Diseño y Publicación de páginas Web

ISBN: 978-84-96157-86-6

Autor: Fernando Resco Vila.
Actualizaciones: Manuel Barnuevo, Luis Gómez Caballero.

Índice de contenidos

1

Herramientas de edición web

- Convenios de notación del manual
- Introducción
- Funciones y características del editor web
- Documentación técnica

1.1 CONVENIOS DE NOTACIÓN DEL MANUAL

En este manual se han seguido los siguientes criterios de notación:

- Las explicaciones están orientadas a la utilización del **ratón**. En los casos en los que se refieran al teclado, se indicará expresamente.
- Mientras no se indique lo contrario, se supone que siempre se utiliza el **botón izquierdo** del ratón.
- **Clic** significa pulsar una vez el botón del ratón.
- **Doble clic** significa pulsar dos veces seguidas y rápidas el botón del ratón.
- **Arrastrar** significa mover el ratón sobre la mesa a la vez que se mantiene pulsado el botón izquierdo del ratón.
- Cuando se hace referencia a la pulsación de una tecla del teclado, ésta aparece en letra negrilla. Por ejemplo, **Esc** indica que se debe pulsar dicha tecla.
- Los procedimientos que aparecen numerados (1, 2, 3, etc.) han de seguirse de forma secuencial. No pase al punto siguiente hasta haber realizado el punto anterior.
- Cuando haya que expresar una decisión entre varios valores en una etiqueta, usaremos el elemento |. Por ejemplo, elegir entre 0 ó 1 sería 0|1.
- En la mayoría de los procesos descritos en este manual, se hace uso de la Barra de Menús; es decir, del menú que se visualiza en la parte superior de la pantalla. Aunque existen también los denominados menús **Contextuales** (menús que se activan con el botón derecho del ratón), éstos sólo se utilizan en algunos casos y se indicará expresamente.
- Si hay que pulsar una tecla y, sin soltarla, pulsar otra, ambas aparecen en letra negrilla. Por ejemplo **Alt+F4** indica que se ha de pulsar la tecla **Alt** y, sin soltarla, pulsar una vez la tecla **F4**.
- En los procedimientos que se detallan en este manual, se da por supuesto que el lector está mínimamente familiarizado con la forma de trabajar con Windows, así como con la utilización de un procesador de texto como el Bloc de Notas o Word.
- También se presupone que ha utilizado alguna vez un navegador web y que es capaz de navegar por Internet o abrir una página web.

1.2 INTRODUCCIÓN

El primer lenguaje de programación que se aprende para realizar páginas web es el HTML (*Hyper Text Marckup Language*).
El motivo para esta situación es que los navegadores web fundamentalmente sólo entienden HTML. Pueden comprender otros lenguajes con la ayuda de añadidos, pero siempre en el núcleo de todo se encuentra el lenguaje HTML.
Existen otros lenguajes más avanzados, como Java, PHP, o todos los involucrados en las tecnologías .Net como Visual Basic o Visual C#, que cuando quieren mostrar una página web "incrustan" código HTML dentro del suyo propio. Este hecho supone que si se van a publicar los resultados en Internet, aprender alguno de estos lenguajes sin conocer previamente HTML resulte una tarea complicada.
Más importante aún, el uso de navegadores web para mostrar cualquier aplicación, no sólo las destinadas a Internet, cada vez está más difundido. El motivo es que si se ejecuta la aplicación en el navegador web ya se dispone de un entorno gráfico que incorpora múltiples funcionalidades sin necesidad de programarlas desde cero.

Para crear páginas web con el lenguaje HTML, sólo es preciso un editor de texto como el Bloc de Notas o Word. Sin embargo, con el fin de ganar en eficiencia a la hora de generar código HTML, se recurre a otras herramientas más avanzadas como Eclipse *(http://www.eclipse.org)*, Adobe Dreamweaver *(http://www.adobe.com/es/products/dreamweaver)* o Microsoft Expression Studio *(http://www.microsoft.com/SPAIN/expression)*, por nombrar sólo algunos de los más difundidos.
La ventaja que ofrece una herramienta de edición web como las anteriormente mencionadas frente al simple editor de texto, es que añaden funcionalidades que facilitan la generación de código HTML. Dentro de las funcionalidades más interesantes, destacan especialmente las ayudas para detectar errores y corregirlos,

ejecutar el código HTML en diversos navegadores web, ordenar el código para hacerlo más legible y mantenerlo fácilmente, y finalmente quizá la más importante, generar código automáticamente sin necesidad de programación.

Como paso previo a aprender a programar en HTML, vamos a conocer las características y funcionalidades principales de la aplicación **Adobe Dreamweaver.** Esta será la herramienta de construcción de páginas web que vamos a utilizar durante todo el curso.

1.3 FUNCIONES Y CARACTERÍSTICAS DEL EDITOR WEB

El primer paso para conocer la herramienta de desarrollo de páginas web que se va a utilizar durante el curso es instalar Dreamweaver.

Se puede descargar una versión de prueba de Adobe Dreamweaver en la dirección **http://www.adobe.com/es/products/dreamweaver/** en el apartado de **Descargas**.

Una vez se dispone de esta versión de prueba, llega el momento de comenzar con la instalación. Tras ejecutar el programa aparecerá la siguiente figura:

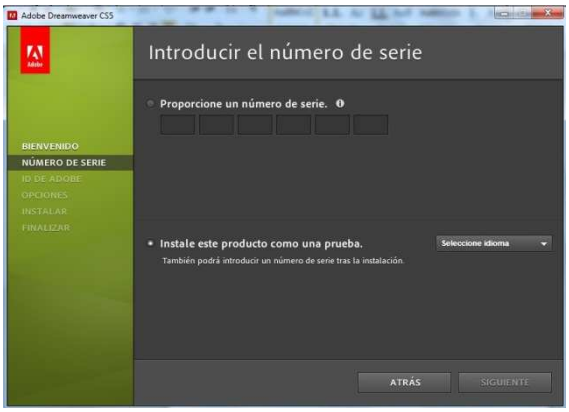


Figura 1. Instalación Dreamweaver en evaluación

Se elige la opción de instalar la aplicación en periodo de evaluación y se pulsa sobre **Siguiente**. A partir de aquí instalaremos la aplicación con las opciones por defecto.

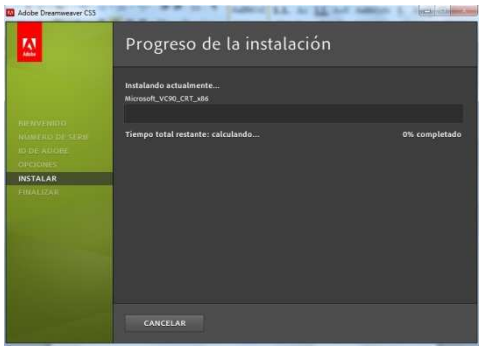


Figura 2. Proceso de instalación

Una vez finalizada la instalación, se podrá abrir desde el botón de inicio del sistema operativo.

Figura 3. Ejecutar Dreamweaver

Entorno de trabajo

Cuando se inicia Dreamweaver, tras aceptar el contrato de licencia, aparecerá una ventana como la de la figura:

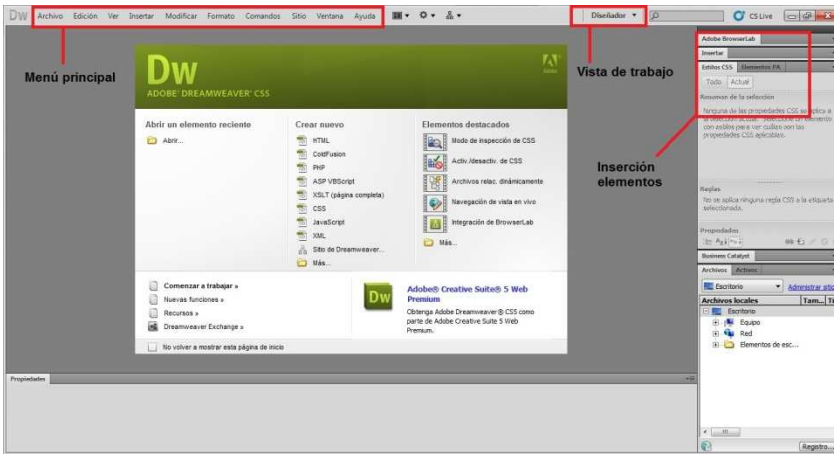


Figura 4. Primera pantalla de Dreamweaver

De la anterior figura se pueden comentar varios detalles. Para empezar, en la zona central de la pantalla se dispone de una ventana que permite **Abrir un elemento reciente**, **Crear nuevo** y **Elementos destacados**.

En el primer caso, **Abrir un elemento reciente**, la aplicación guarda un histórico de los ficheros que se han abierto recientemente, para agilizar su apertura si así se desea.

La opción de **Crear nuevo**, como su nombre indica permite generar el esqueleto de una página web en los distintos lenguajes soportados por la aplicación (HTML, ColdFusion, PHP,…). Este objetivo, con muchas más opciones, está disponible desde el menú principal y su opción **Archivo** y **Nuevo**.

La parte de **Elementos destacados** y la de **Comenzar a trabajar**, es un punto de acceso en línea a distintos tutoriales y análisis sobre las nuevas funcionalidades de Dreamweaver.

El **menú principal** situado en la parte superior de la aplicación permite acceder a la totalidad de las opciones de desarrollo que proporciona la aplicación. Será por tanto muy utilizada durante nuestra fase de desarrollo.

El área de las **vistas de trabajo** permite redistribuir automáticamente todos los elementos de la aplicación para acomodarlos lo mejor posible al tipo de tarea que se esté realizando: diseño, codificación, programación,…

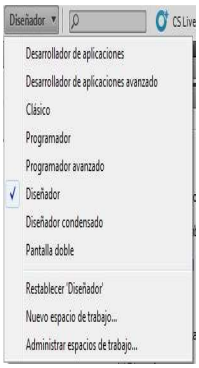


Figura 5. Vistas de trabajo

Probablemente las dos más usadas son las de desarrollo y diseño. En la siguiente figura se puede apreciar como quedaría el entorno de trabajo en ambos casos. Se puede apreciar que aparte de reorganizar los distintos menús, se despliegan algunas opciones como por ejemplo el área de inserción de elementos HTML:

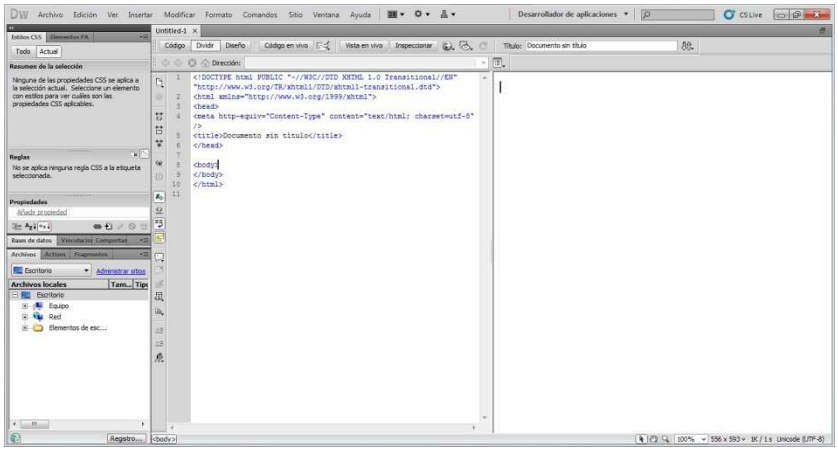


Figura 6. Vista de desarrollador de aplicaciones

Si se mantiene pulsado el ratón sobre cualquiera de los paneles que componen la anterior ventana, dicho panel se convertirá en flotante, pudiendo desplazarlo a la región de la pantalla que se desea. De esta forma se puede reestructurar los distintos paneles según se desee. Si se pulsa **F4**, **se ocultarán todos los paneles** quedando sólo abierto el de código y diseño. Si vuelve a pulsar aparecerán de nuevo.

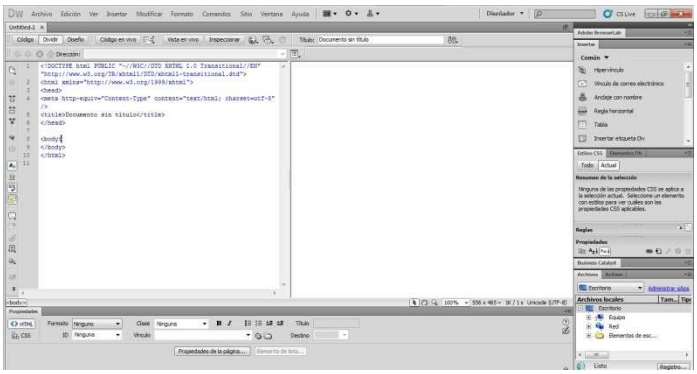


Figura 7. Vista de diseñador

El área de **inserción de elementos** permite generar elementos sin programar. Simplemente hay que contestar a los asistentes que se despliegan en pantalla para generar el código HTML necesario.

Figura 8. Área de inserción de elementos web

Comienzo del trabajo

Tras la primera descripción del entorno de trabajo es el momento de comenzar a trabajar. Para ello vamos a generar un documento básico en HTML y ver los menús implicados en Dreamweaver.

Para un proyecto de múltiples páginas web es aconsejable primero generar la **estructura de un sitio web** que englobe todos los ficheros que se van a utilizar. En nuestro curso, para generar ejemplos o páginas web individuales, no es necesario recurrir a generar el sitio web.

Si se desea ampliar información sobre como generar sitios web, dirigirse al apartado **Capítulo 3: Utilización de los sitios de Dreamweaver** del manual de uso de Dreamweaver (ver el siguiente apartado **1.4 Documentación técnica** de este curso para saber cómo obtenerlo).

Los pasos necesarios para **crear un nuevo documento web** son los siguientes:

1. Desde el menú principal, pulsar sobre **Archivo** y **Nuevo**. Se abrirá un asistente para elegir que tipo de documento se desea crear.

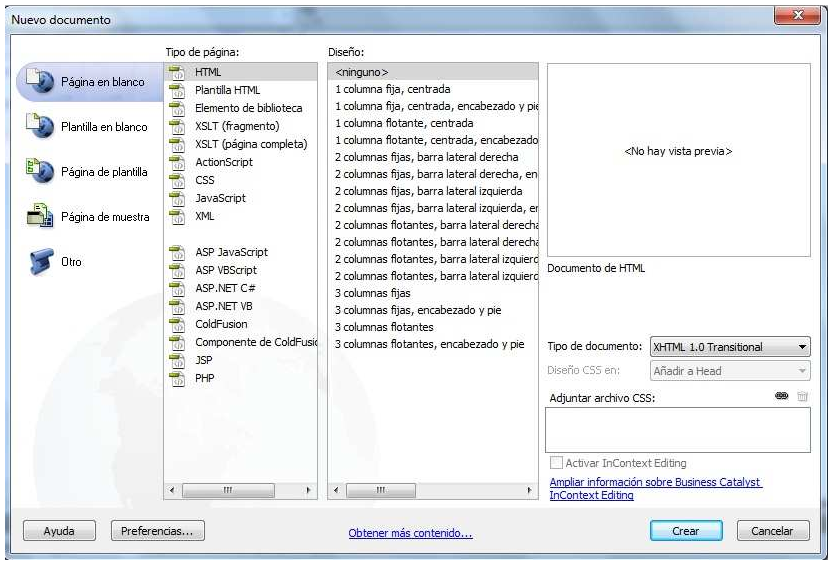


Figura 9. Nuevo documento en Dreamweaver

2. Desde la opción de **Página en blanco** se puede crear una página con diversas estructuras predefinidas en hojas de estilo (una o varias columnas), o directamente una página en blanco con la estructura básica. En la columna derecha se puede ver un previo de cómo quedaría la estructura, del DocType del documento (la versión de HTML que se usa), el fichero CSS (con la hoja de estilo) y como se adjuntará al documento HTML (en nuestro caso incluido en la etiqueta head).

3. En nuestro caso se elegirá el caso más sencillo, como Tipo de página se elige **HTML**, como Diseño se escoge **<ninguno>** y en el Tipo de Documento se seleccionará **HTML 4.01 Transitional**.

4. Pulsar sobre **Crear**. La pantalla se habrá modificado según la siguiente figura:

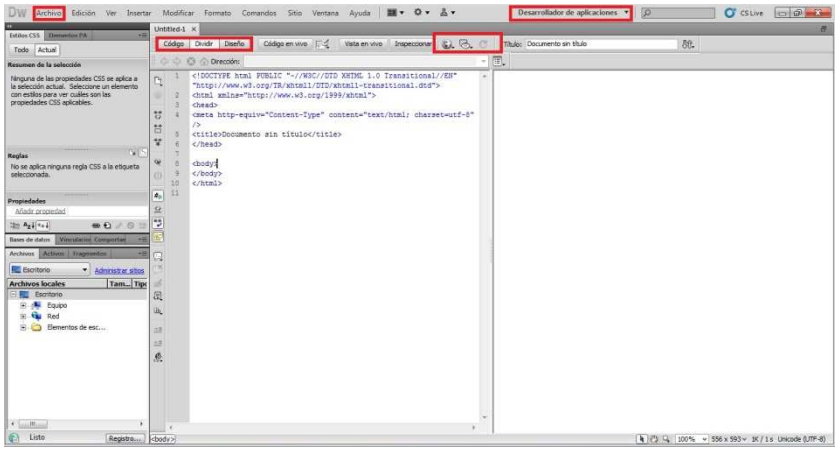


Figura 10. Crear nuevo documento

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
<title>Documento sin título</title>
</head>

<body>
</body>
</html>
```

Código Figura 10. Crear nuevo documento

5. A partir de esta figura podemos apreciar varios elementos significativos en el proceso de creación de páginas web.

    a. En la vista que hemos escogido, **Desarrollo de aplicaciones avanzado**, en la zona central se dispone del código HTML de nuestra página. Es el esqueleto básico con los elementos mínimos. Más adelante, cuando comencemos a conocer el lenguaje HTML se verá el significado de cada elemento.

    b. La zona central muestra dos divisiones horizontales: la del código y la del diseño. Esto es así porque se ha activado la casilla de **Dividir**. Si se pulsa sobre **Código**, sólo se verá la parte de las etiquetas HTML. Si se pulsa sobre **Diseño**, sólo se verá esta parte. Se pueden crear páginas web usando ambas vistas o sólo alguna de ellas.

    c. La casilla de **Vista en vivo**, actúa como un navegador web. Es decir es un previo de cómo se vería en un navegador como Internet Explorer o Firefox.

6. Todavía no le hemos asignado nombre ni título a nuestra página web, por lo que en la **pestaña del código** aparece untitled-3*, y en **Título** aparece Documento sin título. Introducimos en título el texto: Primer Documento. Se verá que el código HTML se modifica en su línea 5.

    a. Donde ponía:

    <title>Documento sin título</title>

    b. Ahora pone:

    <title>Primer Documento</title>

    c. Como se puede apreciar, cualquier cambio que afecte a la aplicación automáticamente se verá reflejado en el código HTML.

7. A continuación vamos a **grabar la página web**. Para ello, se debe pulsar en el menú principal sobre **Archivo** y **Guardar.**

    a. Cuando Dreamweaver pregunta por el nombre de la página web, se introduce cualquiera, pero con extensión html o htm. Por ejemplo: primerdocumento.html.

    b. Se pulsa sobre **Aceptar**.

Figura 11. Primer documento

c. Se puede ver que la pestaña de código ha cambiado con el nombre de la página web. Ha aparecido también la ruta donde se ha guardado el fichero.

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Primer Documento</title>

</head>



<body>

</body>

</html>
```

Código Figura 11. Primer documento

8. Para **modificar este código HTML**, bastará con pulsar sobre la ventana central y teclear las etiquetas HTML que se deseen. También se podría generar código automáticamente con los asistentes que proporciona Dreamweaver. En este caso será esta opción la que se usará.

9. Se creará una tabla.

a. Pulsar detrás de la etiqueta **<body>** y luego sobre **Intro**.

b. Hacer Clic sobre **Insertar** y **Tabla**.

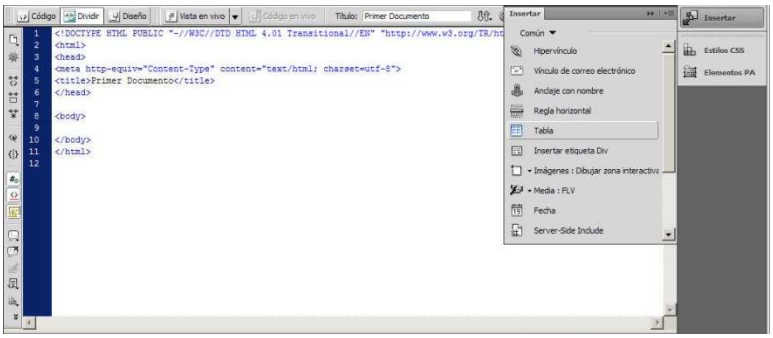


Figura 12. Añadir tabla a documento HTML

c. Se abrirá un asistente en el que habrá que rellenar los datos que se solicitan.

Figura 13. Asistente generación tablas

d. Tras pulsar sobre **Aceptar**, la zona de código y diseño quedarán como la siguiente figura.

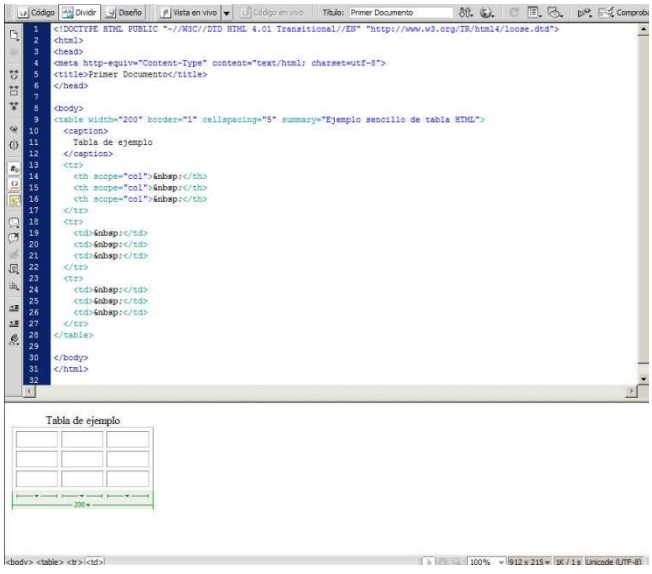


Figura 14. Ejemplo de tabla

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">

<title>Primer Documento</title>

</head>

<body>

<table width="200" border="1" cellspacing="5"
summary="Ejemplo sencillo de tabla HTML">

 <caption>

   Tabla de ejemplo

 </caption>

 <tr>

   <th scope="col"> </th>

   <th scope="col"> </th>

   <th scope="col"> </th>

 </tr>
```

```
  <tr>
    <th scope="row"> </th>
    <td> </td>
    <td> </td>
  </tr>
<tr>
    <th scope="row"> </th>
    <td> </td>
    <td> </td>
  </tr>
</table>
</body>
</html>
```

Código Figura 14. Ejemplo de tabla

10. Todavía teníamos pendiente ver las opciones disponibles en la parte superior derecha de la pantalla principal. Este es el mejor momento para aplicarlas.

Figura 15. Previos, actualizar vista, opciones,...

a. El primero nos permite visualizar e inspeccionar el aspecto real (online) de la página que estamos maquetando sin necesidad de salir de la aplicación. Sus posibilidades mejoran de forma considerable la eficiencia de nuestros trabajos evitando que dependamos de plugins de diferentes navegadores

b. El tercero permite ver la página web directamente en un navegador. Si se pulsa **F12** se usará el navegador por defecto. Si no se desea este, entonces habrá que elegir dentro de los que se muestren en el listado que aparece (siempre y cuando se encuentren instalados).

Figura 16. Vista previa en Dreamweaver

a. El cuarto botón se encarga de mostrar o no **Ayudas visuales** en la vista de diseño. Por ejemplo si se muestran los bordes de las tablas, los mapas de imágenes, etc.

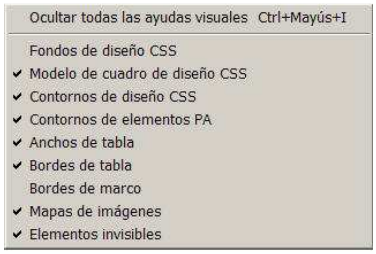


Figura 17. Ayudas visuales

1.4 DOCUMENTACIÓN TÉCNICA

Adobe proporciona multitud de información sobre Dreamweaver en sus páginas web corporativas. Existen foros de consulta, tutoriales, videotutoriales, manuales, ejemplos descargables, solución a preguntas frecuentes, etc. Un buen punto de partida es la dirección:

http://www.adobe.com/es/products/dreamweaver/

Si se desea ampliar información sobre Dreamweaver, existe un **manual descargable en formato PDF** en la siguiente dirección:

http://help.adobe.com/en_US/dreamweaver/cs/using/dreamweaver_cs5_help.pdf

También existe un excelente videotutorial sobre como comenzar a trabajar con Dreamweaver, denominado **Getting started with Dreamweaver CS5**. Está disponible en la siguiente dirección:

http://tv.adobe.com/watch/learn-dreamweaver-cs5/getting-started-gs-what-is-dreamweaver-cs5-/

2

Los lenguajes de marcas

- Características de los lenguajes de marcas
- Mi primer documento HTML
- Estructura de un documento creado con lenguaje de marcas
- Navegadores web
- Declaraciones de tipos de documentos web
- Marcas para dar formato al documento
- Creación de tablas y listas
- Enlaces y direccionamientos
- Marcos y capas

2.1 CARACTERÍSTICAS DE LOS LENGUAJES DE MARCAS

Las siglas del lenguaje HTML provienen de *Hyper Text Markup Language*, o lo que es lo mismo, Lenguaje de Marcas de Hipertexto. HTML por tanto es un lenguaje de etiquetas (también denominadas marcas) que incorpora hipertexto (hipervínculos o simplemente enlaces) a otras partes del documento para facilitar su uso.

Las páginas web actuales son un conjunto de múltiples etiquetas, que son fáciles de leer y escribir por personas y aplicaciones. Estas etiquetas en general, salvo algunas excepciones se escriben por pares y se forman de la siguiente forma:

- **Etiqueta de comienzo**: Compuesta por el carácter especial <, seguido por el nombre de la etiqueta, y terminado por el carácter >. Es importante indicar que no deben dejarse espacios entre los caracteres especiales y el nombre de la etiqueta. Además es indiferente escribir las etiquetas en mayúsculas o minúsculas. Si trabajamos con XHTML, la anterior afirmación es incorrecta, pues todas las etiquetas deben escribirse en minúscula.

- **Etiqueta de final**: Se forma por el carácter <, seguido del carácter /, con el nombre de la etiqueta, y el carácter >.

Con todos estos elementos, el resultado sería similar al siguiente:

<nombre_etiqueta> … >/nombre_etiqueta>

En HTML existen 91 etiquetas que pueden usarse para mostrar en el navegador web los distintos componentes que forman nuestra página.

a, abbr, acronym, address, applet, area, b, base, basefont, bdo, big, blockquote, body, br, button, caption, center, cite, code, col, colgroup, dd, del, dfn, dir, div, dl, dt, em, fieldset, font, form, frame, frameset, h1, h2, h3, h4, h5, h6, head, hr, html, i, iframe, img, input, ins, isindex, kbd, label, legend, li, link, map, menu, meta, noframes, noscript, object, ol, optgroup, option, p, param, pre, q, s, samp, script, select, small, span, strike, strong, style, sub, sup, table, tbody, td, textarea, tfoot, th, thead, title, tr, tt, u, ul, var.

En función de la versión de HTML, alguna de estas etiquetas resultaran obsoletas, como por ejemplo applet, basefont, center, dir, font, isindex, menu, s, strike y u.

Junto con las etiquetas o marcas, existen los denominados **atributos**. Son características que se añaden a las marcas para completar su significado.
Por ejemplo, la etiqueta **** se utilizaba para indicar las características del tipo de letra que se va a usar para mostrar por pantalla un texto. Como es lógico, la primera necesidad que se detecta es indicar que tipografía de texto se van a usar, cuál será el color del texto y evidentemente su tamaño. Este es el cometido de los atributos color, size, y face. En el siguiente ejemplo se ha elegido el color negro, tamaño 6 y como fuentes Verdana, en su defecto Geneva y si faltan las anteriores sans-serif. Todo esto se aplicaría al texto Hola. Podemos ver el resultado a continuación:

Hola

También podríamos haber escrito el nombre del color en inglés:

Hola

2.2 MI PRIMER DOCUMENTO HTML

Un documento HTML presenta la siguiente organización:

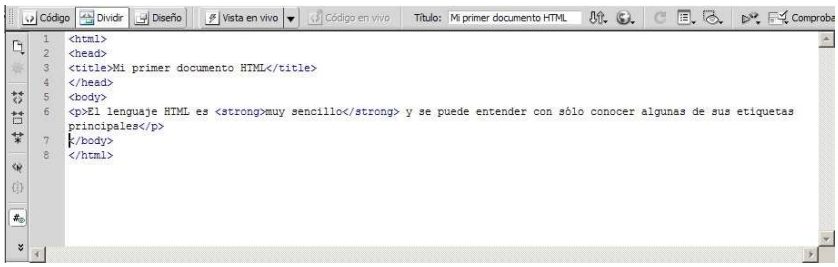


Figura 19. Mi primer documento HTML

```
<html>

<head>

<title>Mi primer documento HTML</title>

</head>

<body>

<p>El lenguaje HTML es <strong>muy sencillo</strong> y se puede entender si se conocen algunas de sus etiquetas principales.</p>

</body>

</html>
```

Código Figura 19. Mi primer documento HTML

Para ver los resultados de esta primera página web, sólo habría que hacer lo siguiente:

1. Abrir un editor de texto como el Bloc de Notas o un editor web como Dreamweaver y crear un archivo nuevo.
2. Teclear el código que aparece en la **Figura 18**.
3. Guardarlo con el nombre que queramos, pero con la extensión html o htm.
4. Abrirlo con cualquier navegador. En este caso usaremos Internet Explorer 8.
5. Tendremos como resultado:

Figura 20 - Mi primer programa en el navegador web

6. Desde este mismo navegador web, pulsaremos sobre la ventana central donde aparece el texto y tras pulsar el botón derecho del ratón, pulsamos sobre **Ver código fuente**. Como su nombre indica, de esta manera accedemos al código fuente de la página web que estamos visualizando en ese momento.

7. Si repetimos esta acción con cualquier otra página web, podremos ver el código HTML de esa página. Evidentemente, en función de la complejidad de la página web, el número de líneas de código y número de etiquetas podrá variar bastante.

2.3 ESTRUCTURA DE UN DOCUMENTO CREADO CON LENGUAJE DE MARCAS

Tras estos primeros pasos, vamos a analizar qué es lo que hemos pedido al navegador web que nos muestre en la pantalla del ordenador. Volvemos a fijarnos en la **Figura 1**.

Las tres etiquetas fundamentales de un documento HTML son **<html>**, **<head>** y **<body>.**

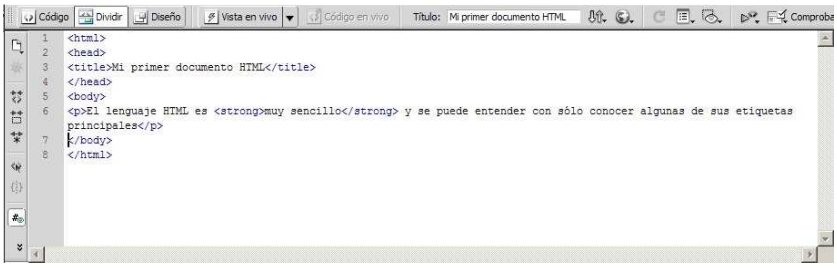


Figura 21. Mi primer documento

- **<html>**: Se usa para indica el comienzo y el final de un documento HTML. Ninguna etiqueta o contenido puede colocarse antes o después de la etiqueta <html>, salvo la excepción de <DOCTYPE>

que se verá más adelante. En el interior de la etiqueta <html> se definen la cabecera y el cuerpo del documento HTML y todo lo que se coloque fuera de la etiqueta <html> se ignora.

- **<head>**: Delimita la parte de la cabecera del documento. La cabecera contiene información sobre el propio documento HTML, como por ejemplo su título y el idioma de la página. Los contenidos indicados en la cabecera no son visibles para el usuario en el navegador, con la excepción de la etiqueta **<title>**, que se utiliza para indicar el título del documento. Dicho título es mostrado por los navegadores web en la parte superior izquierda de la ventana de cada navegador.
- La etiqueta **<meta>** es un caso especial. También aparece dentro del <head>. No tiene etiqueta de cierre. Se utiliza para añadir información sobre el documento, como el idioma, descripción de la página, cuando refrescar una página, etc. Algunos ejemplos:

Etiqueta <META> y atributos

Indica el editor con el que se ha creado la página
<meta name="GENERATOR" content="nombre del editor">

Codificación usada en la página
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">

Recargar una página o redirigirla a otra
<meta http-equiv="refresh" content="20;URL=./indice.htm" >

Palabras claves que los "robots" (Google, Yahoo,…) usarán
<meta name="Keywords" content="palabras separadas por comas" >

Descripción de la página
<meta name="Description" content="descripción de la página" >

Permitir o no el acceso de la página a los robots de búsqueda
<meta name="Robots" content="*valores*">
all para que el robot tenga en cuenta esta página y pueda seguir los links contenidos en la misma (este es el valor predeterminado)
noindex para que no la tenga en cuenta.
none que es lo mismo que noindex, nofollow
index para que la tenga en cuenta
follow para que pueda seguir los links que haya en la página
nofollow lo contrario de follow

Fecha en la que expira una página
<meta http-equiv="expires" content="**Sat, 16 Nov 2002 16:05:00 GMT**" >

No guardar la página en caché
<meta http-equiv="Pragma" content="no-cache" > ó
<meta http-equiv="expires" content="-1" >

Evitar que una página se muestre dentro de un frame
<meta http-equiv="Window-target" content="_top" >
Con **_top** se muestra en página completa, no en frame
Con **_blank** se muestra en una nueva ventana

Autor
<META name="Author" content="nombre">

Idioma
<!-- Para hablantes de inglés americano -->
<meta name="keywords" lang="en-us" content="vacation, Spain, sunshine">
<!-- Para hablantes de inglés británico -->
<meta name="keywords" lang="en" content="holiday, Spain, sunshine">
<!-- Para hablantes de español -->
<meta name="keywords" lang="es" content="vacaciones, España, sol">

Figura 22. Etiqueta <meta >

- **<body>**: Define el cuerpo del documento HTML. El cuerpo encierra todos los contenidos que se muestran al usuario (párrafos de texto, imágenes, tablas, listas…).

NOTA

Esta etiqueta dispone de diversos atributos para indicar colores de fondo, del texto, enlaces,… pero están en desuso pues obligan al conjunto de elementos que engloban. Lo más usado hoy en día para lograr los mismos resultados es recurrir a las hojas de estilo que veremos más adelante.

Un detalle importante. Las líneas en blanco y espacios fuera de cada pareja de etiquetas son ignoradas por el navegador. La utilidad que tienen estos espacios en blanco es organizar las distintas etiquetas, de forma que el código HTML sea más fácil de leer por los usuarios cuando construyen sus páginas web. Algunas herramientas de edición web también colorean automáticamente las distintas etiquetas y atributos, lo que facilita mucho la lectura del código HTML.

primer documento.html

