

## TEMA 2

# HTML Y CSS

APLICACIONES WEB - GIS - CURSO 2017/18



Esta obra está bajo una  
Licencia CC BY-NC-SA 4.0 Internacional.

**Manuel Montenegro** [montenegro@fdi.ucm.es]  
Dpto de Sistemas Informáticos y Computación  
Facultad de Informática  
Universidad Complutense de Madrid

1. **INTRODUCCIÓN. PÁGINA BÁSICA**
2. **ELEMENTOS HTML**
3. **ATRIBUTOS CSS**
4. **FLUJO Y POSICIONAMIENTO**
5. **ELEMENTOS HTML5**
6. **FORMULARIOS**
7. **BIBLIOGRAFÍA**



# INTRODUCCIÓN

Abordaremos dos lenguajes para la creación de páginas web

- **HTML**: (*HyperText Markup Language*)  
Determina la estructura de las páginas web.
- **CSS**: (*Cascading Style Sheets*)  
Determina la presentación de las páginas web.

Estándares mantenidos por el *World Wide Web Consortium*  
([W3C](#))

# UN DOCUMENTO HTML SENCILLO

Creamos un fichero `primera_pagina.html` con el siguiente contenido:

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>Mi primera página web</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <h1>¡Bienvenido!</h1>
    <p>Esto es una página web</p>
  </body>
</html>
```

Abrimos el fichero HTML en el navegador web:



# ESTRUCTURA DE UN DOCUMENTO HTML

- Tipo de documento

```
<!DOCTYPE html>
```

- Cabecera

```
<head>  
  <title>Mi primera página web</title>  
  <meta charset="utf-8"/>  
</head>
```

- Cuerpo

```
<body>  
  <h1>¡Bienvenido!</h1>  
  <p>Esto es una página web</p>  
</body>
```

## TIPO DE DOCUMENTO

```
<!DOCTYPE html>
```

Indica que es un tipo de fichero HTML.

A partir de la versión 5 del estándar de HTML no se indica qué versión del lenguaje HTML se va a utilizar en la página.

En el estándar antiguo (HTML 4.01):

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

# ELEMENTOS HTML

Una página HTML consta de varios **elementos**, que pueden estar anidados unos dentro de otros.

Cada elemento está delimitado por una etiqueta de inicio (*<nombreEtiqueta>*) y una etiqueta de fin (*</nombreEtiqueta>*)

El nombre de la etiqueta determina el tipo de elemento.

Si el contenido de la etiqueta está vacío se puede escribir *<nombreEtiqueta/>* en lugar de *<nombreEtiqueta><nombreEtiqueta/>*



Las etiquetas de inicio y fin han de estar correctamente anidadas:

- Correcto:

```
<p>...<strong>...</strong>...</p>
```

Indica que hay un elemento de tipo **<strong>** anidado dentro de un elemento de tipo **<p>**.

- Incorrecto:

```
<p>...<strong>...</p>...</strong>
```

El elemento `<html>` contiene toda la información sobre la estructura de la página web. Puede incluir un atributo `lang` con el idioma del contenido de la página web.

Contiene, a su vez, dos elementos: `<head>` y `<body>`

## CABECERA (<head>)

Contiene información sobre la página web que es ajena al propio contenido. En nuestro caso:

```
<title>Mi primera página web</title>
```

Indica el título de la página. Este título suele aparecer en la pestaña del navegador

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

Indica que se utilizará la codificación *Unicode* (UTF-8) en el código HTML. [\[+\]](#)

## CUERPO (<body>)

Especifica el contenido de la página web.

En nuestro caso contiene:

- Un encabezado (elemento <h1>)

```
<h1>¡Bienvenido!</h1>
```

**Bienvenido!**

- Un párrafo (elemento <p>)

```
<p>Esto es una página web</p>
```

Esto es una página web

# ENCABEZADOS HTML

Se utilizan los elementos `<h1>`, `<h2>`, ..., `<h6>`.

`<h1>` representa el encabezado de mayor nivel (el más "importante"). El resto especifican encabezados de importancia decreciente.

# EJEMPLO

```
<h1>Encabezado H1</h1>  
<h2>Encabezado H2</h2>  
<h3>Encabezado H3</h3>  
<h4>Encabezado H4</h4>  
<h5>Encabezado H5</h5>  
<h6>Encabezado H6</h6>
```

**Encabezado H1**

**Encabezado H2**

**Encabezado H3**

**Encabezado H4**

**Encabezado H5**

**Encabezado H6**

# PÁRRAFOS

Cada párrafo está contenido dentro de un elemento `<p>`

```
<h1>Párrafos</h1>
```

```
<p>
```

Esto es un ejemplo de párrafo que puede extenderse a lo largo de varias líneas.

```
</p>
```

```
<p>
```

Esto de aquí es otro párrafo completamente distinto. El navegador puede insertar un pequeño espacio de separación entre los dos párrafos.

```
</p>
```

## Párrafos

Esto es un ejemplo de párrafo que puede extenderse a lo largo de varias líneas.

Esto de aquí es otro párrafo completamente distinto. El navegador puede insertar un pequeño espacio de separación entre los dos párrafos.

# ELEMENTOS DE TEXTO

Afectan al formato del texto que contienen.

- `<em>` sirve para enfatizar un texto.  
El texto suele mostrarse en *cursiva*.
- `<strong>` sirve para resaltar aun más un texto.  
Los navegadores suelen mostrarlo en **negrita**.
- `<code>` se utiliza para mostrar el código de un programa.  
Los navegadores suelen mostrarlo en letra de tipo monoespaciada.



## EJEMPLO

Decimos que una función es de **orden superior** si acepta otras funciones como parámetro y/o devuelve funciones como resultado. Entre estas funciones podemos citar `map`, `filter` y la familia de funciones de *folding*.

Decimos que una función es de **orden superior** si acepta otras funciones como parámetro y/o devuelve funciones como resultado. Entre estas funciones podemos citar `map`, `filter` y la familia de funciones de *folding*.

# COMENTARIOS

En HTML van delimitados por las secuencias de símbolos  
<!-- y -->

```
<p>Esto es un párrafo</p>  
<!-- Todo lo que hay en este comentario se ignorará, incluyendo  
      etiquetas <em>HTML</em> -->  
<p>Esto es otro párrafo</p>
```

Esto es un párrafo

Esto es otro párrafo

# CARACTERES ESPECIALES

Los caracteres `<`, `>`, `&`, etc. tienen un significado especial en HTML. Si se quieren utilizar como parte del contenido HTML se han de utilizar sus correspondientes **entidades**.

Una entidad comienza por `&` y termina por `;`

La entidad `&#num;` representa el carácter Unicode con código *num*. Por ej. `&#8704;` representa  $\forall$

Entidad	Carácter
<code>&amp;lt;</code>	<code>&lt;</code>
<code>&amp;gt;</code>	<code>&gt;</code>
<code>&amp;amp;</code>	<code>&amp;</code>
<code>&amp;quot;</code>	<code>"</code>
<code>&amp;euro;</code>	<code>€</code>

# ESPACIOS Y SALTOS DE LÍNEA

HTML interpreta las secuencias de espacios como un único espacio. Los saltos de línea se interpretan como espacios.

```
Esto    es una    línea    con varios    espacios.
```

```
Aquí va otro texto
```

```
fragmentado en varias  
líneas.
```

Esto es una línea con varios espacios. Aquí va otro texto fragmentado en varias líneas.

# ESTRUCTURA VS. ESTILO

Todas las etiquetas de HTML solamente definen la **estructura** del contenido de la página, no su aspecto visual.

El texto delimitado por algunas etiquetas (como `<strong>`) puede aparecer con un determinado formato en el navegador, pero dicho formato no es único para todos los navegadores web.

La apariencia visual de una página se define mediante el lenguaje CSS, y suele definirse en un archivo distinto al documento HTML.

Mismo documento HTML mostrado con distintos estilos:

## Funciones de orden superior

Decimos que una función es de **orden superior** si acepta otras funciones como parámetro y/o devuelve funciones como resultado. Entre estas funciones podemos citar `map`, `filter` y la familia de funciones de *folding*.

## Funciones de orden superior

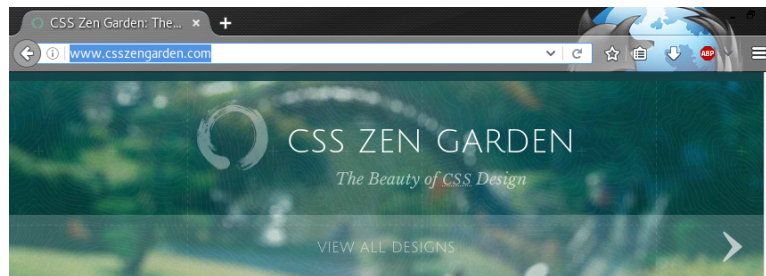
Decimos que una función es de **orden superior** si acepta otras funciones como parámetro y/o devuelve funciones como resultado. Entre estas funciones podemos citar `map`, `filter` y la familia de funciones de *folding*.

## *Funciones de orden superior*

Decimos que una función es de **orden superior** si acepta otras funciones como parámetro y/o devuelve funciones como resultado. Entre estas funciones podemos citar `map`, `filter` y la familia de funciones de *folding*.

<http://www.csszengarden.com/>

Distintos estilos aplicados a una misma página web.



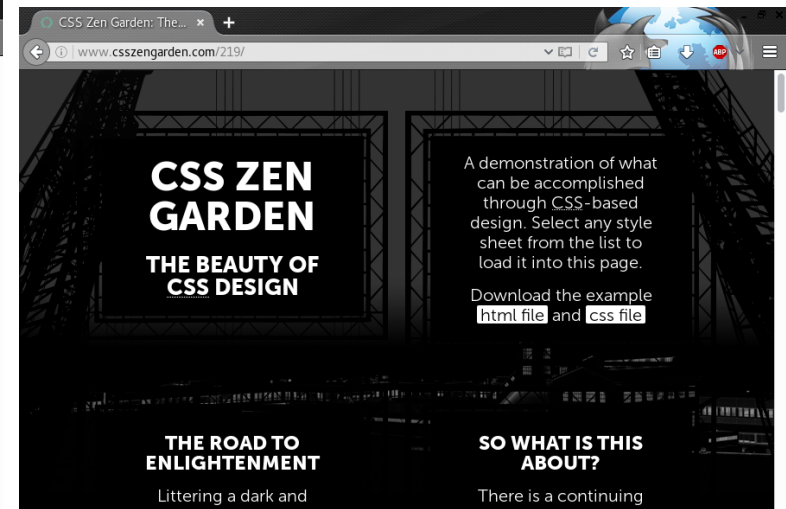
A demonstration of what can be accomplished through CSS-based design. Select any style sheet from the list to load it into this page.

Download the example [HTML FILE](#) and [CSS FILE](#)

### THE ROAD TO ENLIGHTENMENT

Littering a dark and dreary road lay the past relics of browser-specific tags, incompatible DOMs, broken CSS support, and abandoned browsers.

We must clear the mind of the past. Web enlightenment has been achieved thanks to the tireless





# HOJA DE ESTILO CSS

Documento que contiene las definiciones de estilo para una o varias páginas web.

Creamos un fichero **hoja1.css** con el siguiente contenido:

```
h1 {  
    color: blue;  
    text-decoration: underline;  
}  
  
strong {  
    color: blue;  
    font-weight: bold;  
}
```

```
h1 {  
  color: blue;  
  text-decoration: underline;  
}
```

Indica que todos los elementos con la etiqueta **<h1>** aparecerán subrayados y en color azul.

```
strong {  
  color: blue;  
  font-weight: bold;  
}
```

Indica que todos los elementos con la etiqueta **<strong>** aparecerán en negrita y en color azul.

# LA ETIQUETA `<link>`

La etiqueta `<link>` permite asociar una hoja de estilo CSS con un documento HTML.

Debe estar en la sección `<head>` del documento.

