

# 4. HTML

---

Tecnologías en el lado del cliente:

HTML y XHTML

CSS

Javascript



# HTML y XHTML

---

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
  <meta name="TITLE" content="Title of the document" />
  <meta name="KEYWORDS" content="Keywords" />
  <meta name="DESCRIPTION" content="Description" />
  <link rel="stylesheet" type="text/css" href="style.css" />
  <script language="JavaScript" src="script.js" />
</head>
<body bgcolor="#ffffff" />
```

# HTML

## Etiquetas

---

- La gran ventaja de HTML5 es que estandariza el Document Object Model (DOM)
- Lenguaje basado en marcas o etiquetas
- La sintaxis de las etiquetas es:  

```
<etiqueta [atributo1 = "valor1" atributo2 = "valor2" ...]>
```
- Las partes de la etiqueta que van entre "[ ]" son opcionales.
- Un elemento está formado por una etiqueta de apertura (a veces opcional) un contenido (opcional) y una etiqueta de cierre (a veces opcional)  

```
<etiqueta>[SEP][contenido del elemento][SEP]</etiqueta>
```
- Una etiqueta siempre va entre ángulos "< ... >"
- Opcionalmente, puede tener un conjunto de atributos y valores

# HTML

## Estructura

---

- Todo el contenido de un documento HTML se encuentra entre las etiquetas `<html>` `</html>`.
- Cada página se divide a su vez en:
  - ❑ Cabecera: entre las etiquetas `<head>` `</head>`
  - ❑ Cuerpo: delimitado entre las etiquetas `<body>` `</body>`
- Toda la información que se incluye en la cabecera no se renderiza al visualizar el documento y se considera "meta información" (con la excepción del título del documento).
- Es **obligatorio** (salvo en `iframes` y algún caso más) incluir dentro de la cabecera de la página web un título, empleando la etiqueta `<title>`. Es el título que nuestro navegador web muestra en su barra.

# HMTL

## Div (divisiones o bloques)

---

- Las páginas web actuales se crean mediante diferentes bloques `<div>` anidados.
- Habitualmente se trata de un contenedor donde añadimos otros elementos.
- Se modifican mediante CSS para dar forma a la estructura de la página web
- Desde HTML5 disponemos de etiquetas semánticas para los bloques en función de su uso (cabeceras, pie, sección, etc.)

# HTML

## Estructura

---

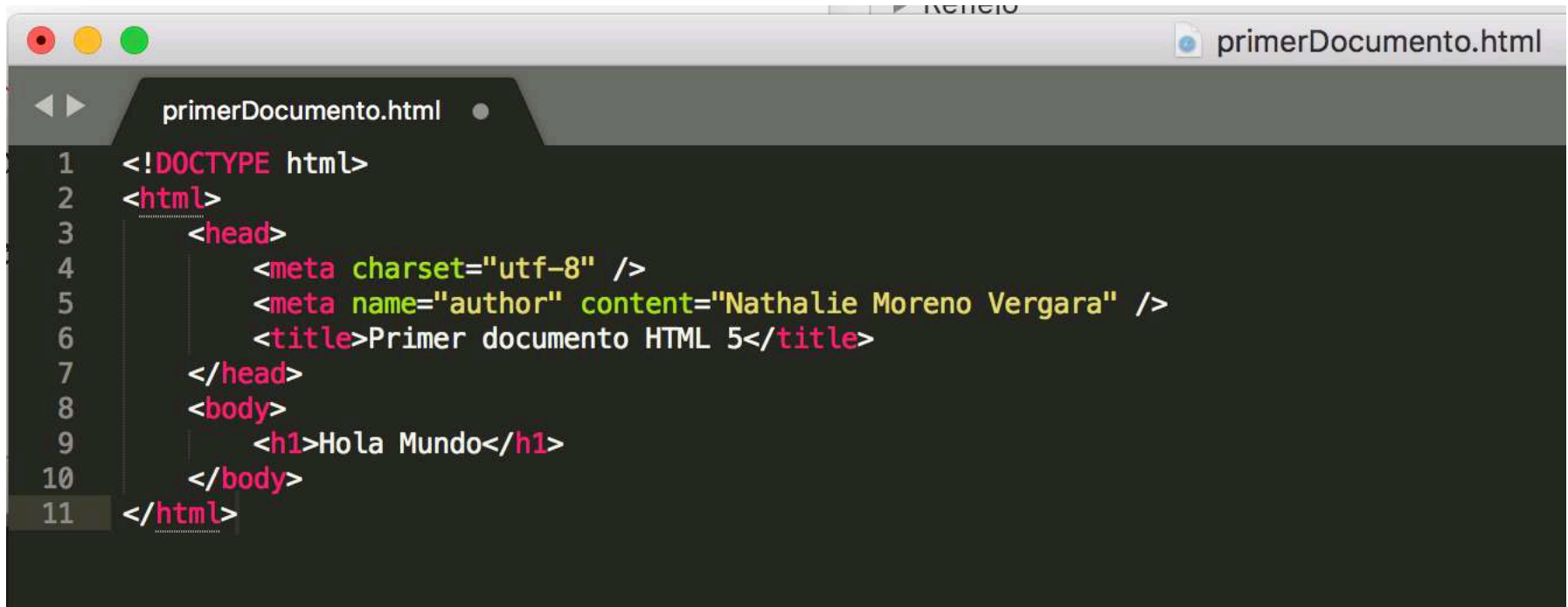
- Algunos elementos nuevos en HTML 5 permiten organizar mejor el contenido de las páginas:
