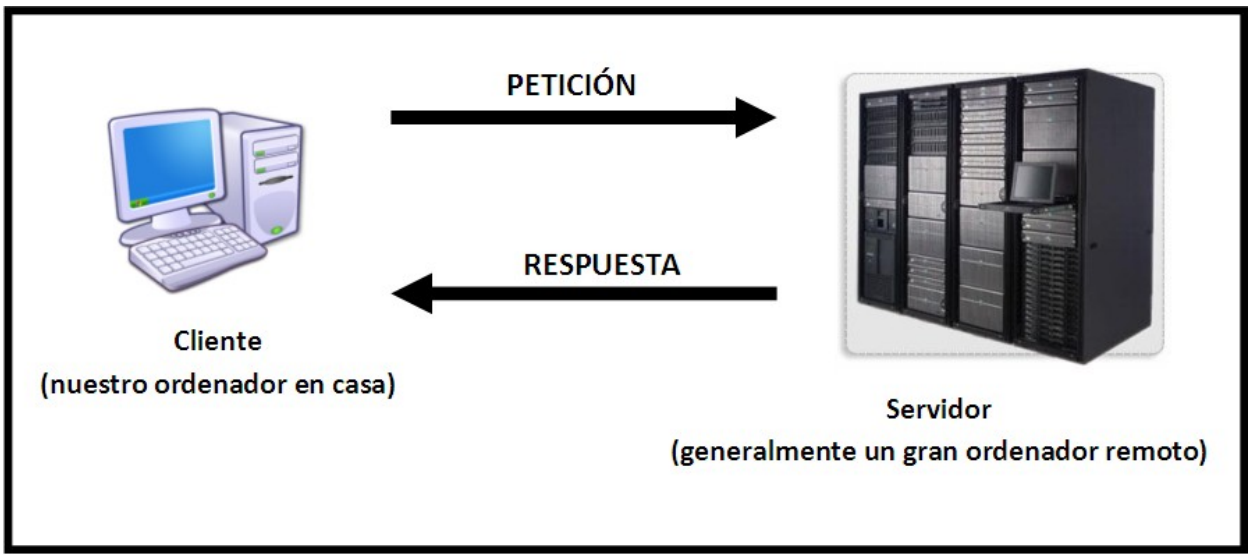
INTRODUCCIÓN: INTERNET Y CÓMO SE TRANSMITE LA INFORMACIÓN.

Vamos a explicar qué ocurre cuando escribimos una dirección web en nuestro navegador (el programa que usamos para ver páginas web), desde que tecleamos la dirección hasta que vemos la página solicitada en nuestro monitor. Por ejemplo, cuando escribimos la dirección http://www.ieselrincon.org

Primero escribimos la dirección o URL del sitio web en nuestro navegador. A continuación y sin que nosotros nos demos cuenta, nuestro navegador solicita la página web al servidor que alberga el sitio ieselrincon.org. Acto seguido, el servidor envía de vuelta los datos a nuestro ordenador a través de Internet. Finalmente, nuestro navegador interpretará los datos, mostrando el resultado en la pantalla de nuestro ordenador.



En el gráfico anterior podemos ver cómo desde el cliente (nuestro ordenador en casa) se envía una petición al servidor y cómo éste devuelve una respuesta con los datos. Cuando decimos http nos referimos a un protocolo de transmisión de datos: esto simplemente son una serie de reglas que usan los ordenadores para comunicarse entre sí a través de internet. Sobre este protocolo no nos hace falta comentar nada más, no vamos a estudiarlo porque no nos resulta necesario para el objetivo del curso.

A partir de aquí, el navegador que estemos utilizando interpretará esos datos y los mostrará en la pantalla. Es por ello, que podemos obtener visualizaciones distintas para cada navegador, porque son éstos los que interpretan los datos obtenidos que son siempre los mismos para una misma petición de página web. Por ejemplo, podemos usar como navegador Internet Explorer ó Firefox, y según usemos uno u otro obtener resultados distintos porque cada navegador interpreta la información de una manera distinta.

Podemos decir, por explicarlo de forma sencilla, que el navegador es aquello que transforma los datos obtenidos para que una persona pueda visualizarlos en su monitor.

NAVEGADORES MÁS USADOS

MOZILLA FIREFOX		Mozilla Firefox es un navegador web libre y de código abierto, desarrollado por la Fundación Mozilla que es una organización sin ánimo de lucro dedicada a la creación y difusión de software libre. Este navegador es uno de los más usados por los programadores web ya que cumple la mayoría de los estándares web conocidos y porque proporciona herramientas muy útiles para el desarrollo y corrección del código informático que hay detrás de las páginas web.
INTERNET EXPLORER		Conocido comúnmente como IE, es un navegador web desarrollado por Microsoft para el sistema operativo Microsoft Windows desde 1995. Este navegador es uno de los más usados por los usuarios ya que viene por defecto en el sistema operativo Windows.
GOOGLE CHROME		Google Chrome es un navegador web desarrollado por Google. Nació en el año 2008, lo que lo convierte en uno de los navegadores más jóvenes del mercado. Es uno de los navegadores más rápidos y ligeros que existe. También es muy usado por los programadores ya que cumple los estándares web e incluye herramientas interesantes.
SAFARI EXPLORER		Safari es un navegador web de código cerrado desarrollado por Apple (fabricante de los famosos ordenadores Macintosh, móviles iPhone, etc.). Está disponible para ordenadores o dispositivos móviles que usan el sistema operativo de Macintosh y también para Microsoft Windows. Es el navegador que nos encontraremos en cualquier ordenador de Apple.

¿QUÉ ES Y PARA QUÉ SIRVE HTML?

HTML es el lenguaje que se emplea para el desarrollo de páginas de internet. Está compuesto por una serie de etiquetas que el navegador interpreta y da forma en la pantalla. HTML dispone de etiquetas para imágenes, hipervínculos que nos permiten dirigirnos a otras páginas, saltos de línea, listas, tablas, etc.

Podríamos decir que HTML sirve para crear páginas web, darles estructura y contenido. Un ejemplo sencillo de código HTML podría ser:

```
<html>

    <body>

        <p>Esto es un párrafo. Bienvenidos a esta página
        web.</p>

    </body>

</html>
```

Este ejemplo está formado por 3 etiquetas HTML. Como podemos observar cada una de las etiquetas debe acabar con su homóloga de cierre. En este caso la etiqueta <html> debe cerrarse con </html>, la etiqueta <body> con </body> y la etiqueta con .

Hay muchas más etiquetas que veremos más adelante pero nos debe quedar claro que por cada etiqueta que abramos, deberemos incluir la correspondiente etiqueta de cierre. Así conseguiremos un código HTML bien formado y que los navegadores puedan interpretar sin ambigüedad.

Este sencillo ejemplo mostraría por pantalla lo siguiente.

