



tema 2 ~ DEW

Introducción

Estructura de una página web

Texto

Listas

Enlaces

Multimedia

Tablas

Formularios

Layout – Diseño y Estructura

▼ Introducción



Define la **estructura y contenido** de un documento (no presentación/apariencia)

Lenguaje de "marcado" basado en "**marcas**" o "**tags**" o "etiquetas" → Cada tag, o marca, puede tener atributos que indican propiedades del tag

```
<h1>Acerca de Google</h1>
```

```
<br />
```

ESTRUCTURA: `<tag atributo="valor">...</tag>`

```
<a href="http://www.google.com" target="blank"> Google </a>
```

Dependiendo de la versión de HTML utilizada, las reglas de marcado son más o menos estrictas



Todo documento HTML5 viene anunciado con una orden **<!DOCTYPE>**

▼ PLANTILLA "BÁSICA"

```
<!DOCTYPE html>
<html lang="es">
<head>
  <meta charset="utf-8">
  <title>HTML</title>
  <link rel="stylesheet" href="estilo.css">
</head>
<body>
  <p>Esta página web es una página HTML válida.</p>
</body>
</html>
```

▼ Estructura de una página web

- Para comentar un texto: `<!-- Texto a comentar -->`

<preámbulo>	Indica cual es la sintaxis del documento
<head> <title>el titulo</title> </head>	Propiedades globales y metadatos
<body> <p>contenidos</p> </body>	Contenidos
</html>	

▼ <preámbulo>

Anuncia el **tipo de documento**, en HTML5 es más simple y directo que en otras versiones

```
<!DOCTYPE html>
<html lang="es">
```

▼ <head>

Proporciona información acerca de la página.

- **DEBE** contener el título de la ventana del navegador → `<title> titulo </titulo>`
- **PUEDE** contener → `<base>`, `<meta>`, `<link>`, `<style>`, `<script>`, `<object>`

Se pueden definir **dos tipos de URLs**:

1. **Absolutas** → `http://www.sitio.com/img/doc.png`
2. **Relativas** → `img/doc.png`
 - a. Dada una URL relativa, la absoluta se obtiene a partir del documento que la referencia.

<meta>	<p>Contiene metainformación sobre el documento, procesable por programa (motores de búsqueda). Hay <u>nombres preestablecidos</u>, como description, keywords, author, ...</p> <p>Puede definir cabeceras HTTP (mediante <code>http-equiv</code>).</p> <p>En dispositivos móviles permite controlar el display.</p>	<pre><meta name="el_nombre" content="los_valores" /> <meta http-equiv="Content-Type" content="text/html; charset=utf-8" /> <meta http-equiv="pragma" content="no-cache" /> <meta name="viewport" content="width=devicewidth; userscalable=0;" /></pre>
<link>	<p>Para relacionar el documento HTML con otro externo</p> <p>Usado típicamente para cargar hojas de estilo CSS</p> <ul style="list-style-type: none"> • type es el tipo MIME • rel es la relación entre el documento actual y el referenciado 	<pre><link type="text/css" rel="stylesheet" href="..." /></pre>
<style>	<p>Para definir hojas de estilo (CSS) <u>incrustadas</u> en el mismo documento HTML</p>	<pre><style type="text/css" > ...definiciones de estilo ... </style></pre>
<script>	Para definir o referenciar scripts	<u>INCRUSTADO (interno):</u>

		<pre><script type="text/javascript" >.... código de scripting ... </script> REFERENCIADO (externo): <script type="text/javascript" src="..." ></script></pre>
--	--	--

▼ <body>

Incluye los **contenidos de la página** y la mayor parte de las **construcciones HTML** que el navegador visualizará.