  - Elemento **SECTION** (<section> ... </section>): identifica una sección del documento
  - Elemento **ARTICLE** (<article>...</article>): artículos, entradas de blog, noticias, etc.
  - Elemento **HEADER** (<header>...</header>): identifica la cabecera de una sección
  - Elemento **FOOTER** (<footer>...</footer>): identifica el pie de una sección
  - Elemento **NAV** (<nav>...</nav>): identifica los enlaces de navegación
  - Elemento **ASIDE** (<aside>...</aside>): todo aquellos que es accesorio
  - Elemento **ADDRESS** (<address>...</address>): información de contacto
  - Elemento **FIGURE** (<figure>...</figure>): identifica un objeto "flotante"
  - Elemento **FIGCAPTION** (<figcaption>...</figcaption>): texto acompañando al "flotante"
  - Elemento **SMALL** (<small>...</small>): letra pequeña de un documento

# HTML

## Ejemplo

---

- Primer ejemplo:



The image shows a code editor window titled "primerDocumento.html". The editor displays the following HTML code:

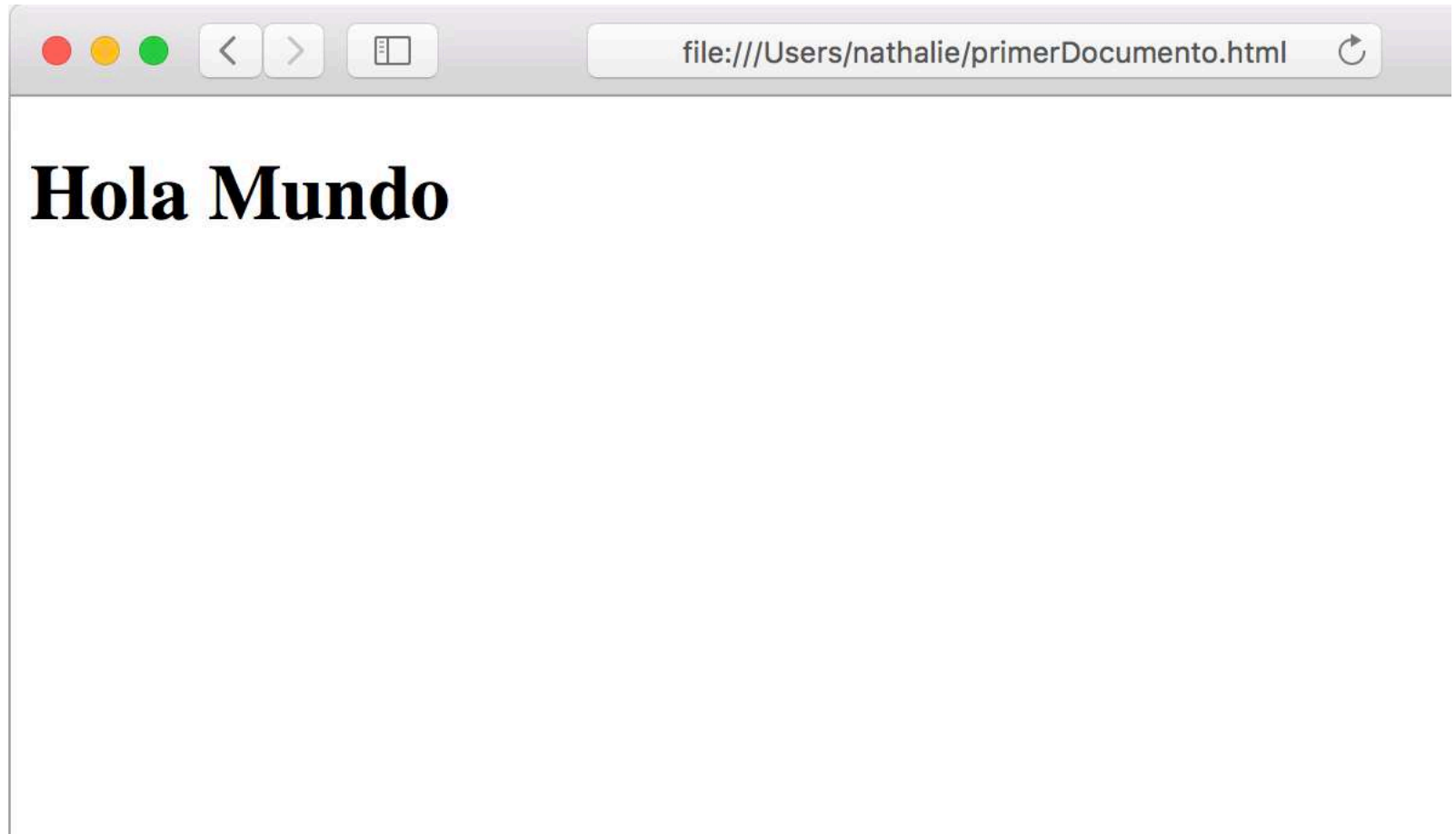
```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <meta charset="utf-8" />
5     <meta name="author" content="Nathalie Moreno Vergara" />
6     <title>Primer documento HTML 5</title>
7   </head>
8   <body>
9     <h1>Hola Mundo</h1>
10  </body>
11 </html>
```

# HTML

## Ejemplo

---

Primer ejemplo:





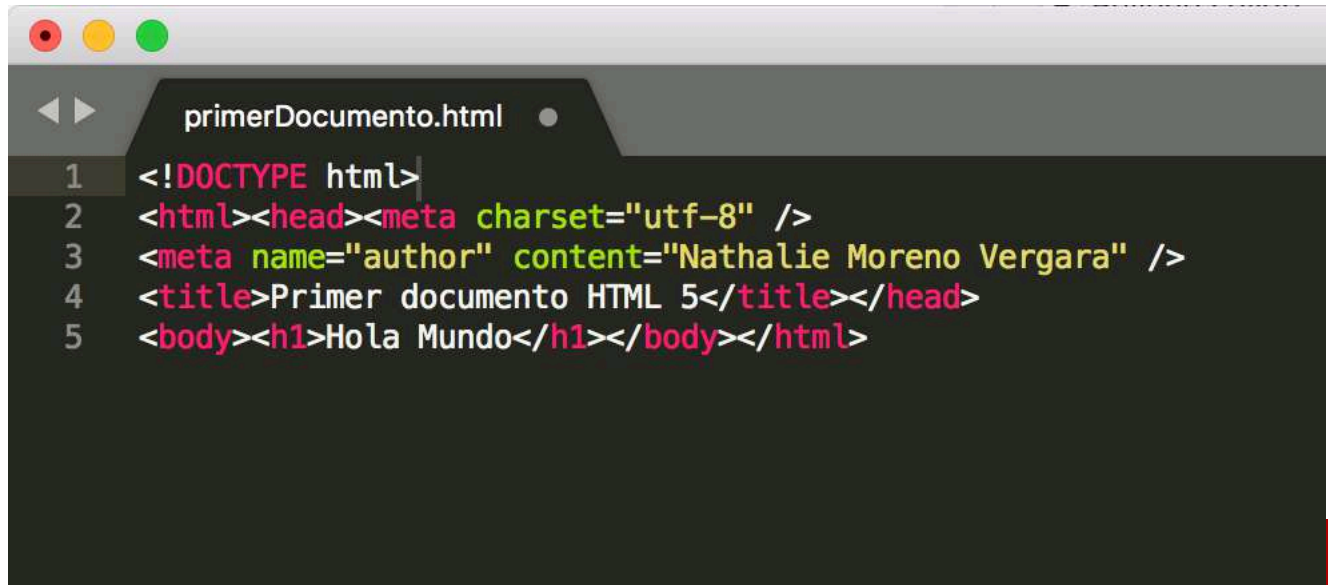
# HTML

## Detalles

---

- HTML no distingue entre mayúsculas y minúsculas en el nombre de las marcas (**case-insensitive**). Es decir, `<html>` es lo mismo que `<HTML>`.
- Se pueden introducir comentarios usando la etiqueta `<!-- comentario -->`
- Los espacios en blanco, tabulaciones y retornos de carro que haya entre marcas serán ignorados por el navegador web y podemos usarlo para formatear adecuadamente el texto.

- Por ejemplo:

A screenshot of a code editor window titled 'primerDocumento.html'. The editor shows five lines of HTML code with syntax highlighting. Line 1: `<!DOCTYPE html>`. Line 2: `<html><head><meta charset="utf-8" />`. Line 3: `<meta name="author" content="Nathalie Moreno Vergara" />`. Line 4: `<title>Primer documento HTML 5</title></head>`. Line 5: `<body><h1>Hola Mundo</h1></body></html>`. The code is displayed on a dark background with a light-colored border at the top and bottom.

