



| | |
|---|--|
| FENW | miw.etsisi.upm.es • 2 |
| <h2>Índice</h2> | |
| <ul style="list-style-type: none"> • Introducción <ul style="list-style-type: none"> ◦ Etiquetas básicas ◦ Estilos físicos ◦ Estilos Lógicos • Listas <ul style="list-style-type: none"> ◦ Listas numeradas ◦ Listas no numerada ◦ Listas descriptivas • Imágenes y enlaces <ul style="list-style-type: none"> ◦ Incorporación de imágenes ◦ Enlaces externos e internos • Tablas <ul style="list-style-type: none"> ◦ Filas y celdas ◦ Bordos y agrupamientos ◦ Características de las celdas | <ul style="list-style-type: none"> • Formularios <ul style="list-style-type: none"> ◦ Elementos Input ◦ Select y textarea ◦ Nuevos elementos HTML5 • Otras etiquetas <ul style="list-style-type: none"> ◦ Video ◦ Audio ◦ Atributos data-* |

HTML (*HyperText Markup Language*)

■ ¿Qué es HTML?

- Es un lenguaje para la publicación y estructuración de documentos y especificación de hipervínculos
- Basado en **marcas**
- Una **marca** afecta a un fragmento de texto asignándole un formato, estructura o un significado determinado
- Un documento HTML contiene tanto la información que se desea presentar, como instrucciones (marcas) para describir su presentación
- Para visualizar documentos HTML se emplea un software denominado **agente de usuario, navegador o browser**
- Es un lenguaje de **hipertexto**, ya que dentro del documento existen áreas sensibles (hipervínculos) que al activarlas permiten acceder a otros elementos
- No se requieren editores especiales, ni intérpretes especiales

Introducción : HTML

■ Documentación:

- HTML: especificación 4.01 <http://www.w3.org/TR/html401/>
- HTML: especificación 4.01 (español) <http://html.conclase.net/w3c/html401-es/cover.html>
- Tutorial HTML: <http://html.conclase.net/tutorial/html/>
- Tutoriales de HTML, CSS, JavaScript: <http://www.w3schools.com/>
- Libros gratuitos sobre WEB: <http://www.librosweb.es/>
- Validador HTML4.01: <http://validator.w3.org/>

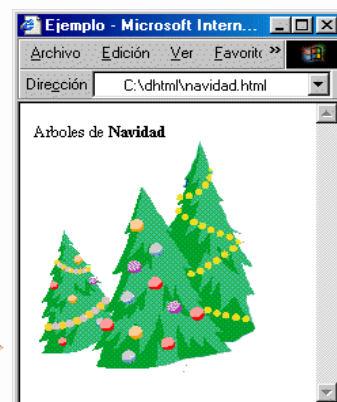
Introducción : HTML5

- Documentación manejada:
 - Web Hypertext Application Technology Working Group (WHATWG) estándar de HTML5
<http://www.whatwg.org/specs/web-apps/current-work/multipage/>
 - W3C estándar del lenguaje HTML5 <http://dev.w3.org/html5/spec>
 - Validador W3C de HTML <http://validator.w3.org>
 - Validador WHATWC de HTML5 <http://html5.validator.nu/>
 - Tutoriales de HTML, CSS, JavaScript <http://www.w3schools.com/>
 - Demos y Ejemplos <http://html5demos.com/>
 - Libros gratuitos sobre WEB <http://www.librosweb.es/>
 - Manual de HTML5
<https://www.ibm.com/developerworks/web/library/wa-html5fundamentals/>
 - W3C estándar de API,s relacionadas con HTML5
<http://dev.w3.org/html5/>
 - WebStorage <http://dev.w3.org/html5/webstorage/>
 - Geolocation Working Group <http://www.w3.org/2008/geolocation/>

HTML (*HyperText Markup Language*)

- Documento HTML y su visualización a través de un navegador

Árboles de
`Navidad`
``



HTML (*HyperText Markup Language*)

■ Antecedentes de HTML (I)

- Tim Berners-Lee en el European Laboratory for Particle Physics (CERN) impulsó su desarrollo junto con el de HTTP (**HyperText Transfer Protocol**) y URLs (**Uniform Resource Locators**)
- **HTML level 0** (1990): principalmente permitía definir párrafos, encabezados, saltos de línea, imágenes, vínculos y listas de elementos
- **HTML level 1** (1992): se definió la estructura actual del documento HTML y se añadieron las etiquetas de negrita e itálica y formularios elementales
- **HTML 2** (1994): desarrollado por el Word Wide Web Consortium (W3C nació en 1994) definió los elementos de formulario INPUT, SELECT y TEXTAREA y redefinió los vínculos, listas, imágenes y títulos

HTML (*HyperText Markup Language*)

■ Antecedentes de HTML (II)

- **HTML 3** (1995): fue sólo un borrador. Principalmente introdujo las tablas, los atributos de alineamiento y las imágenes de fondo
- **HTML 3.2** (1997): introdujo la posibilidad de incluir scripts y hojas de estilo en cascada (CSS) y formalizó el uso de colores en los fondos, textos y enlaces, así como la definición del tamaño y alineamiento de imágenes
- **HTML 4** (1998): añadió la división de la ventana en paneles (frames) junto a la posibilidad de incrustar distintos objetos. También definió los contenedores DIV y SPAN que unidas a CSS permiten definir nuevas etiquetas
- **HTML 4.01** (1999): incorpora leves modificaciones

HTML (*HyperText Markup Language*)

- Bajo el nombre de **HTML5** se agrupan tanto la última versión del lenguaje de marcado HTML, como un conjunto de tecnologías relacionadas con las aplicaciones Web modernas accesibles por medio de API's en JavaScript.
 - Nuevos elementos semánticos como `<section>`, `<article>`, `<footer>`...
 - Visualización de video sin necesidad de un plug-in
 - Elementos y atributos adicionales para el manejo de formularios
 - Dibujo sobre un lienzo (`<canvas>`) por medio de Javascript
 - Geolocalización (**Geolocation**)
 - Almacenamiento en el cliente de información persistente (**WebStorage**)
 - Aplicaciones **Offline** que funcionan sin necesidad de estar conectado a internet
 - **Microdata** para crear tu propio vocabulario semántico
 - ...
- HTML5 es soportado por los navegadores modernos

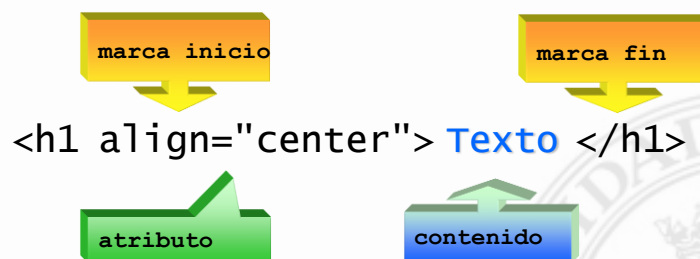
HTML (*HyperText Markup Language*)

- **Metodología de trabajo (I)**
 - Escribir el documento HTML en el editor de textos
 - Guardarlo en modo texto (ASCII), con extensión `'.htm'` o `'.html'`
 - Ejecutar el 'browser' (Firefox, Internet Explorer, Chrome, Opera, Safari, ...)
 - Abrir desde el browser el documento HTML para ver su aspecto
 - Modificar el documento HTML tantas veces como se necesite, salvarlo y volverlo a observar desde el visualizador

Marcas, elementos y atributos (I)

- Todas las marcas comienzan con el carácter `<` y acaban con `>`
- Las marcas pueden tener **atributos** para personalizarlas
- Los atributos se forman con el nombre del atributo seguido del signo `=` y seguido de un valor entrecomillado (p.ej. ``)
- Los elementos HTML suelen estar formados por una marca de inicio, un contenido y, opcionalmente, una marca de final que empieza por `</` (``)
- Algunos elementos HTML sólo tienen la marca de inicio y se llaman elementos vacíos (`
`)

Marcas, elementos y atributos (II)



FENW miw.etsisi.upm.es • 13

Estructura básica de un documento (I)

```

<!DOCTYPE html>
<html>
  <head>
    <title>Document Structure</title>
    "Información general del documento"
  </head>
  <body>
    HELLO WORLD...
  </body>
</html>

```

FENW miw.etsisi.upm.es • 14

HTML5: Estructura básica de documento

```

<!DOCTYPE html>
<html lang="es">
  <head>
    <meta charset="utf-8">
    <title>...</title>
  </head>
  <body>
    ...
  </body>
</html>

```

- Las etiquetas pueden ir en mayúsculas o minúsculas
- Los elementos sin etiqueta de cierre pueden llevar el slash o no (`
` o `
`)
- Las comillas en los valores de los atributos son opcionales
- El codificación del juego de caracteres puede ser otra como `"iso-8859-1"`

Estructura básica de un documento (II)

- En primer lugar se indica la versión de HTML utilizada en el documento por medio de `DOCTYPE`
- La pareja de etiquetas `<html>...</html>` engloba la estructura del documento
- Dentro del documento existen dos secciones: la cabecera del documento y el cuerpo del mismo
- El elemento `head` contiene información general acerca del propio documento y su visualización, como por ejemplo el título (elemento `title`) de la barra de título de la ventana del navegador
- El elemento `body` contiene la información que presenta el documento

Elemento meta

- Se sitúa en la cabecera del documento para proporcionar información adicional del mismo
 - Tipo de codificación:


```
<meta charset="utf-8">
```
 - Para "refrescar" una página o redirigirla a otra:


```
<meta http-equiv="refresh" content="10; url=doc2.html">
```
 - ```
<meta http-equiv="expires"
 content="Sat, 15 Oct 2011 21:07:01 GMT">
```

 indica la fecha en la que expira la página. Indicando `content="-1"` la página no se cachea.
  - ```
<meta name="Generator" content="software editor">
```

 informa del editor con el que se ha creado la página.
 - ```
<meta name="Author" content="Lucía Palacios">
```

 indica el autor de la página. El valor de `name` puede ser `keywords`, `date`, `description`, `copyright`, `robots`, ... Para almacenar en `content` información de palabras clave, descripción, ...