¿Qué ocurriría si una etiqueta que abramos no tiene su correspondiente cierre? Digamos que se trataría de un código HTML mal construido, y los navegadores esto lo pueden interpretar de distintas maneras. Quizás nos muestren la página tal y como esperábamos sin aparente error. Quizás nos muestren una página de error o se quede en blanco el navegador. Nuestro objetivo ha de ser siempre construir páginas HTML bien estructuradas y sin ambiguedades, es decir, hacer un correcto uso del lenguaje para que los navegadores puedan saber exactamente qué es lo que pretendemos mostrar.

ALGO DE HISTORIA

HTML nació públicamente en un documento llamado HTML Tags (Etiquetas HTML), publicado por primera vez en Internet por Tim Berners-Lee en 1991. En esta publicación se describen 22 etiquetas que mostraban un diseño inicial y relativamente simple de HTML. Varios de estos elementos se conservan en la actualidad. Otros se han dejado de usar, y muchos otros se han ido

añadiendo con el paso de los años. De esta manera, podemos hablar de que han existido distintas versiones de HTML a lo largo de la historia de internet. Nosotros vamos a trabajar con el HTML estándar actual, que es el utilizado por los navegadores y páginas web de hoy en día. Sin embargo, no vamos a prestarle atención a las versiones y especificidades de cada versión, ya que el objetivo de este curso es aprender **los fundamentos de HTML** y entender cómo funciona, no conocer la sintaxis o especificidades de una versión concreta. ¿Por qué no le damos importancia a la versión? Porque una persona que cuenta con los fundamentos y comprensión básica sobre el lenguaje es capaz de adaptarse a las características particulares de una versión sin problema. En cambio, centrarse en los detalles de una versión sin conocer los fundamentos hará que no tengamos capacidad para comprender lo que hacemos ni para adaptarnos a los continuos cambios que tienen lugar en el ámbito de los desarrollos web.

¿ES HTML UN LENGUAJE DE PROGRAMACIÓN?

En principio diremos que HTML no es un lenguaje de programación, aunque de forma coloquial muchas veces se oigan referencias a HTML como si lo fuera. HTML es un lenguaje de etiquetas. Estas etiquetas (tag) HTML comunican al navegador cuál es la información a mostrar por pantalla, además del formato de dicha información.

Es por ello que no puede definirse como un lenguaje de programación, sino como un sistema de etiquetas. Veámoslo con un ejemplo.

EJEMPLO PARA ENTENDER EL CONCEPTO DE LENGUAJE DE ETIQUETAS FRENTE A LENGUAJE DE PROGRAMACIÓN

Si analizamos el siguiente algoritmo realizado el lenguaje de programación Java, podremos observar cómo una cosa tan simple como es ejecutar un proceso para escribir los números del 1 al 10, no es posible en HTML. Esto es debido a que HTML no es un lenguaje de programación y no dispone de las sentencias básicas de la programación, como instrucciones para repetir un proceso o, elegir si realizar un proceso u otro en función de una circunstancia que se esté produciendo.

Lenguaje	Código	Salida por pantalla

Java	public class MuestraDelUnoAlDiez {
	
	public static void main (String[] args) {
	
	for(int i=1; i <= 10; i++) {
	
	System.out.println(i);
	
	}
	
	}
	
	}
		1
		
		2
		
		3
		
		4
		
		5
		
		6
		
		7
		
		8
		
		9

		10
HTML	<html> <body> 1 2 3 4 5 6 7 8 9 10 </body> </html>	1 2 3 4 5 6 7 8 9 10

Como podemos observar, en el ejemplo anterior, HTML no tiene la capacidad de contar y debemos escribir nosotros todo lo que queremos que salga por pantalla. Sin embargo, vemos cómo en Java podemos indicar que cuente del 1 al 10 y que lo muestre por pantalla sin escribir elemento a elemento lo que queremos visualizar.

Clásicamente se dice que los lenguajes de programación incluyen tres capacidades básicas de generar flujos de procesos: la secuencial (secuencias de instrucciones), la condicional (capacidad para tomar decisiones o ejecutar un proceso u otro en función del valor de uno o varios parámetros) y la de repetición (capacidad para repetir un proceso un cierto número de veces). Los lenguajes clásicos como C, C++, Java, C#, Visual Basic, Fortran, etc. cuentan con estas capacidades. HTML no cuenta con ellas, no porque sea mejor ni peor sino porque es una cosa distinta.

En resumen, podríamos decir que HTML no es un lenguaje de programación, es un lenguaje de maquetación web o lenguaje de etiquetas destinado a crear estructuras de documentos HTML.

¿CUÁLES SON LAS VERSIONES DE HTML?

HTML fue desarrollado originalmente por Tim Bernes-Lee pero debido al rápido crecimiento de la web, surgió la necesidad de crear un estándar para que tanto los programadores como los navegadores pudieran basarse en unas mismas normas para escribir HTML. Cada versión de HTML establece unas normas respecto a cuáles son las etiquetas válidas y cómo se deben escribir.

Los estándares oficiales HTML son el HTML 2.0, el HTML 3.2, el HTML 4.0 y el HTML 4.01, aunque actualmente se trabaja en el HTML 5. El HTLM 5 ya está empezando a ser usado aunque todavía no es una especificación oficial. El XHTML, una forma más avanzada del HTML que se suponía iba a sustituir a éste, va a quedar integrado dentro del HTML 5.

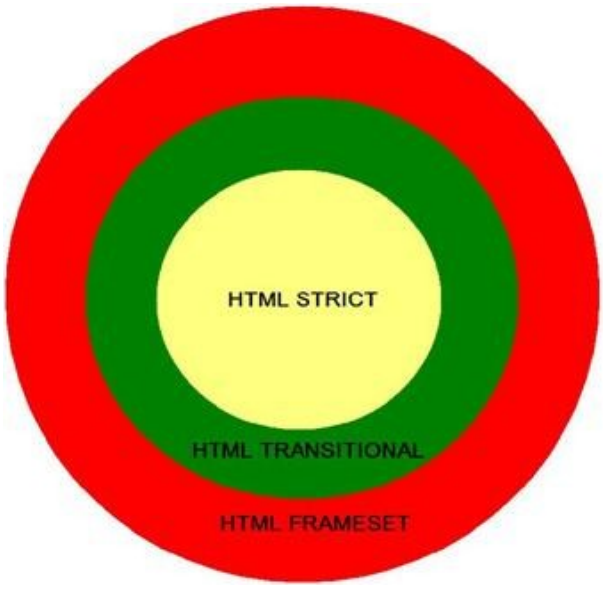
Evolución de HTML:

