

Ejemplo:

```
<table border="3" cellspacing="0" cellpadding="5" width="390">
  <tr>
    <th width="10">Cod</th><th width="80">Nombre</th>
    <th width="300">Significado</th>
  </tr>
  <tr><td>1</td><td>SGML</td><td>Standard Generalized Markup
    Language</td> </tr>
  <tr><td>2</td><td>HTML</td><td>HyperText Markup
    Language</td></tr>
  <tr><td>3</td><td>XML</td><td>Extended Markup Language</td>
  </tr>
</table>
```

Cod	Nombre	Significado
1	SGML	Standard Generalized Markup Language
2	HTML	HyperText Markup Language
3	XML	Extended Markup Language

Figura 2.17: La cabecera <th> centra su contenido automáticamente

## <caption>

Definición: sirve para poner un título a la tabla, aparece justo encima.

Aparición: etiquetas inicial y final obligatorias.

Atributos: %attrs (align desaprobado).

Diseño (Layout) a 3 columnas

LOGO		
Menú Izquierdo	Contenido	Menú Derecho
PIE DE PÁGINA		

Figura 2.18: Tabla con <caption>

### 2.5.7. Marcos

Antiguamente casi todas las páginas contenían marcos, porque son una manera fácil de crear diferentes áreas de navegación.

La separación entre cabecera, pie, menú e información se hacía mediante marcos. Sin embargo, la aparición de las hojas de estilo CSS los ha relegado a segundo plano.

Con las hojas de estilo podemos conseguir diseños más flexibles y variados, además tienen ciertos inconvenientes que los han relegado casi por completo.

Un documento con marcos no tiene sección `body`, en su lugar tiene una sección `<frameset>`, y una sección `<noframes>` en caso de no poder visualizar los marcos.

#### **`<frameset>`**

Definición: divide la ventana en rectángulos (marcos) independientes y redimensionables.

Aparición: etiquetas inicial y final obligatorias.

Atributos: `%coreattrs`, `rows`, `cols`, `onload`, `onunload`.

- `rows`: especifica los marcos horizontales mediante una lista de valores separados por comas. Los valores son longitudes expresadas en px o porcentajes.
- `cols`: especifica los marcos verticales de igual forma.

#### **`<frame>`**

Definición: define el contenido y apariencia de un marco.

Aparición: sin etiqueta de cierre (V).

Atributos: `%coreattrs`, `longdesc`, `name`, `src`, `frameborder`, `marginwidth`, `marginheight`, `noresize`, `scrolling`.

- `name`: nombre del marco.
- `longdesc`: URI a una descripción larga.
- `src`: contenido inicial del marco (un documento HTML, una imagen, etc).
- `frameborder`: 0 no incluye borde, 1 incluye borde.
- `marginwidth`, `marginheight`: es el padding del marco.
- `noresize`: es un booleano para impedir que la ventana sea redimensionable.
- `scrolling`: sirve para incluir una barra de desplazamiento (auto | yes | no).

Cada marco es una ventana o página independiente con sus propiedades y con una URI diferente. Esto supone una diferencia apreciable frente a un diseño hecho mediante capas, donde todas las capas pertenecen a la misma página. Los marcos pueden dificultar la navegación porque dejan sin utilidad a los botones de documento previo (back) y documento siguiente (forward), ya que ambos nos trasladarán fuera del documento con marcos.

Ejemplo:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN"
    "http://www.w3.org/TR/html4/frameset.dtd">
<html>
  <head>
    <title>Marcos</title>
  </head>
    <!-- Definición de marcos -->

