

## TEMA 2

# HTML Y CSS

**APLICACIONES WEB - GIS - CURSO 2019/20**

**Marina de la Cruz** [marina.cruz@ucm.es]  
Dpto de Sistemas Informáticos y Computación  
Facultad de Informática  
Universidad Complutense de Madrid



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

Este documento está basado en <https://manuelmontenegro.github.io/AW-2017-18/02.html#/> de Manuel Montenegro [montenegro@fdi.ucm.es] bajo  
una Licencia CC BY-NC-SA 4.0 Internacional.

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



# INTRODUCCIÓN

Estudiaremos 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.

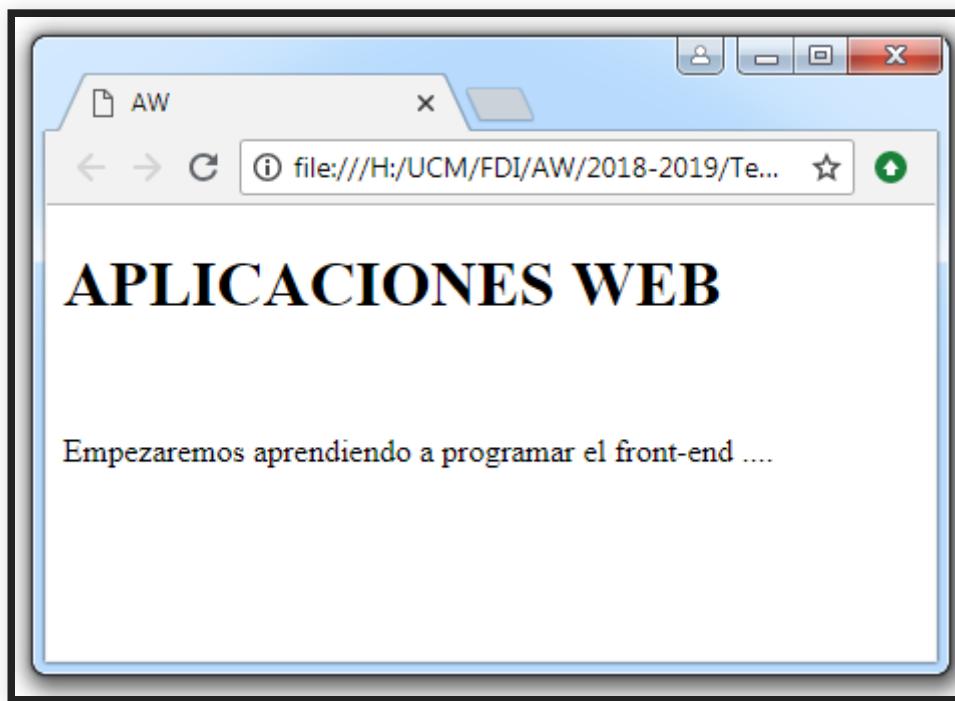
Estándares mantenidos por el *World Wide Web Consortium*  
([W3C](#))

# UN DOCUMENTO HTML SENCILLO

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

```
<!DOCTYPE html>
<html lang="es">
  <head>
    <title>AW</title>
    <meta charset="utf-8"/>
  </head>
  <body>
    <h1>APLICACIONES WEB</h1>
    <br>
    <p>Empezaremos aprendiendo a programar el front-end ....</p>
  </body>
</html>
```

Abrimos el fichero HTML en el navegador web:



# 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>
```

# ESTRUCTURA DE UN DOCUMENTO HTML

- Tipo de documento

```
<!DOCTYPE html>
```

- Cabecera

```
<head>
    <title>AW</title>
    <meta charset="utf-8" />
</head>
```

- Cuerpo

```
<body>
    <h1>APLICACIONES WEB</h1>
    <br>
    <p>Empezaremos aprendiendo a programar el front-end ...</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">
```

## CABECERA (<head>)

Contiene información sobre la página web que es ajena al propio contenido. Puede contener otros elementos.

```
<title>AW</title>
```

- Define el título de la página.
- Texto que aparece en la pestaña del navegador.
- Texto que aparece si se añade la página a favoritos.

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

- Indica la codificación de la página.
- Consultar [Distintos mapas de caracteres](#)

## CUERPO (<body>)

Especifica el contenido de la página web. Contiene múltiples elementos. En el primer ejemplo:

- Un encabezado (elemento <h1>)

```
<h1>APLICACIONES WEB</h1>
```

**APLICACIONES WEB**

- Un párrafo (elemento <p>)

```
<p>Empezaremos aprendiendo a programar el front-end ...</p>
```

Empezaremos aprendiendo a programar el front-end ...

## EL ELEMENTO <html>

Representa la raíz del documento HTML.

Puede (es muy recomendable) incluir un atributo `lang` con el idioma del contenido de la página web.

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

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

# 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 (1)

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 decimal *num*. Por ej. `∀` representa √

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

# CARACTERES ESPECIALES (2)

Carácter	Nombre entidad	Número entidad
©	&copy;	&#169;
Ω	&ohm;	&#8486;
∞	&infin;	&#8734;
™	&trade;	&#8482;
¥	&yen;	&#165;

# 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

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

Excepcionalmente algunas etiquetas (como `<strong>`) configuran el aspecto visual del contenido (no es igual para todos los navegadores).

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

# EJEMPLO

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

El mismo documento 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*.

# 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 **rel** indica la relación que tiene el fichero referenciado mediante **href** y el documento HTML actual (*hoja de estilo*).
- El atributo **href** indica el nombre del fichero que contiene la 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 otras definiciones:

```
body {  
    font-family: sans-serif;  
    background-color: #FFF0F0;  
}  
  
h1 {  
    border-left: 5px solid #006000;  
    padding-left: 10px;  
    color:#006000;  
    background-color: #E0F0E0;  
}  
  
strong {  
    font-family: "Inconsolata", monospace;  
    color: #006000;  
    background-color: #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>`, ...
- Preformatado: `<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>
```

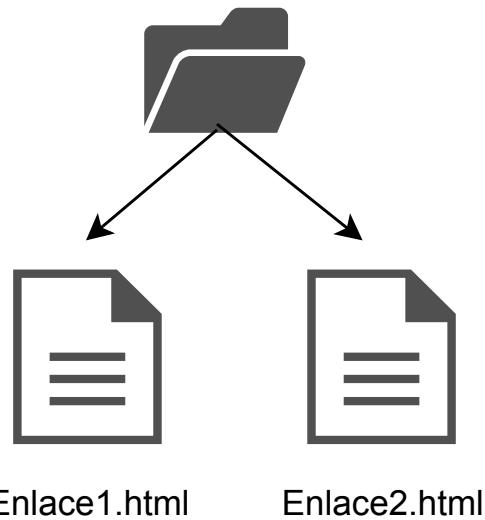
Estás en la página dos.

[Volver a la página uno.](#)

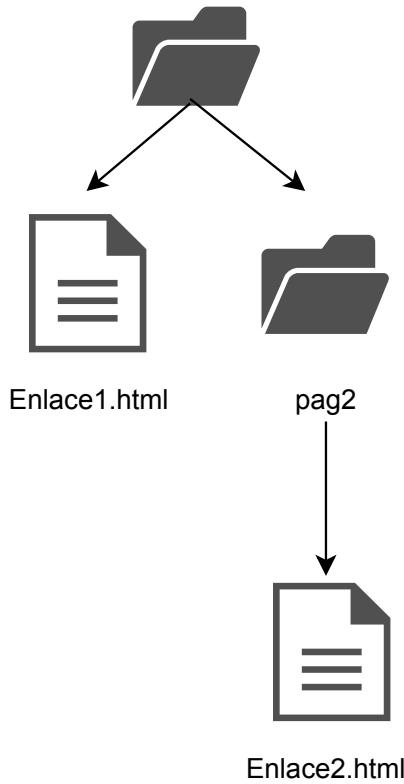
# ENLACES RELATIVOS Y ABSOLUTOS

```
<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** (a la página que contiene el enlace).



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 comienzan 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 enlaza a la URL `http://www.foo.org/index.html`

Los enlaces **absolutos** contienen una **URL completa** incluyendo el protocolo (`http://`, `https://`, etc):

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

## OTROS TIPOS DE URL

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

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

## 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>
```

# CITAS

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

```
<p>Como dijo Groucho Marx: <q>Hijo mío, la felicidad  
está hecha de pequeñas cosas,  
un pequeño yate, una pequeña mansión,  
una pequeña fortuna.</q></p>
```

Como dijo Groucho Marx: "Hijo mío, la felicidad está hecha de pequeñas cosas, un pequeño yate, una pequeña mansión, una pequeña fortuna."

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.

```
ol {  
    list-style-type: lower-alpha;  
}
```

- 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](#) en w3schools.com

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 preformatado. Se respetan  
totalmente los  
saltos de línea     y los espacios     múltiples
```

Aquí otro párrafo.

```
</pre>
```

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

Aquí otro párrafo.