Dentro del "body" podemos incluir →

▼ Texto

Texto básico:

- Texto libre
- Entidades
- Saltos de línea

, párrafos <p>
- Encabezados
<h1>, <h2>...
- Texto preformateado

Texto formateado y semántico:

- Efectos físicos
- Etiquetas para el marcaje

▼ TEXTO BÁSICO

Texto libre	Texto sin estructura		Aquí aparece texto libre con varios espacios	libre	
Entidades	Caracteres especiales		y algunas líneas en blanco	líneas en	Aquí aparece texto libre con varios espacios y algunas líneas en blanco
Salto de línea	Texto libre Texto en nueva línea	Texto libre Texto en nueva línea nea			
Párrafos	Esto es un párrafo Esto es otro párrafo	<p>Esto es un párrafo</p> <p>Esto es otro párrafo</p>			
Encabezados	<h1>, <h2>, <h3>...	<h1>Esto es un encabezado</h1>			
Texto preformateado	Aparece tal cual a como está escrito				

Carácter	Número entidad	Nombre entidad
"	"	"
&	&	&
<	<	<
>	>	>
Á	á	Á
í	é	í
ñ	ñ	ñ
<espacio>	 	

ENTIDADES

```
<pre>
function testFunction(strText){
  alert(strText)
}
</pre>
```

TEXTO PREFORMATEADO

▼ TEXTO FORMATEADO

Negrita	Hola	Hola
Itálica	<i><i>Hola</i></i>	<i>Hola</i>
Superíndice	Hola^{hola}	Holahola
Subíndice	Hola _{hola}	Holahola
Pequeño	<small><small>Hola</small></small>	Hola
Raya horizontal	<hr />	

▼ TEXTO SEMÁNTICO

Énfasis	
Citas	<blockquote>, <cite>, <q>
Abreviaciones	<abbr>, <acronym>, <dfn>
Código fuente	<code>, <kbd>, <var>, <samp>
Direcciones	<address>
Edición	<ins>,

Output

Énfasis

Enfais

Wow una cita

Dige yo “esto no se que hace”

Abrebiación

Abrebiación

Abrebiación

Código

Código

Código

Código

Dirección

Edición

~~Edición~~

```
<em>Enfasis</em>
<strong>Enfais</strong>

<blockquote>Wow una cita</blockquote>
<cite>Dige yo</cite>
<q>esto no se que hace</q>

<abbr>Abrebiación</abbr>
<acronym>Abrebiación</acronym>
<dfn>Abrebiación</dfn>

<code>Código</code>
<kbd>Código</kbd>
<var>Código</var>
<samp>Código</samp>

<address>Dirección</address>

<ins>Edición</ins>
<del>Edición</del>
```

▼ Listas

Para enumerar elementos, se pueden anidar, hay varios tipos:

- **LISTAS NO ORDENADAS**

No importa el orden

<code></code>	
<code>Item 1</code>	• Item 1
<code>Item 2</code>	• Item 2
<code></code>	

- **LISTAS ORDENADAS**

Soporta atributos `type` y `start`, para modificar los 'bullet points'

<code></code>	
<code>Item 1</code>	1. Item 1
<code>Item 2</code>	2. Item 2
<code></code>	

Atributo type	Ejemplos
1	1, 2, 3, 4, 5
A	A, B, C, D, E
a	a, b, c, d, e
I	I, II, III, IV, V
i	i, ii, iii, iv, v

- **LISTA DE DEFINICIÓN DE TÉRMINOS (Diccionario)**

```
<dl>
  <dt>Primer término</dt>
  <dd>Primera definición</dd>
  <dd>Segunda definición</dd>
  <dt>Segundo término</dt>
  <dd>Definición</dd>
</dl>
```

Primer término

Primera definición

Segunda definición

Segundo término

Definición

▼ Enlaces

```
<a href="#"> contenido </a>
p.ej → <a href="http://www.google.com">Google</a>
```

DESTINO LOCAL → URL en el que se referencia a un **punto concreto de una página de hipertexto**

1. Identificar el punto de la página al que quiero ir con el atributo "id"

```
<h2 id="seccion1">Sección 1</h2>
```

2. Poner el enlace a ese punto desde otra página

```
<a href="#seccion1">Ir a la Sección 1</a>
```

3. El navegador carga la página entera y luego desplaza el foco hasta el punto indicado

- **ATRIBUTO TARGET** → Determina **dónde se abre el enlace**, ya sea en la misma ventana (_self, por defecto), en una nueva ventana (_blank), o en una ventana con un nombre determinado (citando el destino).
- **ATRIBUTO TITLE** → Para mostrar un **texto informativo** cuando el usuario pasa el ratón sobre el enlace. Es útil para dar más detalles sobre el destino del enlace.
- **OTROS ATRIBUTOS** → Para definir otras propiedades de los enlaces, como el tipo de **contenido**, el **idioma**, la relación con el documento actual, etc → *accesskey, charset, coords, hreflang, rel, rev, shape, tabindex,*

type (MIME)

```
<a href="https://www.Google.com" target="_blank" title="Haz clic aquí para ir a la pág
```

▼ Multimedia

Las marcas indican → las procedencias del medio, sus propiedades y los controles de usuario a mostrar.