```
1 <!DOCTYPE html>
2 <html><head><meta charset="utf-8" />
3 <meta name="author" content="Nathalie Moreno Vergara" />
4 <title>Primer documento HTML 5</title></head>
5 <body><h1>Hola Mundo</h1></body></html>
```

# HTML

## Referencias a caracteres

---

- Hay caracteres que tienen significado especial ( < > ” ... ) y no pueden emplearse directamente en el código HTML.
- Para ello se utilizan las referencias a caracteres: empiezan por & y acaban con ;
- Ejemplos
  - &lt; <
  - &gt; >
  - &amp; &
  - &quot; ”
  - &aacute; á
  - &eacute; é
  - &iacute; í
  - &oacute; ó
  - &uacute; ú
  - &ntilde; ñ
  - &iquest; ¿

# HTML

## Estructura de texto: encabezado

---

- Permiten describir secciones, subsecciones, subsubsecciones... del documento HTML
- Se pueden agrupar con el elemento **HGROUP**
- Existen seis tipos de encabezados diferentes; de más relevante a menos
  - `<h1>Encabezado 1</h1>`
  - `<h2>Encabezado 2</h2>`
  - `<h3>Encabezado 3</h3>`
  - `<h4>Encabezado 4</h4>`
  - `<h5>Encabezado 5</h5>`
  - `<h6>Encabezado 6</h6>`

**Encabezado 1**

**Encabezado 2**

**Encabezado 3**

**Encabezado 4**

**Encabezado 5**

**Encabezado 6**

# HTML

## Estructura de texto: párrafos

---

- La etiqueta `<p>` sirve para delimitar párrafos.
- El uso de la etiqueta de cierre es opcional: `</p>`
- Además de introducir un retorno de carro, fuerza un segundo retorno de carro para dejar una línea en blanco entre dos párrafos consecutivos.
- Por defecto los párrafos están justificados a la izquierda, pero podemos modificar la alineación horizontal de un párrafo usando CSS.
- **Nota:** en HTML 4.01 se puede cambiar la alineación sin CSS usando el atributo `align` (aunque se desaconseja), pero este atributo ha desaparecido en HTML 5.

## HTML

# Estructura de texto: líneas y marcas

---

- El elemento `<br>` introduce un retorno de carro en la posición en el que es colocado. No tiene etiqueta de cierre, solo de apertura.
- Es posible introducir una línea horizontal con `<hr>`
- Es posible destacar una frase, palabra, o conjunto de letras/letra de una palabra de distintas formas:
  - Palabras en otro idioma (extranjerismos), términos técnicos, etc: `<i>mutation</i>`
  - Palabras que el autor quiere enfatizar: `<em>coche</em>`
  - Palabras importantes para el autor: `<strong>¡Advertencia!</strong>`
  - Palabras importantes para el usuario pero no para el autor: `<mark>gatito</mark>`
  - Palabras a destacar que no entran en las categorías anteriores: `<b>redes</b>`

*mutation* *coche* **¡Advertencia!** **gatito** **redes**

## HTML

# Estructura de texto: fuentes, citas

---

- En HTML 4.01 y anteriores los elementos, **i** y **b** se usaban para indicar formato (itálica y negrita, respectivamente).
- Puede modificarse el tamaño de la fuente utilizando CSS
- **Nota:** El antiguo el elemento **FONT** de HTML 4.01 ha sido eliminado.
- El elemento **PRE** (con etiquetas de apertura y cierre) permite escribir texto respetando espacios y tamaño fijo de glifos
- El elemento **CITE** (<cite>...</cite>) permite destacar el título de un trabajo que se cita (se pone en cursiva por defecto)

# HTML

## Estructura de texto: ejemplo

---

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8" />
<title>Ejemplo2.html</title>
</head>

<body>
<h1> Este ser&acute; el t&uacute;tulo de la p&ágina </h1>
<h2>P&árrafos</h2>

<p>Este es el primer p&árrafo. Aqu&í hay una ruptura de línea
<br>de texto </p>

<p>Y &é;ste es otro p&árrafo.
Observa como hay una l&ínea en blanco entre ellos al mostrarse en el navegador web.</p>
```

# HTML

## Estructura de texto: ejemplo

---

```
<p>
<i>Extranjerismo</i><br>
<mark>Resultado de búsqueda</mark><br>
<em>&eacute;nfasis </em><br>
<strong>Importante</strong><br>
<b>Destacable</b></p>
```

`<pre>` al renderizar esto                    se van a respetar

los espacios en blanco y retornos de carro `</pre>`

```
<hr>
<p> Observa las l&eacute;neas horizontales </p> <hr>
```

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



# HTML

## Estructura de texto: ejemplo

---

Este será el título de la página

### Párrafos

Este es el primer párrafo. Aquí hay una ruptura de línea de texto

Y éste es otro párrafo. Observa como hay una línea en blanco entre ellos al mostrarse en el navegador web.

*Extranjerismo*

Resultado de búsqueda

*énfasis*

**Importante**

**Destacable**

al renderizar esto                      se van a respetar

los espacios en blanco y retornos de carro

---

Observa las líneas horizontales

---

*mutation coche* ¡Advertencia! gatito redes

# HTML

# Listas

---

- Las listas son una forma muy práctica de presentar información.
- En HTML existen varios tipos diferentes de listas:
  - Ordenadas
  - No ordenadas
  - De definición
- Las listas ordenadas, se definen con el elemento **OL**.
- Cada uno de los elementos de la lista se numera consecutivamente, de ahí el nombre de lista ordenada.
- Cada elemento de la lista ordenada se define empleando un elemento **LI**.

```
<ol>  
<li>Java </li>  
<li>Python </li>  
<li>C++</li>  
</ol>
```

```
1. Java  
2. Python  
3. C++
```

# HTML

# Listas

---

- También es posible anidar una lista dentro de otra

```
<ol>  
<li> En la plataforma Java:  
<ol>  
<li> Java </li>  
<li> Groovy </li>  
<li> JRuby </li>  
</ol></li>  
<li> Python </li>  
<li>C++</li>  
</ol>
```

```
1. En la plataforma Java:  
    1. Java  
    2. Groovy  
    3. JRuby  
2. Python  
3. C++
```

# HTML

## Listas

---

- **Listas no ordenadas:** mediante el elemento **UL**
- En estas listas, cada uno de los elementos de la lista aparecerá precedido de algún tipo de símbolo (un punto, un cuadrado, etc.).
- Al igual que en el caso de las listas ordenadas, podemos anidar listas dentro de ella.
- Los elementos se definen usando **LI**
- **Lista de definición:** suelen emplearse para mostrar términos seguidos de su definición.
- Los elementos de las listas no emplean ningún tipo de marca ni numeración para identificar cada elemento, sino que emplean sangrado entre los párrafos.
- Las listas de definición se crean con el elemento **DL**
- Los términos a definir se incluyen en el elemento **DT**
- Sus correspondientes definiciones se incluyen en **DD**
- Al igual que los otros tipos de listas, éstas también pueden anidarse.