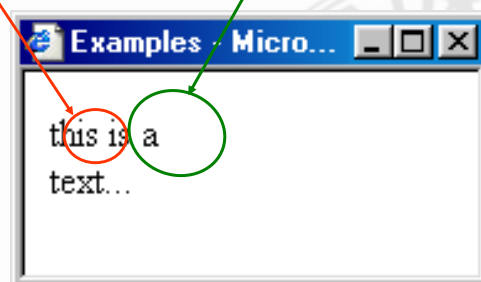


## Párrafos, saltos de línea y espacios

- Los saltos de línea presentes en el documento HTML no se representan
- Tampoco se representan los espacios sobrantes entre palabras, se comprimen a un único espacio
- La longitud de las líneas vienen definidas por el tamaño de la ventana del navegador
- La etiqueta `<br />` se utiliza para producir un salto de línea y retorno de carro. Este elemento no tiene etiqueta de cierre
- La elemento `<p>texto</p>` permite definir párrafos cuyo contenido es *texto*
- La marca `<p>` dispone del atributo en **desuso** `align` que permite definir el alineamiento del texto

## Párrafos, saltos de línea y espacios

```
<html>
...
<body>
 this is a
 text...
</body>
</html>
```



FENW miw.etsisi.upm.es • 19

## Texto preformateado

```
<html>
...
<body>
 This is normal text

 <pre>and this is preformatted</pre>
</body>
</html>
```

FENW miw.etsisi.upm.es • 20

## Comentarios

- En los documentos HTML se pueden incluir comentarios que ayuden a la comprensión de los mismos por parte de los desarrolladores
- Estos comentarios no son visualizados en el navegador
- Comentario de una línea:
 

```
<!-- Comentario de una línea -->
```
- Comentario de varias líneas:
 

```
<!-- Comentario de
.....
varias líneas -->
```

## Caracteres especiales

- Entidades carácter:
  - `<`; representa `<`
  - `>`; representa `>`
  - `&`; representa `&`
  - `é`; representa `é`
  - `€`; representa `€`
  - ; representa espacio en blanco
  - `ñ`; representa la letra ñe
  - Otros muchos ...
- Referencias numéricas:
  - `#ddd`; carácter número `ddd` en decimal
  - `#xddd`; carácter número `ddd` en hexadecimal

## Caracteres especiales

```
<html>
```

• • •

<body>

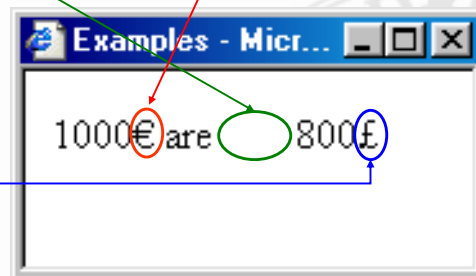
1000&euro;

`are&nbsp;&nbsp;&nbsp;&`

800&pound;

&lt;/body&gt;

&lt;/html&gt;



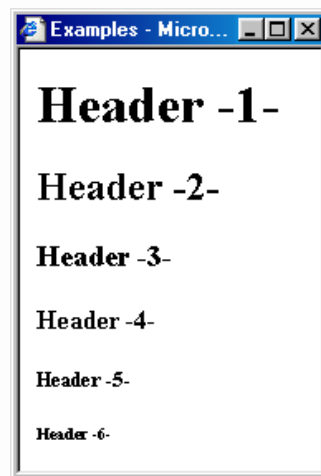
## Encabezados

- Los documentos extensos se suelen estructurar en apartados, secciones... cada una de las cuales puede llevar un encabezado
- Se pueden definir 6 niveles de encabezamiento ó títulos, desde **h1** (encabezado mayor) a **h6** (encabezado menor). Es decir, `<h1>texto</h1>`, `<h2>texto</h2>`, ..., `<h6>texto</h6>`
- Se visualizan en un párrafo independiente, con un fuente de mayor tamaño y en negrita
- El atributo en **align**, que está en **desuso**, permite definir el alineamiento del elemento por medio de los valores **right**, **left**, **justify** o **center**  
Ej: `<h1 align="center">encabezado centrado </h1>`

## Encabezados

...

```
<h1>Header -1-</h1>
<h2>Header -2-</h2>
<h3>Header -3-</h3>
<h4>Header -4-</h4>
<h5>Header -5-</h5>
<h6>Header -6-</h6>
```



## Longitudes absolutas y relativas

- La altura y anchura de diversos elementos HTML se pueden expresar de forma absoluta o relativa:
  - Forma absoluta: se indica la longitud en píxeles de la medida en cuestión (ej.: `width="200"`)
  - Forma relativa: se indica la longitud de la medida como un porcentaje del espacio horizontal o vertical disponible (ej.: `width="50%"`)

## Etiquetas básicas (obsoletas)

- Tanto las etiquetas de estilos físicos como las que se corresponden con los lógicos deben ser [sustituídos por el uso de CSS](#)
- Físicos: Cambian el aspecto visual del texto afectado:
  - `<b> texto </b>` El contenido se visualiza en **negrita**
  - `<i> texto </i>` El contenido se visualiza en *itálica*
  - `<big> texto </big>` El texto afectado se visualiza con un tamaño de fuente mayor que el valor por defecto
  - `<u> texto </u>` El texto afectado se visualiza subrayado
  - etc...
- Lógicos: describen el uso del texto afectado, definiendo uso específico
  - `<em> texto </em>` Se utiliza para resaltar o enfatizar algún texto
  - `<strong> texto </strong>` Se utiliza para resaltar todavía más un texto
  - `<cite> texto </cite>` El texto afectado es una referencia a otras fuentes
  - `<address> texto </address>` El etxto es una dirección
  - etc...

FENW miw.etsisi.upm.es • 27

## Estilos Físicos

...

```

the <i>formula</i> is:
2¹⁰

and the <u>result</u> is:
1024

```

...

The diagram illustrates the visual rendering of the HTML code. A browser window titled 'Examples - Mi...' shows the text: 'the *formula* is 2<sup>10</sup> and the result is **1024**'. Colored dotted lines connect the code to the rendered text: a red line connects '<i>' to the italicized 'formula'; a blue line connects '<sup>' to the superscript '10'; a green line connects '<u>' to the underlined 'result'; and a yellow line connects '<b>' to the bolded '1024'.

FENW miw.etsisi.upm.es • 28

## Fuentes

- La elemento `<font> texto </font>` (en **desuso**) permite personalizar la fuente aplicada al texto afectado por medio de los atributos:
  - `size` especifica el tamaño de la fuente utilizando los valores de 1 (menor) a 7 (mayor). También es posible definir el tamaño relativo al valor actual por medio de +x ó -x (Ej: `size="+2"` indica un tamaño 2 veces superior al actual en la escala de 1 a 7)
  - `color` especifica el color de la fuente
  - `face` especifica la familia de la fuente (`face="verdana"`)

FENW miw.etsisi.upm.es • 29

## Fuentes

```

...

 John Carpenter

 C.I.A. - Dept.Invest.

...

```

Examples - Microsoft Int...