    <frameset rows="50, 300, 50">
      <frame src="Cabecera.html">
      <frameset cols="10%, 90%">
        <frame src="Menu.html">
        <frame src="Informacion.html">
      </frameset>
      <frame src="Pie.html">
    </frameset>
</html>
```

Se construye de forma similar a una tabla con 3 filas y 2 columnas en la fila central.

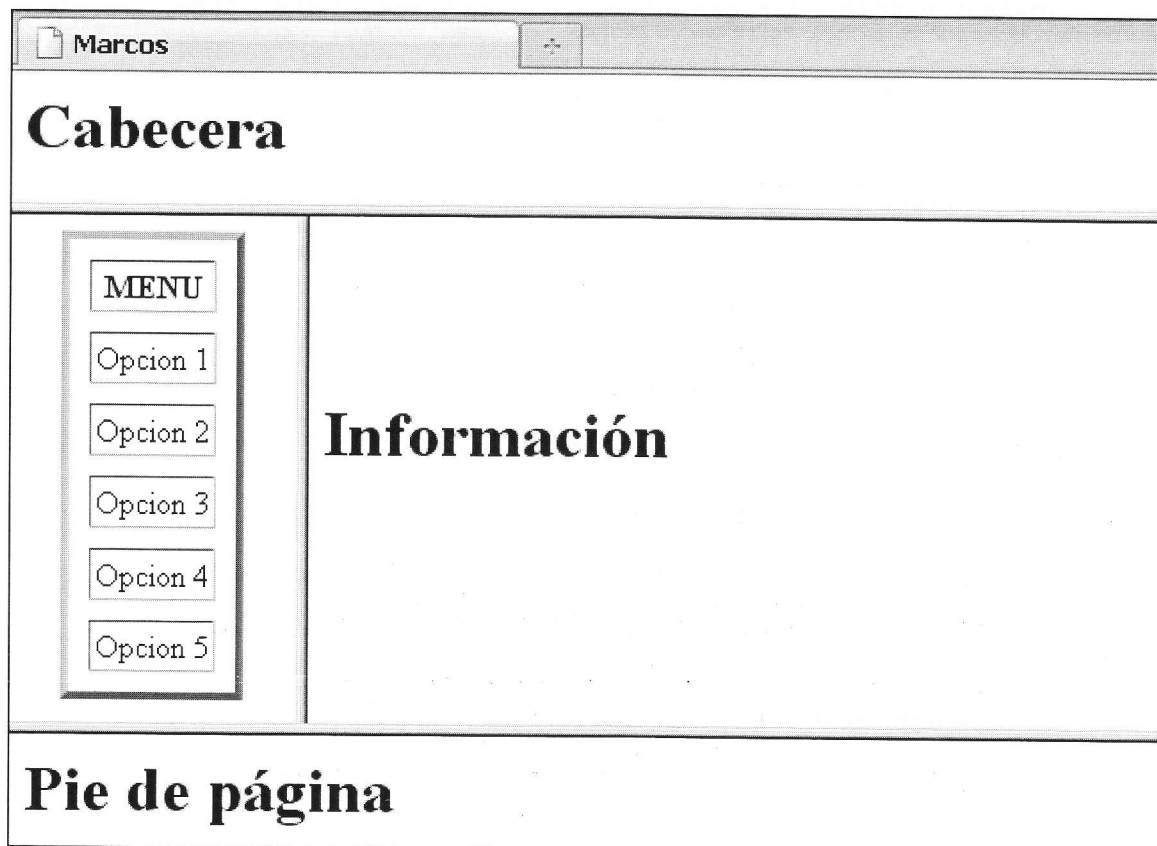


Figura 2.19: Diseño de página a 2 columnas mediante marcos

## 2.5.8. Objetos Multimedia

### <iframe>

Definición: inserta un marco en línea dentro de un documento, es equivalente a insertar una página dentro de otra con el tamaño establecido.

El iFrame es un objeto que se manipula con la misma libertad que cualquier otro elemento dentro del documento. Se suele utilizar para publicidad o sitios de colaboración.

Aparición: etiquetas inicial y final obligatorias.

Atributos: `%coreattrs`, `longdesc`, `name`, `src`, `frameborder`, `marginwidth`, `marginheight`, `scrolling`, `width`, `height` (align desaprobado).

### <object>

Definición: inserta un objeto en el documento.

El tipo de objeto viene determinado por sus atributos y puede ser una imagen, un applet de Java, un vídeo, una animación flash, otro documento HTML, etc.

Aparición: etiquetas de apertura y cierre obligatorias.

Atributos: `%attrs`, `declare`, `classid`, `codebase`, `data`, `type`, `codetype`, `archive`, `standby`, `height`, `width`, `usemap`, `name`, `tabindex`.

- **declare:** es un booleano, cuando aparece indica que solo se trata de una declaración del objeto, por tanto, la instanciación debe producirse después en otra línea con object.
- **classid:** indica la URI donde está la implementación del objeto.
- **codebase:** indica la dirección base para completar los URIs de los demás atributos (por defecto es la misma que el documento).
- **data:** indica la URI donde están los datos del objeto.
- **type:** indica el tipo de datos especificados en data.

La sintaxis es `type="categoría/formato"`. Ejemplos: `audio/basic`, `audio/mp3`, `video/mpeg`, `video/quicktime`, `application/x-shockwave-flash`, etc.

- **codetype:** indica el tipo de datos especificados por classid.
- **archive:** es una lista de URIs (separada por espacios) donde hay archivos útiles para el objeto.
- **standby:** es un texto que presenta el navegador durante la carga de los datos.
- **height, width:** alto y ancho del objeto.
- **usemap:** indica al objeto que utilice el mapa de imágenes del cliente.
- **name:** nombre para designar el elemento.

## <param>

Definición: sirve para inicializar variables de objetos, que serán utilizadas en tiempo de ejecución.

Aparición: sin etiqueta de cierre (V).

Atributos: **id**, **name**, **value**, **valuetype**, **type**.

- **name**: indica el nombre de un parámetro de ejecución.
- **value**: indica el valor inicial del parámetro especificado en name.
- **valuetype**: indica el tipo de atributo **value** (data | ref | object).
- **type**: indica el tipo del recurso cuando **valuetype** = "ref".

Sintaxis:

```
<param name="nombre de parámetro" value="valor de parámetro">
```

La lista de parámetros depende del tipo de objeto, algunos son: **FileName** para el nombre del archivo, **quality** para la calidad de imagen, **wmode** para el fondo, **allowFullScreen** para pantalla completa, **swfversion** para la versión de Flash Player, etc.

Ejemplo 1: Insertar un video local.

```
<object type="video/mpeg" id="hamaca" width="300" height="200"
  data="videos/hamaca.mpg">
  <param name="FileName" value="videos/hamaca.mpg">
  <param name="quality" value="high">
</object>
```

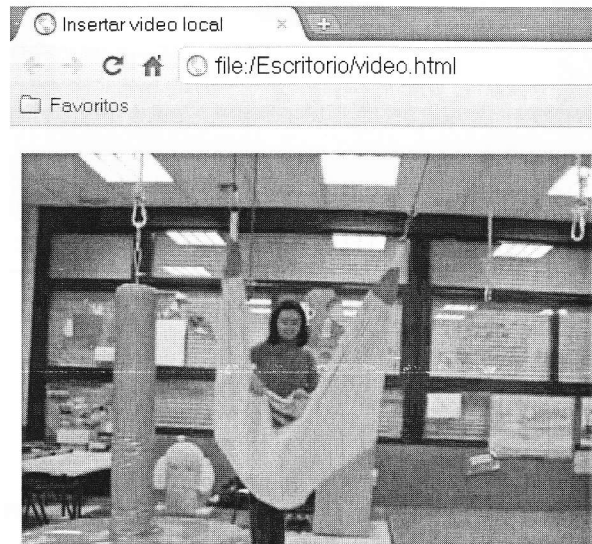


Figura 2.20: Insertar un video local

**Actividad 2.21:**

Descarga un archivo .swf de Internet, ajusta el ancho y alto, e insértalo en tu página.

---

Ejemplo 2: Insertar un video de YouTube®.

Se trata de archivos flash, por tanto, utilizamos **type** (application/x-shockwave-flash), en **data** especificamos la URI que aparece en YouTube.