```
<head>
...
<link rel="stylesheet" href="Hoja1.css"/>
...
</head>
```

El atributo `href` indica el nombre del fichero que contiene la hoja de estilo.

El atributo `rel` indica la relación que tiene el fichero referenciado mediante `href` y el documento HTML actual (*hoja de estilo*).

# EJEMPLO

```
<!DOCTYPE html>
<html>
  <head>
    <title>Párrafos</title>
    <meta charset="utf-8"/>
    <link rel="stylesheet" href="Hoja1.css"/>
  </head>
  <body>
    <h1>Funciones de orden superior</h1>
    <p>Decimos que una función es de
    <strong>orden superior</strong> si acepta otras funciones
    como parámetro y/o devuelve funciones como resultado.
    Entre estas funciones podemos citar <code>map</code>,
    <code>filter</code> y la familia de funciones de
    <em>folding</em>.</p>
  </body>
</html>
```

## Funciones de orden superior

Decimos que una función es de **orden superior** si acepta otras funciones como parámetro y/o devuelve funciones como resultado. Entre estas funciones podemos citar `map`, `filter` y la familia de funciones de *folding*.

Creamos otro fichero **Hoja2.css** con otra hoja de estilo:

```
body {  
    font-family: sans-serif;  
    background-color: #FFF0F0;  
}  
  
h1 {  
    border-left: 5px solid #006000;  
    padding-left: 10px;  
    color: #006000;  
    background: #E0F0E0;  
}  
  
code {  
    font-family: "Inconsolata", monospace;  
    color: #006000;  
    background: #E0E0E0;  
    padding-left: 2px;  
    padding-right: 2px;  
}
```

En la página anterior, cambiamos la declaración `<link>` por la siguiente:

```
<link rel="stylesheet" href="Hoja2.css"/>
```

Resultado:

## | Funciones de orden superior

Decimos que una función es de **orden superior** si acepta otras funciones como parámetro y/o devuelve funciones como resultado. Entre estas funciones podemos citar `map`, `filter` y la familia de funciones de *folding*.

1. INTRODUCCIÓN. PÁGINA BÁSICA
2. **ELEMENTOS HTML**
3. ATRIBUTOS CSS
4. FLUJO Y POSICIONAMIENTO
5. ELEMENTOS HTML5
6. FORMULARIOS
7. BIBLIOGRAFÍA



# ELEMENTOS HTML BÁSICOS

- Hipervínculos: `<a>`
- Citas: `<q>`, `<blockquote>`
- Saltos de línea: `<br>`
- Listas: `<ol>`, `<ul>`, ...
- Preformateado: `<pre>`
- Imágenes: `<img>`
- Tablas: `<table>`, `<tr>`, `<td>`, ...



# HIPERVÍNCULOS

Mediante la etiqueta `<a>` podemos especificar enlaces a otras páginas o recursos.

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

Al hacer clic en el texto delimitado por la etiqueta `<a>`, el navegador cargará la página indicada por el atributo `href`.

# EJEMPLO

En el fichero **Enlace1.html**:

```
<p>Pulsa en <a href="Enlace2.html">este enlace</a> para acceder  
a la página número dos.</p>
```

Pulsa en [este enlace](#) para acceder a la página número dos.

En el fichero **Enlace2.html**:

```
<p>Estás en la página dos.</p>  
<p><a href="Enlace1.html">Volver a la página uno.</a></p>
```

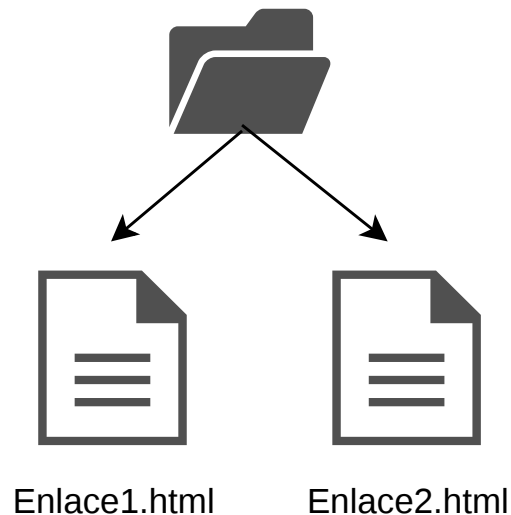
Estás en la página dos.

[Volver a la página uno.](#)

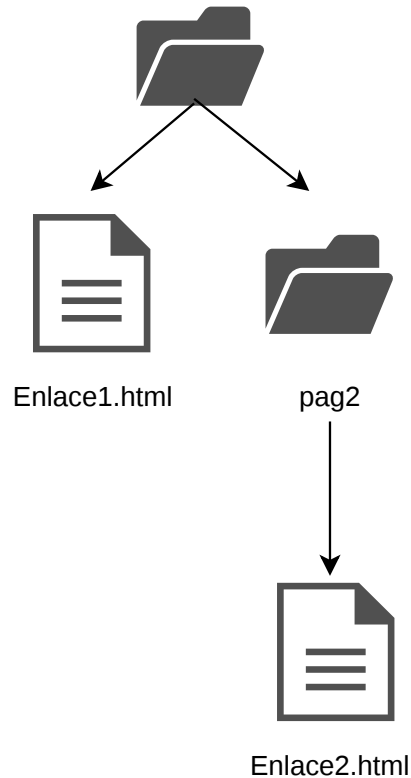
# ENLACES RELATIVOS Y ABSOLUTOS

```
<a href="Enlace2.html">...</a>
```

Se carga la página **Enlace2.html** contenida en el mismo directorio que la página que contiene el enlace **<a>**.



Se trata de un enlace **relativo**



Si tuviésemos el fichero **Enlace2.html** dentro de una carpeta **pag2** tendríamos el siguiente enlace:

```
<a href="pag2/Enlace2.html">...</a>
```

Si queremos enlazar desde **Enlace2.html** a **Enlace1.html**, tendremos:

```
<a href=" ../Enlace1.html">...</a>
```

Los enlaces que comiezan por `/` son relativos al dominio a través del cual se accede a la página.

Por ejemplo, si en `http://www.foo.org/bar/pag1.html` tenemos el siguiente enlace:

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

Se enlace a la URL `http://www.foo.org/index.html`

Si se accede a una página almacenada en el sistema de ficheros el enlace será relativo al directorio raíz.

Los enlaces **absolutos** contienen una URL completa:

```
<a href="http://www.ucm.es">Página de la UCM</a>
```

¿Qué ocurre con el siguiente enlace?

```
<a href="www.ucm.es">Página de la UCM</a>
```

Para que un enlace sea absoluto se ha de incluir la URL, incluyendo el protocolo **http://**, **https://**, etc.

# IDENTIFICADORES

Es posible asociar un identificador a cualquier elemento de un documento HTML.

Para ello se utiliza el atributo **id**.

```
<h1 id="sec1">Sección 1</h1>
```

El identificador permite hacer referencia al elemento:

- Desde un hipervínculo.
- Desde una regla CSS.
- Desde código Javascript.
- ...

Referenciar a un elemento de la página actual desde un hipervínculo: **#id**

```
<a href="#sec1">Ir a la sección 1</a>
```

Al hacer clic en el enlace, el navegador saltará al elemento que tiene **sec1** como identificador.

Para hacer referencia a un elemento contenido en otra página:

```
<a href="pagina2.html#sec21">Ir a la sección de otra página</a>
```

Ejemplo: [Documentación de la API de Java](#)



## OTROS TIPOS DE URL

- **mailto:** Para direcciones de correo electrónico.

```
<a href="mailto:mmontene@ucm.es">Más información</a>
```

- **tel:** Para número de teléfonos (navegadores móviles).

```
<a href="tel:+34987654132">Llamar</a>
```

# CITAS

El elemento `<q>` se utiliza para introducir citas textuales:

```
<p>Como dijo Javier Leach: <q>Esto es tan sencillo que  
en vez de tautología es tontología</q>.</p>
```

Como dijo Javier Leach: “Esto es tan sencillo que en vez de tautología es tontología”.

Si la cita es grande, **<blockquote>** inserta la cita en su propio párrafo:

```
<p>Lo siguiente es una cita de Richard Stallman:</p>
```

```
<blockquote>
```

```
We need to encourage the spirit of cooperation, by respecting  
other people's freedom to cooperate and not advancing schemes  
to divide and dominate them.
```

```
</blockquote>
```

Lo siguiente es una cita de Richard Stallman:

We need to encourage the spirit of cooperation, by  
respecting other people's freedom to cooperate  
and not advancing schemes to divide and dominate  
them.

# SALTOS DE LÍNEA

El elemento `<br>` introduce un salto de línea en el texto.

```
<blockquote>
Volverán las oscuras golondrinas<br>
en tu balcón sus nidos a colgar,<br>
y otra vez con el ala a sus cristales<br>
jugando llamarán.
</blockquote>
```

Volverán las oscuras golondrinas  
en tu balcón sus nidos a colgar,  
y otra vez con el ala a sus cristales  
jugando llamarán.

El elemento `<br>` no tiene etiqueta de cierre.

# LISTAS

Se expresan mediante la etiqueta `<ol>`. Cada elemento de la lista va delimitado por la etiqueta `<li>`.

```
<p>Modo de preparación:</p>
```

```
<ol>
```

```
  <li>Lava y escurre los tomates, el pepino y el pimiento.</li>
```

```
  <li>Introduce en la batidora el pan y los tomates  
    cortados.</li>
```

```
  <li>Quita las semillas al pimiento y ponlo con los  
    tomates.</li>
```

```
  <li>Añadir el ajo picado y la cebolla pelada.</li>
```

```
  <li>Corta el pepino en cuatro o cinco trozos y añadir  
    a la batidora.</li>
```

```
  <li>Batir todo.</li>
```

```
  <li>Añadir la sal, el aceite y vinagre y volver a batir.</li>
```

```
</ol>
```

## Resultado:

### Modo de preparación:

1. Lava y escurre los tomates, el pepino y el pimiento.
2. Introduce en la batidora el pan y los tomates cortados.
3. Quita las semillas al pimiento y ponlo con los tomates.
4. Añadir el ajo picado y la cebolla pelada.
5. Corta el pepino en cuatro o cinco trozos y añadir a la batidora.
6. Batir todo.
7. Añadir la sal, el aceite y vinagre y volver a batir.

El elemento `<ul>` sirve para representar listas no numeradas.

```
<p>Ingredientes:</p>
<ul>
  <li>1 kilo de tomates.</li>
  <li>1 pimiento verde.</li>
  <li>Medio pepino.</li>
  <li>1 rebanada de pan.</li>
  <li>3 cucharadas de aceite de oliva.</li>
  <li>Media cebolla.</li>
  <li>Sal y vinagre</li>
</ul>
```

Ingredientes:

- 1 kilo de tomates.
- 1 pimiento verde.
- Medio pepino.
- 1 rebanada de pan.
- 3 cucharadas de aceite de oliva.
- Media cebolla.
- Sal y vinagre

## Las listas se pueden anidar:

```
<ul>
  <li>1 kilo de tomates.
    <ul>
      <li>Comprados en el mercado, o bien</li>
      <li>Comprados en el súper</li>
    </ul>
  </li>
  ...
</ul>
```

### Ingredientes:

- 1 kilo de tomates.
  - Comprados en el mercado, o bien
  - Comprados en el súper
- 1 pimiento verde.



# ESTILO DE LISTAS

Mediante el atributo CSS `list-style-type` podemos indicar el tipo de viñetas o numeración.

```
ul {  
  list-style-type: square;  
}
```

- 1 kilo de tomates.
- 1 pimiento verde.
- Medio pepino.

```
ul {  
  list-style-type: lower-roman;  
}
```

- a. Lava y escurre los tomates, el pepino y el pimiento.
- b. Introduce en la batidora el pan y los tomates cortados.
- c. Quita las semillas al pimiento y ponlo con los tomates.
- d. Añadir el ajo picado y la cebolla pelada.

Ver: [Tipos disponibles](#)

También pueden expresarse listas con definiciones, a modo de glosario.

- `<dl>` para listas de definiciones.
- `<dt>` para términos.
- `<dd>` para definiciones.

Cada elemento `<dt>` va seguido por un elemento `<dd>`.

```
<dl>
  <dt>Homomorfismo</dt>
  <dd>
    Aplicación entre dos espacios vectoriales que preserva la
    estructura de la suma y producto por escalar.
  </dd>
  <dt>Monomorfismo</dt>
  <dd>
    Homomorfismo inyectivo entre dos espacios vectoriales.
  </dd>
  <dt>Isomorfismo</dt>
  <dd>
    Homomorfismo biyectivo entre dos espacios vectoriales
  </dd>
</dl>
```

Homomorfismo

Aplicación entre dos espacios vectoriales que preserva la estructura de las operaciones de suma y producto por escalar.

Monomorfismo

Homomorfismo inyectivo entre dos espacios vectoriales.

Isomorfismo

Homomorfismo biyectivo entre dos espacios vectoriales.

# TEXTO PREFORMATEADO

La etiqueta `<pre>` permite introducir fragmentos de texto sin formato.

```
<pre>
```

```
Esto es un texto preformateado. Se respetan  
totalmente los  
saltos de línea      y los espacios      múltiples
```

```
Aquí otro párrafo.
```

```
</pre>
```

```
Esto es un texto preformateado. Se respetan  
totalmente los  
saltos de línea      y los espacios      múltiples
```

```
Aquí otro párrafo.
```

# IMÁGENES

Se insertan mediante el elemento `<img>`

```
<p>Foto de Münster tomada en septiembre de 2014:</p>  

```



Las consideraciones sobre paths absolutos y relativos también se aplican al atributo `src`.

Es conveniente incluir el atributo **alt** con una breve descripción de la imagen para personas con ceguera total o parcial, dispositivos con conexión lenta, etc.

```

```

Es posible incluir imágenes dentro de enlaces:

```
<a href="http://www.muenster.de">  
    
</a>
```

El navegador saltará al enlace al hacer clic en la imagen.

Se puede ajustar la anchura y altura de la imagen en la página mediante los atributos CSS **width** y **height**.