**John Carpenter**  
C.I.A. - Dept.Invest.

FENW miw.etsisi.upm.es • 30

## Colores en un documento

- En HTML existen distintos elementos que pueden personalizarse indicando un color
- Existen dos formas para especificar un color:
  - Por medio de su nombre en inglés ("blue", "red"...)
    - Indicando su composición (de 0 a 255) de rojo, verde y azul que forman el color, expresando estos valores en formato hexadecimal según el formato #RRGGBB (`color="#FF0000"` es rojo). Por lo tanto, cada color componente puede tomar un valor entre (00 y FF)

## Más sobre etiquetas

- Anidamiento de etiquetas: cuando una etiqueta se incluye dentro del segmento afectada por otra hablamos de *anidamiento de etiquetas*. Las etiquetas se deben cerrar en orden inverso al que se han abierto. En definitiva, lo correcto es que un elemento HTML esté contenido dentro de otro elemento HTML
- Omisión de etiquetas: algunas etiquetas no precisan cierre, pero se recomienda hacerlo ya que así se facilita la transición hacia XHTML
- Etiquetas ignoradas: cuando un navegador no reconoce una etiqueta, simplemente la ignora

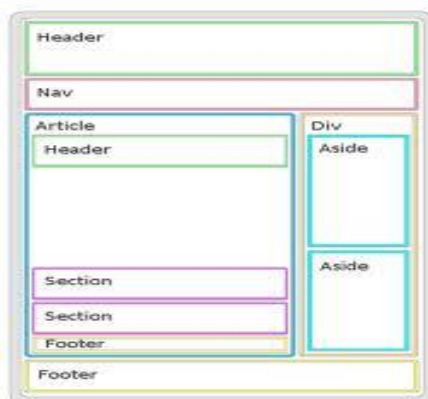
## Atributos genéricos

- Son atributos que poseen todos los elementos a excepción de **base**, **basefont**, **head**, **html**, **meta**, **param**, **script**, **title** y **style**. Estos atributos son:
  - **id**: se utiliza para asignar un nombre único o identificador a un elemento. Este nombre suele emplearse en DHTML
  - **style**: especifica información de estilo para el elemento actual. Es estilo estará definido mediante CSS
  - **class**: asigna un nombre de clase o un conjunto de nombres de clase separados por coma al elemento en cuestión. La clase estará definida mediante CSS
  - **title**: proporciona información consultiva del elemento HTML al que pertenece. En algunos navegadores, al posicionar el cursor sobre el elemento aparece una caja de texto con el contenido especificado en este atributo



## HTML5: Elementos semánticos estructurales

- HTML5 incorpora elementos nuevos que permiten definir la estructura del documento. Estos elementos nuevos dotan de cierta semántica al documento ya que describen el significado de su contenido y han surgido después de identificar patrones comunes en millones de páginas Web.



## HTML5: Elementos semánticos estructurales

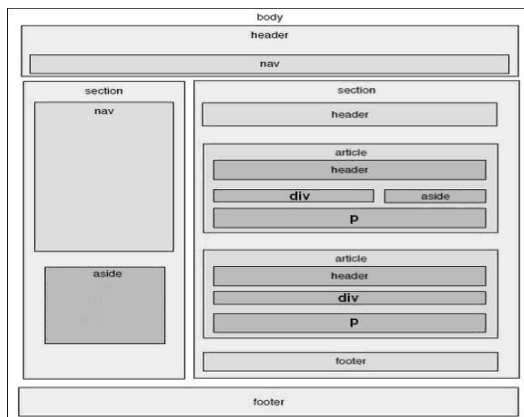
- <section>** (sección): Representa una sección genérica o una agrupación temática de los contenidos, normalmente con un título. *Ejemplos:* capítulos, las distintas páginas con pestañas en un cuadro de diálogo con fichas, las secciones enumeradas de una tesis. Se utiliza bien para agrupar artículos **<article>** en diferentes temas, bien para definir las diferentes secciones de un solo artículo. Es típico que dentro disponga de encabezado (**<header>**) con elementos **h1** - **h6**. En definitiva agrupa contenido relacionado.
- <article>** (artículo): Contiene una pieza independiente de contenido, que puede ser distribuido o reutilizado en otro sitio Web. *Ejemplos:* elemento RSS, una noticia, un post en un foro, entrada en un Blog, artículo en revista o periódico, un widget interactivo. Es típico que disponga de la información relativa al autor y fecha de publicación. En artículos anidados, los elementos del artículo interior representan los artículos relacionadas con los contenidos del artículo exterior. Por ejemplo, los comentarios acerca de un post pueden ser artículos anidados.
- <header>** (cabecera ó encabezado): Es un elemento que sirven como introducción o elemento para la navegación. Suele situarse en la parte superior de elemento que lo contiene aunque puede ir en otro lado. Una página puede tener una única cabecera o múltiples encabezados en cada una de sus secciones o artículos. También se usa para mostrar tabla de contenidos, formularios de búsqueda y logos del sitio.

## HTML5: Elementos semánticos estructurales

- **<footer>** (pie): Este elemento deberá contener información sobre quien escribió el contenido de la sección, artículo, elemento **<aside>**, página, información de copyright, enlaces al contenido relacionado, licencia,... Si el pie contiene apéndices, entonces puede dividirse en distintas secciones. Su situación física suele ser al final del elemento, pero puede ir al principio si por ejemplo contiene el autor de un post.
- **<nav>**: Contiene la funcionalidad principal de navegación para acceder a partes de la propia página o a otras páginas. No todos los grupos de enlaces deben ir dentro de este elemento, sólo los principales. Normalmente, no se utiliza este cuando está contenido en **<footer>** y muy a menudo está contenido dentro de un elemento **<header>**.
- **<aside>**: Define un bloque de contenido relacionado con el contenido principal que lo rodea, pero que no es esencial para el flujo del mismo. El elemento **<aside>** se debe utilizar para un contenido que esté relacionado tangencialmente. Si se tiene un contenido que se considera que debe estar separado del contenido principal, entonces es adecuado utilizar el elemento **<aside>**. Pregúntese si el contenido que hemos colocado dentro de **<aside>** puede ser eliminado y esta acción no resta o altera el significado del contenido principal del documento o de la sección. Se suele utilizar para contenido que va enmarcado, para publicidad, una encuesta, para grupos de elementos de navegación y en general para todo el contenido que es considerado como separado del contenido principal.

## HTML5: Elementos semánticos estructurales

- Se pueden realizar distintas composiciones con los elementos semánticos estructurales.

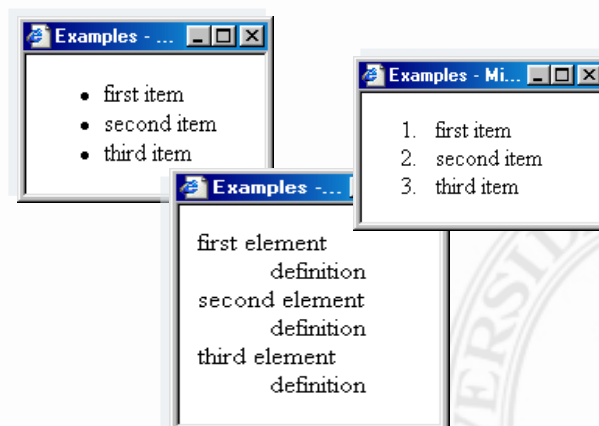


## Listas (I)

- Presentan la información como una sucesión de ítems
- Es una forma bastante común de organizar y representar información (la lista de la compra, los pasos de una receta, diccionario de términos)
- En HTML existen tres tipos de listas:
  - Listas de elementos no numerados
  - Listas de elementos numerados
  - Listas de definiciones o glosarios

## Listas (II)

- Visualización de los tres tipos de listas:



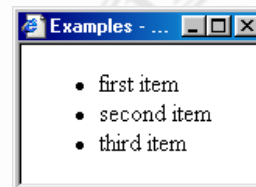
## Listas no numeradas (I)

- Es una sucesión de párrafos cada uno de los cuales va precedido de un símbolo denominado viñeta o "bullet". El comienzo y final de la lista va marcado respectivamente con las etiquetas `<ul>` y `</ul>`
- Cada elemento de la lista va marcado respectivamente por las etiquetas `<li>` y `</li>`

```
...

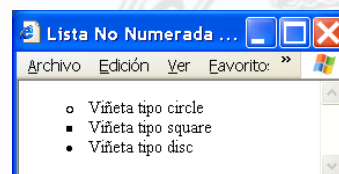
 first item
 second item
 third item

...
```



## Listas no numeradas (II)

- Modificación de la viñeta:
  - El atributo `type` puede modifica el aspecto del "bullet". Puede tomar los valores `circle`, `square` o `disc`
  - Este atributo se puede aplicar a toda la lista si se sitúa en el tag `<ul>` o un elemento en concreto si se sitúa en el tag `<li>` correspondiente
- Ítems de más de un párrafo: se consiguen insertando párrafos o saltos de línea dentro del elemento `<li>`



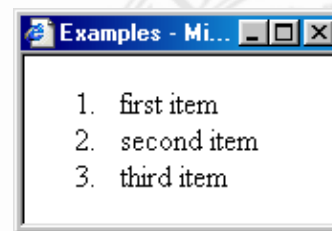
## Listas numeradas (I)

- Es una sucesión de párrafos cada uno de los cuales va precedido de un símbolo de numeración (número o letra) que indica el orden del mismo. El comienzo y final de la lista va marcado respectivamente con las etiquetas `<ol>` y `</ol>`.
- Cada elemento de la lista va marcado respectivamente por las etiquetas `<li>` y `</li>`

```

 first item
 second item
 third item