El texto que pueda aparecer dentro del código, sólo se muestra **en caso de que el navegador NO soporte esa marca**, por ejemplo un navegador que NO soporte audio.

```
<audio ...>
  Si el navegador no soporta "audio"
  <source ... />
</audio>
```

▼ IMAGEN

```

```

- `src` → Ubicación de la imagen (URL relativa/absoluta)
- `height="100" width="100"` → Dimensiones en píxeles o porcentajes sobre el contenedor
- `alt="Una imagen"` → Descripción textual de la img (para cuando no se puede ver)

```
<figure id="..." >
```

- Permite arropar cualquier construcción, incluso una imagen, para que pueda ser referenciada de manera más estructurada y significativa en el código.
- Se utiliza para **agrupar contenido relacionado**, como imágenes, gráficos, diagramas, ilustraciones o cualquier otro elemento multimedia. Su propósito principal es **proporcionar contexto semántico** al contenido dentro de él.
- Ayuda a los **navegadores y motores de búsqueda** a entender la relación entre la imagen y su descripción.

```
<figure>

<figcaption>Descripción de la imagen</figcaption>
</figure>
```

`<figure>` → contiene una imagen (``) y su descripción (`<figcaption>`).

IMAGEN COMO ENLACE

```
<a href="/index.html" >

</a>
```

▼ FORMATOS DE IMAGEN SOPORTADOS

- **GIF**: Adecuado para colores "planos" (hasta 256): logotipos, iconos, etc.; no fotografías
- **PNG**: Mejora de GIF (>256 colores), para los mismos casos
- **JPEG**: Compresión con pérdida, útil para fotografía
- **SVG**: Imagen vectorial, sin pérdida, útil para imagen sintética

▼ RENDERIZADO LOCAL

`<canvas>`

Reserva un espacio en el que se pueden dibujar gráficos mediante **scripting**.

Proporciona una **superficie de dibujo** en la que se pueden crear gráficos, animaciones y visualizaciones.

```
<canvas id="miCanvas" width="400" height="200"></canvas>
```

El atributo id permite **referenciar** este contenedor desde el script.

▼ AUDIO

`<audio>`

Permite reproducir archivos de sonido en una página web.

Se utiliza para **incrustar contenido de audio** en una página web.

Puede contener uno o más elementos `<source>` que especifican la **ubicación y el tipo de archivo de audio**.

```
<audio controls>
  <source src="cancion.ogg" type="audio/ogg">
  <source src="cancion.mp3" type="audio/mpeg"> Tu navegador no soporta el element
</audio>
```

El atributo `controls` agrega **controles de reproducción** (reproducir, pausar, volumen, etc.).

▼ VIDEO

`<video>`

Permite reproducir archivos de sonido en una página web.

Se utiliza para incrustar contenido de video en una página web.

Al igual que con `<audio>`, puede contener uno o más elementos `<source>` que **especifican la ubicación y el tipo de archivo de video**.

```
<video width="320" height="240" controls>
  <source src="peli.ogg" type="video/ogg">
  <source src="peli.mp4" type="video/mp4"> Tu navegador no soporta el elemento de
</video>
```

El atributo `controls` agrega **controles de reproducción** (reproducir, pausar, volumen, etc.).

▼ Tablas

Permiten mostrar información **de manera tabular** (*filas y columnas*)

Uso típico: presentación de datos

Uso desaconsejado: layout de la página

`<table>`

Filas con `<tr>`

Columnas/celdas con `<th>` (cabecera) o `<td>`

```
<table border="1">
  <tr><th>Cabecera 1</th><th>Cabecera 2</th></tr>
  <tr><td>fila 1, celda 1</td><td>fila 1, celda 2</td></tr>
```

```
<tr><td>fila 2, celda 1</td><td>fila 2, celda 2</td></tr>
</table>
```

