El lenguaje de marcas HTML

Todas las páginas web están escritas básicamente en un "lenguaje" llamado HTML. Cuando un navegador solicita una página web, lo que recibe del servidor es precisamente ese código fuente en HTML . La tarea del navegador es saber interpretar ese código HTML en elementos visibles para el usuario (imágenes, párrafos, títulos, etc.).

El lenguaje HTML es un estándar propuesto por el W3C, una organización internacional que promueve el desarrollo y utilización de diferentes estándares abiertos para asegurar el crecimiento sostenible de la web. Esto significa que, en teoría, todos los navegadores deberían interpretar el mismo código HTML de la misma manera, por lo que en este sentido daría igual qué navegador utilizar ya que toda página se debería visualizar de la misma forma (aunque esto en la práctica no es exactamente así). La versión actual de HTML es la número 5, y su especificación se puede consultar en la dirección http://www.w3.org/TR/html51

Para escribir código HTML basta con cualquier editor de texto plano (tal como "gedit" o "notepad.exe", por citar sólo dos posibles) aunque, por comodidad, también se puede optar por utilizar editores HTML especializados con funcionalidades tales como el coloreado de código, el autocompletado, ayudas visuales, etc que facilitan el trabajo (una lista bastante exhaustiva de estos últimos se puede encontrar aquí: http://en.wikipedia.org/wiki/List_of_HTML_editors). En clase, no obstante, nos limitaremos a utilizar un editor de texto simple.

En realidad, el lenguaje HTML no es un lenguaje de programación, sino un simple conjunto de etiquetas que indican la estructura de una página y qué elementos la forman (párrafos, listas, fotografías, etc). A este tipo de lenguajes se les llama "lenguaje de marcas". Toda marca se identifica mediante un nombre predefinido (en minúsculas) encerrado entre el signo < y el signo >, pero pueden haber dos tipos de marcas: las que "se abren y se cierran" conteniendo en su interior un determinado contenido (al cual modifican de alguna manera) y las que aparecen de forma independiente y aislada. Las primeras indican su apertura mediante la sintaxis *<nombreMarca>* y su cierre mediante *</nombreMarca>* (por ejemplo, la etiqueta * * inicia un párrafo y la etiqueta * * lo termina, por lo que para definir un párrafo en nuestra página web deberemos escribir algo así: * Esto es un párrafo. *). Las segundas solamente se escriben una vez mediante la sintaxis *<nombreMarca>* (por ejemplo, la etiqueta *
* introduce un salto de línea). Por otro lado, la mayoría de etiquetas (ya sean de un tipo u otro) incluyen además la posibilidad de incorporar atributos que pueden tomar determinados valores (en la forma *nombreAtributo="valor"*) para modificar el comportamiento de la etiqueta asociada (por ejemplo, la etiqueta *<img>* sirve para incluir una foto en la página, pero para indicar qué foto en concreto se ha de añadir siempre el atributo *src*, así: *<img src="foto.png">* .

A continuación se lista una brevísima selección de etiquetas más habituales como referencia básica para posteriores ejemplos que veremos:

<!DOCTYPE HTML> : Esta etiqueta siempre ha de ser la primera de todo código HTML (es obligatoria), y ha de indicarse una sola vez. Sirve para informar al navegador sobre qué versión de HTML se ha utilizado para escribir dicho código. En clase se asumirá que siempre el uso de la última versión (HTML5), por lo que la etiqueta ha de ser tal como se indica.

<html> : Define el inicio del documento HTML. Obligatoria. Se ha de escribir justo tras la etiqueta *<!DOCTYPE HTML>*
</html> : Define el final del documento HTML.

<head>: Define el inicio de la cabecera del documento HTML. Esta cabecera suele contener información sobre el documento que no se muestra directamente al usuario. Por ejemplo, en ella es posible encontrar las etiquetas *<title></title>*,*<link></link>*,*<style></style>*,*<script></script>*...
</head> : Define el final de la cabecera.