# IMÁGENES

Se insertan mediante el elemento <img>

```

```



Las consideraciones vistas anteriormente 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.

```
  
  
```

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...">
  
</a>
```

Más información en : [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>
```

<th>...</th>	<th>...</th>	<th>...</th>	</tr>
<td>...</td>	<td>...</td>	<td>...</td>	</tr>
<td>...</td>	<td>...</td>	<td>...</td>	</tr>
<td>...</td>	<td>...</td>	<td>...</td>	</tr>

```
</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.

<th>...</th>	<th>...</th>	<th>...</th>
<td>...</td>	<td colspan="2">...</td>	
<td rowspan="3">...</td>	<td>...</td>	<td>...</td>
	<td>...</td>	<td>...</td>
	<td>...</td>	<td>...</td>

# 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

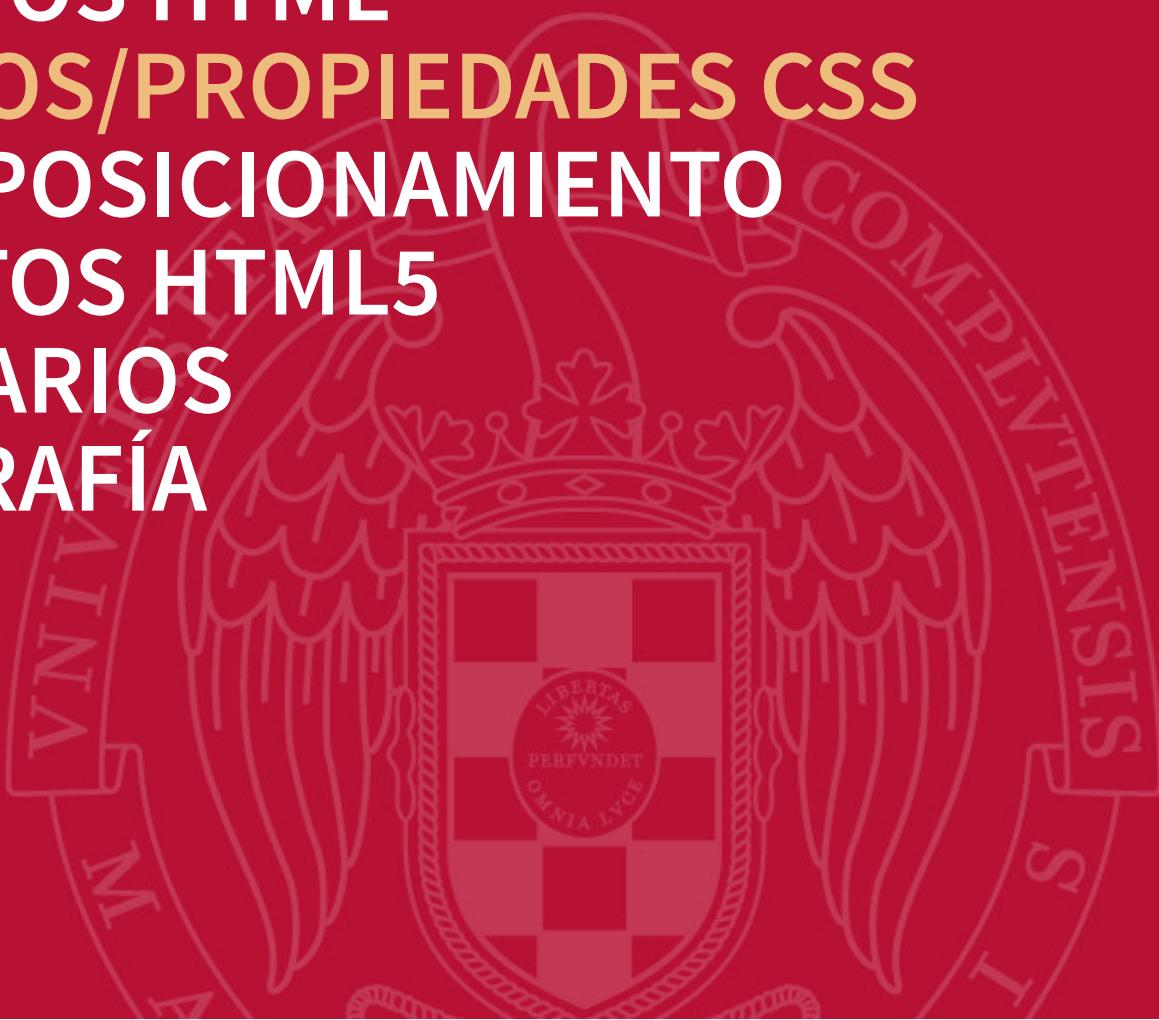
Por defecto, cada celda tiene un borde totalmente separado de las celdas colindantes.

	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/PROPIEDADES CSS
  4. FLUJO Y POSICIONAMIENTO
  5. ELEMENTOS HTML5
  6. FORMULARIOS
  7. BIBLIOGRAFÍA

# ATRIBUTOS/PROPIEDADES 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 **atributos/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</strong>
```

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 {  
    background: lightgray;  
}
```

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 () contenidos en un párrafo (

) de clase **miclase**, ya sea directa o indirectamente a través de otro elemento.

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

Afecta a todos los enlaces () contenidos **directamente** en un párrafo (

) de clase **miclase**.

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

Afecta a todos los párrafos (

) situados **a continuación** de una tabla (

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

Elementos **<img>** que tengan un atributo **src** que valga exactamente **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"] { ... }
```

Enlaces cuyo atributo **href** contenga la cadena **server**.

## PSEUDOCLASES

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

Se aplica a un elemento (`<h1>`) cuando **el ratón está situado sobre él** (o cualquiera de sus descendientes).

```
a:active { ... }
```

Se aplica a un enlace **mientras** se está pulsando sobre él.

```
a:visited { ... }
```

Se aplica a un enlace cuyo destino **ya ha sido visitado**.

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

Filas pares(even)/impares(odd) de una tabla.

En general **nth-child(...)** hace referencia al hermano enésimo dentro de todos los elementos a los que hace referencia el selector que lo precede.

# 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: url(ícono_documento.png);  
}
```

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

## MÁS INFORMACIÓN SOBRE SELECTORES

- [World Wide Web Consortium \(W3C\)](https://www.w3.org/TR/selectors-4/#overview)  
<https://www.w3.org/TR/selectors-4/#overview>
- [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/trysel.asp)  
<http://www.w3schools.com/cssref/trysel.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 ≥ 26
- Firefox ≥ 16
- MS Explorer ≥ 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%, ...

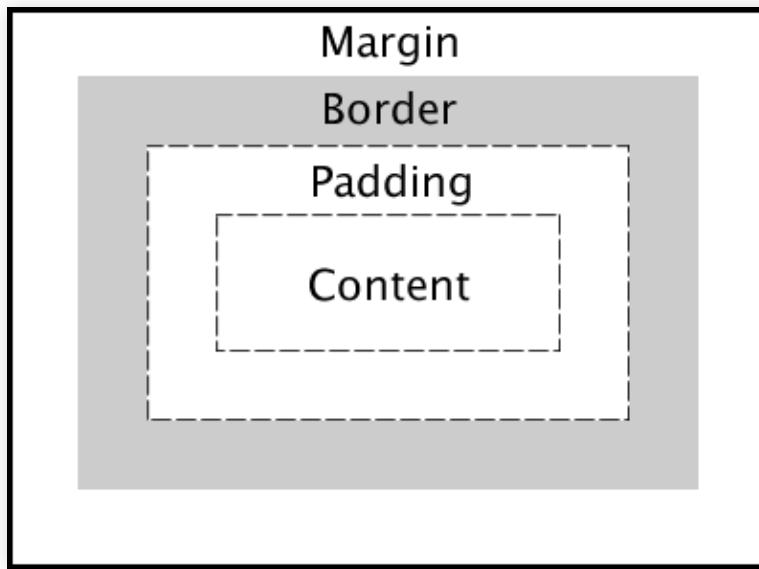
Pueden agruparse en una única propiedad **border**:

```
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.

Más información:

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

# **MODELO DE CAJAS CSS**

Más información:

<https://uniwebsidad.com/libros/css/capitulo-4>

## 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.

# EL BACKGROUND

El background se corresponde con el tamaño total del elemento incluyendo el margen y el padding.

Se puede definir como una única propiedad:

```
background: bg-color bg-image position/bg-size bg-repeat...
```

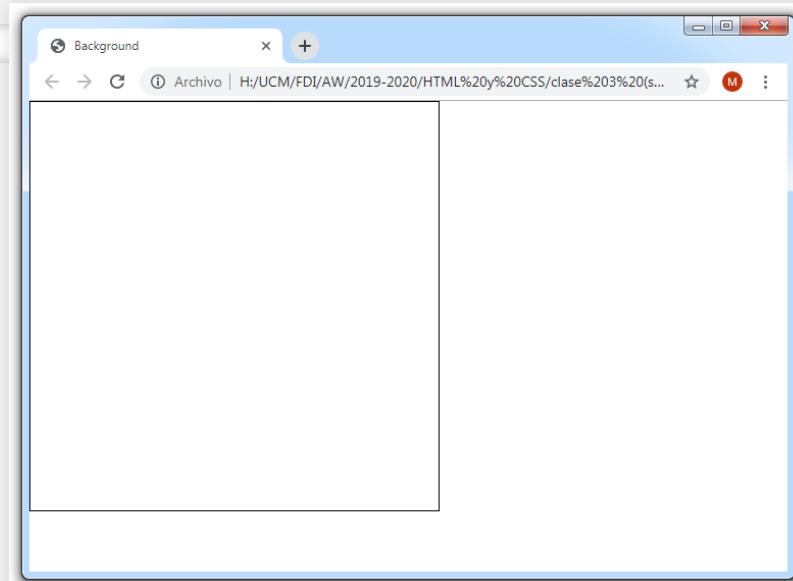
También se puede definir las propiedades de manera independiente:

```
background-color: ...;  
background-image: ...;  
background-position: ...;  
background-size: ...;  
background-repeat...;  
...
```