# HTML

## Listas: ejemplo

---

```
<!DOCTYPE html>
```

```
<!-- Ejemplo3.html-->
```

```
<html>
```

```
<head>
```

```
<meta charset="utf-8" />
```

```
<title>Ejemplo3.html</title>
```

```
</head>
```

```
<body>
```

```
<h2>Listas</h2>
```

```
<h3> Lista ordenada </h3>
```

```
<p> Los <i> lenguajes de programación </i> que  
mas me
```

```
gustan son <i> (en orden de preferencia): </i></p>
```

```
<ol>
```

```
<li> Java </li>
```

```
<li> Python </li>
```

```
<li>C++</li>
```

```
</ol>
```

# HTML

## Listas: ejemplo

---

`<h3>` Listas desordenadas `</h3>`

`<p>`Sin un orden particular, mis `<b>` comidas favoritas `</b>` son las siguientes:`</p>`

`<ul>`

`<li>` La paella `</li>`

`<li>`El cocido gallego `</li>`

`<li>`Merluza`</li>`

`</ul>`

`<h2>` Listas de definición `</h2>`

`<dl>`

`<dt>` Java `</dt>`

`<dd>` Es el mejor lenguaje de programación del mundo`</dd>`

`<dt>`.NET`</dt>`

`<dd>`Es el lado oscuro de la fuerza`</dd>`

`</dl>`

`</body>`

`</html>`

# HTML

# Listas

---

## Listas

### Lista ordenada

Los lenguajes de programación que mas me gustan son (*en orden de preferencia*):

1. Java
2. Python
3. C++

### Listas desordenadas

Sin un orden particular, mis **comidas favoritas** son las siguientes:

- La paella
- El cocido gallego
- Merluza

### Listas de definición

Java

Es el mejor lenguaje de programación del mundo

.NET

Es el lado oscuro de la fuerza

# HTML

# Enlaces

---

- HTML permite crear enlaces entre documentos, esto es, vínculos entre un documento y otros documentos del mismo servidor web, documentos de cualquier otro servidor web, o secciones del mismo documento.

- Para crear enlaces se utiliza el elemento A cuya sintaxis es:

```
<a href="direccion">Texto del enlace</a>
```

- href puede apuntar a un documento que se encuentre en nuestro servidor web.

- Por ejemplo:

```
<a href="http://www.myweb.com/documentospersonales/CV.html">  
Mi currículum  
</a>
```



# HTML

# Enlaces

---

- Aunque la sintaxis del anterior enlace es correcta y válida, **nunca debe emplearse**. No es buena idea indicar la ruta absoluta de un recurso en nuestro propio servidor.
- Si, por ejemplo, cambiamos el directorio donde se encuentra la página web o cambiamos el dominio, cambiará la localización de las páginas.
- Es mucho mejor usar rutas relativas, que van a ser válidas aunque cambiemos de dominio o de directorio.
- Por ejemplo, suponiendo que el recurso está en una página que se encuentra en el mismo directorio desde el que la enlazamos, podríamos cambiar el enlace anterior por:
- `<a href="CV.html"> Mi currículum </a>`
- Los enlaces también pueden apuntar a cualquier otra URI de cualquier otro servidor web:
- `<a href="http://www.google.com"> Enlace a Google</a>`

# HTML

# Enlaces

---

- El tercer sitio al cual puede apuntar un enlace es a otra sección de la propia página.
- A veces, cuando la página es muy extensa, nos puede interesar dar un salto desde una posición a otra.
- Para ello tendremos que indicar empleando el elemento **A** cuáles son los posibles sitios a los cuales se puede "saltar" dentro del documento.
- La sintaxis es:

```
<a id= "nombre"></a>
```

- Para saltar al punto del documento donde se halle esa etiqueta emplearemos un enlace de la forma:
- `<a href="#nombre">volver al índice</a>`

# HTML

# Enlaces

---

- El valor del atributo **href** debe ser el nombre que hemos dado al punto del documento al cual debemos saltar, precedido por el signo de **#**. Es posible desde otra página web enlazar a una sección concreta de otro documento.

- Por ejemplo, supongamos que la página <http://www.ejemplo.es/ejemplo.html> tiene una marca

`<a id= "fin"></a>`

en algún punto del documento.

- Desde otra página, podríamos saltar directamente a esa región del documento mediante el enlace:
- `<a href=" http://www.ejemplo.es/ejemplo.html#fin"> ir al fin </a>`

# HTML

# Enlaces

---

- Podemos utilizar el atributo **target** para indicar en que ventana abrimos el destino del enlace:
- `_blank`: Ventana o pestaña nueva
  - `_parent`: marco padre
  - `_self`: mismo marco
  - `_top`: marco principal de la página
  - *Framename*: en una Ventana o pestaña con ese nombre, si es la primera vez o se cerró la anterior creará una nueva con el nombre indicado

# HTML

# Imágenes

---

- Podemos empotrar imágenes dentro de una página web empleando la etiqueta:

```

```

- El atributo `src` indica la fuente: la localización de la imagen.
- Puede ser una URL a cualquier imagen situada en cualquier página web de Internet, o una ruta relativa o ruta absoluta de esa imagen dentro de nuestro servidor web.
- Es posible especificar el alto y el ancho de la imagen con los atributos `width` y `height`.
- El alto y el ancho se medirá en **píxeles de CSS** (más cuando veamos CSS)
- Si especificamos ambos valores debemos respetar la relación de aspecto.
- Por lo general, es una buena práctica especificar sólo uno de los valores.
- En este caso, el navegador web escalará proporcionalmente la otra dimensión de la imagen y evitará distorsiones.

# HTML

## Imágenes

---

- El atributo **alt** de la etiqueta imagen permite especificar un texto alternativo, que se mostrará en pantalla en caso de no poderse mostrar la imagen.
- Algunos usuarios navegan por Internet sin mostrar imágenes para hacerlo más rápido, o por motivos de seguridad.
- Los invidentes, suelen emplear software que lee la pantalla del ordenador al navegar por Internet, y en el caso de las imágenes lee este texto.
- La etiqueta que se muestra a continuación empotra el logo de Google en una página, con un ancho de 500 píxeles y un tamaño de borde de cuatro píxeles.
- En caso de no poder mostrarse la imagen, se mostrará el texto "Logotipo de Google".

```

```



## Imágenes

---

- Es posible usar una imagen como enlace, es decir, que al hacer clic sobre la imagen nos lleve a otra página web.
- Para ello, simplemente dentro de la etiqueta del enlace en vez de el texto que habitualmente suelen llevar los enlaces, colocamos una etiqueta de imagen:

```
<a href="enlace">  </a>
```