```
<a href="http://www.muenster.de">  
    
</a>
```

Más información: [CSS en línea](#)

# TABLAS

Sirven para representar datos en forma tabular.

Las tablas se delimitan con la etiqueta `<table>`, que a su vez contiene:

- `<caption>`, para el título de la tabla.
- `<tr>`, para delimitar filas.
- `<th>`, para delimitar encabezados.
- `<td>`, para delimitar celdas.

Las tablas **no** deben utilizarse para maquetar una página.



## ESTRUCTURA DE UNA TABLA

`<table>`

`<caption>...</caption>`

<code>&lt;tr&gt;</code>	<code>&lt;th&gt;...&lt;/th&gt;</code>	<code>&lt;th&gt;...&lt;/th&gt;</code>	<code>&lt;th&gt;...&lt;/th&gt;</code>	<code>&lt;/tr&gt;</code>
<code>&lt;tr&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;/tr&gt;</code>
<code>&lt;tr&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;/tr&gt;</code>
<code>&lt;tr&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;/tr&gt;</code>

`</table>`

El título (`<caption>`) es opcional.

# EJEMPLO

```
<table>
  <tr>
    <th></th>
    <th>Exp. 1</th>
    <th>Exp. 2</th>
  </tr>
  <tr>
    <td>Temperatura</td>
    <td>280.5 K</td>
    <td>310.4 K</td>
  </tr>
  <tr>
    <td>Presión</td>
    <td>1.52 atm</td>
    <td>1.58 atm</td>
  </tr>
</table>
```

	<b>Exp. 1</b>	<b>Exp. 2</b>
Temperatura	280.5 K	310.4 K
Presión	1.52 atm	1.58 atm

Los atributos **rowspan** y **colspan** permiten que una celda se extienda más allá de una fila o columna. Reciben el número de filas o columnas que ocupa la celda.

<code>&lt;th&gt;...&lt;/th&gt;</code>	<code>&lt;th&gt;...&lt;/th&gt;</code>	<code>&lt;th&gt;...&lt;/th&gt;</code>
<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td colspan="2"&gt;...&lt;/td&gt;</code>	
<code>&lt;td rowspan="3"&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>
	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>
	<code>&lt;td&gt;...&lt;/td&gt;</code>	<code>&lt;td&gt;...&lt;/td&gt;</code>

# ESTILO DE TABLAS

Mediante CSS podemos dar estilo al título, encabezados, celdas y bordes de la tabla.

```
td, th {  
  border: 2px solid black;  
}
```

Borde alrededor de cada celda

```
th {  
  background: gray;  
  color: white;  
}
```

Color de las celdas cabecera

	Exp. 1	Exp. 2
Temperatura	280.5 K	310.4 K
Presión	1.52 atm	1.58 atm

Por defecto, cada celda tiene un borde totalmente separado de las celdas colindantes.

	Exp. 1	Exp. 2
Temperatura	280.5 K	310.4 K
Presión	1.52 atm	1.58 atm

Podemos fusionar todos los bordes mediante la propiedad **border-collapse:**

```
table {  
  border-collapse: collapse;  
}
```

	Exp. 1	Exp. 2
Temperatura	280.5 K	310.4 K
Presión	1.52 atm	1.58 atm

1. INTRODUCCIÓN. PÁGINA BÁSICA
2. ELEMENTOS HTML
3. **ATRIBUTOS CSS**
4. FLUJO Y POSICIONAMIENTO
5. ELEMENTOS HTML5
6. FORMULARIOS
7. BIBLIOGRAFÍA



# ATRIBUTOS CSS

La estructura básica de un fichero CSS es la siguiente:

```
selector1 {  
    propiedad_1: valor_1;  
    propiedad_2: valor_2;  
    ...  
}  
  
selector2 {  
    propiedad_3: valor_3;  
    propiedad_4: valor_4;  
    ...  
}  
  
...
```

Regla 1

Regla 2

## Cada regla CSS contiene:

- Un **selector**, que determina los elementos a los que afecta el estilo definido.
- Una lista de **propiedades** y **valores**, que definen el estilo propiamente dicho.

Si un mismo conjunto de propiedades/valores se aplican a selectores distintos, pueden fusionarse en una única regla:

```
selector1, selector2, ... {  
    propiedad_1: valor_1;  
    propiedad_2: valor_2;  
    ...  
}
```



# SELECTORES CSS

Ya conocemos el selector más sencillo: un nombre de etiqueta

```
p {  
  color:blue;  
}
```

Esta regla se aplica a todos los elementos `<p>` del documento HTML.

# CLASES

Es posible asignar uno o varios nombres de clase a cualquier elemento del documento HTML. Esto se consigue mediante el atributo **class**.

```
<a href="pag.html" class="rojo">Enlace</a>
```

```
<strong class="rojo resaltado">ATENCIÓN</a>
```

Las clases se utilizan para asignar un mismo estilo a distintos elementos:

```
.rojo {  
  color: red;  
}
```

Afecta a todos los elementos con la clase **rojo**

```
.resaltado {  
  text-decoration: underline;  
  background: gray;  
}
```

Afecta a todos los elementos con la clase **resaltado**

```
.rojo { ... }
```

Afecta a todos los elementos de la clase **rojo**.

```
p.rojo { ... }
```

Afecta a todos los elementos **<p>** de la clase **rojo**.

```
#ident { ... }
```

Afecta al elemento con identificador **ident**.

Por ejemplo: **<p id="ident">...</p>**

## CLASES VS. IDENTIFICADORES

- Un identificador es **único** en todo el documento HTML. No deben existir dos elementos con el mismo identificador.
- Sin embargo, pueden existir dos elementos con la misma clase.

Las clases se utilizan para asignar un mismo estilo a distintos elementos de la página web.

```
* { ... }
```

Afecta a todos los elementos.

```
p.miclase a { ... }
```

Afecta a todos los enlaces (<a>) contenidos en un párrafo (<p>) de clase **miclase**, ya sea directa o indirectamente a través de otro elemento.

```
p.miclase > a { ... }
```

Afecta a todos los enlaces (<a>) contenidos *directamente* en un párrafo (<p>) de clase **miclase**.

```
table + p { ... }
```

Afecta a todos los párrafos (<p>) situados *a continuación* de una tabla (<table>).

```
img[src="icono.png"] { ... }
```

Elementos `<img>` que tengan un atributo `src` que contenga el valor `icono.png`.

```
a[href^="https"] { ... }
```

Enlaces cuyo atributo `href` comience por `https`.

```
a[href$=".php"] { ... }
```

Enlaces cuyo atributo `href` termine por `php`.

```
a[href*="server"] { ... } /* Cadena */  
a[href~="server"] { ... } /* Palabra */
```

Enlaces cuyo atributo `href` contenga la cadena (resp. palabra) `server`.

# PSEUDOCCLASES

```
h1:hover { ... }
```

Se aplica un encabezado (`<h1>`) si el ratón está situado sobre él.

```
a:active { ... }  
a:link { ... }  
a:visited { ... }
```

Enlaces *activos* (se ha pulsado sobre ellos), sin visitar y visitados.

```
table > tr:nth-child(even) { ... }  
table > tr:nth-child(odd) { ... }
```

Filas pares (resp. impares) contenidas directamente en un elemento `<table>`.

## PSEUDOELEMENTOS

```
h1.nuevo::after {  
  content: " [Nuevo]";  
}
```

Inserta la cadena " [Nuevo]" después de todos los encabezados `<h1>` que tengan la clase `nuevo`.

```
a[href$=".pdf"]::before {  
  content: img(icono_documento.png);  
}
```

Inserta una imagen antes de los enlaces que hagan referencia a un fichero `.pdf`.



## MÁS INFORMACIÓN

- [CSS Selectors reference](http://www.w3schools.com/cssref/css_selectors.asp)  
[http://www.w3schools.com/cssref/css\\_selectors.asp](http://www.w3schools.com/cssref/css_selectors.asp)
- [Try CSS selector](http://www.w3schools.com/cssref/tryssel.asp)  
<http://www.w3schools.com/cssref/tryssel.asp>

# ATRIBUTOS DE COLOR

- **color**: Color del elemento (texto, borde, etc.).
- **background-color**: Color de fondo.

```
a {  
  color: green;  
  background-color: yellow;  
}
```

Esto es un enlace

## ESPECIFICAR COLORES

- Colores predefinidos [+]  
`black`, `red`, `DarkViolet`, etc.
- Notación hexadecimal: `#RRVVAA`.
  - `RR` componente rojo (00..FF)
  - `VV` componente verde (00..FF)
  - `AA` componente azul (00..FF)

Números hexadecimales de dos cifras. Su mezcla determina el color resultante.

Ejemplos: `#FF0000`, `#00FF00`, `#0000FF`, `#2E6000`, `#606060`, `#FF4100`, etc.

- Notación decimal: `rgb(r, v, a)`  
Donde `r`, `v` y `a` son números entre 0 y 255, o porcentajes entre 0% y 100%.  
Ejemplos: `rgb(0, 250, 120)`, `rgb(50, 100, 0)`, `color:rgb(50%, 50%, 50%)`, etc.
- Notación decimal con transparencia: `rgba(r, v, a, t)`  
La componente `t` indica la transparencia, desde 0 (transparente) a 1 (completamente opaco).  
Ejemplo: `rgba(0, 250, 120, 0.5)`.

Ver: [Color picker](#)

## EJEMPLO

```
a {  
  color: #00134d;  
  background-color: rgb(221, 204, 255);  
}
```

Esto es un enlace

# GRADIENTES

Se puede utilizar las funciones `linear-gradient`, `radial-gradient` allá donde pueda especificarse un color. Éstas son soportadas por versiones *modernas* de los navegadores:

- Chrome  $\geq 26$
- Firefox  $\geq 16$
- MS Explorer  $\geq 10$

Más información:

[http://www.w3schools.com/css/css3\\_gradients.asp](http://www.w3schools.com/css/css3_gradients.asp)

```
#grad1 {  
    background: linear-gradient(to right, yellow, green);  
}  
  
#grad2 {  
    background: linear-gradient(to right bottom, black,  
                                gray, black);  
}  
  
#grad3 {  
    background: radial-gradient(yellow, white);  
}
```

#grad1

#grad2

#grad3



# BORDES

`border-style:` `none` | `solid` | `dashed` | `dotted` | ...

Indica el estilo de la línea del borde [+].

`border-color:` `#FF0000`

Indica el color de la línea del borde.

`border-width:` `1px` | `3px` | `0.5em` | ...

Indica la anchura de línea. Unidades de medida [+]:

- Absolutas: `3px`, `1cm`, `4mm`, `2in`, ...
- Relativas: `2em`, `50%`, ...

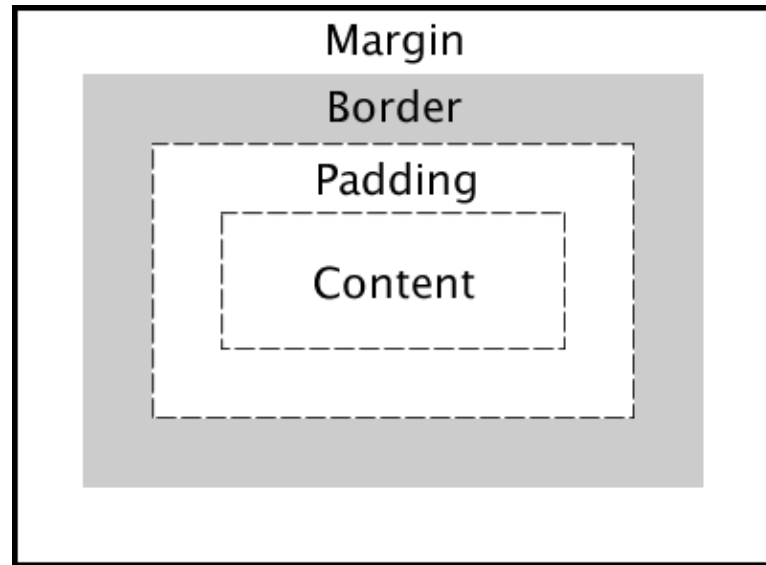
Pueden agruparse en una única propiedad **border**:

```
border: 2px dashed cyan;
```

Es posible especificar el estilo de cada borde por separado:

```
border-top: 1px solid white;  
border-bottom: 2px dashed black;  
border-left-style: dotted;  
border-bottom-color: #FF0000;
```

# MODELO DE CAJAS CSS



(Fuente: [University of Washington](#))

- **Margin:** espacio del borde con respecto a los demás elementos.
- **Padding:** espacio del contenido del elemento con respecto al borde.

# PROPIEDADES MARGIN Y PADDING

```
margin: 2px;  
padding: 3px;
```

Espaciado a todos los lados del elemento.

```
margin: 0px 5px;
```

Espacio de **0px** en lados superior-inferior y **5px** en lados izquierdo-derecho.

```
padding: 0px 5px 1px 2px;
```

**0px** en lado superior, **5px** en derecho, **1px** en lado inferior y **2px** en lado izquierdo.

Puede especificarse el **margin** y **padding** de cada lado individualmente:

```
margin-left: 10px;  
padding-top: 5px;  
margin-right: 1px;  
margin-bottom: 2em;
```

# EJEMPLO

```
<p>Esto es un párrafo normal.</p>
<p class="importante">
  Esto es muy importante!
</p>
<p>Vuelta al párrafo normal.</p>
```

```
p.importante {
  background-color: rgb(255, 200, 200);
  border: 1px dashed rgb(200, 0, 0);
  border-left: 5px solid rgb(200, 0, 0);
  margin-left: 20px;
  padding: 10px 30px;
  font-weight: bold;
}
```

Esto es un párrafo normal.

**Esto es muy importante!**

Vuelta al párrafo normal.

# ATRIBUTOS DEL TEXTO

```
font-family: tipo_de_letra1, tipo_de_letra2, ...
```

Especifica el tipo de letra. Se comprobará la disponibilidad de las fuentes en el orden especificado. Se usará la primera fuente que esté disponible.

Cada tipo de letra especificado puede ser:

- Un nombre **específico**:  
Times New Roman, Georgia, Open Sans, etc.
- Una familia **genérica**:  
serif, sans-serif, monospace, fantasy, etc.

Es muy recomendable que el último tipo de letra especificado en la lista **font-family** sea genérico.

```
font-family: "Open Sans", "Verdana", sans-serif;
```



# INCLUIR FUENTES NUEVAS EN EL CSS

Si disponemos de los ficheros de fuente, podemos definirlos en el CSS para que el navegador utilice dichos ficheros:

```
@font-face {  
  font-family: "Baloo Bhai";  
  src: url("BalooBhai-Regular.ttf")  
}
```

Nombre que asignamos a la fuente

Podemos utilizar esta fuente en el resto del CSS.

