PEC 1
LENGUAJES Y ESTÁNDARES WEB

Carlos Rodríguez López

PRIMERA PARTE

EXPLICA BREVEMENTE LAS RAZONES POR LAS QUE ES IMPORTANTE SEPARAR EL CÓDIGO DEL DISEÑO.

La separación del contenido y del diseño tiene mucha importancia en muchos aspectos. Por una parte, al separar el contenido del diseño, cuando queremos modificar la presentación de una parte de la web, por ejemplo, cambiar un color, simplemente hemos de cambiar el color en nuestra hoja de estilos, sin tener que editar una a una todas las páginas que forman nuestra web, que pueden muchas en páginas grandes. Es decir, se vuelve **más fácil mantener las webs**, actualizarlas.

Por otra parte, para los robots que indexan las webs y que sólo les interesa el contenido, no el formato, también será **más fácil indexar** nuestras webs (necesitan analizar menos bytes y el código a analizar es mucho más sencillo).

Por último, al hacer que el contenido sea independiente del diseño, podemos **mantener diferentes diseños** para mostrarlos en diferentes dispositivos o, incluso, a elección del usuario, o a la hora de reutilizar código para otro proyecto.

EXPLICA BREVEMENTE QUÉ SON LAS PSEUDOCLASES Y LOS PSEUDOELEMENTOS. EXPLICA Y PON UN EJEMPLO PRÁCTICO DEL USO DE 3 PSEUDOCLASES Y 3 PSEUDOELEMENTOS.

Las pseudoclases y los pseudoelementos son **pseudoselectores**, llamados así porque no seleccionan elementos, sino ciertas partes de los elementos o éstos bajo ciertas condiciones.

Se escriben a continuación del selector que modifican. Las pseudoclases precedidas de dos puntos (:) y los pseudoelementos de dos puntos dobles (::).

Los pseudoelementos precisan la selección acotándola, refiriéndose al espacio inmediatamente anterior o posterior, a la primera línea, la primera letra,…

Por su parte, **las pseudoclases permiten dar estilo a determinados estados**, como pasar el ratón por encima, o recibir un clic, o ser el elemento activo de un formulario, pero también se refieren a determinadas relaciones dentro del DOM: el primer hijo, un elemento que es el único hijo, determinadas filas de una tabla, etc.

Como **ejemplo de uso de pseudoelementos**, podríamos poner:

::AFTER

```
[href^=http]::after {
  content: '↗';
}
```

Con este código, a los enlaces http absolutos (que deberían ser externos) les añadimos un símbolo final para indicar que el enlace nos sacará de la página actual. Por ejemplo, si en nuestro html escribimos un enlace como:

```
<a href="http://www.uoc.edu">UOC</a>
```

En la página web aparecerá como: UOC↗.

::FIRST-LETTER

```
p::first-letter {
  font-size: 3em;
  border: 1px solid black;
  display: block;
  float: left;
  padding: 10px;
  margin-right: 5px;
}
```

Con este código damos estilo a la primera letra de cada párrafo, haciéndola más grande y colocándole un borde negro. De este modo, le damos a nuestro diseño un aire antiguo y elegante.

::SELECTION

```
p::selection {
  background-color: red;
}
```

Con este código, al seleccionar una porción de un párrafo, el sombreado será rojo y no azul o el color que por defecto utilice nuestro navegador, lo cual puede ir más acorde al diseño de nuestra web.

Ejemplos de pseudoclases:

:HOVER

```
a:hover {
  color: red;
}
```

Este código cambia el color de los enlaces cuando pasamos el ratón por encima. Es decir, sólo modificamos el color de los elementos que responden a cierto estado.

:DISABLED

```
input:disabled {
  background-color: #ccc;
}
```

Seleccionamos con :disabled todos los inputs que no están habilitados, es decir, aquellos en los que el usuario no puede introducir texto.

:REQUIRED

```
input:required {
  border-color: red;
}
```

Este código resalta los inputs obligatorios de un formulario coloreando el borde de estos de rojo. *Se debe usar algún otro elemento de marcado para mejorar la accesibilidad de nuestro site.*

EXPLICA DE FORMA BREVE QUÉ ES Y CÓMO FUNCIONA EL MECANISMO DE CASCADA EN EL LENGUAJE CSS. ¿CÓMO INDICARÍAS QUE UNA REGLA DEBE DE PREVALECER SIEMPRE EN CASO DE CONFLICTO?