HTML 2.0	En 1995 se publica el estándar HTML 2.0. A pesar de su nombre, HTML 2.0 es el primer estándar oficial de HTML, es decir, el HTML 1.0 no existió como estándar. HTML 2.0 no soportaba tablas. Se simplificaba al máximo la estructura del documento para agilizar su edición, donde la declaración explícita de los elementos body, html y head es opcional.
HTML 3.2	La versión HTML 3.2 se publicó en 1997 y es la primera recomendación de HTML publicada por el W3C (Consorcio internacional). Esta revisión incorporó los últimos avances de las páginas web desarrolladas hasta 1996, como applets de Java y texto que fluye alrededor de las imágenes.
HTML 4.01	La última especificación oficial de HTML se publicó en diciembre de 1999 y se denomina HTML 4.01. Desde la publicación de HTML 4.01, el W3C se centró en el desarrollo del estándar XHTML. Por este motivo, en el año 2004, las empresas Apple, Mozilla y Opera mostraron su preocupación por la falta de interés del W3C en HTML y decidieron organizarse en una nueva asociación llamada WHATWG (Web Hypertext Application Technology Working Group) que comenzó el desarrollo del HTML 5, cuyo primer borrador oficial se publicó en enero de 2008. Debido a la fuerza de las empresas que forman el grupo WHATWG y a la publicación de los borradores de HTML 5.0, en marzo de 2007 el W3C decidió retomar la actividad estandarizadora de HTML, y actualmente W3C está trabajando para el lanzamiento del estándar HTML 5.0, dentro del cual ha decidido integrar el XHTML.
HTML 5.0, HTML 5.1 y HTML 5.2	El consorcio internacional W3C marcó las siguientes fechas para liberación de los estándares de especificación: 2014/2015 para HTML 5.0, 2016 para HTML 5.1 y 2019 para HTML 5.2

Hasta el momento la versión de HTML más utilizada ha venido siendo la 4.01. Esta versión fue definida por la W3C (Comité Internacional que define los estándares web) hace varios años. Actualmente ya está disponible la nueva versión de HTML, denominada HTML 5. Esta versión ya se está usando de modo experimental y se espera que se imponga como estándar en los próximos años.

Además de cada versión, cada una tiene variantes (digamos que "distintas formas"). Cuando escribimos un documento HTML debemos indicar en una línea inicial qué versión y variante es la que estamos usando de forma que cualquier persona que lea ese documento HTML sepa qué versión y variante se ha empleado. Las variantes del HTML 4.01 son:

HTML 4.01 Strict	En este tipo de documentos podemos usar etiquetas HTML 4.01, pero no se aceptan etiquetas obsoletas, es decir, etiquetas propias de versiones más antiguas. Es la versión que si usamos en teoría nos debería dar un resultado óptimo en los navegadores más modernos. Esto no siempre es así, como explicaremos un poco más adelante.
HTML 4.01 Transitional	En este tipo de documentos se pueden usar todas las etiquetas de todas las versiones de HTML. Usar esta variante de HTML plantea el interrogante de si es correcto permitir el uso de etiquetas obsoletas que podrían dejar de funcionar en las proximas versiones de los navegadores. Sin embargo, este es el estándar más usado, porque combina la posibilidad de usar etiquetas más antiguas y etiquetas más modernas, de forma que podamos aspirar a una mejor visualización en la mayor parte de los navegadores.
HTML 4.01 Frameset	Este tipo de documentos tiene soporte para frames. Los frames son unos marcos a modo de pequeñas subventanas dentro de una misma página web que se usaban mucho hace unos años pero que hoy en día se usan cada vez menos. Este tipo de HTML podemos considerarlo anticuado, porque hay otras formas de diseñar páginas web sin frames más modernas y útiles que nos permiten obtener el mismo resultado de forma más eficiente.



En la figura podemos ver cómo el ser más estrictos supone que tengamos que usar un menor número de etiquetas.

¿CUÁL ELIJO, Y CÓMO?

No te preocupes demasiado por utilizar una versión "correcta y concreta" sino por crear páginas web que se vean bien. Para ello debes aprender cómo se construye y cuál es la lógica del HTML, más que una versión concreta de éste. Ten en cuenta que hay diversidad de versiones y que no todos los navegadores se ciñen a los estándares, con lo cual no tiene demasiado sentido preocuparse por ceñirse a una versión. Te puede resultar un poco extraño, pero cuando adquieras experiencia en desarrollos web comprobarás que las normas para los desarrollos web no están 100 % claras.

El XHTML es un lenguaje que va a quedar integrado dentro de HTML 5.

En un archivo HTML debemos indicar qué versión y variante estamos usando. Para indicar ésto hay que poner una línea al principio de la página web (del archivo donde está el contenido). No es una etiqueta, por tanto es algo rara y no hay que cerrarla ni ponerla en minúsculas. Eso sí, debemos ponerla en todos nuestros documentos.

Para HTML 4.01 Strict escribiríamos:

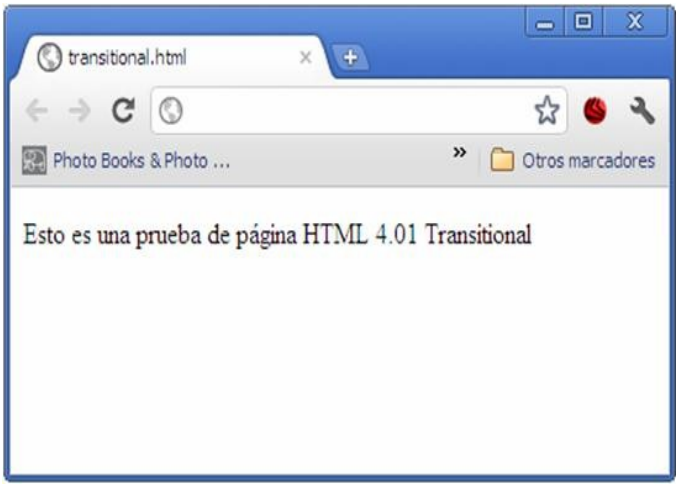
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
```

Para HTML 4.01 Transitional (recomendado) escribiríamos:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
```

EJEMPLO

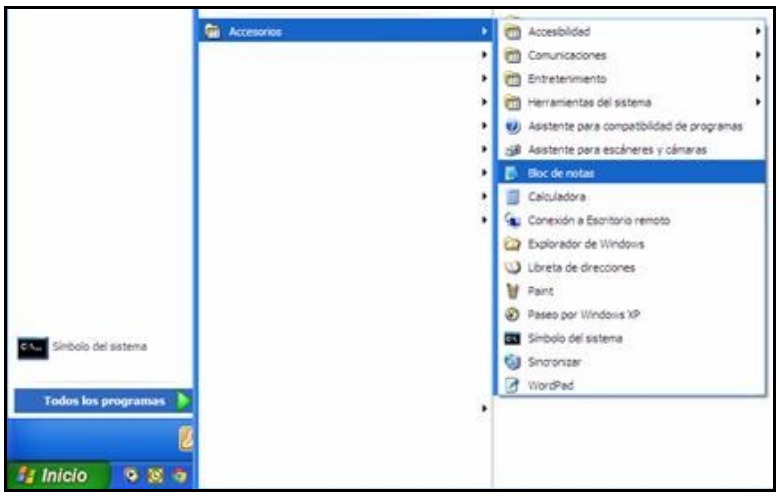
```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

    <body>

        <p>Esto es una prueba de página HTML 4.01 Transitional</p>

    </body>

</html>
```



