridad, **se aplica la que se indicó en último lugar.**

Hasta que no se expliquen más adelante los conceptos de tipo de hoja de estilo y la prioridad, el mecanismo simplificado que se puede aplicar es el siguiente:

1. Cuanto más específico sea un selector, más importancia tiene su regla asociada.
2. A igual ***especificidad***, se considera la última regla indicada.

Como en el ejemplo anterior los dos selectores son idénticos, las dos reglas tienen la misma prioridad y prevalece la que se indicó en último lugar, por lo que el párrafo se muestra de color azul.

En el siguiente ejemplo, la regla CSS que prevalece se decide por lo específico que es cada selector:

```
p { color: red; }
p#especial { color: green; }
* { color: blue; }
<p id="especial">...</p>
```

Al elemento <p> se le aplican las tres declaraciones. Como su origen y su importancia es la misma, decide la especficidad del selector. El selector * es el menos específico, ya que se refiere a "todos los elementos de la página". El selector p es poco específico porque se refiere a "todos los párrafos de la página". Por último, el selector p#especial sólo hace referencia a "el párrafo de la página cuyo atributo id sea igual a especial". Como el selector p#especial es el más específico, su declaración es la que se tiene en cuenta y por tanto el párrafo se muestra de color verde.