

# Creación y aplicación de estilos en la interfaz web. CSS3

# 0. Introducción

Para crear el contenido y dar estructura y semántica a una web se utiliza el lenguaje HTML . Sin embargo, no sólo es importante el contenido, sino que también lo son, y mucho, el diseño y la presentación de la información. Es decir, la utilización del color, la tipografía, la imagen y los elementos interactivos que hacen la interfaz agradable para el usuario que accede a la web.

Así, para mejorar el aspecto y el mantenimiento de un sitio web es indispensable la utilización de estilos en lenguaje CSS , Cascading Style Sheets. Este lenguaje controla la presentación de los documentos HTML de forma que permite separar la presentación del contenido de la página web y le aporta un formato al documento en HTML mucho más rico en diseño. Actualmente, CSS está soportado por todos los navegadores web.

# 1. Hojas de estilos: CSS3

Las hojas de estilos en cascada o CSS corresponden a un conjunto de reglas usadas para definir y crear la presentación de un documento escrito en HTML . El organismo encargado de regular su especificación es el World Wide Web Consortium ( W3C ).

El estándar actual de W3C para la aplicación de estilos es CSS3, y es totalmente compatible con sus versiones anteriores.

Puedes consultar el estándar CSS3  
en: [www.w3.org/TR/CSS](http://www.w3.org/TR/CSS) .



El lenguaje CSS indica en el navegador cómo visualizar el contenido de la página web.



# 1. Hojas de estilos: CSS3

Las ventajas que ofrece la utilización de hojas de estilo son:

- Separar la estructura de las páginas y su contenido (en HTML ), del formato del texto y de la página (en CSS ).
- Tener muchas más posibilidades de formato y presentación de una página web.
- Unificar el diseño de las páginas web del sitio web, para así definir un estilo una sola vez y poder aplicarlo tantas veces como se desee.
- Reutilización de las mismas hojas de estilo para diferentes documentos HTML .
- Estándar de W3C , que ofrece compatibilidad con versiones anteriores y con los distintos navegadores.

# 1. Hojas de estilos: CSS3

## Incluir estilos CSS en un documento HTML

Para crear un documento web con estilos CSS se parte de un documento HTML con los contenidos etiquetados para designar la función de cada elemento dentro del documento, es decir, los encabezados, párrafos, imágenes, enlaces y otros elementos. Seguidamente, se utiliza el lenguaje CSS para definir el aspecto de cada elemento HTML , como el color, el tamaño, el tipo de letra y la posición.

Además, los estilos CSS de un documento web se pueden definir en un documento de texto externo con la extensión .css, o también se pueden definir en el mismo documento HTML .



# 1. Hojas de estilos: CSS3

## Enlazar una hoja de estilos externa

Una hoja de estilos externa es un documento de texto con extensión .css en el que están definidos los estilos de un documento web.

```
h1 {  
  color : #FFFFFF ;  
  background-color : #999999 ;  
}  
  
p {  
  fuente-family : Georgia , Helvetica ;  
}
```

Para enlazar un documento CSS a una página web se utiliza la etiqueta <link>, la cual se especifica en la cabecera, <head>, de la página web.

Es aconsejable utilizar una hoja de estilos externa, puesto que permite su reutilización y facilita el mantenimiento del sitio web. Al tener los estilos centralizados en uno o varios documentos, cualquier modificación resulta más sencilla.

```
<head>  
  <link rel = "stylesheet" href = "../css/estilos.css"/>  
  <meta charset="utf-8" />  
  <title> Primera Página Web </title>  
</head>
```

# 1. Hojas de estilos: CSS3

Los atributos del elemento link más relevantes son:

Atributo	Descripción
href	Especifica la ruta del documento .css.
rel	Define la relación entre los documentos .css y .html. Puede ser <code>stylesheet</code> o bien <code>alternate</code> .
type	Indica en el navegador el tipo de recurso. En este caso: <code>text/css</code> .
title	Especifica un título en el recurso.
media	Especifica los medios en los que se aplica en hoja de estilos.

# 1. Hojas de estilos: CSS3

## Incrustar una hoja de estilos en HTML

Una hoja de estilos puede incrustarse en el documento HTML con el elemento `<style>`, el cual se suele definir en la cabecera ( `<head>` ) del documento web.

```
<head>
  <meta charset="utf-8" />
  <title> Primera Página Web </title>
  <style type = "texto/css" media = "screen">
    body { background: url(foo.gif) red; color: black; }
    p em { background-color: yellow; color: black; }
    .nota { margin-left: 5em; margin-right: 5em; }
  </style>
</head>
```

En el ejemplo anterior se puede ver cómo el elemento `<style>` también admite atributos como `type` y `media`.

Se aconseja utilizar un estilo incrustado cuando un único documento tiene un único estilo. Pero si los mismos estilos se utilizan en distintas páginas web, entonces sería más apropiado utilizar una hoja de estilos externa.



# 1. Hojas de estilos: CSS3

## Importar una hoja de estilos

Una hoja de estilos externa puede ser importada con la regla @import de CSS , que se puede especificar tanto al inicio de un archivo .css como al principio del elemento <style>.

```
<style type = "texto/css" media = "screen, projection" >  
  @import url(http://midominio.com/css/estils.css);  
  @import url(/css/estils2.css);  
  p { background-color: yellow; color: black; }  
</style>
```

## Estilos online

Para especificar estilos online se utiliza el atributo style, que toma por valor cualquier número de propiedades CSS separadas por ";".

```
<p style = "color: red; fuente-family: 'New Century Schoolbook', serif" >  
  Este párrafo muestra el texto rojo y con la fuente New Century Schoolbook, si está disponible.  
</p>
```

Los estilos online pierden muchas de las ventajas de las hojas de estilo al mezclar el contenido con la presentación. Además, los estilos online se aplican a todos los tipos dispositivos de salida (ordenadores, tabletas, móviles...), lo que provoca que se descuadre un diseño adaptativo. Así pues, es por estos motivos que deberían usarse con moderación.

# 1. Hojas de estilos: CSS3

## Características de CSS: cascada y herencia

Dos de las características que hacen que las hojas de estilo tengan un amplio abanico de posibilidades son la cascada y la herencia. La cascada se refiere a la posible combinación de diferentes hojas de estilo, mientras que la herencia se refiere a la capacidad que tienen los elementos del documento HTML de heredar propiedades de sus elementos antecesores.



# 1. Hojas de estilos: CSS3

## Hojas de estilo en cascada

A la hora de visualizar una página web, el navegador debe interpretar los diferentes estilos definidos para cada elemento HTML . Estos pueden estar especificados en diferentes lugares:

- Estilos predeterminados del navegador.
- Estilos especificados por el usuario final.
- Estilos relacionados con el documento especificados por su autor. Estos se pueden definir en tres sitios:
  - En un archivo externo.
  - Al principio del documento, a través del elemento `<style>`.
  - En un elemento HTML específico, a través de su atributo `style`.

Así, existen tres fuentes principales de información de estilos formando una cascada, de tal modo que los estilos del usuario modifican o son prioritarios a los que defina por defecto el navegador, y los estilos del autor del documento serán prioritarios a los anteriores.

De la misma forma, si la definición de un estilo de un autor entra en conflicto con la definición de otro, se aplicará el último que esté definido. Es decir, un estilo definido a través del atributo `style` será prioritario respecto a los definidos a través del elemento `<style>`.



# 1. Hojas de estilos: CSS3

```
<!DOCTYPE html>
<head>
  <html lang = "es">
  <meta charset="utf-8" />
  <title> Primera Página Web </title>
  <style>
    p {
      color: red;
    }
  </style >
</head>
<body>
  <h1> Encabezamiento </h1>
  <p style = "color: blue"> Un párrafo </p>
  <p> Otro párrafo </p>
</body>
</html>
```



Características de CSS: cascada y herencia

Si se fija, el texto del primer párrafo es azul, tal como se indica en el atributo style, en lugar de rojo definido al estilo para los párrafos en el <head>de la página. Como veis, pues, el segundo párrafo ya aparece de color rojo.

# 1. Hojas de estilos: CSS3

## La herencia en las hojas de estilo

Dado que todos los elementos de una página HTML , con excepción del elemento raíz <html>, están contenidos en otro elemento, debe tenerse en cuenta que todo elemento hereda las propiedades de sus antecesores.

Sin embargo, deben tenerse en cuenta las siguientes consideraciones:

- No todas las propiedades se heredan. Esta característica se puede consultar en: [www.w3.org/TR/css-2010/#properties](http://www.w3.org/TR/css-2010/#properties) .
- Si se desea forzar la herencia de una propiedad de un elemento que por defecto no hereda se puede utilizar el valor inherit.
- Si se especifica un valor en una propiedad, este valor prevalecerá sobre el valor heredado.

# 1. Hojas de estilos: CSS3

```
<style>
body {
  color: gray;
}

p{
  font-size: 15px;
}

strong {
  font-style: italic;
  font-size: 20px;
}

div {
  border-style: solid;
  border-width: 1px;
  border-color: red;
  background-color: rgba(255, 0, 0, 0.2);
  margin: 5px;
  width: 300px;
  padding: 5px;
}
</style>
```

```
<body>
  <p> Un párrafo con un texto muy enfatizado </strong> </p>

  <div style = "border-top-style: dashed;">
    <p> Aquí tenemos un párrafo </p>
    <div style = "background-color:rgba(0,0,255,0.2);width:50%;" >
      <p> No se hereda la línea discontinua </p>
    </div>
  </div>
  <br/>
  <div style = "border-top-style: dashed;">
    <p> Aquí tenemos otro párrafo </p>
    <div style = "border-top-style: inherit;background-color:rgba(0,0,255,0.2);width:50%;" >
      <p> Se hereda la línea discontinua </p>
    </div>
  </div>
</body>
```

Un párrafo con un texto muy enfatizado

Aquí tenemos un párrafo

No se hereda la línea  
discontinua

Aquí tenemos otro párrafo

Se hereda la línea  
discontinua

Fíjate cómo el color del texto es gris, ya que el párrafo, al estar contenido dentro del elemento `<body>`, hereda sus estilos. Al mismo tiempo, el elemento `<strong>` hereda los estilos de los elementos `<p>` y `<body>`.

Fíjate también cómo en la primera caja azul que se encuentra dentro de la roja no hereda el atributo `border-top-style`, y como en la segunda caja azul se ha forzado la herencia de la roja, que la contiene con el valor `inherit`.



# 1. Hojas de estilos: CSS3

## Formato de una regla CSS

Para definir un estilo específico, CSS utiliza una regla que consiste en un selector y un bloque donde se declaran las diferentes propiedades que debe tener el estilo.

```
selector { propiedad: valor; propiedad: valor; ... }
```

Donde selector se refiere al elemento que se aplica el estilo.

Donde las diferentes propiedades se declaran en el bloque separadas por ”;”.

# 1. Hojas de estilos: CSS3

```
h1 { fuente-size : 10px ; color : blue ; texto-align : center ; }
```

El estilo definido en el ejemplo anterior establece que las cabeceras h1 tendrán el siguiente aspecto:

- El tamaño de la fuente será de 10 píxeles.
- El color de la fuente será azul.
- El texto se verá centrado.

# 1. Hojas de estilos: CSS3

## Selectores

La función de los selectores es indicar en qué elementos se aplicarán los estilos definidos. Los diferentes tipo de selectores se presentan en la siguiente tabla.

Sintaxis	Descripción
*	Selector universal. Permite aplicar un estilo a cualquier tipo.
nombreElemento	Selector de tipos. Permite aplicar un estilo a un elemento <u>HTML</u> concreto.
.nombreClase	Selector de clase. Permite aplicar un estilo a los elementos que utilicen esta clase.
#idNombre	Selector ID. Permite aplicar un estilo a un elemento que tenga un código de identificación único.
selector[atributo]	Selector de atributo. Permite aplicar un estilo a un elemento que utilice un atributo específico.

## Selectores W3C

Puede consultar los diferentes tipos de selectores en: [goo.gl/ 9mY7Fw](https://goo.gl/9mY7Fw) .