<title>…</title>: En su interior se define el título de la página. Por lo general, éste aparece en la barra de título encima de la ventana.

<link rel="stylesheet" type="text/css" href="/ruta/fichero.css">: Enlaza la presente página web con un fichero externo de tipo CSS para así poder aplicar los estilos definidos en él a los elementos HTML de ésta. Si el código CSS no fuera muy extenso, se podría incluir dentro de la propia cabecera del documento HTML mediante la etiqueta **<style type="text/css">código-CSS</style>**, sin necesidad, por tanto, de recurrir a ficheros externos.

<script type="application/javascript" src="/ruta/codigo.js"></script> : Enlaza la presente página web con un fichero externo con código Javascript para así dotar de mayor interactividad y flexibilidad a los elementos HTML de ésta. Si el código Javascript no fuera muy extenso, se podría incluir dentro del propio código HTML de la página (ya sea en la cabecera o en el cuerpo) mediante la sintaxis **<script type="application/javascript">código-Javascript</script>** sin necesidad, por tanto, de recurrir a ficheros externos.

<body>: Define el inicio del cuerpo del documento (es decir, lo que realmente muestra el navegador). Obligatoria. Dentro del cuerpo es posible encontrar numerosas etiquetas más, algunas de las cuales se listan en las líneas siguientes.
</body> : Define el final del cuerpo.

<h1>…</h1>: El contenido de su interior representa un encabezado (un "título") del documento. Existen más etiquetas del estilo, como *<h2>…</h2>*, *<h3>…</h3>*, etc (hasta *<h6>…</h6>)* las cuales también sirven para representar encabezados, pero cada vez de menor relevancia (es decir, *<h1>* representa el encabezado más prominente, *<h2>* un encabezado algo menos importante, *<h3>* algo menos, y así).

** … ** : El contenido de su interior representa un párrafo.

… : El contenido de su interior (que puede ser un texto pero también otro elemento, como una imagen) representa un enlace que apunta a la URL indicada (o bien a algún fichero local, ya que el valor del atributo *href* también podría ser una ruta). Por ejemplo, *Hola* hará que la palabra "Hola" aparezca marcada dentro de la página web de una forma especial (normalmente, subrayada y de color azul); si entonces el usuario activa esa palabra (haciendo clic sobre ella, por ejemplo), el navegador lo redirigirá a la URL señalada (en este caso, el buscador de Google).

<img src="/ruta/foto.png"> : Indica la presencia de una imagen, cuya ruta (que puede ser también una URL) se ha de especificar como valor del atributo *src*.

**
…
** : El contenido de su interior representa una lista de ítems ordenados (esto es, numerados). Cada uno de estos ítems, a su vez, ha de estar dentro de la etiqueta **
…
** .

**
…
** : El contenido de su interior representa una lista de ítems sin ordenar (esto es, con viñetas). Cada uno de estos ítems, a su vez, ha de estar dentro de la etiqueta **
…
** .

**
** : Inserta un salto de línea

<table>…</table>: Define una tabla de valores. En su interior puede existir una sección (opcional) **<thead>…</thead>** para señalar la cabecera de la tabla (lo que suele ser la primera fila mostrando los títulos de las columnas) y una sección **<tbody>…</tbody>** para señalar cuerpo de la tabla, con su contenido propiamente dicho. Cada fila a mostrar (ya sea en la cabecera o en el cuerpo) se ha de indicar mediante las etiquetas **<tr>…</tr>**, en cuyo interior se definirán las diferentes celdas que la componen (en el caso de la cabecera, las celdas se delimitan mediante **<th>…</th>** y en el cuerpo se delimitan mediante **<td>…</td>**).

** … **: Define una sección de la página, cuyo contenido interno puede estar formado por cualquier conjunto de etiquetas. Sirve para agrupar una porción de código HTML dentro de un bloque identificado con un determinado nombre.