```
<body>
  <h2>Un video de YouTube</h2>
  <object type="application/x-shockwave-flash"
    data="http://www.youtube.com/v/en0EfNXmL6M?version=3"
    width=425 height=350>
  </object>
</body>
```



Figura 2.21: Insertar un video de YouTube

El código que proporciona YouTube:

```
<iframe width="420" height="315"
  src=http://www.youtube.com/embed/en0EfNXmL6M
  frameborder="0" allowfullscreen>
</iframe>
```

Ejemplo 3. Insertar un plugin social.

Vamos a insertar un botón “Me gusta” de facebook®.

En <http://developers.facebook.com/>

Iniciamos sesión en facebook DEVELOPERS, Social Plugins, Like Button.

En el paso 1, introducimos los datos para configurar el botón a nuestro gusto: la URL de nuestra página, el diseño con o sin contador, ancho, perfiles, color, fuente, etc.

Por último, debemos elegir el tipo de etiquetas con las que se genera el código, para HTML 5, para XFBML o con `<iframe>`. Elegimos la última que es la más simple y la más compatible con versiones anteriores de los navegadores.

El código que nos proporciona la herramienta:

```
<iframe
  src="//www.facebook.com/plugins/like.php?href=http://www.juanm
  acr.es/index.html&send=false&layout=standard&width
  =450&show_faces=false&action=like&color=scheme=ligh
  t&font&height=80"scrolling="no" frameborder="0"
  style="border:none; overflow:hidden; width:450px;
  height:80px;" allowtransparency="true">
</iframe>
```

En el paso 2, vamos a obtener las etiquetas `<meta>` que vinculan nuestra web con facebook, se trata de las `<meta property>` que ya vimos anteriormente. Introducimos los datos asociados a la página (o el objeto en general) como el título, tipo, URL, imagen, etc.

Ejemplo:

```
<meta property="og:url" content="http://www.juanmacr.es" />
<meta property="og:title" content="Sitio Web de ayuda al
  estudiante de ciclos de informática" />
```

Añadimos el espacio de nombres como atributo de html:

```
xmlns:og="http://ogp.me/ns#"
```

El resultado estándar, después de publicarlo, sería:

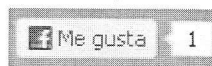


Figura 2.22: Botón Me gusta

De forma similar podemos agregar el botón +1 de Google.

En <http://www.google.com/webmasters/+1/button/>

Rellenamos el formulario de opciones para configurar el botón, y debemos elegir de nuevo las etiquetas para generar el código, válido para HTML 5 o como en este ejemplo:

```
<div class="g-plusone" data-annotation="inline"></div>
<script>...</script>
```

El resultado sería:

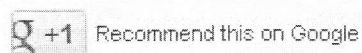


Figura 2.23: Botón +1

Ahora vamos con los botones de Twitter.

En Twitter developers, <https://dev.twitter.com/>

Elegimos uno de los botones, por ejemplo, Twittear:

Rellenamos las opciones de configuración y copiamos el código que debemos pegar en la página donde aparecerá el botón.


Opciones de botones	Previsualización y código
Compartir <input checked="" type="radio"/> Usar el URL de la página URL <input type="radio"/> <input type="text" value="http://"/>	Prueba tu botón, luego copia y pega el código de abajo en el código HTML de tu sitio.  Twittear 40.7K
Texto del Tweet <input checked="" type="radio"/> Usar el título de la página <input type="radio"/> <input type="text" value="Visita este sitio"/>	<pre>&lt;a href="https://twitter.com/share" class="twitter-share-button" data-lang="es"&gt;   Twittear &lt;/a&gt; &lt;script&gt; ... &lt;/script&gt;</pre>
<input checked="" type="checkbox"/> Mostrar el contador	

Figura 2.24: Opciones de Botón twittear