```



## Listas numeradas (II)

- Modificación del símbolo de numeración:
  - El atributo `type` modifica el símbolo de numeración. Puede tomar los valores `1`, `a`, `A`, `i` ó `I` que indican numeración decimal, alfabética en minúscula, alfabética en mayúscula, romana en minúscula y romana en mayúscula
  - Este atributo se puede aplicar a toda la lista si se sitúa en el tag `<ol>`, o a un elemento en concreto si se sitúa en el tag `<li>` correspondiente
- Modificación del orden de la numeración:
  - El número asignado al atributo `start` del tag `<ol>` define el valor de comienzo de la numeración de la lista
  - El número asignado al atributo `value` del tag `<li>` define por qué número continúa la numeración de la lista

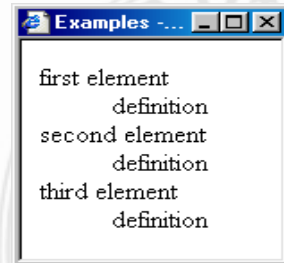
## Listas Descriptivas o Glosarios

- Son listas de términos y sus definiciones asociadas. El comienzo y final de la lista va marcado respectivamente con las etiquetas `<dl>` y `</dl>`
- El término aparece entre los tags `<dt>` y `</dt>`
- La descripción del término va entre `<dd>` y `</dd>`

```
<dl>
 <dt>first element</dt>
 <dd>definition</dd>

 <dt>second element</dt>
 <dd>definition</dd>

 <dt>third element</dt>
 <dd>definition</dd>
</dl>
```



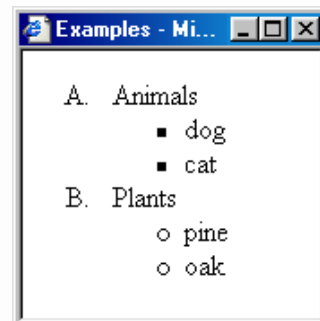
## Listas anidadas

- Son listas en las que uno de sus ítems incluye otra lista

```
<ol type="A">
 Animals
 <ul type="square">
 dogcat

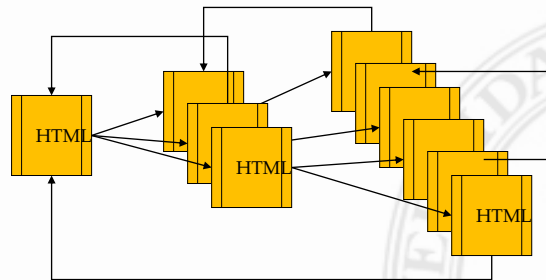
 Plants
 <ul type="circle">
 pineoak


```



## Hipertexto

- Jakob Nielsen en su libro "Hypertext and Hypermedia" definió **hipertexto** como:
  - "Hypertext is non-sequential writing: a directed graph, where each node contains some amount of text or other information"



## Enlaces o Hipervínculos (I)

- Permiten que el usuario se conecte directamente a otros recursos en cualquier parte del mundo o a otras partes del propio recurso



- Se realizan por medio del elemento **a**:  
`<a href="URLdestino">pulse el enlace</a>`

ancla

## Enlaces o Hipervínculos (II)

- El elemento **a** permite definir enlaces. Tiene las siguientes características:
  - Requiere etiqueta de inicio y cierre `<a>...</a>`
  - El contenido del elemento se denomina **ancla**. Si es texto se visualiza -por defecto- en color azul y subrayado
  - Posee el atributo **href** que indica la dirección del documento o recurso con el que se quiere enlazar
  - La dirección del documento o recurso se expresa por medio de los **URL** (*Uniform Resource Locator*)

```
acceso a UPM
```

## URL

- Permiten indicar la ubicación exacta de un recurso en internet y la forma de acceder al mismo
- Un recurso puede permitir el acceso y visualización de un documento HTML, el acceso a un cliente de correo electrónico, el acceso a un fichero situado en un servidor de ftp, etc...
- Cada servicio de Internet tiene su esquema de acceso

```
http://www.w3.org/Addressing/
```

```
http://rfc.net/rfc2396.html (URI: Generic Syntax)
```



## Estructura de los URLs

- Esquema de acceso:

`http://servidor:puerto/ruta/recurso?querystring`

- `http`: es el protocolo utilizado para la comunicación
- `servidor`: puede ser el nombre de una máquina servidora o su dirección IP
- `puerto`: es el número de puerto donde escucha el servidor (por defecto es el 80 para HTTP).
- `ruta`: indica la localización del documento en la máquina servidor
- `recurso`: es el nombre del recurso concreto
- `querystring`: permite enviar información al recurso

## URL relativos

- Son URLs que proporcionan la dirección de un recurso de forma abreviada
- Cada documento HTML tiene una **dirección base** formada por `http://servidor:puerto/ruta`. La URL relativas llevan implícita esta información
- La **dirección base** se puede modificar por medio del elemento `<base href="url_base">` que se sitúa en la sección **head** del documento

## Ejemplos de URL relativos

- Dado el documento cuyo URL es:

☞ `"http://www.myserver.com/docs/mydocs/mypage.html"`

- El URL relativo `"otherpage.html"` se expande a:

`"http://www.myserver.com/docs/mydocs/otherpage.html"`

- El URL relativo `"other/other.html"` se expande a:

`"http://www.myserver.com/docs/mydocs/other/other.html"`

- El URL relativo `"../..../other.html"` se expande a:

`"http://www.myserver.com/other.html"`

## Otros tipos de URLs

- URL file

- `file:///unidad|/ruta/archivo`

- URL ftp

- `ftp://usuario:contraseña@servidor:puerto/ruta/archivo;`  
`type=codigo_tipo`

- URL de tipo mailto:

- `mailto:direccióndecorreo?subject=texto del subject`

- URL de tipo nntp

- `nntp://servidordenews:puerto/grupodenoticias/artículo`

- URL javascript

- `javascript:código javascript`

## Vínculos a partes del propio documento

- En un documento se pueden definir puntos de destino empleando el atributo **id** o **name**.  
`<a name="salto">...</a>` ó `<a id="salto">...</a>`
- Para referirse a estos puntos de salto se indica su nombre precedido del carácter **#**. Ej: `<a href="#salto">...</a>`

```
down

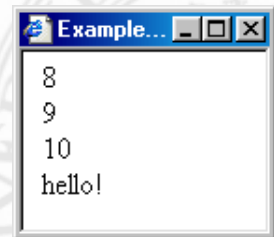
1

2

3

...
10

hello!
```



## Imágenes en los documentos

- El lenguaje HTML no sólo maneja información en formato textual, sino que también permite incorporar imágenes, sonido, vídeo... permitiéndonos hablar de hipermedia en vez de hipertexto
- El elemento **img** permite incorporar imágenes a un documento
- El elemento **img** no posee etiqueta de cierre
- El elemento **img** dispone del atributo obligatorio **src** cuyo contenido es el URL de la imagen que se quiere incorporar
- La imagen estará en formato **gif**, **jpeg** ó **png**...

## Tamaño de una imagen

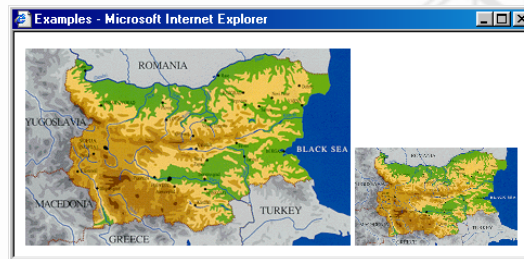
- Las imágenes se visualizan en el navegador con su tamaño original. Los atributos **width** y **height** del elemento **img** permiten definir un tamaño de imagen
- El atributo **alt** que toma como valor un texto, ofrece una descripción alternativa a la imagen

```

```

```

```



## Tablas

- Presentan la información en dos dimensiones distribuyéndola en cuadrículas que conforman filas y columnas
- Para qué se usan:
  - Permiten formatear datos tabulares (horario de autobuses, información extraída de una base de datos, etc.)
  - Componer y diseñar documentos:
    - Posicionar componentes
    - Alineamiento de distintos elementos
    - Disposición de textos en varias columnas

FENW miw.etsisi.upm.es • 57

### Ejemplo de tablas: calendario



| April 2003 |     |     |     |     |     |     |
|------------|-----|-----|-----|-----|-----|-----|
| Sun        | Mon | Tue | Wed | Thr | Fri | Sat |
|            |     | 1   | 2   | 3   | 4   | 5   |
| 6          | 7   | 8   | 9   | 10  | 11  | 12  |
| 13         | 14  | 15  | 16  | 17  | 18  | 19  |
| 20         | 21  | 22  | 23  | 24  | 25  | 26  |
| 27         | 28  | 29  | 30  |     |     |     |

FENW miw.etsisi.upm.es • 58

### Estructura básica de una tabla (I)

- Una tabla está definida por los 2 tags `<table>` y `</table>`. Entre estas dos etiquetas se encuentra el contenido de la tabla
- El contenido de la tabla se describe fila a fila. Cada fila está delimitada por los tags `<tr>` y `</tr>`
- El contenido de cada fila se define por celdas. Cada celda está delimitada por los tags `<td>` y `</td>`, o bien por las etiquetas `<th>` y `</th>`

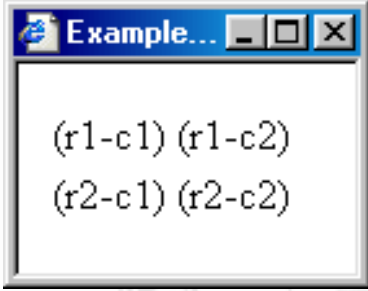
FENW miw.etsisi.upm.es • 59

### Estructura básica de una tabla(II)

```

...
<table>
 <tr>
 <td>(r1-c1)</td>
 <td>(r1-c2)</td>
 </tr>
 <tr>
 <td>(r2-c1)</td>
 <td>(r2-c2)</td>
 </tr>
</table>
...