# 1. Hojas de estilos: CSS3

Así, por ejemplo, se pueden tener los siguientes estilos definidos en una hoja de estilos:

```
<style>
* {
  font-size : 15px ;
  font-family : Arial ;
}

h1.resaltado {
  font-weight : bolder ;
  font-size : 25px ;
  color : white ;
  background-color: #ff80c0 ;
}

p , h2 {
  text-align : center ;
}

.textoAzul {
  color : blue ;
}

#estilo1 {
  font-size : larger
}

a [ target ] {
  background-color : yellow ;
}
</style>
```

```
<body>
<h1 class = "resaltado" > Título resaltado </h1>

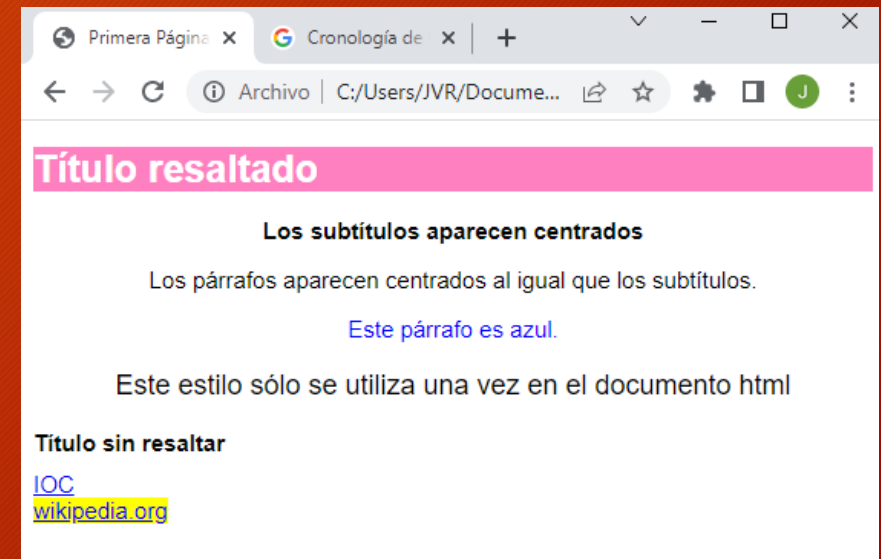
<h2> Los subtítulos aparecen centrados </h2>

<p> Los párrafos aparecen centrados al igual que los subtítulos. </p>

<p class = "textoAzul" > Este párrafo es azul. </p>

<p id = "estilo1" > Este estilo sólo se utiliza una vez en el documento html </p>

<h1> Título sin resaltar </h1>
<a href = "http://ioc.xtex.cat" > IOC </a>
<br/>
<a href = "http://www.wikipedia.org" target = "_top" > wikipedia.org </a>
```



# 1. Hojas de estilos: CSS3

Algunas consideraciones del ejemplo anterior son:

- El atributo class se utiliza para aplicar un estilo de clase al elemento.
- El atributo id se utiliza para aplicar un estilo id único para un elemento.
- Por defecto, el texto de todos los elementos será Arial de 15 px.
- Los links que tengan el atributo target aparecen resaltados.
- Si se desea definir el mismo estilo para diferentes selectores se pueden agrupar separados por una coma.

# 1. Hojas de estilos: CSS3

En la tabla se pueden consultar los diferentes tipos de selectores de atributo, ya que en éstos se pueden especificar qué valores deberían tener.

Sintaxis	Descripción
[atributo]	Selecciona el elemento con este atributo.
[atributo="valor"]	El valor del atributo es el valor.
[atributo^="texto"]	El valor del atributo comienza por texto.
[atributo\$="texto"]	El valor del atributo termina con texto.
[atributo*="text"]	El valor del atributo contiene el texto.
[atributo~="valor"]	El valor del atributo es una lista de palabras separadas por espacios, una de las cuales es exactamente valor.
[atributo =“valor”]	El valor del atributo es una lista de palabras separadas por guiones, comenzando por valor.



# 1. Hojas de estilos: CSS3

## Combinaciones de selectores

Los selectores se pueden combinar para añadir especificaciones a las reglas CSS . En la tabla se pueden ver algunas de las posibles combinaciones de los selectores.

Sintaxis	Descripción
<b>A+B</b>	Selectores adyacentes. Permite aplicar un estilo al elemento que se encuentre a continuación (B) de un elemento específico (A).
<b>A ~ B</b>	Selectores general de hermanos. Permite aplicar un estilo al elemento (B) que sea hermano del elemento (A).
<b>A &gt; B</b>	Selectores de hijos. Permite aplicar un estilo en el primer elemento hijo (B) de un elemento específico (A).
<b>AB</b>	Selectores descendentes. Permite aplicar un estilo a un determinado elemento que se encuentre dentro de un elemento específico.

# 1. Hojas de estilos: CSS3

```
<style>
/* selector general de hermanos */
h1 ~ h2 {
  color : red ;
}
/* selector de hermanos adyacentes */
h1 + h2 {
  color : blue ;
}
/* selector de descendientes*/
div h2 {
  color : green ;
}
/*selector de hijos*/
div > p {
  color : orange ;
  font-weight : bold ;
}
</style>
```

```
<body>
  <h1> Título 1 </h1>
  <h2> Subtítulo 1 es azul </h2>
  <p> El subtítulo 1 está definido inmediatamente después del título1 y también es hermano. </p>
  <hr/>
  <h1> Título 2 </h1>
  <p> Aquí tenemos un párrafo cualquiera. </p>
  <h2> Subtítulo 2 es rojo </h2>
  <p> El subtítulo 2 es hermano del título 2 pero no es consecutivo. </p>
  <hr/>
  <h1> Título 3 </h1>
  <div>
    <h2> Subtítulo 3 es verde </h2>
    <p> El subtítulo 3 NO es hermano del título 3 anterior. Y ese párrafo es naranja. </p>
    <p> Este párrafo también es hijo de la etiqueta < div > y es naranja. </p>
    <span><h2> Subtítulo 3.1 es verde </h2></span>
    <spa ><p> El subtítulo 3.1 también es descendente. </p></span>
    <span><p> Este párrafo no es hijo de la etiqueta <div> y no se ve naranja. </p></span>
  </div>
  <h2> Subtítulo 4 es rojo </h2>
  <p> El subtítulo 4 es hermano del título 3 pero no es consecutivo. </p>
</body>
```

# 1. Hojas de estilos: CSS3

Fíjate en los siguientes puntos:

- El primer subtítulo h2 es hermano adyacente de h1(inmediatamente después del título h1) y, por tanto, se ve de color azul en lugar de rojo. Sin embargo, el subtítulo 2 se ve rojo porque no está definido inmediatamente después del título h1.
- Los subtítulos h2 descendientes de h1 aparecen de color verde, tal y como se ve en el subtítulo 3 y 3.1.
- Los párrafos que sean hijos directos del elemento div se ven de color naranja.





# 1. Hojas de estilos: CSS3

## Pseudoclasses

Las pseudoclasses se añaden al selector de forma que se pueda aplicar un formato determinado al elemento seleccionado.

La forma de definir estilos utilizando pseudoclasses es:

```
selector :pseudoclase {  
    propiedad : valor ;  
    ...  
}
```

En la tabla se especifican las pseudoclasses para los enlaces.

Pseudoclase	Descripción
:link	Permite definir el estilo de los enlaces cuando todavía no se han visitado.
:visited	Permite definir el estilo de los vínculos visitados.
:active	Permite definir el estilo de los elementos cuando se activan (cuando se pulsa el ratón sobre él).
:hover	Permite definir el estilo de los elementos cuando se pasa el ratón sobre él).
:focus	Permite definir el estilo de los elementos al recibir el foco.

# 1. Hojas de estilos: CSS3

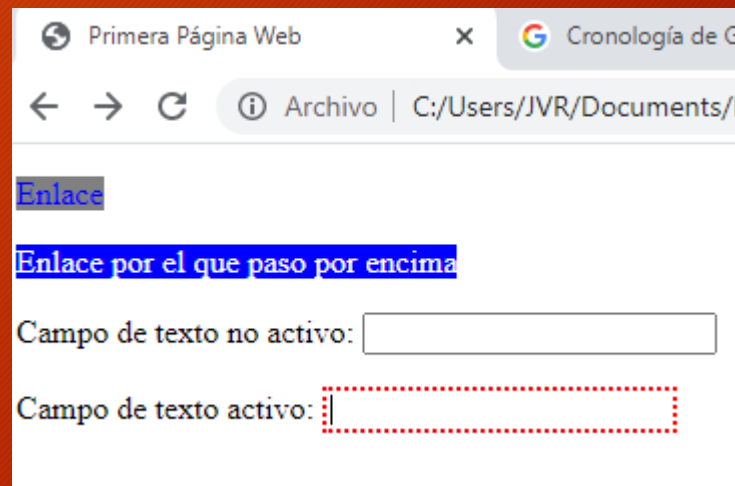
```
<style>
/* Los enlaces (visitados y no visitados)
son de color azul con el fondo gris y sin subrayar. */
a:link ,
a:visited {
    color : blue ;
    background-color : gray ;
    Text-decoration : none ;
}
* Cuando se pasa por encima tendrán el fondo azul y texto blanco */
a:hover {
    color : white ;
    background-color : blue ;
}
* Cuando se hace clic antes de soltar se harán mayores */
a:active {
    font-size : larger ;
    color : white ;
    background-color : blue ;
}
* Cuando se sitúa en un control de formulario <input> tendrá
borde rojo punteado, y un relleno de 2px */
input:focus{
    outline: none;
    border: red 2px dotted;
    padding : 2px;
}
</style>
```

```
<body>
<p><a href = "http://fsf.org" > Enlace </a></p>

<p><a href = "http://gnu.org" > Enlace por el que paso por encima </a></p>

<p> Campo de texto no activo: <input type = "text" name = "input" /></p>

<p> Campo de texto activo: <input type = "text" name = "input" /></p>
</body>
```



# 1. Hojas de estilos: CSS3

Existen otras pseudoclases que aplican los estilos haciendo referencia a la estructura del DOM.

Pseudoclase	Descripción
:root	Representa un elemento que se encuentra en la raíz del documento.
:nth-child (num)	Selecciona el elemento indicado pero con tal que sea el hijo enésimo de su padre.
:nth-last-child (num)	Idéntico al anterior, pero el número indicado empieza a contarse desde el último hijo.
:nth-of-type (num)	Selecciona el elemento indicado pero con tal de que sea el enésimo elemento hijo de este tipo.
:nth-last-of-type (num)	Idéntico a lo anterior, pero el número indicado se empieza a contar desde el último hijo.
:first-child	Selecciona un elemento que sea el primer hijo de otro.
:last-child	Selecciona el elemento indicado, pero con tal de que sean el último hijo de su elemento padre.
:first-of-type	Se refiere al primer elemento de este tipo en el elemento padre.
:last-of-type	Se refiere al último elemento de este tipo en el elemento padre.
:only-of-type	Hace referencia a ese elemento que es el único hijo de su padre.
:empty	Selecciona el elemento indicado pero con tal que no tenga ningún hijo y tampoco puede tener ningún contenido de texto.

Otras pseudoclases

Pseudoclase	Descripción
:lang	De idioma. Permite aplicar estilos en función del estilo especificado.
:not	De negación. Selecciona todos los elementos que no cumplen con la condición de un selector.

Puede consultar las pseudoclases que define CSS en:  
[developer.mozilla.org/es/docs/Web/CSS/Pseudo-classes](https://developer.mozilla.org/es/docs/Web/CSS/Pseudo-classes)



# 1. Hojas de estilos: CSS3

```
<style>
/*Selecciona los elementos descendientes de las listas de tipo especial */
ul.especial li {
  display: inline;
  padding: 5px;
  border-top-left-radius: 8px;
  border-bottom-right-radius: 8px;
}
/* selecciona al primer hijo de la lista*/
ul.especial li:first-child {
  color : blue ;
}
/*selecciona los elementos impares de la lista */
ul.especial li:nth-child(odd) {
  background-color : grey ;
}
/*selecciona los elementos pares de la lista */
ul.especial li:nth-child(even) {
  background-color : pink ;
}
/*selecciona el cuarto hijo de la lista*/
ul.especial li:nth-of-type(4) {
  color : white ;
}
/*selecciona los párrafos que no tienen hermanos párrafos*/
p:only-of-type {
  background-color : grey ;
  color : #ffa0ff ;
  font-weight : bold ;
  font-size : 25px ;
}
/*Los párrafos en inglés aparecen en itálico */
p:lang(en) {
  font-style : italic;
}
:root {
  font-family : arial ;
}
</style>
```

```
<body>
<div>
  <p> Menú especial: </p>
  <ul class = "especial" >
    <li> Elemento 1 </li>
    <li> Elemento 2 </li>
    <li> Elemento 3 </li>
    <li> Elemento 4 </li>
  </ul>
  <div>
    <p> Un párrafo </p>
    <p lang="en"> This paragraph is in english. </p>
  </div>
</div>
</body>
```

## Menú especial:

Elemento 1 Elemento 2 Elemento 3 Elemento 4

Un párrafo

*This paragraph is in english.*

# 1. Hojas de estilos: CSS3

## Pseudoelementos

A diferencia de las pseudoclases, los pseudoelementos no describen el estado de un elemento, sino que se añaden a un selector para definir estilos en una parte concreta del documento HTML .

Los pseudoelementos suelen ir precedidos por "::", pero por compatibilidad con versiones anteriores también pueden ir precedidos por ":".

La forma de definir estilos utilizando pseudoelementos es:

```
selector : :pseudoElement {  
    propiedad : valor ;  
    ...  
}
```

# 1. Hojas de estilos: CSS3

En la tabla podemos ver la descripción de algunos de ellos.

Sintaxis	Descripción
::after	Permite introducir contenido al final del elemento. Es necesario añadir la propiedad <code>content</code> con el valor deseado.
::before	Permite introducir contenido al inicio del elemento. Es necesario añadir la propiedad <code>content</code> con el valor deseado.
::first-letter	Permite definir el estilo de la primera letra del elemento.
::first-line	Permite definir el estilo de la primera línea del elemento.
::selection	Permite definir el estilo de lo que el usuario ha seleccionado con el ratón.