```
h1 {  
  font-family: "Baloo Bhai", Arial, sans-serif;  
  font-weight: normal;  
}
```

```
<h1>Hola!</h1>
```

**Hola!**

## OTROS ATRIBUTOS DE TEXTO

```
font-size: medium | small | large | ... | 20px | 12pt | 95% | ...
```

Tamaño de la letra.

```
font-style: normal | italic | oblique | ...
```

Inclinación de la tipografía.

```
font-weight: normal | bold | bolder | lighter | ...
```

Grosor de la tipografía.

```
font: italic bold 10px serif;
```

Establecer varios atributos a la vez.

# DIMENSIONES DE UN ELEMENTO

Se controlan mediante los atributos `width` y `height`

Por defecto, las dimensiones de un elemento dependen del contenido del mismo y del ancho de la ventana del navegador. Es equivalente a poner:

```
width: auto;  
height: auto;
```

Podemos definir unas dimensiones absolutas:

```
width: 200px;  
height: 3em;
```

o relativas al elemento contenedor:

```
width: 90%;
```

## EJEMPLO

```
p.importante {  
  width: 30%;  
  margin-left: 20px;  
  ...  
}
```

Esto es un párrafo normal, que se extiende a lo largo de varias líneas.

**Esto es muy importante!**

```
p.importante {  
  width: 30%;  
  margin-left: auto;  
  margin-right: auto;  
  ...  
}
```

Esto es un párrafo normal, que se extiende a lo largo de varias líneas.

**Esto es muy importante!**

# IMÁGENES DE FONDO

```
background-image: url(nombre_fichero)  
background-position: left top | left center | left bottom  
                    | center top | center center | center bottom  
                    | right top | right center | right bottom  
background-repeat: repeat | repeat-x | repeat-y | no-repeat
```

## Ejemplo:

```
p.importante {  
    background-image: url(Caution2.png);  
    background-repeat: no-repeat;  
    background-position: right center;  
    background-color: rgb(255, 200, 200);  
    ...  
}
```

Esto es un párrafo normal, que se extiende a lo largo de varias líneas.

**Esto es muy importante!**





¿Qué diferencia hay entre incluir una imagen mediante CSS y mediante un elemento `<img>`?

# REGLAS DE PRECEDENCIA

Supongamos que tenemos dos atributos CSS contradictorios en reglas distintas que afectan al mismo elemento.

**¿Cuál prevalece?**

# EJEMPLO

```
<p>
  Esto es un <a href="http://www.ucm.es">enlace normal</a>, este tiene una
  <a href="http://www.google.com" class="enlace">clase</a>
  y este tiene un
  <a href="http://informatica.ucm.es" id="fdi">identificador</a>.
</p>
```

```
a {
  color: blue;
}

p > a {
  color: red;
}

.enlace {
  color: purple;
}

#fdi {
  color: green;
}
```



## REGLA 1: ESPECIFICIDAD

En caso de conflicto se aplican los atributos con selectores más específicos.

- $p > a$  es un selector más específico que  $a$ , porque afecta a menos elementos.
- Los selectores de clase son más específicos que los selectores sin clase.
- Los selectores de identificador son más específicos que cualquier otro.

En nuestro ejemplo:

Esto es un enlace normal, este tiene una clase y este tiene un identificador.

## REGLA 2: ORDEN DE LAS CLASES EN EL CSS

```
<a href="index.html" class="verde desactivado importante">  
  Volver  
</a>
```

```
.verde { ... }  
.importante { ... }  
.desactivado { ... }
```

Los atributos de la clase **desactivado** prevalecen sobre los de **importante**, que a su vez prevalecen sobre los de **verde**.

En general, prevalecen las clases que se definen después en la hoja de estilo CSS.

## REGLA 3: ATRIBUTO **style**

Es posible (aunque no se recomienda) introducir estilos directamente en el código HTML:

```
<a href="index.html" style="font-weight:bold; color: yellow">  
  Volver  
</a>
```

Estos atributos tienen precedencia sobre los especificados en cualquier hoja de estilo CSS.

## REGLA 4: LA MARCA **!important**

Los atributos CSS con la marca **!important** tienen prioridad sobre los que no lo tienen.

```
a {  
  color: blue !important;  
}  
  
p > a {  
  color: red;  
}
```

Esto es un [enlace normal](#), este tiene una [clase](#) y este tiene un [identificador](#).

# HERENCIA DE PROPIEDADES

Cuando un elemento HTML está contenido dentro de otro, hereda algunas de sus propiedades de estilo.

- Propiedades heredables: `font`, `color`, etc.
- Propiedades no heredables: `border`, `margin`, `padding`, etc.

# EJEMPLO

```
<body>
  <p class="cuadro">
    Introducimos un <a href="...">enlace</a>
    dentro del párrafo.
  </p>
</body>
```

```
body { font-family: sans-serif; }

p.cuadro {
  border: 2px dashed blue;
  padding: 20px;
}
```



Introducimos un [enlace](#) dentro del párrafo.

El enlace hereda el tipo de letra de **<body>**, pero no el borde del párrafo.

Un elemento puede sobrescribir las propiedades heredadas. El valor sobrescrito prevalece sobre el heredado.

```
a {  
  font-family: serif;  
}
```

Introducimos un [enlace](#) dentro del párrafo.

Un elemento puede decidir heredar aquellas propiedades no heredables introduciendo el valor **inherit** a la propiedad correspondiente:

```
a {  
  border: inherit;  
  padding-left: inherit;  
  padding-right: inherit;  
}
```

Introducimos un enlace dentro del párrafo.



# HOJAS DE ESTILO MÚLTIPLES

Es posible asociar varias hojas de estilo a un mismo documento.

```
<!DOCTYPE html>
<html>
  <head>
    ...
    <link rel="stylesheet" href="Hoja1.css"/>
    <link rel="stylesheet" href="Hoja2.css"/>
    <!-- Hoja2.css prevalece sobre Hoja1.css -->
  </head>
  <body>
    ...
  </body>
</html>
```

Las reglas de las hojas de estilo incluidas posteriormente prevalecen sobre los anteriores.

# EL ATRIBUTO *media*

Si se tienen varias hojas de estilo en un mismo documento, se puede asociar cada una de ellas con un dispositivo o configuración diferente.

```
<!DOCTYPE html>
<html>
  <head>
    ...
    <link rel="stylesheet" href="MediaScreen.css"
          media="screen"/>
    <link rel="stylesheet" href="MediaPrint.css"
          media="print"/>
  </head>
  <body>
    ...
  </body>
</html>
```

Se utilizará la hoja *MediaScreen.css* para la visualización en pantalla, *MediaPrint.css* para la impresión en papel.

```
<link rel="stylesheet" href="..." media="all"/>
```

La hoja de estilo se aplica en todos los dispositivos.

```
<link rel="stylesheet" href="..."  
      media="screen and (max-device-width:500px)"/>
```

La hoja de estilo se aplica en los dispositivos con una anchura de pantalla inferior a 500px.

# INSPECTOR DE PROPIEDADES DE FIREFOX

*Herramientas → Desarrollador → Inspector (Ctrl+Mayus+C)*

<https://youtu.be/TAAL707ARPs>

# INSPECTOR DE PROPIEDADES DE CHROME

*Herramientas para desarrolladores* → Pestaña *Elements*  
(Ctrl+Mayus+I)

1. INTRODUCCIÓN. PÁGINA BÁSICA
2. ELEMENTOS HTML
3. ATRIBUTOS CSS
4. **FLUJO Y POSICIONAMIENTO**
5. ELEMENTOS HTML5
6. FORMULARIOS
7. BIBLIOGRAFÍA



# FLUJO Y POSICIONAMIENTO

En HTML se distinguen dos tipos de elementos:

- Elementos de tipo **bloque**:
  - Ocupan todo el ancho de la página.
  - Se colocan de arriba a abajo en la página HTML.
- Elementos de tipo **inline**:
  - Ocupan solo el ancho que necesiten, según el contenido.
  - Se colocan de izquierda a derecha, hasta ocupar el ancho de la página. Saltan a la línea siguiente si es necesario.

Elementos de tipo bloque:

<p>, <h1>, <h2>, ..., <h6>, <ul>, <ol>, <li>, <table>, <blockquote>, <pre>, etc.

Elementos de tipo inline:

<a>, <strong>, <img>, <q>, etc.



# Elementos de tipo bloque: flujo vertical.

```
<p>Párrafo 1</p>  
<p>Párrafo 2</p>  
<blockquote>Cita</blockquote>  
<p>Párrafo 3</p>  
<ul>  
  <li>Elem 1</li>  
  <li>Elem 2</li>  
</ul>
```

Párrafo 1

Párrafo 2


Cita

Párrafo 3

- Elem 1
- Elem 2

## Elementos de tipo inline: flujo horizontal con el texto.

```
<p>  
  Coloco un enlace <a href="#">Enlace</a>,  
  seguido de una imagen ,  
  y una <q>cita</q>  
</p>
```

Coloco un enlace [Enlace](#), seguido de una imagen , y una "cita"

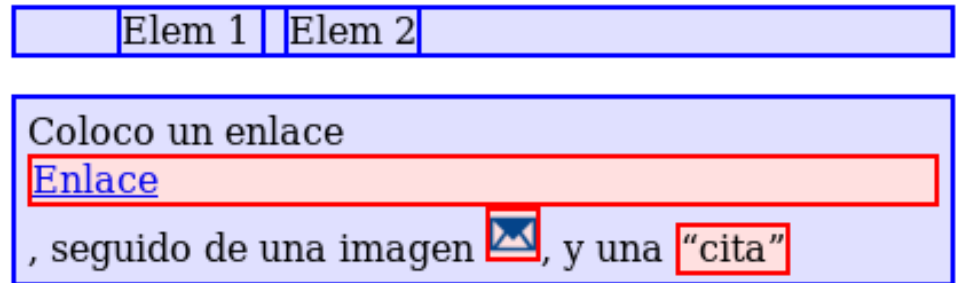
# EL ATRIBUTO `display`

Sirve para modificar el tipo de un elemento (block, inline) u ocultarlo.

```
display: block | inline | inline-block | none | ...
```

En los ejemplos anteriores:

```
li {  
  margin: 4px;  
  display: inline;  
}  
  
a {  
  display: block;  
}
```



# LOS ELEMENTOS MIXTOS INLINE/BLOCK

Los elementos `inline` ignoran algunas propiedades CSS:

- Márgenes superiores e inferiores.
- Anchura y altura.

```
span {  
  background-color: #FFE0E0;  
  border: 2px solid red;  
  margin-top: 10px;  
  width: 50px;  
  height: 50px;  
}
```

No podemos especificar la anchura o altura de un elemento `inline`.

Los elementos de tipo **inline-block** se comportan como elementos en línea (respetan el flujo horizontal) pero permiten tener márgenes y una dimensión específica.

```
span {  
  display: inline-block;  
  background-color: #FFE0E0;  
  border: 2px solid red;  
  margin-top: 10px;  
  width: 50px;  
  height: 50px;  
}
```

No podemos especificar la anchura o altura de un elemento **inline**.



# SECCIONES LÓGICAS

El elemento `<div>` sirve para agrupar distintos elementos, con el fin de aplicar un mismo estilo, o de tratarlos como una unidad lógica.

Los elementos `<div>` son de tipo bloque.

# EJEMPLO

```
<div class="articulo">
  <h2>Articulo 1</h2>
  <p>...</p>
</div>
```

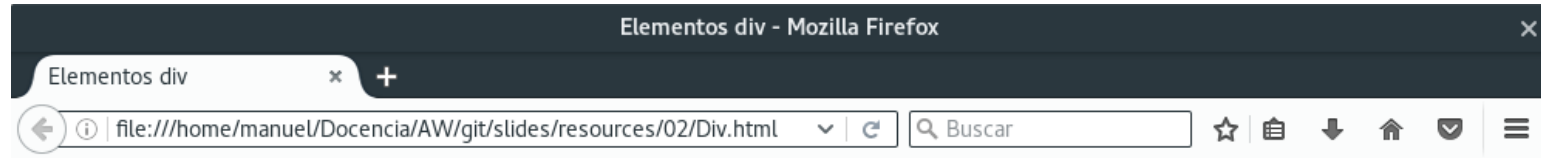
```
<div class="articulo">
  <h2>Articulo 2</h2>
  <p>...</p>
</div>
```

```
div.articulo {
  margin-left: 20px;
  margin-top: 10px;
  padding: 10px 20px;

  background-image: url(...);
  border-radius: 10px;
}
```

```
div.articulo h2 {
  color: #902020;
}
```

# Resultado:



## Bienvenido a mi página!

### Artículo 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque rutrum ornare turpis, eu pellentesque lacus luctus ac. Sed vitae nulla massa. Praesent in est non massa egestas rhoncus ut at nulla. Morbi quis ultricies leo. Pellentesque sollicitudin est et commodo laoreet. Sed suscipit eget sapien id aliquet. Maecenas tincidunt ex est, vitae pulvinar nibh fringilla ut. Vestibulum fermentum risus non lorem tempor euismod. Suspendisse ac enim id dui pellentesque congue vel ut nulla. Mauris molestie ante vel nunc egestas finibus. Praesent tincidunt eros et maximus viverra.

### Artículo 2

Curabitur vitae ipsum ipsum. Aenean quis elit non elit ullamcorper dignissim. Vestibulum tristique non elit in malesuada. Donec interdum mi lacus, congue efficitur mauris efficitur eu. Curabitur ac lacus at turpis dapibus scelerisque non vel lacus. Nullam velit risus, lobortis at sodales in, aliquam lobortis quam. Donec fermentum diam id consectetur tincidunt. Sed sit amet risus quis ante varius sollicitudin. Pellentesque tristique non magna id porttitor. Duis porttitor erat non elit dictum, nec ullamcorper magna auctor. Nulla laoreet porttitor nisl, non varius ipsum commodo et. Duis dignissim elementum lacus, non finibus sem fringilla at. Ut placerat consectetur ante a sagittis.



El elemento `<span>` tiene una finalidad similar a la de `<div>`, pero es un elemento de tipo inline.