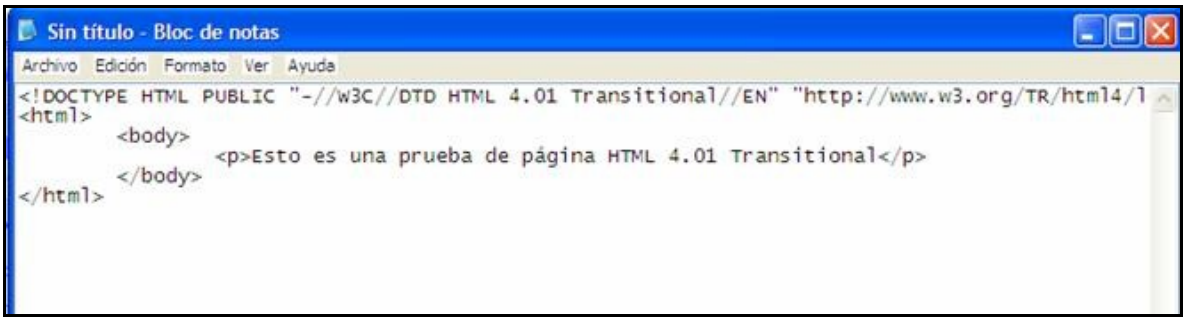
Vista en el navegador Google Chrome que obtendríamos para el documento HTML anterior.

Con este sencillo ejemplo creamos una pagina HTML 4.01 Transitional. Para visualizar esta simple página web procedemos de la siguiente manera: creamos un archivo con el editor de texto que tengamos a nuestra disposición, en nuestro caso utilizaremos el bloc de notas de Windows pero puede ser cualquier editor de texto.

Para abrirlo nos vamos a Inicio --> Todos los programas --> Accesorios --> Bloc de notas



Escribimos en el bloc de notas el código del ejemplo anterior como un simple texto.



A continuación en el menú Archivo elegimos la opción Guardar como… e indicamos que queremos guardar el archivo en el directorio raíz de la unidad C con el nombre ejemplo.html

Una vez realizado todos estos pasos, abrimos el archivo creado haciendo doble click sobre él desde el explorador de archivos de Windows. Si tenemos un navegador web instalado se nos abrirá automáticamente la página web que hemos creado en el ejemplo.

¿CÓMO ESCRIBIR CÓDIGO HTML Y CREAR PÁGINAS WEB?

Los requisitos principales y fundamentales, para escribir código HTML y crear páginas web, son básicamente dos: saber HTML (esto lo conseguirás siguiendo este curso) y un editor de texto . Hay muchos profesionales que crean sus páginas en Dreamweaver, usando Flash u otros programas o tecnologías.

Esta forma de crear páginas web tiene a favor la fácil creación de éstas pero, si quieres hacer páginas web de calidad y tener un control total sobre el código generado, lo primero es saber HTML sin más.

Para crear páginas web obviamente necesitas un ordenador y un navegador (Explorer, Firefox, Chrome o similar) instalado. ¿Necesitamos conexión a internet para crear páginas web? La respuesta es que no: podemos crear nuestras páginas en nuestro propio ordenador y a posteriori subirlas a un servidor remoto para que estén accesibles en internet desde cualquier parte del mundo. En este curso vamos a trabajar inicialmente en local (es decir, en nuestro propio ordenador sin necesidad de conexión a internet), y más adelante veremos cómo subir una web a un servidor.

Ahora indicaremos unos sencillos consejos que deberías tener en cuenta antes de crear tus páginas web.

*** Tomar ideas.**

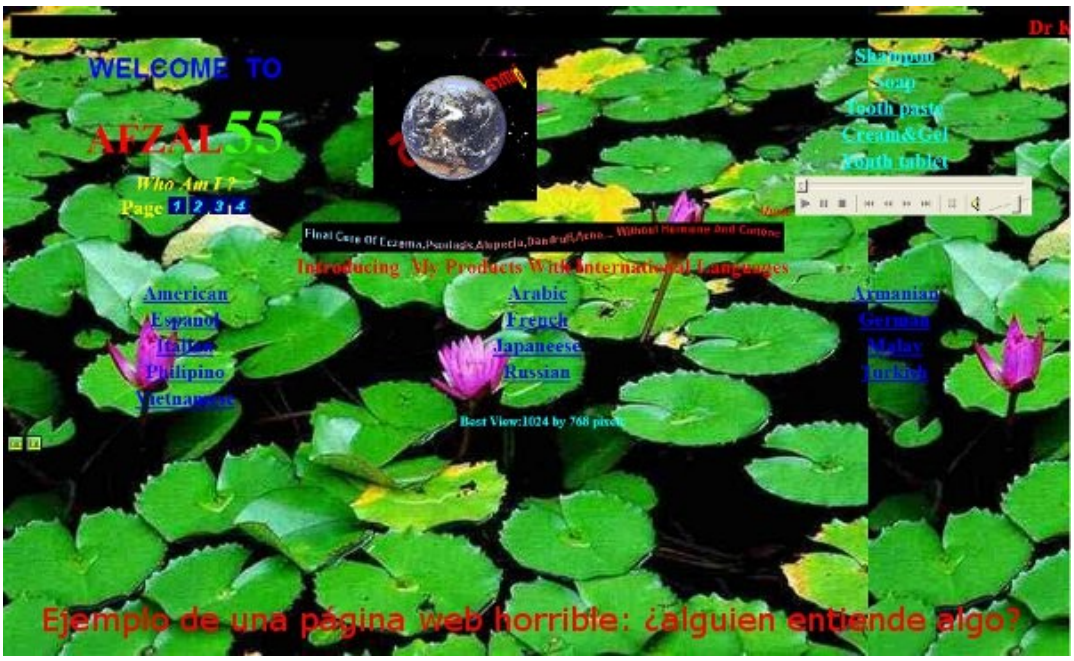
Ojo, tomar ideas no es copiar contenidos con derechos de propiedad intelectual, no debes reproducir el trabajo de otros sin valorar si estás vulnerando el derecho de propiedad intelectual. Sin embargo, es importante inspirarte en otros sitios, buscar contenidos que te puedan servir, combinaciones de colores que se vean bien, y diseños organizados que te puedan ser útiles.

*** No uses toda la gama de colores en tu página web.**

No crees una página web que parezca un arcoiris: inicialmente te puede parecer curiosa, pero un usuario que visite tu página web, se sentirá molesto con demasiados colores.

Por ejemplo, es preferible utilizar el clásico fondo blanco y texto negro o azul, que el fondo rosa con puntos morados y el texto verde fosforescente. Cuida que los colores de tu página tengan una buena combinación, y que hagan fácil la lectura.