```



FENW miw.etsisi.upm.es • 60

### Bordes de tablas

- Las tablas pueden mostrar un borde alrededor
- El atributo **border** define el borde de la tabla indicando el grosor en píxeles (Ej. **border=3**)
- El atributo **frame** permite personalizar la forma del borde indicando uno de estos valores:
  - **hsides**: lados superior e inferior
  - **vsides**: lados izquierdo y derecho
  - **lhs**: lado izquierdo
  - **rhs**: lado derecho
  - **box** y **border**: los cuatro lados
  - **above** y **below**: arriba y abajo.
  - **void**: ninguno (valor por defecto)

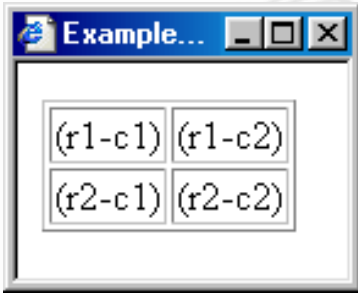
FENW miw.etsisi.upm.es • 61

### Ejemplo de tabla con bordes (I)

```

...
<table border="1">
 <tr>
 <td>(r1-c1)</td>
 <td>(r1-c2)</td>
 </tr>
 <tr>
 <td>(r2-c1)</td>
 <td>(r2-c2)</td>
 </tr>
</table>
...

```



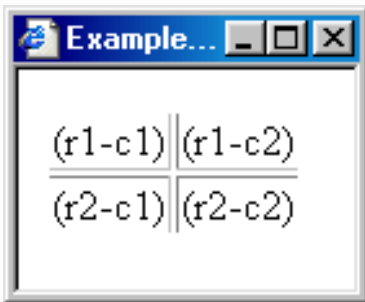
FENW miw.etsisi.upm.es • 62

### Ejemplo de tabla con bordes (II)

```

...
<table border="1" frame="void">
 <tr>
 <td>(r1-c1)</td>
 <td>(r1-c2)</td>
 </tr>
 <tr>
 <td>(r2-c1)</td>
 <td>(r2-c2)</td>
 </tr>
</table>
...

```



### Líneas de división de una tabla

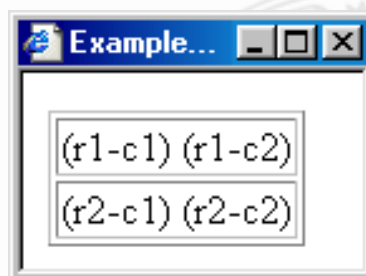
- También es posible hacer visibles las líneas de división entre las celdas de la tabla
- El atributo **rules** permite hacerlo dándole alguno de siguientes valores:
  - **none**: Ninguna línea de división (valor por defecto)
  - **groups**: Sólo aparecerán líneas de división entre grupos de filas (ver THEAD, TFOOT y TBODY) y grupos de columnas (ver COLGROUP y COL)
  - **rows**: Sólo aparecerán líneas de división entre filas
  - **cols**: Sólo aparecerán líneas de división entre columnas
  - **all**: Aparecerán líneas de división entre todas las filas y columnas

### Ejemplo de líneas de división de una tabla

```

...
<table rules="rows">
 <tr>
 <td>(r1-c1)</td>
 <td>(r1-c2)</td>
 </tr>
 <tr>
 <td>(r2-c1)</td>
 <td>(r2-c2)</td>
 </tr>
</table>
...

```





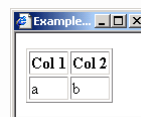
## Definición de celdas

- Las celdas de una tabla se pueden definir por medio de los elementos **td** y **th**
- Estos elementos tienen la etiqueta de cierre opcional
- El elemento **td** muestra el contenido de las celdas alineado a la izquierda
- El elemento **th** muestra el contenido de las celdas centrado y en negrita

## Definición de celdas

### ◦ Ejemplo de **th** y **td**

```
<table border="1">
 <tr>
 <th>Col 1</th>
 <th>Col 2</th>
 </tr>
 <tr>
 <td>a</td>
 <td>b</td>
 </tr>
</table>
```

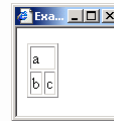


| Col 1 | Col 2 |
|-------|-------|
| a     | b     |

### Celdas que ocupan varias columnas

- El atributo **colspan** de una celda permite que su contenido se expanda en horizontal a varias celdas, es decir, que ocupe varias columnas
- Este atributo toma como valor un número entero positivo superior a 1

```
<table border="1">
 <tr>
 <td colspan="2">a</td>
 </tr>
 <tr>
 <td>b</td>
 <td>c</td>
 </tr>
</table>
```




|   |   |
|---|---|
| a |   |
| b | c |

### Celdas que ocupan varias filas

- El atributo **rowspan** de una celda permite que su contenido se expanda en vertical a varias celdas, es decir, que ocupe varias filas
- Este atributo toma como valor un número entero positivo superior a 1

```
<table border="1">
 <tr>
 <td rowspan="2">a</td>
 <td>b</td>
 </tr>
 <tr>
 <td>c</td>
 </tr>
</table>
```



|   |   |
|---|---|
| a | b |
|   | c |

## Agrupamiento de filas

- Las filas de una tabla pueden agruparse en una cabecera de tabla, un pie de tabla, y una o más secciones de cuerpo de tabla, usando los elementos `thead`, `tfoot` y `tbody` respectivamente.
- Cada uno de estos elementos pueden tener atributos que asignan propiedades a todas las celdas que forman el grupo:

## Ejemplo de agrupamiento de filas

```
<table border="1">
 <thead><tr>
 <td>Header A</td>
 <td>Header B</td>
 </tr></thead>
 <tfoot><tr>
 <td>Foot A</td>
 <td>Foot B</td>
 </tr></tfoot>
 <tbody><tr>
 <td>Body A</td>
 <td>Body B</td>
 </tr></tbody>
</table>
```

`tfoot` debe aparecer antes que la sección `tbody`

|          |          |
|----------|----------|
| Header A | Header B |
| Body A   | Body B   |
| Foot A   | Foot B   |

## HTML5: Otros semánticos estructurales

- **<hgroup>** (grupo de encabezados): Se utiliza para incluir más de un encabezado (h1...h6) si se desea que cuente como un único encabezado en la estructura de encabezados de la página (outline). En <http://code.google.com/p/h5o/> existe una herramienta JavaScript para visualizar la estructura de encabezados de una página (outline).
- **<figure>** (figura): Se debe utilizar para marcar ilustraciones, diagramas, fotos, código de programas,... a los que hace referencia el contenido principal del documento, pero que pueden situarse en otra parte (a un lado, en un apéndice, en otras páginas) sin afectar al flujo del documento
- **<figcaption>** (título o pie de figura): Es un elemento que debe estar contenido en el elemento **figure** al principio o al final permitiendo definir su título o pie.
- **<mark>** (remarca): Se debe utilizar para remarcar una parte de un documento debido a su interés para la actividad actual del usuario. El ejemplo típico es la recuperación con google de un documento en donde aparecen *remarcadas* las palabras con las que se realizó la búsqueda. La diferencia con **strong** y **em** es que estos elementos se utilizan para remarcar contenido en cualquier contexto.
- **<progress>** (progreso): Se debe usar para describir el estado actual de un proceso cambiante. Un ejemplo típico de uso es la barra de progreso de una descarga de información. Tiene dos atributos opcionales: **value** que representa con un valor numérico el estado actual de la tarea y **max** que indica el punto en que la tarea ha terminado.