`<caption>`

Permite añadir un título a una tabla, debe aparecer inmediatamente después de `<table>`

```
<table border="1">
  <caption>El título</caption>
  <tr><th>Cabecera 1</th><th>Cabecera 2</th></tr>
  <tr><td>fila 1, celda 1</td><td>fila 1, celda 2</td></tr>
  <tr><td>fila 2, celda 1</td><td>fila 2, celda 2</td></tr>
</table>
```

Operaciones sobre tablas:

- Agrupar filas (`<thead>` , `<tbody>` , `<tfoot>`)
- Agrupar columnas (`<colgroup>`)
- Expandir filas (`<rowspan>`)
- Expandir columnas (`<colspan>`)

▼ Formularios

El formulario es la construcción HTML que **permite al usuario introducir o seleccionar información** y que envía esa información al servidor web para su procesamiento.

MECANISMO

1. **El cliente completa los datos en el navegador**
2. **Cuando finaliza, pulsa el botón Aceptar**
3. **El navegador construye la petición HTTP**
 - Esta petición incluye los valores de los campos del formulario, identificados por sus atributos `name`.
 - La petición se dirige al **CGI** (*Common Gateway Interface*) especificado en el atributo `action` del formulario.
4. **El servidor ejecuta el CGI solicitado pasándole los datos recibidos:**
 - El servidor recibe la petición y ejecuta el programa **CGI** correspondiente.
 - El **CGI** procesa los datos enviados desde el cliente y realiza las acciones necesarias (por ejemplo, almacenar en una base de datos, enviar correos electrónicos, etc.).
5. **El CGI construye la respuesta como un objeto MIME (en este ejemplo, HTML) que devuelve al cliente solicitante:**
 - Esta respuesta se envía de vuelta al cliente (navegador) como parte de la **respuesta HTTP**.
 - El navegador interpreta la respuesta y muestra la página web resultante al usuario.

```
<form action="..." method="...">
```

- Un formulario contiene **controles** o **campos**, pueden ser cajas de texto, listas desplegables, botones, etc.
- El contenido puede ser **cualquier código HTML** válido (texto, imágenes o incluso elementos de diseño).



Cada control dentro del formulario debe tener un atributo llamado `name` → Se utiliza para **identificar el valor ingresado** por el usuario cuando se envía el formulario (como el nombre de una variable).

Cuando se presiona el botón tipo submit...

1. Todos los datos ingresados por el usuario **se agrupan utilizando el atributo** `name` asociado a cada control.
2. La información agrupada **se envía al servidor web** (mediante una solicitud HTTP de tipo POST) que contiene los valores de los campos del formulario.
3. El servidor **procesa los datos recibidos**. Puede almacenar la información, enviar correos electrónicos, generar respuestas...)

ATRIBUTOS

Atributo `action` : Define la **URL** a la que se enviarán los datos del formulario cuando el usuario lo envíe. Especifica la ubicación del servidor o script que procesará la información.

```
<form action="https://mi-servidor.com/procesar.php" method="post">
```

En este caso, los datos del formulario se enviarán al archivo "procesar.php" en el servidor "mi-servidor.com".

Atributo `method` : Determina el **protocolo HTTP** utilizado para enviar los datos al servidor. Los valores más comunes son "GET" y "POST":

Otros atributos:

`enctype` : Define cómo se codificarán los datos antes de enviarlos al servidor

`onsubmit` : Permite ejecutar una función JavaScript cuando el usuario envía el formulario.

`onreset` : Define una función que se ejecuta cuando se restablecen los valores del formulario.

```
<form action="http://www.misitio.es/buscar" method="post">
  <h2>Buscando en la web misitio</h2>
  <input type="text" name="quebuscas" />
  <input type="submit" />
</form>
```

PRINCIPALES CAMPOS

▼ Etiquetas

```
<label>
```

Se utiliza para asociar texto descriptivo con otros campos del formulario.

```
<label for="nombre">Nombre: </label>
<input type="text" id="nombre" value="" size="30" maxlength="40" />
```

En este caso, la etiqueta "Nombre:" está vinculada al campo de entrada de texto con el atributo `id="nombre"`.

El atributo `for` en la etiqueta `<label>` especifica qué control del formulario está relacionado con esa etiqueta. Debe coincidir con el valor del atributo `id` del control asociado.

▼ Texto

▼ CAJA DE TEXTO

Los campos de entrada de texto se crean utilizando la etiqueta `<input>` con el atributo `type="text"`.