NOTA: El atributo *id* se puede añadir, de hecho, a todas las etiquetas que representen algún elemento del cuerpo de la página (*<h1>*, *<p>*, *<a>*, *<img>*, *<ol>*, *<ul>*, *<table>*…) para identificarlo unívocamente respecto el resto de elementos de esa página. Esto sobre todo es útil si vamos a emplear el lenguaje Javascript como complemento al HTML.

Así pues, un ejemplo de documento HTML simple podría ser el siguiente (las tabulaciones son una ayuda visual para reconocer el principio y el fin de las distintas etiquetas). Para poder visualizarlo en un computador, el lector debería abrir su editor de textos favorito y escribir en él todo este código HTML, a continuación guardarlo en un fichero con extensión ".html" (dentro de cualquier carpeta del sistema) y, finalmente, hacer doble clic sobre ese fichero: se debería abrir entonces el navegador predeterminado mostrando la página web.

```
<!DOCTYPE HTML>
<html>
	<head>
		<meta charset="utf-8">
		<title>Ejemplo de página web simple</title>
	</head>
	<body>
		<h1>Un encabezamiento importante</h1>
		<p>Un párrafo</p>
		<h3>Un encabezamiento menos importante</h3>
		<p>Otro párrafo</p>
		<ul>
			<li>Ítem lista sin ordenar</li>
			<li><a href="http://www.nodejs.org">Otro</a></li>
		</ul>
		<ol>
			<li>Ítem lista ordenada</li>
			<li>Otro ítem</li>
		</ol>
		<table>
			<thead>
				<tr><th>Título1</th><th>Título2</th></tr>
			</thead>
			<tbody>
				<tr><td>1ª celda</td><td>2ª celda</td></tr>
				<tr><td>1ª celda</td><td>2ª celda</td></tr>
			</tbody>
		</table>
	</body>
</html>
```

En cualquier caso, para comprobar que un código HTML cualquiera esté perfectamente escrito y sin errores, podemos hacer uso del validador online que ofrece la propia W3C en http://validator.w3.org

El lenguaje HTML5 es muchísimo más rico y extenso de lo que se ha dejado entrever en las líneas anteriores, pero se sale de los objetivos del curso el estudio de este estándar. Si se desea profundizar en él, además de la especificación oficial recomiendo consultar el tutorial http://www.w3schools.com/html

Formularios HTML

Un formulario HTML es un mecanismo que habilita cierta interacción entre un usuario y un servidor web. Gracias a los formularios un usuario puede enviar datos al servidor web o también transmitirle órdenes (mediante botones u otros elementos), obteniendo siempre una respuesta al instante.

Todo formulario HTML comienza con la etiqueta **<form>** y acaba con la etiqueta **</form>**. Entre estas dos etiquetas se han de colocar los elementos que formarán ese formulario, los cuales pueden ser de muchos tipos (cajas de texto, listas desplegables, "checkboxs", etc.). A continuación se lista una brevísima selección de los elementos más habituales de un formulario, como referencia básica para todo el curso:

<input type="text" name="nombreItem" value="Texto"> : Define una caja de introducción de texto (y le da un determinado nombre mediante el atributo *name)*. El valor del atributo *value* será el texto mostrado por defecto. El valor del atributo *name* junto con el texto finalmente introducido será la pareja de datos clave<->valor que se enviará al destino final.

<input type="password" name="nombreItem" value="Texto"> : Define una caja de introducción de texto con los caracteres enmascarados. El valor del atributo *value* será el texto (enmascarado) mostrado por defecto. El valor del atributo *name* junto con el texto finalmente introducido será la pareja de datos clave<->valor que se enviará al destino final.

<input type="number" name="nombreItem" value="nº por defecto" min="nºmin" max="nºmax"> : Define una caja de introducción de números a partir de una lista dada por el número mínimo y máximo indicados en los atributos *min* y *max*, respectivamente. El valor seleccionado por defecto es el especificado en el atributo *value*. Para que los valores seleccionables no sean consecutivos (es decir, para que exista un salto regular entre ellos mayor que 1, se puede indicar el atributo *step="nºsalto"*. El valor del atributo *name* junto con el número finalmente seleccionado será la pareja de datos clave<->valor que se enviará al destino final.