# EJEMPLO PARA PRUEBA DEL BACKGROUND

```
<div class="bg">  
</div>
```

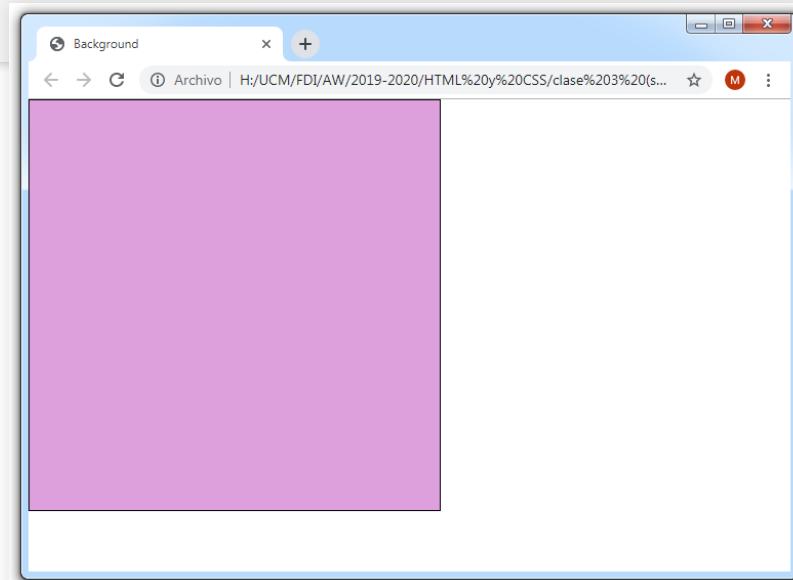
```
* {  
    margin: 0;  
    padding: 0;  
}  
  
.bg {  
    width: 400px;  
    height: 400px;  
    border: 1px solid black;  
}
```



# BACKGROUND-COLOR

Color de fondo del elemento

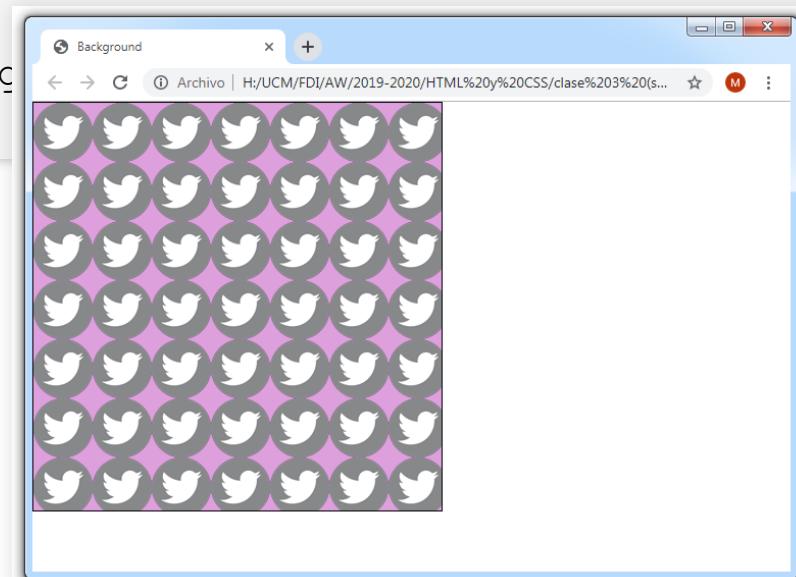
```
.bg {  
    ...  
    background-color:plum;  
}
```



# BACKGROUND-IMAGE

- Imagen de fondo del elemento
- Se sitúa delante del background-color
- Por defecto se repite horizontal y verticalmente
- Por defecto se sitúa en la esquina superior izquierda

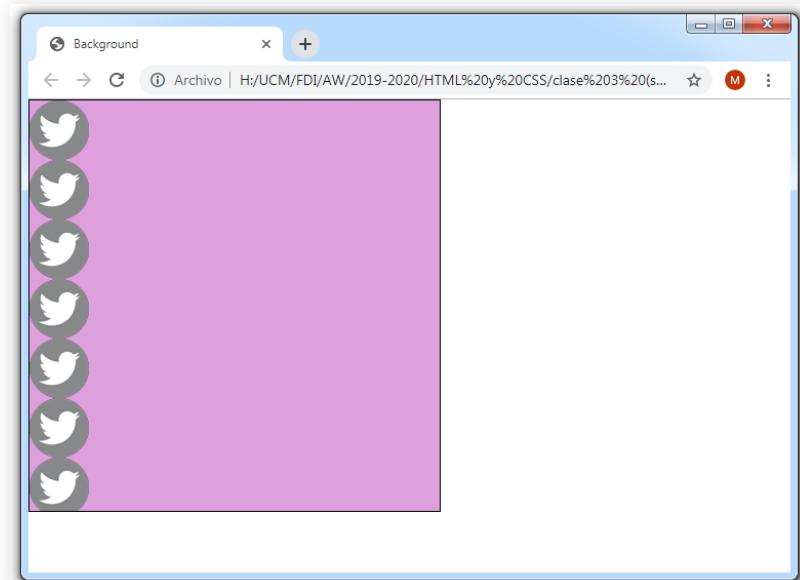
```
.bg {  
    ...  
    background-image:url (twitter.png)  
}
```



# BACKGROUND-REPEAT

- Modo de repetición de la imagen
- Posibles valores: repeat, repeat-x, repeat-y, no-repeat, etc

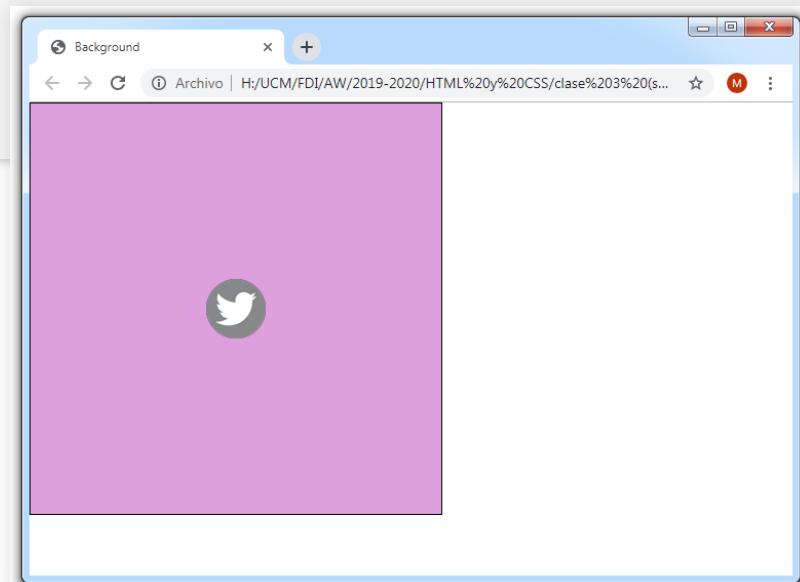
```
.bg {  
    ...  
    background-repeat: repeat-y;  
}
```



# BACKGROUND-POSITION

- Posición de la imagen
- Por defecto en la esquina superior izquierda
- Posibles valores: en unidades, con palabras reservadas (center, top, etc)

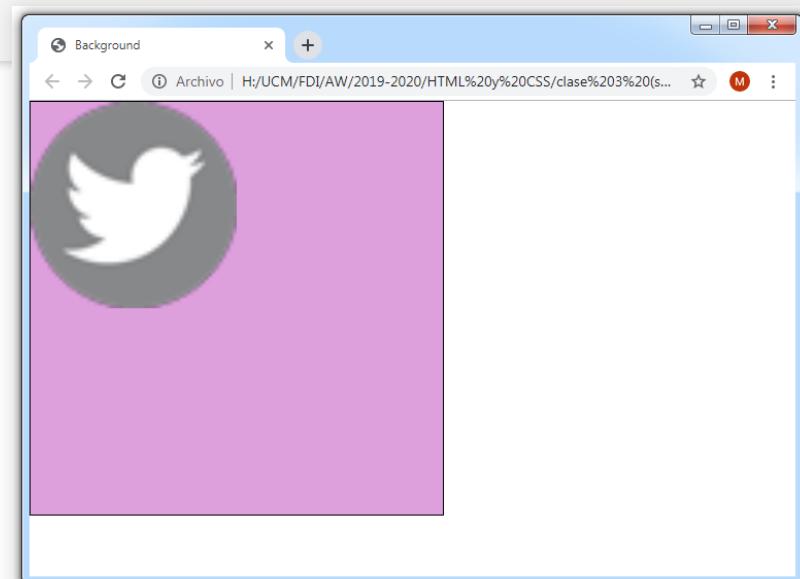
```
.bg {  
    ...  
    background-position: 50% 50%;  
}
```



# BACKGROUND-SIZE

- Tamaño de la imagen
- Se puede especificar el tamaño de la propia imagen o en relación a su contenedor.

```
.bg {  
    ...  
    background-size: 50% 50%;  
}
```



# ATRIBUTOS DEL TEXTO

```
font-family: tipo_de_letral, 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, cursive .

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!

También podemos usar las fuentes gratuitas de Google (+)

Se inserta en el fichero CSS el enlace a la fuente seleccionada:

```
<link href="https://fonts.googleapis.com/css?family=Open+Sans" rel="stylesheet">
```

Posteriormente podemos utilizar esta fuente en el resto del CSS.

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

## 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: 300px;
```