El mecanismo en cascada es el sistema predeterminado por el cual los navegadores (o cualquier tipo de visualizador) deciden el orden de prevalencia de las instrucciones de estilo. Consiste en que, si dos instrucciones son mutuamente excluyentes, por ejemplo, porque hemos definido dos tamaños de texto distintos para un mismo contenido, se aplicará el que se ha definido utilizando selectores más específicos. Por ejemplo, podemos definir un tamaño de texto para todos los párrafos de un artículo en 1em, mediante:

```
article>p { font-size: 1em; }
```

y usar el primer párrafo después de cada título a modo de entradilla, definiendo un tamaño mayor mediante la pseudoclase :fist-of-type:

```
article>p:first-of-type { font-size: 1.2em; }
```

De este modo, el tamaño 1em se aplica a todos los párrafos que sean hijos directos de la etiqueta <article>, pero esa instrucción entra en conflicto con la siguiente. Se podría pensar que el orden en el que estén escritas las instrucciones indica al navegador cuál ha de aplicar (por ejemplo, la última que se haya escrito) pero esto no es así, sino que como hay dos instrucciones indicando el tamaño del texto del primer párrafo, la que se utilizará será la que se ha definido de un modo más específico, en este caso, la que utiliza la pseudoclase.

Si en un determinado momento se quiere alterar ese orden y hacer que una instrucción prevalezca sobre las demás, se debe usar la etiqueta “¡important”:

```
article>p { font-size: 1em ¡important; }
```

Eso indica al navegador que en caso de conflicto, debe prevalecer esa instrucción y no la que por orden de cascada hubiera correspondido. En general, no es una buena práctica el uso de esta etiqueta, ya que al alterar el orden de prevalencia, dificulta la comprensión del código y, por tanto, el mantenimiento.

4.1. EXPLICA PARA QUÉ SIRVEN LAS PROPIEDADES FONT-FAMILY Y @FONT-FACE.

@font-face permite incorporar una fuente tipográfica a nuestra web, de modo que el usuario dispondrá de ella para visualizar correctamente la página. La propiedad font-family, por su parte, permite definir la fuente (o familia de fuentes) que se quiere utilizar para un determinado texto de la web.

4.2. ESCRIBE UN EJEMPLO DE CÓDIGO DE CÓMO APLICARÍAS LA PROPIEDAD *FONT-FAMILY* A UN PÁRRAFO QUE TENGA UN TAMAÑO DE LETRA 0.8EM, UN ESTILO ITÁLICA, JUSTIFICACIÓN COMPLETA Y QUE UTILICE LAS FUENTES ARIAL, HELVETICA, SANS-SERIF.

```
p {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 0.8em;
  font-style: italic;
  text-align: justify;
}
```

4.3. ESCRIBE UN EJEMPLO DE CÓDIGO DE CÓMO USARÍAS LA FUENTE AMARANTH EN UNA WEB UTILIZANDO LA PROPIEDAD *@FONT-FACE*

```
@font-face {
  font-family: 'Amaranth';
  src: url('amaranth.eot'); /* IE9 Compat Modes */
  src: url('amaranth.eot?#iefix') format('embedded-opentype'), /* IE6-IE8 */
       url('amaranth.woff2') format('woff2'), /* Super Modern Browsers */
       url('amaranth.woff') format('woff'), /* Pretty Modern Browsers */
       url('amaranth.ttf')  format('truetype'), /* Safari, Android, iOS */
       url('amaranth.svg#svgAmaranth') format('svg'); /* Legacy iOS */
}
body {
  font-family: 'Amaranth';
}
```

SEGUNDA PARTE

Explicación de las entidades HTML y del CSS utilizados en la Parte 2:

HTML:

He procurado separar completamente el contenido del diseño y etiquetar el html de la forma más semántica posible, a continuación detallo las etiquetas utilizadas.