<input type="checkbox" name="nombreItem" value="valor"> : Define una casilla de verificación. El valor del atributo *name* identifica la casilla concreta; este nombre, junto el valor del atributo *value* correspondiente, será la pareja de datos clave<->valor que se enviará al destino final siempre y cuando la casilla esté seleccionada. Otro atributo interesante de las casillas de verificación es *checked="true"*, que sirve para indicar que la casilla correspondiente estará seleccionada por defecto.

<input type="radio" name="nombreComún" id="idBotón" value="valor"> : Define un botón "radio" (éste es un tipo de botón que suele utilizarse en grupo, junto con otros del mismo tipo, y que tiene la particularidad de ser excluyente: si un solo botón de un grupo es seleccionado, automáticamente el resto de botones del mismo grupo pasan a estar no seleccionados). El valor del atributo *name* ha de ser idéntico para todas los botones "radio" que queramos que pertenezcan al mismo conjunto de opciones excluyentes entre sí. Para identificar entonces a cada botón individualmente se ha de utilizar el atributo *id*. El valor del atributo *value* del botón seleccionado junto con el valor de su atributo *name* (compartido con los otros botones "radio" de su grupo, recordemos) será la pareja de datos clave<->valor que se enviará al destino final. Otro atributo interesante de los botones "radio" es *checked="true"*, que sirve para indicar qué botón estará seleccionado por defecto.

<select name="nombreItem">…</select> : Define un cuadro de opciones desplegables (y le da un determinado nombre mediante el atributo *name)*. Cada una de estas opciones se indicará en su interior a su vez mediante la etiqueta **<option value="valor">Texto</option>,** donde "Texto" es el mensaje que se mostrará al usuario y "valor" es el valor realmente seleccionado cuando el usuario active el texto asociado. Este dato (es decir, el valor del atributo *value*), junto con el valor del atributo *name* correspondiente, será la pareja de datos clave<->valor que se enviará al destino final. Otro atributo interesante de la etiqueta <option> es *selected="true"* , que sirve para indicar la opción seleccionada por defecto.

<textarea name="nombreItem">Texto</textarea>: Define una caja de introducción de texto multilínea (y le da un determinado nombre mediante el atributo *name*). El texto escrito en su interior será el texto mostrado por defecto. El valor del atributo *name* junto con el texto finalmente introducido será la pareja de datos clave<->valor que se enviará al destino.

<button type="submit">Texto</button> : Define el botón que será utilizado para enviar al destino final los datos introducidos y/o seleccionados en los diferentes ítems del formulario. En su interior se puede escribir el texto que se mostrará en el interior del botón (por ejemplo, "Enviar"), pero también se puede indicar una foto (mediante la etiqueta *<img src="…">* estándar) si queremos que el botón tenga una apariencia más sofisticada.

<button type="reset">Texto</button> : Define el botón que será utilizado para limpiar todas las introducciones y/o modificaciones de valores seleccionados que el usuario haya podido efectuar (sin que se produzca ningún envío en ningún momento). En otras palabras, reinicia el formulario al estado por defecto (muy útil en caso de equivocación por parte del usuario, por ejemplo). En su interior se puede escribir el texto que se mostrará en el interior del botón (por ejemplo, "Cancelar"), pero también se puede indicar una foto (mediante la etiqueta *<img src="…">* estándar) si queremos que el botón tenga una apariencia más sofisticada.