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!**

## EJEMPLO CON IMAGEN DE FONDO

```
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!**



# REGLAS DE PRECEDENCIA

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

**¿Cuál prevalece?**

Veremos 4 reglas para determinar la prevalencia.

# 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.

# Visualización de la regla de especificidad en las herramientas para desarrolladores (1)

The screenshot shows a browser window with a toolbar at the top containing various icons. Below the toolbar is a navigation bar with back, forward, and search buttons, followed by a URL bar containing the path: file:///H:/UCM/FDI/AW/2018-2019/Tema%202%20HTML%20y%20CSS/herencia.html. To the right of the URL bar are a star icon and a refresh button.

The main content area displays a paragraph of text: "Esto es un enlace normal, este tiene una clase y este tiene un identificador." A red circle highlights the word "normal" in the first sentence.

On the right side of the screen, the browser's developer tools are open, specifically the "Elements" tab. The DOM tree shows the structure: html > body > p > a. The "Styles" tab is selected, showing the computed styles for the highlighted element. A filter bar above the styles list contains ":hov .cls +".

The styles listed are:

- element.style {  
 color: black; } herencia.css:15
- p > a {  
 color: red; } herencia.css:15
- a {  
 color: blue; } herencia.css:11
- a:webkit-any-link {  
 color: webkit link; cursor: pointer; text-decoration: underline; } user agent stylesheet

At the bottom of the developer tools, there are tabs for Console, Rendering, Coverage, and a dropdown menu for Filter and Default levels.

# Visualización de la regla de especificidad en las herramientas para desarrolladores (2)

The screenshot shows a browser window with the URL `file:///H:/UCM/FDI/AW/2018-2019/Tema%202%20HTML%20y%20CSS/herencia.html`. The page content includes the text: "Esto es un enlace normal, este tiene una clase y este tiene un identificador". To the right, the developer tools' Elements tab is open, showing the DOM structure with nodes `html`, `body`, and `p`. A red arrow points from the text on the page to the `Computed` styles panel. The `Computed` tab is selected, displaying the following styles:

Style	Value
color	rgb(255, 0, 0)
cursor	pointer
display	inline
height	auto
text-decoration-color	rgb(255, 0, 0)
text-decoration-line	underline
text-decoration-style	solid
width	auto

Below the styles, there is a section for `Rendered Fonts`.

## 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.

# Inserción del atributo **style** en las herramientas para desarrolladores (1)

The screenshot shows the Chrome DevTools interface with the 'Elements' tab selected. A red box highlights the code block where the `style` attribute is being added to an anchor tag. A large red arrow points from the bottom left towards the styles panel.

Code in the Elements tab:

```
<!doctype html>
<html lang="es">
  <head>...</head>
  <body>
    ... <a href="#" style="font-weight:bold; color: yellow">
      Volver
    </a> == $0
  </body>
```

Styles panel:

- Filter: element.style
- Rules:
  - element.style {  
 font-weight: bold;  
 color: yellow;  
}
  - a {  
 color: blue;  
}
  - a:-webkit-any-link {  
 color: webkit-link;  
 cursor: pointer;  
 ...
- Source: herencia.css:11
- Source: user agent stylesheet

Computed styles (right panel):

- margin -
- border -
- padding -
  - auto x auto -
  - -
  - -
  - -

## REGLA 4: LA MARCA !important

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

# Visualización de la herencia en las herramientas para desarrolladores

The screenshot shows a browser window with a toolbar at the top and a developer tools panel on the right. The developer tools panel has tabs for Elements, Console, Sources, Network, and Properties. The Elements tab is active, showing the DOM structure:

```
Introducimos un "enlace" dentro del párrafo
...
<a href="...">enlace</a> == $0
"
    dentro del párrafo
html body p.cuadro a
```

The Styles tab is selected, displaying the CSS rules applied to the selected element (`a`):

```
element.style {
}
a:-webkit-any- user agent stylesheet
link {
    color: -webkit-link;
    cursor: pointer;
    text-decoration: underline;
}
Inherited from body
body {
    font-family: sans-serif;
}
```

The Properties tab shows the computed styles for the `a` element, which includes the inherited `font-family` and additional properties like `color`, `cursor`, and `text-decoration`. A large red arrow points from the text "Introducimos un enlace dentro del párrafo" to the developer tools panel.

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

Properties

Filter

margin  
border  
padding  
auto x auto

color: -webkit-link;  
cursor: pointer;  
text-decoration: underline;

Inherited from body

body {  
 font-family: sans-serif;  
}

herencia.css:1

37.3

Un elemento puede sobreescribir las propiedades heredadas. El valor sobreescrito prevalece sobre el heredado.

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

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

# Visualización de la herencia en las herramientas para desarrolladores

The screenshot shows a browser window with a toolbar at the top and a developer tools panel on the right. The developer tools panel has tabs for Elements, Console, Sources, Network, and Properties. The Elements tab is active, showing the DOM structure with a selected element. The Styles tab is also visible. A red box highlights the CSS rules for the selected element, which includes styles from both the current rule and the body element. The Properties tab shows a visual representation of the element's bounding box with color-coded layers for margin, border, and padding.

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

```
Introducimos un "
...
<a href="#">enlace</a> == $0
"
dentro del párrafo
html body p.cuadro a
```

Styles Event Listeners DOM Breakpoints Properties Accessibility

Filter .hov .cls |

```
a {
    herencia.css:1
    font-family: serif;
}
```

a:-webkit- user agent stylesheet

```
any-link {
    color: -webkit-link;
    cursor: pointer;
    text-decoration: underline;
}
```

Inherited from body

```
body {
    herencia.css:1
    font-family: sans-serif;
}
```

margin -

border -

padding -

auto x auto

Filter Show all

color

rgb(0...

Un elemento puede **heredar propiedades no heredables** estableciendo el valor **inherit** en la propiedad correspondiente:

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

Introducimos un enlace dentro del párrafo.

# MÚLTIPLES HOJAS DE ESTILO

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**

Una página puede tener asociadas distintas hojas de estilo para distintos dispositivos.

```
<!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 y **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.

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"

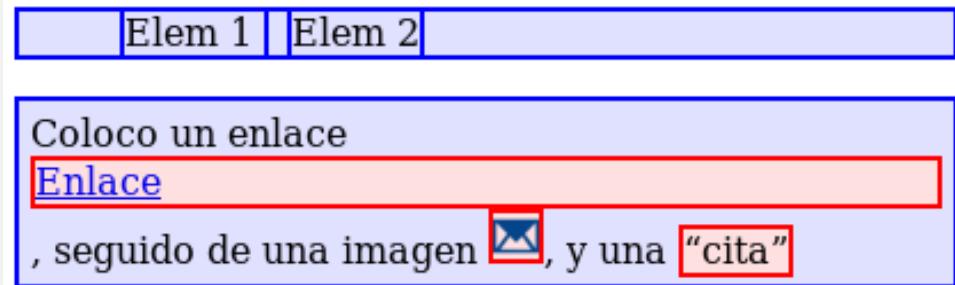
# EL ATRIBUTO `display`

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 (flujo horizontal) pero pueden tener márgenes (top y bottom) 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;
    border-radius: 10px;
}
```

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

Elementos div - Mozilla Firefox

Elementos div

file:///home/manuel/Docencia/AW/git/slides/resources/02/Div.html

Bienvenido a mi página!

## Articulo 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.

## Articulo 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.

Generador Lorem Ipsum

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:

The screenshot shows a Firefox browser window titled "Elementos float - Mozilla Firefox". The address bar displays "file:///home/manuel/Docencia/AW/git/slides/resc". The main content area contains four paragraphs. Paragraph P1 is at the top. Paragraph P2 is a floating element positioned to the left of P3. Paragraph P3 is to the right of P2. Paragraph P4 is at the bottom. The paragraph containing P2 is highlighted with an orange background.

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

The screenshot shows a Mozilla Firefox browser window with the title "Elementos float - Mozilla Firefox". The address bar displays "file:///home/manuel/Docencia/AW/git/slides/resc". The browser interface includes standard buttons for back, forward, search, and menu.

The content area contains four paragraphs (P1, P2, P3, P4) arranged with floating properties:

- 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. (This paragraph is highlighted with a yellow background)
- 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.

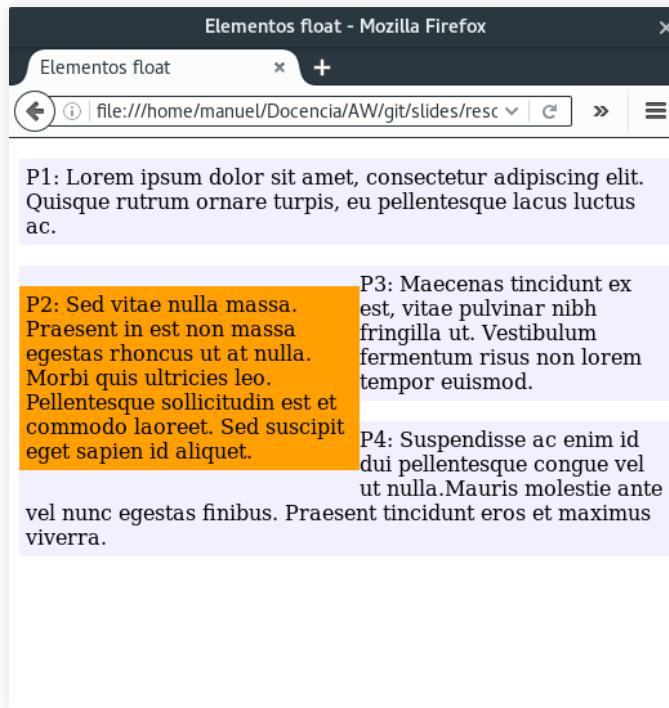
## ¿QUÉ PASA SI...?

- 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.

The screenshot shows a Firefox browser window titled "Elementos float - Mozilla Firefox". The address bar displays "file:///home/manuel/Docencia/AW/c". The main content area contains four paragraphs (P1, P2, P3, P4) arranged with different floating properties applied to them.

- P1:** A light blue box containing the text "P1: Lorem ipsum dolor sit amet, consectetur adipiscing elit. Quisque rutrum ornare turpis, eu pellentesque lacus luctus ac."
- P2:** An orange box containing the text "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:** A light blue box containing the text "P3: Maecenas tincidunt ex est, vitae pulvinar nibh fringilla ut. Vestibulum fermentum risus non lorem tempor euismod."
- P4:** A light blue box containing the text "P4: Suspendisse ac enim id dui pellentesque congue vel ut nulla. Mauris molestie ante vel nunc egestas finibus. Praesent tincidunt eros et maximus viverra."

```
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 lacinia 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 lacinia 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  
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.