```
1  <html>
2
3      <head>
4          <title>Mi primer documento HTML</title>
5      </head>
6
7      <body>
8          <p>El lenguaje HTML es <strong>muy sencillo</strong>
9          y se puede entender si se conocen algunas de sus etiquetas principales.</p>
10     </body>
11
12 </html>
```

Figura 173. Código con marcas coloreadas

Los **comentarios**, como hemos visto, se generan mediante la pareja de etiquetas <!-- y -->. Es decir nuestros comentarios quedarían según el siguiente ejemplo:

<!-- Para hablantes de español -->

2.4 NAVEGADORES WEB

Hasta ahora hemos visto un navegador web, Internet Explorer, que viene ya viene incorporado en el sistema operativo Windows.

Sin embargo no es el único que se utiliza. Se pueden apreciar en la siguiente imagen otros navegadores web utilizados habitualmente, junto con su cuota de mercado:

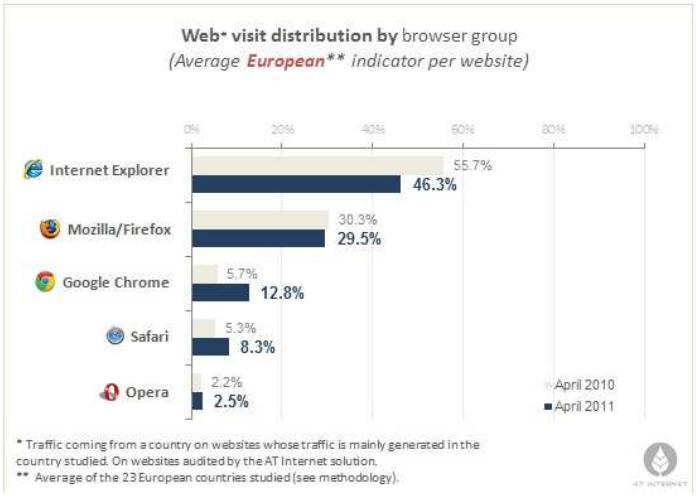


Figura 24. Navegadores web

Estos datos son generales y varían con el tiempo, el tipo de usuario, y el sistema operativo que use. Por ejemplo, los usuarios del sistema operativo MAC OS utilizan mayoritariamente Safari. Firefox se usa mayoritariamente en Windows y Linux por usuarios técnicos y cada año está ganando cuota de mercado a Internet Explorer. Chrome es nuevo en el mercado, con menos de un año de vida y ganando adeptos con el paso del tiempo.

Hasta hace muy poco tiempo, era muy importante prever que navegador web iba a interpretar nuestra página web, porque todos no entendían igual las etiquetas del estándar HTML. Por decirlo así, **algunos navegadores no cumplían la normativa estándar**. Hoy en día esta situación ha cambiado y por norma general las páginas web se ven igual en cualquier navegador.

A pesar de esta aceptación progresiva del estándar, y dado que los navegadores están continuamente actualizándose, para evitar problemas de visualización de nuestra página web es aconsejable abrirla en varios navegadores.

Algunas herramientas de desarrollo web como Dreamweaver nos ayudan a saber cómo se van a ver nuestras páginas web en diversos navegadores.

1. Desde la página principal de Dreamweaver, si pulsamos la tecla **F12**, nuestra página web se abrirá con el navegador web predeterminado en el sistema operativo.

2. Si se desea ver la página web con otro navegador **que esté instalado en nuestro sistema operativo**, lo más rápido es pulsar sobre el icono de **Vista previa** y elegirlo de la lista de opciones que se nos ofrece.

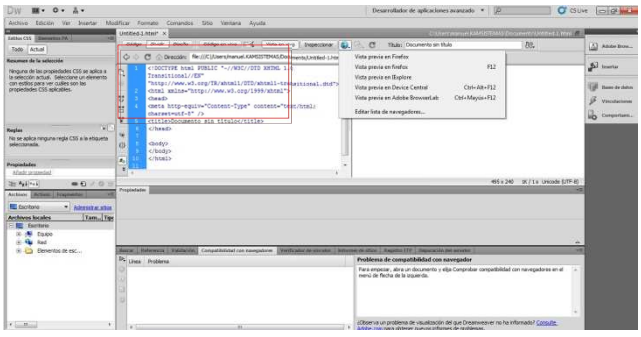


Figura 25. Vista previa en Dreamweaver

Podemos **descargar alguno de los navegadores** más usados en las siguientes direcciones:

- **Internet Explorer**

http://www.microsoft.com/latam/windows/internet-explorer/

- **Firefox**

http://www.mozilla-europe.org/es/firefox/

- **Safari**

http://www.apple.com/es/safari/download/

- **Opera**

http://www.opera.com/download/

2.5 DECLARACIONES DE TIPOS DE DOCUMENTOS WEB

Como ya hemos visto en el primer capítulo, al crear un documento HTML nuevo en Dreamweaver, tendremos un resultado como el siguiente:

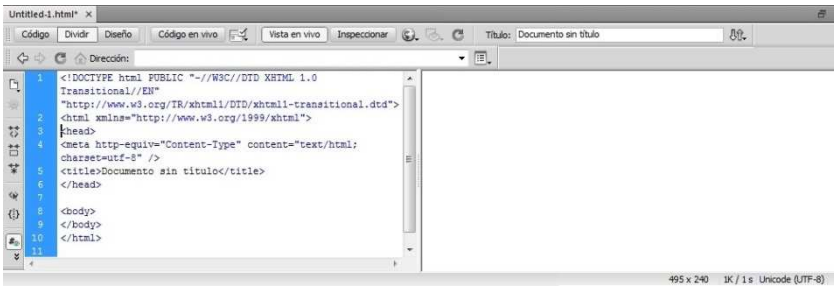


Figura 26. Crear una nueva página web en Dreamweaver

La clave para definir qué tipo de documento HTML estamos generando es la etiqueta <DOCTYPE>. Su cometido es indicar que versión de HTML estamos utilizando, para que el navegador web conozca que conjunto de marcas vamos a usar.

Aquí conviene hacer una parada y un poco de historia, pues aclarará muchos detalles sobre las versiones de HTML y su situación actual.

La última versión oficial y estándar de HTML es HTML 4.01. Fue publicado y aprobado el 24 de diciembre de 1999 por el organismo W3C (*World Wide Web Consortium*, en http://www.w3.org), cuyo cometido es elaborar las normas que deben seguir los diseñadores de páginas web para crear las páginas HTML. Tras la publicación de esta versión de HTML, su trabajo de estandarización se paró y se centró en desarrollar un nuevo estándar denominado XHTML, una versión mejorada que vio la luz el 26 de Enero de 2000. XHTML 1.0 es una adaptación de HTML 4.01 al lenguaje XML, por lo que mantiene casi todas sus

etiquetas y características, pero añade algunas restricciones y elementos propios de XML. El estándar XHTML 1.0 incluye el 95% del estándar HTML 4.01, ya que sólo añade pequeñas mejoras y modificaciones menores. Existen dos borradores no definitivos de las versiones 1.1 y 2.0 de XHTML.

La falta de interés del W3C en HTML provocó que en 2004 diversas empresas como Apple, Mozilla y Opera decidieran organizarse en una nueva asociación llamada WHATWG (*Web Hypertext Application Technology Working Group* en http://www.whatwg.org). Su objetivo fundamental es evolucionar HTML y fruto de estos trabajos surgió el primer borrador del estándar HTML 5, el 22 de enero de 2008. Debido al interés despertado, el W3C decidió retomar la actividad estandarizadora de HTML, estando pendiente su aprobación definitiva antes de finalizar el año 2010. El número de empresas que apoyan HTML 5 crece por momentos, siendo alguna de las últimas incorporaciones más notables Google, pues todas sus últimas páginas web ya soportan este futuro estándar. Para comprobar cuales, basta con ver el código fuente de la página en cuestión desde cualquier navegador web y comprobar que DocType declaran.

En función del *DocType* podemos trabajar con distintas versiones de HTML:

- **HTML 4.01 Strict:**

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

- **HTML 4.01 Transitional:**

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

- **HTML 4.01 Frameset:**

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

- **XHTML 1.0 Strict:**

<!DOCTYPE HTML PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

- **XHTML 1.0 Transitional:**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

- **XHTML 1.0 Frameset:**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Frameset//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">

- **XHTML 1.1 - DTD:**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN" "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">

- **XHTML Basic 1.1 - DTD:**

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML Basic 1.1//EN" "http://www.w3.org/TR/xhtml-basic/xhtml-basic11.dtd">

- **HTML 5 (todavía no definitivo):**

<!DOCTYPE HTML>

Cada una de estas versiones y mejoras de HTML poseen normas que están escritas en inglés, en un lenguaje con frecuencia bastante complejo y con cierta asiduidad poco práctico. Por tal motivo no es conveniente ni necesario leerse desde el principio al fin estas normas, pues con las indicaciones que vamos a dar en este manual es suficiente para programar con soltura en HTML.

Si se desea información sobre algún detalle, estas normas pueden consultarse de forma gratuita en las siguientes direcciones:

- Especificación oficial de HTML 4.01:

http://www.w3.org/TR/html401/

- Especificación oficial de XHTML 1.0:

http://www.w3.org/TR/xhtml1/

2.6 MARCAS PARA DAR FORMATO AL DOCUMENTO

La mayoría del contenido de las páginas web está formado por texto con distintos formatos. Por este motivo es muy importante conocer que etiquetas y atributos utiliza HTML para manipular texto en pantalla.
Una página web no difieren de otros medios de publicación de textos a la hora de estructurar y resaltar los documentos. Así, igual que en otros medios es necesario que se pueda definir párrafos y secciones o por ejemplo remarcar el texto con letra en negrita o cursiva.

Párrafos

En la siguiente imagen podemos apreciar un ejemplo de varios párrafos con distintos formatos.

Figura 27. Dar formato a texto con párrafos

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Ejemplos</title>

</head>

<body>

        <p align="center">Este es un texto centrado</p>

        <p align="right">Este segundo texto es <strong>más largo</strong>.<br> Ocupa varias líneas y el navegador lo ajustará al tamaño de la ventana.</p>

</body>

</html>
```

Código Figura 28. Dar formato a texto con párrafos

Como puede apreciarse, la etiqueta clave en la generación de párrafos es ** ** y ** **. Los párrafos son marcas de bloque que se ajustan por el navegador web a la totalidad de la pantalla. Para decidir el tipo de ajuste se recurre al atributo general **align**.

Acciones	Etiquetas y atributos
Definir párrafo	…
Alinear párrafos	<P ALIGN=LEFT|CENTER |RIGHT|JUSTIFY>texto</P>

Figura 29. Párrafos en HTML

De nuevo vemos una etiqueta que no tiene cierre. Se trata del **salto de línea** o **
**. Su cometido es como su nombre indica introducir un “retorno de carro” al texto. En XHTML, una de las normas que lo diferencia del HTML “puro”, es que no es posible que una etiqueta no tenga cierre. En este caso, lo ideal sería poner **
</br>.** Sin embargo, habitualmente se sustituye por **
.**

Secciones

Evidentemente, con párrafos solamente no se pueden construir la totalidad de los textos en pantalla de una página web. Aquí es donde entran las secciones jerárquicas. El lenguaje HTML posee 6 niveles de secciones. Se puede apreciar un ejemplo en la siguiente figura:

Figura 30. Ejemplo de texto estructurado en secciones

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
<head>
<title>Texto estructurado con secciones</title>
</head>
<body>
        <h1>Titular de la página</h1>
                <p>Párrafo de introducción...</p>
        <h2>Esta es la primera subsección</h2>
                <p>nuevo párrafo de contenido...</p>
        <h3>Otra subsección más</h3>
                <p>Más párrafos de contenido...</p>
</body>
</html>
```

Código Figura 31. Ejemplo de texto estructurado en secciones

De nuevo en este caso, los atributos son generales, sólo permitiéndose el alineamiento. La sintaxis de las etiquetas de secciones es la siguiente:

Acciones	Etiquetas y atributos
Definir sección	<h?> …</h?> ? es un valor de 1 a 6. Ej.: ** Título **
Alinear secciones	<H? ALIGN=LEFT| CENTER| RIGHT></H?> ? es un valor de 1 a 6

Figura 18. Secciones en HTML

Línea horizontal y espacio en blanco

En este apartado de las secciones, existe otra etiqueta que sólo posee la parte de inicio. Se trata de **<hr>**, y su cometido es indicar al navegador web que pinte una **línea horizontal en pantalla**. Igual que en el caso de <br>, si estamos ante un *DocType* de XHTML, deberemos escribir **<hr/>**.

El atributo clave de <hr> es el tamaño de la línea y su grosor. Si no se ponen atributos, el navegador pintará una línea horizontal que abarca todo el ancho del navegador. Si queremos tener una línea de 200 píxeles de ancho, alineada a la izquierda y de 1 pixel de grosor, la etiqueta quedaría según el siguiente ejemplo:

<hr align="left" width="200" size="1">

En el caso del **espacio en blanco**, se recurre a insertar una secuencia de caracteres que comienza por &. Se usa la secuencia ** ** como elemento sustitutivo.

Podemos ver varios ejemplos de utilización en la siguiente figura:

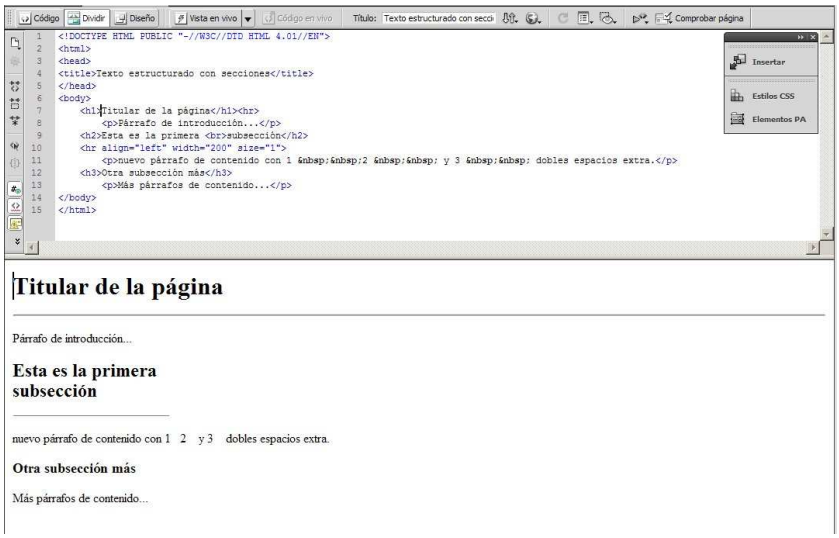


Figura 19. Estructura con líneas en blanco y espacios

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>

<head>

<title>Texto estructurado con secciones</title>

</head>

<body>

        <h1>Titular de la página</h1><hr>

                <p>Párrafo de introducción...</p>

        <h2>Esta es la primera <br>subsección</h2>

   <hr align="left" width="200" size="1">

                <p>nuevo párrafo de contenido con 1   2    y 3    dobles espacios extra.</p>

        <h3>Otra subsección más</h3>

                <p>Más párrafos de contenido...</p>

</body></html>
```

Código Figura 33. Estructura con líneas en blanco y espacios

Codificación de caracteres

En HTML existen distintos caracteres especiales que comienzan por el símbolo & y que están reservados. Su objetivo es sustituir aquellos símbolos que pueden ocasionar problemas a la hora de interpretarlos por el navegador, ya sea porque son utilizados para definir sus etiquetas (como <, >, y “) o porque se trate de símbolos propios de idiomas distintos del inglés (por ejemplo las vocales acentuadas, la ñ, símbolos de puntuación como ¿, ¡, etc.). Por tanto para asegurarse que no va a existir problemas con los caracteres especiales, lo mejor es reemplazar estos por sus correspondientes números de entidad o nombres de entidad.

El conjunto de caracteres **ISO-8859-1**, por defecto en la mayoría de los navegadores web, es muy extenso, siendo los siguientes códigos especiales algunos de los más usados:

Carácter	Núm. Entidad	Entidad	Descripción
"	&#34;	&quot;	Dobles comillas
‘	&#39;	&apos;	Apostrofe
&	&#38;	&amp;	Y
<	&#60;	&lt;	Menos que
>	&#62;	&gt;	Más que
			Espacio
·	&#183;	&middot;	Punto central
×	&#215;	&times;	Multiplicación
÷	&#247;	&divide;	División
¬	&#172;	&not;	Negación
±	&#177;	&plusmn;	Más o menos
ñ	&#241;	&ntilde;	Letra ñ
Ñ	&#209;	&Ntilde;	Letra Ñ
á	&#225;	&aacute;	Letra á
Á	&#193;	&Aacute;	Letra Á
€	&#8364;	&euro;	Moneda Euro

Figura 20. Algunos caracteres especiales

Los cinco primeros son obligatorios para que cualquier navegador web pueda indicar que es compatible con HTML y XHTML.

Podemos encontrar un ejemplo de párrafo de texto con distintas entidades en el siguiente ejemplo:

** Este párrafo contiene caracteres acentuados **

En la Wikipedia se puede consultar la lista completa de las 252 entidades HTML definidas en la siguiente dirección:

http://en.wikipedia.org/wiki/List_of_XML_and_HTML_character_entity_references.

Marcas básicas de fuentes de texto

El estándar HTML incluye numerosas marcas para indicar el formato de texto, tanto su énfasis, colores o tamaño.
La etiqueta **** marca un texto indicando que su importancia es mayor que la del resto del texto. La etiqueta **** indica que un determinado texto es de la mayor importancia dentro de la página. Los navegadores por defecto consideran que se debe poner en cursiva, mientras que se muestra en negrita.

Como veremos más adelante, en las hojas de estilo, estos comportamientos por defecto se pueden redefinir. Si se quiere forzar a que aparezca el texto en **negrita, cursiva o subrayado**, habrá que usar las etiquetas **, y <u>** respectivamente.

Acciones	Etiquetas y atributos
Mayor importancia	texto
Negrita	texto
Énfasis	texto
Cursiva	texto
Subrayado	<u>texto</u>

Figura 21. Resaltar texto

En el siguiente ejemplo se puede ver el comportamiento de las anteriores etiquetas:

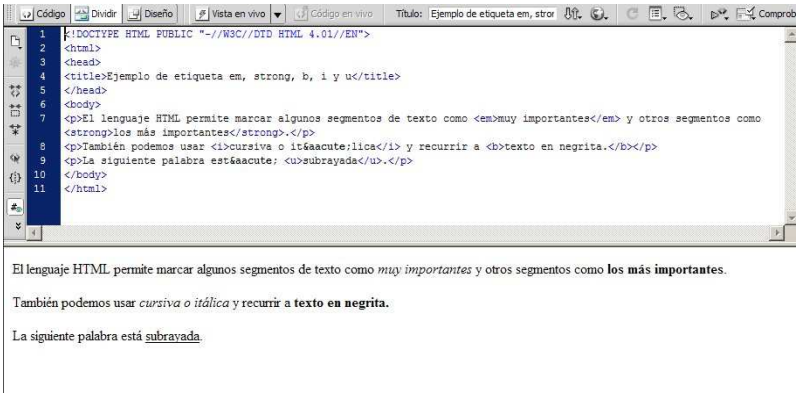


Figura 22. Ejemplos de texto resaltado

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>

<head>

<title>Ejemplo de etiqueta em, strong, b, i y u</title>

</head>

<body>
```

```
<p>El lenguaje HTML permite marcar algunos segmentos de texto como <em>muy importantes</em> y otros segmentos como <strong>los más importantes</strong>.</p>

<p>También podemos usar <i>cursiva o it&aacute;lica</i> y recurrir a <b>texto en negrita.</b></p>

<p>La siguiente palabra est&aacute; <u>subrayada</u>.</p>

</body>

</html>
```

Código Figura 36. Ejemplos de texto resaltado

Debido a que la etiqueta **<font>** ha quedado obsoleta en las últimas versiones de HTML, se recomienda prescindir de esta y usar etiquetas más genéricas para definir tanto texto con colores, tamaños de fuentes o directamente recurrir a las hojas de estilos. Estos últimos elementos, las hojas de estilos o CSS se verán más adelante.

Marcadores genéricos de texto

Hasta ahora hemos visto etiquetas que permiten marcar texto bajo determinadas con circunstancias. Estas etiquetas poseen un comportamiento predeterminado para el navegador, por lo que no conviene cambiarlos. Pero si se desea añadir nuevos comportamientos, entonces se necesita una etiqueta genérica que nos permita aplicarle nuevos estilos de texto. Como por ejemplo que nuestro texto resaltado no sólo tenga el comportamiento propio de una etiqueta , sino que además posea el color azul.

Si se desea marcar un texto en cualquier circunstancia, entonces es el momento de utilizar la etiqueta **.** Se le conoce como marcador de contenido definido general.

Veamos algunos ejemplos:

Figura 23. SPAN con estilos y clases

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>

<head><title>Ejemplo de etiqueta span con distintos estilos</title></head>

<body>

<h2>Ejemplo de etiqueta span</h2>

<p>La <span style="font-size:10pt; color:red; text-transform:uppercase">etiqueta span</span> permite definir comportamientos de una forma más flexible, tanto incrustando los estilos como usando hojas de estilos. Se pueden dar definiciones de estilos a elementos comunes como correo (<span class="correo" id="Correo_princ" style="color:blue;" title="Correo" dir="ltr" lang="es">midireccion@dominio.com</span>).</p>

</body>

</html>
```

Código Figura 37. SPAN con estilos y clases

En la anterior figura, podemos apreciar que se ha aplicado un estilo que ha convertido en mayúsculas el texto “etiqueta span”, lo ha pintado en pantalla con un tamaño de 10 puntos y con color rojo.

La dirección de correo ha sufrido otra transformación: se ha pintado en azul, se escribe de izquierda a derecha, y se indica al navegador que nuestro idioma es el español. Pero aquí aparece un elemento interesante, la clase del estilo de la etiqueta, o class. En función de cómo se defina el estilo de la clase en la hoja de estilo podremos pintar el texto con un tipo de letra, otro color sobrescribiendo el comportamiento del estilo, tamaño de texto y aplicarle multitud de transformaciones. Como puede apreciarse la flexibilidad que ofrece este sistema de etiquetado de texto es muy grande.

Hoy en día no se entiende como posible el uso de HTML sin CSS por las múltiples posibilidades de personalización y mantenimiento que ofrece. Las hojas de estilo se verán con mucho más detalle más adelante en el capítulo 6.

Texto preformateado

Span no es la única etiqueta genérica existente en HTML. Hasta ahora hemos indicado que los espacios y líneas en blanco fuera de etiquetas y en los párrafos son ignorados, y que es aconsejable utilizar códigos para caracteres especiales.
Sin embargo, en ocasiones, es necesario mostrar los espacios en blanco y caracteres de un texto que no se puede modificar. Se trata de un caso habitual cuando una página web debe mostrar directamente el texto generado por alguna aplicación.
Entonces es el momento de usar **la etiqueta <pre> que mostrará el texto tal como se ha escrito**, respetando todos los espacios en blanco y todas las nuevas líneas.

La siguiente figura muestra la diferencia tan notable que surge al utilizar la etiqueta <pre> y al no usarla.

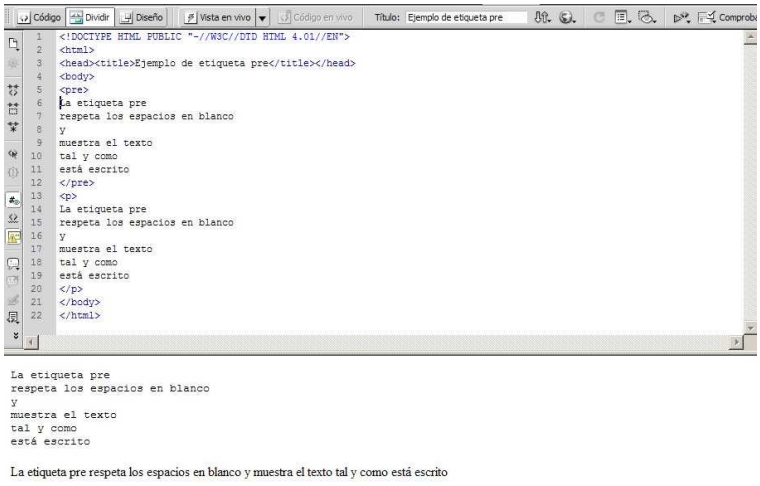


Figura 24. Diferencias uso de <PRE>

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01//EN">

<html>

<head><title>Ejemplo de etiqueta pre</title></head>

<body>
```

```
<pre>
La etiqueta pre
respeta los espacios en blanco
y
muestra el texto
tal y como
está escrito
</pre>
<p>La etiqueta pre
respeta los espacios en blanco
y
muestra el texto
tal y como
está escrito
</p>
</body>
</html>
```

Código Figura 38. Diferencias uso de <PRE>

Dejar constancia de cambios al texto

Cuando se edita o modifica el texto, suele ser frecuente querer citar otro texto o que quede constancia de en qué forma se ha cambiado dicho texto. Para lograr este objetivo se recurre a las etiquetas **<ins>** y **<del>.**

Su descripción formal es la siguiente:

Acciones	**Etiquetas y atributos**
Inserción	**<ins>texto</ins>** cite = "url". Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se realizó la modificación. datetime = "fecha". Especifica la fecha y hora en la que se realizó el cambio
Borrado	**~~texto~~** cite = "url". Indica la URL de la página en la que se puede obtener más información sobre el motivo por el que se realizó la modificación. datetime = "fecha". Especifica la fecha y hora en la que se realizó el cambio

Figura 25. Sintaxis <ins> y <del>

Un ejemplo del uso de ambas estás disponible a continuación:

Figura 26. Ejemplo etiquetas ins y del

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>

<head><title>Ejemplo de etiqueta ins y del</title></head>

<body>

<h2>Ejemplo de etiqueta ins y del</h2>
```

```
<p>Cuando se edita o modifica el texto, suele ser frecuente querer citar otro texto o que quede constancia de en que forma se ha cambiado dicho texto. Para lograr este objetivo se recurre a las <del datetime="20100201" cite="http://www.mipaginaweb.es.es/informacion.html"> etiquetas p y em</del> <ins datetime="20100201" cite="http://www.mipaginaweb.es.es/informacion.html">etiquetas ins y del</ins>.</p>

</body>

</html>
```

Código Figura 40. Ejemplo etiquetas ins y del

Otra situación muy frecuente en multitud de artículos y noticias es **citar textualmente porciones de texto**. En tales casos se recurre al uso de la etiqueta **<blockquote>**.

Acciones	Etiquetas y atributos
Citar	**<blockquote>texto</ blockquote >** cite = "url". Indica la URL de la página web donde se saca la cita.

Figura 27. Etiqueta blockquote

Figura 28. Ejemplo uso de etiqueta blockquote

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML
4.01//EN">

<html>

<head><title>Ejemplo de etiqueta
blockquote</title></head>

<body>

<p>Según el W3C, el valor del atributo
<em>cite</em> en las etiquetas
<strong>blockquote</strong> tiene el

siguiente significado:</p>
```

```
<blockquote
cite="http://www.w3.org/TR/html401/struct/text.html"
>"El valor de este

atributo es una dirección URL que indica el
documento original de la cita."</blockquote>

</body>

</html>
```

Código Figura 42. Ejemplo uso de etiqueta blockquote

Marcadores avanzados de texto

Dejamos para el final las etiquetas más recientes dentro del estándar HTML y que suelen incorporarse por páginas web avanzadas. Nos estamos refiriendo a las etiquetas para siglas, citas, abreviaturas y definiciones. Su sintaxis es la siguiente:

Acciones	Etiquetas y atributos
Sigla o acrónimo	<acronym title="texto" >…</ acronym> Texto es el significado de la sigla
Abreviatura	<abbr title="texto" >…</abbr> Texto es significado de la abreviatura
Cita	<cite title="texto"> …</cite> Texto indica a quien pertenece la cita, o su origen
Definición	<dfn title="texto"> Término</dfn> Texto define el termino

Figura 29. Etiquetas avanzadas

En el siguiente ejemplo veremos el uso que podemos hacer de estas etiquetas:

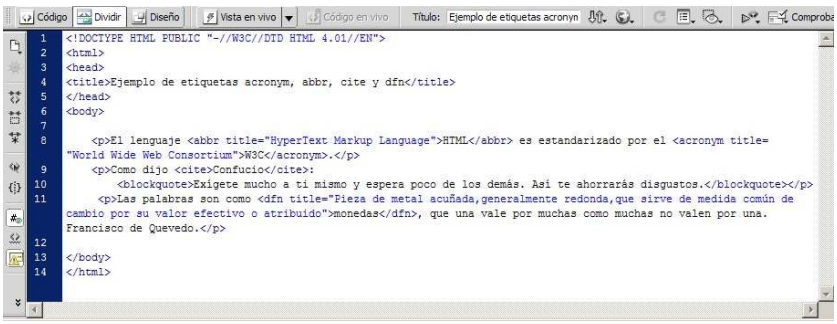


Figura 30. Ejemplos etiquetas avanzadas

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">

<html>

<head>

<title>Ejemplo de etiquetas acronym, abbr, cite y dfn</title>
```

```
</head>

<body>

        <p>El lenguaje <abbr title="HyperText Markup Language">HTML</abbr> es estandarizado por el <acronym title= "World Wide Web Consortium">W3C</acronym>.</p>

        <p>Como dijo <cite>Confucio</cite>:

                <blockquote>Exígete mucho a ti mismo y espera poco de los demás. Así te ahorrarás disgustos.</blockquote></p>

    <p>Las palabras son como <dfn title="Pieza de metal acuñada,generalmente redonda,que sirve de medida común de cambio por su valor efectivo o atribuido">monedas</dfn>, que una vale por muchas como muchas no valen por una. Francisco de Quevedo.</p>

</body>

</html>
```

Código Figura 44. Ejemplos etiquetas avanzadas

Como algunos de los resultados no se muestran en Dreamweaver con toda la amplitud que nos interesaría, en este caso hemos recurrido a abrirlo en el navegador web Firefox, pulsando sobre F12 dentro de esta aplicación. Se puede apreciar que al pasar el cursor del ratón sobre los elementos destacados, se muestra la definición o significado de la sigla o abreviatura. En la imagen siguiente se pueden ver todos estos efectos juntos. Evidentemente al ejecutar la página web y pasar el cursor sobre cada elemento, sólo se activa el seleccionado.

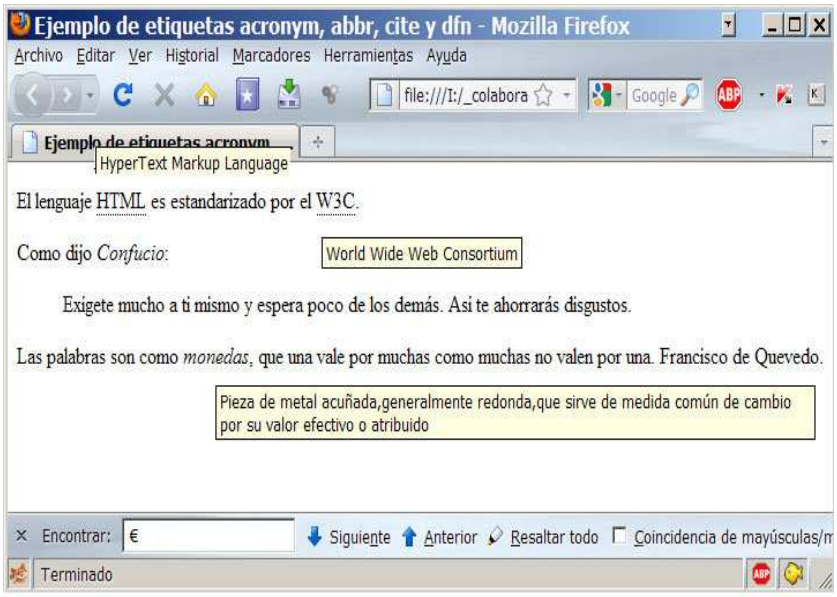


Figura 31. Resultados en Firefox

2.7 CREACIÓN DE TABLAS Y LISTAS

Desde el comienzo del lenguaje HTML, es posible mostrar la información en formato tabulado o tablas y generar listados de contenidos.

Las tablas igual que fuera del entorno web, pueden contener elementos simples, agrupaciones de filas y de columnas, cabeceras y pies de tabla, subdivisiones, cabeceras múltiples y otros elementos más complejos.

Las listas pueden contener índices numéricos o no y estar ordenadas o desordenadas.

Las tablas si se usan para lo que fueron ideadas, es decir para mostrar texto tabulado, ofrecen una opción perfecta para mostrar el texto ordenado. Sin embargo, su uso para distribuir elementos de la página web por pantalla, resulta muy polémico.

Puede resultar obvio, pero las tablas sólo deben utilizarse para mostrar información tabular. Otros usos, como definir la estructura de las páginas web, han quedado obsoletos y resultan totalmente desaconsejables por los problemas que plantean.

La técnica usada consistía en que la cabecera de la página era una fila de una gran tabla, el pie de página era otra fila de esta tabla y la zona de contenidos estaba formada por varias columnas dentro de esa gran tabla. Dentro de estos elementos había mezcla de una gran variedad de elementos, como imágenes, textos, enlaces e incluso los más osados incluían otros elementos de estructura como capas o marcos con acceso a otras páginas web.

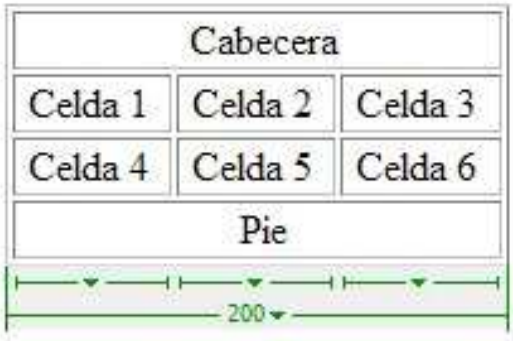


Figura 32. Tablas mal usadas

Conforme se complica la estructura, al navegador web le resulta más complicado dibujar la tabla, se complica en exceso el código HTML y su mantenimiento se hace tremendamente complejo.

Hoy en día existen muchos mecanismos más rápidos, eficientes y sin los problemas provocados por el uso de las tablas de forma errónea. Más adelante veremos los más usados como las hojas de estilos (CSS), los marcos y las capas.

Tablas básicas

Podemos ver un ejemplo de tabla en la siguiente imagen:

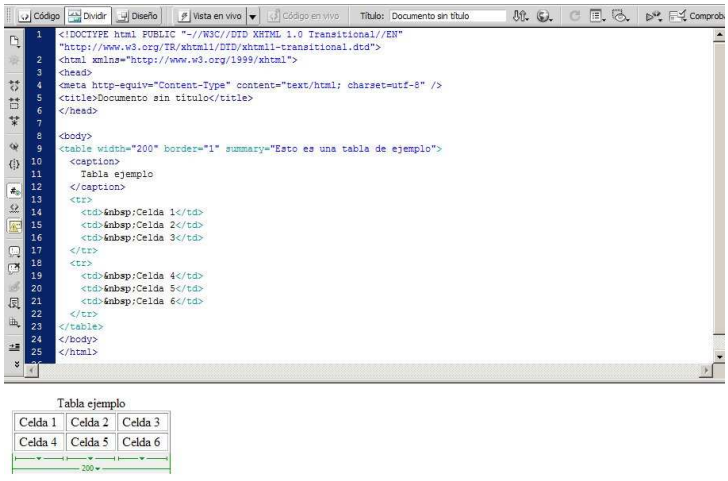


Figura 33. Ejemplo de tabla sencilla

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Documento sin título</title>

</head>
```

```
<body>

<table width="200" border="1" summary="Esto es una tabla de ejemplo">

 <caption>   Tabla ejemplo  </caption>

 <tr>

  <td> Celda 1</td>

  <td> Celda 2</td>

  <td> Celda 3</td>

 </tr>

 <tr>

  <td> Celda 4</td>

  <td> Celda 5</td>

  <td> Celda 6</td>

 </tr>

</table>

</body>

</html>
```

Código Figura 47. Ejemplo de tabla sencilla

En la anterior figura podemos ver el uso de las etiquetas más utilizadas para crear tablas como son **<table>**, **<tr>** para crear filas y **<td>** para crear celdas o columnas.

Al definir una tabla, primero se debe indicar las características generales de la tabla, luego definir las filas, y dentro de estas que columnas se tienen y sus particularidades. La sintaxis de estas etiquetas es la siguiente:

Acciones en tablas	Etiquetas y atributos
Definir una tabla	<table></table>
Borde tabla	<table border=0|1> 0 sin borde y 1 con borde
Sumario de tabla	<table summary = "texto" Usado por buscadores y personas discapacitadas
Ancho deseado	<table width=?|%> ? son pixeles en pantalla % es porcentaje, por ej. 30%
Espacio entre celdas	<table cellspacing=?> ? son pixeles en pantalla

Figura 34. Etiquetas y atributos de tablas

Acciones en filas	Etiquetas y atributos
Definir una fila	<tr></tr>
Alineación filas	<tr align=left|right|center|justify valign=top|middle|bottom>

Figura 35. Etiquetas filas

Acciones en columnas	Etiquetas y atributos
Definir columnas	<td></td>
Alineación filas	<td align=left|right|center|center valign=top|middle|bottom>
Sin cortar líneas	<td nowarp>
Ancho columna	<td width=?|%> ? número de pixeles % porcentaje tamaño tabla
Altura columna	<td height=?|%> ? número de pixeles % porcentaje tamaño tabla
Describir celda	<td abbr=”texto”> Texto descripción de celda (para personas invidentes)
Alcance	<td scope=”col|row|colgroup|rowgroup” scope (columnas, filas, grupo de columna o fila) las celdas para las que esta celda será su cabecera

Figura 36. Etiqueta columna

Resulta muy habitual que algunas de las celdas de la tabla se utilicen como cabecera de las demás celdas de la fila o de la columna. En este caso, HTML define la etiqueta **<th>** para indicar que una celda es cabecera de otras celdas. Si uso es similar a utilizar el atributo scope en la etiqueta <td> que permite indicar si la celda es cabecera de la fila o de la columna (<th scope="row"> y <th scope="col">).

La sintaxis de la etiqueta <th> es la siguiente:

Acciones	Etiquetas y atributos
Definir cabecera	<th></th>
Alineación	<th align=”left|right|center|justify” valign=top|middle|bottom>
Sin cortar líneas	<th nowarp>
Ancho columna	<th width=?|%> ? número de pixeles % porcentaje tamaño tabla
Altura columna	<th height=?|%> ? número de pixeles % porcentaje tamaño tabla

Describir celda	<th abbr=”texto”> Texto descripción de celda (para personas invidentes)
Alcance	<th scope=”col|row|colgroup|rowgroup” scope (columnas, filas, grupo de columna o fila) las celdas para las que esta celda será su cabecera

Figura 37. Etiqueta <th>

Una forma interesante de establecer el título o leyenda de una tabla es recurriendo a la etiqueta <caption> Esta etiqueta debe colocarse inmediatamente después de la etiqueta <table> y cada tabla sólo puede incluir una etiqueta <caption>. La sintasis de esta etiqueta es la siguiente:

Acciones	**Etiquetas y atributos**
Título tabla	<caption></caption>
Alineación	< caption align=”top|bottom”>

Figura 38. Etiqueta <caption>

Fusión de columnas y filas

Hasta ahora hemos omitido expresamente el uso de dos atributos de las filas y columnas muy usados. Se tratan de **rowspan** y **colspan**. Las tablas complejas suelen juntar varias columnas para formar otra más ancha o unir varias filas para formar otra más alta. Esto es lo que se conoce como fusión de columnas y filas.

En la siguiente figura se pueden ver ambos casos en acción en dos ejemplos:

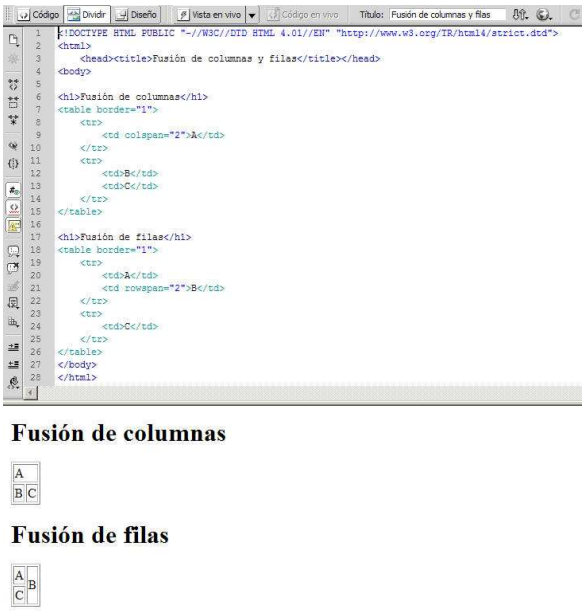


Figura 39. Atributos col span y rowspan

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

<html>

	<head><title>Fusión de columnas y filas</title></head>

<body>

<h1>Fusión de columnas</h1>

<table border="1">

	<tr>

		<td colspan="2">A</td>

	</tr>

	<tr>

		<td>B</td>

		<td>C</td>

	</tr>

</table>
```

```
<h1>Fusión de filas</h1>
<table border="1">
	<tr>
		<td>A</td>
		<td rowspan="2">B</td>
	</tr>
	<tr>
		<td>C</td>
	</tr>
</table>
</body>
</html>
```

Código Figura 53. Atributos col span y rowspan

Aunque a primera vista el uso de rowspan y colspan parezca sencillo, con relativa frecuencia es donde más errores se producen a la hora de generar tablas. El motivo de dicha situación es que muchos desarrolladores se olvidad de una restricción que impone HTML: las tablas deben disponer de una estructura regular. Es decir, una fila debe de tener el mismo número de columnas que el resto de las filas de la tabla. De forma análoga, toda columna debe tener el mismo número de filas. Si es necesario reducir el número de filas o columnas para que se cumpla la anterior norma se utilizará rowspan o colspan para reducir su número y así acomodarse a la norma.