Puedes consultar los pseudoelementos que define CSS en:  
[developer.mozilla.org/es/docs/Web/CSS/Pseudoelementos](https://developer.mozilla.org/es/docs/Web/CSS/Pseudoelementos)



# 1. Hojas de estilos: CSS3

```
<style>
div::first-line { color : red ; }
h1::before { content : "Título" ; }
p::after { content : "." ; }
p::first-letter { text-transform : uppercase ; }
</style>
```

```
<body>
<h1> 1 </h1>
<div>
  <p>este es el primer párrafo largo y es rojo</p>
  <p>este es el segundo párrafo y no es rojo</p>
</div>
</body>
```



# 1. Hojas de estilos: CSS3

## Unidades de medida CSS

Algunas propiedades CSS indican el tamaño de la letra, el ancho, los márgenes, etc. CSS clasifica las unidades de medida en dos tipos: absolutas y relativas .

Unidad	Descripción
<b>cm</b>	Centímetros
<b>mm</b>	Milímetros
<b>in</b>	Pulgadas, <i>inches</i> . Equivale a 2,54 cm
<b>pt</b>	Puntos. Medida que equivale a $1/72$ de una pulgada
<b>pc</b>	Picas. Medida que equivale a 12 pt

Unidades de medida absolutas

Unidad	Descripción
<b>em</b>	Medida relativa al tamaño del tipo de letra de su contenedor.
<b>rem</b>	<i>Root em</i> . Medida relativa al tamaño del tipo de letra general. Tiene como referencia la unidad que utiliza el elemento raíz ( <i>root</i> ), y no el elemento contenedor.
<b>px</b>	Píxeles. Medida relativa a la resolución de la pantalla.
<b>ex</b>	Medida relativa a la altura de la letra x. Cambia si se cambia la fuente que utilizamos.
<b>ch</b>	Medida relativa a la altura del carácter numérico "o". Cambia si se cambia la fuente que utilizamos.
<b>%</b>	Porcentaje.

Unidades de medida relativas

Unidad	Descripción
<b>vw</b>	<i>Viewport width</i> . 1 vw equivale a $(1/100)$ de la anchura de la ventana, es decir, al 1%. Para reflejar un valor del 15% deberíamos poner 15 vw.
<b>vh</b>	<i>Viewport height</i> . 1 vh equivale a $(1/100)$ de la altura de la ventana, es decir, al 1%. Para reflejar un valor del 15% deberíamos poner 15 vw.
<b>vmin</b>	Funcionamiento similar a los anteriores, pero existe una evaluación previa de qué eje, x o y (anchura, altura), es menor, y selecciona como referente el eje que lo sea.
<b>vmax</b>	Idéntico al anterior, pero cogiendo como referente el eje que sea mayor de los dos.

Unidades de medida relativas al viewport

# 1. Hojas de estilos: CSS3

```
<style>
.medida1 { font-size : 16pt ; }
.medida2 { font-size : 16px ; }
.medida3 { font-size : 1.5em ; }
.medida4 { font-size : 1.5rem ; }
.medida5 { font-size : 1.5vw ; }
</style>
```

```
<body>
  <p class = "medida1"> Tamaño del texto en puntos (absoluta) </p>
  <p class = "medida2" > Tamaño del texto en píxeles (relativo a la resolución) </p>
  <p class = "medida3" > Tamaño del texto en em (relativo al contenedor) </p>
  <p class = "medida4" > párrafo tamaño en rem (relativo al elemento root) </p>
  <p class = "medida5" > Tamaño del texto en vw (relativa en el viewport) </p>
<hr/>
<div class = "medida3">
  <p class = "medida1" > Tamaño del texto en puntos (absoluta) </p>
  <p class = "medida2" > Tamaño del texto en píxeles (relativo a la resolución) </p>
  <p class = "medida3" > Tamaño del texto en em (relativo al contenedor) </p>
  <p class = "medida4" > Tamaño del texto en rem (relativo al elemento root) </p>
  <p class = "medida5" > Tamaño del texto en vw (relativa en el viewport) </p>
</div>
</body>
```

Tamaño del texto en puntos (absoluta)

Tamaño del texto en píxeles (relativo a la resolución)

Tamaño del texto en em (relativo al contenedor)

párrafo tamaño en rem (relativo al elemento root)

Tamaño del texto en vw (relativa en el viewport)

Tamaño del texto en puntos (absoluta)

Tamaño del texto en píxeles (relativo a la resolución)

Tamaño del texto en em (relativo al contenedor)

Tamaño del texto en rem (relativo al elemento root)

Tamaño del texto en vw (relativa en el viewport)

Aunque a menudo se utiliza la unidad de medida píxel, una buena opción es “rem”, ya que especifica los tamaños relativos al elemento root , a diferencia de “em”, cuyo tamaño puede costar controlar porque puede variar en función de dónde se encuentre.

También puede comprobar que si modificamos el tamaño de la ventana del navegador, el texto expresado vw se redimensiona.



# 1. Hojas de estilos: CSS3

## Propiedades CSS

Existen múltiples atributos o propiedades que se pueden definir con CSS . Para conocer el amplio abanico que ofrece CSS se pueden consultar las siguientes webs:

- Developer Mozilla: [developer.mozilla.org/es/docs/Web/HTML/Attributes](https://developer.mozilla.org/es/docs/Web/HTML/Attributes)
- W3C : [www.w3.org/community/webed/wiki/CSS/Properties](http://www.w3.org/community/webed/wiki/CSS/Properties)

Dado que el estándar CSS3 está en continua evolución, no todos los navegadores han adoptado algunas de sus propiedades, o si lo han hecho, muchas veces necesitan un prefijo. Estos prefijos se utilizarán cuando así sea necesario y pueden verse en la tabla.

Prefijo	Navegador
-moz-	Prefijo para el navegador Firefox.
-ms-	Prefijo para el navegador Internet Explorer.
-webkit-	Prefijo para los navegadores Chrome y Safari.
-o-	Prefijo para el navegador Opera.

# 1. Hojas de estilos: CSS3

Sin embargo, cabe mencionar que no todos los navegadores se comportan de la misma manera frente a la misma hoja de estilo, y esto se debe a que algunos no cumplen los estándares establecidos. Por eso se puede recurrir a páginas como [caniuse](#) para ver si determinadas características ya han sido implementadas, y por qué navegadores.

## 'Shorthand properties'

Las propiedades abreviadas permiten establecer los valores de varias propiedades CSS simultáneamente. Estas propiedades pueden hacer referencia a los márgenes, bordes, rellenos y también al fondo del documento web.

Propiedades	
<b>Fuente</b>	color, font-size, font-family, font-weight, font-style
<b>Párrafos</b>	line-height, text-decoration, text-align, text-indent, text-transform, text-shadow, text-overflow, text-wrap, list-style
<b>Fondo</b>	background-color, background, background-image, background-origin, background-repeat, background-position, background-attachment
<b>Tablas</b>	border-spacing, border-collapse, caption-side, empty-cells

# 1. Hojas de estilos: CSS3

```
<style>
/*Gradiente de 180 grados que va de azul a blanco*/
.gradiente1 {
  background : linear-gradient( 180deg , blue , white ) ;
}
/*Gradiente de 90 grados que va de granate a naranja a naranja más claro, la cantidad de cada c
.gradiente2 {
  background : linear-gradient( 90deg , #660000 10% , #f80 30% , #ffc 60% ) ;
}
/*Gradiente radial circular que va de negro a blanco */
.gradiente3 {
  background : radial-gradient( circle , black , white ) ;
}
/*Gradiente elíptico que va de negro a blanco*/
.gradiente4 {
  background : radial-gradient( ellipse , black , white ) ;
}
/* Los párrafos aparecen indentados y tienen una sombra de un píxel, separada 2px y gris */
p {
  text-indent : 50px ;
  text-shadow : 1px 2px #ccc ;
  padding : 50px ;
}
/* Los títulos tienen un relleno de 5px*/
h1 {
  padding : 5px ;
}
</style>
```

```
<body>
  <h1 class = "gradiente2" > Título con un gradiente lineal vertical. </h1>
  <h1 class = "gradiente1" > Título con un gradiente lineal horizontal. </h1>

  <p class = "gradiente3" > Este párrafo tiene un gradiente radial circular </p>

  <p class = "gradiente4" > Este párrafo tiene un gradiente radial elíptico </p>
  <p> Otro párrafo </p>
</body>
```



# 1. Hojas de estilos: CSS3

Además, debe tenerse en cuenta que a pesar de que los navegadores presentan algunas configuraciones similares, como el tipo de letra serif de color negro, por ejemplo. Difieren bastante en los valores de márgenes verticales de los encabezados (margin-top y margin-bottom), la tabulación izquierda de los elementos de una lista (margin-left o padding-left) o bien la interlineado (line-height).

**Título con un gradiente lineal vertical.**

**Título con un gradiente lineal horizontal.**

Este párrafo tiene un gradiente radial circular

Este párrafo tiene un gradiente radial elíptico

Otro párrafo

# 1. Hojas de estilos: CSS3

## El modelo caja

Los navegadores, de forma automática, crean y colocan cada elemento HTML en una estructura que tiene formato de caja. Así, por defecto, la caja de los elementos HTML de tipo block ocupa todo el ancho de la ventana, y la de los elementos inline sólo toman la anchura necesaria.

En principio, la caja de cada elemento no es visible, puesto que no muestra ningún color de fondo ni borde. Sin embargo, se puede modificar con CSS para cambiar sus dimensiones, colores, borde y posición.

Se pueden modificar las propiedades de la caja de cualquier elemento HTML , pero generalmente se utiliza el elemento `<div>` para organizar y dar estructura al diseño de las páginas web.

### Elemento "`<div>`"

Este elemento crea una caja que por defecto ocupa todo el ancho del navegador. Se utiliza para definir el estilo de una sección de una página web.

### Elemento "`<span>`"

Este elemento crea una caja que por defecto sólo ocupa el ancho necesario. Se utiliza para definir el estilo de una porción de contenido de una sección de una página web.

# 1. Hojas de estilos: CSS3

## Dimensiones de la caja

Las partes que componen una caja en su orden de visualización desde el punto de vista del usuario son:

- Contenido , content : se refiere al contenido del elemento; por ejemplo, el texto, una imagen, etc.
- Relleno, padding : espacio libre opcional existente entre el contenido y el borde.
- Borde , border : línea que envuelve el contenido y el relleno.
- Imagen de fondo , background image : imagen que se muestra por detrás del contenido y el relleno.
- Color de fondo , background color : color que se muestra detrás del contenido y el relleno.
- Margen , margin: separación opcional existente entre la caja y el resto de cajas adyacentes.

El relleno y el margen que define el modelo de caja son transparentes, por tanto, se visualiza la imagen o el color de fondo que se haya definido.

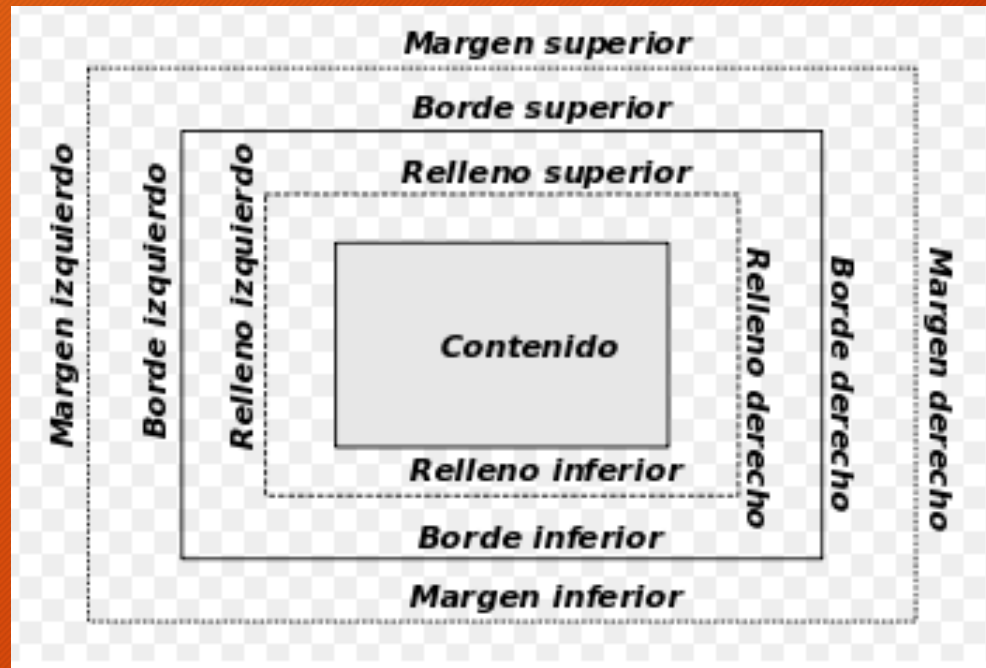
## Imagen y color de fondo del modelo de caja

Si en una caja se definen el color y la imagen de fondo, la imagen tiene mayor prioridad y se visualiza. Pero si la imagen no cubre por completo la caja del elemento también se visualiza el color de fondo.