# 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 ru P2: Sed vitae nulla massa. tesque lacus luctus ac.  
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: Maecel  
Vestibulum  
P4: Suspen  
nulla. Maur  
tincidunt ei  
os et maximus viverra.

vinar nibh fringilla ut.  
tempor euismod.  
que congue vel ut  
stas finibus. Praesent

# 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.

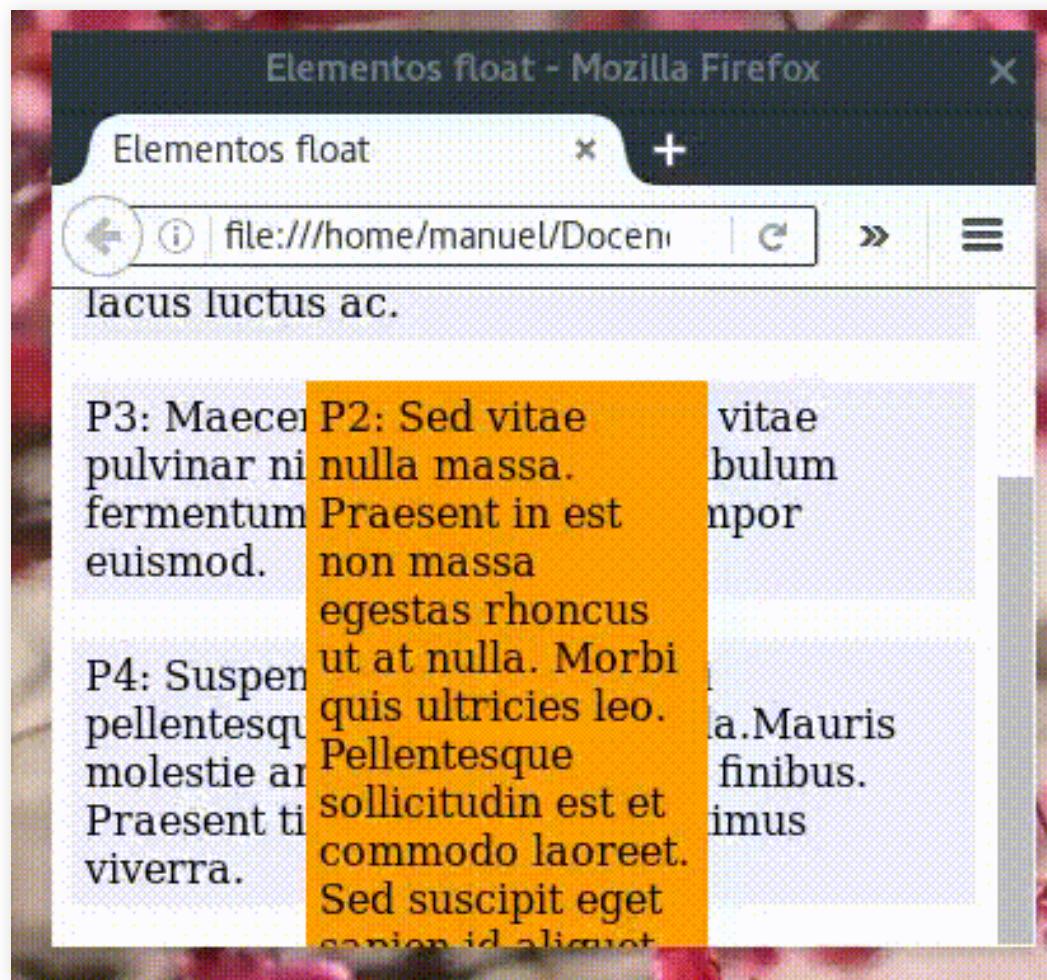
P4: Suspendisse ac enim id dui pellentesque congue vel ut

# 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 (display:flex)

Tipo de posicionamiento que permite configurar la disposición de los elementos dentro de un contenedor.

Soporte en navegadores: consultar <https://caniuse.com/#search=flex>

The screenshot shows a browser window displaying the Can I use... website at <https://caniuse.com/#search=flex>. The search bar contains 'flex'. Below the search bar, it says '2 results found'. The first result is 'CSS Flexible Box Layout Module' with a brief description: 'Method of positioning elements in horizontal or vertical stacks. Support includes all properties prefixed with flex, as well as display: flex, display: inline-flex, align-content, align-items, align-self, justify-content and order.' To the right of the description is a table showing usage statistics:

Usage	% of all users
Spain	94.77% + 1.63% = 96.39%
unprefixed:	94.52% + 1.33% = 95.85%
Global	92.41% + 3.31% = 95.71%
unprefixed:	92.13% + 2.69% = 94.82%

# EJEMPLO

```
<div class="contenedor">
  <div class="elemento">1</div>
  <div class="elemento">2</div>
  <div class="elemento">3</div>
  <div class="elemento">4</div>
</div>
```

```
.contenedor {
  display: flex;
  width: 50%;
  background-color: blue;
}

.elemento {
  margin: 4px;
  font-size: 30px;
  text-align: center;
  background-color: white;
}
```

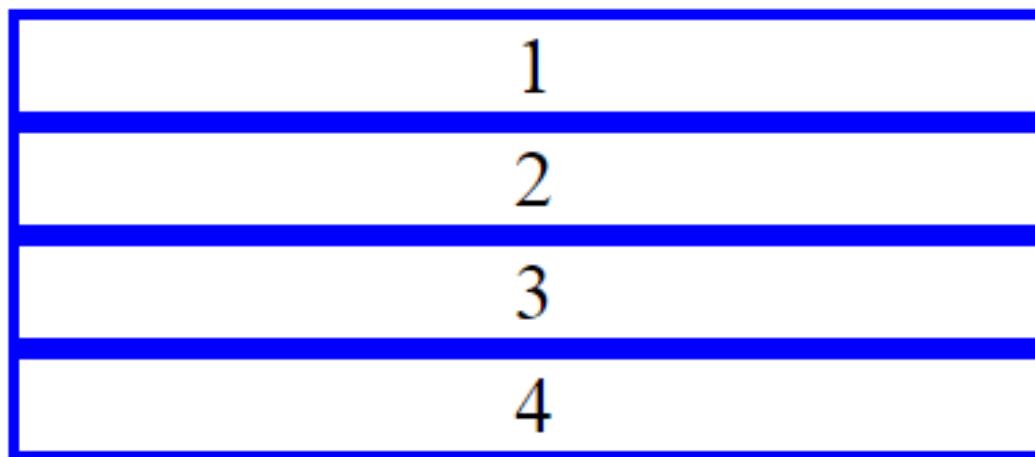


La propiedad **flex-direction** determina el **eje principal**.

- **flex-direction: row** (valor por defecto)



- **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** (valor por defecto)



- **flex-end**



- **center**



- **space-between**



- **space-around**



- **space-evenly**



# Disposición a lo largo del eje secundario:

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

- **flex-start**



- **center**



- **flex-end**



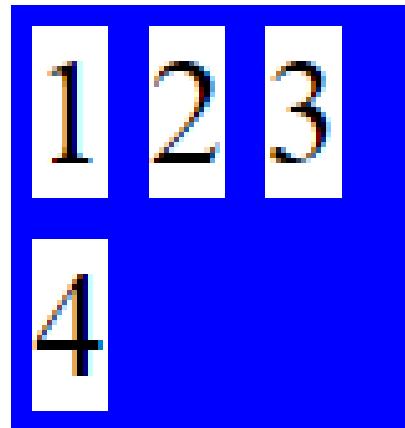
- **stretch** (valor por defecto)



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

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

- **wrap**



- **nowrap** (valor por defecto)

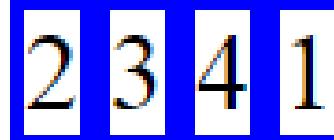


# PROPIEDADES DE LOS ELEMENTOS HIJOS

`order: número`

Determina el orden del elemento dentro del parente.

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



```
align-self: flex-start | flex-end | center | ...
```

Alineación de un elemento hijo con respecto al eje secundario.

Por ejemplo, si asignamos al cuarto elemento

**align-self: flex-end**

y al segundo elemento elemento

**align-self: center**



`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:



## **flex-basis, flex-grow, flex-shrink y flex**

Los atributos **flex-basis**, **flex-grow** y **flex-shrink** trabajan conjuntamente.

El atributo **flex** se utiliza para expresar los tres atributos a la vez (flex-grow, flex-shrink, flex-basis).

## flex-basis

Es el tamaño "de base" (inicial) de un elemento.

- Puede crecer si el contenedor tiene espacio sobrante.
- Puede decrecer si al contenedor le falta espacio.
- Espacio sobrante: la suma de los tamaños de los elementos es inferior al tamaño del contenedor.
- Falta de espacio: la suma de los tamaños de los elementos es superior al tamaño del contenedor.

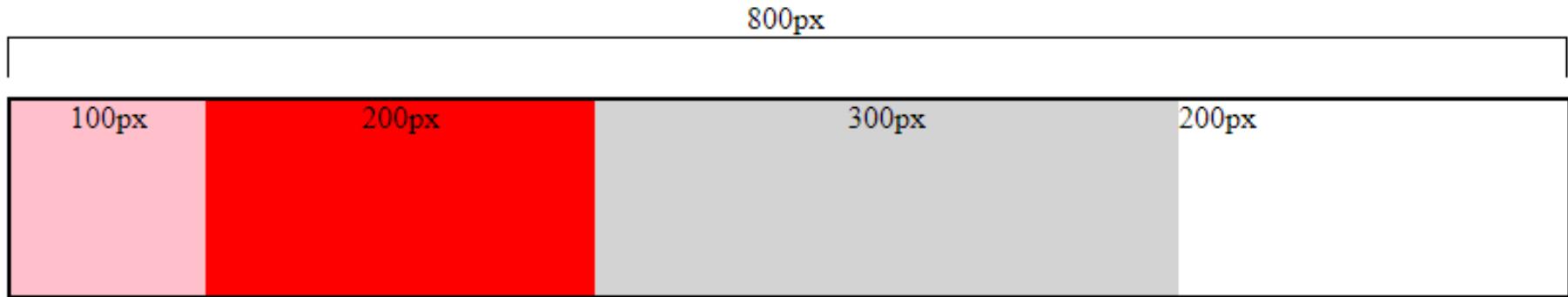
# ejemplo con espacio sobrante (1)

```
<div class="contenedor">
  <div class="elemento1"></div>
  <div class="elemento2"></div>
  <div class="elemento3"></div>
</div>
```

```
.contenedor {
  display:flex;
  width: 800px;
  height: 100px;
  border:solid;
  border-color:black;
  border-width: 2px;
}

.elemento1 {
  flex-basis: 100px;
  background-color: pink;
}
.elemento2 {
  flex-basis: 200px;
  background-color: red;
}
.elemento3 {
  flex-basis: 300px;
  background-color: lightgray;
}
```

## ejemplo con espacio sobrante (2)



La suma de los tamaños de los elementos es 600px (100px + 200px + 300px) y el contendor es de 800px.