## HTML5: Otros semánticos estructurales

- **<meter>** (medidor): representa una medida escalar en un rango definido (con un valor máximo y mínimo), o un valor fraccionario como por ejemplo, el uso de un disco, la relevancia de la respuesta a una consulta, la fracción de votos que ha obtenido un candidato. No se debe usar para expresar un peso, una altura o una edad ya que normalmente se desconoce su valor máximo. Tiene 6 atributos:
  - **min** establece el límite inferior del rango
  - **max** establece el límite superior del rango
  - **value** establece valor actual del rango
  - **low** establece umbral a partir del cual se considera un valor bajo del rango
  - **high** establece umbral a partir del cual se considera un valor alto del rango
  - **optimum** establece el valor que se considera óptimo del rango
- **<time>** (fecha y hora): Se usa para marcar tiempos y fechas. La información enmarcada no tiene porque ser una fecha válida, podría ser "el próximo jueves". Cuenta con 2 atributos:
  - **datetime** establece la fecha, y opcionalmente la hora, correspondiente con la información contenida en un formato no ambiguo. Si no se utiliza este atributo, entonces la información enmarcada por **<time>** debería seguir un formato correcto: "2012-01-06" fecha, "2012-01-06T23:45:45.015Z" fecha y hora con milisegundos(UTC), "2012-01-06T23:45:45.015-02:00" fecha y hora con milisegundos con el desplazamiento de zona horaria
  - **pubdate** indica que la fecha es de publicación del artículo antecesor más cercano y si no existe un artículo antecesor entonces es la fecha de publicación del documento entero

## Formularios

- Permiten al usuario introducir información por medio de cajas de texto, casillas de verificación, botones de opciones... para ser procesada en el servidor
- Esta información normalmente se usa en el servidor para generar contenidos dinámicos
  - Ejemplo: si se quieren saber los vuelos disponibles entre dos ciudades, será necesario indicar los nombres de las ciudades origen y destino, así como el periodo de consulta

## Ejemplo de formulario

Working with forms - Microsoft Internet Explorer

**Information Form**

|                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                            |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>Personal Data</b></p> <p>Name: <input type="text"/></p> <p>First name: <input type="text"/></p> <p>Last name: <input type="text"/></p> <p>Sex: <input type="radio"/> man <input type="radio"/> woman</p> <p>Birth: <input type="text"/>/ <input type="text"/>/ <input type="text"/></p> | <p><b>Address Data</b></p> <p>e-mail: <input type="text"/></p> <p>Address: <input type="text"/></p> <p>City: <input type="text"/></p> <p>Country: <input type="text"/> Zip: <input type="text"/></p> <p>Code: <input type="text"/></p>                                                                                                     |
| <p><b>Comments and Uploads</b></p> <p>Insert your comments here</p> <p><input type="text"/> Examiner...</p>                                                                                                                                                                                   | <p><b>Interest Areas</b></p> <p><input type="checkbox"/> HTML <input type="checkbox"/> DHTML <input type="checkbox"/> JavaScript</p> <p><input type="checkbox"/> ASP <input type="checkbox"/> PHP <input type="checkbox"/> ColdFusion</p> <p><input type="checkbox"/> Java <input type="checkbox"/> C++ <input type="checkbox"/> Basic</p> |

\* needed fields

## Definición de un formulario

- Se realiza por medio del elemento **form**
- El elemento **form** requiere de etiqueta de apertura y cierre (**<form>***definición\_de\_controles***</form>**)
- Dentro de la secuencia **<form> ...</form>** se describen elementos HTML y los controles propios del formulario (cuadros de texto, botones de opciones, casillas de verificación, etc.)
- Los formularios no se pueden anidar

## Atributos del elemento **form**

- **action**: define la URL donde se van a procesar los datos del formulario
- **method**: define el método de envío de la información del formulario al servidor. Puede tomar los valores **post** ó **get**
- **enctype**: define el formato de los datos transferidos indicando el tipo MIME
- **name**: define el nombre del formulario. Se utiliza en los scripts en el cliente

## Tipos de controles de formularios

- entradas
  - cajas de texto
  - cajas de texto de múltiples líneas
  - palabras clave (password)
  - ficheros (file)
- casillas de verificación (checkbox)
- casillas de elección (radio)
- botones
  - envío (submit)
  - reinicialización (reset)
  - pulsadores (button)
- menús de selección (select)

## Elementos para definir controles

- Elemento **input**: no tiene etiqueta de cierre. Permite definir varios controles indicando el valor de su atributo **type**:
  - cuadros de texto (**text**)
  - palabras clave (**password**),
  - botones de tipo **submit**, **reset** y **button**
  - casillas de verificación (**checkbox**)
  - casillas de elección (**radio**)
  - ficheros (**file**)
  - ocultas (**hidden**)
  - botones con imagen (**image**)
- Menús de selección (**select**)
- Cuadros de texto multilínea (**textarea**)

## Elemento **input** de tipo **text**

- Permite definir cajas de texto fijando el valor del atributo **type** al valor **text**.
- Atributos:
  - **name**: permite identificar este elemento
  - **value**: valor por defecto de caja de texto
  - **maxlength**: limita el número de caracteres que se pueden introducir en la caja de texto
  - **size**: limita la longitud visible del cuadro

```
<input type="text" name="entrada" size="15" maxlength="20">
```

## Elemento **input** de tipo **password**

- Es una caja de texto para introducir palabras reservadas, por lo que el texto tecleado se visualizará como una sucesión de asteriscos
- Los atributos son similares a los del elemento **input** de tipo **text**.



```
<input name="pclave" type="password" size="15" maxlength="15">
```



## Elemento **input** de tipo **submit**

- Define un botón de envío. Al pulsarlo provoca la transferencia de los datos del formulario a la URL especificada en el atributo **action** del mismo
- Atributos:
  - **name**: permite identificar este control
  - **value**: establece el título del botón

A small rectangular button with a light gray background and the text "Send Data" in a dark gray font.

```
<input type="submit" value="Send Data">
```

## Elemento **input** de tipo **reset**

- Permite definir un botón que al pulsar en él provoca que los controles del formulario vuelvan a recuperar sus valores por defecto.
- Atributos:
  - **name**: permite identificar este control
  - **value**: establece el título del botón

A small rectangular button with a light gray background and the text "Reset Data" in a dark gray font.

```
<input type="reset" value="Reset Data">
```

## Elemento **input** de tipo **button**

- Permite definir un botón genérico al que se puede asociar una acción utilizando algún lenguaje de scripting en cliente (javascript, vbscript)
- Atributos:
  - **name**: permite identificar este control
  - **value**: establece el título del botón

Push Me...

```
<input type="button" value="Push Me..."
onclick="alert('Hello World...')">
```



## Elemento **input** de tipo **checkbox**

- Permite definir una casilla de verificación. La casilla puede estar marcada o no. Pulsando sobre ella cambiará su estado.
- Atributos:
  - **name**: permite identificar este control
  - **value**: valor que se enviará al servidor si la casilla se encuentra marcada
  - **checked**: hace que la casilla aparezca inicialmente marcada

☐ HTML

```
<input type="checkbox" name="interest" value="1">HTML
```

## Elemento **input** de tipo **radio**

- Permite definir una casilla de elección
- Atributos:
  - **name**: permite identificar este control
  - **value**: valor que se enviará al servidor si la casilla se encuentra marcada
  - **checked**: hace que la casilla aparezca inicialmente marcada
- Cuando varias casillas de elección comparten el mismo nombre son mutuamente excluyentes

☐ man ☐ woman

```
<input type="radio" name="sex" value="m">man
<input type="radio" name="sex" value="w">woman
```

## Elemento **input** de tipo **image**

- Permite definir un botón gráfico que al pulsar en él provoca la transferencia de los datos del formulario a la URL especificada en el atributo **action** del mismo
- Atributos:
  - **src**: URL del gráfico que se va a visualizar
  - **alt**: texto alternativo al gráfico

## Elemento **input** de tipo **file**

- Permite definir un control para enviar ficheros al servidor junto con los datos del formulario
- El fichero seleccionado se enviará al servidor para ser tratado allí
- Atributos:
  - **name**: permite identificar este control



```
<input type="file" name="fichero"
accept="application/x-zip-compressed" >
```

## Elemento **textarea**

- Permite definir un cuadro de texto donde se pueden escribir las líneas que se deseen
- Requiere de etiqueta de inicio y fin  
`<textarea>contenido</textarea>`
- El *contenido* del elemento **textarea** es el valor inicial del cuadro de texto multilínea
- Atributos:
  - **name**: permite identificar este control
  - **rows**: número de líneas visibles
  - **cols**: número de caracteres visibles por línea

## Elemento `select`

- Permite definir menús de selección
- Este elemento tiene etiqueta de cierre obligatoria `<select>opciones</select>`
- Las opciones del menú de selección se indican por medio del elemento `option` que tiene etiqueta de cierre obligatoria (`<option>opción</option>`).