# 1. Hojas de estilos: CSS3

Así pues, cada caja tiene un área de contenido, y opcionalmente le rodean unas áreas de relleno, borde y margen



# 1. Hojas de estilos: CSS3

## Propiedades para la caja

Propiedad	Propiedades
<b>Margen de la caja</b>	<code>margin-left, margin-right, margin-top, margin-bottom, margin</code>
<b>Fondo</b>	<code>background-color, background-image</code>
<b>Ancho del borde</b>	<code>border-top-width, border-right-width, border-bottom-width, border-left-width, border-width</code>
<b>Color del borde</b>	<code>border-top-color, border-right-color, border-bottom-color, border-left-color, border-color</code>
<b>Estilo del borde</b>	<code>border-top-style, border-right-style, border-bottom-style, border-left-style, border-style</code>
<b>Relleno de la caja</b>	<code>padding-left, padding-right, padding-top, padding-bottom, padding</code>

La propiedad `box-sizing` se utiliza para modificar la altura y el ancho de la caja de los elementos HTML , ya que por defecto se calcula sin tener en cuenta el relleno y el ancho del borde.

Así pues, la propiedad `box-sizing` puede tomarse por valor `content-box`, que es el valor por defecto de las cajas, y no se tiene en cuenta el tamaño del relleno y del borde para calcular el tamaño de la caja. O bien el valor `border-box`, donde sí se tiene en cuenta el tamaño del relleno y del borde para calcular el tamaño de la caja.

# 1. Hojas de estilos: CSS3

## Propiedades para la caja

Propiedad	Propiedades
<b>Margen de la caja</b>	<code>margin-left, margin-right, margin-top, margin-bottom, margin</code>
<b>Fondo</b>	<code>background-color, background-image</code>
<b>Ancho del borde</b>	<code>border-top-width, border-right-width, border-bottom-width, border-left-width, border-width</code>
<b>Color del borde</b>	<code>border-top-color, border-right-color, border-bottom-color, border-left-color, border-color</code>
<b>Estilo del borde</b>	<code>border-top-style, border-right-style, border-bottom-style, border-left-style, border-style</code>
<b>Relleno de la caja</b>	<code>padding-left, padding-right, padding-top, padding-bottom, padding</code>

La propiedad `box-sizing` se utiliza para modificar la altura y el ancho de la caja de los elementos HTML , ya que por defecto se calcula sin tener en cuenta el relleno y el ancho del borde.

Así pues, la propiedad `box-sizing` puede tomarse por valor `content-box`, que es el valor por defecto de las cajas, y no se tiene en cuenta el tamaño del relleno y del borde para calcular el tamaño de la caja. O bien el valor `border-box`, donde sí se tiene en cuenta el tamaño del relleno y del borde para calcular el tamaño de la caja.



# 1. Hojas de estilos: CSS3

```
<style>
.div1 {
  width : 300px ;
  height : 100px ;
  border : 1px solid blue ;
}
.div2 {
  width : 300px ;
  height : 100px ;
  padding : 50px ;
  border : 1px solid red ;
}
.div3 {
  width : 300px ;
  height : 100px ;
  border : 1px solid blue ;
  box-sizing : Border-box ;
}
.div4 {
  width : 300px ;
  height : 100px ;
  padding : 50px ;
  border : 1px solid red ;
  box-sizing : Border-box ;
}
</style>
```

```
<body>
  <h2> Cajas sin box-sizing </h2>
  <div class = "div1" >
    <p> Esta caja es más pequeña </p>
    <p> (ancho 300px y altura 100px). </p>
  </div>
  <br>
  <div class = "div2">
    <p> Esta caja es mayor </p>
    <p> (ancho 300px y altura 100px). </p>
  </div>

  <h2> Cajas con box-sizing </h2>
  <div class = "div3">
    <p> Las dos cajas tienen el mismo tamaño </p>
    <p> (ancho 300px y altura 100px). </p>
  </div>
  <br>
  <div class = "div4" > Fantástico!! </div>
</body>
```

## Cajas sin box-sizing

Esta caja es más pequeña  
(ancho 300px y altura 100px).

Esta caja es mayor  
(ancho 300px y altura 100px).

## Cajas con box-sizing

Las dos cajas tienen el mismo tamaño  
(ancho 300px y altura 100px).

Fantástico!!

# 1. Hojas de estilos: CSS3

## Posicionamiento

Los navegadores crean y posicionan de forma automática todas las cajas de los elementos que forman cada página HTML siguiendo el flujo normal de ésta. Sin embargo, CSS permite modificar la posición en la que se muestra la caja, para así poder conseguir un diseño determinado.

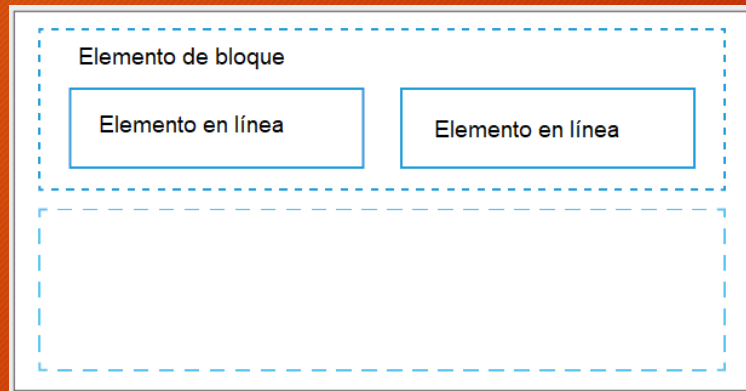
Para indicar los diferentes tipos de posicionamiento de la caja se utiliza la propiedad CSS `position` , y los valores que puede tomar son `static`, `relative`, `absolute` o `fixed`.

# 1. Hojas de estilos: CSS3

## Posicionamiento estatico o normal

El posicionamiento de las cajas es estático si no se indica lo contrario. En este modelo sólo se tiene en cuenta si el elemento es en bloque o en línea, las propiedades width y height y su contenido.

En el modelo de caja, la distancia entre dos cajas se controla mediante márgenes laterales.



Así pues, vea que por defecto los elementos en bloque se muestran uno debajo del otro, y que los elementos en línea se muestran uno detrás del otro. Y si un elemento se encuentra dentro de otro, el elemento padre se llama elemento contenedor y determina tanto la posición como el tamaño de las cajas de su interior.



# 1. Hojas de estilos: CSS3

## Posicionamiento relativo

Modelo que consiste en posicionar una caja con su posicionamiento normal y después desplazarla.

El desplazamiento de la caja se controla con las propiedades top, bottom, right y left. Así pues, el valor de estas propiedades se utiliza para mover las cajas de forma descendente, ascendente, hacia la izquierda y hacia la derecha, respectivamente. Si se utilizan valores negativos su efecto será el inverso. Fíjate, pues, que este comportamiento es poco intuitivo y puede causar errores si no se está acostumbrado.

```
<style>
div {
  position : relative ;
  width : 250px ;
  height : 180px ;
  border : 1px solid black ;
  margin : 2px ;
  box-sizing : Border-box ;
  padding : 2px ;
  background-color : rgba ( 255 , 0 , 0 , 0.2 ) ;
}

.cajaInline {
  height : 20px ;
  display : inline ;
}

.cajaBlock {
  width : 142px ;
  height : 20px ;
  margin-top : 8px ;
}

.cajaRel {
  position : relative ;
  top : 10px ;
  left : 5px ;
  width : 50px ;
  height : 20px ;
  display : inline ;
}
```

```
.cajaAbs {
  position : absolute ;
  top : 70px ;
  left : 100px ;
  width : 50px ;
  height : 20px ;
  display : inline ;
}

.cajaFija {
  position : fixed ;
  top : 180px ;
  left : 100px ;
  width : 50px ;
  height : 20px ;
  display : inline ;
}

.cajaFlotanteDerecha {
  float : right ;
  width : 50px ;
  height : 20px ;
}

.cajaFlotanteIzquierda {
  float : left ;
  width : 50px ;
  height : 20px ;
}

body {
  font-family : arial ;
  font-size : 0.8rem ;
}
</style>
```

# 1. Hojas de estilos: CSS3

```
<body>
  <div>
    <p> Caja contenedora: las cajas están en posición estática. </p>
    <div class = "cajaInline" > caja1 </div>
    <div class = "cajaInline" > caja2 </div>
    <div class = "cajaInline" > caja3 </div>
    <div class = "cajaBlock" > caja4 </div>
  </div>
  <div>
    <p> Caja contenedora: las cajas están en posición estática, excepto la caja2, que está en posición relativa. </p>
    <div class = "cajaInline"> caja1 </div>
    <div class = "cajaRel" > caja2 </div>
    <div class = "cajaInline" > caja3 </div>
    <div class = "cajaBlock" > caja4 </div>
  </div>
</body>
```

Caja contenedora: las cajas están en posición estática.

caja1 caja2 caja3  
caja4

Caja contenedora: las cajas están en posición estática, excepto la caja2, que está en posición relativa.

caja1 caja2 caja3  
caja4

El desplazamiento relativo de una caja no afecta al resto de cajas adyacentes, las cuales se muestran en la misma posición.



# 1. Hojas de estilos: CSS3

## Posicionamiento absoluto

En este modelo, la posición de la caja se establece de forma absoluta respecto a su elemento contenedor. El elemento contenedor de referencia será el que esté posicionado de cualquier forma diferente de `position:static`.

Si no hay ningún elemento que pueda hacer de referencia, entonces lo será la ventana del navegador, que no debe confundirse con el elemento `<body>`.

La nueva posición de la caja se indica mediante las propiedades `top`, `right`, `bottom` y `left`, de tal modo que:

- El valor de la propiedad `top` indica el desplazamiento desde el borde superior de la caja hasta el borde superior del elemento contenedor de referencia.
- El valor de la propiedad `right` indica el desplazamiento desde el borde derecho de la caja hasta el borde derecho del elemento contenedor de referencia.
- El valor de la propiedad `bottom` indica el desplazamiento desde el borde inferior de la caja hasta el borde inferior del elemento contenedor de referencia.
- El valor de la propiedad `left` indica el desplazamiento desde el borde izquierdo de la caja hasta el borde izquierdo del elemento contenedor de referencia.

Las cajas posicionadas de forma absoluta salen del flujo normal de la página , lo que provoca que el resto de elementos se muevan.



# 1. Hojas de estilos: CSS3

```
<body>
  <div>
    <p> Caja contenedora: las cajas están en posición estática. </p>
    <div class = "cajaInline"> caja1 </div>
    <div class = "cajaInline"> caja2 </div>
    <div class = "cajaInline"> caja3 </div>
    <div class = "cajaBlock"> caja4 </div>
  </div>
  <div>
    <p> Caja contenedora: las cajas están en posición estática, excepto la caja2 que está en posición absoluta. </p>
    <div class = "cajaInline"> caja1 </div>
    <div class = "cajaAbs"> caja2 </div>
    <div class = "cajaInline"> caja3 </div>
    <div class = "cajaBlock"> caja4 </div>
  </div>
</body>
```

La caja 2 está posicionada de manera absoluta, lo que provoca que el resto de elementos de la página modifiquen su posición y ocupen su sitio

Caja contenedora: las cajas están en posición estática.

caja1 caja2 caja3  
caja4

Caja contenedora: las cajas están en posición estática, excepto la caja2 que está en posición absoluta.

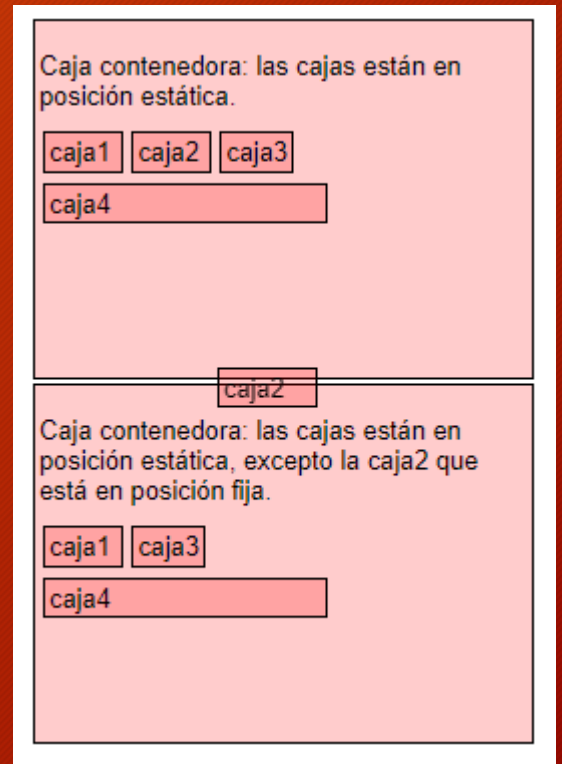
caja1 caja3 caja2  
caja4

# 1. Hojas de estilos: CSS3

## Posicionamiento fijo

En este modelo, la caja se convierte en un elemento inamovible, por lo que su posición en la pantalla siempre es la misma independientemente de los demás elementos y de si el usuario sube o baja la ventana del navegador. Es un caso particular de posicionamiento absoluto.

```
<body>
<div>
  <p> Caja contenedora: las cajas están en posición estática. </p>
  <div class = "cajaInline"> caja1 </div>
  <div class = "cajaInline"> caja2 </div>
  <div class = "cajaInline"> caja3 </div>
  <div class = "cajaBlock"> caja4 </div>
</div>
<div>
  <p> Caja contenedora: las cajas están en posición estática, excepto la caja2 que está en posición fija. </p>
  <div class = "cajaInline"> caja1 </div>
  <div class = "cajaFija"> caja2 </div>
  <div class = "cajaInline"> caja3 </div>
  <div class = "cajaBlock"> caja4 </div>
</div>
</body>
```



# 1. Hojas de estilos: CSS3

## Posicionamiento flotante

En este modelo se desplaza la caja todo lo posible a la derecha o a la izquierda de la línea en la que se encuentra, y entonces esta caja deja de formar parte del flujo normal de la página, por lo que el resto de cajas ocupan el su sitio.