El contenedor tiene un espacio sobrante de 200px, ¿cómo se distribuye? ... depende de **flex-grow**

## flex-grow

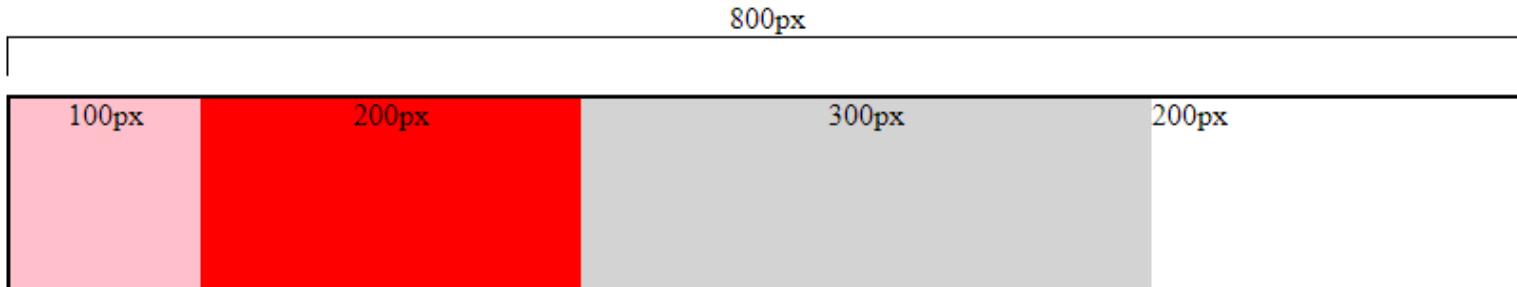
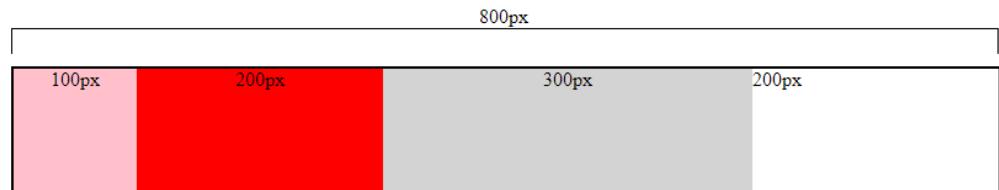
Indica la proporción del espacio sobrante que se asignará al elemento

- Si vale 0, el elemento no se expande (valor por defecto).
- Si vale n, el elemento crecerá en n unidades.
- Unidad: espacio sobrante del contenedor dividido entre la suma de los flex-grow de sus elementos.

# ejemplo con espacio sobrante (3)

```
.elemento1 {  
    flex-basis: 100px;  
    background-color: pink;  
    flex-grow: 0;  
}  
  
.elemento2 {  
    flex-basis: 200px;  
    background-color: red;  
    flex-grow: 0;  
}  
  
.elemento3 {  
    flex-basis: 300px;  
    background-color: lightgray;  
    flex-grow: 0;  
}
```

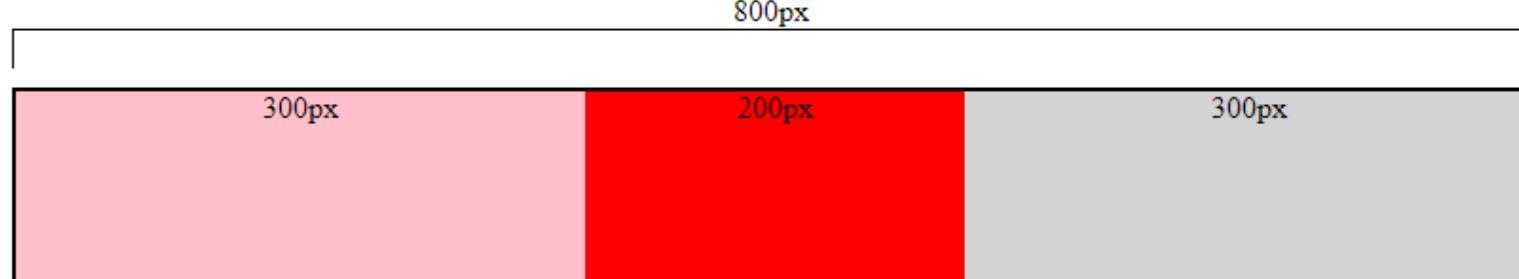
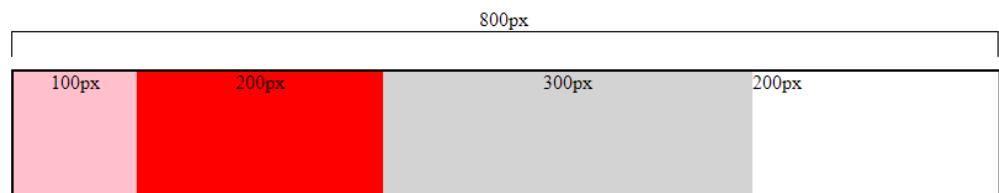
## Situación de partida



# ejemplo con espacio sobrante (4)

```
.elemento1 {  
    flex-basis: 100px;  
    background-color: pink;  
    flex-grow: 1;  
}  
  
.elemento2 {  
    flex-basis: 200px;  
    background-color: red;  
    flex-grow: 0;  
}  
  
.elemento3 {  
    flex-basis: 300px;  
    background-color: lightgray;  
    flex-grow: 0;  
}
```

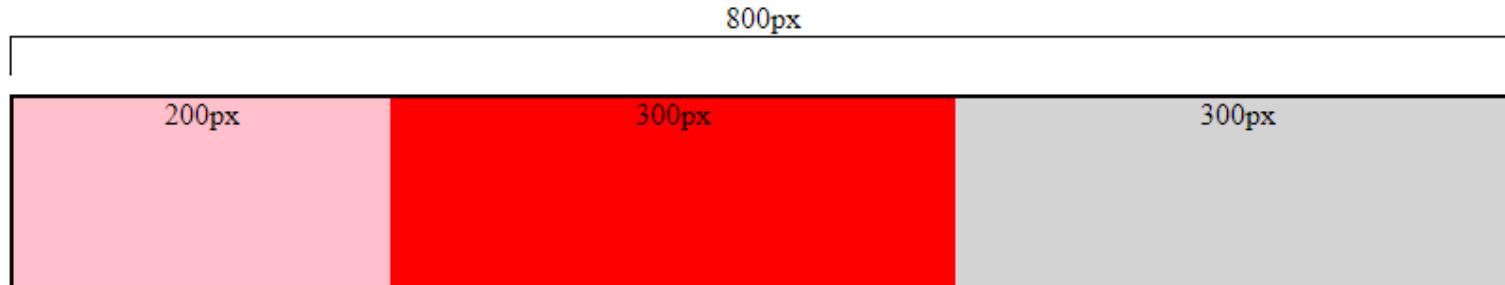
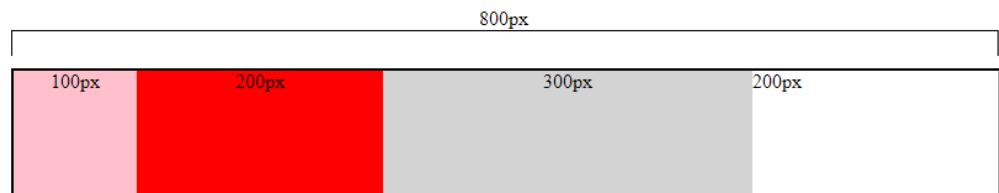
## Situación de partida



# ejemplo con espacio sobrante (5)

```
.elemento1 {  
    flex-basis: 100px;  
    background-color: pink;  
    flex-grow: 1;  
}  
  
.elemento2 {  
    flex-basis: 200px;  
    background-color: red;  
    flex-grow: 1;  
}  
  
.elemento3 {  
    flex-basis: 300px;  
    background-color: lightgray;  
    flex-grow: 0;  
}
```