En la siguiente figura podremos ver un ejemplo de un uso inadecuado de rowspan y colspan.

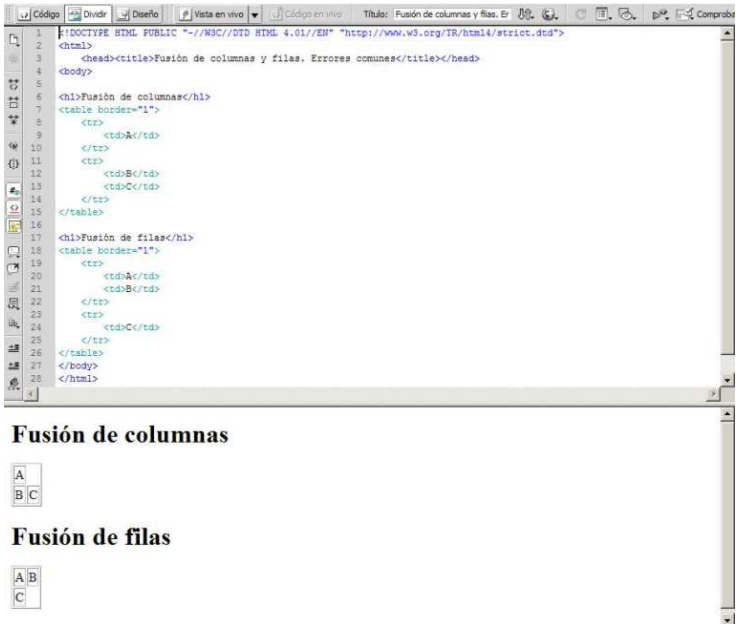


Figura 40. Errores en colspan y rowspan

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

<html>

    <head><title>Fusión de columnas y filas. Errores comunes</title></head>

<body>

<h1>Fusión de columnas</h1>

<table border="1">

    <tr>

        <td>A</td>

    </tr>

    <tr>

        <td>B</td>

        <td>C</td>

    </tr>

</table>
```

```
<h1>Fusión de filas</h1>
<table border="1">
        <tr>
                <td>A</td>
                <td>B</td>
        </tr>
        <tr>
                <td>C</td>
        </tr>
</table>
</body>
</html>
```

Código Figura 54. Errores en colspan y rowspan

Tablas avanzadas

Resulta muy habitual que las tablas más avanzadas posean muchos más elementos que filas y columnas. Ya hemos vistos un primer adelanto con el uso de títulos para las tablas. El asunto se complica cuando la tabla posee una sección de cabecera, una sección de pie y varias secciones de datos. Además, también es posible agrupar columnas y filas de forma lógica según el tipo de datos que posean, aplicándoles estilos gráficos similares.

Quizá uno de los mejores ejemplos podemos encontrarlos en las tablas que usa la aplicación Excel para temas de contabilidad.

Presupuesto Personal

	Ene	Feb	Mar	Abr	May	Jun	Jul	Ago	Sep	Oct	Nov	Dic	Año
Gastos totales	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €
Metálico/extras	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €
Ingresos													
Sueldo													0 €
Intereses/dividendos													0 €
Varios													0 €
Total	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €
Gastos													
Inicio													
Hipoteca/alquiler													0 €
Servicios													0 €
Teléfono fijo													0 €
Teléfono móvil													0 €
Reparaciones del hogar													0 €
Mejoras del hogar													0 €
Seguridad													0 €
Suministros para el jardín													0 €
Total	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €
Gastos diarios													
Alimentación													0 €
Cuidado de los niños													0 €
Tintorería													0 €
Cenar fuera													0 €
Servicio de limpieza de la casa													0 €
Cuidado del perro													0 €
Total	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €
Transportes													
Combustible													0 €
Seguro													0 €
Reparaciones													0 €
Lavado del coche/otros servicios													0 €
Aparcamiento													0 €
Transporte público													0 €
Total	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €	0 €

Figura 41. Tabla compleja

HTML soluciona este problema mediante las etiquetas **<tbody>, <thead>** y **<tfoot>**. Por otro lado, la sintaxis de estas etiquetas es la genérica propia de cualquier tabla.

La estructura habitual consiste en que la cabecera de la tabla se define con la etiqueta <thead>, el pie de la tabla se define mediante <tfoot> y cada sección de datos se define con una etiqueta <tbody>. Cada tabla puede contener solamente una cabecera y un pie, pero puede incluir un número ilimitado de secciones. Si se define una cabecera y/o un pie, las etiquetas <thead> y/o <tfoot> deben colocarse inmediatamente antes que cualquier etiqueta <tbody>.

El uso habitual de la etiqueta <tbody> es el de realizar **agrupaciones de filas**. En la siguiente figura podemos apreciar una agrupación por filas según el esquema anterior.

Exportaciones de productos hortofruticolas

AÑO	Ventas anuales Expansión de ventas (Toneladas) Tomates	Lechuga	Zanahorias	Pimientos
2006	-	-	-	-
2007	3	5	8	4
2008	4	4	7	3
2009	5	7	9	8
AÑO	Tomates	Lechuga	Zanahorias	Pimientos
	Expansión de ventas			

Figura 42. Tabla avanzada agrupada por filas

El código necesario para producir esta tabla se puede ver en la siguiente figura:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

        <head>

                <meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />

                <title>Ejemplo de tabla
avanzada</title>

        </head>

<body>

        <h3>Exportaciones de productos
hortofruticolas</h3>

        <table summary="Ventas anuales">

                <caption>Ventas anuales</caption>
```

Código Título tabla compleja

El primer elemento clave es el título englobado en el <caption>. Para conocer el funcionamiento recurriremos a seleccionar el texto vista de Código mostrado por Dreamweaver, quedando al momento referenciado en la vista de Diseño.

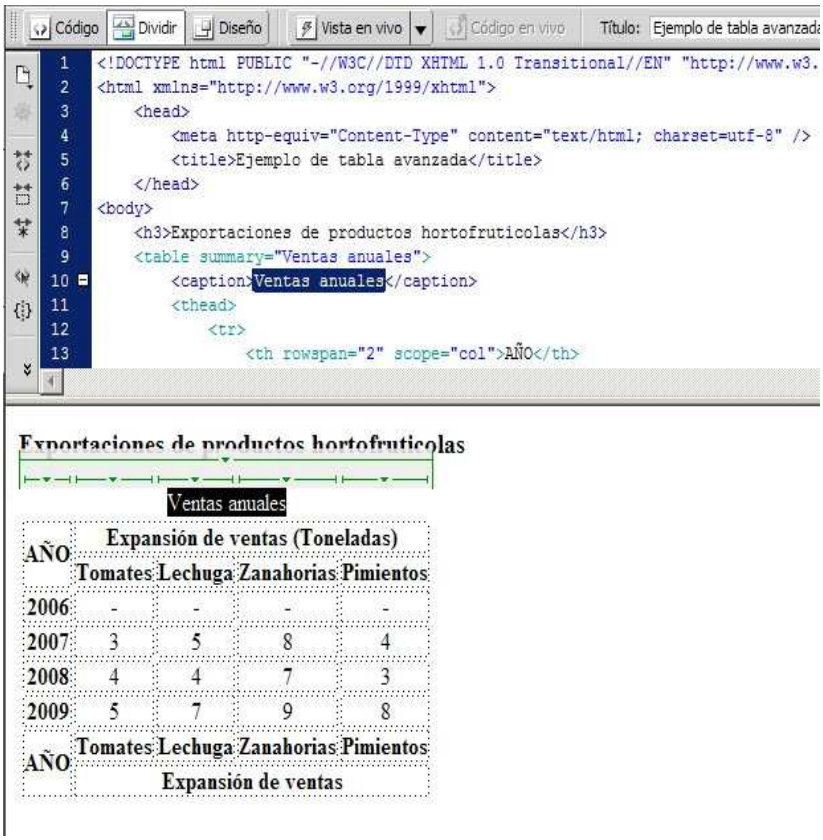


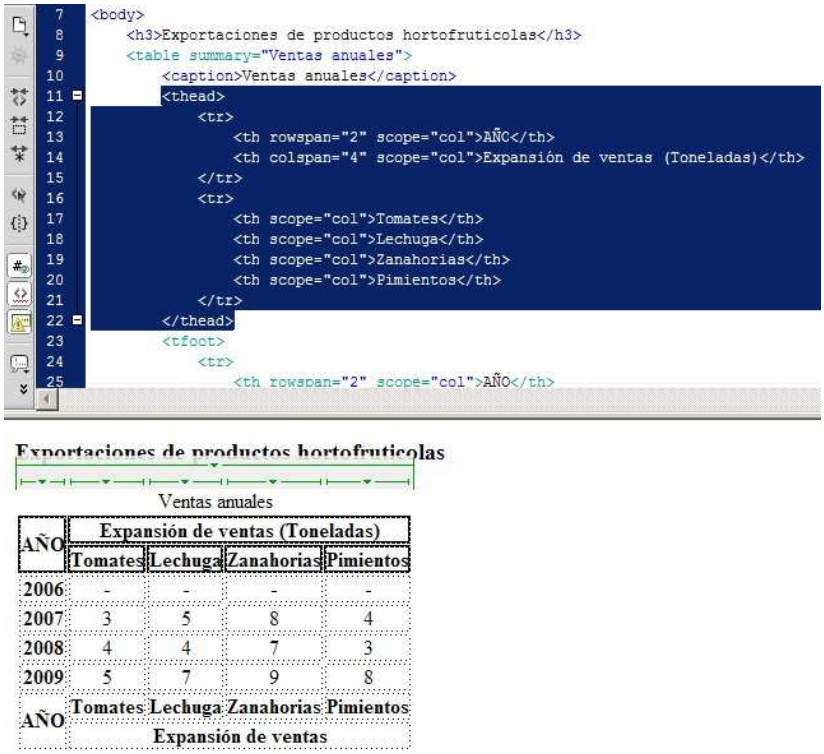
Figura 43. Título tabla compleja

A continuación se incluyen las cabeceras agrupadas por thead y th.

```
<thead>
    <tr>
        <th rowspan="2" scope="col">AÑO</th>
        <th colspan="4" scope="col">Expansión de ventas (Toneladas)</th>
    </tr>
    <tr>
        <th scope="col">Tomates</th>
        <th scope="col">Lechuga</th>
        <th scope="col">Zanahorias</th>
        <th scope="col">Pimientos</th>
    </tr>
</thead>
```

Código Cabecera compuesta

Hasta aquí lo que hemos “dibujado” se puede ver en la siguiente figura.



```
<body>
    <h3>Exportaciones de productos hortofruticolas</h3>
    <table summary="Ventas anuales">
        <caption>Ventas anuales</caption>
        <thead>
            <tr>
                <th rowspan="2" scope="col">AÑO</th>
                <th colspan="4" scope="col">Expansión de ventas (Toneladas)</th>
            </tr>
            <tr>
                <th scope="col">Tomates</th>
                <th scope="col">Lechuga</th>
                <th scope="col">Zanahorias</th>
                <th scope="col">Pimientos</th>
            </tr>
        </thead>
        <tfoot>
            <tr>
                <th rowspan="2" scope="col">AÑO</th>
```

Exportaciones de productos hortofruticolas

Ventas anuales

AÑO	Expansión de ventas (Toneladas)			
	Tomates	Lechuga	Zanahorias	Pimientos
2006	-	-	-	-
2007	3	5	8	4
2008	4	4	7	3
2009	5	7	9	8
AÑO	Tomates	Lechuga	Zanahorias	Pimientos
	Expansión de ventas			

Figura 44. Cabecera compuesta

El siguiente paso es incluir el pie de la tabla, que engloba sus elementos con tfoot. Se puede apreciar la obligación de anteceder tfoot al cualquier tbody.

```
                <tfoot>
                        <tr>
                                <th rowspan="2"
scope="col">AÑO</th>
                                <th
scope="col">Tomates</th>
                                <th
scope="col">Lechuga</th>
                                <th
scope="col">Zanahorias</th>
                                <th
scope="col">Pimientos</th>
                        </tr>
                        <tr>
                                <th colspan="4"
scope="col">Expansión de ventas</th>
                        </tr>
                </tfoot>
```

Figura 45. Código tabla avanzada agrupada por filas

Si repetimos el proceso de marcar el texto en Dreamweaver, se verá resaltado el pie en la vista de Diseño.

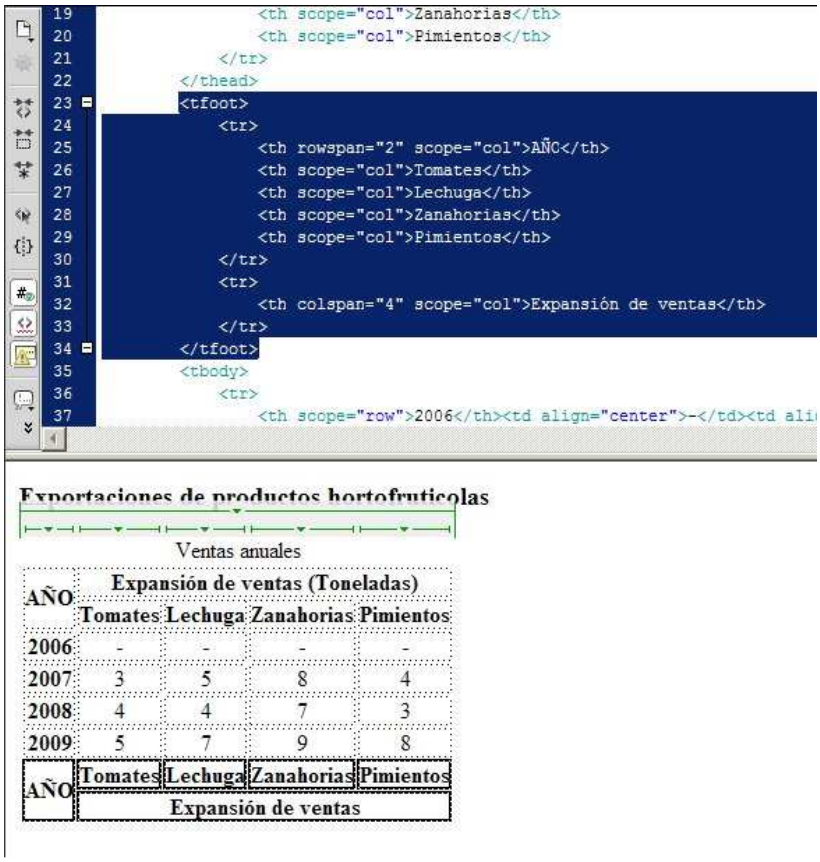


Figura 46. Pie compuesto en tabla compleja

El cuerpo de la tabla, incluido dentro de etiquetas tbody se aprecia en el siguiente código:

```
<tbody>
<tr>
<th scope="row">2006</th><td align="center">-</td><td align="center">-</td><td align="center">-</td>
<td align="center">-</td>
</tr>
<tr>
<th scope="row">2007</th><td align="center">3</td><td align="center">5</td><td align="center">8</td>
<td align="center">4</td>
</tr>
<tr>
<th scope="row">2008</th><td align="center">4</td><td align="center">4</td><td align="center">7</td>
<td align="center">3</td>
</tr>
```

```
                    <tr>

                                <th
scope="row">2009</th><td align="center">5</td><td
align="center">7</td><td align="center">9</td>

          <td align="center">8</td>

                    </tr>

               </tbody>

          </table>

</html>

</body>
```

Código Cuerpo de tabla compuesto y cierre programa

De nuevo remarcamos el texto en la vista de código de Dreamweaver, correspondiente al contenido de tbody. Se verá resaltado en la vista de Diseño la parte correspondiente.

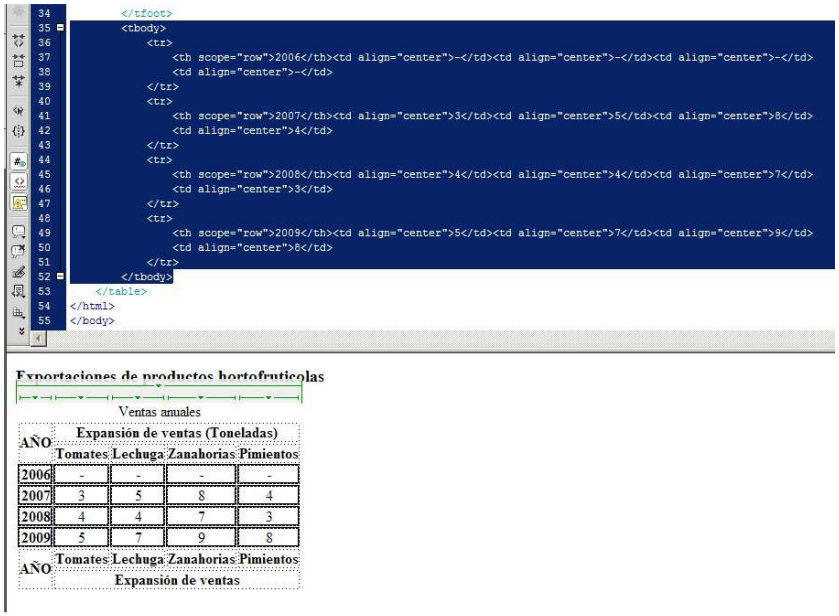


Figura 47. Cuerpo de tabla delimitado por tbody

Si lo que se desea es realizar **agrupaciones por columnas**, es necesario recurrir a las etiquetas **<col>** y **<colgroup>**. La etiqueta <col> se utiliza para asignar los mismos atributos a varias columnas de forma simultánea. De esta forma, la etiqueta <col> no agrupa columnas, sino que sólo asigna atributos comunes a varias columnas. Es importante resaltar que la etiqueta <col> solo puede usarse dentro de una tabla o de una etiqueta <colgroup>.

Se puede ver un ejemplo de uso en la siguiente figura:

```
<colgroup>
    <col style="width: 7.6em" />
    <col style="width: 3.8em" />
</colgroup>
<colgroup>
    <col span="3" style="width: 1.52em" align="right" />
    <col style="width: 2.28em" />
</colgroup>
```

Figura 48. Uso <etiqueta <col>

Hay que tener mucho cuidado con ambas etiquetas pues su uso no está muy extendido debido a que la mayoría de navegadores no soportan muchas de sus funcionalidades.

Se pretende demostrar este escenario con una situación típica. En el siguiente ejemplo, lo que pretenderemos será alinear las columnas de texto y darles color, gracias al uso de atributos generales de alineación y aplicar estilos de color. Para ello habrá que incluir las siguientes dos líneas de etiquetas:

<colgroup align="right" span="1" style="color:red;" />
<colgroup align="center" span="4" style="color:blue;" />

Su ubicación será justo después de la etiqueta **caption** que genera el título de la tabla.

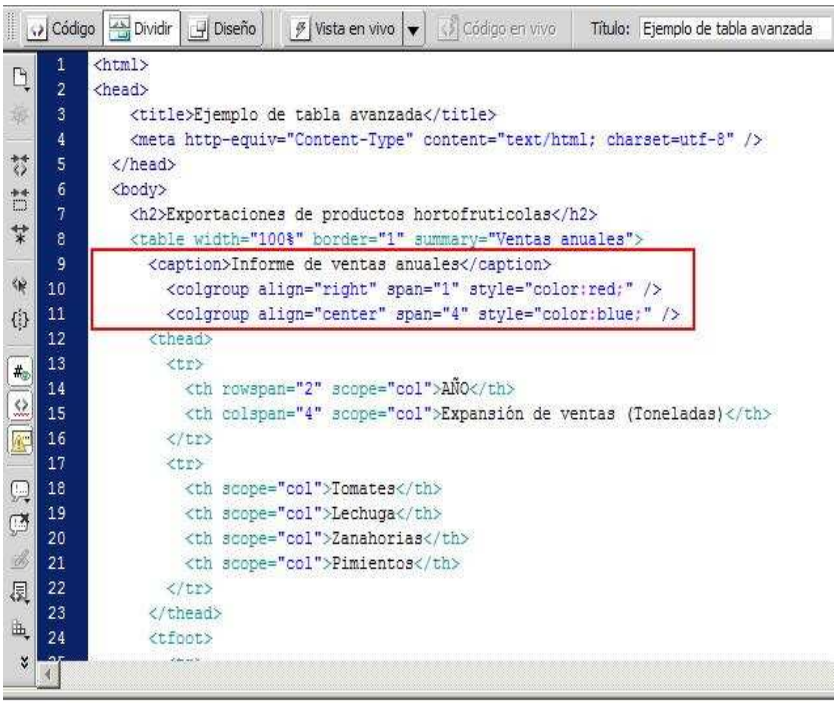


Figura 49. Código de columnas agrupadas

Un ejemplo de esta situación de incompatibilidad es que el navegador de Dreamweaver ha ignorado el color aplicado y parte de los estilos. Es decir, tanto la vista de Diseño como la vista en vivo muestran la siguiente interpretación del código anterior.

Exportaciones de productos hortofruticolas

Informe de ventas anuales

AÑO	Expansión de ventas (Toneladas)			
	Tomates	Lechuga	Zanahorias	Pimientos
2006	-	-	-	-
2007	3	5	8	4
2008	4	4	7	3
2009	5	7	9	8
AÑO	Tomates	Lechuga	Zanahorias	Pimientos
	Expansión de ventas			

Figura 50. Tabla avanzada con Dreamweaver

No sólo Dreamweaver presenta problemas. Existen algunos inconvenientes extra con diversos navegadores web, pues Firefox, Chrome, y Safari sólo soportan los atributos span and width. Otro detalle importante, es que el uso de <caption> junto con <colgroup> provoca que la página no valide en algunos navegadores (Safari por ejemplo).

El único navegador que actualmente interpreta correctamente los resultados es Internet Explorer. Si desde Dreamweaver se ejecuta la página, con el botón de Vista previa y elegimos este navegador, se tendrán los siguientes resultados:

Figura 51. Tabla avanzada en Internet Explorer

Listas

El cometido de las listas es agrupar palabras o frases breves que poseen un significado si se muestran juntas. Por ejemplo, los elementos de un menú, un conjunto de ingredientes para una receta de cocina, una serie de tareas…

El lenguaje HTML define tres tipos de listas, las listas no ordenadas (donde es indiferente el orden), las listas ordenadas (el orden se establece con alguna secuencia de letras o números) y listas de definición (con términos definidos como en un diccionario). Los tres tipos de listas suelen aparecer con sangría a la izquierda como en el siguiente ejemplo:

- Elemento 1
- Elemento 2
- Elemento 3

Listas no ordenadas

La etiqueta **<ul>** encierra todos los elementos de la lista, mientras que la etiqueta **<li>** identifica a cada uno de sus elementos. El resultado consiste en cada uno de los elementos precedidos por un punto pequeño y con una sangría de texto, como se puede apreciar si se teclea el código en Dreamweaver.

Figura 52. Listas no ordenadas

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Ejemplo de etiqueta UL</title>

</head>
```

```
<body>
        <h1>Menú</h1>
        <ul>
        <li>Inicio</li>
                <li>Noticias</li>
                <li>Artículos</li>
                <li>Contacto</li>
        </ul>
   </body>
</html>
```

Código Figura 66. Listas no ordenadas

Listas ordenadas

Dado que aquí sí que importa el orden, cada elemento se muestra en pantalla precedido por una secuencia numérica. Las etiquetas utilizadas son **<ol>** para englobar a la lista y **<li>** para cada uno de los elementos individuales.

Figura 53. Ejemplo de lista ordenada

```
<h1>Menú</h1>
<ol>
<li>Inicio</li>
        <li>Noticias</li>
        <li>Artículos</li>
        <li>Contacto</li>
</ol>
```

-Código Figura 67. Ejemplo de lista ordenada

Listas de definición

En este caso, la etiqueta que engloba a todos los elementos individuales es **<dl>**, mientras que **<dt>** se usa para el término de la definición y **<dd>** para la descripción de la definición.

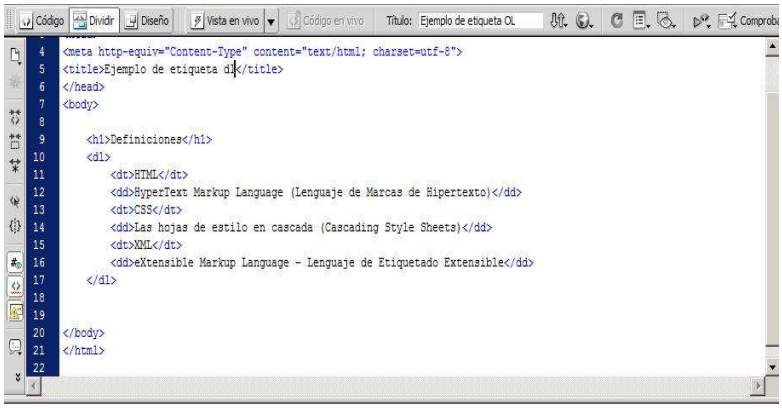


Figura 54. Etiquetas de definición de términos

```
<h1>Definiciones</h1>

    <dl>

        <dt>HTML</dt>

        <dd>HyperText Markup Language (Lenguaje de Marcas de Hipertexto)</dd>

        <dt>CSS</dt>

        <dd>Las hojas de estilo en cascada (Cascading Style Sheets)</dd>

        <dt>XML</dt>

        <dd>eXtensible Markup Language – Lenguaje de Etiquetado Extensible</dd>

    </dl>
```

Código Figura 68. Etiquetas de definición de términos

2.8 ENLACES Y DIRECCIONAMIENTOS

Los enlaces o hipertexto son parte de la esencia del lenguaje HTML. La clave de los enlaces radica en que permiten crear documentos interactivos. Una página web con enlaces permite relacionar áreas de contenido con otras áreas ya sea en el mismo documento o en otro distinto.

Los enlaces en HTML basan su funcionamiento en el uso de URLs (*Uniform Resource Locator*). Resumiento mucho, una URL no es otra cosa que una dirección que identifica al documento al que pretendemos acceder o referenciar y la forma de llegar hasta dicho documento.

Un ejemplo típico de URL sería:

http://www.dominio.com/dir1/paginaweb.htm?consulta=5#seccion

Si analizamos el ejemplo anterior se pueden extraer varios elementos:

- **Protocolo**: http://
- **Servidor**: www.dominio.com
- **Ruta**: /dir1/paginaweb.html
- **Consulta**: ?consulta=5.
    - Aporta información adicional necesaria para que el servidor localice correctamente el recurso que se quiere acceder. Si hay varias consultas, estas debe ir separadas por el símbolo &.
- **Sección**: #seccion
    - Permite que el navegador se posicione automáticamente en una sección de la página web. Siempre comienza con el carácter #.

El uso de los símbolos anteriores, **:**, **=**, **&** y **/** para separar las partes de una URL provoca que estos caracteres se encuentren reservados y no se puedan usar libremente. Además hasta hace muy poco, las

direcciones con caracteres distintos de los propios del idioma inglés también estaban reservadas, porque no todos los navegadores los entendían.
Toda esta situación hace que resulte muy frecuente codificar estos caracteres especiales en la URL, sobre todo si estamos trabajando con XHTML. Los códigos más usados son los siguientes:

Carácter	Carácter codificado
á	%E1
é	%E9
í	%ED
ó	%F3
ú	%FA
Á	%C1
É	%C9
Í	%CD
Ó	%D3
Ú	%DA
ñ	%F1
Ñ	%D1
/	%2F
:	%3A
=	%3D
"	%22
'	%60
espacio en blanco	%20
?	%3F
@	%40
&	%26

Figura 55. Codificación en caracteres seguros

A continuación, se puede ver varios **ejemplos de direcciones mal codificadas** y su correspondiente dirección correcta:

http://www.dominio.com/directorio/codificación.html

http://www.dominio.com/regiones/españa.html

http://www.dominio.com/dirección página.html

La forma **correcta** sería la siguiente:

http://www.dominio.com/directorio/codificaci%F3n.html

http://www.dominio.com/regiones/espa%F1a.html

http://www.dominio.com/direcci%F3n%20p%E1gina.html

Enlaces internos, externos, relativos y absolutos

Lo habitual es que las páginas web tengan enlaces a otras páginas almacenadas en su propio servidor, pero también tengan enlaces a recursos externos almacenados en otros servidores. En el primer caso, estos enlaces se conocen como internos mientras los segundos se refieren a enlaces externos.

Retomando la situación de los enlaces internos, resulta evidente que al conocer la estructura de ficheros del servidor no sea necesario conocer todos los elementos (protocolo, servidor y ruta) para acceder a todas las páginas.

Es decir, si nuestra dirección es:

http://www.dominio.com/dir1/dir2/dir3/pagina.html

Para acceder a otra página que se encuentre en la misma ruta sólo habrá que poner en el navegador web la URL correspondiente a pagina2.html. En este punto estamos presuponiendo que la dirección base es http://www.dominio.com/dir1/dir2/dir3/. Esto es lo que se conoce como **direccionamiento relativo**.

Las URL completas también se llaman **URL absolutas**, ya que el navegador no necesita disponer de información adicional para localizar el recurso enlazado. Si se utilizan siempre las URL absolutas, los enlaces están completamente definidos. Por ejemplo:

http://www.dominio.com/dir1/dir2/dir3/pagina2.html

Definir URLs absolutas resulta sencillo pues basta con poner toda la información sobre esa dirección. Sin embargo, su uso complica el código al incrementar el tamaño de los enlaces, y obliga al navegador web a recorrer **siempre** todos los directorios para acceder a una página. Si estamos ante direcciones externas, **de las que no conocemos su estructura**, no queda más remedio que utilizar el direccionamiento absoluto. Pero si estamos ante direcciones internas lo más habitual es utilizar direccionamiento relativo, no sólo para que el usuario le sea más sencillo manejar los enlaces, sino para que también el navegador web tenga menos trabajo en acceder a esas páginas.

Aunque el ejemplo mostrado es el caso más sencillo de URL relativa, existen otros casos más avanzados en los que se prescinde de parte o toda la ruta del recurso que se enlaza. A continuación se muestran algunos de los casos más habituales. Aquellos usuarios que hayan manejado sistemas operativos con línea de comandos (DOS, UNIX, Linux,…) estarán acostumbrados a la notación que entienden los navegadores web, pues es la usada en sus sistemas de ficheros y directorios.

Origen y destino enlace en mismo directorio

URL origen
http://www.dominio.com/dir1/dir2/dir3/pagina1.html
URL destino absoluta
http://www.dominio.com/dir1/dir2/dir3/pagina2.html
URL destino relativa
pagina2.html

Figura 56. Ejemplos direccionamiento 1

Destino del enlace en nivel superior

URL origen
http://www.dominio.com/dir1/dir2/dir3/pagina1.html

URL destino absoluta
http://www.dominio.com/dir1/dir2/pagina2.html
URL destino relativa
../pagina2.html

Figura 57. Ejemplos direccionamiento 2

Destino del enlace en nivel inferior

URL origen
http://www.dominio.com/dir1/dir2/dir3/pagina1.html
URL destino absoluta
http://www.dominio.com/dir1/dir2/dir3/dir4/pagina2.html
URL destino relativa
dir4/pagina2.html

Figura 58. Ejemplos direccionamiento 3

Destino del enlace alejado

URL origen
http://www.dominio.com/dir1/dir2/dir3/pagina1.html
URL destino absoluta
http://www.dominio.com/dir6/pagina2.html
URL destino relativa
/dir6/pagina2.html

Figura 59. Ejemplos direccionamiento 4

Hay que hacer varias matizaciones sobre los ejemplos anteriores. Si se debe subir varios directorios, se podrá repetir todas las veces necesarias el elemento **../**. Por ejemplo: **../../../paginas.html**. De igual forma, si es preciso navegar a directorios inferiores, se podrá completar la dirección según sea necesario hasta llegar al directorio requerido. Por ejemplo: **dir3/dir4/dir5/pagina2.html**. En el último ejemplo, la página destino está muy alejada del origen. En ese caso es mejor partir del origen del servidor. Para ello se usa el símbolo **/.** Por ejemplo: **/pagina3.html**.

Etiquetas para enlaces

El lenguaje HTML utiliza la etiqueta **<a>** para definir enlaces. En la siguiente figura se pueden ver algunos ejemplos:

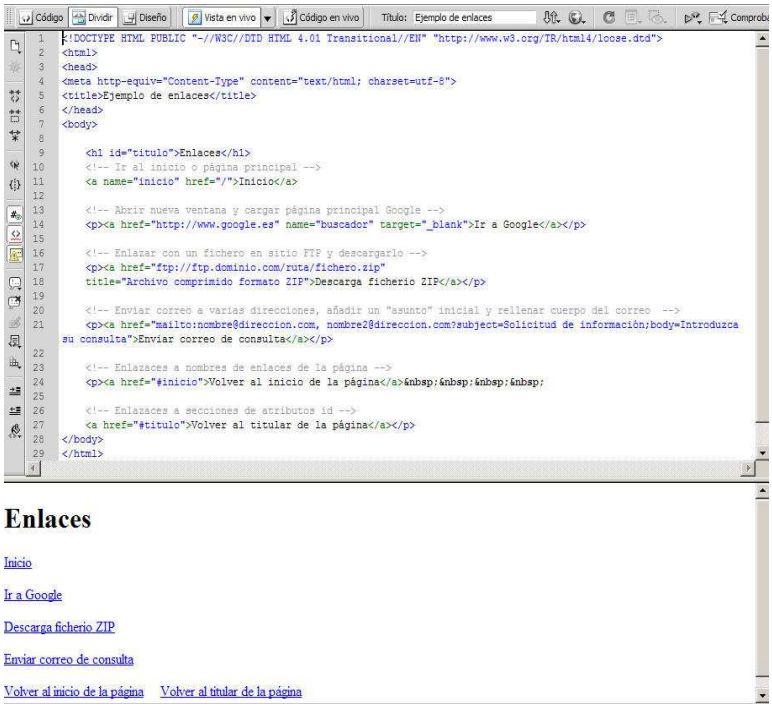


Figura 60. Ejemplos de uso de <a>

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">

<title>Ejemplo de enlaces</title>

</head>

<body>

<h1 id="titulo">Enlaces</h1>

   <!-- Ir al inicio o página principal -->

   <a name="inicio" href="/">Inicio</a>

      <!-- Abrir nueva ventana y cargar página principal
Google -->

           <p><a href="http://www.google.es"
name="buscador" target="_blank">Ir a
Google</a></p>
```

```
    <!-- Enlazar con un fichero en sitio FTP y
descargarlo -->

   <p><a href="ftp://ftp.dominio.com/ruta/fichero.zip"

   title="Archivo comprimido formato ZIP">Descarga
ficherio ZIP</a></p>

   <!-- Enviar correo a varias direcciones, añadir un
"asunto" inicial y rellenar cuerpo del correo  -->

         <p><a href="mailto:nombre@direccion.com,
nombre2@direccion.com?subject=Solicitud de
información;body=Introduzca su consulta">Enviar
correo de consulta</a></p>

   <!-- Enlazaces a nombres de enlaces de la página --
>

         <p><a href="#inicio">Volver al inicio de la
página</a>    

      <!-- Enlazaces a secciones de atributos id -->

         <a href="#titulo">Volver al titular de la
página</a></p>

</body>

</html>
```

Código Figura 74. Ejemplos de uso de la etiqueta <a>

Su sintaxis se puede apreciar en la siguiente figura:

Acciones	Etiquetas y atributos
Crear enlace	… url es dirección página web
Nombre enlace	 Texto es nombre del enlace
Destino enlace	 _top ventana original completa, cancelando así todos los demás marcos _parent en el Framset padre inmediato al marco actual _self en el mismo marco o página; _blank ventana nueva y sin nombre
Acción OnClick	 *** evento que se ejecuta cuando se pulsa el enlace
Acción OnMouseOver	 *** evento que se ejecuta cuando

	se pasa el cursor por el enlace
Acción OnMouseOut	 *** evento que se ejecuta cuando se quita el cursor por del enlace
Idioma recurso	
Tipo contenido	
Relación documento	
Codificación usada	

Figura 61. Etiqueta enlace <a>

Tras ver el anterior ejemplo, conviene resaltar algunos detalles.