La propiedad CSS que permite posicionar una caja de forma flotante se llama float, y puede tomarse por valores left, right, none y inherit.

```
<body>
<div>
  <p> Caja contenedora: las cajas están en posición estática. </p>
  <div class = "cajaBlock"> caja1 </div>
  <div class = "cajaBlock"> caja2 </div>
  <div class = "cajaBlock"> caja3 </div>
  <div class = "cajaBlock"> caja4 </div>
</div>
<div>
  <p> Caja contenedora: las cajas están en posición estática, excepto la caja2 que está en posición fija. </p>
  <div class = "cajaFlotanteDerecha"> caja1 </div>
  <div class = "cajaBlock"> caja2 </div>
  <div class = "cajaBlock"> caja3 </div>
  <div class = "cajaBlock"> caja4 </div>
</div>
</body>
```

Caja contenedora: las cajas están en posición estática.

caja1

caja2

caja3

caja4

Caja contenedora: las cajas están en posición estática, excepto la caja2 que está en posición fija.

caja2

caja3

caja4

caja1



# 1. Hojas de estilos: CSS3

Se pueden utilizar todos los tipos de posicionamiento, ya que en función de lo que queramos conseguir nos interesará posicionar la caja de un elemento utilizando un tipo u otro. Sin embargo, el más utilizado es el flotante.

Además, la propiedad CSS clear permite especificar si un elemento puede estar junto a los elementos flotantes que lo preceden o debe moverse hacia abajo por debajo de ellos. Esta propiedad puede aplicarse tanto a los elementos flotantes como no flotantes.

```
<!DOCTYPE html>
<html lang = "es">
<head>
  <meta charset="utf-8" />
  <title> Primera Página Web </title>
  <style>
    p {
      float: left;
      max-width:150px;
      min-height: 50px;
      background-color: lightblue;
      margin-left: 5px;
      padding: 20px 2px 2px 20px;
    }

    p.clear {
      clear: both;
      background-color: blue;
    }
  </style>
</head>
<body>
<div>
  <p> Texto del primer párrafo </p>
  <p> Texto del segundo párrafo </p>
  <p class = "clear" > Texto del párrafo que se desea apartar </p>
</div>
</body>
</html>
```

Texto del primer  
párrafo

Texto del segundo  
párrafo

Texto del párrafo que  
se desea apartar

# 1. Hojas de estilos: CSS3

## Visualización en el navegador

Cuando el navegador visualiza una página web crea una caja para cada elemento HTML , y ésta se puede modificar con CSS . Por tanto, los factores que tiene en cuenta a la hora de generar cada caja son:

- El tamaño de la caja, si es que se ha establecido a través de las propiedades width y height.
- El tipo de elemento HTML , es decir, block o inline .
- El posicionamiento de la caja.
- Las relaciones entre elementos (donde se encuentra cada elemento, elementos descendentes, etc.).
- Otro tipo de información, como el tamaño de las imágenes y de la ventana del navegador.

Además CSS define propiedades para controlar la visualización de las cajas. Estas propiedades son display, visibility, overflow y z-index.

Propiedad	Valores
display	inline, block, none, list-item, run-in, inline-block, table, inline-table, table-row-group, , table-header-group, table-footer-group, table-row, table-column-group, table-column, table-cell, table-caption, inherit
visibility	visible, hidden, collapse, inherit
overflow	visible, hidden, scroll, auto, inherit
z-index	auto, numero, inherit



# 1. Hojas de estilos: CSS3

Las propiedades `display` y `visibility` permiten esconder cualquier elemento de la página. La propiedad `display` permite esconder un elemento para que desaparezca de la página y así los otros elementos puedan ocupar su sitio. Por el contrario, `visibility` permite hacer invisible un elemento; el navegador crearía la caja, pero no la mostraría.

De hecho, la propiedad `display` ofrece muchas más posibilidades que esconder o mostrar un elemento, y en realidad modifica la forma en que se visualiza un elemento. Así pues, los valores más utilizados de esta propiedad son:

- `inline`: muestra el elemento como en línea, independientemente de si lo es o no.
- `block`: muestra el elemento como en bloque, independientemente de si lo es o no.
- `none`: oculta el elemento y hace que desaparezca de la página.



# 1. Hojas de estilos: CSS3

```
<div>
  <p> Caja2. display:none</p>
  <p> Caja7. visibility:hidden</p>
  <div class = "cajaFlotanteIzquierda" > Caixa1 </div>
  <div class = "cajaFlotanteIzquierda" style = "display:none" > Caixa2 </div>
  <div class = "cajaFlotanteIzquierda" > Caja3 </div>
  <div class = "cajaFlotanteIzquierda" > Caja4 </div>
  <div class = "cajaFlotanteIzquierda" > Caja4 </div>
  <div class = "cajaFlotanteIzquierda" > Caja6 </div>
  <div class = "cajaFlotanteIzquierda" style = "visibility:hidden" > Caixa7 </div>
  <div class = "cajaFlotanteIzquierda" > Caja8 </div>
</div>
```

Caja2. display:none

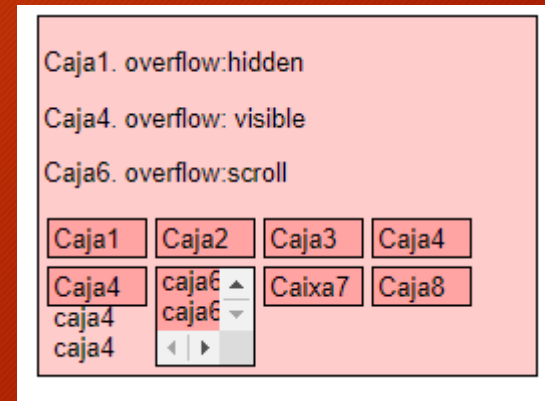
Caja7. visibility:hidden

Caixa1	Caja3	Caja4	Caja4
Caja6		Caja8	

# 1. Hojas de estilos: CSS3

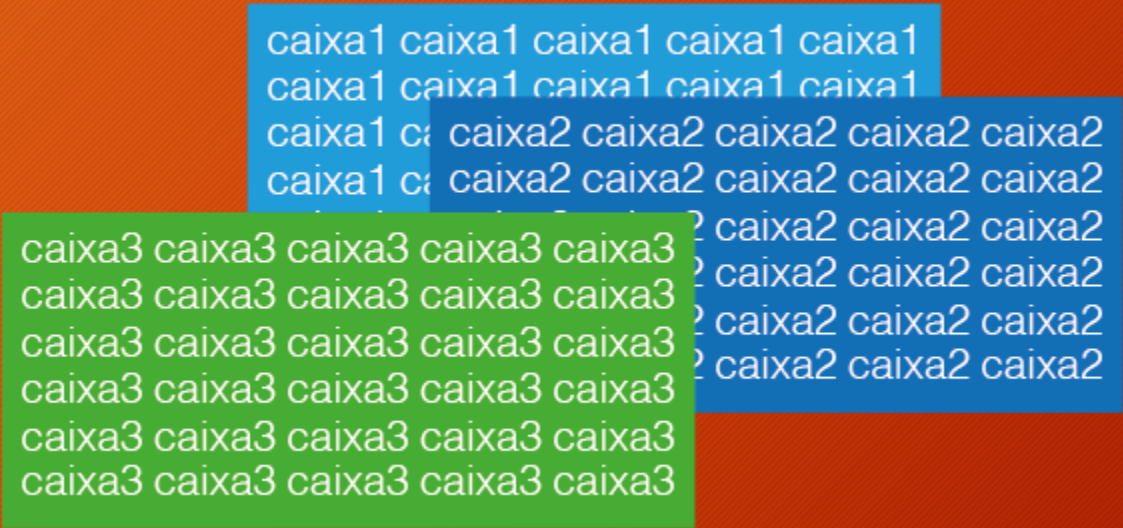
La propiedad `overflow` se utiliza para controlar la forma en que se visualizan los contenidos que sobresalen de la caja que tiene definida.

```
<body>
<div>
  <p> Caja1. overflow:hidden </p>
  <p> Caja4. overflow: visible </p>
  <p> Caja6. overflow:scroll </p>
  <div class = "cajaFlotanteIzquierda" style = "overflow:hidden" > Caja1 caja1 caja1 </div>
  <div class = "cajaFlotanteIzquierda" > Caja2 </div>
  <div class = "cajaFlotanteIzquierda" > Caja3 </div>
  <div class = "cajaFlotanteIzquierda" > Caja4 </div>
  <div class = "cajaFlotanteIzquierda" style = "overflow:visible" > Caja4 caja4 caja4 </div>
  <div class = "cajaFlotanteIzquierda" style = "height: 50px; overflow:scroll" > Caja6 caja6 caja6 </div>
  <div class = "cajaFlotanteIzquierda" > Caixa7 </div>
  <div class = "cajaFlotanteIzquierda" > Caja8 </div>
</div>
</body>
```



# 1. Hojas de estilos: CSS3

La propiedad z-index controla la posición tridimensional de un elemento que se establece sobre el tercer eje Z , permitiendo así establecer qué caja se ve por encima de otra





# 1. Hojas de estilos: CSS3

## Reglas arroba

Las reglas arroba comienzan siempre con el símbolo “@” seguido de un nombre clave; se pueden declarar en una hoja externa o incrustada.

Regla	Descripción
@import	Agrega los estilos <u>CSS</u> de un documento externo.
@media	Aplica las reglas contenidas en el tipo de dispositivo que se especifica.
@font-face	Especifica una fuente no incluida en el navegador que se descargará el usuario.
@charset	Especifica cuál es el juego de caracteres que utilizaremos dentro del archivo <u>CSS</u> .
@page	Establece las dimensiones, orientación y márgenes del cuadro de un documento.
@supports	Especifica una o varias condiciones, que en caso de cumplirse se aplicarán los estilos definidos.

# 1. Hojas de estilos: CSS3

## Regla @media

```
/* Definición de diferentes estilos por diferentes tipos de medios.*/
@media print {
    body { fuente-size : 10pt }
}
@media screen {
    body { fuente-size : 12pt }
}
@media screen, print {
    body { line-height : 1.2 }
}
```

## Regla @font-face

```
/* Inclusión de una fuente que prevemos que el navegador no tendrá instalada, para así utilizarla después en la definición del estilo para los párrafos. */
@font-face {
    font-family : DeliciousRoman ;
    src : url ( "Delicious-Roman.otf" ) ;
}
p {
    font-family : DeliciousRoman , Helvetica , Arial , sans-serif ;
}
```

# 1. Hojas de estilos: CSS3

## Regla @charset

```
/* activa el juego de caracteres para la hoja de estilo en Unicode UTF-8 */  
@charset "UTF-8";  
/* activa el juego de caracteres para la hoja de estilo Latin-9 (lenguas del oeste de Europa, con el símbolo del euro) */  
@charset 'iso-8859-15';
```

## Regla @page

```
/* definimos los márgenes de impresión de la página. */  
@page {margin-left: 1cm; margin-right: 0.5cm;}
```

## Regla @supports

```
/*si la propiedad display toma por valor flex, se aplicarán los siguientes estilos:*/  
@supports (display: flex) {  
  section { display : flex }  
}
```



# 1. Hojas de estilos: CSS3

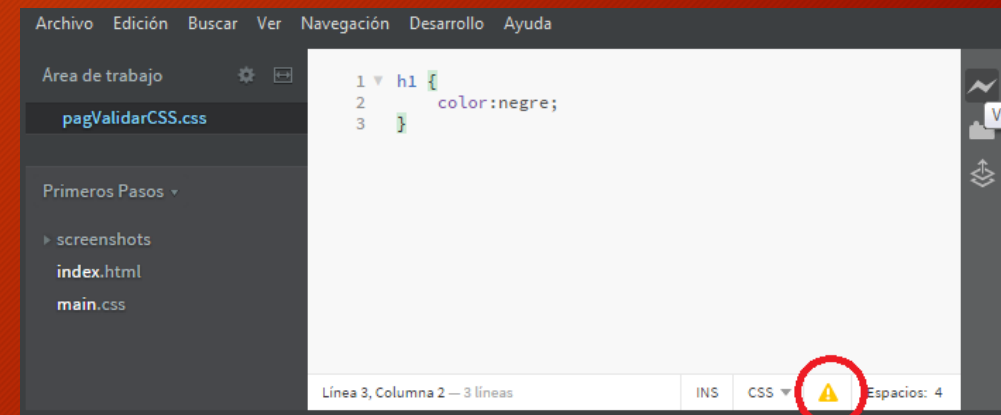
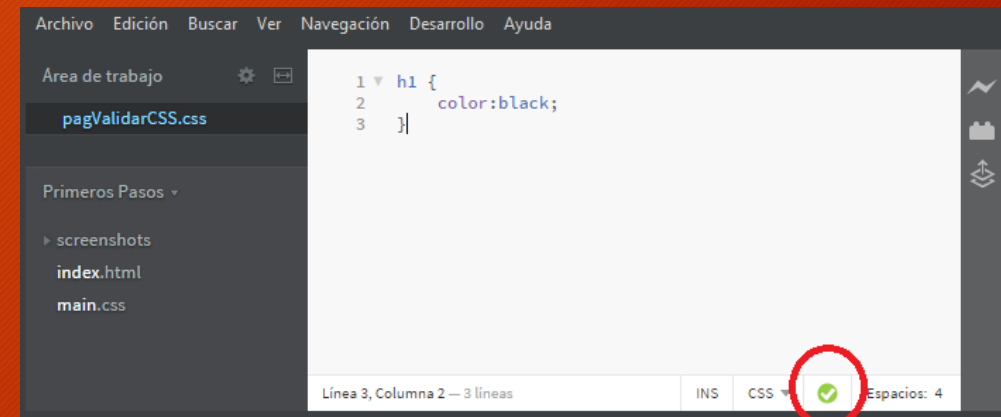
## Test y validación CSS

Para verificar que el código CSS es correcto, la gran mayoría de los editores de HTML y CSS contienen unas herramientas que permiten realizar una serie de pruebas para validarlo.