# HTML

## Tablas

---

- Las tablas son una forma compacta y clara de mostrar información.
- En HTML, se definen mediante el elemento **TABLE**
- Pueden tener una descripción: **CAPTION**
- Se puede definir una cabecera con **THEAD** y la cabecera de las columnas con **TH**
- Y un pie con **TFOOT**
- El número de columnas en la cabecera y el pie deben ser iguales
- El cuerpo se define con **TBODY**
- No es necesario poner **TBODY** si solo tenemos cuerpo en la tabla
- Las filas se definen con el elemento **TR**
- Los datos en un fila se definen con **TD**
- Empleando el atributo border podemos indicar si la tabla no se utiliza simplemente para organizar otros elementos (típicamente se dibuja un borde).
- El valor del atributo puede ser una cadena vacía o "1"



# HTML

## Tablas: ejemplo

---

- La siguiente etiqueta define una tabla de  $3 \times 3$  "con borde":

```
<table border="1">
```

```
<tr><td>Celda A1</td><td>Celda B1</td><td>Celda C1</td></tr>
```

```
<tr><td>Celda A2</td><td>Celda B2</td><td>Celda C2</td></tr>
```

```
<tr><td>Celda A3</td><td>Celda B3</td><td>Celda C3</td></tr>
```

```
</table>
```

Celda A1	Celda B1	Celda C1
Celda A2	Celda B2	Celda C2
Celda A3	Celda B3	Celda C3

# HTML

## Tablas: ejemplo

---

➤ Otro ejemplo:

```
<TABLE border="1">
<CAPTION>Cups of coffee consumed by each senator</CAPTION>
<thead>
<TR><TH>Name<TH>Cups<TH>Type of Coffee<TH>Sugar?
<tbody>
<TR><TD>T. Sexton<TD>10<TD>Espresso<TD>No
<TR><TD>J. Dinnen<TD>5<TD>Decaf<TD>Yes
<TR><TD>A. Soria<TD colspan="3"><em>Not available</em>
</TABLE>
```

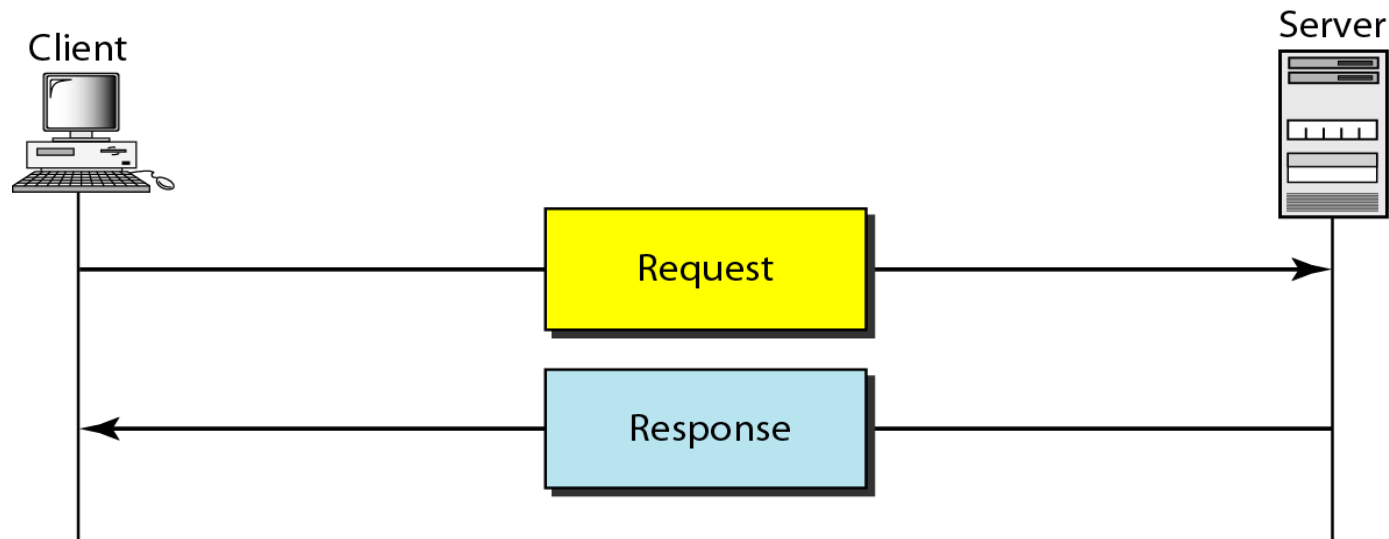
Name	Cups	Type of Coffee	Sugar?
T. Sexton	10	Espresso	No
J. Dinnen	5	Decaf	Yes
A. Soria	<i>Not available</i>		

# HTML

## Formularios: recordando HTTP

---

- Protocolo de transferencia de hipertexto
  - Permite acceder a los recursos de la Web
  - Utiliza TCP sobre el puerto 80
  - Es un protocolo sin estado
    - A través de una conexión TCP viaja una petición y una respuesta

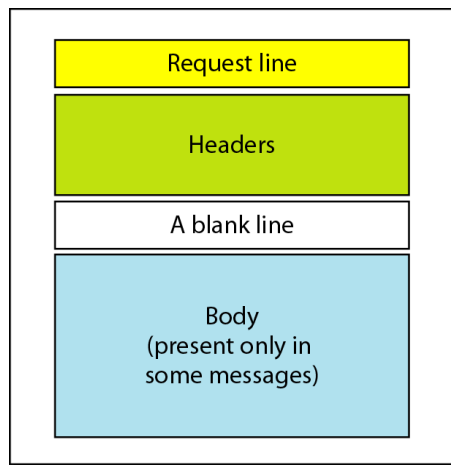


# HTML

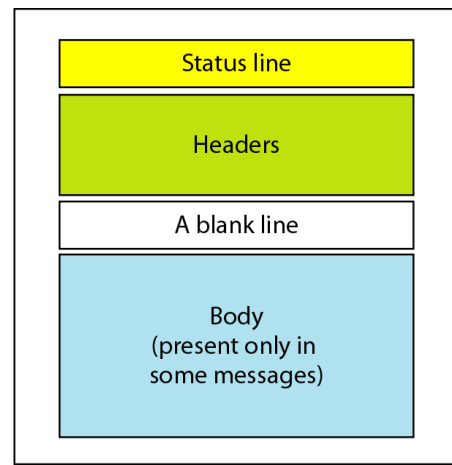
## Formularios: recordando HTTP

---

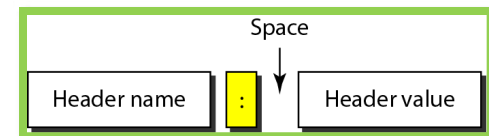
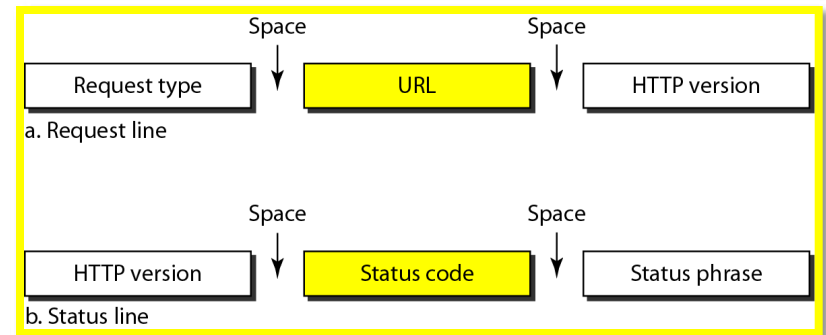
- Un mensaje de petición consta de:
  - Una línea de petición
    - Tipo de petición, URL, versión HTTP
  - Una cabecera
    - Formado por un conjunto de líneas en las que se especifica información adicional
  - Un cuerpo (opcional)
- Un mensaje de respuesta consta de:
  - Una línea de estado
    - Versión HTTP, código estado (3 dígitos), frase de estado
  - Una cabecera y un cuerpo (opcional)