La etiqueta **hreflang** se usa para indicar el idioma de la página a la que se enlaza. Este objetivo se nombra mediante diversas abreviaturas que identifican los distintos idiomas. Por ejemplo, es para español, es-AR para español argentino, es-ES español de España, en para inglés, en-US inglés de Estados Unidos, fr para francés, zn para chino, jp para japonés,…

El atributo **type** sirve para informar al navegador sobre el tipo de contenido que se enlaza. Se indica mediante una cadena de texto cuyos posibles valores también están estandarizados. Algunos de los valores de los contenidos más utilizados son los siguientes:

- "text/html" (páginas HTML)
- "image/png|jpg|gif" (imágenes con formato PNG, GIF, JPG,…)
- "text/css" (hojas de estilo CSS), "application/rss+xml" (archivos RSS)….

Los atributos **rel** y **rev** permiten indicar la relación que la página actual tiene con la página a la que se enlaza (atributo rel) y la relación que tiene la página enlazada con la página actual (atributo rev). Su uso no resulta muy habitual porque prácticamente sólo beneficia a los buscadores en su proceso de indexado y complica la generación de enlaces.

Los tipos de relación definidos son los siguientes:

- alternate – Indica que es una versión alternativa al documento actual (puede ser una versión en otro idioma o una versión preparada para otro medio, como una impresora o un dispositivo móvil)
- stylesheet – Indica que se ha enlazado una hoja de estilos
- start – Indica que se trata del primer documento de una colección de documentos.
- next – Indica que es el documento que sigue al actual dentro de una secuencia lógica de documentos (por ejemplo, los capítulos de un libro)
- prev - Indica que es el documento que precede al actual dentro de una secuencia lógica de documentos (por ejemplo, los capítulos de un libro).
- contents – Indica que el recurso enlazado es el documento que contiene la tabla de contenidos de la colección de documentos (por ejemplo, el índice de un libro).
- bookmark – Establece el enlace actual como un "marcador" o "favorito". Un marcador es un enlace que constituye un punto de entrada muy importante dentro del documento.

El atributo **charset** tiene como cometido indicar al navegador web que tipo de codificación se está utilizando en el enlace. Estos valores están estandarizados y las codificaciones más utilizadas son UTF-8 y ISO-8859-1, aunque existen decenas de códigos definidos (ISO-10646-UCS-2, IBM852, Big5-HKSCS, windows-1252, HZ-GB-2312).

Un último detalle sobre los atributos name e id. Es mucho más eficiente recurrir a referenciar a elementos que usen id, que crear enlaces "de forma artificial" que gracias a su etiqueta name podamos referenciarlos. Es decir, en lugar de crear una etiqueta:

<a name="titular">

Es más adecuado aprovechar las etiquetas ya existentes para mediante el atributo genérico id crear puntos de referencias. Por ejemplo:

<h1 id="titular">Texto</h1>

Enlaces a recursos cargados automáticamente

Hasta ahora hemos visto enlaces en los cuales es necesario pulsar sobre ellos para ir a otro lugar del documento o navegar a otra página web. Frente a este tipo de enlaces, existen otros que permite cargar automáticamente el contenido que engloban sin intervención del usuario. Las etiquetas usadas para estos casos son **<script>** y **<link>**. Cuando el navegador encuentra alguna de estas dos etiquetas, descarga los recursos enlazados y los aplica a la página web.

El uso de **<script>** está íntimamente asociado al lenguaje javascript. La etiqueta <script> (tanto cuando enlaza, como cuando incluye directamente el código) puede aparecer en cualquier parte del documento HTML, aunque normalmente se incluye dentro de la cabecera de la página (<head>...</head>).

La sintaxis de esta etiqueta es la siguiente:

Acciones	Etiquetas y atributos
Ejecución código	< script src="url"></ script > url es la dirección archivo
Tipo contenido	< script type = "tipo_contenido"></ script > Tipo_contenido habitualmente es JavaScript
No modifica página	< script defer = "defer"></script> Código no modifica página web
Tipo caracteres	< script charset = "tipo_caracteres"></ script >

Figura 62. Etiqueta <script>

La sintaxis de la etiqueta **<link>** es la siguiente:

Acciones	Etiquetas y atributos
Acceso a enlace	<link href="url"> url es dirección página web

Idioma recurso	<link hreflang =”idioma”>
Tipo contenido	<link type = “tipo_contenido”>
Relación documento	<link rel = “tipo_relación”> <link rev = “tipo_relación”>
Tipo de medio	<link media="tipo_medio"> Indica el medio para el que debe aplicarse la relación
Codificación usada	<link charset = “tipo_caracteres”>

Figura 63. Etiqueta link

Hoy en día el uso fundamental de la etiqueta **<link>** es cargar hojas de estilo. En cuanto a la sintaxis, la mayoría de los atributos de <link> son semejantes a los disponibles en los enlaces de tipo <a>. Al contrario que <script>, la etiqueta <link> solamente se puede incluir dentro de la cabecera del documento. Se pueden añadir tantas etiquetas <link> como sean necesarias, pero siempre dentro de <head>...</head>.

Un atributo distintivo es **media** y hace referencia al medio para el que es válida la relación con el recurso enlazado. Los medios disponibles también están estandarizados, siendo los más comunes screen para los contenidos mostrados en pantalla, print para las impresoras y handheld para los dispositivos móviles.

En la siguiente figura se pueden ver varios ejemplos de uso. El primer elemento <script> se encarga de cargar un fichero javascript externo y el segundo ejecuta código javascript incrustado que abre una ventana informativa.

En el caso de las etiquetas <link>, la primera carga una hoja de estilo externa, la segunda contenido RSS (Really Simple Syndication o sindicación realmente simple), y el último informa de que existe contenido imprimible en formato PDF.

Figura 64. Ejemplos de las etiquetas link y script

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">
```

```
<title>Ejemplo de script y link</title>

<script type="text/javascript"
src="http://www.ejemplo/js/inicializar.js"></script>

<script type="text/javascript">

window.onload = function() { alert("La página se ha
cargado completamente"); }

</script>

<link rel="stylesheet" type="text/css"
href="//css/comun.css">

<link rel="alternate" type="aplication/rss+xml"
title="Resumen de todos los artículos del blog"
href="/feed.xml">

<link rel="stylesheet" type="text/css"
href="//css/comun.css">

<link media="print" title="El tutorial en PDF"
type="aplication/pdf" rel="alternate"
href="http://www.dominio.com/curso/programación.p
df">

</head>

<body><h1> Ejemplo de script y link</h1></body>

</html>
```

Código Figura 78. Ejemplos de las etiquetas link y script

2.9 MARCOS Y CAPAS

Marcos

La mayoría de las páginas web disponen de estructuras con varias columnas, cabeceras, pies de página, secciones, etc. Hasta ahora hemos visto como formatear párrafos con texto, enlaces y tablas, pero no hemos indicado como estructurar páginas web complejas. En el pasado, durante un tiempo se utilizó para esta tarea a las tablas con los problemas ya comentados. La alternativa que proporciona el lenguaje HTML son los marcos, capas y como se verá más adelante mediante hojas de estilo.

Las etiquetas que pone a nuestra disposición el lenguaje HTML para crear marcos son **<frameset>, <frame>** y **<noframe>**. Si lo que pretendemos es crear marcos en línea o flotantes, se recurre a **<iframe>**.

La etiqueta <frameset> engloba una colección de etiquetas <frame> que construyen la ventana del navegador. Los atributos column y row de la etiqueta frameset permiten definir el número de columnas y filas y su tamaño. La etiqueta <frame> define que documento se carga (página HTML o cualquier otro

elemento multimedia). El cometido de la etiqueta <noframe> como su nombre indica es mostrar un contenido alternativo cuando no se haya podido cargar la parte contenida en los frames o marcos. Se suele utilizar para avisar de esta situación, es decir que se necesitan marcos para cargar la página y como indicador de que contenido poseen los diferentes marcos. Se puede apreciar la sintaxis de estas etiquetas en las siguientes figuras:

Acciones	Etiquetas y atributos
Definir colección frames	<frameset></frameset>
Id frameset	<frameset id =”texto”></frameset> Texto es identificación frameset
Título frameset	<frameset title=”titulo_frameset”></frameset>
Borde	<frameset border =”?”></frameset> ? número de pixeles; 0 = sin borde <frameset frameborder =”0|no|1|yes”></frameset>
Color borde	<frameset bordercolor =”no|código color”></frameset>
Filas	<frameset row=”fila”></frameset>
Columnas	<frameset column=”columna”></frameset>
Espacio frames	<frameset framespacing=”?”></frameset> ? número de pixeles de espacio entre frames

Figura 65. Etiqueta frameset

Acciones	Etiquetas y atributos
Definir frames	<frame scr=”url”></frame> url es dirección element que carga
Id frame	<frame id =”texto”></frame> Texto es identificación frame
Nombre frame	<frame name =”texto”></frame> Texto es identificación frame (referenciado por etiqueta target)
Título frame	<frame title=”titulo_frame”></frame>
Borde	<frame border =”?”></frame> ? número de pixeles; 0 = sin borde <frameset frameborder =”0|no|1|yes”></frameset>
Descripción	<frame longdesc =”texto”></frame> Texto descripción larga del frame
Margen altura	<frame marginheight =”?”></frame> ? son pixeles
Margen ancho	<frame marginwidth =”?”></frame> ? son pixeles
Sin cambio tamaño	<frame noresize =”noresize”></frame>
Sin barra desplazamiento	<frame scrolling =”no|yes”></frame>

Figura 66. Etiqueta frame

La etiqueta **<noframe>** no presenta una sintaxis específica. Se limita a englobar entre una etiqueta de apertura y otra de cierre los contenidos que aparecerán si no se pueden cargar los marcos.

En el caso de la etiqueta **<iframe>,** los valores son muy similares a los de la etiqueta <frame>, pero debido a algunas peculiaridades lo veremos con algo más de detalle más adelante.

El uso de los atributos frameborder, border y framespacing puede resultar complicado porque los navegadores web no se ponen de acuerdo en cuales aceptar. Los atributos frameborder y framespacing suelen ser aceptados por Internet Explorer, mientras que Firefox acepta border. Por tanto es aconsejable probar los resultados en varios navegadores.

Algunos matices con los atributos de la etiqueta <frame>. El atributo **scrolling** permite que el marco muestre una barra de desplazamiento cuando los contenidos del marco se desbordan. Si no se desea que determinado marco sea ajustado por el navegador de acuerdo al tamaño de la pantalla, entonces se recurre a **noresize**. Ambos atributos presentan comportamientos algo distintos en los navegadores web existentes, con lo que de nuevo se recomienda probar nuestras páginas web en distintos navegadores.

En cuanto a los atributos marginheight y marginwidth, el navegador dibuja un espacio entre los contenidos de cada marco y su borde. Con marginwidth se asigna el ancho y con marginheight se asigna la altura.

Dreamweaver, al igual que otras herramientas de desarrollo web suelen ofrecer diversos asistentes para generar marcos automáticamente. Estas opciones se pueden encontrar a la hora de crear un nuevo documento HTML.

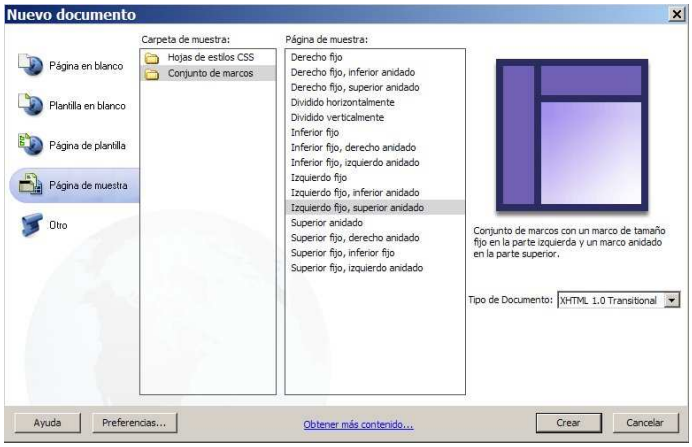


Figura 67. Asistente creación marcos en Dreamweaver

Tras generar la estructura con Dreamweaver, para el ejemplo en este caso se va a utilizar un editor de texto para completar los atributos. Dreamweaver al manejar *frames* o marcos tiene una costumbre muy incomoda a la hora de depurar código: Cuando tratas de previsualizar el resultado de la página con las etiquetas frameset y frame, te cierra la página con la que estás y te carga la última página html que ha cargado el marco. Por tanto, de momento nos olvidamos de Dreamweaver. El ejemplo de código es el siguiente:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
   "http://www.w3.org/TR/html4/frameset.dtd">
<HTML>
<HEAD>
<TITLE>Un ejemplo sencillo de marcos</TITLE>
</HEAD>
<FRAMESET cols="20%, 80%">
  <FRAMESET rows="100, 200">
     <FRAME src="menu.html">
     <FRAME src="../imagenes/verano-13.jpg">
  </FRAMESET>
  <FRAME src="inicio.html">
```

```
<NOFRAMES>
    <P>Este marco contiene:</p>
    <UL>
      <LI><A href="menu.html">El menu de la página</A>
      <LI><IMG src="contenido de verano-13.jpg" alt="Viajes">
      <LI><A href="inicio.html">Web principal</A>
    </UL>
 </NOFRAMES>
</FRAMESET>
</HTML>
```

Figura 68. Ejemplo de frame sencillo

Tras ejecutar este código en el navegador web Firefox, obtenemos el siguiente resultado. Como puede apreciarse, Firefox activa por defecto los bordes de los marcos. Dado que no los hemos definido, es el propio navegador web el que decide, y en este caso ha optado por mostrarlos.

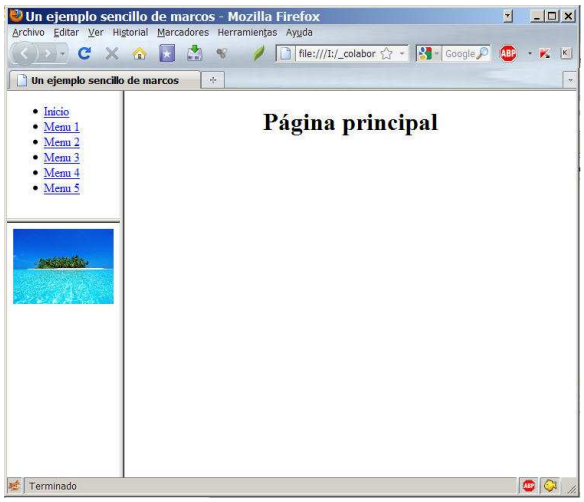


Figura 69. Etiquetas frameset y frame en acción

Debido al mal uso que se ha hecho en el pasado de los marcos, muchas páginas web evitan que se cargue su propia página desde un marco. Es decir que otra página web pueda por ejemplo poner su cabecera o logo en un marco y enlazar colocando debajo otro marco que carga otra página web como su fuese suya, con el consiguiente consumo de recursos para el servidor de la página cargada.

Manejo de filas y columnas en los marcos

Conviene hacer algunas matizaciones sobre los atributos de <frameset>. Los atributos **row** y **column** aceptan listas de valores separados por comas que indican la anchura o altura de los marcos. Para ello se utiliza el valor en pixeles o el porcentaje relativo del espacio restante. Cuanto más valores se tengan, más columnas o filas de marcos se tendrá. El navegador en el caso de los marcos posee un comportamiento peculiar. Intenta ajustar lo más cerca posible el tamaño de un marco, pero a diferencia de las tablas, si se exceden los márgenes indicados, el marco no se desborda. Por el contrario, el marco crea una barra de desplazamiento vertical u horizontal según sea el caso para que el usuario se desplace por el resto de los contenidos, salvo que se trate del marco principal o central que nunca tendrá barra de desplazamiento. Otro efecto añadido es que además calcula que proporción del espacio disponible en la ventana del navegador le toca a cada marco. Por ejemplo:

<frameset rows="150,300,150">

La anterior etiqueta crea tres filas de marcos, cada una extendiéndose a lo largo de la ventana del navegador. El primer marco y último tiene indicado que ocuparan 150 pixeles de alto, mientras que el segundo marco ocupara 300 pixeles. Sin embargo, a menos que la ventana del navegador sea exactamente de 600 pixeles de alto, el navegador ajustará de forma proporcional y automáticamente el primer y segundo marco para que ocupen un cuarto del espacio de pantalla disponible. El marco central ocupara la mitad del espacio disponible. Esta situación sería equivalente a:

<frameset rows="25%,50%,25%">

Por supuesto, si el conjunto de porcentajes no abarcan el 100% del espacio, el navegador ajusta automáticamente y proporcionalmente cada fila para abarcar la diferencia. Como a veces resulta conveniente poder manejar esta parte “sobrante” se recurre al uso del valor *. Su uso indica al navegador que ajuste las columnas y filas a cualquier espacio que quede, añadiendo un marco que ocupe este espacio. Por ejemplo, cuando el navegador encuentra la siguiente etiqueta **<frameset cols="100,*">,** crea una columna de ancho fijo de 100 pixeles y entonces crea otro marco que ocupa todo el espacio restante. Algunos ejemplos más avanzados del uso del asterisco podrían ser:

<frameset cols="10,*,10">

<frameset rows="*,100,*">

Las variaciones de uso pueden ser muchas. Por ejemplo, se puede anteceder al asterisco de un valor numérico. Por ejemplo:

<frameset cols="10%,3*,*,*">

En este caso se crean cuatro columnas. La primera ocupa 10% del espacio total disponible. El navegador entonces asigna tres quintos del restante espacio a la segunda columna y la tercera y cuarta reciben cada una un quinto del espacio restante.

Etiqueta iframe

La etiqueta iframe (por *inline frame* o *marco incorporado* en inglés) es un elemento HTML que permite insertar o incrustar un documento HTML dentro de un documento HTML principal. Un iframe puede considerarse como un agujero que se abre en una página web y a través del cual se muestra otra página web. En ocasiones se utiliza para mostrar contenidos externos al sitio web como si fueran parte del mismo sitio. Otras veces se emplea para incluir una aplicación común a varios sitios web de una misma empresa. Por ejemplo una aplicación de validación de usuarios.

Fue introducido en el navegador Microsoft Internet Explorer en 1997 y durante mucho tiempo solo fue soportado por este navegador. La etiqueta **<iframe>** actualmente es ya ampliamente soportado por gran variedad de navegadores. Podemos ver un ejemplo en la siguiente figura:

```
<html>
  <head>
    <title>IFrames</title>
  </head>
   <body>
  <h1> Contenido de nuestra web</h1>
      <iframe src="http://www.google.es/"
     width="600" height="400" scrolling="auto" frameborder="1" transparency>
     <p>Texto alternativo para navegadores que no aceptan iframes.</p>
    </iframe>
    <h2> Otro contenido de nuestra web</h2>
  </body>
</html>
```

Figura 70. Ejemplo de iframe

Cuando abrimos este ejemplo en Firefox, tendremos el resultado que se puede apreciar en la siguiente figura. Como se puede apreciar, se ha creado un iframe con una achura y altura determinada. La página que cargamos, el inicio de Google, posee un tamaño mayor del indicado, por lo que para desplazarte por esta página se dispone de barras de desplazamiento.

Figura 71. Iframes en Firefox

Capas

Ya hemos visto una forma de agrupar elementos de forma genérica, la etiqueta . Otra de las más usadas es la etiqueta , utilizada para crear capas. La característica de este código HTML es que se encarga de agrupar los elementos de la página en diferentes divisiones en función de su finalidad: la zona de la cabecera de la página, la zona de contenidos, una zona lateral para el menú y otras secciones menores, la zona del pie de página, etc. A cada zona se le aplica unos estilos diferentes, mediante hojas de estilo.

El nombre de la etiqueta div tiene su origen en la palabra división, ya que esta etiqueta define zonas o divisiones dentro de una página HTML. En cualquier caso, casi todos los diseñadores web utilizan la palabra "capa" para referirse a una "división". Aunque se trata de un error grave (las capas se crean mediante una propiedad de CSS llamada z-index) es preferible seguir llamando *"capas"* a las zonas definidas con la etiqueta para poder entenderse con el resto de diseñadores.

Las páginas web complejas que están bien diseñadas utilizan decenas de etiquetas . Con mucha diferencia, los atributos más utilizados con esta etiqueta son **id** (para identificar la capa de forma única) y **class** (para aplicar estilos CSS a la capa).

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html;
charset=utf-8">

<title>Ejemplo de etiqueta div</title>

</head>

<body>
```

```
<body>
        <DIV id="cliente-boyer" class="cliente">
                <P><SPAN class="cliente-
titulo">Información sobre el cliente:</SPAN>
                <TABLE class="cliente-datos">
        <TR><TH>Apellido:<TD>Boyer</TR>

        <TR><TH>Nombre:<TD>Esteban</TR>
                        <TR><TH>Tel:<TD>(91) 123
45 67</TR>
        <TR><TH>Email:<TD>esb@fdominio.com</
TR>
                </TABLE>
        </DIV>
        <DIV id="cliente-lafuente" class="cliente">
                <P><SPAN class="cliente-
titulo">Información sobre el cliente:</SPAN>
                <TABLE class="cliente-datos">
        <TR><TH>Apellido:<TD>Lafuente</TR>
```

```
        <TR><TH>Nombre:<TD>Albert</TR>
                        <TR><TH>Tel:<TD>(93) 123
45 67</TR>

        <TR><TH>Email:<TD>albert@dominio.com
</TR>
                </TABLE>
        </DIV>
</body>
</html>
```

Código Figura 86. Ejemplo de uso de etiqueta div

Como no hemos incluido la parte CSS en el código, este no aparece aplicado. Cada atributo class hace referencia a una clase que forma parte de una hoja de estilo y que se encargaría de aplicar color, tamaño de fuente, alineación y un largo etcétera de transformaciones si se desea. Más adelante, en el capítulo 6 trataremos con detalle las hojas de estilo.

A pesar de que no se apliquen las hojas de estilos porque no se las hemos incluido, la ventana de diseño de Dreamweaver ya puede ofrecer una idea de cómo se ha agrupado el contenido de la página.

Figura 72. Agrupación con div

En este momento, no se va a profundizar en el proceso de diseñar una página web mediante , ya que todavía no conocemos el uso de hojas de estilo (CSS). En cuanto tratemos las hojas de estilo usaremos constantemente esta etiqueta, pues junto con son las más usadas en tales situaciones.

3

Imágenes y elementos multimedia

- Inserción de imágenes: formatos y atributos
- Mapas de imágenes
- Inserción de elementos multimedia: audio, video y programas
- Marquesinas

Las imágenes son uno de los elementos más importantes de una página web. Si se abre cualquier página web al azar, lo más probable es que contenga muchas imágenes. Aquí es interesante hacer la distinción entre imágenes parte del documento web y aquellas que están sólo de “adorno”. En este último caso, como ocurría con las tablas, es muy aconsejable recurrir a hojas de estilo para mostrarlas, con el fin de optimizar la página y hacerla más eficiente cuando se dibuja en pantalla.

3.1 INSERCIÓN DE IMÁGENES: FORMATOS Y ATRIBUTOS

La etiqueta para las imágenes es **<img>**. A continuación describiremos sus atributos más característicos.

Acciones	Etiquetas y atributos
Mostrar imagen	<img src=”url”> url es dirección imagen
Alinear imagen	<img src=”url” align=top|bottom|middle|right|left>
Ancho y alto imagen	<img src=”url” width="?|%" height="?|%"> ? ancho y alto en pixeles % en porcentaje
Nombre imagen	<img src=”url” name=”texto”>
Descripción larga	<img src=”url” longdesc=”url”>
Texto alternativo	<img src=”url” alt=”texto”>
Borde	<img borde=”pixeles”>
Espacio horizontal	<img hspace=”pixeles”>
Espacio vertical	<img borde=”pixeles”>

Figura 73. Formatos de la etiqueta <img>

Evidentemente los atributos clave de la etiqueta <img> son **src** y **alt.** El primero, src, indica la dirección donde se va a encontrar el fichero gráfico que se va a cargar. El direccionamiento, igual que en los enlaces puede ser absoluto y relativo. Los formatos gráficos aceptados por los navegadores web actuales son GIF, JPG y PNG.

El atributo **alt** contiene el texto alternativo que aparecerá en pantalla si la imagen no se puede cargar. Este atributo es muy importante para que los buscadores puedan indexar los contenidos gráficos y por motivos de accesibilidad para que aplicaciones usadas por personas con discapacidades visuales puedan leerles el contenido de la pantalla.

Los atributos **height** y **width** son utilizados para indicar el alto y ancho de la imagen en pixeles. No son obligatorios, y si se omiten, la imagen se pintará en pantalla a su tamaño original.

Si se quiere ampliar la imagen, sólo hay que poner valores mayores a los de la imagen original. Evidentemente para reducir el tamaño de la imagen se deben poner valores inferiores al original. Si se omite sólo uno de estos atributos, el otro se escalará proporcionalmente. Es decir, el navegador web calculará el ancho o alto necesario para mantener la proporción.

Si usamos porcentajes, estos valores se referirán al tamaño de la imagen. Por ejemplo, un valor del 30% en height o en weight, indica que el ancho o alto de la imagen que se pinte en pantalla tendrá el 30% de su tamaño original.

El uso de la etiqueta **longdesc** no es muy habitual, y se reserva para indicar la dirección donde se encontrará el fichero que aporta más información sobre la imagen.

Los atributos **hspace** y **vspace** se establece la distancia en pixeles de la imagen a los objetos que se encuentran horizontal y verticalmente.

Un último detalle importante. Si usamos XHTML, dado que la etiqueta <img> no tiene elemento de cierre, habrá que terminarla según la siguiente estructura:

****

Esta forma de cerrar las etiquetas que no tienen elementos de cierre es obligatoria si se trabaja con cualquiera de las versiones de XHTML. Otro detalle importante con respecto a XHTML y las comillas: En HTML es aconsejable poner los atributos entre comillas, pero no es obligatorio. En XHTML siempre es obligatorio poner entre comillas los valores de los atributos.

En la siguiente figura podemos apreciar distintos usos de los atributos de la etiqueta <img>.

Figura 74. Ejemplo de uso de <img>

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />

<title>Ejemplo de uso de la etiqueta img</title>

</head>
```

```
<body>

<p><img src="../imagenes/logo_KAMS.jpg"
alt="logotipo KAMS" name="KAMS" width="157"
height="69" align="left" id="KAMS"
longdesc="http://www.kamssistemas.com/info.html"
/></p>

<p><br />

        <img
src="http://www.google.es/intl/en_com/images/logo_p
lain.png" alt="Logo de Google"
longdesc="http://www.google.es" />

</p>

</body>

</html>
```

Código Figura 88. Ejemplo de uso de <img>

La segunda imagen puede no previsualizarse bien con algunos navegadores, pues algunos no soportan bien las imágenes con extensión PNG. En este caso recurrimos a Firefox, que no tiene problemas con esta extensión de imágenes y tras abrir la página web creada, tendremos el resultado siguiente:

Figura 75. Resultados ejemplos con <img>

3.2 MAPAS DE IMÁGENES.

El funcionamiento de un mapa de imágenes se basa en definir distintas zonas dentro de una imagen para que al pinchar sobre dichas zonas se nos dirija a determinadas URLs. La idea en principio resulta interesante, pero dado que su implementación resulta algo pesada y laboriosa, no compensa con frecuencia el esfuerzo.

Los mapas de imagen fueron muy utilizados en el pasado. Hoy en día su uso ha decaído mucho quedando reducido fundamentalmente a páginas web de empresas de viajes o con contenido geográfico que recurren a una imagen con un mapa, y que en función de que zona de dicho mapa se pulse, te redirige a una URL con contenido asociado con esa zona geográfica.

El funcionamiento de un mapa de imágenes se basa en introducir primero la imagen con la etiqueta **<img>**. A continuación se definen las zonas que tendrán un mapa de imagen con la etiqueta **<map>**. Estas zonas tendrán forma rectangular, oval o poligonal. Finalmente se define cada zona que se puede pulsar con la etiqueta **<area>.**

La sintaxis de estas etiquetas es la siguiente:

Acciones	**Etiquetas y atributos**
Crear mapa	**<map name ="texto" >…</map>** Siendo texto el nombre de la imagen
Área de mapa de imágenes	**<area>** **href="url"** URL a la que se accede al pinchar sobre el área **nohref = "nohref"** URL para las áreas que no son seleccionables **shape="default| rect | circle | poly"** Tipo de área que se define: toda la imagen, rectangular, circular o poligonal. **coords = "coordenadas"** Coordenadas son lista de números separados por comas. *Rectangular* = X1,Y1,X2,Y2 (coordenadas X e Y del vértice superior izquierdo y coordenadas X e Y del vértice inferior derecho). *Circular* = X1,Y1,R (coordenadas X e Y del centro y radio del círculo). *Poligonal* = X1,Y1,X2,Y2,...,XnYn (coordenadas de los vértices del polígono.

Figura 76. Etiquetas map y area

En la siguiente figura se puede ver un ejemplo de uso de estas etiquetas y las dos zonas de interacción generadas. Para lograr este objetivo, primero se define una capa con la etiqueta , y dentro de esta el mapa y las áreas pulsables. La primera nos dirigiría a la dirección web http://www.google.com, y la segunda a http://www.google.es.

Figura 77. Mapa de imagen generado sobre el logo de Google

```
<html>

<head>

<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />

<title>Ejemplo mapa de imágenes</title>

</head>

<body>
```

```
<div id="imgmapdiv">

<map name="imgmap">

        <area shape="poly"
coords="119,14,116,92,10,93,8,8,119,14"
href="http://www.google.com" alt="1">

        <area shape="poly"
coords="123,92,263,95,262,14,130,12,123,92"
href="http://www.google.es" alt="2">

    </map>

 </div>

 <p>

    <img
src="http://www.google.com/intl/en_ALL/images/logo
.gif" alt="Move mouse over image"
usemap="#imgmap">

 </p>

</body>

</html>
```

Código Figura 91. Mapa de imagen generado sobre el logo de Google

Debido a que crear un mapa de imágenes puede resultar una tarea pesada, pues delimitar las áreas de interacción no siempre resulta sencillo, es muy frecuente recurrir a páginas web o aplicaciones que te permite generar el código necesario para una imagen dada. Básicamente estas aplicaciones se encargar de generar las coordenadas del mapa de imágenes de forma gráfica. Es decir se va seleccionando cada uno de los vértices de la figura, la aplicación detecta sus coordenadas y las copias al código.

Dos buenos ejemplos de este tipo de recursos pueden encontrarse en las siguientes direcciones web:

HTML - Image map Creator WYSIWYG

http://www.kolchose.org/simon/ajaximagemapcreator/

Wikipedia

http://en.wikipedia.org/wiki/Image_map

3.3 INSERCIÓN DE ELEMENTOS MULTIMEDIA: AUDIO, VIDEO Y PROGRAMAS

El estándar HTML permite incorporar todo tipo de elementos a las páginas web: videos, Flash, applets Java, sonidos, aplicaciones, etc. La característica común a todos estos elementos multimedia es que habitualmente los navegadores web no los entienden. Es necesario instalar alguna aplicación añadida que permita "traducir" el manejo de estos contenidos.

Para insertar elementos multimedia se recurre a la etiqueta **<object>**. La siguiente figura muestra su sintaxis:

Acciones	Etiquetas y atributos
Definir un objeto	<object data="url"></object> url es dirección de los datos
Tipo de objeto	<object type="tipo_contenido"s"></object>
Altura objeto	<object height="?"></object> ? alto en pixeles
Ancho objeto	<object width="?"></object> ? ancho en pixeles

Figura 78. Etiqueta object

Los atributos **classid**, **codebase** y **codetype** no se muestran en la anterior figura, pues depende mucho del tipo de objeto del que se trate, desapareciendo en buena parte de ellos.

Los posibles valores del atributo **type** están estandarizados. Los más habituales son text/html, image/jpeg, audio/mpeg, video/quicktime, text/css, and text/javascript. Se pueden encontrar muchos más elementos en la siguiente dirección:

http://www.iana.org/assignments/media-types/

Con la ayuda del siguiente ejemplo de código, se va a introducir el uso de las etiqeutas **<param>** y **<embeb>** junto con **<object>**. El primer detalle que se aprecia es que la vista de diseño de Dreamweaver no es capaz de interpretar los resultados. Si se recurre a la vista en vivo o se abre la página web en un navegador, por ejemplo Firefox, ya se podrán ver los resultados como si se tratara de un navegador web típico.