En primer lugar, observo que el documento consta de un único artículo, sin elementos externos. Comienza con el título del artículo, el nombre del autor y la fecha. A continuación hay una serie de subtítulos (todos del mismo nivel) y párrafos. Por último, hay un pie en el que se cita la fuente del artículo. Por lo tanto, una estructura semántica lógica podría ser:

Artículo {	<article>
Encabezado {	<header>
Título	
Autor y fecha	
}	</header>
Cuerpo {	<section>
Párrafos	
…	…
Subtítulos	
Párrafos	
…	…
}	</section>
Pie {	<footer>
Párrafo	
}	</footer>
}	</article>

En base a esta estructura, se ha organizado el marcado de la página tal y como se muestra en la tabla anterior.

Además, el lenguaje utilizado es de uso exclusivo geográfico, por lo que se precisa en la etiqueta <html lang="es-ar">

La elección de las etiquetas <article/>, <header/> y <footer/>, no requieren mayores explicaciones. Se podría haber hecho de otras formas, pero la elección es lógica en base al análisis planteado de la estructura de la web.

Más discusión habría en el uso de <section/> para el cuerpo, porque se podría haber prescindido de esta etiqueta, pasando directamente al código del contenido. El diseño es simple y no hubiera requerido usar esa etiqueta. No obstante, parece que lo más correcto es trasladar la estructura semántica al código. De este modo, si en el futuro hubiera que aplicar algún estilo a todo el cuerpo, podría utilizarse el selector **`section`**.

Cabe destacar que no se ha utilizado la etiqueta <hr/> para crear las líneas rosas horizontales que separan encabezado, cuerpo y pie, por considerar que es un elemento de diseño que sería prescindible en otro diseño y que no tiene un significado semántico como podría ser el de separar dos secciones de temátics distintas, por lo que se crearán mediante CSS.

Vamos ahora a ver cómo se ha resuelto la agrupación de información formada por Autor y Fecha:

```
<div class="autor-fecha">
    …
</div>
```

En primer lugar, se ha de justificar el uso de en lugar de <section/>. Se ha considerado que los elementos autor y fecha no guardan una relación semántica como para dotar al conjunto de identidad semántica propia. Sin

embargo, sí guardan relación visual, formando un bloque de información contextual. Por esta razón se ha descartado utilizar la etiqueta <section/> y se ha optado por agruparlos bajo una etiqueta .

Además, se ha optado por asignarle la clase class=”autor-fecha”, que se usará en el CSS para asignarle estilos al bloque. Como es la única instancia de este tipo, se podía haber optado por usar un id=”autor-fecha” en lugar de la clase. Sin embargo, es frecuente que en una misma página se publiquen varios artículos y aunque estemos creando una página en la que sólo se mostrará un artículo, el mismo código HTML y el mismo CSS podrían servir para una página principal en la que se mostrasen varios artículos, por lo que esta opción favorece el mantenimiento. La última razón para elegir la clase, es que cuando se aplican estilos CSS, siempre es preferible utilizar los selectores menos específicos posibles. Eso hace que sea más fácil el mantenimiento del código, para no caer en el uso de la etiqueta ¡important.

Dentro del div autor-fecha, se ha optado por considerar que cada línea es un párrafo distinto. No requiere mucha explicación, pero el hecho de separar cada elemento en un párrafo, responde también a la intención de separar la semántica del diseño. En otro diseño podrían estar en la misma línea o en cualquier otra posición, lo cual podremos conseguir sin retocar el html.

Como me he encontrado con el nombre de una persona, he decidido marcarlo también en el código, al tiempo que practico el uso de microdatos, que es algo que conocía pero que nunca había usado. Del mismo modo, he marcado la fecha con la etiqueta time. Me ha parecido el marcado más semántico posible.

Dentro del cuerpo he usado una clase “disclaimer” para dar estilo a un párrafo especial. Para las etiquetas de código he usado la etiqueta <code/> y he tenido que escapar los símbolos “<” y “>”.

Se han especificado los significados de las abreviaturas HTML, CSS y JS tanto en los párrafos del cuerpo como en los subtítulosh2, con la etiqueta <abbr/>.

Para la cita “above the fold”, siguiendo lo trabajado en el módulo de clase he asignado la etiqueta <i/>. Es algo que tampoco había hecho nunca, cambiar el idioma para un texto literal en otro idioma dentro de una web. He dudado mucho entre utilizar <q/> o <i/>. Creo que el hecho de utilizar el mismo ejemplo en los apuntes, es poco didáctico en ese sentido, pero entiendo que es más correcto usar <i/>.