Ejemplo mal diseño, colores e imágenes:



*** Se ve mejor si cabe en la pantalla.**

Procura que tu diseño esté basado en el tamaño de la pantalla, no busques escribir una novela completa en la página inicial de tu web. Es preferible que las páginas no tengan scroll horizontal y que el scroll vertical no sea demasiado largo.

*** Administra tus imágenes.**

Tu página se verá más atractiva si usas imágenes. No obstante, debes usarlas de una forma moderada porque un excesivo número de imágenes puede ser también perjudicial si no permite una buena lectura del contenido.

*** Haz que tu sitio sea fácil de navegar.**

Por muy obvia que parezca esta recomendación, debes dividir la información en diferentes secciones y ubicar los enlaces a las mismas donde la gente espere encontrarlos.

Hay algunos sitios donde sólo es posible desplazarse hacia delante o hacia atrás. En realidad, eso no es "navegar". Es necesario que el visitante pueda elegir en todo momento qué sección quiere visitar y en qué orden: por ejemplo tener un menú es algo generalmente aconsejable. No te olvides de incluir en todas las páginas un enlace a la página de inicio (la primera página) para facilitar la navegación.

Ejemplo de menú:

También es interesante incluir el logotipo de tu sitio en todas las páginas, porque no siempre el visitante ingresa al sitio por la página de inicio; si proviene del enlace de un buscador, es probable que ingrese por cualquier sección, en ese caso, el logotipo en todas las páginas ayuda a ubicar al posible visitante donde está.

*** Mantén tu sitio actualizado.**

Nada es más desagradable que volver a un sitio tiempo después de una visita y no encontrar ningún cambio. Realizar cambios frecuentes crea en los visitantes la idea de que el contenido es valioso y vale la pena darse una vueltecita a menudo.

*** Prográmate un esquema de actualización y cúmplelo.**

Renueva tus promociones, agrega más información sobre nuevos productos, servicios o artículos, incluye testimonios de clientes o usuarios satisfechos, usa tu imaginación.

También puedes incluir la fecha de la última actualización en un lugar visible de tu página de inicio, o bien mantener un apartado donde se vea que los contenidos están actualizados.

*** Promociona tu web: on-line y off-line.**

Envía tu sitio a los principales motores de búsqueda y directorios (google, yahoo, etc.). Dedica tiempo a realizar un trabajo a conciencia, el mismo se justificará plenamente con un mejor posicionamiento en los buscadores.

Después de todo ¿de qué te sirve tener un sitio en Internet si nadie lo encuentra?

En este curso aprenderemos a utilizar palabras claves (keywords) que describan tu negocio lo mejor posible. Aquí conviene situarse del lado de quien busca, ¿qué keywords utilizará tu posible cliente para buscar un sitio de tus características? Haz una lista, pregunta a amigos/as y conocidos.

No te olvides de incluir tu dirección (http://www.tusitio.com) en toda papelería y comunicación que emitas: tarjeta comercial, papelería, facturas, remitos, recibos, folletos, faxes, bolsas, publicidad, etc.

*** Información para tus clientes.**

Incluye un enlace al pié de cada página para que puedan comunicarse contigo. En este curso aprenderemos cómo hacerlo. El agregar información con la dirección física, teléfono completo (con el código del país), fax, etc, ayuda a crear confianza. Después de todo tú y tu web existen en el "mundo real" no son un ente imaginario perdidos en el ciberespacio.

Cuando te sea posible, incluye un campo donde tus usuarios puedan ingresar el email para recibir un boletín o newsletter con novedades sobre tus productos, esto ayuda a crear confianza y lealtad a tu página. Otra opción es la suscripción mediante RSS ó Atom, que son servicios para poder recibir las novedades que se van publicando en tu página.

Recuerda que ahora puedes utilizar las redes sociales para promocionar tu web o empresa online: Facebook, YouTube, Slideshare, Linkedin, etc.

Todas estas ideas pueden ayudarte a tener un sitio web de calidad y bien posicionado en los buscadores. No tienes por qué cumplirlos todos, ni son todos los puntos a tener en cuenta para que tu sitio esté el primero en los buscadores, pero pueden resultar útiles. No te preocupes si ahora algunos conceptos no te quedan claros porque a medida que trabajes en desarrollos web irás adquiriendo una visión más completa.

CONCEPTOS BÁSICOS. ESTRUCTURA BÁSICA DE UNA PÁGINA HTML. ETIQUETAS HEAD, BODY.

Vamos a explicar conceptos básicos de HTML. En primer lugar veremos cuál es la estructura básica que toda página HTML debe cumplir. Para ello introduciremos las etiquetas HTML, HEAD y BODY, y sus respectivas etiquetas de cierre.

Toda página web viene definida con la siguiente estructura básica (recuerda que la primera línea es una etiqueta relativa a la versión/variante de HTML que declaramos usar y que esta primera etiqueta no es estrictamente necesaria. Sirve únicamente para indicar qué estándar de HTML es el que declaramos usar).

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

    <head>

        <title>Título de la página web</title>

        ...

    </head>

    <body>

        Cuerpo de la página web

    </body>

</html>
```

Vamos a analizar más detenidamente las distintas secciones que componen la página.

La etiqueta <html> define que se trata de código HTML.

CABECERA DEL DOCUMENTO (HEAD)

La cabecera del documento es la sección comprendida entre <head> y </head>. En ella se debe encontrar, obligatoriamente, el título (entre las etiquetas <title> y </title>).

El **título** de la página debe describir su contenido por ejemplo: <title>Manuales y tutoriales sobre programación</title>

No sería adecuado usar el título <title>Página de Inicio</title> porque éste no dice nada por sí solo. Debemos usar títulos que sean descriptivos relativos al contenido de la página.

Además de la etiqueta title dentro de la sección de cabecera se suelen incluir otras etiquetas. La siguiente tabla muestra un resumen de elementos que pueden ir dentro de la etiqueta head.

Etiqueta en cabecera	Función	¿Es obligatoria?
<title>	Da un título al documento HTML	Sí
<base>	Define ruta de acceso	No
<link>	Define archivos vinculados	No
<meta>	Define metadatos como descripción y palabras clave	No
<script>	Delimita scripts incluídos	No
<style>	Delimita definición de estilos	No

Comentaremos brevemente estas etiquetas a continuación.

Etiqueta <base>. Sirve para indicar la URL base en caso de especificarse URLs relativas dentro de la página web. Por ejemplo <base href="http://ww.ieselrincon.org/images/" target="_blank"> haría que si escribimos como ruta de una imagen "logo.png" dicha ruta sea en realidad "http://www.ieselrincon.org/images/logo.png"

Etiquetas <link>. Sirven para indicar que el documento html está relacionado con otro archivo o recurso externo. Por ejemplo <link rel="stylesheet" type="text/css" href="estilos.css"> sirve para indicar que el documento HTML está vinculado al archivo estilos.css

Etiqueta <style>. Sirve para incluir estilos CSS que permiten dotar de colores, bordes, imágenes de fondo, etc. a los elementos de la página web.

Etiquetas <meta>. Sirven para incluir información que no se muestra como parte de la página web pero sirve para informar de características de la página web, como su descripción breve y sus palabras clave.

Ejemplo:

<meta name="description" content="Didáctica y divulgación de la programación">

<meta name="keywords" content="didáctica, divulgación, programación, aprender">

En este caso las etiquetas le indican a los buscadores el contenido de nuestras páginas (description) y algunas palabras clave (keywords) para su localización. Esto es muy útil para que nuestra página aparezca en los buscadores (google, bing, yahoo, etc.).

Etiquetas <script>. Sirven para incluir código en lenguajes de script.

Dentro de la cabecera en muchas páginas se incluye código en JavaScript, que es un lenguaje de programación que los navegadores son capaces de reconocer e interpretar. El código JavaScript se reconoce porque va comprendido entre las etiquetas