Request message



Response message

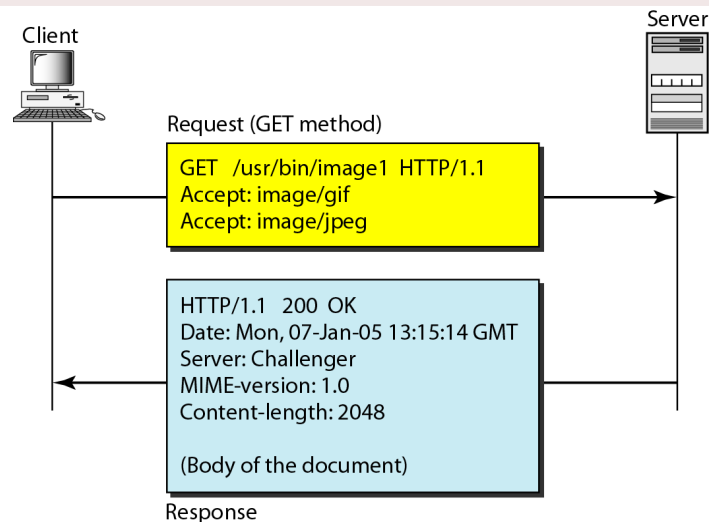


# HTML

## Formularios: recordando HTTP

---

Método	Acción
GET	Solicitud de un documento a un servidor
HEAD	Solicitud de información sobre un documento (no el documento en sí)
POST	Envío de alguna información del cliente al servidor
PUT	Envío de un documento del servidor al cliente
TRACE	Eco de petición entrante
CONNECT	Reservado
OPTIONS	Solicitud de algunas opciones disponibles
DELETE	Solicitud de eliminación de recurso



# HTML

## Formularios: recordando HTTP

---

- Mensajes de respuesta (estados y frases)
  - 200 OK
  - 301 Moved Permanently
  - 400 Bad Request
  - 404 Not Found
  - 505 HTTP Version Not Supported

# HTML

# Formularios

---

- Los formularios se emplean para recoger datos del usuario en una página web y enviarlos al servidor.
- Estos datos pueden recogerse mediante **campos de texto**, **checkboxes**, **listas de selección**, etc.
- En el servidor, debe haber algún programa (como los que aprenderemos a desarrollar lo largo de este curso) capaz de recoger esos datos y procesarlos de algún modo.
- **HTML** y **http** se encargan sólo de enviarlos al servidor, pero no definen qué se hace ni proporcionan ningún mecanismo para hacer algo con ellos.
- Los formularios se crean con el elemento **FORM**.
- Dentro de esta etiqueta es donde debemos colocar todos los elementos del formulario destinados a recoger la entrada del usuario.

# HTML

# Formularios

---

- Los atributos más importantes de la etiqueta de formularios son: **action** y **method**
- **action**: especifica la URL donde se encuentra el programa en el servidor que se va a encargarse de procesar la información enviada por el formulario.
- Es posible enviar los datos a través de correo electrónico especificando el valor del atributo:

**action="mailto: direccion\_de\_correo"**

- En este caso también deberemos añadir el atributo **enctype="text/plain"** para que el correo recibido no resulte ilegible.
- No obstante, lo más habitual es enviar los datos a una URL en el servidor usando el método GET o POST.
- La URL puede especificarse tanto de modo relativo como de modo absoluto.



# HTML

# Formularios

---

- `method = " POST / GET "`
- Indica el método que se va a emplear para enviar la información del formulario al servidor.
- POST envía los datos dentro del cuerpo de la petición.
- El método GET añade los argumentos del formulario a la URL especificada en el atributo `action`, utilizando como separador de las distintas piezas de información el signo de interrogación"?" (cadena de consulta)
- El método más usado es POST.
- Si queremos mandar el formulario a una dirección de correo electrónico, es obligatorio usar este método.

# HTML

# Formularios

---

- Veamos a continuación qué elementos podemos colocar dentro de un elemento **FORM**
- El principal elemento para recoger información es **INPUT**.
- Tiene un atributo con nombre **type** que permite especificar el tipo concreto de campo de entrada que queremos emplear.
- Puede ser: text, password, checkbox, radio, submit, reset, file, hidden, image, button, color, range, tel, email, date, time, number, url, search

## Formularios: text

---

- En un campo de entrada de texto podemos añadir los siguientes atributos a la etiqueta input:
  - **name** = "nombre": asigna un nombre que identificará el campo de texto. Cuando desde un programa en el servidor queramos recuperar el contenido del campo de texto deberemos indicar este nombre.
  - **maxlength** = "n": especifica el número máximo de caracteres que el usuario podrá incluir en el campo de texto
  - **size** = "n": establece el tamaño del campo de texto en la pantalla.
  - **value** = "texto": permite especificar el valor por defecto que va a aparecer en la caja de texto
  - **disabled**: desactiva el campo de texto, por lo que el usuario no podrá escribir nada en él.

# HTML

## Formularios: ejemplo text

---

```
<form action=" http://www.formulario.es/ejemplo" method="post"
enctype="text/plain" name="ejemplo">
<p>Introduce tu lenguaje de programación favorito:
<input type="text" maxlength="20" size="20" name="lenguaje">
</form>
```

Introduce tu lenguaje de programación favorito:

## Formularios: radio

---

- En los botones de radio se pueden emplear los siguientes atributos:
  - **name** = "nombre": asigna un nombre único a la pieza de información que representarán los botones de radio que pertenezcan al mismo grupo. Éste es el identificador que emplearemos en el servidor para recuperar esta pieza de información. Debe ser el mismo para todos los elementos radio de un mismo grupo.
  - **value** = "valor": define el valor que tomará la variable indicada en el atributo anterior y que se enviará al servidor. Es decir, al seleccionar uno de los botones el valor que tomará la variable que se envía al servidor será este atributo.
  - **checked**: permite indicar cuál es el elemento que está seleccionado por defecto dentro de un grupo.
  - **disabled**: permite desactivar el botón

# HTML

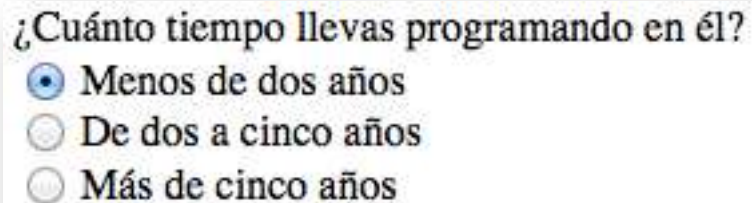
## Formularios: ejemplo radio

---

➤ Por ejemplo, al incluir esto dentro de un formulario:

```
¿Cuánto tiempo llevas programando en él?<br>
<input type="radio" name="tiempo" value="0-2" checked> Menos de dos años<br>
<input type="radio" name="tiempo" value="2-5"> De dos a cinco años<br>
<input type="radio" name="tiempo" value="5-"> Más de cinco años<br>
```

➤ Obtenemos



¿Cuánto tiempo llevas programando en él?