Así pues, un ejemplo de formulario HTML simple podría ser el siguiente. Para poder visualizarlo en un computador, el lector debería abrir su editor de textos favorito y escribir en él todo este código HTML, a continuación guardarlo en un fichero con extensión ".html" (dentro de cualquier carpeta del sistema) y, finalmente, hacer doble clic sobre ese fichero: se debería abrir entonces el navegador predeterminado mostrando el formulario.

```
<!DOCTYPE HTML>
<html>
    <head>
        <meta charset="utf-8">
        <title>Ejemplo de formulario simple</title>
    </head>
    <body>
        <form>
            Caja de texto: <input type="text" name="Caja" value="Texto por defecto"><br>
            Caja de contraseña: <input type="password" name="Contra" value="Texto"><br>
            Sólo 1,3,5: <input type="number" name="Num" value="3" min="1" max="5" step="2"><br>
            Valor 'a':  <input type="checkbox" name="check1" value="a">
            Valor 'b': <input type="checkbox" name="check2" value="b" checked="true"><br>
            Valor 'x': <input type="radio" name="Encuesta" id="rad1" value="x">
            Valor 'y': <input type="radio" name="Encuesta" id="rad2" value="y" checked="true"><br>
            Desplegable:
                <select name="Desplegable">
                    <option value="1">Hola</option>
                    <option value="2" selected="true">Adiós</option>
                </select> <br>
            Área de texto: <textarea name="Area">Texto por defecto</textarea><br>
            <button type="submit">Enviar</button>
            <button type="reset">Cancelar</button>
        </form>
    </body>
</html>
```

En los párrafos anteriores se ha repetido varias veces la expresión "enviar los datos introducidos al destino final" pero en ningún momento hemos establecido cuál era ese "destino final". Lo más habitual es que dicho destino final sea una página diferente de la que se encarga de mostrar el formulario, una página no visible específicamente programada para recibir los datos introducidos por el usuario en el formulario, procesarlos de la manera adecuada (conectando, si hiciera falta, a bases de datos) y devolver el resultado final; normalmente, este tipo de páginas "no gráficas" responsables de manipular datos internamente están desarrolladas en lenguajes avanzados como PHP, Python, Java, etc. ya que HTML no está diseñado para ello. No obstante, también es posible hacerlo todo en una sola página: tanto la visualización del formulario (mediante el código HTML pertinente) como la manipulación de los datos introducidos (mediante código PHP específico).

En el primer caso, donde el "destino final" de los datos introducidos en un formulario es una página externa diferente de la responsable de mostrar el formulario en sí, deberíamos indicar en la etiqueta *<form>* la ruta de dicha página externa; esto se hace mediante el atributo **action**, así, por ejemplo: *<form action="/ruta/pagina.php">* . En el segundo caso, donde no hay página separada para procesar los datos sino que es la misma página que muestra el formulario, el atributo *action* no será necesario; por tanto, las etiquetas *<form>* se podrán escribir "tal cual" o, también (es equivalente) especificando "/" como valor del atributo *action*, así: *<form action="/">*.

Otro atributo importante de la etiqueta *<form>* es method, el cual sirve para indicar si el envío de datos al destino se realizará utilizando el método GET (*method="get"*) –el usado por defecto si no se indica nada- o POST (*method="post"*).

No deja de ser interesante como ejercicio abrir con el navegador el fichero ".html" con el código del formulario mostrado como ejemplo en las páginas anteriores y fijarse en la barra de direcciones una vez se ha clicado sobre el botón de "Enviar": podemos observar cómo se genera automáticamente una querystring (recordemos que por defecto el método de envío de datos es GET) con todos las parejas clave<->valor definidas tras la introducción o selección de los diferentes ítems del formulario. Un ejemplo de querystring generada por el formulario cuyo código HTML se ha mostrado anteriormente podría ser así: *"?Desplegable=2&Area=Un+texto&Caja=Otro+texto&Contra=1234&Número=3&check1=a&check2=b&Encuesta=y"*. Como se puede ver, la contraseña en las querystrings se muestra tal cual (¡hay que tener mucho ojo con esto; mejor usar POST en estos casos!), y, en este ejemplo en concreto, se han seleccionado las dos cajas de verificación y el botón de "radio" activado ha sido el segundo.