Por último, en el pie utilicé la etiqueta <q cite=””> para referenciar el artículo original. También es algo que no había hecho hasta ahora y que me ha hecho investigar y aprender que blockquote se usa para citas largas y que cite, como eetiqueta, se usa para referenciar trabajos creativos: libros, películas, música,..

Como comentario final al html, dado que el código está escrito en html5, se ha añadido un pequeño javascript al final de la página para que sea soportado por IE 8 y anteriores.

CSS:

Empezaremos por comentar que para los colores se han usado colores con nombre lo más próximos a los originales posible, pero que tal vez no coincidan con los originales. Hubiera sido más preciso utilizar una herramienta cuentagotas y usar el código hexadecimal exacto, pero el uso de nombres resulta más legible para una actividad de este tipo y también ha sido una buena ocasión para buscar una tabla con los 140 colores con nombre disponibles y conocerlos un poco mejor.

Para los elementos rosas se ha usado el fuchsia. Para el autor y la fecha el darkgray, al igual que para el párrafo de disclaimer. Para los códigos se ha usado dodgerblue y para la cita, green.

Los primeros códigos personalizados dan estilo a los títulos. Al título h1 se le asigna un tamaño de fuente de 2em, un subrayado, alineación centrada y un margen inferior de 25px para separarlo del siguiente bloque de texto: autor y fecha.

Como vimos cuando explicamos el código HTML, las líneas rosas horizontales no se codificaron como <hr>, sino que se dejaron para implementar con CSS como bordes. Se ha optado por un borde inferior en el header y un border superior en el footer, aunque también se podía haber hecho con los bordes del cuerpo. Se les ha asignado un grosor de 2px, línea continua y color fuchsia.

A continuación, formateamos el bloque autor-fecha:

Lo primero que hacemos es colocarlo verticalmente. Como anteriormente ya habíamos establecido el margen del elemento superior (el título h1), ahora simplemente establecemos el margen superior en 0px, por si algún navegador pudiese añadir algún margen adicional que no deseemos.

Se formatean los dos párrafos del bloque autor-fecha con color de fuente darkgray, texto en negrita, alineación derecha y un margen inferior de 5px.

Al nombre del autor se le ha aumentado un poco el tamaño de la fuente (1.1em) y a la fecha se le ha reducido (0.9em).

Para asegurar la colocación vertical de la fecha, se utiliza un margen superior expresamente de cero (ya que anteriormente se le dio un margen de 5px a los dos párrafos). También se ha puesto la fecha en cursiva (font-style: italic;).

El cuerpo del artículo lleva el texto de los párrafos justificado. Además, dado que en el pie también tengo un párrafo que quiero justificar y que los párrafos de la cabecera ya los he alineado a la derecha con un selector más específico (.autor-fecha p), aquí utilizo el selector menos específico posible, para afectar a todos los párrafos que no tengan ninguna otra alineación definida.

Selecciono el tercer párrafo mediante su clase específica .disclaimer y le asigno color y cursiva. Al hacerlo de esta forma, si en cualquier otra parte de la

web hubiera otro disclaimer, simplemente con utilizar la clase en el html, se le aplicaría automáticamente el mismo estilo.

Para las etiquetas de código he usado directamente el selector de elemento, ya que todos los códigos tienen el mismo formato: negrita, cursiva y color azul. Si tuviésemos, por ejemplo, distintos colores para diferenciar lenguajes, tendríamos que haber usado alguna clase para distinguirlos.

Para la cita en inglés he preferido, en cambio, usar una clase, ya que podría haber otras citas en la misma web que utilizasen otro elemento, por ejemplo: , o <q class="quote"></q>. Así, no importa qué etiqueta se utilice ni habrá que usar, por ejemplo, para una cita que no requiera cambiar el idioma y que ocupe todo el párrafo, etiquetas html adicionales que no aportan valor semántico.

Para la referencia del pie he puesto el texto en negrita usando, de nuevo, el selector de elemento.

Por último, para que los navegadores anteriores a IE9, para los cuáles hemos creado mediante javascript las etiquetas html5 que no reconocerían, les asignamos el valor de display: block; ya que por defecto se mostrarían como inline.