- ☒ Menos de dos años
- ☐ De dos a cinco años
- ☐ Más de cinco años

## Formularios: checkbox

---

- Define una o más casillas de las cuales el usuario puede marcar una, varias o ninguna.
- Podemos especificar los siguientes atributos de `input`:
  - `name` = "nombre": asigna un nombre único a la pieza de información que representa un grupo de casillas. Éste es el identificador que emplearemos en el servidor para recuperar esta pieza de información.
  - `checked`: permite indicar si una casilla está o no seleccionada por defecto.
  - `disabled`: desactiva la casilla.

# HTML

## Formularios: ejemplo checkbox

---

- Si incluimos estas etiquetas dentro de un formulario:

¿En qué entornos has usado ese lenguaje?:<br>

<input type="checkbox" name="entornos" value="escritorio" checked> Escritorio<br>

<input type="checkbox" name="entornos" value="web" checked> Web<br>

<input type="checkbox" name="entornos" value="movilidad"> Movilidad<br>

<input type="checkbox" name="entornos" value="empotrado"> Dispositivos  
empotrados<br>

- El navegador web mostrará algo parecido a:

¿En qué entornos has usado ese lenguaje?:

☒ Escritorio

☒ Web

☐ Movilidad

☐ Dispositivos empotrados



## Formularios: submit y reset

---

- Para enviar el formulario al servidor se suele utilizar una entrada de tipo **submit**
- Cuando el usuario haga clic en ese botón, todos los datos del formulario serán enviados al programa del servidor o a la dirección de correo indicada en **action**.
- Con esta entrada podemos emplear los siguientes atributos:
  - **value** = "valor": permite definir el texto que se va a mostrar en el botón de envío.
  - **disabled**: desactiva el botón.
- Para añadir un botón que reinicie el formulario debemos indicar una entrada de tipo **reset**, con los mismos atributos que tiene la entrada **submit**

## HTML

# Formularios: ejemplo submit y reset

---

- Si añadimos a nuestro formulario

```
<input type="submit" value="enviar">  
<input type="reset" value="reset">
```

- Obtenemos



## Formularios: password y hidden

---

- Las entradas tipo **password** son campos de texto en el cual cuando se teclea no se ve el contenido tecleado, sino que se muestran sólo "\*" u otro carácter similar.
- Su propósito es permitir introducir contraseñas de forma discreta
- Los campos de tipo **hidden** son campos ocultos (invisibles), que no se mostrará en pantalla de ningún modo.
- Se utilizan para almacenar en el formulario valores que permiten identificar objetos en el servidor cuando el formulario es recibido
- Podemos usar los siguientes atributos en los campos ocultos:
  - **name** = "nombre": asigna un nombre al campo oculto.
  - **value** = "valor": valor que toma el campo oculto, y que el usuario no puede modificar de ningún modo.

## Formularios: listas de selección

---

- El elemento **SELECT** permite definir listas desplegables en las cuales el usuario podrá elegir uno o varios valores.
- Este elemento soporta los siguientes atributos:
  - `name = "nombre"`: asigna un nombre a la lista; este será el nombre mediante el cual podremos identificar en el servidor la selección del usuario.
  - `size = "n"`: permite definir el número de opciones que van a ser visibles en la lista. Si `n=1` en la lista se presentará como una lista desplegable (al estilo de un combo box), mientras que si se indica otro valor la lista se representará como una caja como barra de desplazamiento vertical.
  - `multiple`: permite seleccionar varias opciones a la vez. Si se omite este atributo, el usuario sólo podrá seleccionar una de las opciones.
  - `disabled`: desactiva la lista.

## Formularios: listas de selección

---

- Para especificar cada una de las opciones de la lista se utiliza el elemento **OPTION**, que admite los siguientes parámetros:
  - **value**: indica el valor que será asociado al parámetro **name** de `<select>` cuando se envíe el formulario.
  - **selected**: indica que esta opción va a estar seleccionada por defecto. De no indicarse en ninguna de las opciones de la lista que está seleccionada por defecto, no aparecerá ninguna seleccionada.

# HTML

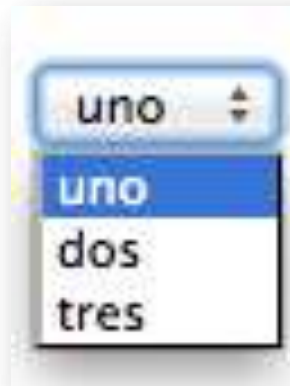
## Formularios: ejemplo listas

---

- Por ejemplo, incluyendo estas etiquetas en un formulario:

```
<select size=1 name="opcion">  
<option value="uno">uno</option>  
<option value="dos">dos</option>  
<option value="tres">tres</option>  
</select>
```

- Obtenemos



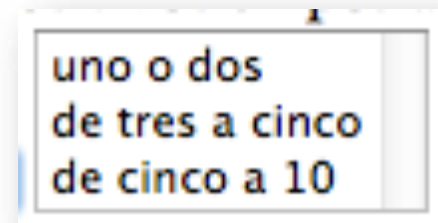
## Formularios: ejemplo listas

---

- Mientras que incluyendo estas etiquetas:

```
<select size=3>  
<option value="uno">uno o dos </option>  
<option value="dos"> de tres a cinco </option>  
<option value="tres"> de cinco a 10</option>  
<option value="cuatro"> de 10 a 25 </option>  
<option value="cinco"> de 25 a 50 </option>  
<option value="seis"> más de 50</option>  
</select>
```

- Obtenemos una lista como ésta:



## Formularios: áreas de texto

---

- Para incluir campos de texto que puedan contener múltiples líneas empleamos el elemento `TEXTAREA`
- A diferencia de la etiqueta `input` cuando el atributo `type = "text"`, en este campo de texto el usuario podrá introducir múltiples líneas.
- Si no escribimos ningún texto entre la etiqueta de apertura y de cierre, el campo de texto aparecerá vacío en el formulario.
- Si introducimos algún texto, éste será valor por defecto del campo de texto en el formulario.
- Al igual que suele suceder con cualquier campo de texto, si el usuario introduce más texto del que se puede representar en él, automáticamente aparecerá una barra de desplazamiento vertical.