```
<p>Curabitur vitae <span class="rojo">ipsum ipsum</span> ...</p>
```

```
span.rojo {  
  color: red;  
}
```

Curabitur vitae ipsum ipsum. Aenean

# POSICIONAMIENTO FLOTANTE

```
float: left | right
```

Cuando a un elemento se le asigna una propiedad **float**, es eliminado del flujo normal de posicionamiento, y pasa a ser considerado **flotante**.

Los elementos colindantes se sitúan alrededor del elemento flotante.

# EJEMPLO

```
<p>P1: Lorem ipsum ...</p>
<p>P2: Sed vitae ...</p>
<p>P3: Maecenas tincidunt...</p>
<p>P4: Suspendisse ac ...</p>
```

```
p {
    padding: 5px;
    background-color: #F0F0FF;
}

p.flotante_izq {
    float: left;
    width: 40%;
    background-color: #FFA000;
}

p.flotante_dch {
    float: right;
    width: 40%;
    background-color: #FFA000;
}
```



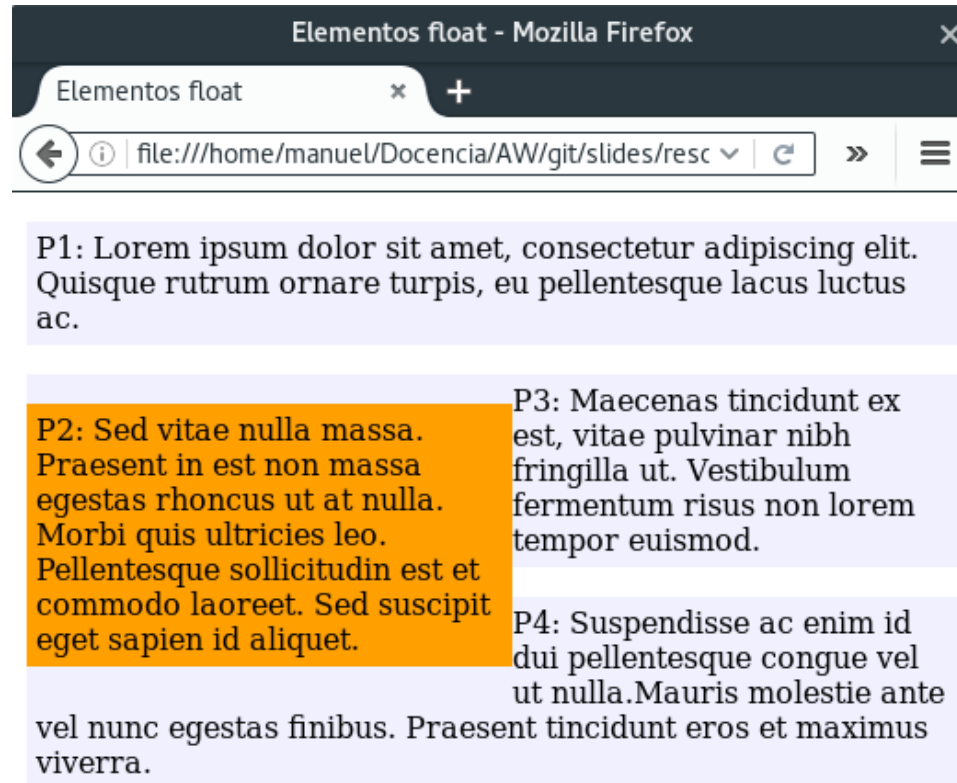
P1: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque rutrum ornare turpis, eu pellentesque lacus luctus ac.

P2: Sed vitae nulla massa. Praesent in est non massa egestas rhoncus ut at nulla. Morbi quis ultricies leo. Pellentesque sollicitudin est et commodo laoreet. Sed suscipit eget sapien id aliquet.

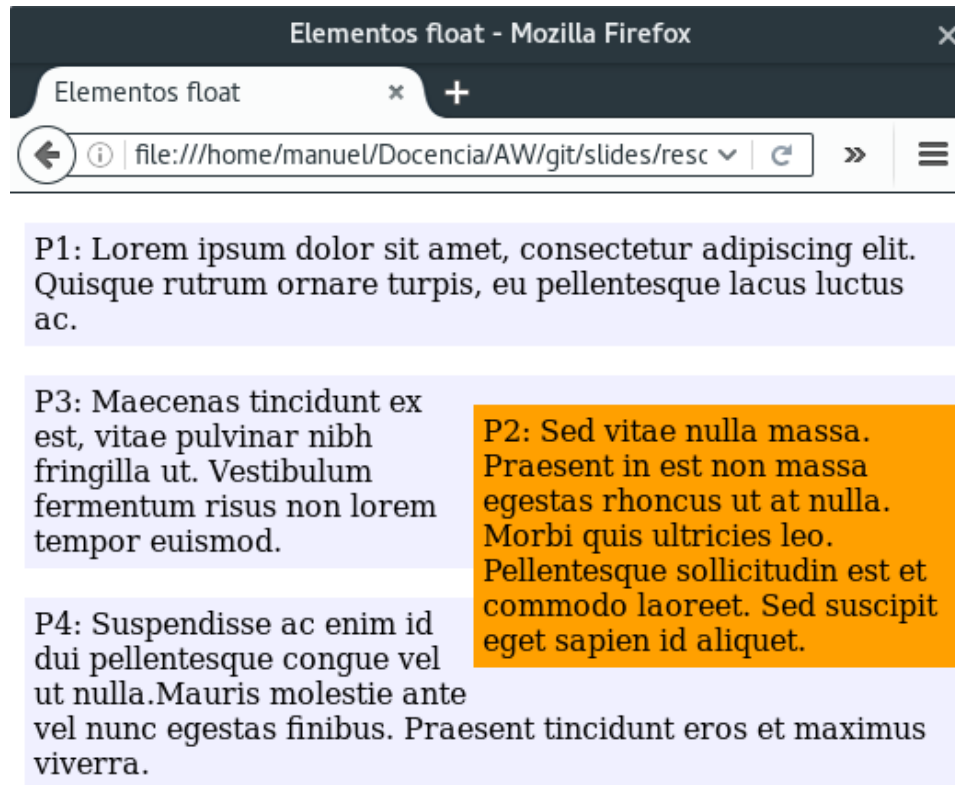
P3: Maecenas tincidunt ex est, vitae pulvinar nibh fringilla ut. Vestibulum fermentum risus non lorem tempor euismod.

P4: Suspendisse ac enim id dui pellentesque congue vel ut nulla. Mauris molestie ante vel nunc egestas finibus. Praesent tincidunt eros et maximus viverra.

Tras añadir la clase `flotante_izq` al segundo párrafo:



Tras cambiar la clase `flotante_izq` en P2 por `flotante_dch`:



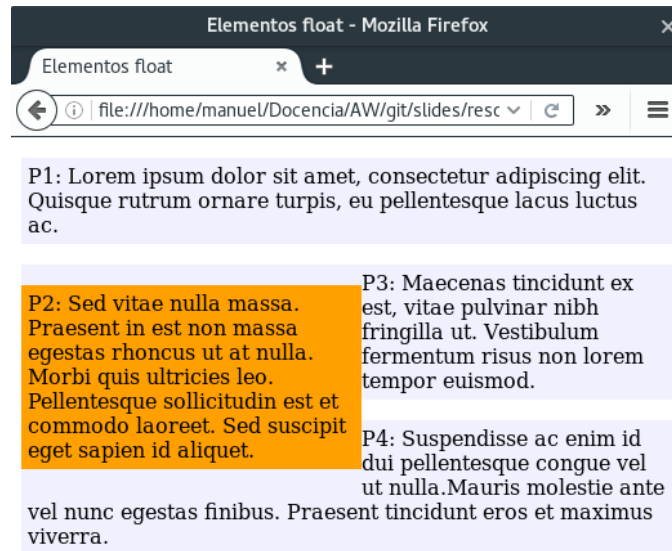
## ¿QUÉ PASA SI...?

- Añadimos la clase `flotante_dch` a P1 y P2.
- Añadimos la clase `flotante_izq` a P1 y `flotante_dch` a P2.

# LA PROPIEDAD clear

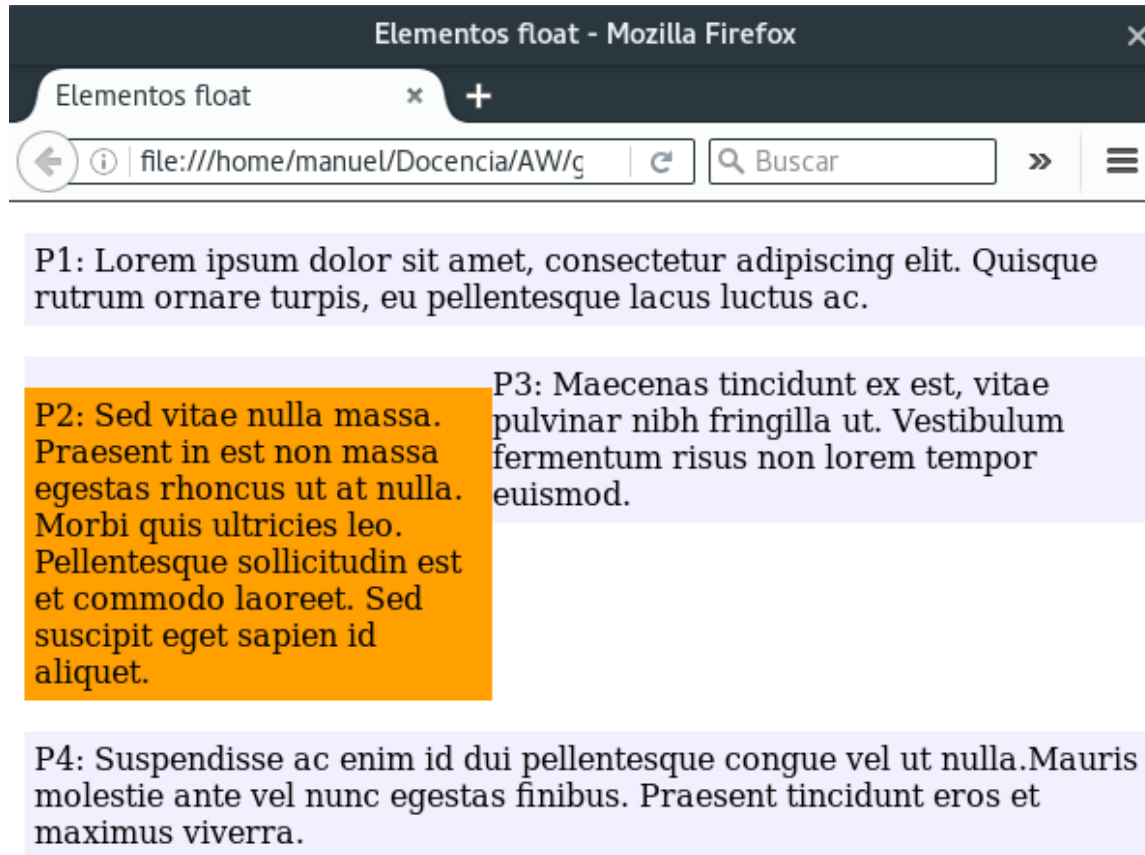
Sirve para evitar que un elemento se distribuya alrededor de elementos flotantes.

Por ejemplo, partiendo de la siguiente situación:





Añadimos la propiedad `clear:left` al estilo de P4.