Si se quiere realizar este test con el editor Brackets, lo que se hará será utilizar la extensión CSSLint.

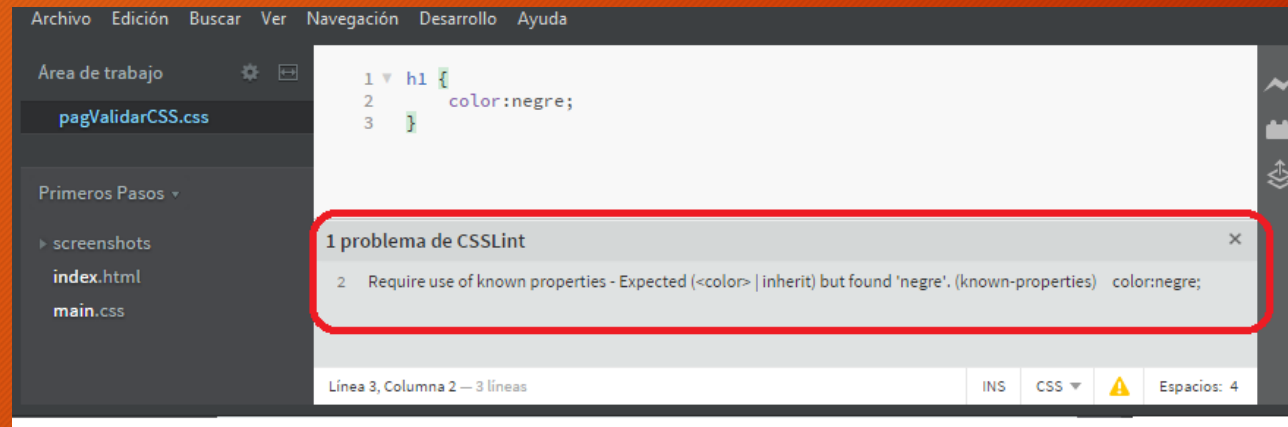
Esta extensión añade un icono a la barra de estado, de forma que cada vez que se guarda la hoja de estilos se valida.

En caso de que no haya ningún error se muestra un icono de color verde. En caso de que la haya, se muestra un triángulo amarillo.



# 1. Hojas de estilos: CSS3

Este validador no sólo detecta los errores, sino que también proporciona consejos para mejorar las hojas de estilos. Tal como se muestra en la imagen, haciendo clic en el triángulo naranja puede ver los errores detectados.



# 1. Hojas de estilos: CSS3

Alternativamente, también se puede verificar que el código en CSS es correcto mediante la web [jigsaw.w3.org/css-validator](http://jigsaw.w3.org/css-validator) .

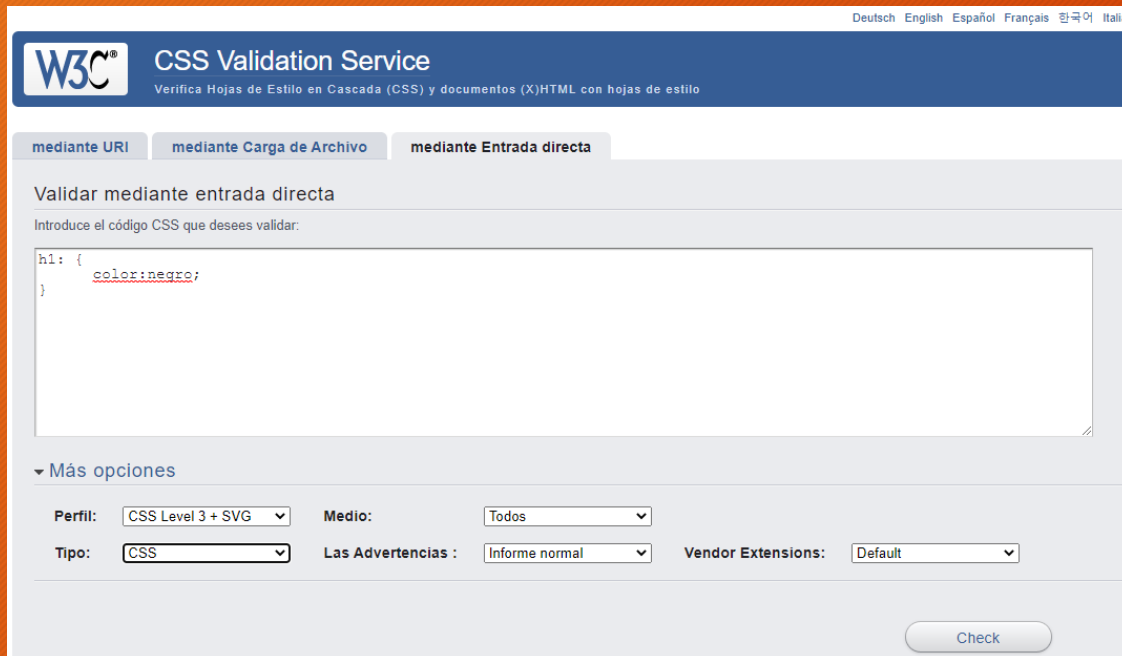
La validación se puede llevar a cabo de varias formas:

- Indicando la URL completa de la página que deseamos validar.
- Subiendo el documento CSS .
- Escribiendo el código CSS directamente.

Entonces seleccionamos More Options y allí nos aseguramos que en Perfil está seleccionado CSS Versión 3 . Por último, pulsamos el botón Check.



# 1. Hojas de estilos: CSS3



W3C® CSS Validation Service  
Verifica Hojas de Estilo en Cascada (CSS) y documentos (X)HTML con hojas de estilo

mediante URI mediante Carga de Archivo mediante Entrada directa

Validar mediante entrada directa

Introduce el código CSS que desees validar:

```
h1: {  
  color: negro;  
}
```

▼ Más opciones

Perfil: CSS Level 3 + SVG Medio: Todos Tipo: CSS Las Advertencias: Informe normal Vendor Extensions: Default

Check

## Resultados del Validador CSS del W3C para TextArea (CSS versión 3 + SVG)

**Disculpas! Hemos encontrado las siguientes errores (1)**

**URI : TextArea**

3

Error de análisis sintáctico `h1: { color: negro; }`

Vemos la pantalla de información de los errores CSS encontrados.

## Resultados del Validador CSS del W3C para TextArea (CSS versión 3 + SVG)

**¡Enhorabuena! No error encontrado.**

¡Este documento es [CSS versión 3 + SVG](#) válido!

En caso de que no hubiera ningún error CSS , se mostraría una pantalla como la de la imagen.

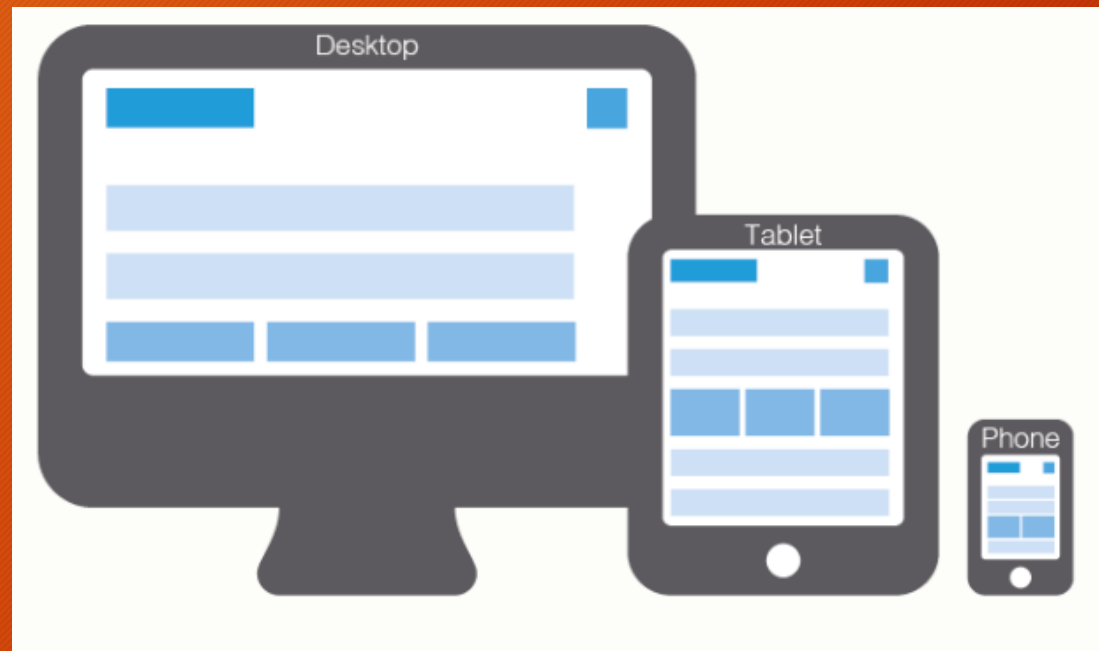
## 2. Diseño web adaptativo

El creciente uso de dispositivos móviles y tabletas para acceder a Internet en sustitución del ordenador pone de manifiesto que muchos sitios web no están optimizados para visualizarse correctamente en estos dispositivos. Así, muchos de los dispositivos suelen estar limitados por el tamaño de la pantalla y requieren un enfoque diferente sobre cómo presentar el contenido.

El contenido de una web podría presentarse por pantalla en una columna en un teléfono móvil o en dos columnas en una tableta, y en tres columnas en un ordenador.

## 2. Diseño web adaptativo

En consecuencia, tal y como se puede apreciar en la figura , existen diferentes tamaños de pantalla en teléfonos móviles, tabletas, ordenadores de sobremesa, consolas de juegos, televisores, etc. Los tamaños de pantalla serán siempre cambiantes, por lo que es importante que el sitio web pueda adaptarse a cualquier tamaño de pantalla en la actualidad o en el futuro.





## 2. Diseño web adaptativo

El diseño adaptativo se logra con el uso de las hojas de estilo CSS aplicadas a los elementos HTML de las páginas web. Así, se puede cambiar el tamaño, esconder, encoger, agrandar o mover el contenido de la página para hacerlo visible y adecuado a cualquier pantalla.

Se define el diseño adaptativo o, en inglés, Responsive Web Design, como una técnica de diseño y desarrollo web que permite que un sitio web se vea correctamente en todos los tipos de dispositivos: ordenadores de sobremesa, portátiles, tabletas, teléfonos móviles, televisores , etc.

## 2. Diseño web adaptativo

### Técnicas para crear un diseño adaptativo

Para que el sitio web sea adaptativo es importante que disponga de varias versiones para que se pueda visualizar correctamente desde cualquier dispositivo con distintos tamaños de pantalla y resoluciones. De esta forma, al menos habría que disponer de:

- Una versión para teléfono móvil.
- Una versión por móvil en modo apaisado.
- Una versión para la tableta.
- Una versión para ordenador de escritorio.



## 2. Diseño web adaptativo

No debe olvidarse que la experiencia de uso es muy diferente con un dispositivo u otro, ya que el tamaño de la pantalla de un móvil suele ser más pequeño, a pesar de que su resolución puede ser mayor que la de un monitor de 17”.

Así que no sólo es necesario adaptar el contenido a la pantalla, sino que también es necesario tener en cuenta las necesidades del usuario y las capacidades del dispositivo. Por eso, también habrá que considerar lo que quiere hacer el usuario en primer lugar y mostrarle la información en función de este criterio. Tampoco debe suponerse que el usuario con un teléfono móvil no querrá acceder a toda la información del sitio, y por tanto habrá que idear la manera de acceder a la información, cómo se muestra y en qué orden.

Las técnicas que se utilizarán para crear el diseño adaptativo son:

- **Media queries** que permiten saber que tipo de dispositivo y cuales son las dimensiones de la pantalla del usuario
- **Un grid flexible**, es decir, un diseño de página basado en una cuadrícula que pueda adaptarse a la resolución de la pantalla.
- **Contenidos e imágenes flexibles** a los que, utilizando estilos CSS , se aplican cambios de escala dinámicos para que se adapten al tamaño de la pantalla.



## 2. Diseño web adaptativo

### Diseño web fluido 'vs.' adaptativo

Una web tiene un diseño líquido o fluido cuando el contenido se ajusta horizontalmente al tamaño de la pantalla sin necesidad de la barra de desplazamiento horizontal ( scroll ).