```
<script type="text/javascript">
// Aquí iría el código
</script>
```

CUERPO (BODY) DEL DOCUMENTO HTML

El **cuerpo** (body) del documento html es normalmente lo más importante. Es aquí donde debemos colocar el contenido de nuestra página: texto, fotos, etc.

El cuerpo está delimitado por las etiquetas <body> y </body>.

A lo largo del curso iremos estudiando qué elementos pueden existir dentro de la etiqueta body, así como cuál es su sintaxis y organización.

SIGNIFICADO DE DEPRECATED

Cuando se trabaja en programación y desarrollos web veremos que con frecuencia aparece el término inglés “deprecated”. Algunas veces se trata de traducir al español por “deprecado”, pero en realidad este término no existe en español. En su lugar podemos usar “desaprobado” o “no recomendado”.

Al igual que hoy día disponemos de computadoras más potentes y efectivas que hace 20 años por evolución de la tecnología, también podemos decir que los lenguajes evolucionan y cambian.

Así, algunas formas sintácticas o expresiones HTML que se consideraban válidas hace unos años han pasado a considerarse no recomendados para los desarrollos web actuales. Sin embargo, dado la gran cantidad de páginas web que se construyeron usando estas formas del lenguaje anticuadas y a que los programadores no podían adaptarse de un día para otro a los cambios en los lenguajes, en lugar de no permitir estas expresiones, las nuevas versiones de los lenguajes las clasifican como deprecated o no recomendadas.

Un ejemplo de uso **no recomendado** (deprecated) es la inclusión de un atributo bgcolor en la etiqueta body de un documento HTML.

Por ejemplo <body bgcolor=”yellow”> hace que la página web tenga un fondo amarillo mediante el uso de un atributo, bgcolor, que está clasificado como deprecated. Esto significa que no debemos usar el atributo bgcolor porque se ha definido otra forma de dar color de fondo al elemento body de un documento HTML (en concreto mediante el uso de estilos CSS).

Las sintaxis, atributos, expresiones, etc. que están catalogadas como deprecated siguen siendo aceptadas por los navegadores, aunque con el paso del tiempo pasarán de deprecated a “**not supported**”, es decir, no serán reconocidas por los navegadores. Por tanto en ningún caso debemos usarlas.

Dejando claro que no debemos usarlas, sin embargo conocer las formas deprecated puede tener cierto interés. ¿Por qué?

1. Nos permiten conocer la evolución del lenguaje

2. En caso de que tengamos que revisar, corregir o modificar código desarrollado **por otra persona**

donde se usan formas deprecated, sabremos identificar y subsanar los problemas existentes.

La existencia de formas deprecated en los desarrollos web de hoy en día se debe a múltiples motivos: miles de páginas web no se han actualizado a los nuevos estándares y se mantienen como fueron construidas hace años. Muchos programadores siguieron usando formas deprecated (y en algunos casos siguen usando algunas de estas formas). Muchos programas para creación semiautomatizada de páginas web usaban o usan formas **deprecated**.

A lo largo del curso estudiaremos formas deprecated, que fueron muy utilizadas en su día pero cuyo uso no es recomendado hoy en día. La inclusión de estas formas en este curso obedece a ser capaces de reconocer y corregir problemas en desarrollos web con los que nos podamos encontrar. Cuando un elemento esté considerado como deprecated lo indicaremos explícitamente.

BODY BGCOLOR (DEPRECATED)

Atributo que fue muy usado en el pasado para especificar el color de fondo de la página. El color se define en base hexadecimal de la siguiente forma (#rrggbb) en el orden rojo, verde, azul (Red, Green, Blue). Es decir, cada color se define con el carácter # seguido de 6 letras ó números. Cada combinación de letras o números da lugar a distintos colores. También se puede usar el nombre en inglés de los colores predefinidos en los navegadores (red, blue, green, etc.).

Sintaxis (deprecated): <body bgcolor=”#0000FF”> o <body bgcolor=”blue”>

En este ejemplo el color azul vemos que lo podemos poner tanto con su código como simplemente escribiendo blue, porque es un color básico. Los colores no básicos sólo podremos indicarlos como código hexadecimal. Una buena ayuda para la seleccion de colores con #rrggbb la puedes encontrar escribiendo en un buscador como Google, Yahoo o Bing el texto “colores hexadecimales” y entrando a cualquiera de las páginas que te permiten seleccionar un color y te dicen cuál es su código.

Recordar: bgcolor es un atributo de body que no debe usarse por estar deprecated o not supported.

BODY TEXT (DEPRECATED)

Atributo que fue muy en el pasado para definir el color del texto. Su formato es el mismo que el de bgcolor. Si no se pone nada es negro.

Sintaxis (deprecated): <body text=”#0000FF”> o <body text=”blue”>

Recordar: text es un atributo de body que no debe usarse por estar deprecated o not supported.

BODY BACKGROUND (DEPRECATED)

Atributo que fue muy usado en el pasado para especificar la ruta y nombre de archivo (URL) de la imagen (formato GIF, JPG o PNG habitualmente) que será usada como fondo del documento. Esta se verá como mosaico para cubrir toda la zona de visualización del navegador si es pequeña (lo habitual era poner como fondo una imagen pequeña y dejar que se repitiera, porque así la página carga más rápido).

Sintaxis tipo (deprecated): <body background="ruta/archivo.gif">

Por ejemplo: <body background="http://www.aprenderaprogramar.com/images/BgTexture.jpg">

Cuando se trabaja en desarrollos web es conveniente especificar la ruta de modo relativo, esto quiere decir que si cambiamos el directorio completo donde están nuestras páginas (nuestros archivos html), desde C:\ a C:\webs por ejemplo, la ruta especificada debe seguir siendo válida. En el primer ejemplo que pusimos, la ruta utilizada es absoluta. Dentro de un servidor también se pueden usar rutas relativas, el formato sería similar a éste: <bodybackground="images/BgTexture.jpg">

Como ves en este ejemplo no aparece el nombre del dominio. De esta forma si cambiamos el dominio, las rutas de las imágenes seguirán siendo correctas.

Veamos otro ejemplo. Supongamos la siguiente estructura de directorios y archivos:

- APR2HTML/pagina1.html: el archivo denominado pagina1.html está dentro de la carpeta APR2HTML.
- APR2HTML/gifs/fondo.gif
- APR2HTML/ejemplos/pagina2.html: el archivo denominado pagina2.html está dentro de una subcarpeta denominada ejemplos, que se encuentra dentro de la carpeta APR2HTML.

Si desde la página pagina1.html queríamos poner como fondo la imagen fondo.gif, que se encuentra en el directorio gifs, se debía poner: <bodybackground="gifs/fondo.gif">

Si desde la página pagina2.html queríamos poner como fondo la imagen fondo.gif, que se encuentra en el directorio gifs, se debía poner: <bodybackground="../gifs/fondo.gif">

Fíjate en ../ pues es lo que indica al navegador que debe acudir al directorio superior. Si quisiéramos subir dos niveles, por ejemplo si tenemos la página en APR2HTML/ejemplos/miweb/pagina3.html escribiríamos "../../gifs/fondo.gif". Aunque el atributo background está deprecated, lo dicho para las rutas es útil cuando trabajes con desarrollos web actuales.

MÁRGENES: LEFTMARGIN, TOPMARGIN, RIGHTMARGIN Y BOTTOMMARGIN (DEPRECATED)

Además de bgcolor, text y background, también se usaban atributos para definir márgenes, todos ellos clasificados como deprecated hoy día.

Para especificar los márgenes se utilizaba el atributo margin, con su correspondiente indicación indicadora de lateral afectado. Así se utilizaba "leftmargin" para el margen izquierdo, "topmargin" para el margen uperior, "rightmargin" para el margen de la derecha y "bottommargin" para el margen inferior.

Los márgenes se suelen medir en pixeles, término inglés que alude a cada uno de los pequeños puntos que conforman la imagen en una pantalla (ten en cuenta que una pantalla tiene unas dimensiones en pixeles de ancho y pixeles de alto). Para establecer márgenes de 10 pixeles en todos los sentidos se escribían expresiones de este tipo (deprecated):