```
clear: left | right | both | none
```

Los elementos que tengan la propiedad **clear** con un valor distinto de **none** no se distribuirán alrededor de los elementos que *floten* por la izquierda (**left**), por la derecha (**right**) o por ambos lados (**both**).

# CENTRADO DE ELEMENTOS

Partimos de la siguiente situación en la que hemos asignado la propiedad `width:50%` a P2:

P1: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque rutrum ornare turpis, eu pellentesque lacus luctus ac.

P2: Sed vitae nulla massa.  
Praesent in est non massa egestas  
rhoncus ut at nulla. Morbi quis  
ultricies leo. Pellentesque  
sollicitudin est et commodo  
laoreet. Sed suscipit eget sapien id  
aliquet.

P3: Maecenas tincidunt ex est, vitae pulvinar nibh fringilla ut.  
Vestibulum fermentum risus non lorem tempor euismod.

P4: Suspendisse ac enim id dui pellentesque congue vel ut  
nulla. Mauris molestie ante vel nunc egestas finibus. Praesent  
tincidunt eros et maximus viverra.

Asignando la propiedad `margin-left:auto` a P2, convertimos el margen izquierdo de este párrafo en un margen *elástico*, que se expande todo lo posible.

P1: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque rutrum ornare turpis, eu pellentesque lacus luctus ac.

P2: Sed vitae nulla massa. Praesent in est non massa egestas rhoncus ut at nulla. Morbi quis ultricies leo. Pellentesque sollicitudin est et commodo laoreet. Sed suscipit eget sapien id aliquet.

P3: Maecenas tincidunt ex est, vitae pulvinar nibh fringilla ut. Vestibulum fermentum risus non lorem tempor euismod.

P4: Suspendisse ac enim id dui pellentesque congue vel ut nulla. Mauris molestie ante vel nunc egestas finibus. Praesent tincidunt eros et maximus viverra.

Si, además de lo anterior, añadimos `margin-right:auto` a P2, convertiremos ambos márgenes en elásticos. El elemento aparecerá centrado.

P1: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque rutrum ornare turpis, eu pellentesque lacus luctus ac.

P2: Sed vitae nulla massa.  
Praesent in est non massa egestas  
rhoncus ut at nulla. Morbi quis  
ultrices leo. Pellentesque  
sollicitudin est et commodo  
laoreet. Sed suscipit eget sapien id  
aliquet.

P3: Maecenas tincidunt ex est, vitae pulvinar nibh fringilla ut. Vestibulum fermentum risus non lorem tempor euismod.

P4: Suspendisse ac enim id dui pellentesque congue vel ut nulla. Mauris molestie ante vel nunc egestas finibus. Praesent tincidunt eros et maximus viverra.

# POSICIONAMIENTO ABSOLUTO

Es posible utilizar CSS para situar elementos en una posición arbitraria de la página.

```
position: absolute;  
left: coordenada_x;  
top: coordenada_y;
```

Las posiciones son relativas a la esquina superior izquierda de la **página HTML**.

Cuando un elemento recibe la propiedad `position: absolute`, deja de formar parte del flujo de la página y se sitúa en la posición indicada por las propiedades `left` y `top`.

Por ejemplo, si aplicamos al párrafo P2 del ejemplo anterior las siguientes propiedades:

```
position: absolute;  
left: 100px;  
top: 20px;  
width: 40%;  
background-color: #FFA000;
```

## Obtenemos:

P1: Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Quisque rutrum. P2: Sed vitae nulla massa. Praesent in est non massa  
egestas rhoncus ut at nulla. P3: Maecenas vestibulum nibh fringilla ut.  
Morbi quis ultricies leo. Pellentesque sollicitudin est  
et commodo laoreet. Sed P4: Suspendisse quisque congue vel ut  
nulla. Mauris suscipit eget sapien id aliquet. Etas finibus. Praesent  
tincidunt eros et maximus viverra.



# POSICIONAMIENTO RELATIVO

```
position: relative;  
left: coordenada_x;  
top: coordenada_y;
```

Los elementos con posicionamiento relativo no son eliminados del flujo de la página.

Los atributos **left** y **top** indican la posición del elemento tomando como origen la posición en la que estaría situado el elemento según el flujo de la página.

En el ejemplo anterior, cambiamos `position:absolute`  
por `position:relative`:

P1: Lorem ipsum dolor sit amet, consectetur adipiscing elit.  
Quisque rutrum ornare turpis, eu pellentesque lacus luctus ac.



P2: Sed vitae nulla massa.  
Praesent in est non massa  
egestas rhoncus ut at  
nulla. Morbi quis ultricies  
leo. Pellentesque  
sollicitudin est et  
commodo laoreet. Sed  
suscipit eget sapien id  
aliquet.

P3: Maecenas tincidunt ex est, vitae pulvinar nibh fringilla ut.  
Vestibulum fermentum risus non lorem tempor euismod.

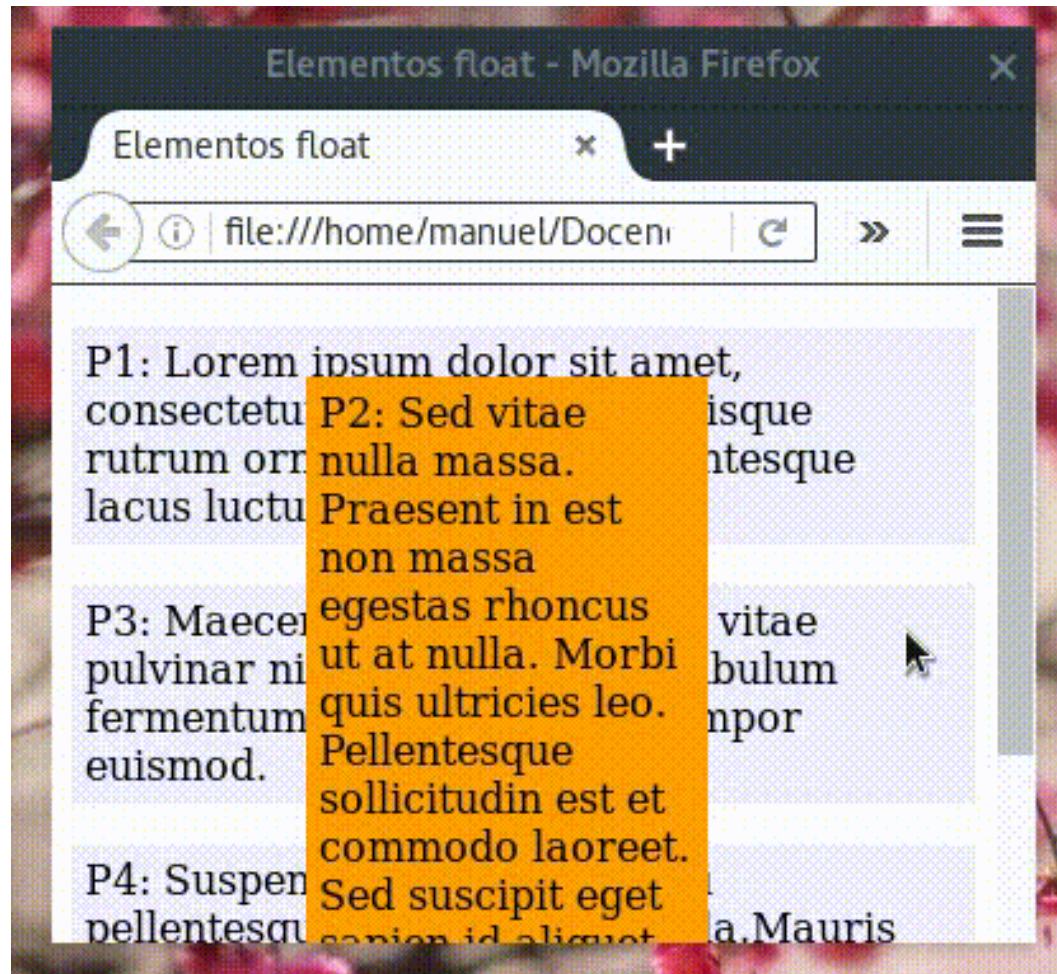
P4: Suspendisse ac enim id dui pellentesque congue vel ut

# POSICIONAMIENTO FIJO

```
position: fixed;  
left: coordenada_x;  
top: coordenada_y;
```

Los elementos con `position: fixed` se sitúan en una posición independiente de la parte de la página que se esté visualizando en el navegador.

En el ejemplo anterior, asignamos la propiedad **position: fixed** a P2:



# POSICIONAMIENTO FLEXIBLE (FLEXBOX)

Las últimas versiones de CSS incorporan un nuevo tipo de posicionamiento (*Flexbox*), que permite configurar la disposición de los elementos dentro de un contenedor.

Soporte en navegadores:

- Firefox 38 o posterior.
- Chrome 21 o posterior.
- Safari 6.1 o posterior.
- Opera 12.1 o posterior.
- Edge 12 o posterior.
- Internet Explorer lo soporta parcialmente.

## EJEMPLO

```
<div class="externo">
  <div class="interno">1</div>
  <div class="interno">2</div>
  <div class="interno">3</div>
  <div class="interno">4</div>
</div>
```

```
div.externo {
  display: flex;
  width: 700px;
  padding: 10px;
  ...
}

div.interno {
  width: 100px;
  height: 100px;
  ...
}
```



La propiedad `flex-direction` determina el **eje principal**.

- `flex-direction: row`



- `flex-direction: column`



El **eje secundario** es el perpendicular al principal.

# Disposición a lo largo del eje principal:

```
justify-content: flex-start | flex-end | center  
                | space-between | space-around
```

- flex-start



- space-between



- flex-end



- space-around



- center

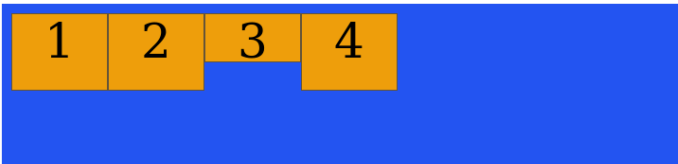




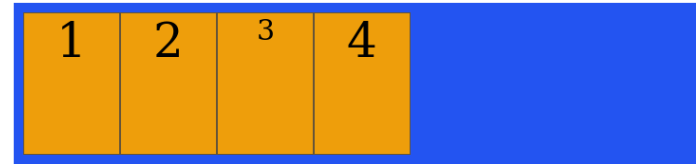
# Disposición a lo largo del eje secundario:

`align-items: flex-start | flex-end | center | stretch | baseline`

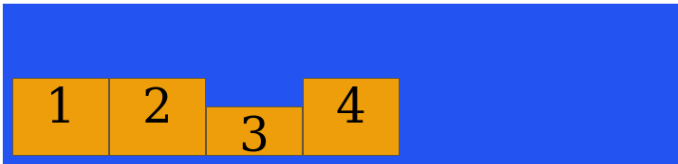
- **flex-start**



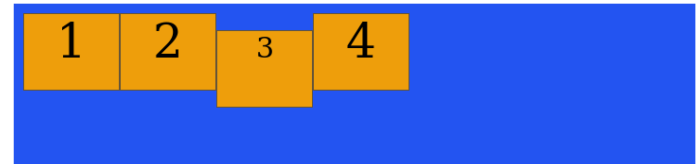
- **stretch**



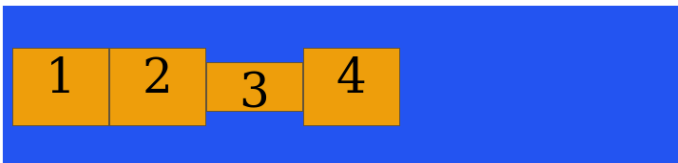
- **flex-end**



- **baseline**



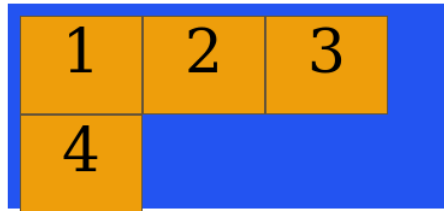
- **center**



Si los elementos se "salen" del contenedor, podemos especificar cómo se colocan:

```
flex-wrap: wrap | nowrap | ...
```

- wrap



- nowrap

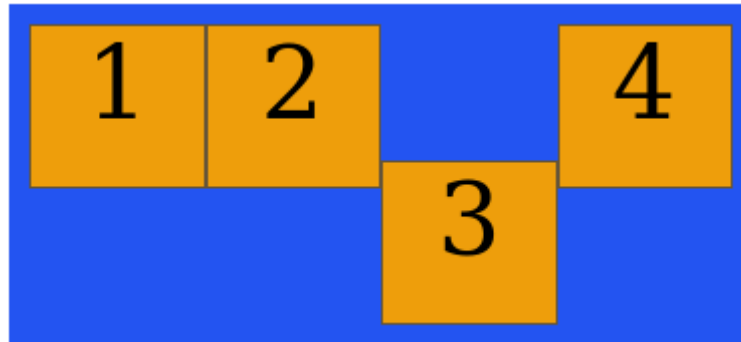


# PROPIEDADES DE LOS ELEMENTOS HIJOS

`align-self: flex-start | flex-end | center | baseline | stretch`

Alineación de un elemento hijo con respecto al eje secundario.

Por ejemplo, si asignamos `align-self: flex-end` al tercer elemento:

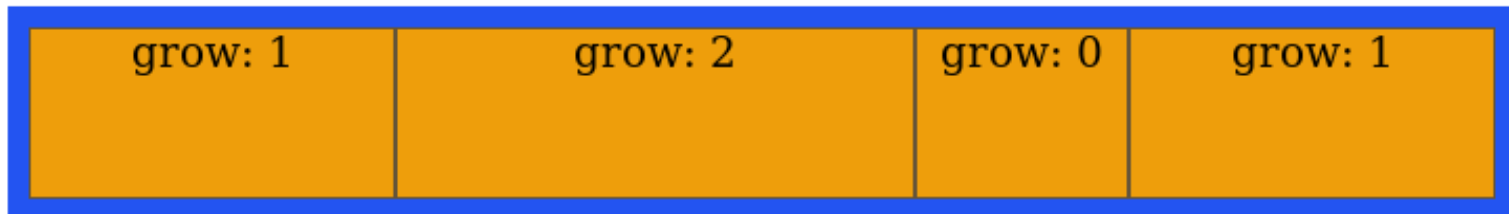


`flex-grow: número`

Determina cómo se distribuye el espacio del eje principal entre los elementos.

El número indica el **peso** del elemento a la hora de ocupar el espacio del padre. Si toma el valor **0** no se expande.

Por ejemplo:



`order: número`

Determina el orden del elemento dentro del padre.

Esto permite colocar elementos en un orden distinto en el que aparecen en el código HTML:

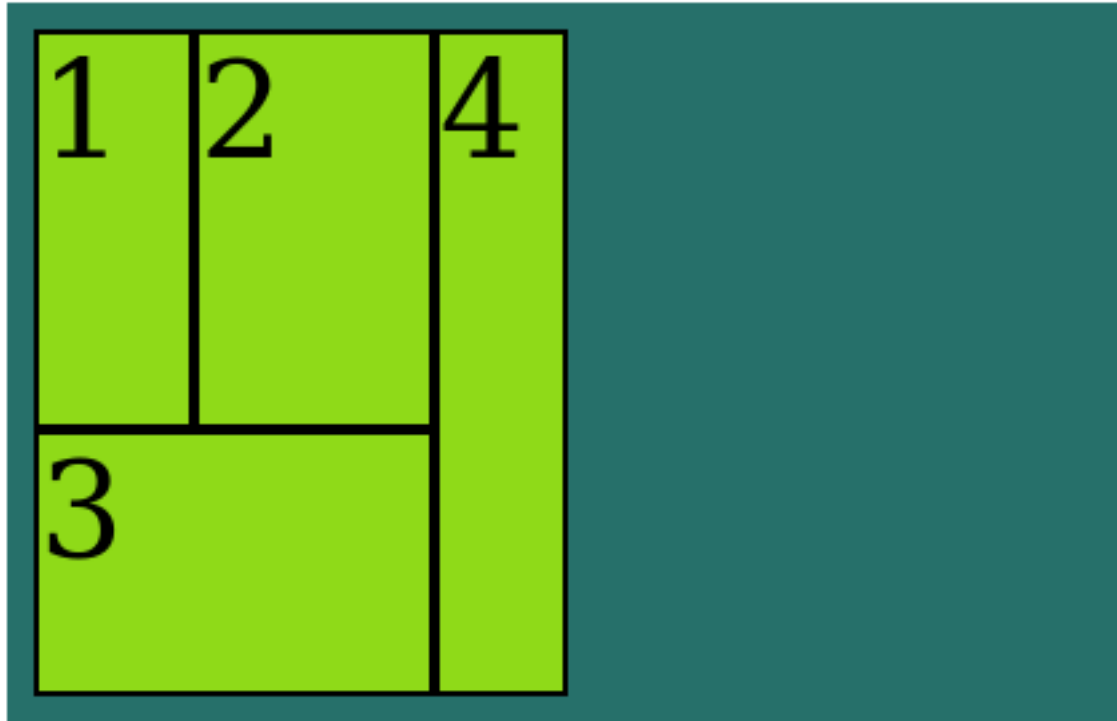


Más información:

<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

# POSICIONAMIENTO EN REJILLA (GRID)

Más recientemente aún, se ha incorporado un mecanismo de colocación de elementos en forma de rejilla flexible:



## Soporte en navegadores:

- Firefox 52 o posterior.
- Chrome 57 o posterior.
- Safari 10.1 o posterior.
- Opera 44 o posterior.
- Edge 15 (parcialmente).
- Internet Explorer lo soporta parcialmente.
- Opera Mini no lo soporta.

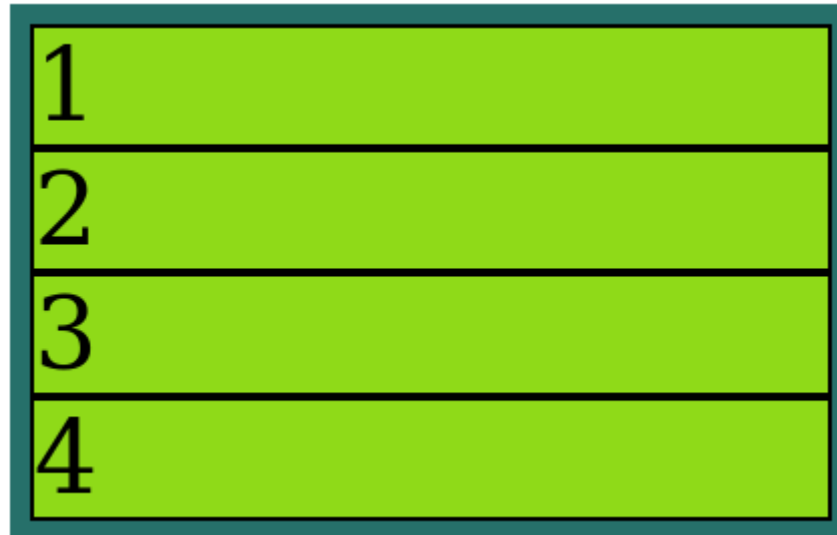


# EJEMPLO

Partimos del siguiente código:

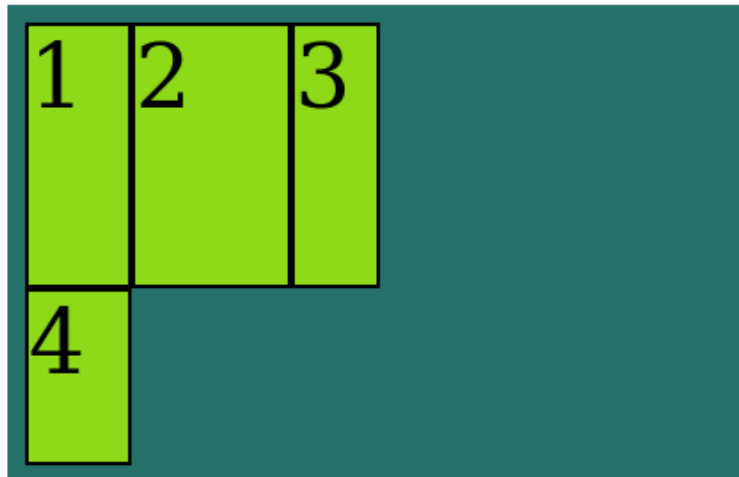
```
<div class="contenedor">
  <div id="uno">1</div>
  <div id="dos">2</div>
  <div id="tres">3</div>
  <div id="cuatro">4</div>
</div>
```

```
div.contenedor {
  ...
  width: 400px;
  padding: 10px;
  display: grid
}
```



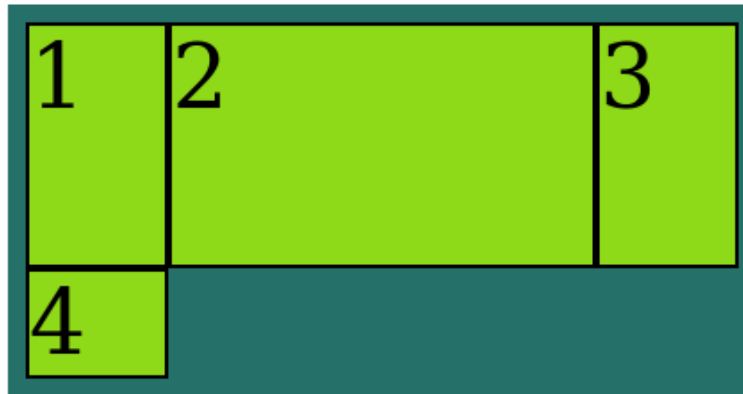
Mediante `grid-template-rows` y `grid-template-columns` indicamos el número de filas y columnas, y la longitud de cada una de ellas.

```
div.contenedor {  
    ...  
    /* Tres columnas */  
    grid-template-columns: 150px 100px;  
    /* Dos filas */  
    grid-template-rows: 60px 90px 50px;  
}
```



Es posible especificar unidades relativas, indicando los **pesos** de cada una de las filas o columnas.

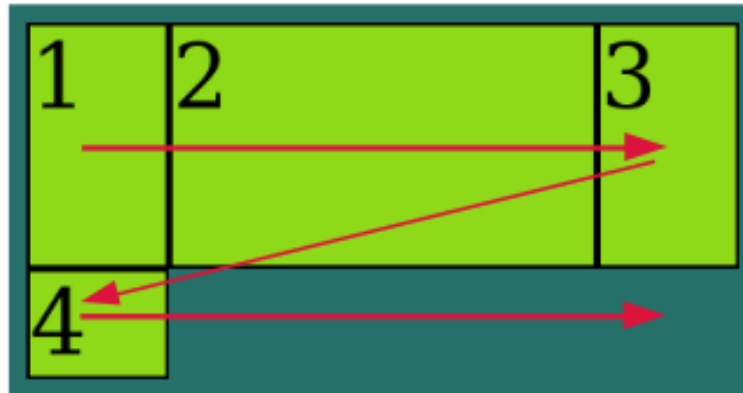
```
div.contenedor {  
  ...  
  /* Tres columnas */  
  grid-template-columns: 1fr auto;  
  /* Dos filas */  
  grid-template-rows: 1fr 3fr 1fr;  
}
```



**auto** = no ocupar espacio extra

## POSICIÓN DE CADA ELEMENTO

Por defecto, los elementos se van situando de izquierda a derecha y de arriba a abajo en la cuadrícula...



...pero podemos especificar la posición de elementos individualmente:

```
grid-column: inicio / fin  
grid-row:    inicio / fin
```

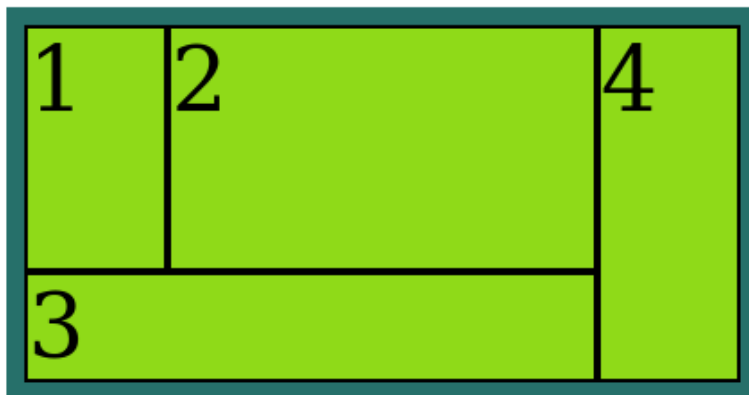
El elemento se situará desde la fila/columna *inicio* hasta la fila/columna *fin*, excluyendo esta última.

Se comienza a contar desde el **1**.

Alternativamente, podemos utilizar:

```
grid-column: inicio / span numFilas  
grid-row:    inicio / span numColumnas
```

```
#tres { grid-row: 2 / 3; grid-column: 1 / 3; }  
#cuatro { grid-row: 1 / 3; grid-column: 3 / 4; }
```

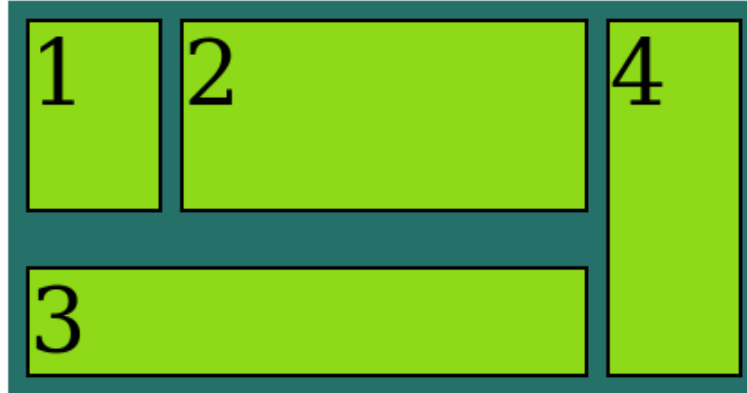


También podíamos haber utilizado:

```
#tres { grid-row: 2 / span 1; grid-column: 1 / span 2; }  
#cuatro { grid-row: 1 / span 2; grid-column: 3 / span 1; }
```

## Especificar espacio entre las celdas:

```
div.contenedor {  
  ...  
  grid-column-gap: 10px;  
  grid-row-gap: 30px;  
}
```



## OTRAS PROPIEDADES

- `justify-items`, `align-items`  
Similares a las de Flexbox, permiten alinear un elemento dentro de su celda. Por defecto toman el valor `stretch`.
- `grid-template-areas`, `grid-area`  
Otra forma de especificar la posición de los elementos.

Más información:

<https://css-tricks.com/snippets/css/complete-guide-grid/>



1. INTRODUCCIÓN. PÁGINA BÁSICA
2. ELEMENTOS HTML
3. ATRIBUTOS CSS
4. FLUJO Y POSICIONAMIENTO
5. **ELEMENTOS HTML5**
6. FORMULARIOS
7. BIBLIOGRAFÍA



# ELEMENTOS HTML5

Es frecuente utilizar secciones lógicas `<div>` para dividir las distintas partes que componen una página web.