El problema surge tanto en pantallas muy grandes como en pantallas muy pequeñas, ya que el diseño se desajusta, bien porque aparecen grandes espacios en blanco o porque las imágenes se miniaturizan y los textos se vuelven ilegibles.

Para obviar estos problemas, los diseñadores pueden crear varias versiones en CSS según la pantalla donde deba visualizarse, soportando así una carga de trabajo adicional sobre el mantenimiento de las diferentes versiones.

En un diseño adaptativo o responsive , el diseño se basa en una cuadrícula flexible y los bloques se ordenan y se jerarquizan de tal modo que los elementos se muestran en una o varias columnas, en función del tamaño de la pantalla.



## 2. Diseño web adaptativo

### Media Queries

Una media query es una consulta que da información del tipo de medio en el que se visualiza la página web, para así poder limitar los estilos que se van a aplicar. El resultado de esta consulta será un booleano, así que cuando sea verdadero se aplicarán las reglas de estilo correspondientes, siguiendo las normas de cascada.

Veamos con un ejemplo la sintaxis de una media query :

Especificamos la consulta al de <head>la página HTML a través del elemento <link>.

```
<link rel = "stylesheet" media = "screen and (width:780px)" href = "estilos780.css" />
```

Y la misma consulta se podría especificar en un documento CSS a través de una regla @media:

```
@media screen and (width:780px) {  
    /*...*/  
}
```



## 2. Diseño web adaptativo

Los distintos tipos de medios que podemos especificar son:

- screen (valor por defecto): para presentación en pantallas de ordenadores no paginados.
- print : para la salida por una impresora.
- projection : para presentación en proyectores.
- handheld : para dispositivos de mano (típicamente pequeña pantalla, de ancho limitado).
- aural : para sintetizadores de voz.
- braille: para presentación en dispositivos Braille.
- tty : para pantalla en celdas de caracteres.
- tv : para presentación en televisores.
- all : para todos los dispositivos de salida.



## 2. Diseño web adaptativo

En la siguiente tabla se pueden ver los criterios que podemos usar en las consultas.

Criterio	Valor	Mín./Máx.	Descripción
width	Largo	Sí	Para examinar el ancho de la zona de visualización del navegador.
height	Largo	Sí	Para examinar la altura de la zona de visualización del navegador
device-width	Largo	Sí	Para examinar el ancho físico de la pantalla de difusión.
device-height	Largo	Sí	Para examinar la altura física de la pantalla de difusión.
orientation	<i>Landscape o portrait</i>	No	Para examinar si el usuario utiliza tableta táctil verticalmente, <i>portrait</i> u horizontalmente, <i>landscape</i> .
aspect-ratio	Ratio	Sí	Para examinar el coeficiente ancho/alto.
device-aspect-ratio	Ratio	Sí	Para examinar el coeficiente físico ancho/alto de la pantalla.
color		Sí	Para examinar si el soporte de difusión utiliza el color (valor por defecto en caso de que no se haya especificado), o el blanco y negro o una escalera de grises.
color-index	Número	Sí	Para examinar el número de colores de la tabla de colores.
monochrome	Número	Sí	Para examinar el número de niveles de gris para los dispositivos monocromos.
resolution	DPI	Sí	Para examinar la resolución de la pantalla de visualización expresada en DPI.
scan	<i>Progressive o interlace</i>	No	Para examinar el tipo de exploración de pantallas de televisión.
grid	Número	No	Para examinar si la pantalla de difusión utiliza una cuadrícula con un único tamaño de fuente.

Se pueden redactar consultas utilizando operadores lógicos como son not , and y only . Además, las listas separadas por comas (",") se comportan como el operador or.

## 2. Diseño web adaptativo

Para examinar si la ventana tiene una anchura de 700 px o más y la pantalla está en horizontal:

```
@media (min-width: 700px) and (orientation: landscape) {  
    /*...*/  
}
```

Para examinar si la ventana tiene una anchura de 700 px o más, la pantalla está en horizontal y el dispositivo es un televisor:

```
@media tv and (min-width: 700px) and (orientation: landscape) {  
    /*...*/  
}
```

Para examinar si el ancho mínimo es 700 px o si el dispositivo está en horizontal:

```
@media (min-width: 700px), handheld and (orientation: landscape) {  
    /*...*/  
}
```

Para examinar pantallas que no tengan una resolución de 780 px:

```
@media screen and (not width:780px) {  
    /*...*/  
}
```

Puede consultar la siguiente web para obtener más información sobre las media queries :  
[developer.mozilla.org/es/docs/Web/CSS/Media\\_Queries/Using\\_media\\_queries](https://developer.mozilla.org/es/docs/Web/CSS/Media_Queries/Using_media_queries).

## 2. Diseño web adaptativo

### El viewport

Prácticamente todos los navegadores de dispositivos móviles, al entrar en un sitio web, analizan su tamaño y lo escalan para que se muestre completamente en la pantalla; en consecuencia, según esté optimizada la web, es posible que el resultado sea muy pequeño. Veamos con un ejemplo la sintaxis de una media query :

El viewport es el área visible del navegador. En los navegadores para móviles no se corresponde con el tamaño real de la pantalla en píxeles, sino en el espacio que la pantalla está emulando que tiene.

Para solucionar esta situación se puede alterar el viewport que está configurado en el navegador a través del elemento HTML, `<meta>` que ubicaremos en el de `<head>` la página web.



## 2. Diseño web adaptativo

Las características o parámetros que podemos modificar del viewport se especifican en el atributo content separadas por coma

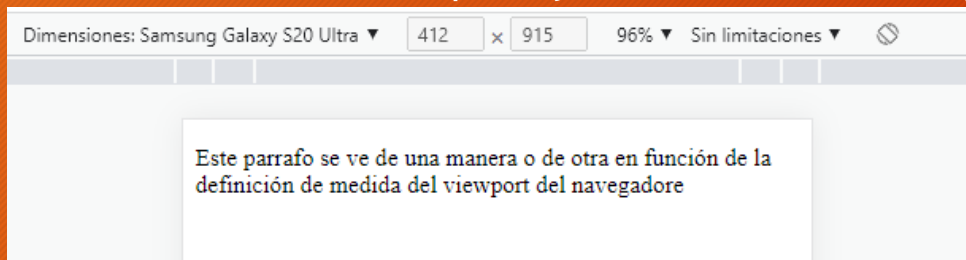
Paràmetre	Descripció
width	Amplada del <i>viewport</i> .
height	Altura del <i>viewport</i> .
initial-scale	Escala inicial del document.
minimum-scale	Escala mínima configurable del document.
maximum-scale	Escala màxima configurable del document.
user-scalable	Si es permet o no a l'usuari fer zoom.

De esta forma, podemos fijar el ancho y la altura del viewport con píxeles (320 px, 480 px, etc.), o bien también podemos usar dos constantes: y device-width, device-height referentes a la anchura ya la altura del dispositivo móvil.

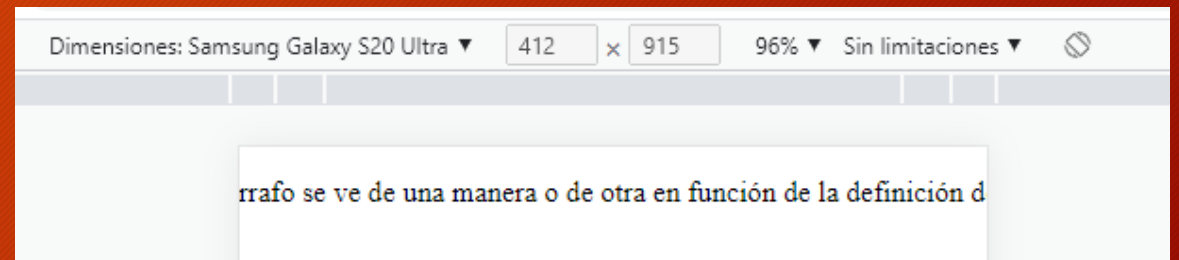
## 2. Diseño web adaptativo

```
<!DOCTYPE html>
<head>
  <html lang = "es">
  <meta charset="utf-8" name = "viewport" content="width=device-width, initial-scale=1.0"/>
  <title> Primera Página Web </title>
</head>
<body>
  <p> Este parrafo se ve de una manera o de otra en función de la definición de medida del viewport del navegador </p>
</body>
</html>
```

En las siguientes imagenes vemos cómo se visualiza el contenido de la página web en un teléfono móvil si no se ha definido el viewport, y cómo se ve si el viewport está definido.



Viewport definido

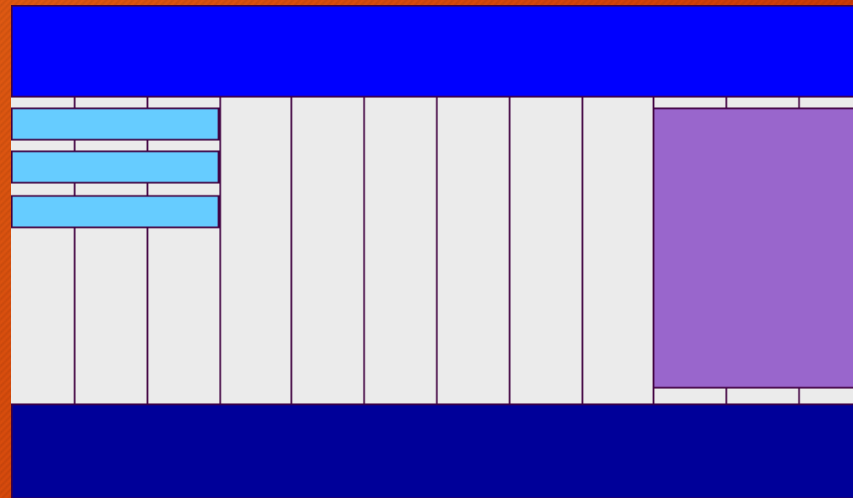


Viewport no definido

## 2. Diseño web adaptativo

### Desarrollar un grid fluido

Hay páginas web que están estructuradas en lo que se llama grid , que significa que la página está dividida en columnas. De esta forma es muy sencillo colocar los elementos, tal y como se aprecia en la imagen.



Este diseño se consigue definiendo para cada fila, row , la distribución del número de columnas, col , que nos interese. Veamos en el ejemplo siguiente el código CSS para la creación de un grid fluido de 12 columnas y la aplicación de este grid en HTML .



## 2. Diseño web adaptativo

### Archivo CSS

```
* {
  box-sizing : Border-box ;
}

@media only screen and (max-width: 768px) {
  /* Para teléfonos móviles: */
  [ class *= "col-" ] {
    width : 100% ;
  }
}

.row:after {
  content : "" ;
  clear : both ;
  display : block ;
}

[class*="col-"] {
  float : left ;
  padding : 15px ;
}

@media only screen and (min-width: 768px) {
  .col-1 { width : 8.33% ; }
  .col-2 { width : 16.66% ; }
  .col-3 { width : 25% ; }
  .col-4 { width : 33.33% ; }
  .col-5 { width : 41.66% ; }
  .col-6 { width : 50% ; }
  .col-7 { width : 58.33% ; }
  .col-8 { width : 66.66% ; }
  .col-9 { width : 75% ; }
  .col-10 { width : 83.33% ; }
  .col-11 { width : 91.66% ; }
  .col-12 { width : 100% ; }
}
```

```
html {
  font-family : "Lucida Sans" , sans-serif ;
}

.header {
  background-color : #9933cc ;
  color : #ffffff ;
  padding : 15px ;
}

.footer {
  background-color : #9933cc ;
  color : #ffffff ;
  padding : 15px ;
  text-align : center ;
}

.menu ul {
  list-style-type : none ;
  margin : 0 ;
  padding : 0 ;
}
```

```
.menu li {
  padding : 8px ;
  margin-bottom : 7px ;
  background-color : #33b5e5 ;
  color : #ffffff ;
  box-shadow : 0 1px 3px rgba ( 0 , 0 , 0 , 0.12 ) , 0 1px 2px rgba ( 0 , 0 , 0 , 0.24 ) ;
}

.menu li :hover {
  background-color : #0099cc ;
}

.aside p {
  height : 100px ;
  padding : 8px ;
  margin : 0 ;
  background-color : #33b5e5 ;
  color : #ffffff ;
  box-shadow : 0 1px 3px rgba ( 0 , 0 , 0 , 0.12 ) , 0 1px 2px rgba ( 0 , 0 , 0 , 0.24 ) ;
}
```

## 2. Diseño web adaptativo

### Archivo HTML

```
<!DOCTYPE html>
<html lang = "es">
  <meta charset="utf-8" />
  <title> Primera Página Web </title>
  <link rel="stylesheet" href="./css/grid.css" />
</html>
<body>
  <header>
    <div class = "header">
      <h1> Estudios de grado superior de Informática </h1>
    </div>
  </header>

  <content>
    <div class = "row">
      <div class = "col-9">
        <div class = "row">
          <nav>
            <div class = "col-3 menu">
              <ul>
                <li> ASIR </li>
                <li> DAM </li>
                <li> DAW </li>
              </ul>
            </div>
          </nav>
          <div class = "col-9">
            <article>
              <h2> M1. Sistemas informáticos </h2>
              <p> Bloque 1. 108 horas (6 horas semanales) </p>
              <p> Bloque 2. 72 horas (6 horas semanales) </p>
            </article>
          </div>
        </div>
      </div>
      <div class = "row">
        <div class = "col-3"></div>
        <div class = "col-9">
          <article>
            <h2> M2. Bases de datos </h2>
            <p> Bloque 1. 106 horas (6 horas semanales) </p>
            <p> Bloque 2. 79 horas (6 horas semanales). Incluye trabajo en equipo y práctica virtual obligatoria. </p>
          </article>
        </div>
      </div>
    </div>
  </div>
```