Otro ejercicio también interesante es modificar en la propia barra de direcciones alguno de los elementos presentes en la querystring (cambiando sus valores por otros diferentes, añadiendo parejas clave<->valor nuevas, o eliminándolas) y pulsar Enter: haciendo esto se puede comprobar que el resultado de alterar la querystring manualmente en la barra de direcciones del navegador (y seguidamente pulsar Enter) es equivalente a alterar los datos dentro de los elementos del formulario (y seguidamente pulsar en el botón de envío).

Breve apunte sobre otra tecnología web: CSS

La tarea del "lenguaje" HTM5 es indicar qué elementos han de aparecer en una página web, pero este lenguaje no sirve para establecer la apariencia visual que estos elementos han de ofrecer al usuario. Es decir, HTML5 define la estructura de las páginas web pero no su estética (por eso los ejemplos del apartado anterior se veían tan poco vistosos). Esta "carencia" del HTML5 está diseñada con toda la intención, para separar lo que corresponde al contenido en sí de la página de lo que corresponde a su representación externa; con esto se consigue desarrollar código mucho más claro, modular, flexible y versátil.

La tecnología encargada de definir el aspecto y formato de los elementos definidos por HTML5 es el estándar CSS, propuesto también por el W3C. Su versión actual es la 2.1 (http://www.w3.org/TR/CSS2) pero la mayoría de navegadores ya tienen implementados muchos aspectos de la siguiente versión (la nº3) del estándar (http://www.w3.org/Style/CSS/current-work).

La información de estilo puede ser adjuntada como un documento separado (en forma de fichero con extensión ".css") o escribirse dentro del mismo documento HTML5. En el primer caso, dentro de la cabecera del código de la página web se ha de utilizar la etiqueta *<link>* para hacer referencia a la hoja de estilo externa (es decir, al documento CSS) y en el segundo se ha de utilizar la etiqueta *<style>* para escribir el código CSS directamente allí. El primer método tiene la ventaja de facilitar mucho el trabajo del desarrollador porque logra tener un lugar común y centralizado (la hoja de estilo) desde donde definir la apariencia visual de todo un sitio web: como un mismo documento CSS puede vincularse a múltiples páginas HTML5, si se realizara algún cambio en el estilo asociado a un determinado tipo de elemento presente en la hoja de estilo, este cambio afectaría automáticamente al instante a todos los elementos de ese tipo que existieran en todas las páginas web vinculadas a esa hoja de estilo. El segundo método es más utilizado para cambio de estilo puntuales.

Si una página HTML5 no tuviera asociada ninguna hoja de estilo, sus elementos son mostrados utilizando un aspecto predefinido por el navegador que se esté utilizando, el cual incorpora hojas de estilo por defecto para estos casos.

CSS funciona a base de reglas: cada regla consiste en un selector y un bloque de declaraciones; el selector sirve para especificar el elemento (o elementos, si se indican separados por comas) que van a ser afectados por el bloque de declaraciones,y dicho bloque consiste –como su nombre indica- en un conjunto de declaraciones encerradas entre llaves y separadas entre sí por puntos y comas -";"- que establecen los estilos concretos a aplicar a ese/os elemento/s. Cada declaración está formada a su vez por el nombre de una propiedad CSS determinada seguido del el símbolo dos puntos –":"- y finalmente el valor concreto asignado a esa propiedad (valor que, si incluye algún espacio en blanco, deberá estar escrito entre comillas). Por ejemplo, la regla…:

h1,p {color:red; background-color:blue;}

…indica que todos los elementos *<h1>…</h1>* y * … * (selectores h1 y p) existentes en todas las páginas HTML5 que estén vinculadas a dicha regla tendrán un color de letra rojo y un color de fondo azul (declaraciones *color:red* y *background-color:blue)*.