1. **Atributo** `name`: Define el nombre de la variable.

```
<input type="text" name="nombre" value="" size="30" maxlength="40" />
```

2. **Atributo** `id`: Para asociar etiquetas de texto con el control correspondiente.
3. **Atributo** `value`: Predefinir un valor para el campo.

```
<input type="text" name="email" value="test@mail.com" size="40" maxlength="40" />
```

4. **Atributo** `size`: Determina el ancho visual del campo (en caracteres).
5. **Atributo** `maxlength`: Establece la longitud máxima permitida para el texto ingresado.

▼ PASSWORD

`<input type="password">` se utiliza para crear una caja de texto en la que los usuarios pueden ingresar contraseñas o información confidencial → el contenido ingresado en esta caja se oculta en el navegador.



Es importante destacar que **no** se trata de cifrado de información en sí mismo.

```
<p>Clave: <input type="password" name="clave" value="" size="20" maxlength="20">
```

▼ ÁREA DE TEXTO

`<textarea>` se utiliza para crear una caja de texto multilinea en formularios HTML → ingresar texto más extenso y formateado.

- Especificar el número de columnas (`cols`) y filas (`rows`)
- El texto dentro de la etiqueta `<textarea>` se muestra como contenido inicial en la caja de texto.

```
<textarea name="dir" cols="50" rows="6">
ETSINF
Edificio 1E
Universidad Politécnica de Valencia
Camino de Vera S/N
46019 Valencia
Spain
</textarea>
```

▼ OCULTO

`<input type="hidden">` se utiliza para agregar información adicional a un formulario sin que el usuario la vea en la interfaz → se envía al servidor cuando se envía el formulario.

Usos comunes:

- **Autenticación:** Puede utilizarse para almacenar tokens de autenticación o información de sesión.
- **Trazabilidad:** Permite rastrear la fuente de los datos enviados desde el formulario.

- **Valores predeterminados:** Puede establecer valores iniciales para campos específicos sin mostrarlos al usuario.

```
<input type="hidden" name="token" value="abc123">
```

▼ Botones de opción

`<input type="radio">` se utiliza para crear una lista de **opciones excluyentes** en un formulario → Los usuarios pueden seleccionar una sola opción de la lista.

- **Atributo `name`:** Todas las opciones excluyentes deben tener el mismo atributo `name`.
- **Atributo `value`:** Cada opción tiene un atributo `value` que contiene el valor aplicable.

```
<p>Género:<br />
<input type="radio" name="genero" value="hombre" />Hombre
<input type="radio" name="genero" value="mujer" />Mujer</p>
<input type="radio" name="genero" value="mujer" checked="checked" />Mujer
```

▼ Cajas de verificación

`<input type="checkbox">` Permiten a los usuarios seleccionar una o más opciones de una lista.

```
<p><input type="checkbox" name="interesado" checked="checked"
value="yes" />Interesado en DEW</p>

<p>Deportes de interés<br />
<input type="checkbox" name="deporte" value="0" />Sillonbol<br />
<input type="checkbox" name="deporte" value="Loco" />Salto base<br />
<input type="checkbox" name="deporte" value="comorl" />Cricket</p>
```

▼ Listas

`<select name="..." size="..." multiple="...">` Utilizados para crear menús desplegables o listas combinadas.

1. **Atributo `size`:** Permite controlar la cantidad de elementos visibles al usuario. Puede mostrar una lista con barras de desplazamiento.
2. **Atributo `multiple`:** Permite la selección de varios elementos a la vez. Los usuarios pueden mantener presionada la tecla **CONTROL**.
3. **Cada elemento se cita con `<option>`**
4. **Agrupación con `<optgroup>`:** Se pueden agrupar opciones relacionadas utilizando la etiqueta `<optgroup>`.

```
<select size="4" name="selDia">
  <option value="Lun">Lunes</option>
  <option value="Mar">Martes</option>
  <option value="Mie">Miercoles</option>
  <option value="Jue">Jueves</option>
  <option value="Vie">Viernes</option>
  <option value="Sab">Sabado</option>
  <option value="Dom">Domingo</option>
</select>
```

▼ Botones (input, button)

▼ INPUT

Botón normal (`<input type="button" ... />`):

- Este botón se utiliza para ejecutar un **script JavaScript** cuando se presiona.

- No está asociado directamente con el envío de formularios.