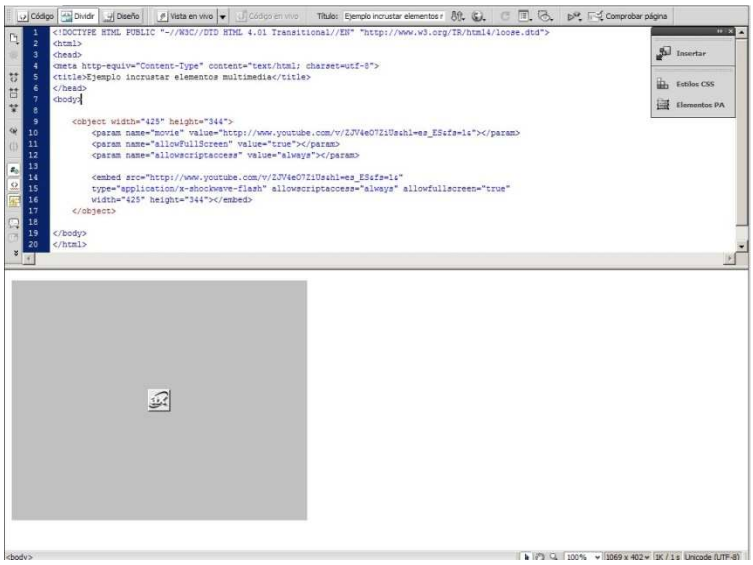


Figura 79. Incrustar elementos multimedia

Figura 80. Resultados en Firefox

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Ejemplo incrustar elementos multimedia</title>

</head>

<body>
```

```
        <object width="425" height="344">

                <param name="movie" value="http://www.youtube.com/v/ZJV4eO7ZiUs&hl=es_ES&fs=1&"></param>

                <param name="allowFullScreen" value="true"></param>

                <param name="allowscriptaccess" value="always"></param>

      <embed src="http://www.youtube.com/v/ZJV4eO7ZiUs&hl=es_ES&fs=1&"

      type="application/x-shockwave-flash" allowscriptaccess="always" allowfullscreen="true"

      width="425" height="344"></embed>

          </object>

</body>

</html>
```

Código figura 93. Incrustar elementos multimedia

La primera etiqueta, **<param>**, se usa para incluir parámetros requeridos para la apertura correcta de los contenidos multimedia, en este caso un video en formato Flash.

Sus atributos son **name** para el nombre del parámetro y **value**, para el valor de dicho parámetro.

El caso de **<embeb>** es muy particular. Sus parámetros son idénticos al caso de la etiqueta <object> aunque su comportamiento es distinto.

Uno de los principales inconvenientes de <object> es la forma de incluir vídeos en formato Flash en las páginas HTML. Algunos navegadores como Internet Explorer, cuando muestran el contenido de un elemento multimedia con object, no visualicen el vídeo hasta que se ha descargado completamente. Si se trata de un vídeo largo, esta solución no es válida para el usuario. Por este motivo, se utiliza una solución alternativa para incluir vídeos Flash en las páginas HTML: el uso de la etiqueta <embed>.

Aunque esta solución funciona correctamente, no se trata de una solución válida desde el punto de vista del estándar de los estándares web. Esta situación es especialmente importante en XHTML, pues en tales casos, las páginas que incluyan esta solución no pasarán correctamente el proceso de validación.

Dreamweaver incorpora diversos asistentes para añadir automáticamente elementos multimedia a nuestro código HTML. Evidentemente presta especial atención a aquellos que dependen del propio desarrollador de la aplicación, como son elementos Flash (SWF) o vídeos en el anterior formato (FLV). Estas opciones están disponible dentro del menú lateral de **Insertar**, dentro del apartado de **Común** y **Media**.

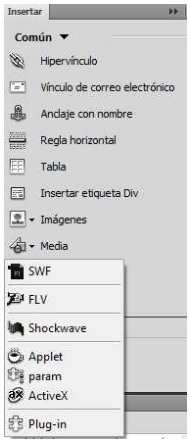


Figura 81. Asistente elementos multimedia en Dreamweaver

A veces resulta complicado saber cuáles son los parámetros adecuados para incorporar elementos multimedia en una página web. Esto se debe a que con frecuencia dependen de la aplicación externa que se haya instalado en el navegador web para que entienda el formato multimedia.

Para solucionar este problema, multitud de páginas web que manejan elementos multimedia, como por ejemplo YouTube, incluyen un apartado donde se indica cual es el código HTML que hay que incluir en nuestra página web para poder visualizar adecuadamente el contenido.

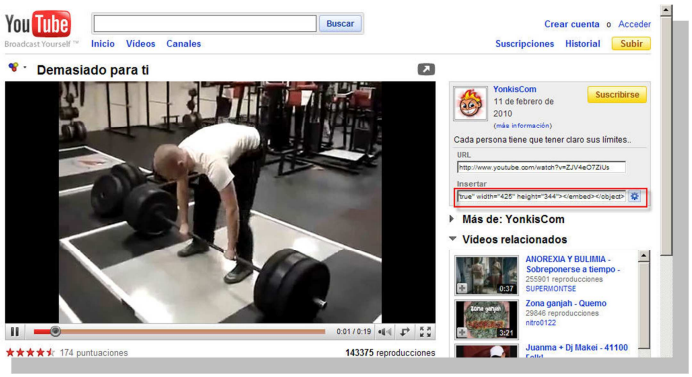


Figura 82. Video incrustado en página web

Se puede apreciar en la anterior figura la marca a la zona donde está disponible el código HTML que se debe copiar para incrustar este vídeo en cualquier página web.

3.4 MARQUESINAS

La etiqueta **<MARQUEE></MARQUEE>** crea una zona de la pantalla con un texto en su interior que se desplaza. No es un elemento muy aceptado por todos los navegadores, funcionando sólo en algunos como Internet Explorer. Sus parámetros son los siguientes:

align = top / middle / bottom Indica si el texto del interior de la marquesina se alinea en la zona alta (top), en la baja (bottom) o en el centro (middle).

bgcolor = "codigo de color" Indica el color del fondo de la marquesina.

direction = left / right Indica hacia qué lugar se desplaza el texto, hacia la izquierda (left) o hacia la derecha (right)

height = número o % Indica la altura de la marquesina en puntos o porcentaje en función de la ventana del navegador.

width = número o % Indica la anchura de la marquesina en puntos o porcentaje en función de la ventana del navegador.

loop = número / infinite Indica el número de veces que se desplazará el texto por la marquesina. Si se indica infinite, se desplazará indefinidamente.

scrolldelay = número. Indica el número de milisegundos que tarda en reescribirse el texto por la marquesina, a mayor número más lentamente se desplazará el texto.

Se puede ver un ejemplo de esta etiqueta en la siguiente página web que desplaza un texto en la **Vista en vivo** de Dreamweaver. Se ha utilizado sólo la siguiente etiqueta:

<**marquee bgcolor** = "#FFFFFF" **width** = 50% **scrolldelay** = 0 > Bienvenido a mi web en Internet.</**marquee**>

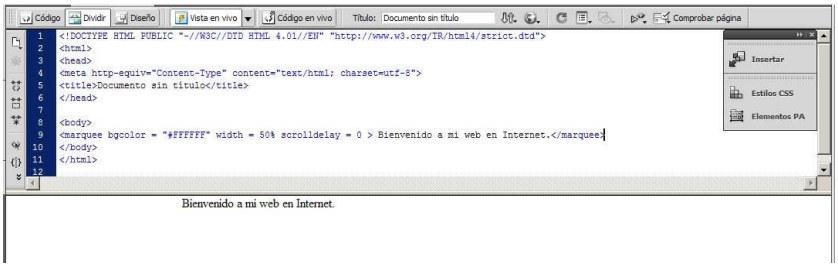


Figura 83. Marquesina en Dreamweaver

4

Formularios en la construcción de páginas web

- Características
- Elementos y atributos de formulario. Controles de formulario. Formularios y eventos
- Criterios de accesibilidad y usabilidad en el diseño de formularios
- Etiquetas y los atributos que se utilizan para definir los controles que forman los formularios en función de las interacciones a manejar
- Creación de formularios e integración en páginas web para incluir interactividad en las mismas

4.1 CARACTERÍSTICAS

El objetivo de un formulario es el de proporcionar un mecanismo para que el usuario interactué con la página web. La idea clave que se esconde bajo el uso de los formularios es recolectar información en la interacción con el usuario y luego ejecutar una determinada acción a partir de los datos recopilados. Quizá el ejemplo más típico del uso de un formulario podría ser los campos de búsqueda de Google. Se introducen los valores de búsqueda y "las páginas web" de Google realizan una acción: encontrar los resultados que coinciden con los criterios de búsqueda y mostrárselos al usuario por pantalla.

4.2 ELEMENTOS Y ATRIBUTOS DE FORMULARIO. CONTROLES DE FORMULARIO. FORMULARIOS Y EVENTOS

El lenguaje HTML proporciona dos etiquetas básicas para la definición de un formulario: **<form>** e **<input>**.

A continuación se puede apreciar un formulario en la **Vista en vivo** de Dreamweaver, que incorpora las anteriores etiquetas.

Figura 84. Formulario con etiquetas form e input

El código del anterior ejemplo es el siguiente:

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Etiquetas formularios</title>

</head>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Etiquetas formularios</title>

</head>

<body>

<h1> Ejemplo de formularios</h1>

        <form action="#" method="post">

                Escribe tu nombre

                <input name="nombre" type="text" value="">

                <input name="Enviar" type="submit">

        </form>

</body>

</html>
```

Código Figura 98. Formulario con etiquetas form e input

La sintaxis de la etiqueta form es la siguiente:

Acciones	Etiquetas y atributos
Definir formulario	<form action="url" method="POST|GET"> </form> url es dirección de los datos
Codificación	<form enctype="application/x-www-form-urlencoded | multipart/form-data"></object>
Formatos aceptados	<form accept="tipo_de_contenido"></formt>

Figura 85. Etiqueta form

Los atributos más habituales son action y method. El primero de ellos, **action** indica la dirección de la página web en el servidor que se encarga de procesar los datos del usuario. En nuestro ejemplo no hemos incluido ninguna acción, por lo que se ha rellenado este campo con el valor #. Lo más habitual es que el procesamiento de este formulario se realice por un lenguaje de más alto nivel que HTML, como PHP, Java, Perl, o ASP.

Una alternativa en este momento del manual es enviar los resultados del formulario por correo. Para lograr este objetivo se debe poner dentro de las comillas del atributo action el siguiente contenido:

action="mailto:direccion@correo.com?cc=otradireccion@correo.es&subject=Asunto%20del%20Correo&body=Hola,%20esto%20es%20una%20prueba. %0D%0AResponde%20cuando%20puedas."

Siendo:

Valor	Significado
mailto	Dirección de correo principal
cc	Dirección de correo con copia a
subject	El asunto del mensaje
body	Cuerpo de texto del mensaje

Figura 86. Parámetros mailto

Un problema común a la hora de incluir el texto del mensaje en un enlace es que, a primera vista, no hay quien ponga espacios ni varias líneas en el mismo. Esto se soluciona utilizando la codificación reservada a caracteres de control y extendidos. Para poner un espacio escribiremos %20 y para cambiar de línea %0D%0A.

El atributo **method** se encarga de enviar los datos del formulario. Sus valores son GET y POST, y no pertenecen al lenguaje HTML sino que son parte del protocolo de comunicaciones HTTP. A la hora de usar uno u otro método de envío hay que tener en cuenta los siguientes detalles:

- GET admite como máximo el envío de unos 500 bytes de información.
    - Como no podemos estar midiendo bytes de texto para ver si sobrepasamos el límite, lo que se suele hacer es considerar que el método GET no permite más de 256 caracteres. De hecho algunos navegadores web no admiten más caracteres, por lo que es la limitación más importante a tener en cuenta.

- GET no permite el envío de archivos adjuntos con el formulario.
- Los datos enviados mediante GET se ven en la barra de direcciones del navegador. Los datos enviados con POST no se muestran en la barra de direcciones, aunque con algunos conocimientos de HTTP es posible verlos también.

Al final, el uso de uno u otro método suele estar asociado al tipo de datos que se envíen y el uso que se pretende de ellos. Tradicionalmente, los datos de consulta se suelen enviar con GET, y los de inserción, modificación, borrado,… se envían con POST. Evidentemente si se adjunta un fichero también es preciso usar POST.

En cuanto al uso de los atributos **enctype y accept,** el primero es necesario a la hora de adjuntar ficheros, mientras que **accept** permite indicar el tipo de formatos de ficheros que se pueden adjuntar. Se introduce aquí sus extensiones separadas por comas.

Los elementos de formulario como botones y cuadros de texto también se denominan **campos del formulario o controles del formulario**. Estos elementos se indican con la etiqueta **<input>**. Esta etiqueta no tiene cierre, con lo cual si se usa en XHTML habrá que terminarla según la siguiente estructura: **<input … />**. Su sintaxis es la siguiente:

Acciones	Etiquetas y atributos
Definir control	<input type="text|checkbox|radio|submit| password|reset|hiden" name"nombre" value = "valor" alt= "texto"> Type es el tipo del control Name el nombre Valor es el valor inicial Texto es texto alternativo
Atributos específicos controles	<input size="tamaño"> <input maxlength = "número"> <input checked = "checked"> <input disabled = "disabled"> <input readonly = "readonly"> <input src = "url">

Figura 87. Etiqueta input

Los controles, según una definición no formal, son cada uno de los elementos que permiten al usuario interactuar con la página web. Estos controles están identificados por su atributo **name** o nombre, su valor inicial en el atributo **value**, y su tipo en el atributo **type**. Los atributos size, maxlengh, checked, disabled, redonly, src y hidden son atributos específicos del tipo de control con el que te encuentres, por que aparecen en algunos y no en otros. Se verán más adelante cuando se detalle cada uno de estos controles.

En función del tipo de información que utilizan para interactuar con el usuario, el lenguaje HTML y la etiqueta input proporciona **diez tipos distintos de controles.**

Cuadro de texto

Se trata del elemento más utilizado en los formularios. En el caso más sencillo, se muestra un cuadro de texto vacío en el que el usuario puede escribir cualquier texto:

Figura 88. Ejemplo de etiqueta input (type=text)

A continuación se muestra el código HTML correspondiente al ejemplo anterior:

**Nombre
**
<input type="text" name="nombre" value="">

El atributo **type** diferencia a cada uno de los diez controles que se pueden crear con la etiqueta **<input>**. Para los cuadros de texto, su valor es **text**. El atributo **name** es el más importante en los campos del formulario. De hecho, si un campo no incluye el atributo name, sus datos no se envían al servidor. El valor que se indica en el atributo name es el nombre que utiliza la aplicación del servidor para obtener el valor de este campo de formulario.

Cuando el usuario pulsa el botón de envío del formulario, el navegador envía los datos a una aplicación del servidor para que procese la información y genere una respuesta adecuada. En el servidor, la aplicación que procesa los datos debe obtener en primer lugar toda la información introducida por el usuario. Para ello, utiliza el valor del atributo name para obtener los datos de cada control del formulario.

Como el valor del atributo name se utiliza en aplicaciones programadas, es esencial ponerse de acuerdo con el programador de la aplicación, no se debe modificar su valor sin modificar la aplicación y no se deben utilizar caracteres problemáticos en programación (espacios en blanco, acentos y caracteres como ñ o ç).

El atributo **value** se emplea para establecer el valor inicial del cuadro de texto. Si se crea un formulario para insertar datos, los cuadros de texto deberían estar vacíos. Por lo tanto, o no se añade el atributo value o se incluye con un valor vacío **value=""**. Si por el contrario se crea un formulario para modificar datos, lo lógico es que se muestren inicialmente los datos guardados en el sistema. En este caso, el atributo value incluirá el valor que se desea mostrar:

<input type="text" name="nombre" value="José Lopez " />

Si no se especifica un tamaño, el navegador muestra el cuadro de texto con un tamaño predeterminado. El atributo **size** permite establecer el tamaño, en caracteres, con el que se muestra el cuadro de texto. Su uso es imprescindible en muchos formularios, en los que algunos campos como la dirección deben mostrar más caracteres de lo normal (<input size="100" ...) y otros campos como el código postal deben mostrar menos caracteres de lo normal (<input size="5"...).

Además de controlar el tamaño con el que se muestra un cuadro de texto, también se puede limitar el tamaño del texto introducido. El atributo **maxlength** permite establecer el máximo número de caracteres que el usuario puede introducir en un cuadro de texto. Su uso es imprescindible para campos como el código postal, el número de la Seguridad Social y cualquier otro dato con formato predefinido y limitado.

Por último, el atributo **readonly** permite que el usuario pueda ver los contenidos del cuadro de texto pero no pueda modificarlos y el atributo **disabled** deshabilita un cuadro de texto de forma que el usuario no pueda modificarlo y además, el navegador no envía sus datos al servidor.

Cuadro de contraseña

La única diferencia entre este control y el cuadro de texto normal es que el texto que el usuario escribe en un cuadro de contraseña no se ve en la pantalla. En su lugar, los navegadores ocultan el texto utilizando asteriscos o círculos, por lo que es ideal para escribir contraseñas y otros datos sensibles.

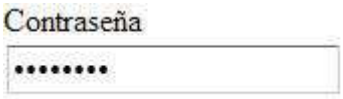


Figura 89. Ejemplo de etiqueta input (type=password)

Su código HTML es:

**Contraseña
**

<input type="password" name="contrasena" value="">

Cambiando el valor del atributo **type** por **password** se transforma el cuadro de texto normal en un cuadro de contraseña. Todos los demás atributos se utilizan de la misma forma y tienen el mismo significado.

Checkbox

Los checkbox o **casillas de verificación** son controles de formulario que permiten al usuario seleccionar y deseleccionar opciones individualmente. Aunque en ocasiones se muestran varios **checkbox** juntos, cada uno de ellos es completamente independiente del resto. Por este motivo, se utilizan cuando el usuario puede activar y desactivar varias opciones relacionadas pero no excluyentes.

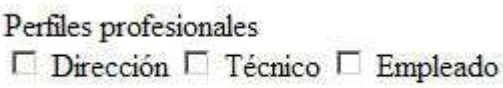


Figura 90. Ejemplo de etiqueta input (type=checkbox)

Su código es:

**Perfiles profesionales
**

<input name="puesto_directivo" type="checkbox" value="direccion"> Dirección

<input name="puesto_tecnico" type="checkbox" value="tecnico"> Técnico

<input name="puesto_empleado" type="checkbox" value="empleado"> Empleado

El valor del atributo **type** para estos controles de formulario es **checkbox**. Como se muestra en el ejemplo anterior, el texto que se encuentra al lado de cada checkbox no se puede establecer mediante ningún atributo, por lo que es necesario añadirlo manualmente fuera del control del formulario. Si no se añade un

texto al lado de la etiqueta <input> del checkbox, el usuario sólo ve un pequeño cuadrado sin ninguna información relativa a la finalidad de ese checkbox.

El valor del atributo **value**, junto con el valor del atributo **name**, es la información que llega al servidor cuando el usuario envía el formulario.

Si se quiere mostrar un checkbox seleccionado por defecto, se utiliza el atributo **checked**. Si el valor del atributo es **checked**, el checkbox se muestra seleccionado. En cualquier otro caso, el checkbox permanece sin seleccionar. En HTML no es necesario informar el valor de este atributo. Sin embargo en XHTML la situación es distinta. Aunque resulta redundante que el nombre y el valor del atributo sean idénticos, es obligatorio indicarlo de esta forma porque los atributos en XHTML no pueden tener valores vacíos

Radiobutton

Los controles de tipo **radiobutton** son similares a los controles de tipo checkbox, pero presentan una diferencia muy importante: son mutuamente excluyentes. Los radiobutton se utilizan cuando el usuario solamente puede escoger una opción entre las distintas opciones relacionadas que se le presentan. Cada vez que se selecciona una opción, automáticamente se deselecciona la otra opción que estaba seleccionaba.

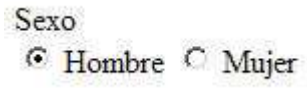


Figura 91. Ejemplo de etiqueta input (type=radio)

Su código es:

**Sexo
**

<input type="radio" name="sexo" value="hombre" checked="checked"> Hombre

<input type="radio" name="sexo" value="mujer"> Mujer

El valor del atributo **type** para estos controles de formulario es radio. El atributo **name** se emplea para indicar los radiobutton que están relacionados. Por lo tanto, cuando varios radiobutton tienen el mismo valor en su atributo name, el navegador sabe que están relacionados y puede deseleccionar una opción del grupo de radiobutton cuando se seleccione otra opción.

Botón de envío de formulario

La mayoría de formularios dispone de un botón para enviar al servidor los datos introducidos por el usuario:

Figura 92. Ejemplo de etiqueta input (type=submit)

Su código es:

<input type="submit" name="buscar" value="Buscar">

El valor del atributo **type** para este control de formulario es **submit**. El navegador se encarga de enviar automáticamente los datos cuando el usuario pincha sobre este tipo de botón. El valor del atributo **value** es

el texto que muestra el botón. Si no se establece el atributo value, el navegador muestra el texto predefinido **Enviar consulta**.

Botón de reseteo del formulario

Aunque su uso era muy popular hace unos años, la mayoría de formularios modernos ya no utilizan este tipo de botón. Se trata de un botón especial que borra todos los datos introducidos por el usuario y devuelve el formulario a su estado original:

Figura 93. Ejemplo de etiqueta input (type=reset)

Su código es:

<input type="reset" name="limpiar" value="">

El valor del atributo **type** para este control de formulario es **reset**. Cuando el usuario pulsa este botón, el navegador borra toda la información introducida y muestra el formulario en su estado original. Si el formulario no contenía originalmente ningún valor, el botón de reset lo vuelve a mostrar vacío. si el formulario contenía información, el botón reset vuelve a mostrar la misma información original.

Como es habitual en los botones de formulario, el atributo **value** permite establecer el texto que muestra el botón. Si no es utiliza este atributo, el navegador muestra el texto predefinido del botón, que en este caso es **Restablecer**.

Ficheros adjuntos

Los formularios también permiten adjuntar archivos para subirlos al servidor. Aunque desde el punto de vista de HTML y del navegador no existe ninguna limitación sobre el número, tipo o tamaño total de los archivos que se pueden adjuntar, todos los servidores añaden restricciones por motivos de seguridad.

Figura 94. Ejemplo de etiqueta input (type=file)

Su código es:

Fichero adjunto

<input type="file" name="adjunto">

El valor del atributo **type** para este control de formulario es **file**. El navegador se encarga de mostrar un cuadro de texto donde aparece el nombre del archivo seleccionado y un botón que permite navegar por los directorios y archivos del ordenador del usuario.

Si se incluye un control para adjuntar archivos, es obligatorio añadir el atributo **enctype** en la etiqueta <form> del formulario. El valor del atributo enctype debe ser **multipart/form-data**, por lo que la etiqueta <form> de los formularios que permiten adjuntar archivos siempre es:

<form action="..." method="post" enctype="multipart/form-data"> ...</form>

Campos ocultos

Los campos ocultos se emplean para añadir información oculta en el formulario. Estos campos no se ven en pantalla.

<input type="hidden" name="url_previa" value="/directorio/archivo.html" >

El valor del atributo **type** para este control de formulario es **hidden**. Los campos ocultos no se muestran por pantalla, de forma que el usuario desconoce que el formulario los incluye.

Normalmente los campos ocultos se utilizan para incluir información que necesita el servidor pero que no es necesario o no es posible que la establezca el usuario.

Botón de imagen

El aspecto de los botones de formulario se puede personalizar por completo, ya que incluso es posible utilizar una imagen como botón:

<input type="image" name="enviar" src="downloads.gif">

El valor del atributo **type** para este control de formulario es **image**. El atributo src indica la URL de la imagen que debe mostrar el navegador en lugar del botón normal.

Su principal ventaja es que permite personalizar por completo la estética de los botones y mostrarlos con un aspecto homogéneo en todos los navegadores. El principal inconveniente es que ralentiza la carga del formulario y que si se quiere modificar su aspecto, es necesario crear una nueva imagen.

Botón genérico

Algunos formularios complejos necesitan botones más avanzados que los de enviar datos (type="submit") y resetear el formulario (type="reset"). Por ese motivo, el estándar HTML define un botón de tipo genérico:

Figura 95. Ejemplo de etiqueta input (type=button)

Su código será:

<input type="button" name="guardar" value="Modificar Configuración">

El valor del atributo **type** para este control de formulario es **button**. Si pruebas a pulsar un botón de este tipo, verás que el navegador no hace nada: no envía los datos al servidor y no borra los datos introducidos. Este tipo de botones sólo son útiles si se utilizan junto con el lenguaje de programación JavaScript. Si la página incluye código JavaScript, los botones de este tipo se pueden programar para que realicen cualquier tarea compleja cuando se pulsa sobre ellos.

Formularios en Dreamweaver

La herramienta Dreamweaver dispone de diversos asistentes que facilitan el trabajo de crear los distintos elementos de un formulario. Para acceder a estas opciones, se debe recurrir a los paneles presentes en el menú **Insertar** disponibles en la parte superior derecha de la ventana principal. El proceso que se debe seguir es el siguiente:

1. Tras seleccionar dentro de la vista de código donde se desea generar el formulario, se pulsa sobre **Insertar**.

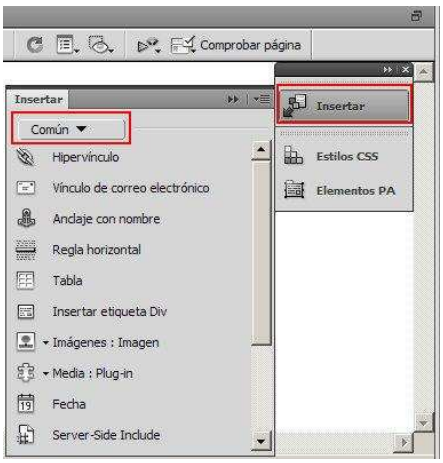


Figura 96. Insertar elementos de un formulario con Dreamweaver

2. El siguiente paso es elegir el tipo de objeto se quiere insertar. Por defecto se estará seleccionado **Común**.
3. Se despliegan los objetos y se elige Formularios.

Figura 97. Objetos de formularios

4. A partir de aquí será cuestión de elegir el tipo de controles de formulario que se desea incorporar para que se inicie el asistente correspondiente.

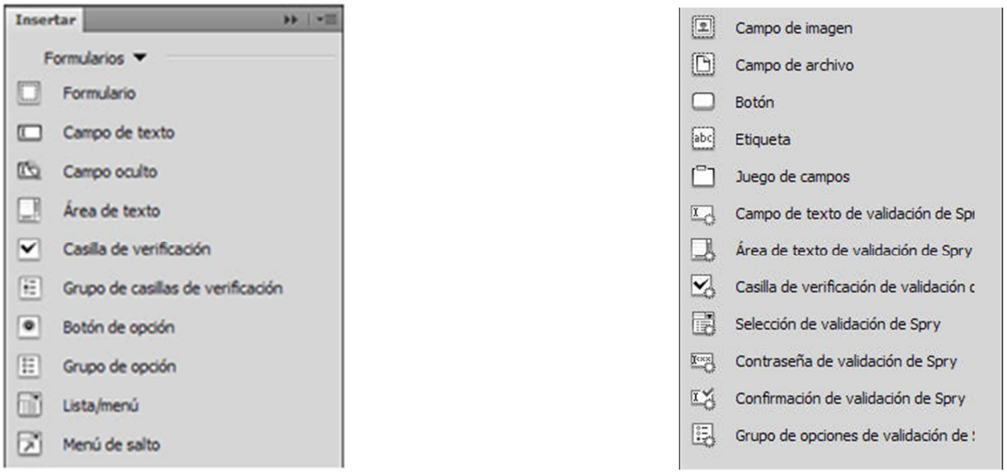


Figura 98. Asistente para controles de formularios

5. El primer elemento que se debe elegir, como es obvio es el propio formulario. Para ello se elige el objeto **Formulario** y se abrirá un asistente en el que habrá que rellenar los campos propios de esta etiqueta.

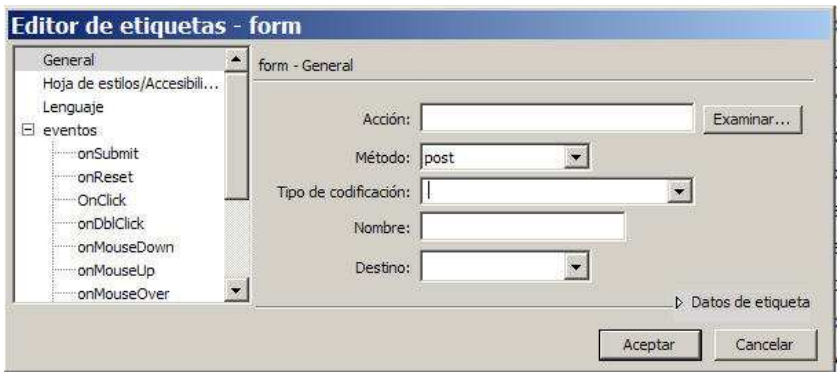


Figura 99. Asistente para etiqueta form

6. Con este proceso, en la vista de código de Dreamweaver, sólo se habrá completado en nuestro ejemplo de código la parte de **<form action="#" method="post"></form>**.

7. Evidentemente habrá que continuar con el resto de los elementos que se encuentran dentro de la etiqueta form. Por ejemplo si se elige un campo de texto, se pulsa sobre la opción del objeto **Campo de texto** y aparecerá de nuevo un asistente en el que rellenar los diversos atributos que componen a la etiqueta input cuando su tipo es texto.

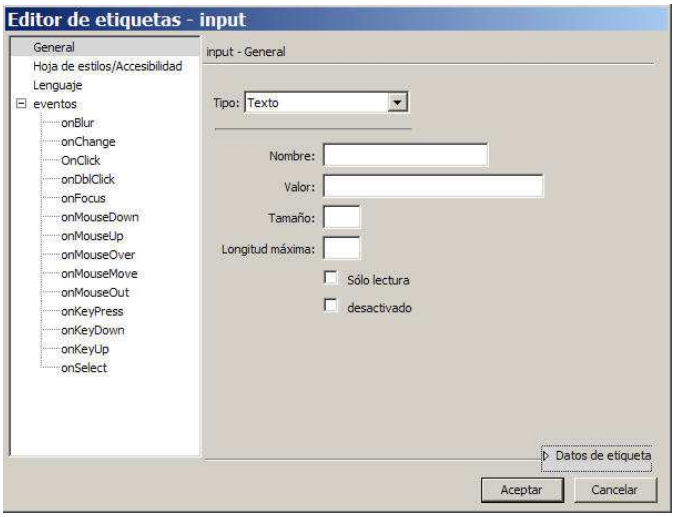


Figura 100. Asistente de campo de texto en Dreamweaver

8. El proceso de elegir distintos controles se repetiría con todos los objetos que se necesiten hasta completar el contenido de nuestro formulario.

Eventos

Los eventos están asociados fundamentalmente a javascript o vbscript. Es decir a la ejecución de un script que realiza una acción, codificada en alguno de los anteriores lenguajes, y que se ejecuta cuando se activa el

evento sobre el que se define. El conocimiento de javascript o vbscript no es el objetivo de este curso, por lo que no entraremos en detalle en su uso. Se recomienda que a continuación de este curso se aprenda javascript, pues es un aliado muy importante en el desarrollo de páginas web en HTML. Además, javascript es el núcleo de otro de los lenguajes de programación de moda, AJAX.

Con el fin de proporcionar más **interactividad a los formularios**, se explicará en el **punto 4.4 del curso** el funcionamiento de los eventos que "entiende" el lenguaje HTML gracias a la ayuda de javascript.

4.3 CRITERIOS DE ACCESIBILIDAD Y USABILIDAD EN EL DISEÑO DE FORMULARIOS

Los formularios junto con los enlaces son los elementos principales de interacción con el usuario en una página web. En el caso de los formularios, el contenido se tiene que poder leer y permitir que se responda, pues ahí radica la clave de su interacción. Por todo ello, la accesibilidad y usabilidad deben ser dos detalles fundamentales a tener en cuenta.

Agrupar campos y etiquetarlos

Utilizando solamente las etiquetas <form> y <input> es posible diseñar la mayoría de formularios de las aplicaciones web. No obstante, HTML define algunos elementos adicionales para mejorar la estructura de los formularios creados.

El lenguaje HTML proporciona la etiqueta **<fieldset>** para agrupar campos del formulario y la etiqueta **<legend>** asigna un nombre a cada grupo. La etiqueta <fieldset> agrupa todos los controles de formulario a los que encierra. El navegador muestra por defecto un borde resaltado para cada agrupación. La etiqueta <legend> se incluye dentro de cada etiqueta <fieldset> y establece el título que muestra el navegador para cada agrupación de elementos.

Se puede ver el efecto en la siguiente figura que muestra un ejemplo en Dreamweaver en su vista en vivo:

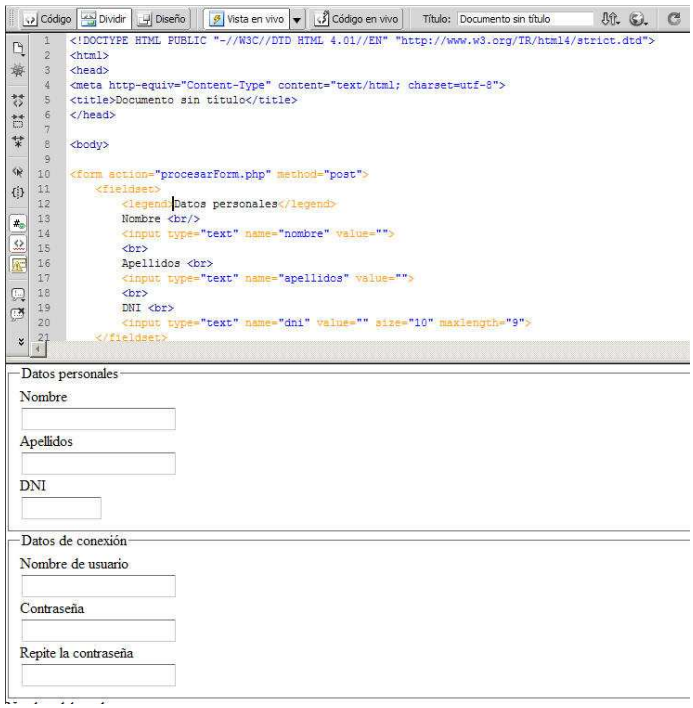


Figura 101. Formulario con campos agrupados

A continuación se puede encontrar el código correspondiente al ejemplo anterior.

La primera parte del código incluye las cabeceras de la página web junto con la definición del primer conjunto de campos de formulario agrupados, el nombre, apellido y DNI del usuario.

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title>Documento sin título</title>

</head>

<body>

<form action="procesarForm.php" method="post">

        <fieldset>

                <legend>Datos personales</legend>

                Nombre <br/>

                <input type="text" name="nombre" value="">

                <br>
```

```
                Apellidos <br>

                <input type="text" name="apellidos" value="">

                <br>

                DNI <br>

                <input type="text" name="dni" value="" size="10" maxlength="9">

        </fieldset>
```

Figura 102. Formulario agrupado. Parte 1

Ahora se repite el proceso para un segundo bloque de campos de nuestro formulario. En este caso se solicitará al usuario que introduzca sus datos de conexión como nombre de usuario y contraseña.