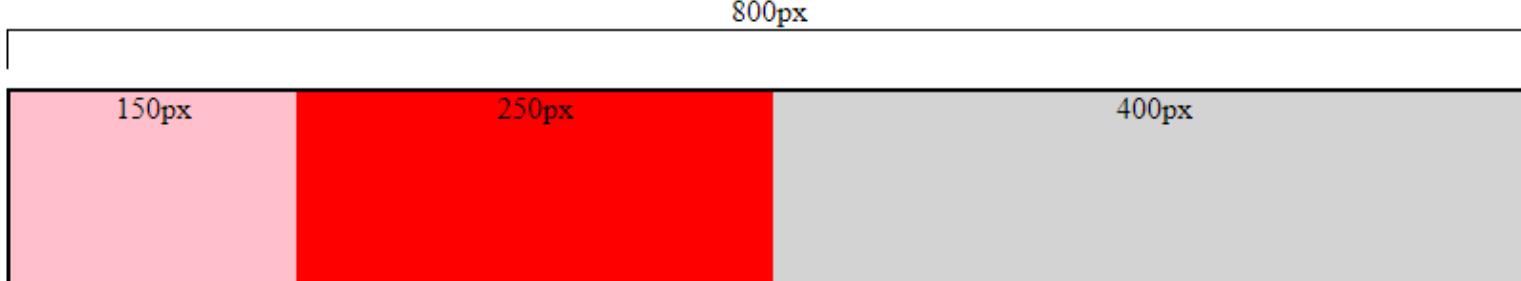
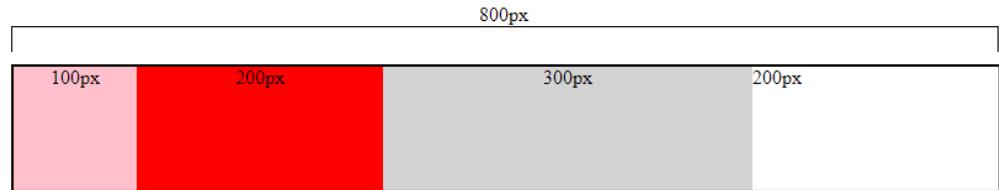
## Situación de partida



# ejemplo con espacio sobrante (6)

```
.elemento1 {  
    flex-basis: 100px;  
    background-color: pink;  
    flex-grow:1;  
}  
  
.elemento2 {  
    flex-basis: 200px;  
    background-color: red;  
    flex-grow:1;  
}  
  
.elemento3 {  
    flex-basis: 300px;  
    background-color: lightgray;  
    flex-grow:2;  
}
```

## Situación de partida



## flex-shrink

Su funcionamiento es similar (pero inverso) al de **flex-grow**.

Indica la proporción en que se reducirá el elemento si al contenedor le falta espacio.

- Si vale 0, el elemento no se contrae.
- Si vale n, el elemento contraerá en n unidades.
- Unidad: espacio que le falta al contenedor dividido entre la suma de los flex-shrink de sus elementos.
- El valor por defecto es 1.

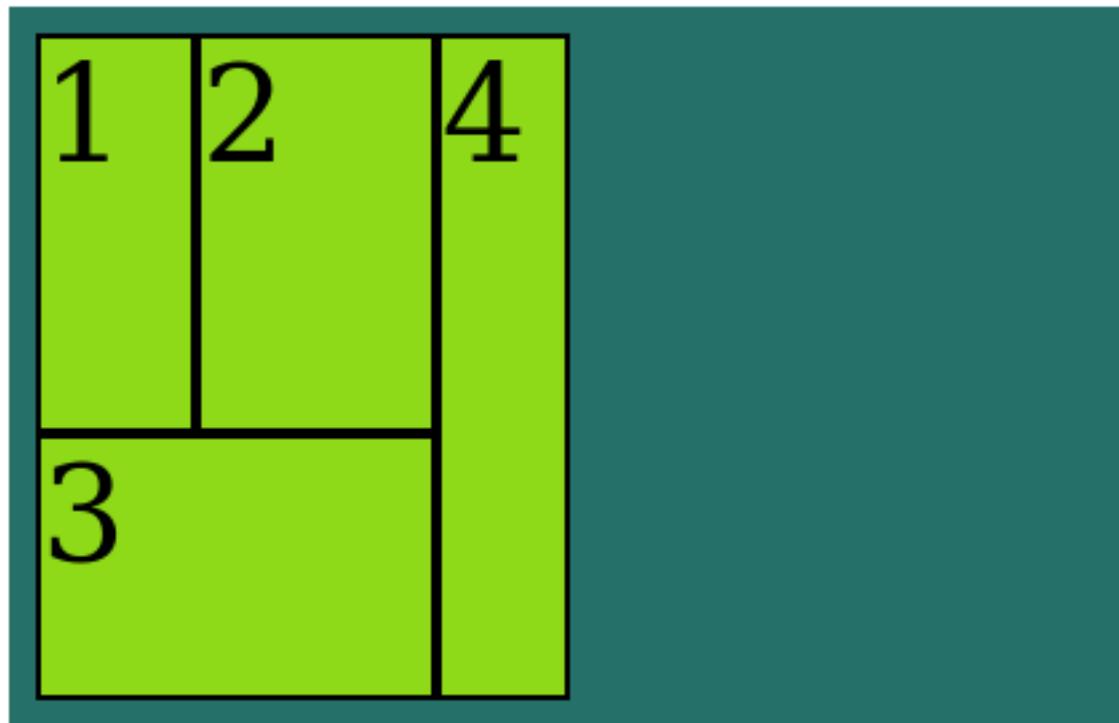
Más información:

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

[https://www.w3schools.com/css/css3\\_flexbox.asp](https://www.w3schools.com/css/css3_flexbox.asp)

# POSICIONAMIENTO EN REJILLA (GRID)

Mecanismo de posicionamiento de elementos en forma de rejilla flexible.



# EJEMPLO

Partimos del siguiente código:

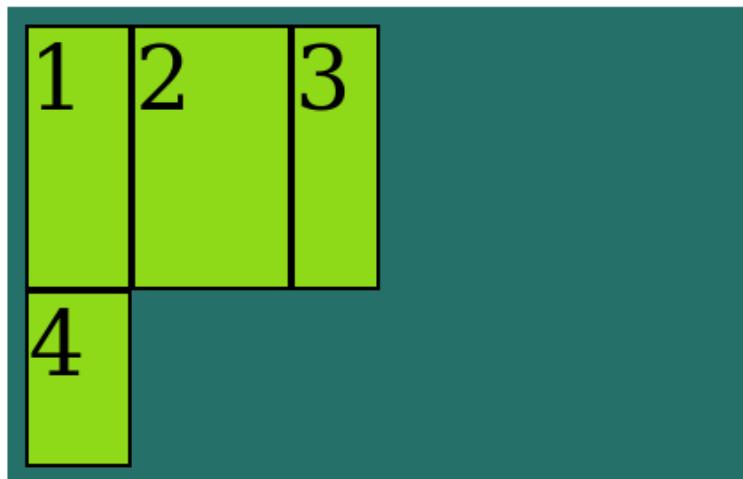
```
<div class="contenedor">
  <div id="uno" class="elemento">1</div>
  <div id="dos" class="elemento">2</div>
  <div id="tres" class="elemento">3</div>
  <div id="cuatro" class="elemento">4</d
</div>
```



```
.contenedor {
  width: 400px;
  padding: 10px;
  display: grid;
  background-color: green
}
.elemento {
  font-size: 30px;
  background-color: lime;
  border: solid black 2px
}
```

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

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



Es posible especificar unidades relativas, indicando los **pesos** de cada una de las filas o columnas ( en % o "fr" ).

```
.contenedor {  
    ...  
  
    grid-template-rows: 2fr 1fr;      /* Dos filas */  
  
    grid-template-columns: 1fr 3fr 1fr;    /* Tres columnas */  
}
```



Se puede utilizar **auto** para que el navegador calcule automáticamente el tamaño de las filas/columnas.

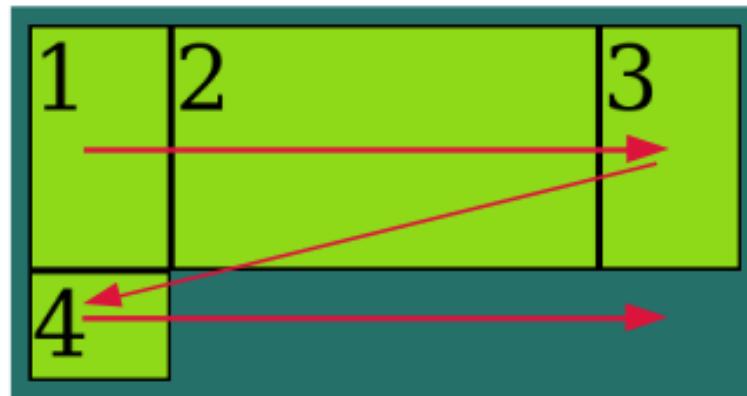
```
<div class="contenedor">  
  <div class="elemento">1</div>  
  <div class="elemento">2</div>  
  <div class="elemento">3</div>  
  <div class="elemento">4</div>  
  <div class="elemento">5</div>  
  <div class="elemento">6</div>  
  <div class="elemento">7</div>  
  <div class="elemento">8</div>  
  <div class="elemento">9</div>  
</div>
```

```
.contenedor {  
  width: 90%;  
  padding: 10px;  
  display: grid;  
  background-color: green;  
  grid-template-rows: auto auto auto;  
  grid-template-columns: auto auto auto;  
}
```

1	2	3
4	5	6
7	8	9

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

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

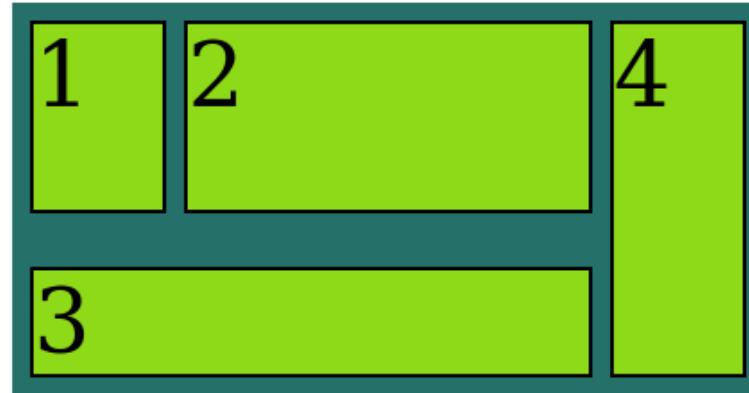


También podíamos haber utilizado:

```
#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:

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



## OTRAS PROPIEDADES

- **justify-items, align-items**  
Similares a las de Flex, 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



# APORTACIONES DE HTML5

- Etiquetas semánticas
- Reproducción de audio y video
- Mejora de los formularios
- Otros (elementos para dibujar y soporte para gráficos vectoriales)

# ETIQUETAS SEMÁNTICAS

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.

## ALGUNAS ETIQUETAS SEMÁNTICAS

- <**header**> Contenido situado en la parte superior de la página (título, logo, etc.) o de una sección de ésta.
- <**main**> Contenido principal.
- <**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.
- <**aside**> Contenido complementario a la página (suele estar en el margen derecho).
- <**article**> Composición autocontenido 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>
  <main>
    <article>
      ...
    </article>
    <article>
      ...
    </article>
  </main>
  <footer>
    ...
  </footer>
</body>
```