Botón de envío (`<input type="submit" />`):

- Este botón fuerza el envío del formulario al servidor cuando se presiona.
- Es útil para enviar datos ingresados por el usuario a través del formulario.

Botón de reinicio (`<input type="reset" />`):

- Al presionar este botón, se restablecen todos los campos del formulario a sus valores originales.
- Es útil cuando se desea borrar los datos ingresados sin enviar el formulario.

Botón de imagen (`<input type="image" src="..." />`):

- Similar al botón normal, pero se muestra como una imagen en lugar de texto.
- También puede ejecutar un **script JavaScript** cuando se presiona.

▼ BUTTON

Botón normal (`<button type="..." />`):

- Este botón tiene un contenido específico que aparece como título del botón.
- Puede contener texto o incluso elementos HTML e imágenes.

```
<button type="submit">Adiós</button>
<button type="reset"><strong>Reiniciar?</strong> (asegúrate)</button>
<button type="button"></button>
```

▼ Ficheros

`<input type="file" .../>` Permiten a los usuarios seleccionar uno o más archivos desde su dispositivo para transferirlos (subirlos) al servidor.

- **Atributo** `accept` : Define los tipos MIME (formatos) de archivo que el campo de entrada debe aceptar.
- **Envío de datos en codificación** `multipart/form-data` : El formulario se envía al servidor utilizando la codificación `multipart/form-data`.

```
<form action="/subirImg" method="post" enctype="multipart/form-data">
  <input type="file" name="fileUpload" accept="image/*" />
  <br /><input type="submit" />
</form>
```

▼ Conjuntos de campos

`<fieldset>` `<legend>` Se utilizan para agrupar campos dentro de un marco con un título.

```
<fieldset>
  <legend><em>Información</em></legend>
  Nombre: <input type="text" name="nombre" size="20" /><br />
  Apellidos: <input type="text" name="apellidos" size="20" />
</fieldset>
```

▼ Validaciones

Para especificar características que los datos introducidos por el usuario deben cumplir. Se indican mediante nuevos atributos o nuevos valores para el atributo **type**.

- **Atributo** `required` : Cuando se aplica al elemento **input** indica que el campo es obligatorio.
 - Si un usuario intenta enviar el formulario sin completar ese elemento, el navegador mostrará una advertencia.

```
<input type="text" required="required" ... />
```

- **Atributo `placeholder`** : Para colocar un mensaje (no un valor) dentro de un campo de entrada. Proporciona instrucciones o una pista sobre el tipo de información que se espera del usuario.

```
<input type="text" placeholder="Debes dar un valor" />
```

- **Nuevos valores para el atributo `type`** : HTML5 introdujo nuevos valores para el atributo **type** que pueden combinarse con **required** y otros atributos.
 - Cuando se utiliza **type="email"**, el navegador verifica que el valor ingresado tenga un formato de dirección de correo electrónico válido

```
<input type="email" />
```

- **Atributo `pattern`** : Permite mejorar el control sobre los datos introducidos por el usuario.

```
<input type="url" pattern="https?://.+" ... />
```

- El argumento de **pattern** es una expresión regular en JavaScript.
- Para lograr un control más específico, se puede desarrollar código JavaScript personalizado que cumpla con las propiedades deseadas.

▼ Layout – Diseño y Estructura

▼ DISEÑO

En HTML podemos identificar dos tipos de elementos:

EN LÍNEA (inline)

- Ocupan el espacio estrictamente necesario allí donde se definen
- ` <i> <u> ...`

EN BLOQUE (block-level)

- Ocupan todo el ancho del documento y aparece una línea en blanco delante y otra detrás del elemento
- `<p> <h1> ...`



Un elemento en línea **no puede contener** elementos en bloque

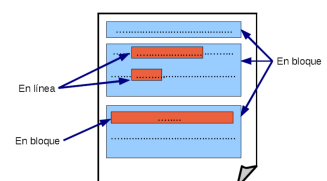
Un elemento en bloque

puede contener cualquier tipo y número de elementos (en línea o en bloque)

```
<h1>Elementos en bloque</h1>
<p>Los <strong>elementos en bloque</strong> siempre emp
<code>&lt;p></code> no aparecerán en la misma l&iacu
```

Elementos en bloque

Los **elementos en bloque** siempre empiezan en una nueva línea. Los elementos `<h1>` y `<p>` no aparecerán en la misma línea, mientras que los elementos en línea fluyen con el resto del texto.