```
        <fieldset>

                <legend>Datos de conexión</legend>

                Nombre de usuario<br>

                <input type="text" name="nombre"
value="" maxlength="10">

                <br>
```

```
                Contraseña<br>

                <input type="password"
name="password" value="" maxlength="10">

                <br>

                Repite la contraseña<br/>

                <input type="password"
name="password2" value="" maxlength="10">

        </fieldset>
```

```
        <input type="submit" name="enviar"
value="Enviar">    

        <input type="reset" name="limpiar"
value="Borrar">    

</form></body></html>
```

Figura 103. Formulario agrupado. Parte 2

Finalmente, sólo queda introducir los botones de Enviar, Borrar y cerrar adecuadamente las distintas etiquetas que todavía estaban pendientes.

Figura 104. Formulario agrupado. Final

Cuando abrimos este código en un navegador se aprecia que los distintos campos del formulario se encuentran agrupados por un recuadro con un título que lo define. Este recuadro se ajusta al tamaño de la ventana del navegador con el que se haya abierto la página web.

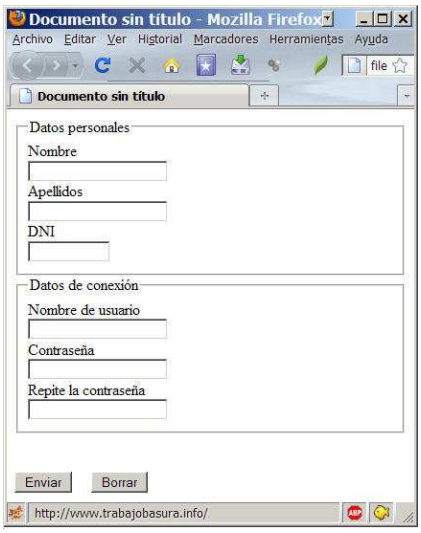


Figura 105. Formulario agrupado en Firefox

Etiquetas para títulos de formularios

A pesar de las facilidades que proporcionan fieldset y leyend, todos los controles de formulario salvo los botones presentan una carencia muy importante: no disponen de la opción de establecer el título o texto que se muestra junto al control. En el código HTML del ejemplo anterior, el nombre de cada campo se incluye en forma de texto normal, sin ninguna relación con el campo al que hace referencia.

Afortunadamente, el lenguaje HTML incluye una etiqueta denominada **<label>** y que se utiliza para establecer el título de cada campo del formulario. El anterior ejemplo definiría sus nombres de campos con la ayuda de <label> según la siguiente estructura:

**<label for="nombre">Nombre</label>
**

<input type="text" id="nombre" name="nombre" value="">

**<label for="apellidos">Apellidos</label>
**

<input type="text" id="apellidos" name="apellidos" value="">

**<label for="dni">DNI</label>
**

<input type="text" id="dni" name="dni" value="" size="10" maxlength="9">

Controles sin <input>

La etiqueta <input> permite crear diez tipos diferentes de controles de formulario. Sin embargo, algunas aplicaciones web utilizan otros elementos de formulario que no se pueden crear con <input>. Las listas desplegables y las áreas de texto disponen de sus propias etiquetas (<select> y <textarea> respectivamente).

Las **áreas de texto o textarea** son útiles cuando se debe introducir una gran cantidad de texto, ya que es mucho más cómodo de introducir que en un campo de texto normal.

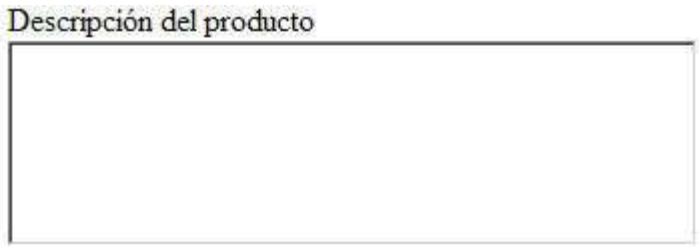


Figura 106. Etiqueta textarea

Su código es:

**<label for="nombre">Nombre del producto</label>
**

**<input type="text" id="nombre" name="nombre" value="">
**

**<label for="descripcion">Descripción del producto</label>
**

<textarea id="descripcion" name="descripcion" cols="40" rows="5"></textarea>

Se puede apreciar que dispone de dos atributos, **row** y **column** que se encargan de definir el número de filas y columnas de texto que posee. El inconveniente que presentan es que HTML no permite definir el tamaño en texto del control. Si se desea disponer de esta opción es preciso recurrir a lenguajes como javascript que "cuenten" el número de caracteres hasta alcanzar un límite deseado.

Las **listas desplegables o select** permiten mostrar uno o varios valores cada vez y seleccionar un valor o varios. De eso se encargan los atributos **size** que indica el nombre de la lista desplegable **y** multiple="multiple" que permite elegir varios valores.

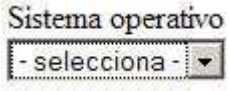


Figura 107. Ejemplo de etiqueta select

El código HTML del ejemplo anterior se muestra a continuación:

```
<label for="so">Sistema operativo</label> <br>
<select id="so" name="so">
<option value="" selected="selected">- selecciona -
</option>
<option value="windows">Windows</option>
<option value="mac">Mac</option>
<option value="linux">Linux</option>
<option value="otro">Otro</option>
</select>
```

Figura 108. Ejemplo de código con select

4.4 ETIQUETAS Y LOS ATRIBUTOS QUE SE UTILIZAN PARA DEFINIR LOS CONTROLES QUE FORMAN LOS FORMULARIOS EN FUNCIÓN DE LAS INTERACCIONES A MANEJAR

Los formularios en lenguaje HTML proporcionan una interactividad limitada. Básicamente se permite introducir información y enviar un mensaje de correo con la información recabada y muy poco más.

Si se desea mejorar esta interacción se debe recurrir a diversos scripts en otros lenguajes de programación (CGIs, perl, asp, php, javascript, etc). El conocimiento de estos lenguajes no es el objetivo de este curso, pero si daremos algunas pinceladas de su uso en casos sencillos.

La clave de estas interacciones más elaboradas radica en dos apartados:

1. En el atributo **accion** del formulario.
    a. El lector escribe la información rellenando campos o haciendo clic sobre botones, y luego presiona un botón de envío para mandar la información a una dirección (que puede ser una dirección de correo electrónico) o a la ejecución de un script dinámico.
    b. En el segundo caso, la ejecución de un script, este se encargará de procesar los datos suministrados por el formulario al enviar la información con los métodos POST o GET. La sintaxis sería similar a:
        i. <form action=”procesarFormulario.EXT. siendo EXT la extensión del fichero que identifica al procesador del formulario en función de su lenguaje de programación. Por ejemplo php, pl, cgi, asp…
2. Los **eventos** son otra fuente primordial de interactividad, al permitir ejecutar métodos o funciones de otros lenguajes previamente definidos dentro de las etiquetas **<script>**. Los lenguajes más utilizados dentro de estas etiquetas son javascript y VBscript.
    a. El uso de eventos es algo muy difundido a la hora de programar en lenguaje HTML. Además los métodos y funciones usados en tales casos no requieren conocimientos avanzados de estos lenguajes de script. Con el fin de mejorar algo la interactividad de los formularios, en este caso recurriremos a javascript.

Formularios y javascript

Según la Wikipedia, "***JavaScript*** *es un lenguaje de scripting basado en objetos, utilizado para acceder a objetos en aplicaciones. Principalmente, se utiliza integrado en un navegador web permitiendo el desarrollo de interfaces de usuario mejoradas y páginas web dinámicas*".

Una excelente definición pero que deja sin mencionar tres de las claves de javascript:

1. Es una evolución hacia el lenguaje Java con sus clases, métodos y objetos. Se queda a medio camino usando básicamente el concepto de objeto.

2. El lenguaje javacript es un lenguaje del lado del cliente. Es decir, los scripts generados se ejecutan por parte del cliente del ordenador del usuario. En nuestro caso por tanto todos los cálculos los realiza el navegador web. Esta situación contrasta con los lenguajes del lado del servidor, cuyos scripts JSP, PHP, ASP o CGI son procesados en un servidor remoto que envía los cálculos hechos para que el navegador cliente sólo tenga que pintarlos en pantalla.

3. La práctica totalidad de los navegadores actuales entienden javascript sin necesidad de añadidos. Esta situación se debe en buena parte a que javascript es multiplataforma, es decir funciona en varios sistemas operativos como Windows, Linux, UNIX, etc. Frente a esto, VBscript sólo está incorporado en Windows. La mayoría de los servidores web que alojan los sitios web de Internet están basados en UNIX o Linux. De ahí que javascript esté más difundido que VBscript.

Si se desea **acceder a multitud de scripts de ejemplo de javascript**, tutoriales, cursos, etc, una excelente fuente es Javascript Source en la dirección http://javascript.internet.com/.

Retomando el tema de los formularios y campos de formulario de una página, estos pueden ser accedidos desde un script JavaScript. El objeto **forms** está estructurado como una matriz que almacena un elemento por cada formulario que aparece en la página, en el orden en que estén en el código fuente. De este modo, en la definición del ejemplo siguiente:

<FORM METHOD=POST NAME="Formulario">

Se puede acceder a este objeto de cualquiera de estas formas:

document.forms[0] (primer formulario)
document.forms["Formulario"]
document.Formulario

Por el hecho de "entender" javascript, nuestro navegador ya realiza determinadas tareas al definir un formulario en HTML. El objeto formulario en javascript posee propiedades, métodos y eventos que podremos utilizar. Vamos a definir estos elementos y luego pasaremos a usarlos con algún ejemplo.

Propiedades del objeto formulario

Propiedad	Descripción
Action	Cadena que contiene el valor del atributo ACTION del tag FORM
Elements	Array que contiene una entrada por cada elemento en el formulario (como campos de texto, listas de selección, botones,...)
encoding	Cadena que contiene el tipo MIME utilizado para codificar los contenidos del formulario que se envían al servidor. Refleja el atributo ENCTYPE de la etiqueta FORM
Name	Cadena que contiene el valor del atributo NAME de la etiqueta FORM
Target	Cadena que contiene el nombre de la ventana objetivo del submit (ventana en la que se publican los resultados)
Length	Refleja el número de elementos del formulario
Method	Cadena que contiene el valor del atributo METHOD de la etiqueta FORM

Figura 109. Propiedades form

Métodos del objeto form

Método	Descripción
handleEvent	Invoca el manejador del evento especificado
Reset	Simula la pulsación del botón del ratón sobre un botón de reset del formulario
Submit	Envía el formulario

Figura 110. Métodos form

Eventos de formulario

Evento	Descripción
onReset	Captura la pulsación del botón reset
onSubmit	Captura la pulsación del botón submit

Figura 111. Eventos form

Acceso a los elementos de un formulario

Para acceder a los elementos de un formulario desde un script, se utiliza el *array* **elements**. Este *array* contiene una entrada para cada objeto del formulario (**button, checkbox, fileupload, hidden, password, radio, reset, select, submit, text, *y* textarea**), en el orden en que aparece el código fuente. Estas entradas se corresponden con los controles de HTML.

Cada elemento de un formulario posee propiedades y métodos que pueden utilizarse en los scripts para actuar sobre el formulario. A continuación vamos a describir los más importantes en las siguientes figuras:

Propiedad	Descripción
checked	Indica el estado actual del cuadro de selección
defaultChecked	Indica el estado por defecto del elemento
Name	Indica el nombre del elemento tal y como está especificado en la etiqueta INPUT
Value	Indica el valor el elemento tal y como está especificado en la etiqueta INPUT

Método	Descripción
click()	Simula la pulsación del botón del ratón sobre el cuadro de selección

Evento	Descripción
onClick	Captura la pulsación del botón del ratón sobre el cuadro de selección

Figura 112. Cuadro de selección (checkbox)

Propiedad	Descripción
Checked	Indica el estado actual del botón de selección
defaultChecked	Indica el estado por defecto del elemento
Name	Indica el nombre del elemento tal y como está especificado en la etiqueta INPUT
Value	Indica el valor del elemento tal y como está especificado en la etiqueta INPUT
Index	Indica el índice del botón de selección actualmente seleccionado en el grupo
Length	Indica el número de botones de selección en el grupo

Método	Descripción
click()	Simula la pulsación del botón del ratón sobre el botón de selección

Evento	Descripción
onClick	Captura la pulsación del botón del ratón sobre el botón de selección

Figura 113. Botón de selección (radio)

Es similar al cuadro de selección o **checkbox** con la diferencia de que varios elementos de este tipo pueden agruparse con un solo nombre de modo que sólo uno de ellos podrá estar seleccionado.

Con lo visto hasta ahora, se puede ver un ejemplo de interacción más avanzada realizado en Dreamweaver. Se trata de una calculadora sencilla. Permite introducir un número y calcular su cuadrado y su raíz cuadrada. La clave del formulario, evidentemente aparte del código javascript, radica en el evento **onChange**, que procesa los datos en cuanto se produce un cambio de su valor inicial y ejecuta Calcula(parámetros).

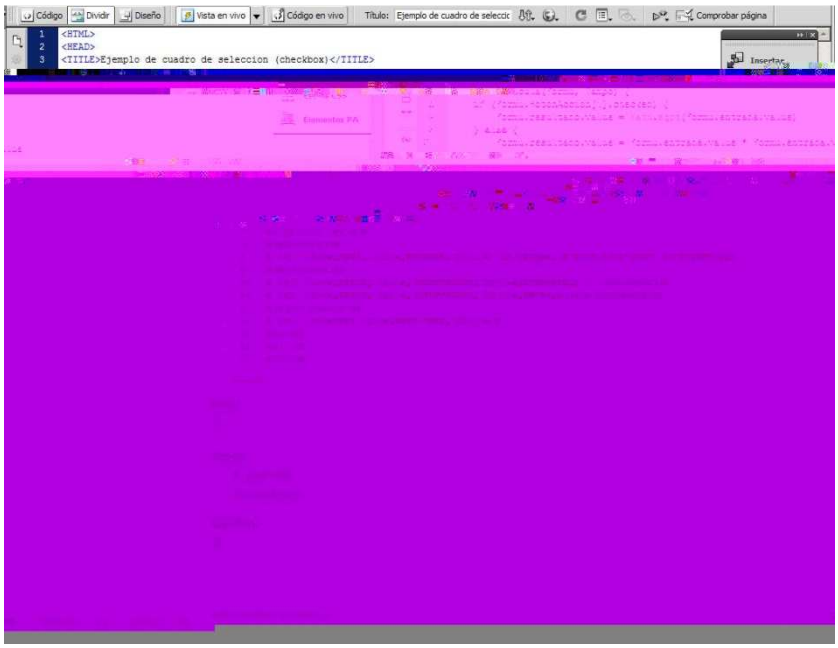


Figura 114. Calcular cuadrado y raíz cuadrada

El código HTML del ejemplo anterior se muestra a continuación. La primera parte se corresponde con el código javascript que se encarga de procesar las operaciones con los distintos valores introducidos.

```
<HTML>

<HEAD>

<TITLE>Ejemplo de cuadro de selección
(checkbox)</TITLE>

<SCRIPT>

<!--

function Calcula(Formu, Campo) {

   if (Formu.BotonAccion[1].checked) {

      Formu.resultado.value =
Math.sqrt(Formu.entrada.value)

   } else {

      Formu.resultado.value = Formu.entrada.value *
Formu.entrada.value

   }

}

//-->

</SCRIPT>

</HEAD>
```

```
<BODY>

<FORM METHOD=POST>

<P>Valor:<BR>

<INPUT TYPE="text" NAME="entrada" VALUE=0
onChange="Calcula(this.form, this.name);">

<P>Acción:<BR>

<INPUT TYPE="radio" NAME="BotonAccion"
VALUE="cuadrado"> Al cuadrado<BR>

<INPUT TYPE="radio" NAME="BotonAccion"
VALUE="raiz2"> Raiz cuadrada<BR>

<P>Resultado:<BR>

<INPUT TYPE=text NAME="resultado" VALUE=0>

</FORM>

</BODY>

</HTML>
```

Código Figura 128. Calcular cuadrado y raíz cuadrada

Propiedad	Descripción
defaultValue	Indica el valor por defecto del elemento tal y como está especificado en la etiqueta INPUT
name	Indica el nombre del elemento tal y como está especificado en la etiqueta INPUT
value	Indica el valor actual del elemento

Método	Descripción
focus()	Da el foco al elemento (convierte el cuadro de texto en la zona activa actual)
blur()	Quita el foco del elemento
select()	Selecciona el texto actual del cuadro de texto

Evento	Descripción
onChange	Captura el cambio de valor del campo de texto
onFocus	Captura la recuperación del foco por parte del cuadro de texto
onBlur	Captura la pérdida del foco
onSelect	Captura la selección del texto

Figura 115. Cuadro de texto (text) y Área de texto (textarea)

Propiedad	Descripción
options	Array que contiene una entrada por cada elemento de la lista de selección
selectedIndex	Da el índice de la opción de la lista seleccionada actualmente
options[n]. value	Indica el valor del elemento n de la lista, tal y como está especificado en la etiqueta OPTION
options[n]. text	Contiene la cadena de texto que representa la opción n de la lista
options[n]. selected	Indica si la opción n está actualmente seleccionada
options[n]. defaultSelected	Indica si la opción n es la seleccionada por defecto

Método	Descripción
focus()	Da el foco al elemento
blur()	Quita el foco del elemento

Evento	Descripción
onChange	Captura el cambio de selección en la lista
onFocus	Captura la recuperación del foco por parte de la lista de selección
onBlur	Captura la pérdida del foco

Figura 116. Lista de selección (select)

Como ejemplo, dado el siguiente código HTML:

```
1 <SELECT NAME="Ejemplo" onFocus="Ver()">
2 <OPTION SELECTED VALUE="Opción 1">1
3 <OPTION VALUE="Opción 2">2
4 <OPTION VALUE="Opción 3">3
5 </SELECT>
```

Figura 117. Control select con evento

Se podrá acceder a las propiedades de la lista de selección, tomando los valores que aparecen a la derecha:

Ejemplo.options[1].value = "Opción 2"
Ejemplo.options[2].text = "3"
Ejemplo.selectedIndex = 0
Ejemplo.options[0].defaultSelected = true
Ejemplo.options[1].selected = false

Botones submit y reset

Los botones **submit** y **reset** poseen las propiedades **name** y **value**, pudiendo utilizarse con ellos el método **click()** y el evento **onClick()**.

El botón submit posee un método y un evento específicos: el método **submit()** y el evento **onSubmit**. El método **submit()** realiza el emisión del formulario (es equivalente a pulsar el botón **submit**). El evento **onSubmit** se produce cuando se envía el formulario, permitiendo realizar operaciones con los datos del formulario antes de realizar el envío (por ejemplo, validación de campos).

Poner todo en marcha

Con todo lo visto, se puede ampliar algo más las funcionalidades de nuestra calculadora. En este ejemplo es muy importante una de las características que nos permiten en general los objetos y en nuestro caso el objeto formulario: hacer referencia a dicho formulario a partir de un elemento del mismo.

Por ejemplo:

var Formu = Digito.form, permite obtener el formulario que contiene el botón Digito.

Antes de ver el ejemplo en Dreamweaver, un último detalle. Con el fin de reducir el número de botones de la calculadora del ejemplo, se ha "eliminado" el botón = o **resultado**. Para obtener igual efecto, se ha integrado la obtención del resultado tras la pulsación de cualquier tecla de operaciones.

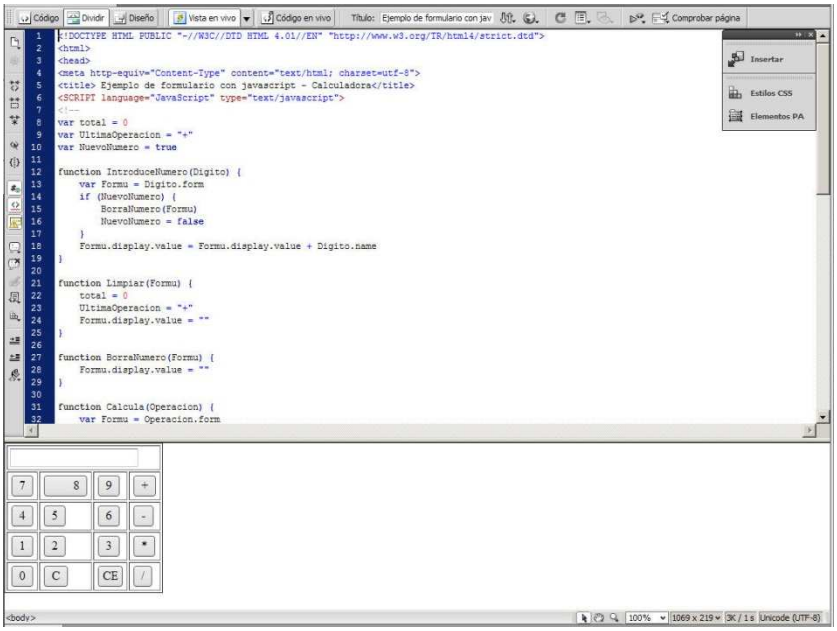


Figura 118. Calculadora con HTML y javascript

El código HTML y javascript correspondiente al anterior ejemplo se puede apreciar en las siguientes capturas.

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

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<title> Ejemplo de formulario con javascript - Calculadora</title>

<SCRIPT language="JavaScript" type="text/javascript">

<!--

var total = 0

var UltimaOperacion = "+"

var NuevoNumero = true

function IntroduceNumero(Digito) {

   var Formu = Digito.form
```

```
    if (NuevoNumero) {
        BorraNumero(Formu)
        NuevoNumero = false
    }
    Formu.display.value = Formu.display.value +
Digito.name
}
function Limpiar(Formu) {
    total = 0
    UltimaOperacion = "+"
    Formu.display.value = ""
}
function BorraNumero(Formu) {
    Formu.display.value = ""
}
```

```
 function Calcula(Operacion) {

    var Formu = Operacion.form

    var Expresion = total + UltimaOperacion +
Formu.display.value

    UltimaOperacion = Operacion.value

    total = eval(Expresion)

    Formu.display.value = total

    NuevoNumero = true

}

//-->

</SCRIPT>

</head>
```

Figura 119. Formulario con HTML y javascript. Parte 1

Una vez se ha finalizado con la parte javascript, llega el momento de incorporar la parte HTML de nuestra calculadora. La clave de este bloque de código radica en que cuando se pulsa sobre el botón correspondiente, se envía el valor pulsado a la función javascript que procesa los datos.

```
<body>

<FORM>
        <TABLE BORDER=1>
                <TR>
        <TD COLSPAN=4><INPUT TYPE=text
NAME=display VALUE="" onFocus="this.blur();"></TD>
    </TR>
                <TR>
                        <TD><INPUT TYPE=button
NAME="7" VALUE=" 7 "
onClick="IntroduceNumero(this);"></TD> <TD><INPUT
TYPE=button NAME="8" VALUE="        8 "
onClick="IntroduceNumero(this);"></TD>
                        <TD><INPUT TYPE=button
NAME="9" VALUE=" 9 "
onClick="IntroduceNumero(this);"></TD>
                        <TD><INPUT TYPE=button
NAME="+" VALUE=" + "
onClick="Calcula(this);"></TD>
    </TR>
    <TR>
                        <TD><INPUT TYPE=button
NAME="4" VALUE=" 4 "
onClick="IntroduceNumero(this);"></TD>
                        <TD><INPUT TYPE=button
NAME="5" VALUE=" 5 "
onClick="IntroduceNumero(this);"></TD>
                        <TD><INPUT TYPE=button
NAME="6" VALUE=" 6 "
onClick="IntroduceNumero(this);"></TD>
                        <TD><INPUT TYPE=button
NAME="-" VALUE=" - " onClick="Calcula(this);"></TD>
                </TR>
```

```
                    <TR>
                            <TD><INPUT TYPE=button
NAME="1" VALUE=" 1 "
onClick="IntroduceNumero(this);"></TD>
                            <TD><INPUT TYPE=button
NAME="2" VALUE=" 2 "
onClick="IntroduceNumero(this);"></TD>
                            <TD><INPUT TYPE=button
NAME="3" VALUE=" 3 "
onClick="IntroduceNumero(this);"></TD>
                            <TD><INPUT TYPE=button
NAME="*" VALUE=" * "
onClick="Calcula(this);"></TD></TR>
      <TR>
          <TD><INPUT TYPE=button NAME="0"
VALUE=" 0 " onClick="IntroduceNumero(this);"></TD>
                            <TD><INPUT TYPE=button
NAME="C" VALUE=" C "
onClick="Limpiar(this.form);"></TD>
                            <TD><INPUT TYPE=button
NAME="CE" VALUE="CE"
onClick="BorraNumero(this.form);"></TD>
                            <TD><INPUT TYPE=button
NAME="/" VALUE=" / " onClick="Calcula(this);"></TD>
                    </TR>
          </TABLE>
</FORM>

</body>
</html>
```

Figura 120. Formulario con HTML y javascript. Parte 2

5.5 CREACIÓN DE FORMULARIOS E INTEGRACIÓN EN PÁGINAS WEB PARA INCLUIR INTERACTIVIDAD EN LAS MISMAS

La forma más habitual de interactuar con un formulario utilizando sólo código HTML es enviar la información introducida por correo. El formato de la etiqueta form sería el siguiente:

<form action=”mailto:nombre@dominio.com ?subject=asunto”
method=”post” enctype="text/plain">

Si se recurre sólo a HTML, la interactividad que se puede lograr es muy reducida. Con la ayuda de javascript y HTML, se pueden hacer que varios formularios o controles interactúen entre sí, pasándose datos o modificando su comportamiento en función de cómo se hayan rellenado previamente. Se puede apreciar la apariencia gráfica de ambos formularios a través de la vista en vivo de Dreamweaver.

Figura 121. Control que pasa valores a otro control

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title> Pasar valores de un control a otro</title>
<script type="text/javascript">
function favBrowser()
{
var mylist=document.getElementById("myList");
document.getElementById("favorito").value=mylist.options
[mylist.selectedIndex].text;
}
</script>
</head>

<body>
<form>
Elija su navegador favorito:
<select id="myList" onchange="favBrowser()">
  <option>Internet Explorer</option>
  <option>Firefox</option>
  <option>Opera</option>
</select>
<p>Su navegador favorito es:<input type="text"
id="favorito" size="20"></p>
</form>
</body>

</html>
```

Código Figura 134. Control que pasa valores a otro control

El comportamiento de este código provoca que cuando se selecciona por ejemplo el valor Firefox en el primer select, se rellena este valor en el campo del input. Como puede apreciarse, la clave se encuentra en que al cambiar el valor del primer formulario (evento onChange), se ejecuta el código javascript favBrowser() que introduce el valor cambiado a la variable favorito, que es el valor del atributo id del input.

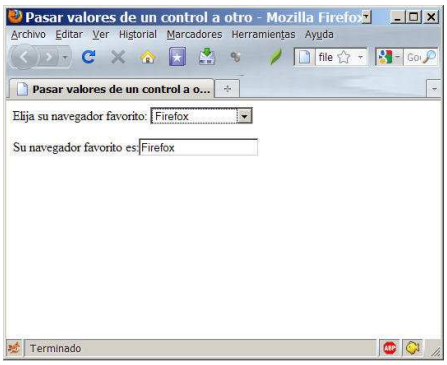


Figura 122. Ejecutar pasar valores entre controles

Esta misma estructura puede ser utilizada para cualquiera de los otros controles del campo input (checkbox, radiobutton,…), con lo que se logra que uno o varios controles interactúen. Ejemplos típicos de este uso son las listas desplegable con países y en función de los elegidos mostrar sus ciudades, o calendarios con los meses y sus días, productos y sus precios, etc.

5

Plantillas en la construcción de páginas web

- Funciones y características
- Campos editables y no editables
- Aplicar plantillas a páginas web

5.1 FUNCIONES Y CARACTERÍSTICAS

A la hora de crear una nueva página web, lo habitual es diseñar todas las páginas con un formato similar. Es decir, tener las mismas cabeceras, los mismos pies, los mismos colores de texto, columnas, etc. Durante el proceso de creación lo más normal es reutilizar copias de los documentos ya creados, y trabajar sobre estas copias, modificando simplemente su contenido. Esta es la forma más sencilla de tener páginas con una estructura basada en la estructura de otras ya creadas previamente.

Dreamweaver soluciona este problema mediante el uso de lo que denomina plantillas.

5.2 CAMPOS EDITABLES Y NO EDITABLES

Las plantillas son una especie de copia de la página en la que van a estar basadas el resto de páginas del sitio web, pero que incluyen la posibilidad de establecer unas zonas editables y otras zonas que serán fijas, que no podrán ser modificadas. No es posible modificar las propiedades de una página que está basada en una plantilla, a excepción del título. Cuando se desea que existan páginas con, por ejemplo, diferente color de fondo, es necesario crear plantillas diferentes con los distintos colores, y basar las páginas en una u otra plantilla, según el color de fondo que se desee para cada una.

Cuando se modifica el diseño de una plantilla, se pueden actualizan todas las páginas basadas en ella. Las plantillas son archivos con la extensión DWT que se guardan en el sitio web, dentro de una carpeta llamada Templates.

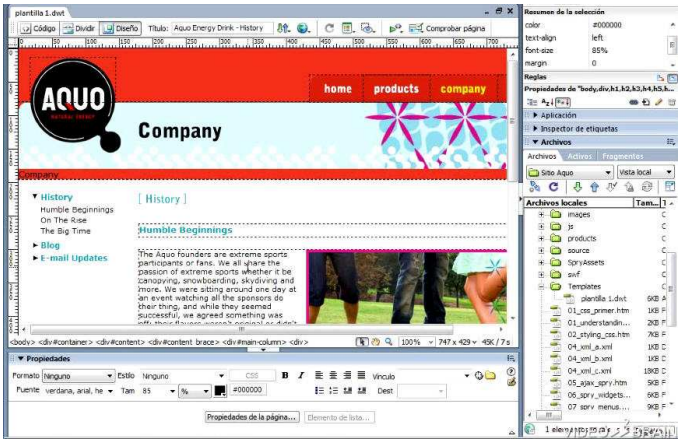


Figura 123. Ejemplo de plantilla

Una plantilla contiene cuatro tipos de regiones:

- **Una región editable:** Una región no bloqueada de un documento basado en plantilla, es decir, una sección que el usuario de la plantilla puede editar. El autor de una plantilla puede especificar cualquier área de la plantilla como editable. Para que una plantilla sea efectiva, deberá contener al menos una región editable. En caso contrario, las páginas basadas en la plantilla no se podrán editar.

- **Una región repetida**: Una sección del diseño del documento que se establece de forma que el usuario de la plantilla pueda añadir o eliminar copias de la región en un documento basado en la plantilla según resulte oportuno. Por ejemplo, puede definir que una fila de una tabla se repita. Las secciones repetidas son editables para que el usuario de la plantilla pueda editar el contenido del elemento repetido, mientras que el diseño propiamente dicho está controlado por el autor de la plantilla.

    - Existen dos tipos de regiones repetidas que se pueden insertar en una plantilla: región repetida y tabla repetida.

- **Una región opcional**: Una sección de la plantilla en la que hay contenido (como texto o una imagen) que puede aparecer o no en un documento. En la página basada en la plantilla, el usuario de la plantilla suele controlar si el contenido se mostrará.

- **Un atributo de etiqueta editable:** Permite desbloquear un atributo de etiqueta de una plantilla de modo que el atributo pueda editarse en una página basada en plantilla. Por ejemplo, puede "bloquear" qué imagen aparece en el documento, pero dejar que el usuario de la plantilla establezca alineación izquierda, derecha o central.

Crear plantillas

Las plantillas pueden crearse desde cero, o a partir de una página ya existente. Si se desea **crear una plantilla desde cero** la mejor forma es a través del **panel Activos**.

El panel **Activos** se puede abrir a través del menú **Ventana**, opción **Activos**. Una vez abierto el panel hay que seleccionar el botón, para poder trabajar con las plantillas. Los botones disponibles permiten actualizar la lista, crear una nueva plantilla, editar una plantilla seleccionada en la lista, o eliminarla. Podemos ver estos elementos en la siguiente figura:

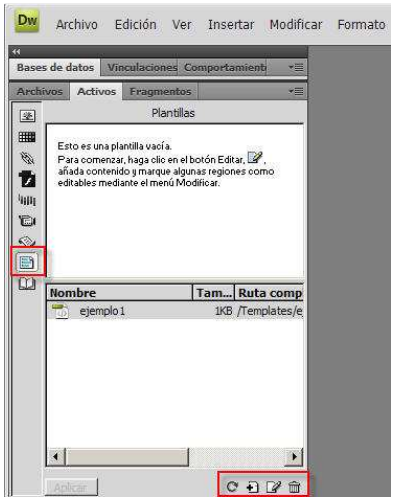


Figura 124. Opciones para plantillas

Para crear una nueva plantilla hay que pulsar sobre el botón que tiene un + (si este botón no está activado, pulsa con el botón derecho del ratón y elige **Nueva Plantilla**). Cuando se pulsa ese botón aparece un nuevo documento en la lista de plantillas, al que es posible cambiarle el Nombre pulsando previamente sobre él. Las plantillas se guardan en el sitio web actual, dentro de la carpeta Templates, que se crea automáticamente.

Para **crear una plantilla a partir de un archivo existente** es necesario abrir dicho archivo, y después guardarlo como plantilla a través del menú **Archivo**, opción **Guardar como plantilla**.

Cuando se pulsa dicha opción, aparece una ventana como la de la figura. En ella es necesario especificar el nombre con el que va a ser guardada la plantilla, a través del campo Guardar como, y elegir, de entre la lista de sitios, el Sitio en el que se va a guardar.

Establecer regiones editables en una plantilla

Todos los elementos de una plantilla están bloqueados por defecto. Es necesario establecer las zonas que sí podrán ser editadas en las páginas que se basen en dicha plantilla. Para **editar una plantilla** hay que abrirla en Dreamweaver. Una forma de abrirla es a través del panel Activos, pulsando dos veces sobre ella, o estando seleccionada pulsando sobre el botón que tiene un lápiz.

Una vez abierta la plantilla es posible **establecer sus propiedades** a través de Propiedades de la página. Como recordarás, para abrir esta ventana puedes:

- Pulsar la combinación de teclas **Ctrl+J o**
- Hacer clic sobre el menú **Modificar** y elegir la opción **Propiedades de la página**. Pulsar con el botón derecho sobre el documento abierto en Dreamweaver, y en el menú contextual que aparece seleccionar la última opción, que es Propiedades de la página.

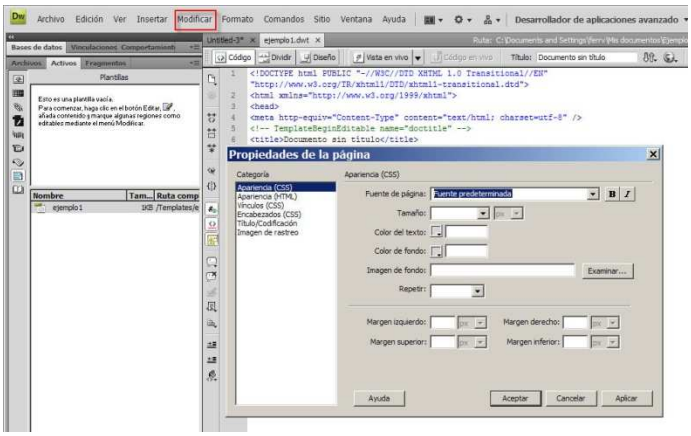


Figura 125. Propiedades de la página

Para **insertar una región editable** hay que situar el puntero en el lugar en el que se desea insertar, seleccionar la región o texto y después dirigirse al menú **Insertar**, **Objetos de plantilla**, opción **Región editable**, o pulsar la combinación de teclas **Ctrl+Alt+V**.

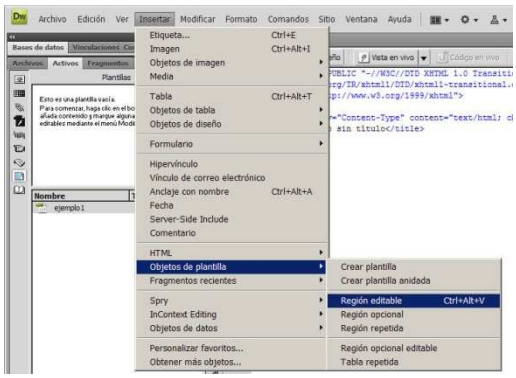


Figura 126. Insertar región editable

Si el actual documento HTML no es una plantilla, lo convertirá en plantilla y generará la región. Lo primero de todo, evidentemente es asignar un nombre.

Figura 127. Nueva región editable

Dentro de la plantilla no pueden existir dos regiones editables con el mismo nombre. Posteriormente puede modificarse el nombre a través del inspector de propiedades de la región editable.

Las zonas editables aparecen representadas en Dreamweaver, en zona de código englobados por comentarios que aluden a esta situación y en zona de diseño como un recuadro con una etiqueta de color turquesa. En la etiqueta aparece el nombre de la zona editable. Dentro del recuadro es posible insertar objetos, que aparecerán por defecto en aquellos documentos que se basen en la plantilla. Estos objetos, al estar dentro de la zona editable, podrán ser modificables en las páginas.

Todos los objetos que se encuentren fuera de estas zonas editables aparecerán también en las páginas, pero no podrán ser modificados. En este caso, si se usa una imagen con un logotipo aparecería en todas las páginas que se basaran en esta plantilla, mientras que todo lo que insertáramos dentro de la zona editable EditRegion1 de nuestro ejemplo podría ser modificado.

5.3 APLICAR PLANTILLAS A PÁGINAS WEB

Basta con buscar en Google los términos plantilla dreamweaver o plantillas web para encontrar multitud de ejemplos de plantillas listas para su uso. En la mayoría de los casos sólo hay que personalizar algunos detalles como título de la página y secciones para disponer de una página web completa.

Al aplicar una plantilla a un documento existente, la plantilla sustituye el contenido del documento por el contenido preestablecido de la plantilla. Por tanto es muy aconsejable realizar siempre una copia de seguridad del contenido de la página antes de aplicarle una plantilla.

Tanto si hemos creado una plantilla como la hemos descargado y se desea usarla en nuestro diseño de páginas web, el proceso que se sigue para poder usarla se puede realizar desde la ventana de Activos o desde el propio documento. En el **caso de Activos**:

1. Abra el documento en el que desea aplicar la plantilla.
2. En el panel **Activos** (Ventana y Activos), seleccione la categoría Plantillas situada en la parte izquierda del panel. Aquí caben dos opciones:
    a. Arrastre la plantilla que desea aplicar desde el panel Activos a la ventana de documento.
    b. Seleccione la plantilla que desea aplicar y haga clic en el botón **Aplicar** del panel **Activos**.
3. Si hay contenido en el documento que no se puede asignar automáticamente a una región de plantilla, aparecerá el cuadro de diálogo **Nombres de regiones no coincidentes**.
4. Seleccione un destino para el contenido empleando el menú **Mover contenido a la nueva región**; siga uno de estos procedimientos:
    a. Seleccione una región de la nueva plantilla donde mover el contenido existente.
    b. Seleccione **Ningún lugar** para eliminar el contenido del documento.
5. Para mover todo el contenido no resuelto a la región seleccionada, haga clic en **Utilizar para todo**.
6. Haga clic en **Aceptar** para aplicar la plantilla o haga clic en **Cancelar** para cancelar la aplicación de la plantilla al documento.

En el caso de que queramos **aplicar una plantilla a un documento existente,** el proceso es mucho más sencillo.

1. Tras abrir el documento en el que desea aplicar la plantilla, se selecciona en el menú principal **Modificar**, **Plantillas** y **Aplicar plantilla a página**.
2. Tras abrir el documento en el que desea aplicar la plantilla, se selecciona en el menú superior**, Plantillas** y **Aplicar plantilla a página**.
3. Aparecerá el cuadro de diálogo Seleccionar plantilla.
4. Elija una plantilla de la lista y haga clic en **Seleccionar**. Si hay contenido en el documento que no se puede asignar automáticamente se repiten los pasos dados en el caso de Activos.

Si se desean **deshacer los cambios realizados en la plantilla**, bastará con acceder a las opciones de **Edición** y **Deshacer aplicar plantilla**. De esta forma el documento vuelve al estado previo a que se aplicara la plantilla.

6

Hojas de estilo en la construcción de páginas web

- Funciones y características
- Selectores y reglas de estilo
- Tipos de estilos
- Atributos de estilo para fuentes, color y fondo, texto y bloques (párrafos). Creación de ficheros de estilo
- Hojas de estilo y accesibilidad

6.1 FUNCIONES Y CARACTERÍSTICAS.

El lenguaje de programación denominado CSS (Cascade Style Sheet) o hoja de estilo en cascada tiene como cometido maquetar los contenidos de las páginas web, dotándolos de estructura y estilos visuales.

Hojas de Estilo en Cascada (Cascading Style Sheets), es un mecanismo simple que describe cómo se va a mostrar un documento en la pantalla, o cómo se va a imprimir, o incluso cómo va a ser pronunciada la información presente en ese documento a través de un dispositivo de lectura. Esta forma de descripción de estilos ofrece a los desarrolladores el control total sobre estilo y formato de sus documentos.

CSS se utiliza para dar estilo a documentos HTML y XML, separando el contenido de la presentación. Los estilos definen la forma de mostrar los elementos HTML y XML. CSS permite a los desarrolladores Web controlar el estilo y el formato de múltiples páginas Web al mismo tiempo. Cualquier cambio en el estilo marcado para un elemento en la CSS afectará a todas las páginas vinculadas a esa CSS en las que aparezca ese elemento.

Las hojas de estilo son el complemento perfecto para el lenguaje HTML encontrándose asociado de tal manera que hoy en día, uno sin el otro no tiene sentido. El lenguaje HTML posee pocos elementos para formatear visualmente contenidos en pantalla. CSS se inventó para remediar esta situación, proporcionando a los diseñadores web con sofisticadas oportunidades de presentación soportadas por todos los navegadores. Al mismo tiempo, la separación de la presentación de los documentos del contenido de los mismos, hace que el mantenimiento del sitio sea mucho más fácil. Esta asociación de lenguajes permite que el lenguaje HTML sea el que genere los contenidos y CSS la presentación.

Con la integración de ambos lenguajes, HTML y CSS, y mediante una sintaxis especial se podrán definir detalles como los tipos de fuentes que se usaran, colores del texto, número de columnas, si se organizan como cabecera, pie o cuerpo central de texto y un larguísimo etcétera. En la siguiente figura, se puede ver un ejemplo de código CSS generado por uno de los asistentes de Dreamweaver.

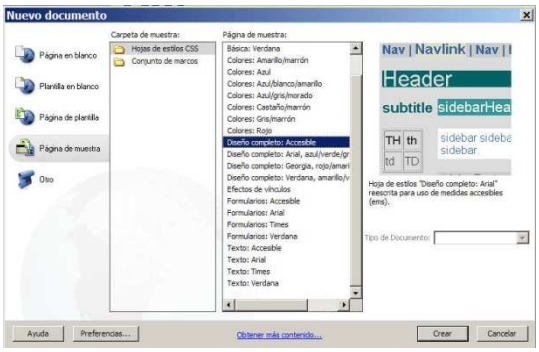


Figura 128. Asistente CSS

Tras pulsar sobre **Crear** se generará un código como el que aparece en la figura siguiente:

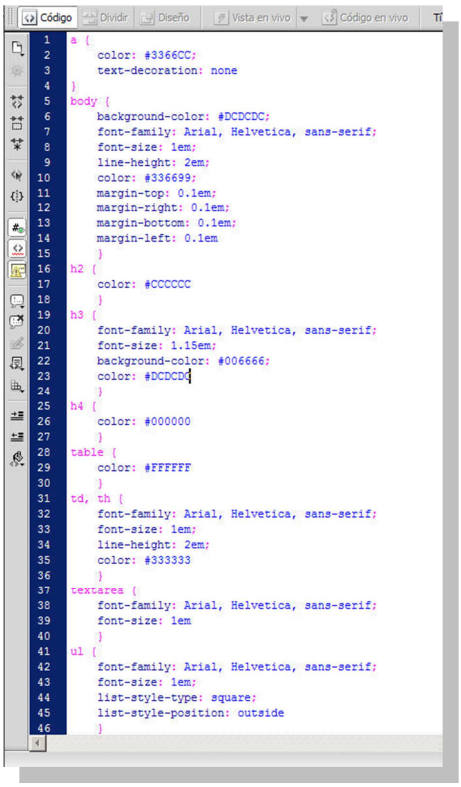


Figura 129. Ejemplo de código generado por Dreamweaver

Como puede apreciarse, igual que ocurre con HTML y sus etiquetas, la denominación de cada regla está basada en su nombre en inglés. Por tanto conocer algo del idioma de Shakespeare ayudará mucho a la hora de saber cuál es la regla CSS para cualquier necesidad. Por ejemplo, las correspondencias de los reglas para color, borde, estilo de fuente, imagen de fondo, o ancho del borde superior, se corresponden con color, border, font-style, background-image, border-top-width respectivamente. Se puede apreciar que es prácticamente una traducción literal del español al inglés separado por el carácter -.

Se puede consultar el conjunto de reglas CSS actualmente disponibles en la **Guía de referencia rápida de CSS 2.1** en la siguiente dirección: **http://www.w3c.es/Divulgacion/GuiasReferencia/CSS21/**

6.2 SELECTORES Y REGLAS DE ESTILO

CSS funciona a base de reglas, es decir, declaraciones sobre el estilo de uno o más elementos. Las hojas de estilo están compuestas por una o más de esas reglas aplicadas a un documento HTML o XML. La regla tiene dos partes: un selector y la declaración. A su vez la declaración está compuesta por una propiedad y el valor que se le asigne.

Digamos que queremos un color rojo como fondo de nuestra página web:

Usando HTML este objetivo se puede conseguir así:

<body bgcolor="#FF0000">

Con CSS el mismo resultado puede lograrse así:

body {background-color: #FF0000;}

La sintaxis es muy simple, pues consiste en escribir el nombre de la etiqueta html que queremos aplicarle un estilo o selector seguido de una llave "{" y luego la definición con los estilos que queremos aplicar. Una vez que tenemos todo el código listo, cerramos la llave con "}". Es decir la estructura es la siguiente:

Selector {propiedad: valor}

Si por ejemplo queremos cambiar el tamaño de fuente de toda la página web, le aplicaríamos el estilo a la etiqueta **<body>** y quedaría de esta forma:

body{

font-size:16px;

}

Cuando queremos aplicar estilos a un bloque que creamos nosotros, antes del nombre tenemos que insertar un punto. Por ejemplo, se ha creado un bloque (capa) llamado cabecera donde queremos asignarle un tamaño de 780px de ancho. Esta tarea se haría según el siguiente ejemplo:

.cabecera{ width:780px;}

Es necesario matizar que al final de cada atributo se debe poner siempre un punto y coma. Salvo que este sea único o sea el último de la lista.

Los selectores pueden aparecer individualmente o agrupados, separándolos con comas. En el siguiente ejemplo, h1{color: red;}, el selector, <h1>, le dice al navegador la parte del documento que se verá afectada por esa regla. Si se desea aplicar el mismo efecto se podría hacer según:

h1, h2, h3 {color: red;}

O lo que es lo mismo:

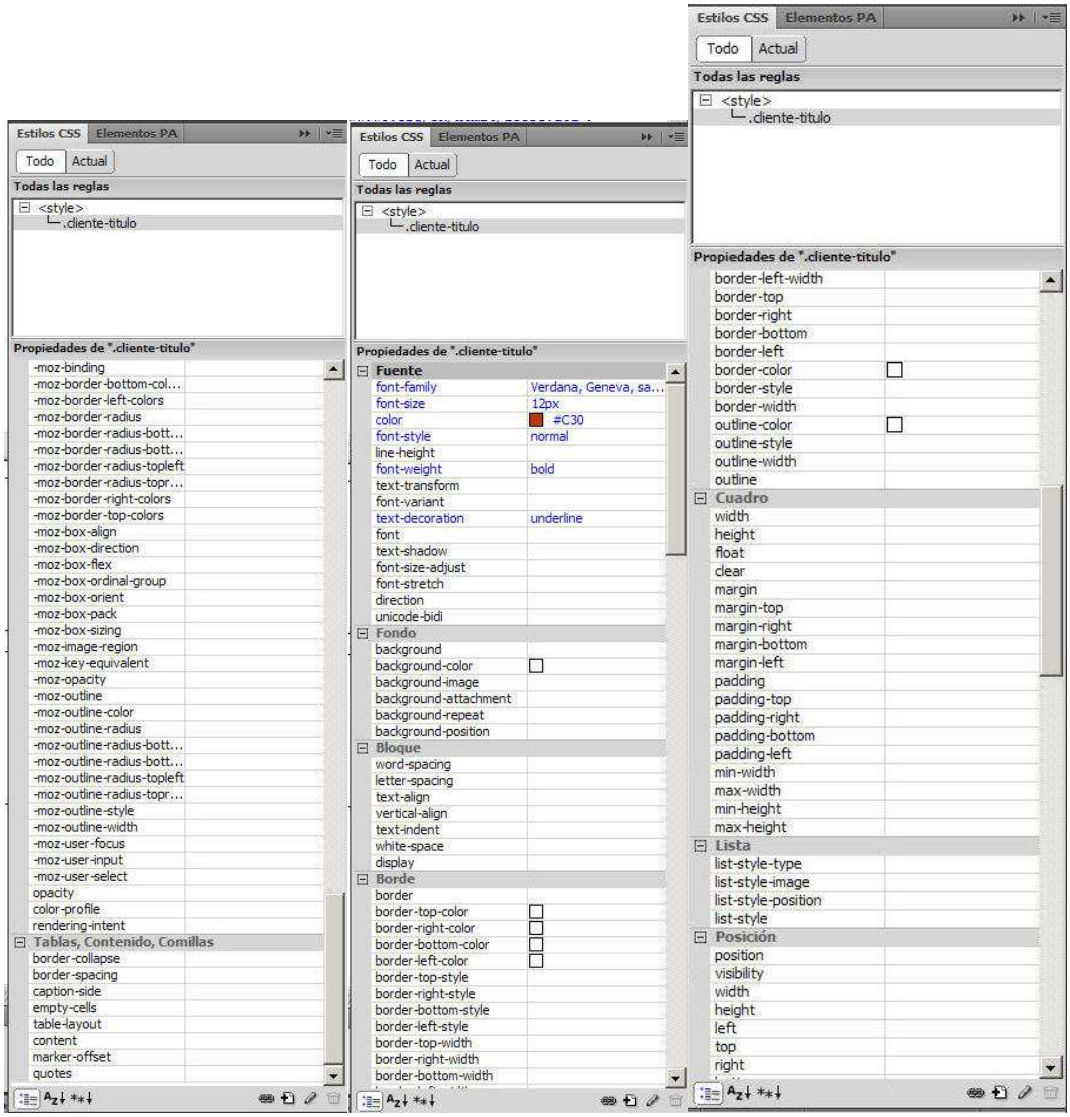
h1 {color: red;}
h2 {color: red;}
h3 {color: red;}

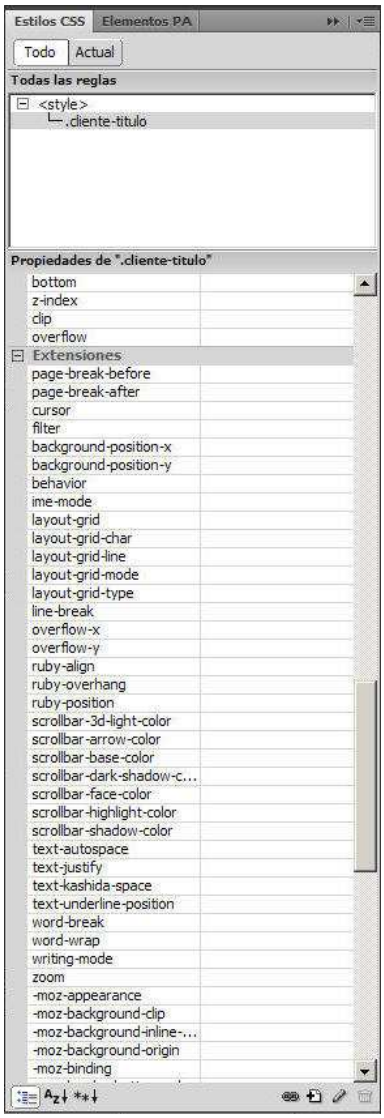
6.3 TIPOS DE ESTILOS

Las tres formas más conocidas de dar estilo a un documento son las siguientes:

- Utilizando el elemento **<style>**, en el interior del documento al que se le quiere dar estilo, y que generalmente se situaría en la sección **<head>.** De esta forma los estilos serán reconocidos antes de que la página se cargue por completo.
- Utilizando una hoja de estilo externa que estará vinculada a un documento a través del elemento **<link>**, el cual debe ir situado en la sección **<head>**.
- Utilizando estilos directamente sobre aquellos elementos que lo permiten a través del atributo **style** dentro de <body> y su etiquetas. Este tipo de definición del estilo pierde las ventajas que ofrecen las hojas de estilo al mezclarse el contenido con la presentación.

Dado que Dreamweaver, la herramienta de desarrollo de páginas web que hemos usado, facilita bastante la asignación de formatos y hojas de estilos, conviene profundizar algo en el tema. Para empezar el conjunto de reglas que pone a nuestra disposición Dreamweaver se puede apreciar en las siguientes capturas:





Como ya hemos visto, el mecanismo que se sigue para aplicar hojas de estilo es teclearlas directamente en la etiqueta <head>, enlazarlas con la etiqueta <link> o directamente aplicarlas son el atributo style en la etiqueta que se desee.

En el siguiente ejemplo se apreciaran los tres mecanismos.

1. Para empezar, partimos de una página web sin estilos con la que previamente se generaron capas.

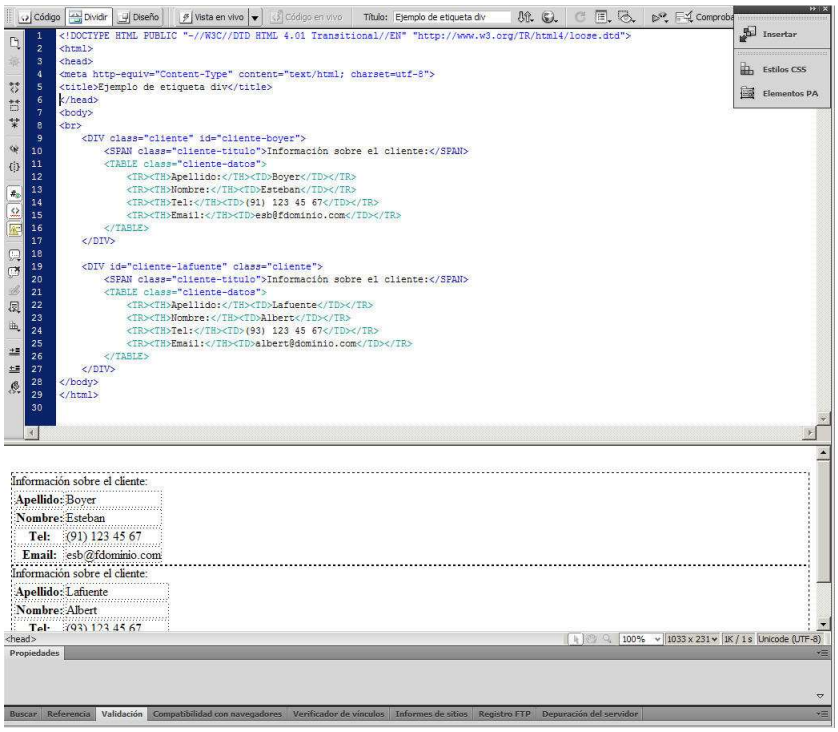


Figura 130. Ejemplo para aplicar estilos CSS

2. Se pulsa sobre el bloque de insertar elementos, en el botón de **Estilos CSS**.

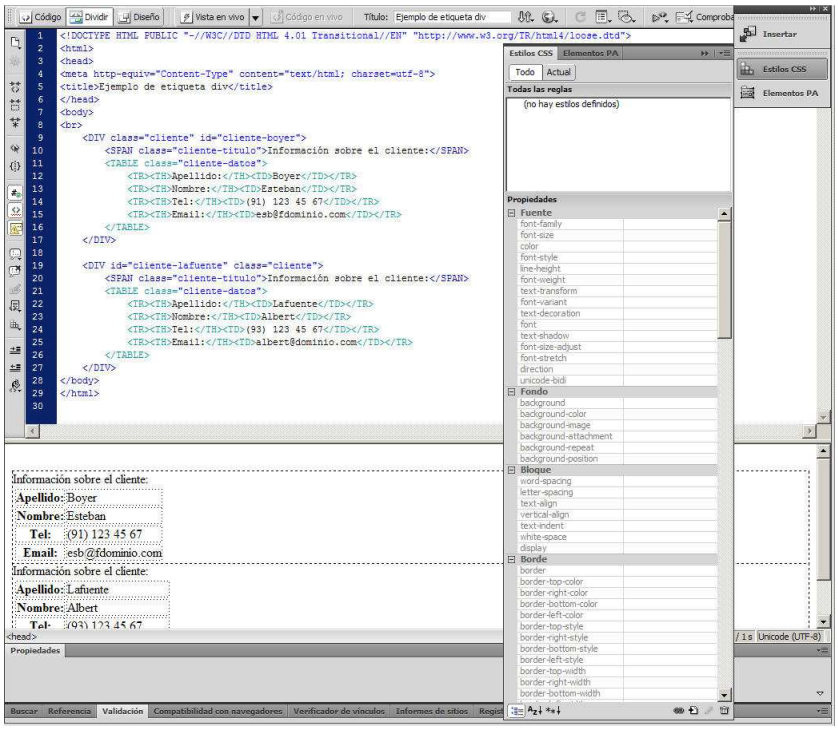


Figura 131. Insertar Estilos CSS

3. Se puede apreciar en el código que ya están usándose algunos estilos, gracias al atributo **class**. Se puede apreciar esta situación especialmente en las etiquetas div, span y table.