## Atributos de las opciones de un `select`

- Los elementos `option` tienen los siguientes atributos:
  - `value`: valor que se enviará al servidor si la opción se encuentra seleccionada
  - `selected`: indica que esta opción aparece seleccionada por defecto
- Si a ninguna opción se le asigna el atributo `selected`, la primera opción aparecerá seleccionada por defecto

FENW miw.etsisi.upm.es • 91

### Elemento **select**

```
<select name="country">
 <option value="E1">Spain</option>
 <option value="E2">France</option>
 <option value="E3">Italy</option>
 <option value="A1">Brazil</option>
 <option value="A2">Chile</option>
</select>
```

FENW miw.etsisi.upm.es • 92

### Agrupamiento de opciones de un **select**

- El elemento **optgroup** permite a los autores agrupar opciones lógicamente
- Este elemento tiene el atributo **label** que especifica el rótulo del grupo de opciones
- Los elementos **optgroup** no se pueden anidar

## Agrupamiento de opciones de un **select**

```
<select name="country" size="1">
 <option value="" selected="selected"></option>
 <optgroup label="Europe">
 <option value="E1">Spain</option>
 <option value="E2">France</option>
 <option value="E3">Italy
 </optgroup>
 <optgroup label="America">
 <option value="A1">Brazil</option>
 <option value="A2">Chile</option>
 </optgroup>
</select>
```



## Atributos del elemento **select**

### • Los atributos de este elemento son:

- **name**: permite identificar este control
- **multiple**: Indica que en el menú de selección se puede seleccionar más de una única opción
- **size**: Este atributo toma un valor numérico y si su valor es mayor que 1 transforma el menú de selección en una lista con desplazamiento ("scrolled list box") que tendrá visibles tantas opciones como indique dicho valor

## Atributos del elemento **select**

```
<select name="country" size="5" multiple="multiple">
 <option value="E1">Spain</option>
 <option value="E2">France</option>
 <option value="E3">Italy</option>
 <option value="A1">Brazil</option>
 <option value="A2">Chile</option>
</select>
```



## Elemento **label**

- permite la identificación de los elementos de un formulario
- requiere etiqueta de apertura y cierre
- asocia rótulos a los controles (campos de texto, casillas de verificación o radio y menús)
- atributo:
  - **for**: especifica el nombre del control asociado
- ejemplo:

```
<label for="usuario">Usuario:</label>
<input type="text" id="usuario">
<label for="pclave">P. Clave:</label>
<input type="password" id="pclave">
```



## Elemento **button**

- equivalente a elementos **input**, pero ofrece más posibilidades de representación
- tiene etiqueta de cierre obligatoria
- atributos:
  - **type**: especifica el tipo de botón (**submit** | **reset** | **button**)
  - **name**: asigna el nombre al control
  - **value**: valor inicial del control

## Agrupación de elementos en formularios

- El elemento **fieldset** (con etiqueta de cierre obligatoria) permite a los autores agrupar temáticamente controles
- Este elemento mostrará un recuadro alrededor de los controles que abarca
- Incorporando el elemento **legend** (con etiqueta de cierre obligatoria) en el contenido del elemento **fieldset** se asigna un título al recuadro
- El atributo **align** (**en desuso**) del elemento **legend** define la situación del título asignado (**left** o **right**)

FENW
miw.etsisi.upm.es • 99

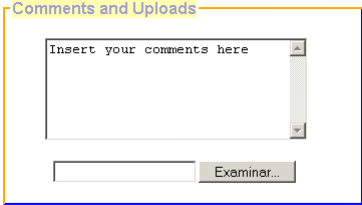
## Ejemplo agrupación de elementos

```

<fieldset>
 <legend>Comments and Uploads</legend>
 <textarea name="comments" cols="30" rows="6"
 wrap="virtual">Insert your comments here
 </textarea>

 <input accept="application/x-zip-compressed"
 type="file" name="file">
</fieldset>

```



FENW
miw.etsisi.upm.es • 100

## Dirigir el foco hacia un elemento

- Se puede dirigir el foco hacia un elemento por medio de los siguientes atributos:
  - **accesskey=tecla**: pulsando normalmente **Alt + tecla** se situará el foco en el elemento correspondiente. Los agentes de usuario suelen tener asociada alguna acción sobre ciertas teclas, por lo que hay que tener cuidado a la hora de escoger las teclas que se desean utilizar. Este atributo no está definido para los controles select.
  - **tabindex=Num**: se define un orden de navegación de los distintos elementos de un formulario y por medio de la tecla tabulador se pasa de uno a otro. El orden de navegación es ascendente según el valor asignado a **Num**.

## Controles deshabilitados y sólo lectura

- Se puede deshabilitar el acceso a un control por medio del atributo `disabled`
- Los controles deshabilitados no tienen acceso al foco ni son enviados al servidor
- Se pueden definir controles de sólo lectura por medio del atributo `readonly`
- Los controles de sólo lectura pueden recibir el foco y son enviados al servidor

## HTML5: Nuevos controles de formulario

- Los nuevos controles de la clase `<input>` son:
  - `type="search"` se usa para que el usuario introduzca una cadena con la que realizar una búsqueda. En Chrome y Safari cuando se introduce información aparece una "x" a la derecha del campo que si se pulsa elimina toda la cadena introducida.
  - `type="email"` se usa para que el usuario introduzca una cadena con formato de email. En Firefox, Chrome y Opera el formulario no es enviado si la cadena no tiene un formato correcto.
  - `type="url"` se usa para que el usuario introduzca una cadena con formato de url. En Firefox y Chrome el formulario no es enviado si la cadena no tiene un formato correcto. En Opera si no se pone el protocolo el navegador incluye el prefijo "http://"
  - `type="tel"` se usa para que el usuario introduzca una cadena con formato de teléfono.
  - `type="number"` se usa para definir una caja de texto numérica con unas flechas asociadas para incrementar y decrementar el valor. Funciona en Opera, Safari y Chrome y si se introduce una cadena no numérica el control se envía al servidor conteniendo la cadena vacía. Posee los atributos nuevos:
    - `min` valor mínimo del número
    - `max` valor máximo del número
    - `step` incremento/decremento del número al utilizar las flechas

## HTML5: Nuevos controles de formulario

- Los nuevos controles de la clase `<input>` son:
  - `type="date"` es un control para fecha en formato `aaaa-mm-dd`. En Opera se despliega un calendario para escoger una fecha pulsando en ella. En Chrome y Safari se visualiza como un campo de texto con flechas para subir y bajar. En Opera y Chrome se valida la fecha antes de ser enviada en Safari no. No funciona en IE y Firefox. Atributos nuevos:
    - `min` fecha mínima aceptable
    - `max` fecha máxima aceptable
    - `step` incremento/decremento de los días aceptables desde el valor mínimo.
  - `type="month"` es un control para fecha en formato `aaaa-mm`. Mismo comportamiento y atributos que `type="date"`.
  - `type="week"` es un control para fecha en formato `aaaa-W1-52`. Por ejemplo, 2007-W5 es la 5ª semana de 2007. Mismo comportamiento y atributos que `type="date"`.
  - `type="time"` es un control para introducir la hora en formato de 24 horas como por ejemplo 23:15:50. Se visualiza igual en Opera, Chrome y Safari y tiene el mismo comportamiento y atributos que `type="date"`.
  - `type="datetime"` y `type="datetime-local"` es un control para introducir la fecha hora en formato definido para la etiqueta `<time>`. Se visualiza igual en Opera, Chrome y Safari y tiene el mismo comportamiento y atributos que `type="date"`.

## HTML5: Nuevos controles de formulario

- Los nuevos controles de la clase `<input>` son:
  - `type="range"` es una barra deslizante (slider) que permite definir un número aproximado dentro de un intervalo. Un ejemplo de su utilización podría ser un cuestionario en donde una persona indica el grado de satisfacción o cumplimiento sobre un determinado tema. Funciona en Opera, Safari y Chrome. Posee los atributos nuevos:
    - `min` valor mínimo del número
    - `max` valor máximo del número
    - `step` incremento/decremento al mover la barra deslizante.
  - `type="color"` es un control diseñado para introducir un color. Solo funciona en Opera.
- El nuevo control `<keygen>` genera un par de claves. Cuando el formulario es enviado la clave privada se almacena en el navegador y la pública es enviada con el resto del formulario. No funciona en IE9 ni en Safari. Atributos:
  - `challenge` cuyo valor es empaquetado con la clave pública
  - `keytype` cuyo valor indica el algoritmo de encriptación (de momento solo "rsa")
- El nuevo control `<output>` mostrará un cálculo hecho con javascript a partir de otra información que será enviado con el resto del formulario. El típico ejemplo sería el total con IVA de una compra. No es soportado actualmente

## HTML5: Nuevos atributos en formularios

- **placeholder** es un atributo que permite visualizar una leve sugerencia sobre cómo debe ser el contenido del mismo. En cuanto el control adquiere el foco, la sugerencia desaparece de su contenido. No funciona en IE.
- **required** es un atributo que indica que el control que lo posee debe rellenarse correctamente: no se puede dejar vacío y debe cumplir con el formato del control ya sea el propio, ya sea el establecido por medio del atributo **pattern**. No es aceptado por `<button>`, `<range>`, `<color>` y `<hidden>`. Si se envía el formulario con un `submit()` de Javascript no se tiene en cuenta este atributo a no ser que se invoque al método `click()` de un botón de tipo **submit**.
- **pattern** permite definir el formato de la información que se introduce en el control por medio de una expresión regular al estilo de Perl con la salvedad que la cadena introducirá al completo debe concordar con dicha expresión regular. El control no es enviado al servidor hasta que haya concordancia entre la expresión y su contenido. Funciona con Firefox, Opera y Chrome. No permitido en el elemento **textarea**. Si el tipo de control (ej:email) entra en conflicto con el patrón, no habría forma de enviar el formulario
- **disabled** permite deshabilitar un control. En Firefox, Opera, IE se puede aplicar también al elemento `<fieldset>` para deshabilitar todos los controles que contenga.