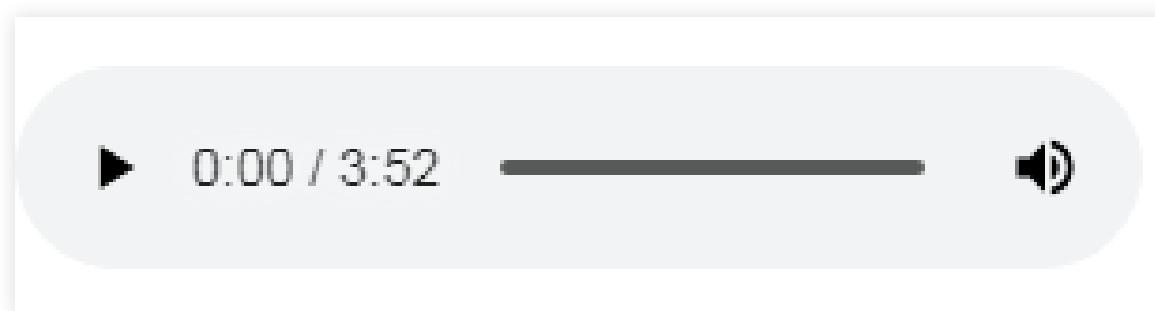
# REPRODUCCIÓN DE AUDIO Y VIDEO

Las etiquetas `<audio>` y `<video>` permiten insertar elementos multimedia.

```
<audio src="fichero de audio.mp3"></audio>
```

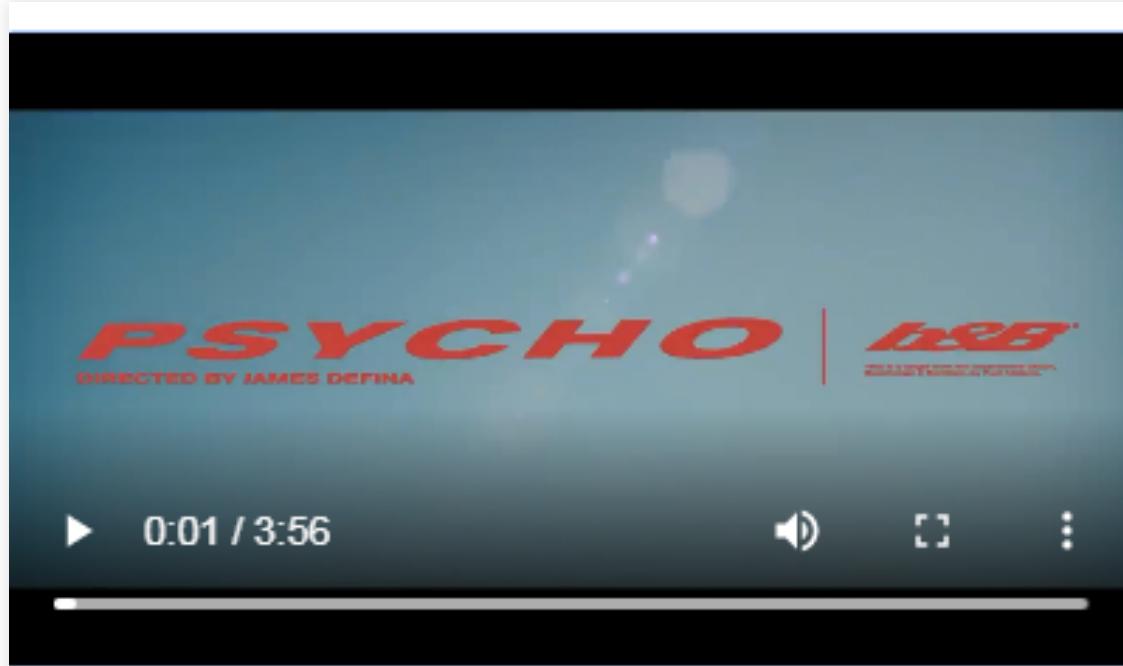
Si el navegador no soporta la etiqueta mostrará el contenido comprendido entre `<audio>` y `</audio>`.

```
<audio src="fichero de audio.mp3" controls></audio>
```



Es aconsejable dimensionar siempre que se utilice la etiqueta <video>.

```
<video src="fichero de video.mp4" controls></video>
```



# VALIDADOR HTML5

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

The screenshot shows a Mozilla Firefox browser window with the title bar "The W3C Markup Validation Service - Mozilla Firefox". The address bar contains the URL "https://validator.w3.org". The main content area displays the "Markup Validation Service" page. At the top left is the W3C logo. To its right is the text "Markup Validation Service" and "Check the markup (HTML, XHTML, ...) of Web documents". Below this are three tabs: "Validate by URI" (selected), "Validate by File Upload", and "Validate by Direct Input". Under the "Validate by URI" tab, there is a sub-section titled "Validate by URI" with the sub-instruction "Validate a document online:" followed by an "Address:" input field containing a placeholder URL. Below this is a link "More Options". A large "Check" button is centered below the input fields. At the bottom of the page, a descriptive text explains the validator's function and links to other validation tools. On the left side of the bottom section is a small button labeled "I ❤ VALIDATOR". On the right side, there is a "5856" counter next to a "Flattr" button.

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](#).

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

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

# FORMULARIOS GET VS POST

```
<form action="procesar_formulario" method="método">  
    Nombre: <input type="text" name="nombre">  
    <input type="submit" value="Enviar consulta">  
</form>
```

Enviamos la siguiente información:

Nombre: Gerardo

# MÉTODO GET

```
GET http://.../procesar_formulario?nombre=Gerardo
```

La información se envía como parte de la URL.

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

# MÉTODO POST

```
POST http://.../procesar_formulario  
... 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 desplegables
- Etiquetas

# BOTONES

```
<button type="submit">Enviar</button>
<button type="reset">Borrar formulario</button>
<button type="button">Otro</button>
```

Enviar

Borrar formulario

Otro

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 (atributo **value**).

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">

Campo para una única línea de texto.

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



Me llamo Elisa

Datos enviados:

```
nombre_campo=Me+llamo+Elisa
```

Si no existe el atributo **name** el campo no se envía al servidor.

# OTROS TIPOS DE CUADROS DE TEXTO

```
<div>Contraseña: <input type="password" name="contrasenya"></div>
<div>Edad: <input type="number" name="edad"></div>
<div>Peso: <input type="range" name="peso" min="30" max="200"></div>
<div>Color: <input type="color" name="color_favorito"></div>
<div>Fecha de nacimiento:<input type="date" name="fecha_nacimiento">
```

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

The image shows a web page with a light gray background and a white rectangular form area. Inside the form, there are five input fields:

- Contraseña:** A password input field containing six dots (.....).
- Edad:** A number input field containing the value "15".
- Peso:** A range input field with a horizontal slider bar and a small square slider handle positioned near the center.
- Color:** A color input field displaying a solid orange square.
- Fecha de nacimiento:** A date input field containing the value "01/12/2003".

# 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:

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

Fumador:

Fumador:

```
<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:

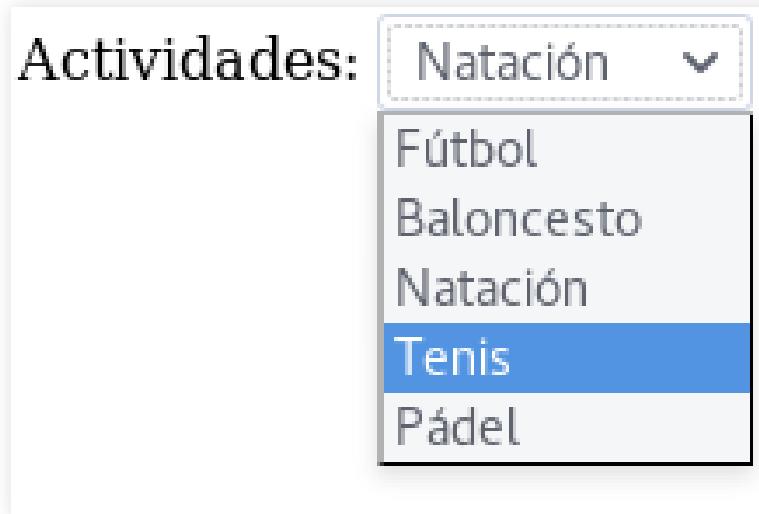
```
<input type="radio" name="sexo" value="hombre"> Hombre  
<input type="radio" name="sexo" value="mujer"> Mujer
```

Sexo:  Hombre  Mujer

Al enviar el formulario se añade el parámetro **sexo=hombre** a la petición.

# 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>
```



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">
```

# VALIDACION DE FORMULARIOS (HTML5)

Además de los nuevos tipos de entrada (color, date, email, etc), HTML5 También define nuevas propiedades/atributos que permiten la validación de formularios en el cliente:

- **required**: el campo es obligatorio.
- **pattern**: expresión regular.
- **min**: valor mínimo.
- **max**: valor máximo.

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/>