▼ span

Permite agrupar varios elementos en línea sin afectar al renderizado de sus componentes. Para marcar un conjunto de elementos que **comparten una característica común**.

- Herramienta útil para aplicar estilos específicos a partes específicas del contenido en línea.

```
<p>La relaci&ocute;n obtenida es <span class="importante">e=mc<sup>2</sup></span></p>
```

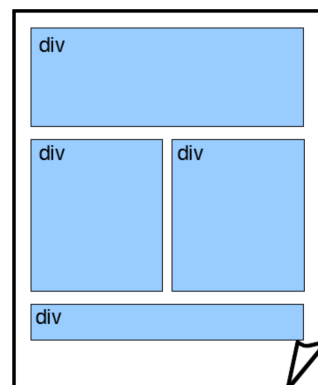
La relación obtenida es **E=MC²**, con implicaciones en ...

▼ div

Elemento de nivel de bloque que puede **agrupar cualquier número y tipo de elementos**.

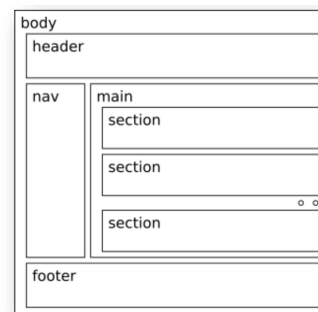
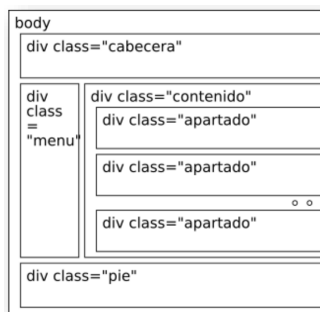
Se utiliza para **crear secciones o subsecciones en un documento**. No afecta al renderizado de sus componentes.

- Una vez diferenciados, se les puede aplicar un estilo específico desde CSS.
- Uso:** A menudo se utiliza para marcar secciones de un documento.

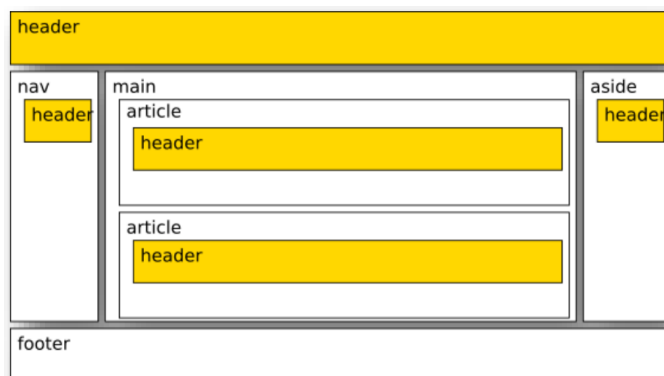


▼ ESTRUCTURA

HTML5 incorpora **elementos que reemplazan y especializan los "divs anónimos"**.

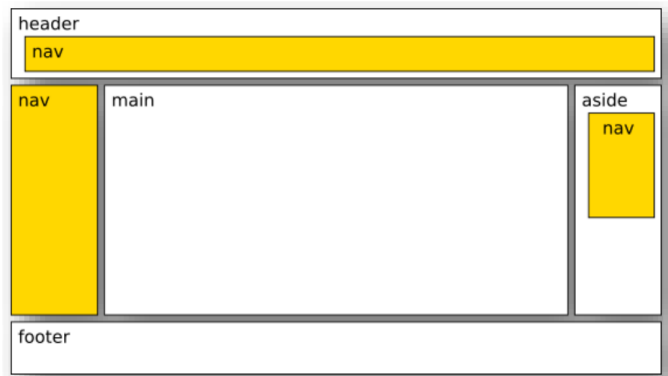


- <header>**: Representa el **encabezado o contenido inicial del elemento** que lo contiene. (Puede contener otros elementos particulares, como título, texto de introducción).