Además de aplicar estilos "indiscriminadamente"a todos las etiquetas de un mismo tipo, también es posible aplicar estilos a elementos concretos (que se "suman" a los estilos más generales). Para ello esos elementos HTML5 han de incorporar el atributo ***class="nombreClase"*** y en la hoja de estilo el selector correspondiente ha de ser indicado así: ***.nombreClase*** (notar el punto inicial). De esta manera, sólo los elementos de esa clase se verán afectados por los estilos declarados bajo ese selector. Incluso, yendo más allá, es posible aplicar estilos a un elemento individual específico (los cuales se superponen a los posibles estilos más generales que pudieran haberse definido). Para ello ese elemento HTML5 ha de incorporar el atributo ***id="identificadorÚnico"*** y en la hoja de estilo el selector correspondiente ha de ser indicado así: ***#identificadorÚnico*** (notar la almohadilla inicial). De esta manera, sólo ese elemento en particular se verá afectado por los estilos declarados bajo ese selector.

Un ejemplo básico de código HTML5 que incluye estilos CSS podría ser éste. Para poder visualizarlo en un computador, el lector debería abrir su editor de textos favorito y escribir en él todo este código, a continuación guardarlo en un fichero con extensión ".html" (dentro de cualquier carpeta del sistema) y, finalmente, hacer doble clic sobre ese fichero: se debería abrir entonces el navegador predeterminado mostrando la página web.

```
<!DOCTYPE html>
<html>
    <head>
        <style type="text/css">
        p, h1       { color:red; background-color:blue; }
        #parrafo1 { text-transform:uppercase; }
        .centrado  { text-align:center; }
        </style>
    </head>
    <body>
        <p>Este párrafo tiene color de letra rojo y fondo azul</p>
        <p id="parrafo1">Éste está en mayúsculas (y también es de color rojo con fondo azul)</p>
        <p class="centrado">Éste está centrado (y también es de color rojo con fondo azul)</p>
        <h1 class="centrado">Este título también está centrado y es de color rojo con fondo azul</h1>
    </body>
</html>
```

Existen muchas propiedades CSS relacionadas con todo tipo de aspectos relacionados con la presentación de contenido: fondos, fuentes de letra, márgenes, tamaños, bordes, visibilidad, alineamiento y posicionamiento de elementos, etc, etc son sólo algunas de las características que se pueden manipular mediante hojas de estilo. Tal como se puede ver, el estándar CSS es muchísimo más rico y extenso de lo que se ha vislumbrado en las líneas anteriores, pero no hay tiempo material para ahondar más en este lenguaje. Si se desea profundizar en el conocimiento de este "lenguaje", además de la especificación oficial recomiendo la consulta del estupendo tutorial http://www.w3schools.com/css

---

Nota sobre el lenguaje Javascript

No está de más mencionar, finalmente, que en el proceso de desarrollo de una página web, además del lenguaje HTML5 (necesario para definir su contenido y estructura) y de las hojas CSS (necesarias para definir su aspecto y formato), normalmente interviene una tercera tecnología: el lenguaje Javascript, responsable de dotar a las páginas web de capacidades avanzadas (como el uso de variables, condicionales, bucles, etc) propias de un lenguaje de programación convencional. Todos los navegadores incorporan un intérprete de Javascript encargado de "ejecutar" las partes escritas en este lenguaje que estén incrustadas dentro del código HTML general de las páginas. Gracias al lenguaje Javascript podemos dotar a las páginas web de un comportamiento "dinámico" (es decir, podemos hacer que su contenido no sea fijo sino que pueda variar en cada visualización dependiendo del resultado de la "ejecución" del código Javascript pertinente, la cual puede ser disparada automáticamente o bien "bajo demanda" a partir de detectar el navegador determinados eventos provocados por el propio visitante de la página, como un clic o desplazamiento de ratón, una selección de un elemento, etc, etc), mostrándose así con un comportamiento interactivo (es decir, bidireccional) y mucho más atractivo para el visitante que no el de una simple página web inerte que solo muestre contenido unidireccionalmente. Si se desea iniciarse en el conocimiento de este lenguaje (incluyendo también el aprendizaje del DOM), recomiendo consultar los fabulosos tutoriales disponibles en el sitio http://www.w3schools.com/js .