## HTML5: Nuevos atributos en formularios

- **readonly** permite que el contenido de un control no pueda ser modificado
- **multiple** permite seleccionar varios ficheros (`type="file"`) o incluir varias direcciones de correo (`type="email"`) separadas por comas. No funciona en IE.
- **autocomplete** permite configurar si el control dispone de esta funcionalidad con los valores `"on"` y `"off"`. Este atributo se puede fijar en el elemento **form** para que afecte a todos los controles del mismo. Funciona en Firefox y Chrome.
- **autofocus** permite que el foco se sitúe en el control que posee este atributo nada más entrar en la página. No funciona en IE
- **form** este atributo permite asociar un control situado en un formulario con otro formulario cuyo **id** coincida con el valor del atributo. De esta manera cuando se envíe el formulario con el **id** indicado, se incluirá el control perteneciente al primer formulario. No funciona en IE.

## HTML5: Nuevos atributos en formularios

- **wrap** es un atributo de `<textarea>` que por defecto tiene el valor `"soft"`. Cuando se le asigna el valor `"hard"`, todos los retornos de carro introducidos por el navegador son incluidos en el contenido cuando el `<textarea>` es enviado al servidor
- **list** permite establecer una serie de valores para autocompletar el control que se activa en Firefox cuando se pulsa una letra y Opera cuando adquiere el foco. Va en combinación con `<datalist id="lista"><option value="valor1">...</datalist>` y el valor que debe tomar es el **id** del elemento `datalist`
- **novalidate** es un atributo de `<form>` que inhibe las validaciones nativas en sus controles (pattern, required, controles como email...). Funciona con Firefox, Opera y Chrome

## Elemento `iframe` (I)

- requiere etiqueta de apertura y cierre
- permite insertar un marco en línea dentro de un bloque de texto
  - similar al elemento `object`
- no pueden ser redimensionados
- atributos:
  - **src**: localización del contenido inicial del marco en línea
  - **name**: nombre del marco en línea
  - **width**: anchura del marco en línea
  - **height**: altura del marco en línea

FENW miw.etsisi.upm.es •

## Elemento `iframe` (II)

- ejemplo:
 

```
<IFRAME src="destino.html"
width="200"
height="200"
frameborder="0"
align="center">
 Texto alternativo al marco
</IFRAME>
```



109

FENW miw.etsisi.upm.es • 110

## Atributo `target`

- El atributo `target` se puede aplicar a los **enlaces** (elemento `a`) y a los **formularios** (elemento `form`) ó al elemento `base` para indicar cual es el marco de destino donde se cargará el documento solicitado
- El atributo `target` también puede tomar los valores:
  - `_blank`: indica que el documento se carga en una ventana nueva
  - `_self`: indica que el documento se carga en el mismo marco
  - `_parent`: indica que el documento se carga en el panel padre inmediato del marco actual
  - `_top`: indica que el documento se carga en la ventana original completa (desaparece la estructura de marcos)

## Elemento **object** (I)

- especifica todos los datos requeridos por un objeto para su representación en el cliente
- solución para la inclusión de futuros medios
- requiere etiqueta de apertura y cierre
- ejemplo:

```
<OBJECT
 codetype="application/java"
 classid="AudioItem"
 width="15" height="15">
 <PARAM name="sonido"
 value="Hola.au|Welcome.au">
 Applet que ejecuta un sonido de bienvenida
</OBJECT>
```

## Elemento **object** (II)

- si no es posible mostrar el objeto, se mostrará el contenido que aparezca entre las etiquetas
  - Es posible anidar objetos por orden de preferencia
- ejemplo:

```
<!-- objeto que muestra la tierra -->
<OBJECT title="La tierra vista desde el espacio"
 classid="http://www.observer.mars/LaTierra.py">
 <!-- sino, intenta el video MPEG -->
 <OBJECT data="LaTierra.mpeg" type="application/mpeg">
 <!-- sino, intenta la imagen GIF -->
 <OBJECT data="LaTierra.gif" type="image/gif">
 <!-- sino, muestra el texto -->
 La tierra vista desde el espacio
 </OBJECT>
 </OBJECT>
</OBJECT>
```



## Elemento `object` (III)

- atributos:
  - `classid`
    - URI del objeto a mostrar (p.e., un applet)
  - `data`
    - URI de los datos del objeto (p.e., una imagen)
  - `codetype`
    - tipo de contenido del objeto cuando se emplea `classid`
  - `type`
    - tipo de contenido del objeto cuando se emplea `data`
    - este atributo es opcional, pero recomendado
  - `standby`
    - mensaje que se muestra mientras se carga el objeto

## Elemento `object`(IV)

- ejemplos:
 

```
<!-- objeto que muestra una imagen -->
<OBJECT
 data="eui.gif" type="image/gif"
 width="183" height="166" standby="cargando imagen...">
 E.U. Informática
</OBJECT>

<!-- objeto que carga otro documento -->
<OBJECT
 data="doc1.html" type="text/html"
 width="130" height="80"
 standby="Cargando doc. HTML...">
 Otro doc. HTML
</OBJECT>

<!-- objeto que muestra un documento pdf-->
<OBJECT
 data="doc.pdf" type="application/pdf"
 style="width: 500px; height:550px;" >
 ATENCIÓN: No se puede mostrar el documento
</OBJECT>
```

## HTML5: Video

- Antes de HTML5 el video se incorporaba en las páginas por medio de plugins que se integraban en el navegador. Con HTML el video y el audio son manejados por el navegador e incluso pueden ser manipulados por medio de JavaScript
- El problema principal del trabajo con video es que cada navegador soporta unos determinados **formatos** de video con unos **codecs** concretos. En <http://www.mirovideoconverter.com> y <http://firefogg.com> hay herramientas de conversión de videos.

```
<video width="320" height="240" controls>
 <source src="movie.mp4" type="video/mp4" />
 <source src="movie.ogv" type="video/ogg" />
 <source src="movie.webm" type="video/webm" />
</video>
```

## HTML5: Video

- El elemento **video** tiene etiqueta de abertura y cierre y posee los siguientes atributos:
  - **src** que contendrá el fichero del video.
  - **width** anchura en pixels del contenedor del video. El video se escalará al valor mayor de **width** y **height** sin que se distorsione la relación original. El resto de espacio horizontal o vertical quedará sin ocupar.
  - **height** altura en pixels del contenedor del video
  - **controls** la presencia de este atributo hace que sean visibles los controles del reproductor.
  - **autoplay** la presencia de este atributo hace que el video se reproduzca el video en cuanto se pueda.
  - **loop** la presencia de este atributo hace que el video se reproduzca otra vez, cada vez que termine.

## HTML5: Video

- El elemento `video` también posee los siguientes atributos:
  - `preload` indica si el video y los metadatos asociados con el mismo son precargados. Toma los valores:
    - `auto` indica que el video y sus metadatos son descargados por el navegador antes de que se requiera su visualización.
    - `none` el video es cargado cuando se requiere su visualización.
    - `metadata` solamente se descargan los metadatos antes de que se requiera la visualización del video.
  - `poster` contiene una `url` de una imagen que se mostrará antes de comenzar la visualización del video
  - `audio` con el valor "muted" inhibe el sonido del video
- En el interior del elemento video pueden existir varios elementos source que definen distintos ficheros de video y su formato. Ej:

```
<source src="movie.mp4" type="video/mp4" />
<source src="movie.ogg" type="video/ogg" />
```

## HTML5: Audio

- HTML5 permite incorporar sonido en las páginas, de una forma análoga al video, por medio del elemento audio.
- `<audio>` puede incorporar los atributos `preload`, `autoplay`, `controls` y `loop` similares al elemento video

```
<audio controls autoplay loop preload="auto">
 <source src="horse.ogg" type="audio/ogg" />
 <source src="horse.mp3" type="audio/mp3" />
 Your browser does not support the audio element.
</audio>
```