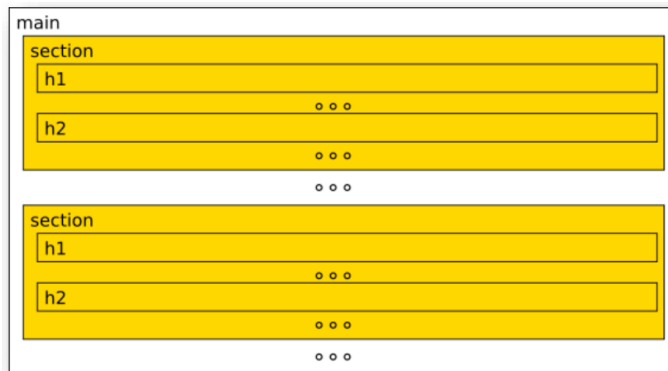


- <nav>**: Proporciona enlaces de navegación, ya sea dentro del

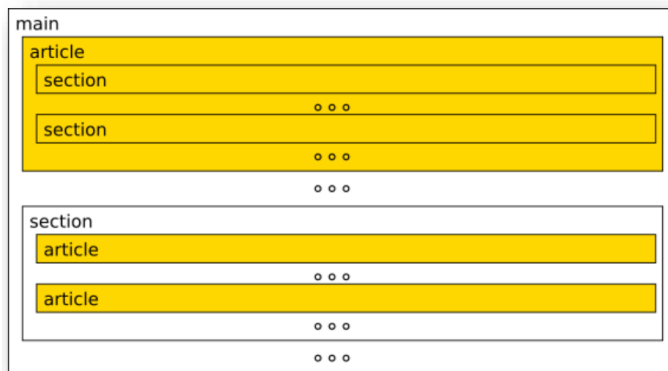
documento actual o a otros documentos.



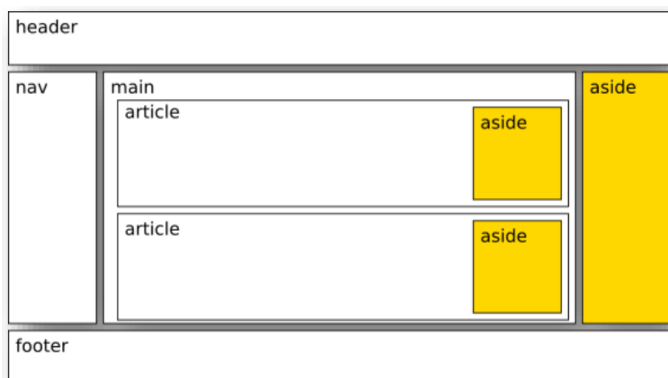
- **<section>**: Define una sección en un documento. Las secciones deben tener siempre un encabezado. Puede contener otros elementos **<section>**.



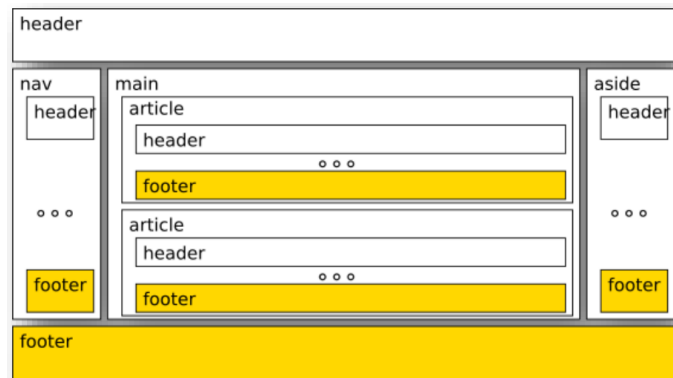
- **<article>**: Define contenido que es autónomo e independiente del resto del contenido del documento. Representa una sección que puede ser considerada como un artículo independiente y completo por sí mismo. Cada **<article>** debe ser identificado, normalmente con un encabezado. Puede contener otros elementos **<article>**. Muchas veces es intercambiable por **<section>**.




- **<aside>**: Agrupa contenido que es secundario al contenido principal al que acompaña. Puede contener un bloque de anuncios, un grupo de enlaces externos, o información adicional como una biografía del autor. Generalmente, se muestra al lado del contenido principal, como una barra lateral.



- `<footer>` : Define el pie de página de un documento o sección. Suele incluir información general como datos del autor, información de contacto, derechos de autor, etc. Se puede tener varios elementos `<footer>` en un documento, y es común que se coloquen al final del contenido que representan.



 [tema 3 ~ DEW](#)