```
<DIV class="cliente" id="cliente-boyer" style="border: 1px solid #000000;">

        <SPAN class="cliente-titulo">Información sobre el cliente:</SPAN>

        <TABLE class="cliente-datos">

        <TR><TH>Apellido:</TH><TD>Boyer</TD></TR>

        <TR><TH>Nombre:</TH><TD>Esteban</TD></TR>

        <TR><TH>Tel:</TH><TD>(91) 123 45 67</TD></TR>

        <TR><TH>Email:</TH><TD>esb@fdominio.com</TD></TR>

        </TABLE>

</DIV>
```

Código Figura 145 con clases de estilos.

a. Sin embargo, dichas "clases" de las hojas de estilo no están definidas. Vamos proceder a definirlas.

4. En primer lugar, habrá que crear estas clases. Desde el panel de Estilos CSS, se pulsa sobre donde aparece **<no hay estilos definidos>**.

5. Una vez activado este elemento, con el botón derecho se hace Clic y del menú desplegable que surge se pulsa sobre **Nuevo**.

6. En la ventana que se abre escribimos **cliente-titulo** en el **Nombre del selector**.

Figura 132. Creando una nueva regla

7. Nada más pulsar sobre **Aceptar**, se añade el nombre de la regla al listado disponible y se abre otra ventana para definir sus propiedades. En este caso sólo se elegirán atributos de tipo de texto. Se elige una tipografía Verdana, Geneva o sans-serif; un tamaño de fuente de 12 pixeles, un estilo de fuente oblicuo o cursiva, en negrita, subrayado y convertido en mayúsculas la primera letra de cada palabra. Además se elige un color de texto con tonalidad rojiza.

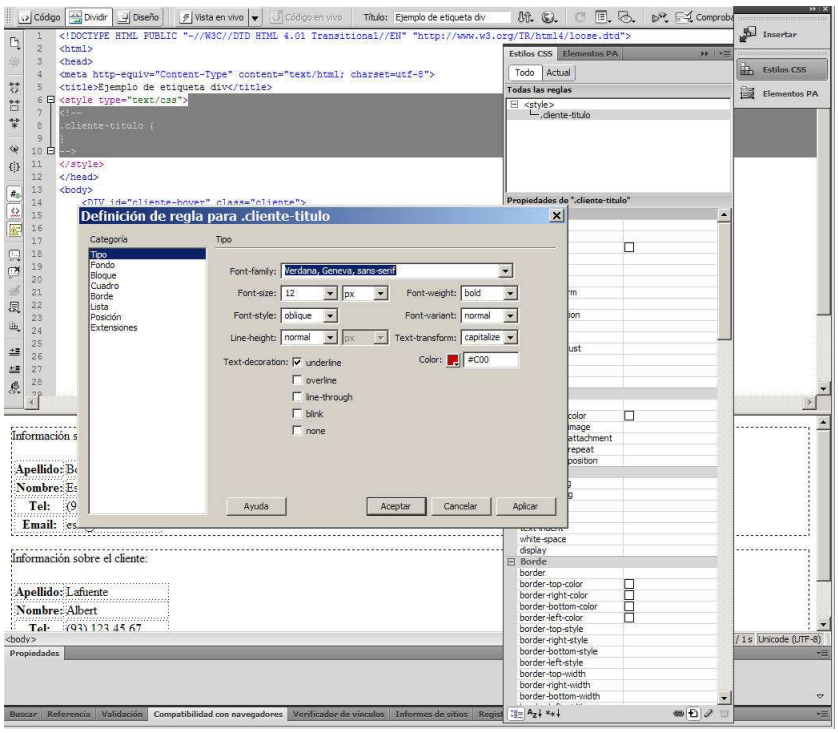


Figura 133. Definición de regla

8. Cuando se pulsa sobre **Aceptar**, se genera el correspondiente código de la regla de la hoja de estilos.

    a. Como puede apreciarse, la etiqueta <style> se genera dentro de la etiqueta <head>.

    b. Su tipo es type=text/css.

    c. Cada selector o referencia al atributo class de HTML se precede por un punto. A continuación se engloba entre los simbolos { y } la definición del estilo.

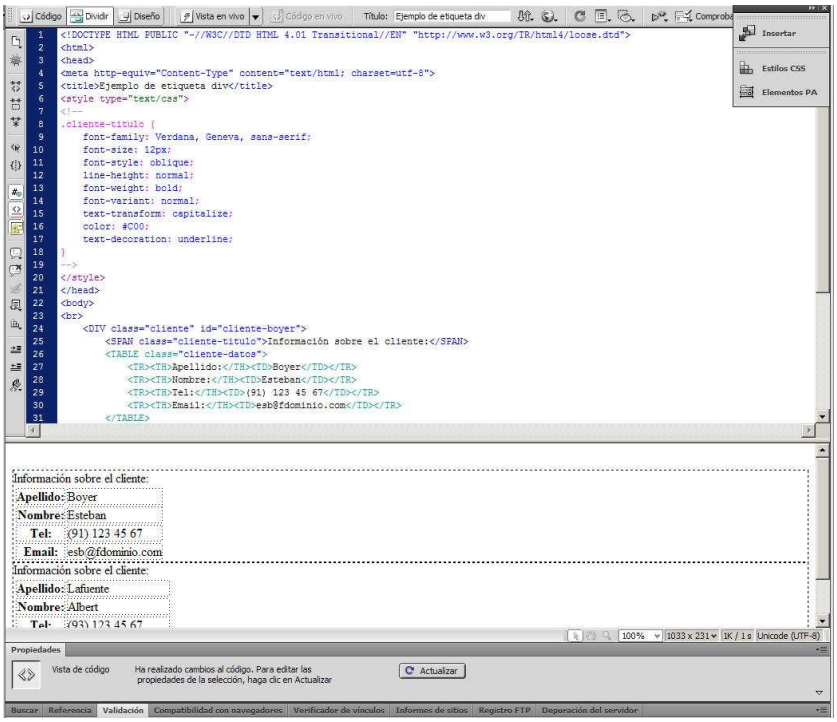


Figura 134. Regla CSS incrustada en <head>

El código correspondiente a la hoja de estilo sería el siguiente:

```
<style type="text/css">
<!--
.cliente-titulo {
	font-family: Verdana, Geneva, sans-serif;
	font-size: 12px;
	font-style: oblique;
	line-height: normal;
	font-weight: bold;
	font-variant: normal;
	text-transform: capitalize;
	color: #C00;
	text-decoration: underline;
}
-->
</style>
```

Código Figura 148. Regla CSS incrustada en <head>

9. El siguiente paso es generar una hoja de estilo que vamos a almacenar como fichero independiente. Un **archivo que contiene estilos CSS** debe tener una extensión **.css**, por ejemplo **estilos.css**. Generalmente, los estilos CSS se colocan en un archivo aparte, aunque se pueden poner en el

mismo archivo html, esto no es recomendable ya que aumentan el tamaño de la pagina web y hacen más pesada su carga.

a. El procedimiento que se sigue es **enlazar el fichero con la etiqueta <link>** donde le indicaremos la dirección del archivo. Para lograr esto, tendremos que poner esa etiqueta en la cabecera <head>. Por ejemplo:

<link href="estilos.css" rel="stylesheet" type="text/css">

b. Existe otra forma de referenciar hojas de estilos. Se denomina **importar un fichero CSS** y es poco conocida y usada. Se pueden ver varios ejemplos de este formato a continuación:

```
@import "subs.css";
@import "print-main.css" print;
@media print {
  body { font-size: 10pt }
}
h1 {color: blue }
```

c. Si la ubicación de nuestra hoja de estilos es otra, por ejemplo, se encuentra en "/css/estilos.css" solo lo cambiamos en el atributo href.

d. Utilizamos el asistente de creación de nuevos documentos con uno de los ejemplos disponibles. Ya hemos visto este proceso por lo que lo omitiremos. Este proceso genera un nuevo fichero que se va a grabar como estilos.css.

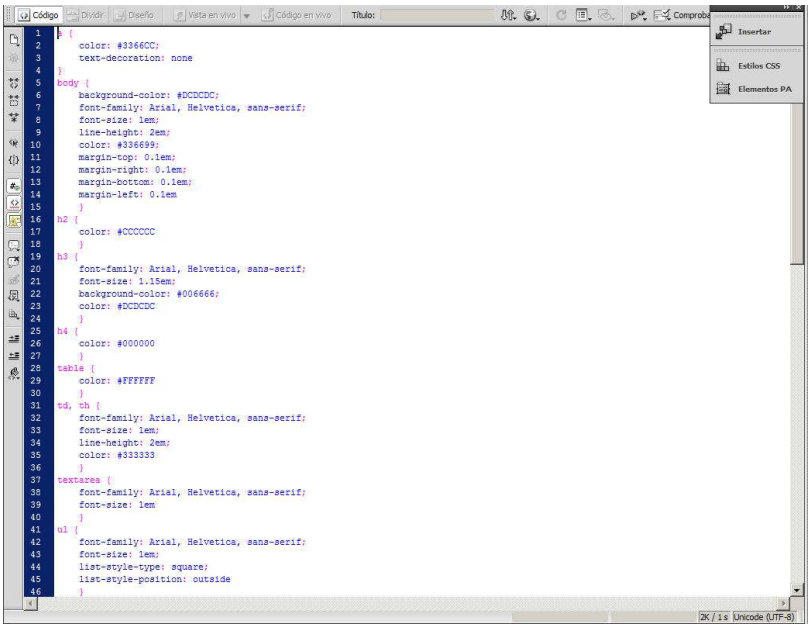


Figura 135. Fichero estilos.css

10. Tras aplicar la etiqueta <link>, se puede apreciar en la vista de diseño que se modifica la apariencia gráfica.

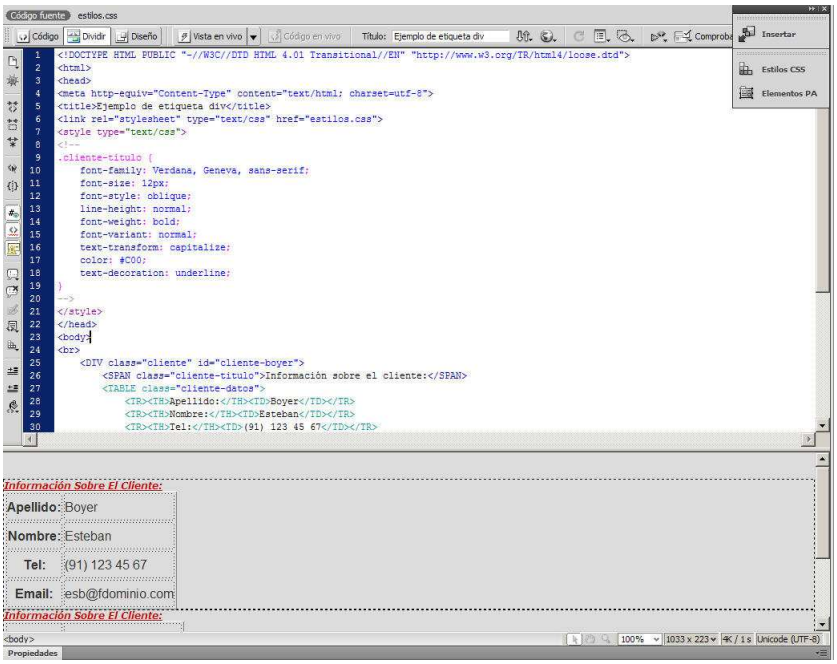


Figura 136. Etiqueta link con hoja de estilos

11. Finalmente sólo queda por utilizar el atributo style dentro de una etiqueta HTML. En este caso se va a modificar la capa definida en la etiqueta div.

    e. Pulsamos sobre la ventana de código en Dreamweaver en la primera etiqueta div tras el comienzo de <body>. Aparece un icono que nos indica si queremos activar el navegador de código.

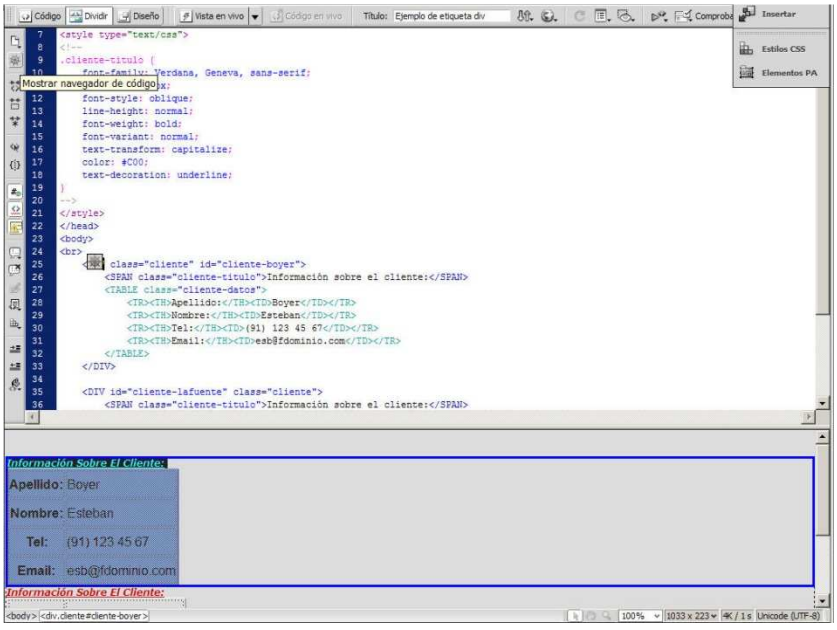


Figura 137, Acceso a navegador de código

    f. Si se activa se nos indica que estilos están siendo aplicados mediante herencia de otras hojas de estilo. En este caso debido al fichero estilos.css.

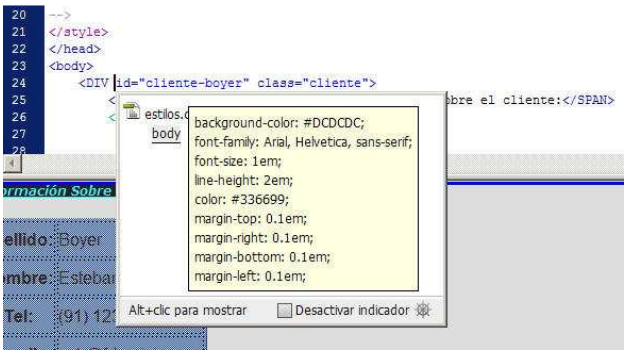


Figura 138. Estilos heredados de otras hojas de estilo

g. Se hace clic con el botón derecho del ratón y se elige **Editar etiqueta div…,** se abrirá una ventana desde la que se tendrá acceso a editar la hoja de estilos. Se elige un borde de 1 pixel de grosor, línea de estilo sólido y color negro.

Figura 139. Editor de etiquetas div

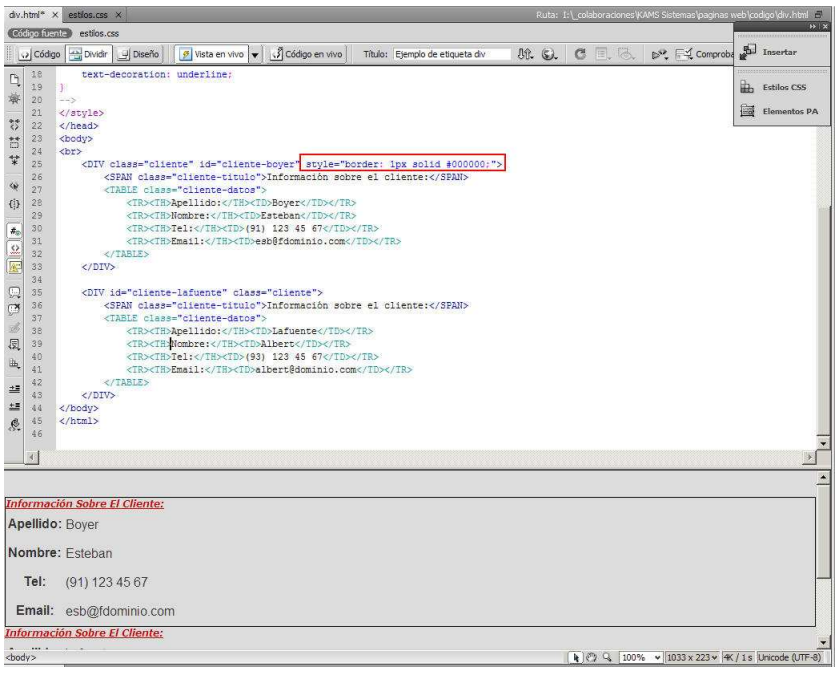


Figura 140. Capa con con CSS definido por atributo style

La etiqueta que nos interesa quedaría tal que:

<div class”cliente” id=”cliente-boyer” style=”border: 1px solid #00000;”>

6.4 ATRIBUTOS DE ESTILO PARA FUENTES, COLOR Y FONDO, TEXTO Y BLOQUES (PÁRRAFOS). CREACIÓN DE FICHEROS DE ESTILO

Ya se ha visto que cada hoja de estilo está compuesta del selector y la definición. Cuando se desea crear una hoja de estilo se debe completar la definición de dicha hoja, especificando sus atributos y propiedades. Estos atributos se pueden aplicar a prácticamente todas las etiquetas del lenguaje HTML. Incluso se pueden sobreescribir las características por defecto de algunas de ellas, como se ha visto anteriormente con la etiqueta **<body>**. A continuación se describe el uso de los atributos más usados para modificar fuentes, texto, párrafos, fondo de la página, y los bloques o capas.

Modificar fuentes

Estos son los atributos que cambian el aspecto de las fuentes:

- **font-family:** Este atributo cambia el tipo de fuente.
  - El siguiente ejemplo indica al navegador que por orden de prioridad imprima el texto con tipografía verdana y luego sans-serif.

font-family: verdana,sans-serif;

- **font-size:** Cambia el tamaño de la fuente.
  - **font-size: 15px;**
    - **siendo px la unidad, en pixeles.**

- **color:** Con este atributo cambiamos el color de la fuente.
  - **S**e puede seleccionar el color mediante el nombre del color en inglés (red, gray, green, etc), formato RGB (rgb(255, 255, 255) o valor hexadecimal (#F4F4F4)).
  - La sintaxis sería así: **color: #fff;**

- **font-style:** Se cambia el estilo de la fuente.
  - Se puede elegir entre normal, oblique e italic.
  - Un ejemplo de su sintaxis sería de la forma: **font-style: italic;**

Modificar texto y párrafos

- **text-align:** Con este atributo modificamos la alineación del texto, pudiendo elegir entre left, center, right y justify.
    - Por ejemplo: **text-align: left;**

- **text-decoration:** Si queremos que nuestro texto este subrayado, sobrerayado o tachado, escribiremos **text-decoration:** seguido de alguna de estas palabras: underline, overline, line-through.
    - Si no se desea aplicar decoración se utiliza none. Por ejemplo: **text-decoration: none;**

Modificar el fondo

- **background-image:** Con este atributo le decimos al navegador que queremos una imagen de fondo.
    - Por ejemplo: **background-image: url(/imagenes/fond.jpg);**

- **background-color:** Se selecciona un color de fondo.
    - Por ejemplo: **background-color: #000000;**

Modificar un bloque o capa

Si queremos modificar una capa o bloque utilizamos estos atributos:

- **margin:** Le especificamos un margen en px o em.
    - Este atributo se puede hacer más específico. Por ejemplo: **margin-left (margen izquierdo)**, **margin-right (margen derecho)**, **margin-top (margen superior)**, o **margin-bottom (margen inferior).**

- **padding:** Se indica el espacio entre la orilla del elemento y el espacio del contenido del elemento.
    - Al igual que el atributo margin, podemos especificarlo aun mas mediante **padding-left (margen izquierdo), padding-right (margen derecho), padding-top (margen superior), padding-bottom (margen inferior)**.

- **border:** Se aplica un borde al bloque.
    - El formato posee tres partes: el tamaño, el tipo y el color.
    - Por ejemplo: **border: 1px solid #000000;**

- Primero especificamos el tamaño del borde, de 1 pixel y luego el tipo (solid, double, etc), Finalmente el color.

- **float:** Como su nombre lo indica, con este atributo hacemos flotar el bloque.
  - Se puede elegir right, none, left.
  - Por ejemplo: **float: left;**

- **clear:** Este atributo se usa para modificar la alineación de los elementos que están a los lados.
  - Se dispone de: both, none, right, y left.
  - Por ejemplo: **clear: both;**

6.5 HOJAS DE ESTILO Y ACCESIBILIDAD

El uso de hojas de estilo está perfectamente adaptado a una de las normas básicas de la accesibilidad: separar los datos de los estilos, pero sin que estos interfieran en los contenidos.

Se puede encontrar más información sobre accesibilidad y CSS en el documento del organismo W3C **Técnicas CSS para las Pautas de Accesibilidad al Contenido en la Web 1.0** en la dirección:

http://www.w3.org/TR/2000/NOTE-WCAG10-CSS-TECHS-20001106/

http://www.discapnet.es/web_accesible/tecnicas/css/WCAG10-CSS-TECHS_es.html (traducido al español)

A continuación se puede encontrar algunas conclusiones resumidas y obtenidas del citado documento:

1. Use un **número mínimo de hojas de estilo** en su sitio.

2. Use hojas de estilo **vinculadas** en vez de estilos incrustados (directamente en las etiquetas) y evite hojas de estilo incrustadas (directamente en la página).

3. Si tiene más de una, **use el mismo nombre de clase** ("class") **para el mismo concepto** en todas las hojas de estilo.

4. Use la unidad **"em"** para fijar el **tamaño de letra**.

5. Utilice **unidades de medida relativas y porcentajes**. Utilice medidas absolutas de longitud sólo cuando las características físicas del medio de salida sean conocidas, por ejemplo con imágenes de mapa de bits.

6. Proporcione un **equivalente textual** para cualquier imagen o texto importantes generados por la hoja de estilo (por ejemplo, mediante las propiedades "background-image", "list-style", o "content").
    - CSS2 permite a los usuarios acceder a representaciones alternativas de los contenidos especificadas en valores de los atributos cuando se emplean juntos los siguientes:
        - selectores de atributos.
        - la función attr() y la propiedad "content".
        - los seudo-elementos :before (antes) y :after (después)

7. Asegúrese de que **todo contenido importante aparezca dentro del objeto del documento**. El texto generado por las hojas de estilo no forma parte del código fuente del documento y no estará disponible para las ayudas técnicas que acceden al Modelo de Objeto del Documento Nivel 1 ([DOM1]).

    CSS2 incluye diferentes mecanismos que permiten generar contenido desde la hoja de estilo:
    - Los seudo-elementos :before y :after y la propiedad "content". Cuando éstos se emplean conjuntamente, permiten la inserción de marcadores antes o después del contenido del elemento.
    - Las propiedades "cue", "cue-before", y "cue-after". Estas propiedades permiten a los usuarios reproducir un sonido antes o después del contenido de un elemento.

8. Especifique siempre un **tipo de letra genérico por defecto**.

9. Use las siguientes propiedades CSS2 para controlar la información de la fuente: "font", "font-family", "font-size", "font-size-adjust", "font-stretch", "font-style", "font-variant", y "font-weight", en lugar de los siguientes elementos y atributos de tipo de letra **desaconsejados** en HTML: FONT, BASEFONT, "face", y "size".

10. Si tiene que usar los elementos HTML para controlar la información sobre el tipo de letra, utilice **BIG y SMALL, que no están desaconsejados**.

11. Las siguientes propiedades CSS2 se pueden emplear para dar **estilo al texto**:
    - Mayúsculas/minúsculas: "text-transform" (para mayúsculas, minúsculas y primera letra mayúscula).
    - Efectos de sombra: "text-shadow"
    - Subrayado: "text-decoration".

12. **No use los elementos BLINK o MARQUEE**. Estos elementos no aparecen en ninguna especificación W3C para HTML (es decir, son elementos no estándares)

    Si se emplea contenido parpadeante (por ejemplo, un titular que aparece y desaparece a intervalos regulares), proporcione un mecanismo para detener el parpadeo. Con CSS, "text-decoration: blink" producirá el efecto de parpadeo y además permitirá al usuario detener el efecto desactivando las hojas de estilo o redefiniendo la regla en una hoja de estilo de usuario.

13. **Utilice hojas de estilo para dar estilo al texto, mejor que representar el texto con imágenes**. Usar texto en lugar de imágenes significa que la información estará disponible para un mayor número de usuarios y permitirá a los usuarios redefinir los estilos del autor y cambiar los colores o los tamaños de letra más fácilmente.
    Si es necesario utilizar un mapa de bits para crear un efecto de texto (letra especial, transformación, sombras, etc.) el mapa de bits debe ser accesible.

14. Las siguientes propiedades CSS2 pueden ser usadas para **controlar el formateo y posición del texto**:
    - Sangría: "text-indent". **No utilice BLOCKQUOTE** o cualquier otro elemento estructural **para hacer sangrías en el texto**.
    - Espaciado de letras o palabras: "letter-spacing", "word-spacing". Por ejemplo, en lugar de escribir "H O L A" (que los usuarios generalmente reconocen como la palabra "hola", pero que un lector de pantalla leería como letras independientes) los autores pueden crear el mismo efecto visual aplicando a "HOLA" la propiedad "word-spacing". Los textos sin espacios serán transformados en discurso más fácilmente.
    - Espacio en blanco: "white-space". Esta propiedad controla la interpretación del espacio en blanco del contenido de un elemento.
    - Dirección del texto: "direction", "unicode-bidi".
    - Los pseudoelementos: first-letter y :first-line permiten a los autores hacer referencia a la primera letra o línea de un párrafo del texto.

15. **Use números en vez de nombres, para especificar colores.**

    Use estas propiedades CSS para especificar colores:
    - "color", para el color de primer plano del texto.
    - "background-color", para el color de fondo.
    - "border-color", "outline-color" para colores de bordes.
    - Para colores de vínculos, haga referencia a las pseudo-clases :link, :visited, y :active.

16. Asegúrese de que los **colores de primer plano y de fondo tienen buen contraste**. Si especifica el color de primer plano, siempre debe especificar también el color de fondo (y viceversa).

17. Asegúrese de que **la información no se transmite sólo a través del color**.

18. Emplee **UL para listas sin ordenar y OL para las ordenadas** (por ejemplo, utilizar marcadores de forma apropiada) conjuntamente con CSS para proporcionar indicaciones contextuales.(1)

19. Hasta que CSS2 sea ampliamente utilizada por las aplicaciones de usuario o éstas permitan al usuario controlar la interpretación de las listas a través de otros medios, los autores deberían considerar el proporcionar **pistas contextuales en las listas anidadas no numeradas**. (Ver ejemplo en el documento original)

20. **Los contenidos deben ser maquetados**, ubicados, colocados en capas y alineados **mediante hojas de estilo** (sobre todo mediante las propiedades CSS de float y colocación absoluta) y no mediante tablas:
    - Las propiedades "text-indent", "text-align", "word-spacing" y "font-stretch", permiten a los usuarios controlar el espaciado sin añadir espacios adicionales. Utilice 'text-align:center' en lugar del **elemento desaconsejado CENTER**.
    - Con las propiedades 'margin', 'margin-top', 'margin-right', 'margin-bottom' y 'margin-left', los autores pueden crear espacios en los cuatro lados del contenido de un elemento, **en lugar de añadir espacios de no separación ( )**.
    - Con las propiedades "float", "position", "top", "right", "bottom" y "left", el usuario puede controlar la posición visual de casi cualquier elemento con independencia de donde aparezca el elemento en el documento. Las propiedades de ubicación pueden ser usadas para crear notas marginales (que se numerarán automáticamente), barras laterales, efectos similares a los marcos, encabezamientos y pies simples y otras más.
    - La propiedad "empty-cells" permite a los usuarios dejar vacías celdas de tablas y poder proporcionarles bordes en la pantalla o en papel. **Una celda de datos que debe estar vacía no debería ser llenada con un espacio en blanco o un espacio "non-breaking" sólo para lograr un efecto visual**.

21. Proporcione **textos equivalentes para todas las imágenes**, incluyendo las imágenes invisibles o transparentes. Si los diseñadores de contenido no pueden usar hojas de estilo y deben utilizar **imágenes invisibles o transparentes** (por ejemplo, con IMG) para diseñar con imágenes en las páginas, **deberían especificar alt=""** para ellas.

22. Utilice las **hojas de estilo para crear líneas y bordes**. Las líneas y bordes pueden transmitir la noción de "separación" a los usuarios que pueden ver, pero este sentido no puede ser deducido fuera de un contexto visual.

    Utilice las propiedades CSS para especificar los estilos de los bordes:
    - "border", "border-width", "border-style", "border-color".
    - Para las tablas, "border-spacing" y "border-collapse".
    - Para contornos dinámicos, "outline", "outline-color", "outline-style" y "outline-width".

23. Asegúrese de que **la presentación del contenido es comprensible cuando no se aplican hojas de estilo**. Los autores deberían diseñar siempre documentos que tengan sentido sin hojas de estilo (por ejemplo, el documento debería escribirse en un orden "lógico") y entonces aplicar hojas de estilo para lograr efectos visuales.

24. Estudie la necesidad de utilizar **propiedades auditivas** de CSS2, las cuales proporcionan información para usuarios invidentes y usuarios de navegadores de voz de manera parecida al tipo de letra que proporciona información visual.

    Las siguientes propiedades forman parte de hojas de estilo en cascada de CSS2.
    - "volume" controla el volumen del texto hablado.
    - "speak" determina si el contenido se pronunciará y, en caso afirmativo, si se debe deletrear o leer como palabras.
    - "pause", "pause-before", y "pause-after" controla las pausas antes y después de anunciar el contenido. Permite a los usuarios separar los contenidos para mejorar la comprensión.
    - "cue", "cue-before", y "cue-after" especifican un sonido que se reproducirá antes y después del contenido, lo que puede ser valioso para la orientación (parecido a una imágen visual).
    - "play-during" controla los sonidos de fondo durante la presentación del elemento (parecido a un imagen de fondo).
    - "azimuth" and "elevation" proporcionan una dimensión al sonido, lo que permite a los usuarios distinguir las voces, por ejemplo.
    - "speech-rate", "voice-family", "pitch", "pitch-range", "stress", y "richness" controlan las cualidades de los contenidos hablados. Cambiando estas propiedades para diferentes elementos, los usuarios pueden ajustar con detalle la presentación sonora-auditiva de los contenidos.
    - "speak-punctuation" y "speak-numeral" controlan la forma de decir los números y la puntuación, lo que afecta la calidad de la experiencia para la navegación por voz.
    - la propiedad "speak-header" describe cómo se debe decir la información sobre los encabezados antes de una celda de tabla.

25. Cree **distintas hojas de estilo para** adaptar la presentación del documento a **diferentes dispositivos de salida** (Braille, sintetizadores de voz o dispositivos TTY, pantalla, móvil, etc.) mediante los "tipos de medios" de CSS2 (empleados con las reglas @media) Las reglas "@media" también pueden reducir los tiempos de descarga porque permiten a las aplicaciones de usuario ignorar reglas inapropiadas.

Notas

(1) La siguiente hoja de estilo CSS2 muestra cómo especificar números compuestos para listas anidadas creadas tanto con elementos UL como OL. Los ítems están numerados como "1", "1.1", "1.1.1", etc. Ejemplo:

```
<style type="text/css">
ul, ol { counter-reset: item }
li { display: block }
li:before {
content: counters(item, ".");
counter-increment: item;
}
</style>
```

7

Técnicas de accesibilidad y usabilidad

- Accesibilidad web, ventajas de la accesibilidad
- Aplicaciones para verificar la accesibilidad de sitios web (estándares)
- Usabilidad web, importancia de la usabilidad
- Diseño de sitios web usables. Adaptación de sitios web usables

7.1 ACCESIBILIDAD WEB, VENTAJAS DE LA ACCESIBILIDAD

El principal objetivo de la accesibilidad web es el de permitir a cualquier usuario, independientemente del tipo de discapacidad que sufra, el acceso a los contenidos del sitio y permitirle la navegación necesaria para realizar las acciones deseadas.

Los sitios web accesibles no solamente facilitan el acceso de sus contenidos a los usuarios discapacitados, sino que también permiten ofrecer la misma funcionalidad con dispositivos muy limitados (dispositivos sin pantalla o con pantallas minúsculas, dispositivos sin teclado ni ratón, etc.).

Desde hace varios años, todas las páginas de la administración pública, por ley, deben ser accesibles.

Las cuatro principales ventajas de diseñar un sitio web completamente accesible son las siguientes:

- Los sitios accesibles separan completamente diseño y contenido.
- Un sitio accesible puede ser accedido por multitud de dispositivos diferentes sin necesidad de reescribir el código HTML.
- Los sitios accesibles son los únicos con una audiencia potencial global, ya que permiten el acceso a todos los usuarios y a todos los dispositivos.
- Generalmente, los sitios accesibles son más fáciles de utilizar también para los usuarios sin discapacidades. Esto se debe fundamentalmente a que dichos sitios web estructuran su contenido de forma que implican una secuencia lógica de lectura. Las estructuras de las páginas web suelen basarse en hojas de estilo que no afectan a la consistencia de los contenidos en pantalla.

La creación de sitios accesibles puede realizarse siguiendo las recomendaciones establecidas por el W3C y que se recogen en el documento **Web Content Accessibility Guidelines o WCAG. (**http://www.w3.org/WAI/intro/wcag.php).

La versión WCAG 1.0 (http://www.w3.org/TR/WCAG10/) que se utiliza en la actualidad se publicó en 1999, mientras que la versión WCAG 2.0 (http://www.w3.org/TR/WCAG20/) se encuentra todavía en borrador y se actualizó por última vez el día 30 de abril de 2008.

Las recomendaciones del WCAG 1.0 están formadas por 65 requisitos que un sitio web debe cumplir para considerarse accesible. Los requerimientos se agrupan en prioridades. Los requisitos de prioridad 1 son de obligado cumplimiento, los de prioridad 2 son recomendables y los de prioridad 3 son deseables.

En función del tipo y número de requisitos que cumpla un sitio web, este puede alcanzar un nivel u otro de accesibilidad:

- Si un sitio web cumple con todos los requisitos de prioridad 1, se considera que el sitio es conforme al **nivel A de accesibilidad.**
- El **nivel AA de accesibilidad** está reservado para los sitios que cumplan todos los requisitos de prioridad 1 y prioridad 2.
- Finalmente, los sitios que cumplen los 65 requisitos, son conformes **al nivel AAA de accesibilidad**.

La lista completa con los 65 requisitos de los tres niveles de accesibilidad se puede consultar en:

http://www.w3.org/TR/WCAG10/full-checklist.html

A continuación se hace un repaso a aquellos elementos de accesibilidad más destacables, en función del tipo y uso que hacen de las etiquetas HTML.

- **Generales**

1. Se debe proporcionar un texto alternativo para todas las imágenes, objetos y otros elementos no textuales (mediante los atributos alt y longdesc).
2. Si se usa el color como elemento informativo, esta situación debe ser especificada de formas alternativas para las personas o dispositivos que no pueden distinguir los colores.
3. Marcar claramente (mediante los atributos lang y xml:lang) las variaciones del idioma del texto o de los elementos textuales (<caption>) respecto del idioma principal de la página.
4. Las hojas de estilo no deben influenciar a la hora de poder leer un documento.
5. Cuando existan contenidos dinámicos y estos cambien, se debe informar de en qué consisten dichos cambios.
6. Ningún elemento debe parpadear en la pantalla.
7. El contenido del sitio se debe escribir con un lenguaje sencillo y limpio.

- **Uso de mapas de imagen**

1. Es preciso proporcionar un enlace textual por cada una de las regiones del mapa de imagen.
2. Utilizar mapas de imagen en el cliente, en vez de mapas de imagen de servidor.

- **Uso de tablas**

1. Utilizar cabeceras de fila y de columna.
2. Si la tabla tiene varios niveles de cabeceras, utilizar las agrupaciones disponibles (<thead>, <tfoot>).

- **Uso de marcos (frames)**

1. Es necesario introducir un título, nombre e identificador a cada frame para su identificación y facilitar la navegación.

- **Uso de applets y scripts**

1. Cuando se ejecutan los applets y los scripts de una página se producen unos resultados. Si se desactivan estos elementos se debe proporcionar información equivalente o páginas alternativas que reproduzcan los mismos resultados. Este apartado es clave, pues por ejemplo, una de las primeras cosas que hacen los lectores de texto para invidentes es desactivar javascript.

- **Páginas con contenidos multimedia (audio y vídeo)**

1. Se debe incluir una descripción textual del contenido multimedia.
2. Cuando existan video o animaciones, es preciso que los textos descriptivos de los elementos multimedia estén sincronizados en el momento de la ejecución.

- **Dar una alternativa cuando no se pueden cumplir los anteriores requisitos**

1. Proporcionar una página alternativa con la mayor cantidad posible de contenidos y que cumpla con los requisitos anteriores.

7.2 APLICACIONES PARA VERIFICAR LA ACCESIBILIDAD DE SITIOS WEB (ESTÁNDARES)

Dado que la herramienta de creación de páginas web que se usa en este curso, Dreamweaver, incluyen funciones de validación de accesibilidad, lo lógico es recurrir a estas en primer lugar.

Para ello se deben seguir los siguientes pasos:

1. Desde el menú superior, se selecciona la opción **Sitio** e **Informes.** Se nos abrirá una ventana desde la que elegiremos el tipo de informes que deseamos, así como si se aplica sólo a la página actual como al sitio web completo.

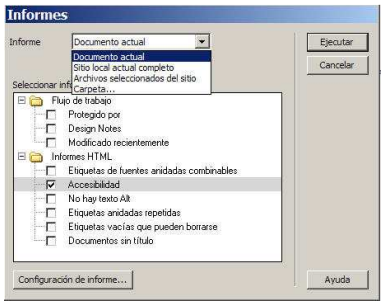


Figura 141. Informes

2. Para configurar las opciones que se desean se activa la casilla de **Accesibilidad** y se pulsa sobre **Configuración de informe**.

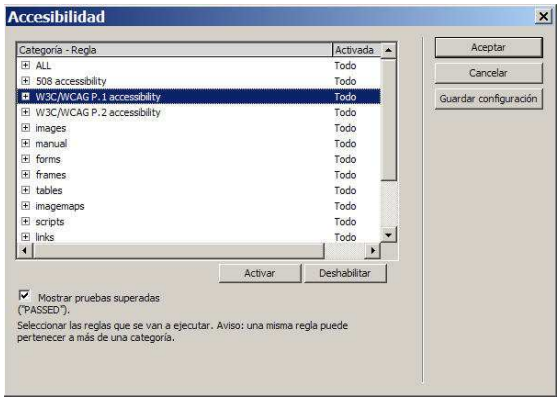


Figura 142. Opciones de configuración de accesibilidad

Otros recursos a la hora de validar si nuestros documentos web cumplen las normas de accesibilidad son:

- W3C Validation Service en http://validator.w3.org/

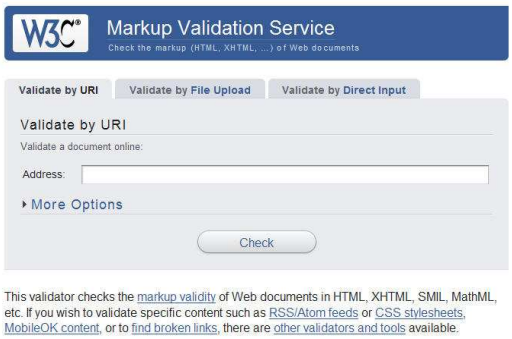


Figura 143. Validación de W3C

- TAW en http://www.tawdis.net

Figura 144. Accesibilidad con TAW

- Fundación SIDAR en http://www.sidar.org

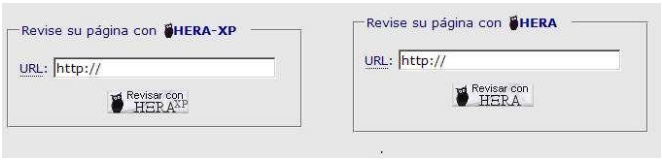


Figura 145. Accesibilidad con SIDAR

7.3 USABILIDAD WEB, IMPORTANCIA DE LA USABILIDAD

Existe una norma sobre usabilidad con frecuencia repetida que dice que, por cada enlace o acción que debe realizar un usuario para acceder a un contenido en una página web, existe otro que desiste. Quizá esta afirmación sea algo exagerada pero no está falta de razón en el fondo. Si a un usuario le cuesta manejar una página web o localizar rápidamente lo que quiere, se desmotivará fácilmente con el tiempo y optará por otras alternativas más sencillas. Todos nos hemos encontrado con páginas que a pesar de tener buenos contenidos, un diseño gráfico espectacular, resultan complicadas de manejar y que debido a esto fácilmente nos olvidamos de ellas. En el mundo de las páginas web de las empresas, que un usuario desista de usar tu página puede suponer mucho dinero de perdidas, por lo que es un tema que se tiene mucho en cuenta.

Sobre usabilidad se lleva hablando mucho tiempo y existen multitud de artículos en Internet. Sin embargo, la clave de todos está en que a la hora de crear un sitio web, sus **páginas web deben estar orientadas a los usuarios**. Aquí se enlaza con los apartados anteriores de accesibilidad, pues cuando se habla de orientación a los usuarios se habla de todos los usuarios. Por sus propias características, un sitio accesible facilita también que este sea más usable por cualquier usuario. Por tanto para diseñar páginas web accesibles basta con seguir normas de ergonomía clásica y aquellas derivadas de las guías de accesibilidad.

A continuación podemos ver algunos de los elementos que conviene tener en cuenta a la hora de hacer una página web más usable:

- El usuario ha de ser quien inicie las acciones y controle las tareas.

- La interfaz ha de ser simple, fácil de aprender y usar, con funcionalidades accesibles y bien definidas.
- El uso del diseño ha de ser fácil de entender, independientemente de la experiencia, conocimiento, capacidades lingüísticas o nivel de concentración del usuario.
- Hay que controlar la información mostrada, que se ha de reducir al mínimo necesario.
- El diseño ha de comunicar la información necesaria al usuario de forma efectiva, independientemente de las condiciones ambientales o de las capacidades sensoriales del mismo.
- Hay que posibilitar el descubrimiento interactivo y el aprendizaje ensayo-error.
- Hay que posibilitar la reversibilidad y la recuperabilidad de las acciones. Es decir, es necesario contemplar los potenciales errores de los usuarios.
- Se ha de facilitar la aplicación de los conocimientos adquiridos. Es decir si por ejemplo se dispone de un buscador en tu página web, su funcionamiento debe ser idéntico en todas las páginas. No se puede variar su funcionamiento y forma de mostrar los resultados si se desea lograr un aprendizaje rápido.
- Los diseños gráficos espectaculares no son muy adecuados si lo que se pretende es que el usuario preste atención al contenido.
    - La simplicidad de diseño favorece la trasmisión de información. Eso no impide que la apariencia gráfica de la página web sea agradable a la vista.
    - Minimizar el peso de los elementos gráficos. Si la página tarda en cargarse estaremos perdiendo usuarios que no desean esperar.
- Los contenidos tienen que estar optimizados para la web. Ser breves y utilizar hipervínculos para separar desarrollos y ejemplos del cuerpo del artículo.
- Utilizar en la presentación y redacción tres características básicas:
    - Texto conciso.
    - Diseño que permita la lectura por encima, como enumerar los puntos, separar títulos y encabezados, utilizar negritas y resaltados o listas con viñetas.
    - Lenguaje objetivo y directo.
- La navegación dentro del sitio web debe permitir al usuario saber dónde está en cada momento y cómo poder volver allí cuando lo desee.

Proceso de diseño orientado a usabilidad

Las anteriores recomendaciones deben integrase en un proceso general de diseño de una página web. La idea clara que se esconde en este proceso es que el usuario de las páginas es quien va a tener la clave de si estas son accesible o no. Por ello se debe tener en cuenta los siguientes detalles:

- **Definición clara de los objetivos**, entendiendo a los usuarios y contemplando factores como la edad, la experiencia en el manejo de aplicaciones web, las limitaciones físicas, las necesidades más especiales, el entorno de trabajo, las influencias sociales y culturales... Aunque parezca algo evidente, muchos diseñadores de páginas web olvidan el usuario al que la página web va dirigido.

No tiene sentido utilizar el mismo diseño para una página web de un videojuego, que la que requerirá un banco.

- Una vez se dispone de un diseño, este debe ser comunicado a los usuarios mediante **prototipos** que reflejen lo más cercanamente el flujo de tareas definido. Se puede tratar de incluir más aspectos y comprobar la reacción a los mismos de los usuarios objetivo o tratar de centrarse en los detalles de dichos aspectos, en su funcionalidad.

- Mediante **pruebas**, en el proceso de diseño, la participación del usuario proporciona la inestimable ayuda de determinar en qué medida el producto se está ajustando a las necesidades y a las expectativas creadas. No se trata tanto de evaluar la eficiencia de las tareas y los posibles errores en el diseño, sino de conocer las percepciones del usuario, su satisfacción, sus preguntas, sus problemas,...

- Después de las pruebas con los usuarios va a ser preciso el **rediseño** en mayor o menor medida, tras el cual inevitablemente, es preciso de nuevo el test, volviendo así a iniciar el ciclo.

7.4 DISEÑO DE SITIOS WEB USABLES. ADAPTACIÓN DE SITIOS WEB USABLES

Aunque pueda resultar una afirmación muy drástica, si es preciso adaptar un sitio web ya finalizado para que sea usable, estamos ante un gran problema: Más valdría empezar desde cero.

Evaluar la usabilidad de nuestro sitio web puede ser algo verdaderamente útil, ya que descubrir qué errores de diseño tiene nuestra web es el primer paso para poder corregirlos y realizar una adaptación adecuada. En cuanto a en qué momento del proyecto es más recomendable evaluar el sitio web, se debe seguir la siguiente regla: *Cuanto más tarde peor*, ya que será más costoso rediseñar todo un sitio ya acabado, que reconducir la línea de desarrollo por mejores caminos. Hay varias formas de evaluación:

- **Expertos**
    - Una opción bastante recomendable es encargar a un experto que evalúe nuestra web. Las herramientas que se utilizan están basadas en evaluaciones heurísticas de sitios web. Veremos un ejemplo más adelante. En Internet hay muchos profesionales independientes y empresas que se dedican a esta tarea.

- **Encuestas**
    - Otra opción es utilizar encuestas para comprobar la usabilidad de nuestro sitio web. La encuesta debería ser diseñada por un experto y realizada sobre usuarios actuales o potenciales de nuestro sitio web.

- **Pruebas de usabilidad**
    - Una prueba o test de usabilidad es contar con un grupo reducido de usuarios en una sala, que realizarán una navegación "asistida" por el sitio web a probar. El encargado de la prueba tomará nota de qué problemas encuentran los usuarios para realizar las tareas que se les hayan indicado, y así conocer qué errores de diseño tiene el sitio web.

Evaluación heurística de sitios web

Los procesos de evaluación heurística consisten en contestar un test sobre la usabilidad del sitio web. Existen miles de ellos y como suele decirse, cada experto tiene el suyo. A continuación podemos ver un ejemplo que puede dar una orientación clara sobre que se persigue a la hora de analizar si un sitio web posee buena usabilidad.

Generales

- ¿Cuáles son los *objetivos* del sitio web? ¿Son concretos y bien definidos? ¿Los contenidos y servicios que ofrece se corresponden con esos objetivos?

- ¿Tiene una *URL* correcta, clara y fácil de recordar? ¿Y las URL de sus páginas internas? ¿Son claras y permanentes?

- ¿Muestra de forma precisa y completa *qué contenidos o servicios ofrece* realmente el sitio web?

  Esto está relacionado directamente con el diseño de la página de inicio, que debe ser diferente al resto de páginas y cumplir la función de 'escaparate' del sitio.

- ¿La *estructura* general del sitio web está orientada al usuario?

  Los sitios web deben estructurarse pensando en el usuario, sus objetivos y necesidades. No se debe calcar la estructura interna de la empresa u organización, al usuario no le interesa cómo funciona o se organiza la empresa.

- ¿El *look & feel* general se corresponde con los objetivos, características, contenidos y servicios del sitio web?

  Por ejemplo, los colores empleados. Aunque el significado que comunica un determinado color es muy subjetivo y dependiente de la cultura y el entorno, y por lo tanto diferente para cada usuario, ciertas combinaciones de colores ofrecen una imagen más o menos formal, seria o profesional, como pueden ser los tonos de azules con el blanco, que trasmiten una imagen corporativista.

- ¿Es *coherente* el diseño general del sitio web?

  Se debe mantener una coherencia y uniformidad en las estructuras y colores de todas las páginas. Esto sirve para que el usuario no se desoriente en su navegación.

- ¿Es *reconocible* el diseño general del sitio web?

  Cuánto más se parezca el sitio web al resto de sitios web, más fácil será de usar.

- ¿El sitio web se *actualiza periódicamente*? ¿Indica cuándo se actualiza?

Las fechas que se muestren en la página deben corresponderse con actualizaciones, noticias, eventos...no con la fecha del sistema del usuario.

Identidad e Información

- ¿Se muestra claramente la *identidad* de la empresa-sitio a través de todas las páginas?
- El *Logotipo*, ¿es significativo, identificable y suficientemente visible?
- El *eslogan* o tagline, ¿expresa realmente qué es la empresa y qué servicios ofrece?
- ¿Se ofrece algún enlace con *información sobre la empresa*, sitio web, 'webmaster',...?
- ¿Se proporciona mecanismos para ponerse en *contacto* con la empresa?

  (Email, teléfono, dirección postal, fax...)

- ¿Se proporciona información sobre la protección de *datos de carácter personal* de los clientes o los *derechos de autor* de los contenidos del sitio web?
- En artículos, noticias, informes...¿Se muestra claramente información sobre el *autor*, *fuentes* y *fechas* de creación y revisión del documento?

Lenguaje y Redacción

- ¿El sitio web habla el *mismo lenguaje que sus usuarios*?

  Se debe evitar usar un lenguaje corporativista. Así mismo, hay que prestarle especial atención al idioma, y ofrecer versiones del sitio en diferentes idiomas cuando sea necesario.

- ¿Emplea un lenguaje *claro y conciso*?
- ¿Es *amigable*, familiar y cercano?

  Es decir, lo contrario a utilizar un lenguaje constantemente imperativo, mensajes crípticos, o tratar con "desprecio" al usuario.

- ¿1 *párrafo* = 1 *idea*?

  Cada párrafo es un objeto informativo. Trasmita ideas, mensajes...Se deben evitar párrafos vacíos o varios mensajes en un mismo párrafo.

Rotulado

- Los rótulos, ¿son *significativos*?

  Ejemplo: evitar rótulos del tipo "haga clic aquí".

- ¿Usa rótulos *estándar*?

  Siempre que exista un "estándar" comúnmente aceptado para el caso concreto, como "Mapa del Sitio" o "Acerca de...".

- ¿Usa un único *sistema de organización*, bien definido y claro?

No se deben mezclar sistemas de organización diferentes. Los diferentes sistemas de organización son básicamente: alfabético, geográfico, cronológico, temático, orientado a tareas, orientado al público y orientado a metáforas.

- ¿Utiliza un *sistema de rotulado* controlado y preciso?

  Por ejemplo, si un enlace tiene el rótulo "Quiénes somos", no puede dirigir a una página cuyo encabezamiento sea "Acerca de", o un enlace con el rótulo "Ayuda" no puede dirigir a una página encabezada con "FAQs".

- El *título de las páginas*, ¿Es correcto? ¿Ha sido planificado?

  Relacionado con la 'findability' del sitio web.

Estructura y Navegación

- La *estructura de organización y navegación*, ¿Es la más adecuada?

  Hay varios tipos de estructuras: jerárquicas, hipertextual, facetada,...

- En el caso de estructura *jerárquica*, ¿Mantiene un equilibrio entre Profundidad y Anchura?
- En el caso de ser puramente *hipertextual*, ¿Están todos los clusters de nodos comunicados?

  Aquí se mide la distancia entre nodos.

- ¿Los *enlaces* son fácilmente *reconocibles* como tales? ¿su caracterización indica su estado (visitados, activos,...)?

  Los enlaces no sólo deben reconocerse como tales, sino que su caracterización debe indicar su estado (para orientar al usuario), y ser reconocidos como una unidad (enlaces que ocupan más de una línea).

- En menús de navegación, ¿Se ha controlado el número de elementos y de términos por elemento para no producir *sobrecarga memorística*?

  No se deben superar los 7±2 elementos, ni los 2 o, como mucho, 3 términos por elemento.

- ¿Es *predecible la respuesta del sistema* antes de hacer clic sobre el enlace?

  Esto está relacionado con el nivel de significación del rótulo del enlace, aunque también con: el uso de globos de texto, información contextual (indicar formato y tamaño del documento o recurso con el que vincula el enlace), la barra de estado del navegador,...

- ¿Se ha controlado que no haya *enlaces que no llevan a ningún sitio*?

  Enlaces que no llevan a ningún sitio: Los enlaces rotos, y los que enlazan con la misma página que se está visualizando (por ejemplo enlaces a la "home" desde la misma página de inicio)

- ¿Existen *elementos de navegación que orienten* al usuario acerca de dónde está y cómo deshacer su navegación?

...como breadcrumbs, enlaces a la página de inicio,...recuerde que el logo también es recomendable que enlace con la página de inicio.

- Las *imágenes enlace*, ¿se reconocen como pulsables? ¿incluyen un atributo 'title' describiendo la página de destino?

  En este sentido, también hay que cuidar que no haya imágenes que parezcan enlaces y en realidad no lo sean.

- ¿Se ha evitado la *redundancia de enlaces*?
- ¿Se ha controlado que no haya *páginas "huérfanas"*?

  Páginas huérfanas: que aún siendo enlazadas desde otras páginas, éstas no enlacen con ninguna.

Lay-Out de la Página

- ¿Se aprovechan las *zonas de alta jerarquía informativa* de la página para contenidos de mayor relevancia?

  (Como por ejemplo la zona central)

- ¿Se ha evitado la *sobrecarga informativa*?

  Esto se consigue haciendo un uso correcto de colores, efectos tipográficos y agrupaciones para discriminar información. Al igual que en los elementos de un menú de navegación, los grupos diferentes de objetos informativos de una página, no deberán superar el número 7±2.

- ¿Es una interfaz limpia, sin *ruido visual*?
- ¿Existen *zonas en "blanco"* entre los objetos informativos de la página para poder descansar la vista?
- ¿Se hace un uso correcto del *espacio visual* de la página?

  Es decir, que no se desaproveche demasiado espacio con elementos de decoración, o grandes zonas en "blanco", y que no se adjudique demasiado espacio a elementos de menor importancia.

- ¿Se utiliza correctamente la *jerarquía visual* para expresar las relaciones del tipo "parte de" entre los elementos de la página?

  (La jerarquía visual se utiliza para orientar al usuario)

- ¿Se ha controlado la *longitud de página*?

  Se debe evitar en la medida de lo posible el scrolling. Si la página es muy extensa, se debe fraccionar.

Búsqueda

- ¿Se encuentra fácilmente *accesible*?

Es decir: directamente desde la home, y a ser posible desde todas las páginas del sitio, y colocado en la zona superior de la página.

- ¿Es fácilmente *reconocible* como tal?
- ¿Permite la *búsqueda avanzada*?

(Siempre y cuando, por las características del sitio web, fuera de utilidad que la ofreciera)

- ¿Muestra los *resultados* de la búsqueda de forma comprensible para el usuario?
- ¿La *caja de texto* es lo suficientemente ancha?
- ¿Asiste al usuario en caso de *no poder ofrecer resultados* para una consultada dada?

Elementos Multimedia

- ¿Las *fotografías* están bien recortadas? ¿son comprensibles? ¿se ha cuidado su resolución?
- ¿Las *metáforas visuales* son reconocibles y comprensibles por cualquier usuario?

(Prestar especial atención a usuarios de otros países y culturas)

- ¿El uso de *imágenes o animaciones* proporciona algún tipo de valor añadido?
- ¿Se ha evitado el uso de *animaciones cíclicas*?

Ayuda

- Si posee una *sección de Ayuda*, ¿Es verdaderamente necesaria?

Siempre que se pueda prescindir de ella simplificando los elementos de navegación e interacción, debe omitirse esta sección.

- En *enlace a la sección de Ayuda*, ¿Está colocado en una zona visible y "estándar"?

La zona de la página más normal para incluir el enlace a la sección de Ayuda, es la superior derecha.

- ¿Se ofrece *ayuda contextual* en tareas complejas (transferencias bancarias, formularios de registro...)?

- Si posee *FAQs*, ¿es correcta tanto la elección como la redacción de las preguntas? ¿y las respuestas?

Accesibilidad

- ¿El *tamaño de fuente* se ha definido de forma relativa, o por lo menos, la fuente es lo suficientemente grande como para no dificultar la legibilidad del texto?
- ¿El *tipo de fuente*, efectos tipográficos, ancho de línea y alineación empleados facilitan la lectura?
- ¿Existe un alto *contraste* entre el color de fuente y el fondo?
- ¿Incluyen las imágenes atributos '*alt*' que describan su contenido?
- ¿Es *compatible* el sitio web con los diferentes navegadores? ¿Se visualiza correctamente con diferentes resoluciones de pantalla?

Se debe prestar atención a: JScript, CSS, tablas, fuentes...

- ¿Puede el usuario disfrutar de todos los contenidos del sitio web sin necesidad de tener que descargar e instalar *plugins* adicionales?
- ¿Se ha controlado el *peso* de la página?

  Se deben optimizar las imágenes, controlar el tamaño del código JScript...

- ¿Se puede *imprimir* la página sin problemas?

  Leer en pantalla es molesto, por lo que muchos usuarios preferirán imprimir las páginas para leerlas. Se debe asegurar que se puede imprimir la página (no salen partes cortadas), y que el resultado es legible.

Control y Retroalimentación

- ¿Tiene el usuario todo el control sobre el interfaz? Se debe evitar el uso de ventanas pop-up, ventanas que se abren a pantalla completa, banners intrusivos...
- ¿Se informa constantemente al usuario acerca de lo que está pasando? Si la página tarda en cargar puede pensar que se ha colgado si no se le informa.
- ¿Se informa al usuario del resultado de sus acciones?
- Cuando se produce un *error*, ¿se informa de forma clara y no alarmista al usuario de lo ocurrido y de cómo solucionar el problema?
- ¿Posee el usuario libertad para actuar? ¿Desaparecen opciones del navegador, del ratón,…?
- ¿Se ha controlado el tiempo de respuesta del usuario? El tiempo máximo que esperará un usuario sin incomodarse son 10 segundos.

Bibliografía

CREACION DE PAGINAS WEB EDICION 2003 (GUIAS VISUALES)
de PARDO NIEBLA, MIGUEL
ANAYA MULTIMEDIA, 2002

CREATING WEB SITES BIBLE (2ND EDITION)
de CROWDER, DAVID A. y BAILEY, ANDREW
IDG BOOKS, 2004

CSS
de MCFARLAND, DAVID SAWYER
ANAYA MULTIMEDIA, 2010

COMPENDIUM HTML
de BORN, GÜNTER
MARCOMBO, S.A., 2001