```
<body leftmargin="10px" topmargin="10px" rightmargin="10px" bottommargin="10px"></body>
```

RESUMEN

En el pasado se usaban una serie de atributos para la etiqueta body que permitían dotar el cuerpo de la página de un aspecto o presentación determinada. Prueba en tu navegador estas formas y

comprueba si tu navegador las reconoce. Hoy día el uso de estas formas está considerado una mala práctica, por lo que no deben usarse, aunque sí conocerse por si hubiera que corregir webs donde existan.

En lugar de estos atributos hoy día se usan técnicas CSS.

FORMATOS DE TEXTO EN HTML: NEGRITA, CURSIVA, TACHADO, SUBRAYADO, SUPERÍNDICE, SUBÍNDICE.

Vamos a ver una serie de etiquetas y atributos básicos para la creación de páginas webs. No entraremos a describir una a una todas las etiquetas y atributos de que HTML dispone. Simplemente indicaremos las que han sido más utilizadas en el pasado, aunque algunas se consideren deprecated (no recomendadas).

Las primeras etiquetas que veremos son las que se usaban en el pasado para dar formato al texto. Para ello tenemos una serie de etiquetas que escribimos en HTML envolviendo la palabra o el texto y que transforman ese texto en el formato que nosotros le hayamos querido dar. Algunas de estas etiquetas están no recomendadas (deprecated) por lo que no debemos emplearlas. Otras etiquetas han adquirido un nuevo significado en las últimas versiones de HTML, pero no vamos a entrar a definir este nuevo significado. Debido a su amplia difusión en el pasado conviene conocer los que fueron usos tradicionales de estas etiquetas, a medida que avances en el conocimiento de HTML podrás comprobar cómo para algunas etiquetas se consideran hoy día nuevos significados.

ETIQUETA	USO	OBSERVACIONES
…	Poner texto en negrita	Puede ser sustituido por CSS.
…	Poner texto en negrita	Puede ser sustituido por CSS.
…	Poner texto en cursiva	Puede ser sustituido por CSS.
…	Poner texto en cursiva	Puede ser sustituido por CSS.
<u>…</u>	Poner texto subrayado	Deprecated. Sustituir por CSS.
…	Poner texto más pequeño	Puede ser sustituido por CSS.
<big>…</big>	Poner texto más grande	Puede ser sustituido por CSS.
_…	Poner texto subíndice	Puede ser sustituido por CSS.

[…]	Poner texto superíndice	Puede ser sustituido por CSS.
<strike>…</strike>	Poner texto como tachado	Deprecated. Sustituir por CSS.
<s>…</s>	Poner texto como tachado	Deprecated. Sustituir por CSS.
~~…~~	Poner texto como tachado	Puede ser sustituido por CSS.

Como verás, algunas de las etiquetas que vamos a explicar están obsoletas (deprecated en inglés). Estas etiquetas en principio no deben de ser usadas porque dejarán de existir en las nuevas versiones a partir de HTML 5 y los navegadores es posible que dejen de reconocerlas en un futuro. Los motivos para incluirlas en este curso son:

- Son etiquetas que han sido muy populares en el pasado y te puedes encontrar muchas páginas webs que hacen uso de ellas.

- Son etiquetas reconocidas por prácticamente todos los navegadores actuales.

- Son una buena forma de introducirnos en los lenguajes propios de desarrollos webs desde el punto de vista didáctico. Una vez se entiendan estos conceptos, es más fácil abordar aspectos más avanzados como las hojas de estilo CSS.

Las etiquetas deben rodear al texto. Es decir, la etiqueta debe abrirse y cerrarse, conteniendo el texto o la palabra que queramos transformar en su interior. Por ejemplo:

Este texto aparecerá escrito en negrita. Se pueden combinar diferentes formatos, o sea, diferentes etiquetas. Por ejemplo, si queremos que un texto esté en negrita y en cursiva escribiríamos esto: Este texto aparecerá escrito en negrita y en cursiva.

Cuando combines, ten cuidado a la hora de cerrar las etiquetas. Debes cerrar las etiquetas por orden, de la más interior a la más exterior.

Veamos las etiquetas que hemos citado en la tabla anterior.

NEGRITA

Existen dos etiquetas que hacen que nuestro texto se convierta en negrita. La utilización de cualquiera de ellas es en principio indiferente (aunque pueda atribuírseles un significado diferente a cada una de ellas no vamos a prestarle atención a esto ahora). La primera es la etiqueta y la otra es la etiqueta . Aquí va un ejemplo de código y lo que veríamos en pantalla:

```
Esta palabra la vamos a poner en <b>negrita</b> y esta otra <strong>también</strong>
```

Esta palabra la vamos a poner en **negrita** y esta otra **también**

Normalmente se preferirá usar técnicas CSS en lugar de esta etiqueta, pero es una etiqueta que debemos conocer.

CURSIVA

Para escribir un texto en cursiva se ha utilizado mucho en el pasado la etiqueta (que por supuesto debes cerrarla con la etiqueta). También se ha utilizado la etiqueta . Como en el caso de la negrita, aunque podrían atribuírseles distintos significados no vamos a prestarle atención a esta cuestión ahora. Aquí presentamos un ejemplo:

```
Esta palabra la vamos a poner en <i>cursiva</i> y esta otra <em>también</em>
```

Esta palabra la vamos a poner en *cursiva* y esta otra *también*

SUBRAYADO U (DEPRECATED)

Para que la palabra o el texto quedara subrayado se usó en el pasado el rodearlo con la etiqueta <u> y cerrarlo con su correspondiente etiqueta </u>. Así se subrayaría una frase:

```
<u>Así subrayaríamos una frase importante</u>
```

Así subrayaríamos una frase importante

Está etiqueta está obsoleta (deprecated), lo que significa que ya no se recomienda su uso. Para lograr el resultado deseado se deben usar hojas de estilo CSS como veremos más adelante.

PALABRAS MÁS GRANDES O MÁS PEQUEÑAS

Puede que en una frase queramos destacar un palabra por medio de una variación de tamaño sin necesidad de utilizar los encabezados (los encabezados son etiquetas especiales que explicaremos más adelante). La variación de tamaño se podía conseguir gracias a las etiquetas <big> y <small>. Sus propios nombres en inglés nos indican cuáles eran sus funciones: <big> agrandará el texto y <small> lo disminuirá. No recomendamos su uso ya que las nuevas versiones de HTML no van a admitir esta etiqueta. La modificación del tamaño del texto se debe hacer a través de técnicas CSS.

Cada vez que se escribe una etiqueta big, se hace el texto un punto más grande. Estas etiquetas también se podían combinar, por lo que si escribimos dos veces la etiqueta <big>, haremos crecer la palabra dos puntos. Un ejemplo sería el siguiente:

```
Esta palabra se va a escribir <small>pequeñita</small>, esta se va a escribir <big>más grande</big> y ésta <big><big>más grande aún</big></big>.
```

Esta palabra se va a escribir pequeñita, esta se va a escribir más grande y ésta más grande aún.

SUPERÍNDICES Y SUBÍNDICES

Mediante HTML también podemos escribir expresiones con símbolos matemáticos. Gracias a las etiquetas siguientes podrás escribir subíndices y superíndices fácilmente. La etiqueta <sub> te servirá para escribir un subíndice y <sup> será la etiqueta para un superíndice. Así nos queda un ejemplo como el siguiente:

```
Gracias a estas etiquetas podemos escribir cualquier expresión con símbolos matemáticos como esta: H<sub>2</sub>O o números elevados a potencias 7<sup>3</sup>.
```

Gracias a estas etiquetas podemos escribir cualquier expresión con símbolos matemáticos como esta: H_2O o números elevados a potencias 7^3.

Los subíndices y superíndices con estas etiquetas pueden ser sustituidos por técnicas de CSS, pero muchas personas prefieren usar estas etiquetas.

TEXTO TACHADO

Existen tres etiquetas que se han venido usando para conseguir que un texto quede tachado. Hablamos de las etiquetas <strike>, <s> y <del>. Todas ellas ofrecen el mismo resultado. Aquí presentamos una muestra:

```
Puedo proceder a tachar una palabra <strike>así</strike>, <s>así</s> o <del>así</del>
```

Puedo proceder a tachar una palabra ~~así~~, ~~así~~ o ~~así~~

La etiqueta strike está deprecated, lo que significa que ya no se recomienda su uso. La etiqueta s también fue deprecated, aunque a partir de HTML 5 se ha redefinido su significado. Para lograr el tachado de un texto se recomienda usar técnicas CSS (hojas de estilo) como veremos más adelante.

EJERCICIO

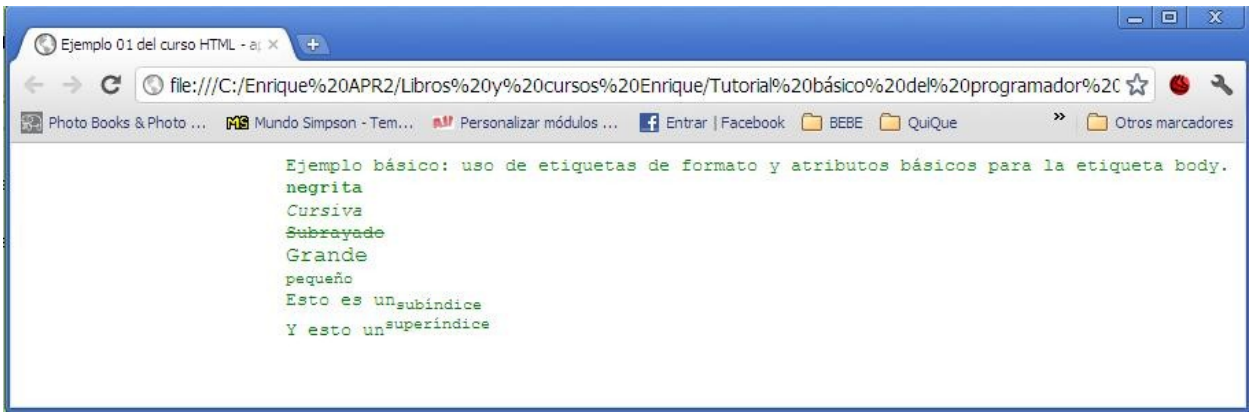
En el siguiente código hay elementos que en las versiones más recientes de HTML se consideran deprecated o not supported. Escribe el código en un editor de textos como el bloc de notas o Notepad++, guárdalo con un nombre como ejemplo.html y visualízalo en tu navegador.

Responde a las siguientes preguntas:

¿Qué etiquetas de las empleadas sería recomendable no utilizar y reemplazar mediante uso de técnicas CSS?

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">

<html>

    <head>

        <title>Ejemplo 01 del curso HTML - aprenderaprogramar.com</title>

    </head>

    <body text="green">

        <pre>

            Ejemplo básico: uso de etiquetas de formato y atributos básicos
            para la etiqueta body.

            <strong>negrita</strong>

            <em>Cursiva</em>

            <del>Subrayado</del>

            <big>Grande</big>

            <small>pequeño</small>

            Esto es un<sub>subíndice</sub>

            Y esto un<sup>superíndice</sup>

            </pre>

    </body>

</html>
```

Ejemplo de resultado que se puede obtener en algunos navegadores. Ten en cuenta que al usarse atributos deprecated la respuesta de diferentes navegadores puede no ser la misma.



Nota 1: Hemos utilizado la etiqueta <pre> que comentaremos más adelante.

Nota 2: si estás utilizando Notepad++ y obtienes una visualización extraña de las tildes, por ejemplo “Ejemplo bÃ¡sico” en lugar de “Ejemplo básico” recuerda elegir como codificación UTF-8 sin BOM. También añade la etiqueta <meta charset="utf-8"> dentro de la cabecera del código para indicar el juego de caracteres que se debe emplear.