```
<a href="https://twitter.com/share" class="twitter-share-button"
  data-lang="es">
  Twittear
</a>
<script> ... </script>
```

El resultado sería:

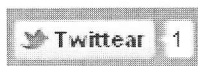


Figura 2.25: Botón Twittear



### 2.5.9. Formularios

Inicialmente HTML estaba diseñado para que el usuario recibiera datos del servidor, pero no para que el usuario enviara datos al servidor, con el fin de solucionar esta deficiencia se diseñaron los formularios.

Un formulario es un área del documento en la que insertamos elementos de entrada de información, llamados comúnmente controles de formulario, que permiten introducir datos, marcar opciones, elegir una opción de un grupo, etc.

Es conveniente indicar que el formulario solo sirve para recabar información del usuario, pero el tratamiento posterior de esta información lo hace normalmente un programa externo al documento llamado CGI (Common Gateway Interface).

Estos programas realizan diversas tareas como:

- Verificar y manipular los datos.
- Almacenar los datos en un archivo o en una base de datos.
- Reenviar los datos por correo electrónico.
- Leer los datos y devolver algún resultado al usuario.

Hay varias formas de construir estos programas, pero la más habitual es utilizar un lenguaje de programación para crear un script en el servidor, los lenguajes más utilizados son C, Perl, Java, ASP y sobretodo PHP.

#### **<form>**

Definición: sirve para insertar un formulario en el documento.

Aparición: etiquetas inicial y final obligatorias.

Atributos: `%attrs, action, method, enctype, accept, accept-charset, name, target, onsubmit, onreset`.

- **action:** indica la URI del programa encargado de tratar los datos del formulario.  
(Por ejemplo: `action="registro/registroCliente.php"`)
- **method:** indica el método usado para pasar los datos al programa (get | post).
- **enctype:** indica el tipo de contenido usado para enviar los datos.  
(Solo para `method="post"`).
- **accept:** es una lista de tipos de contenido aceptados por el servidor.  
(Sirve para filtrar archivos de tipos no aceptables).
- **accept-charset:** codificación de caracteres usada para los datos.
- **onsubmit:** sirve para ejecutar alguna acción cuando el formulario es enviado.
- **onreset:** sirve para realizar alguna acción cuando el formulario es reseteado.

**<Form>** es un elemento de bloque que puede contener a su vez a los elementos:

```

{
  <input>
  <button>
  <select> <option> <optgroup>
  <textarea>

```

## <input>

Definición: sirve para insertar un elemento, definido por 'type', dentro del formulario.

Aparición: sin etiqueta de cierre (V).

Atributos: %attrs, type, name, value, size, maxlength, checked, src, alt, disabled, readonly, usemap, ismap, tabindex, acceskey, onfocus, onblur, onselect, onchange, accept (align desaprobado).

- **type**: indica el tipo de control de formulario.  
(text | password | checkbox | radio | submit | reset | file | hidden | image | button).
- **name**: nombre del control.
- **size**: establece el ancho del control en pixels o en caracteres.
- **maxlength**: establece el nº máximo de caracteres para control de tipo texto o password.
- **checked**: marca la casilla para un control de tipo checkbox o radio.
- **src**: indica la URI donde se encuentra la imagen para type = "image".
- **disabled**: este booleano deshabilita el control y no se puede introducir datos.
- **readonly**: este booleano impide que el usuario haga cambios en el control.
- **accept**: lista de tipos mime válidos para subir ficheros.
- **value**:
  - para cuadros de texto es el valor inicial que aparece en el recuadro
  - para botones de tipo radio y checkbox es el valor que se envía al servidor
  - para botones de acción es el título del botón

Ejemplo:

```

<form action="manejador.html" enctype="multipart/form-data"
      method="post">
  USUARIO:
  <input type="text" name="usuario" value="user"> <br>
  CONTRASEÑA: <input type="password" name="pass"> <br>
  SOCIO: <input type="checkbox" name="socio" checked> <br>
  SEXO: <input type="radio" name="genero" value="m"> Masculino
        <input type="radio" name="genero" value="f"> Femenino <br>

```