```
<body>
  <div class="titulo">
    ...
  </div>
  <div class="menu_izquierdo">
    <ul>...</ul>
  </div>
  <div class="contenido">
    <div class="articulo">
      ...
    </div>
    <div class="articulo">
      ...
    </div>
  </div>
  <div class="pie_de_pagina">
    ...
  </div>
</body>
```

Esta división, aunque correcta, no proporciona demasiada información al navegador sobre la **semántica** de los elementos de la página web.

En HTML5 se introdujeron nuevas etiquetas para ayudar al navegador a distinguir los distintos elementos de la página.

## **VENTAJAS**

- El navegador puede alterar la presentación de algunos elementos de la página (p. ej. para personas con discapacidad visual).
- Los buscadores pueden obtener información sobre la estructura de la página.

# ELEMENTOS DE MARCADO SEMÁNTICO

- **<header>** Contenido situado en la parte superior de la página (título, logo, etc.) o de una sección de ésta.
- **<nav>** Enlaces para navegar por el sitio web.
- **<footer>** Contenido situado en la parte inferior de la página o de una sección de ésta.
- **<section>** Sección lógica (suele contener encabezado y pie de página).
- **<aside>** Contenido complementario a la página (suele estar en el margen derecho).
- **<article>** Composición autocontenida en la página (artículo de un blog, post de un foro, comentario de un usuario, etc.)

En el ejemplo anterior:

```
<body>
  <header>
    ...
  </header>
  <nav>
    <ul>...</ul>
  </nav>
  <div class="contenido">
    <article>
      ...
    </article>
    <article>
      ...
    </article>
  </div>
  <footer>
    ...
  </footer>
</body>
```

## LA ETIQUETA `<time>`

Es una división lógica para mostrar una fecha y/o hora.

Permite almacenar "internamente" la fecha en un formato legible por el navegador para su procesamiento.

Abrimos a las

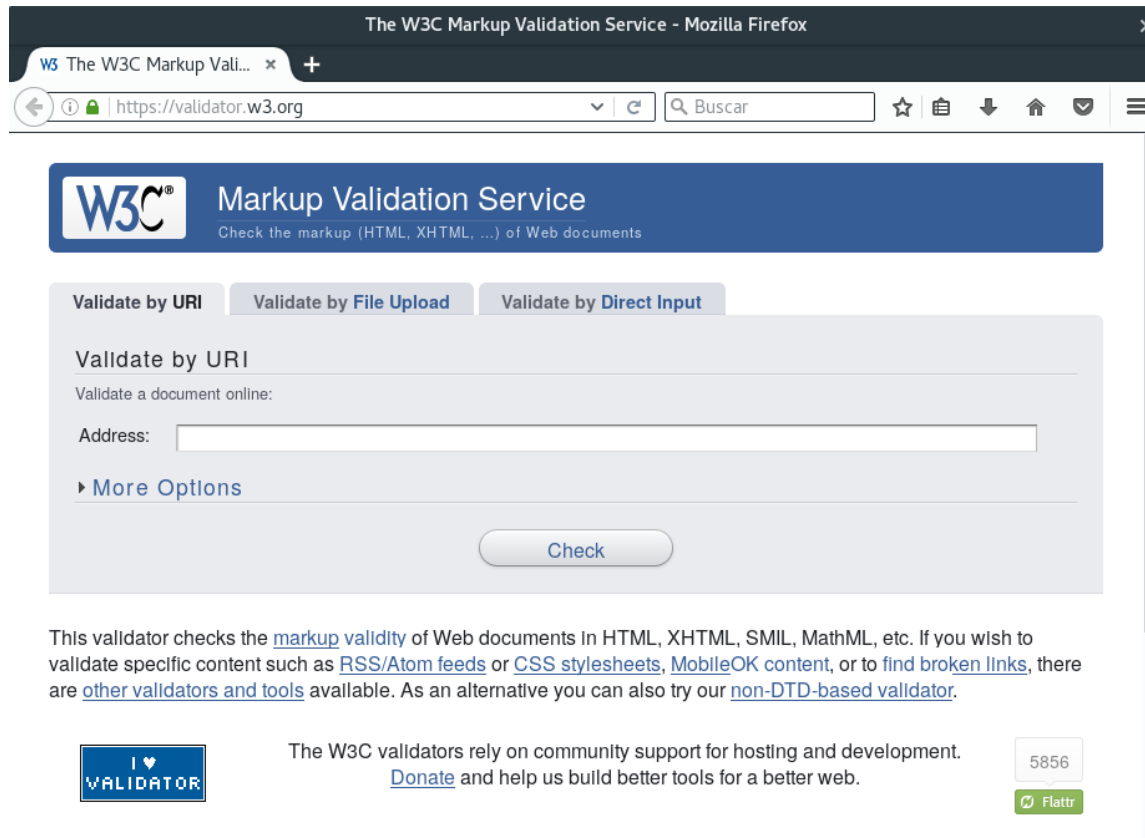
```
<time datetime="2014-04-04 10:00">10 de la mañana</time>.
```

## OTROS ELEMENTOS HTML5

- **Gráficos:** `<canvas>`.
- **Multimedia:** `<audio>`, `<video>`, `<embed>`, etc.
- **Marcado semántico:** `<figure>`, `<menuitem>`, `<progress>`, etc.

# VALIDADOR HTML5

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



The screenshot shows the W3C Markup Validation Service website in a Mozilla Firefox browser. The browser's address bar displays the URL <https://validator.w3.org/>. The website's header features the W3C logo and the text "Markup Validation Service" with the subtitle "Check the markup (HTML, XHTML, ...) of Web documents". Below the header, there are three tabs: "Validate by URI", "Validate by File Upload", and "Validate by Direct Input". The "Validate by URI" tab is selected, showing a form with the label "Validate by URI" and the instruction "Validate a document online:". Below this, there is a text input field labeled "Address:". A link labeled "More Options" is positioned below the input field. A "Check" button is located at the bottom of the form. Below the form, a paragraph of text explains the validator's purpose and provides links to additional resources. At the bottom of the page, there is a "I ♥ VALIDATOR" button, a paragraph of text about community support, a "Donate" link, a "Flattr" button, and a counter showing "5856".

The W3C Markup Validation Service - Mozilla Firefox

W3 The W3C Markup Vali... x +

https://validator.w3.org

W3C® Markup Validation Service  
Check the markup (HTML, XHTML, ...) of Web documents

Validate by URI Validate by File Upload Validate by Direct Input

Validate by URI

Validate a document online:

Address:

► More Options

Check

This validator checks the [markup validity](#) of Web documents in HTML, XHTML, SMIL, MathML, etc. If you wish to validate specific content such as [RSS/Atom feeds](#) or [CSS stylesheets](#), [MobileOK content](#), or to [find broken links](#), there are [other validators and tools](#) available. As an alternative you can also try our [non-DTD-based validator](#).

I ♥ VALIDATOR

The W3C validators rely on community support for hosting and development. [Donate](#) and help us build better tools for a better web.

5856

Flattr



1. INTRODUCCIÓN. PÁGINA BÁSICA
2. ELEMENTOS HTML
3. ATRIBUTOS CSS
4. FLUJO Y POSICIONAMIENTO
5. ELEMENTOS HTML5
6. **FORMULARIOS**
7. BIBLIOGRAFÍA



# FORMULARIOS HTML

Sirven para solicitar información al usuario de la página.

Cuando el usuario envía la información, se realiza una petición HTTP al servidor. Esta petición contiene los datos introducidos por el usuario.

En este tema solamente trataremos la especificación de los formularios. Más adelante veremos cómo procesa el servidor esta información.

# DEFINICIÓN DE UN FORMULARIO

```
<form method="post" action="procesar_formulario.html">  
  <!-- Contenido del formulario -->  
</form>
```

- Atributo **method**: método HTTP que se utilizará para enviar la información al servidor (GET o POST).
- Atributo **action**: URL sobre la que se realizará la petición.

# FORMULARIOS GET VS POST

```
<form action="ProcesarFormulario.html" method="método">  
  Nombre: <input type="text" name="nombre">  
  <input type="submit">  
</form>
```

Enviamos la siguiente información:

Nombre:

# MÉTODO GET

GET `http://.../ProcesarFormulario.html?nombre=Gerardo`

La información se envía como parte de la URL.

`http://.../destino?param1=val1&param2=val2&...`

# MÉTODO POST

POST `http://.../ProcesarFormulario.html`  
... Cabeceras ...

Content-Type: `application/x-www-form-urlencoded`  
Content-Length: 14  
`nombre=Gerardo`

Cuerpo de la petición

La información se envía en el cuerpo de la petición, con el mismo formato que el método GET:

`param1=val1&param2=val2&...`

## MÉTODO GET

Adecuado cuando se puede reproducir la misma petición varias veces, sin "efectos secundarios".

Se puede volver a realizar la petición introduciendo la misma URL en el navegador.

Ejemplo: búsquedas, consultas, etc.

## MÉTODO POST

Adecuado para enviar información que modifica el estado del servidor.

Ejemplo: altas, bajas, modificaciones, etc.

# ELEMENTOS EN UN FORMULARIO

- Botones.
- Cuadros de texto.
- Casillas seleccionables.
- Listas.

# BOTONES

```
<input type="submit" value="Enviar">  
<input type="reset" value="Borrar formulario">  
<input type="button" value="Borrar formulario">
```



Los botones de tipo **submit** realizan la petición indicada por el atributo **method** del formulario a la URL indicada por el atributo **action**.

Los botones de tipo **reset** reinician los elementos del formulario a sus valores por defecto.

Los botones de tipo **button** no hacen nada, a menos que se maneje su evento de pulsación en *Javascript*.



## CUADROS DE TEXTO

```
<input type="text" name="nombre_campo" value="Valor por defecto">
```

Datos enviados:

nombre\_campo=Me+llamo+Elisa

## OTROS TIPOS DE CUADROS DE TEXTO

```
<input type="password" name="contrasena">  
<input type="number" name="edad">  
<input type="range" name="peso" min="30" max="200">  
<input type="color" name="color_favorito">  
<input type="date" name="fecha_nacimiento">
```

Los campos de entrada de tipo **range**, **color** y **date** pueden no mostrarse correctamente en algunos navegadores.

Nombre:

Contraseña:

Edad:

Peso:

Color:

Fecha de nacimiento:

# ÁREAS DE TEXTO

Permiten la introducción de varias líneas de texto

```
<textarea name="observaciones" id="text_observ">  
Texto por defecto  
</textarea>
```

```
#text_observ {  
  display: block;  
  width: 30em;  
  height: 5em;  
  background: linear-gradient(to bottom, white, #E0E0E0);  
  border: 1px solid #A0A0A0;  
  color: blue;  
}
```

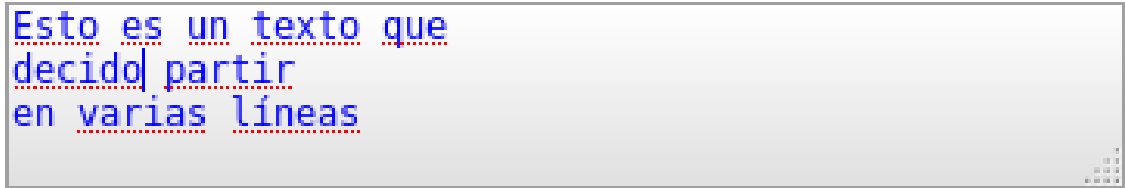
Observaciones:

Texto por defecto



A la hora de enviar el formulario, los saltos de línea introducidos en el `<textarea>` se traducen en secuencias `%0D%0A` (salto de línea `\n` + retorno de carro `\r`).

Observaciones:



```
Esto es un texto que  
decido| partir  
en varias líneas
```

```
observaciones=Esto+es+un+texto+que%0D%0Adecido+partir%0D%0Aen  
+varias+l%C3%ADneas
```

## CAJAS DE SELECCIÓN (CHECKBOX)

Fumador/a:

```
<input type="checkbox" name="fuma" value="si">
```

Fumador: ☐

Fumador/a:

```
<input type="checkbox" name="fuma" value="si" checked>
```

Fumador: ☒

Si la casilla está marcada al enviar el formulario, se añadirá el parámetro **fuma=si** a la petición. Si no, el parámetro no aparecerá en la petición.

## SELECCIÓN EXCLUSIVA (RADIO BUTTON)

Sexo:

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

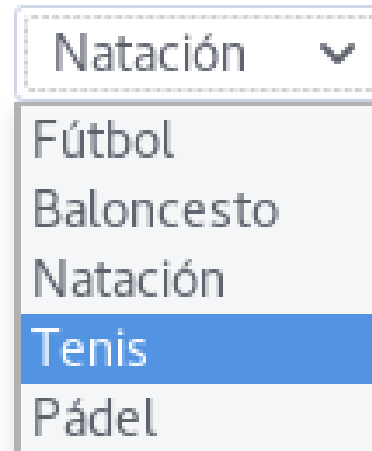
Sexo: ☒ Hombre ☐ Mujer

Al enviar el formulario se añade el parámetro **sexo=hombre** a la petición. Si no se selecciona ninguna opción, no aparece el parámetro **sexo** en la petición.

# LISTAS DESPLEGABLES

```
<select name="opt">
  <option value="fut">Fútbol</option>
  <option value="bal">Baloncesto</option>
  <option value="nat">Natación</option>
  <option value="ten">Tenis</option>
  <option value="pad">Pádel</option>
</select>
```

Actividades:



Natación ▼

- Fútbol
- Baloncesto
- Natación
- Tenis
- Pádel

Al enviar el formulario tras seleccionar **Tenis** se añade el parámetro **opt=ten** a la petición.

## ETIQUETAS (<label>)

Se representan como texto normal, pero van asociadas a un componente del formulario.

Cuando el usuario hace clic sobre la etiqueta, el componente correspondiente se activa.

```
<label for="txt_nombre">Nombre:</label>  
<input type="text" name="nombre" value="Defecto" id="txt_nombre">
```



1. INTRODUCCIÓN. PÁGINA BÁSICA
2. ELEMENTOS HTML
3. ATRIBUTOS CSS
4. FLUJO Y POSICIONAMIENTO
5. ELEMENTOS HTML5
6. FORMULARIOS
7. **BIBLIOGRAFÍA**



# BIBLIOGRAFÍA

- E. Robson, E. Freeman  
Head First HTML and CSS  
O'Reilly (2012)
- HTML Reference  
<http://www.w3schools.com/tags/>
- CSS Reference  
<http://www.w3schools.com/cssref/>