## Formularios: áreas de texto

---

- El **TEXTAREA** soporta (entre otros) los siguientes atributos:
- `name = "nombre"`: especifica el nombre con el cual se va a identificar el campo de texto; este será el nombre asociado en el servidor al texto introducido por el usuario.
  - `cols = "c"`: especifica el número de columnas visibles del área de texto.
  - `rows = "r"`: especifica el número de filas visibles del área de texto.
  - `disabled`: deshabilita el área de texto.
  - `readonly`: convierte el área de texto en un área de sólo lectura que no puede ser modificada por el usuario.

# HTML

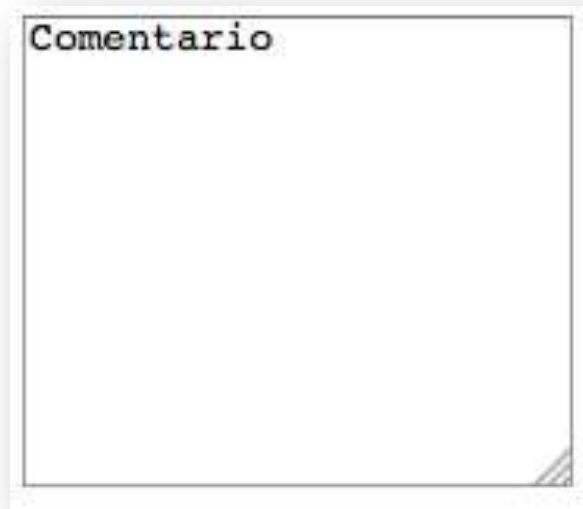
## Formularios: ejemplo de textarea

---

- Si añadimos a nuestro formulario

```
<textarea name="comentario" cols="20"  
rows="10">Comentario</textarea>
```

- Obtendremos un área de texto como:



# XHTML

# Motivación

---

- En los primeros tiempos de la Web no existían estándares
- Los navegadores interpretaban el código HTML de maneras diferentes
- Algunos navegadores definían atributos o etiquetas propias, lo cual formaba parte, en ocasiones, de una táctica para distinguirse de sus competidores y ganar cuota de mercado
- Es habitual encontrar código HTML que comprueba el navegador en el que se visualiza para usar una u otra característica
- Los navegadores admiten código con errores
- **Resultado:** caos en la calidad del código HTML

# XHTML

# Motivación

---

- Aún persiste el caos en las páginas Web  
(<https://validator.w3.org/>)

Error found while checking this document as HTML 4.01 Transitional!			
Result:	1 Error, 4 warning(s)		
Address :	<input type="text" value="http://www.unicaja.es/"/>		
Encoding :	utf-8	<input type="button" value="(detect automatically)"/>	
Doctype :	HTML 4.01 Transitional	<input type="button" value="(detect automatically)"/>	
Root Element:	html		

Errors found while checking this document as HTML 4.01 Transitional!			
Result:	13 Errors, 5 warning(s)		
Address :	<input type="text" value="http://www.lcc.uma.es/"/>		
Encoding :	windows-1252	<input type="button" value="(detect automatically)"/>	
Doctype :	HTML 4.01 Transitional	<input type="button" value="(detect automatically)"/>	
Root Element:	html		

# XHTML

# Motivación

➤ Aún persiste el caos en las páginas Web

Errors found while checking this document as HTML 4.01 Strict!	
Result:	51 Errors, 95 warning(s)
Address :	<input type="text" value="http://www.uma.es/"/>
Encoding :	utf-8 <input type="button" value="(detect automatically)"/>
Doctype :	HTML 4.01 Strict <input type="button" value="(detect automatically)"/>
Root Element:	HTML

Showing results for http://www.uma.es/

Checker Input

Show ☐ source ☐ outline ☐ image report 

Check by

## Errors (9)

Press the Message Filtering button to collapse the filtering options and error/warning/info counts.

**Errors (9)** · [hide all errors](#) · [Show all errors](#)



- ☒ A document must not include both a `meta` element with an `http-equiv` attribute whose value is `content-type`, and a `meta` element with a `charset` attribute.
- ☒ Using the `meta` element to specify the document-wide default language is obsolete. Consider specifying the language on the root element instead.

# XHTML

# Motivación

---

- Aún persiste el caos en las páginas Web

Errors found while checking this document as XHTML 1.0 Transitional!			
Result:	365 Errors, 284 warning(s)		
Address :	<input type="text" value="http://www.telecinco.es/"/>		
Encoding :	<input type="text" value="utf-8"/>	<input type="text" value="(detect automatically)"/>	
Doctype :	<input type="text" value="XHTML 1.0 Transitional"/>	<input type="text" value="(detect automatically)"/>	
Root Element:	<input type="text" value="html"/>		
Root Namespace:	<input type="text" value="http://www.w3.org/1999/xhtml"/>		

# XHTML

# Motivación

---

- Los estándares W3C pretenden aliviar este problema
- Los validadores permiten comprobar si nuestra página Web sigue con los estándares

<http://validator.w3.org/>

- El código HTML caótico que se puede encontrar en la Web contrasta con el estricto espíritu de la programación... y dificulta su colaboración
- ¿Cómo obtener información a partir de HTML mal formado?
- XML es un lenguaje de marcado estricto en su sintaxis, genérico, con posibilidad de comprobación automática y con muchas herramientas asociadas

# XHTML

## Características

---

- XHTML=XML + HTML
- XHTML es un lenguaje de marcado tan estricto como XML orientado a la Web
- A diferencia de HTML en XHTML:
  - Se distinguen mayúsculas y minúsculas: no es lo mismo `<html>` que `<HTML>`
  - Todas las etiquetas se escriben en minúscula: `<html>` (~~no `<HTML>`~~)
  - Todas las etiquetas de apertura deben tener una de cierre, salvo las que no exigen un cierre, que deben de la forma `<etiqueta />`
  - Los valores de los atributos deben estar encerrados entre comillas:  
`<select size="1">`
  - Los elementos deben formar un árbol (no se puede cerrar una etiqueta si no fue la última en abrirse): `<p><b>hola</b></p>`



# XHTML

## Características

---

➤ A diferencia de HTML en XHTML (cont.):

- No hay atributos sin valor:

```
<input type="radio" name="lenguaje" value="Java" checked="checked" />
```

- Hay que poner el espacio de nombres XML:

```
<!DOCTYPE html>
```

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

- En el estándar HTML 5 se definen dos sintaxis: una HTML y otra XHTML
- Salvo por las diferencias sintácticas, XHTML y HTML son muy similares
- Lo que ya sabemos de HTML nos resultará útil para escribir XHTML
- Las páginas de JavaServer Faces serán XHTML
- Hoy en día XHTML está en **desuso** como estándar para la Web

# Para ampliar conocimientos

- Etiquetas HTML:  
<https://www.w3schools.com/tags/>
  - Especificación HTML 5:  
<http://www.w3.org/TR/2014/REC-html5-20141028>
-