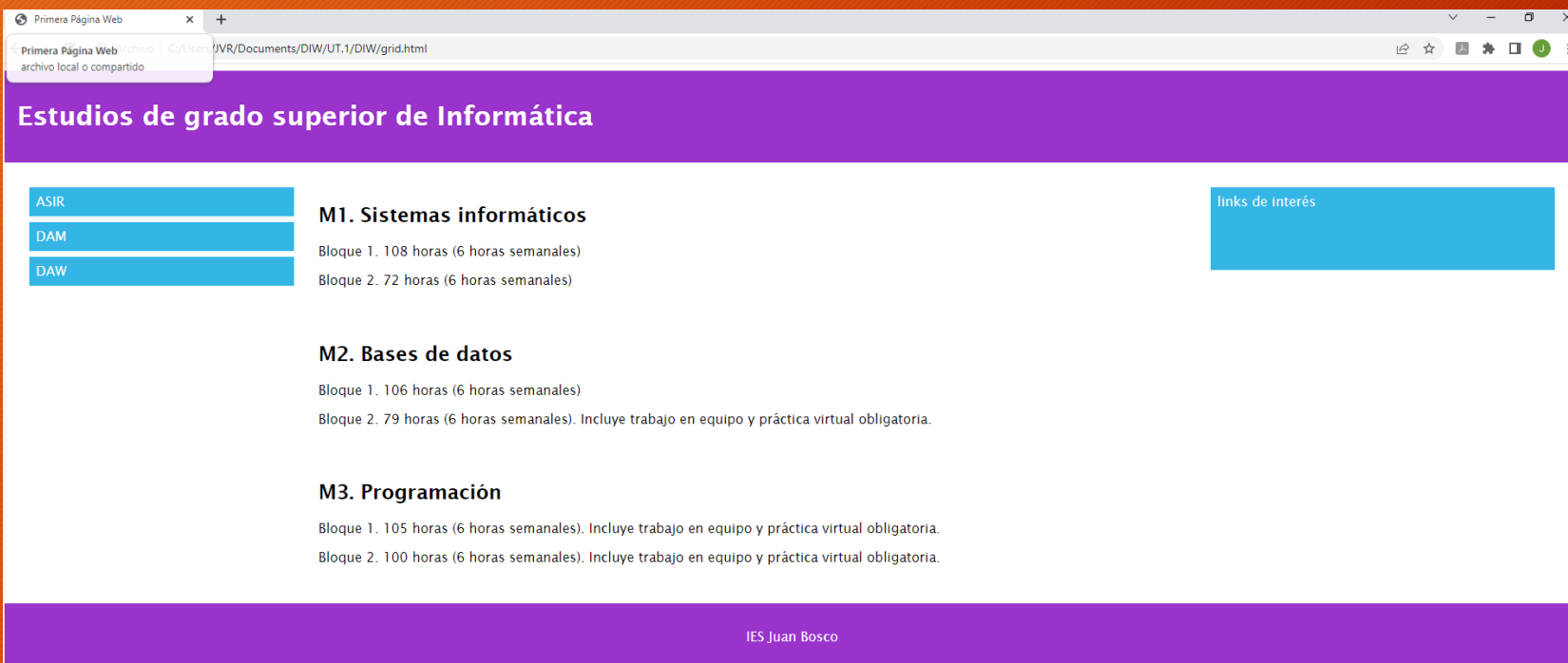
```
    <div class = "row">
      <div class = "col-3"></div>
      <div class = "col-9">
        <article>
          <h2> M3. Programación </h2>
          <p> Bloque 1. 105 horas (6 horas semanales). Incluye trabajo en equipo y práctica virtual obligatoria. </p>
          <p> Bloque 2. 100 horas (6 horas semanales). Incluye trabajo en equipo y práctica virtual obligatoria. </p>
        </article>
      </div>
    </div>

    <div class = "col-3">
      <div class = "row">
        <div class = "col-12">
          <aside>
            <div class = "aside">
              <p> links de interés </p>
            </div>
          </aside>
        </div>
      </div>
    </div>
  </content>

  <footer>
    <div class = "footer">
      <p> IES Juan Bosco </p>
    </div>
  </footer>
</body>
</html>
```

## 2. Diseño web adaptativo

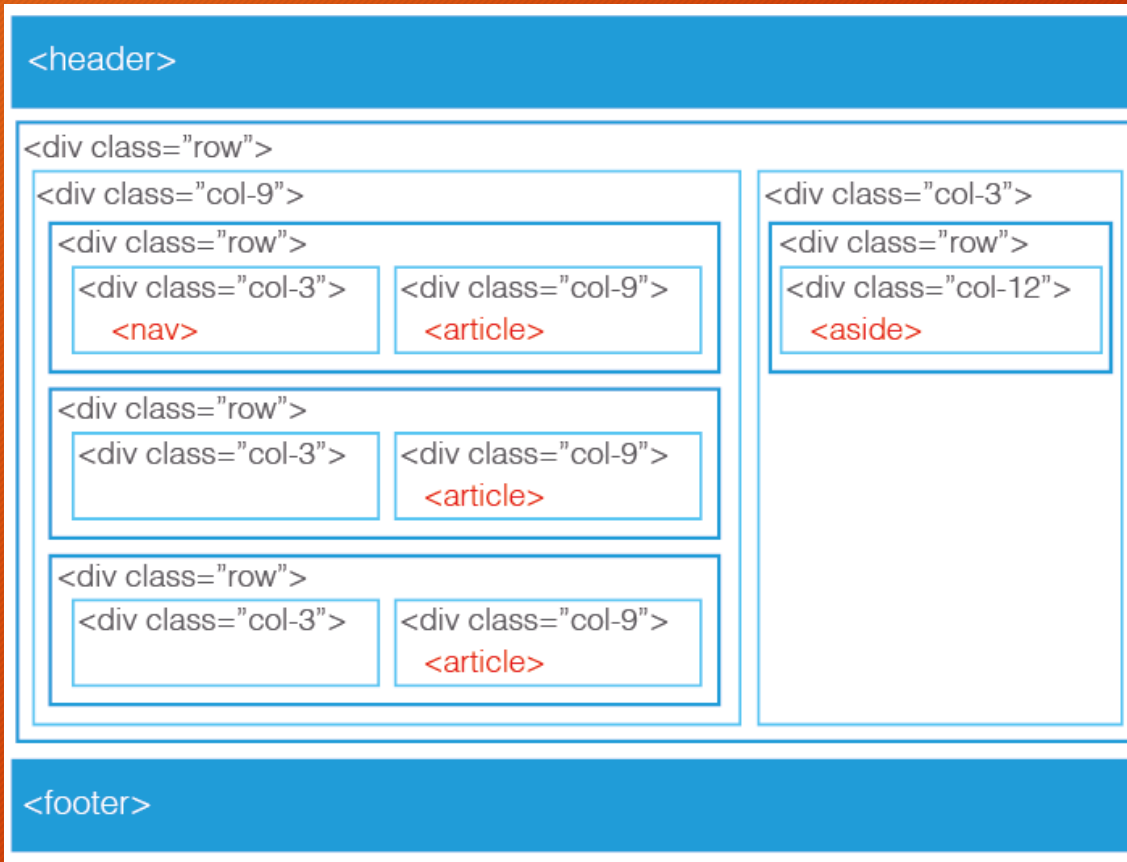
### Visualización página web





## 2. Diseño web adaptativo

### División del espacio



## 2. Diseño web adaptativo

Fíjete en el ejemplo anterior, como en una fila ( row ) sólo puede haber columnas ( col ) y que el contenido siempre se sitúa dentro de las columnas. Al mismo tiempo tiene que dentro de una columna puede haber otra fila, la cual también se dividirá en 12 columnas.

Con este diseño de grid se ve que los bloques se posicionan en un sitio u otro en función del tamaño de la pantalla. Este funcionamiento se debe a que la posición de las columnas es flotante, y que para tamaños de pantalla inferiores a 768 px la anchura de la columna es del 100%.

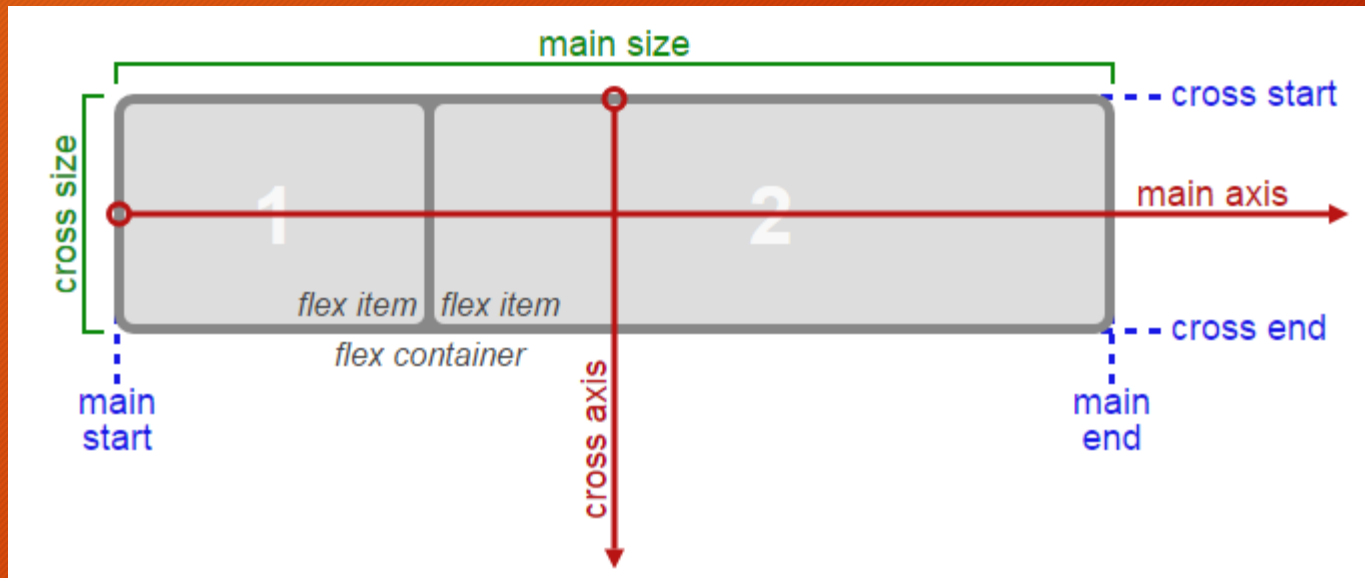
Cabe remarcar la dificultad que puede haber en este tipo de grid si se quiere cambiar la disposición de algunos de los bloques cuando se cambia el tamaño de la pantalla, como por ejemplo hacer que el menú de navegación (los botones ASIR , DAM y DAW ) aparezcan encima del bloque links de interés en lugar de después del header .

## 2. Diseño web adaptativo

### Desarrollar un grid flexible

El desarrollo de un grid flexible se basa en el nuevo modelo de caja, flexbox . Así pues, podemos crear una caja o contenedor flexible a través de la propiedad CSS `display: flex` . De esta forma, la declaración de esta propiedad convierte de forma automática sus elementos hijos directos en flexibles.

Un contenedor flexible tiene un eje principal, `main axis` , que es la dirección en la que se posicionan los elementos flexibles, y también tiene un eje transversal, perpendicular al eje principal. Estos dos ejes tienen una serie de propiedades que controlan cómo se posiciona cada elemento flexible en relación a los demás.





## 2. Diseño web adaptativo

En la tabla se muestran las diferentes propiedades CSS que nos permitirán crear el contenedor flexible.

Propiedad	Descripción
<code>flex-direction</code>	Especifica cómo se sitúan los elementos dentro del contenedor. Los valores que puede tomar son <i>row</i> , <i>row-reverse</i> , <i>column</i> , <i>column-reverse</i> .
<code>flex-wrap</code>	Especifica si el contenedor tiene una o varias líneas. Los valores que puede tomar son: <i>no-wrap</i> , <i>wrap</i> , <i>wrap-reverse</i> .
<code>flex-flow</code>	Propiedad abreviada de las propiedades anteriores. <i>flex-flow: row wrap</i> .

```
#cajaFlex {  
  display: flex ;  
  display: -webkit-flex ;  
  flex-Direction: row ; -webkit-flex-direction : row ;  
  flex-Wrap : Wrap ; -webkit-flex-wrap : wrap ;  
}
```

Ejemplo

## 2. Diseño web adaptativo

### Contenidos e imágenes flexibles

Los contenidos y las imágenes que formarán parte de la página web también deben ser flexibles. No es objetivo de esta unidad cómo mostrar imágenes flexibles; por tanto, nos limitaremos a tratar el contenido flexible.

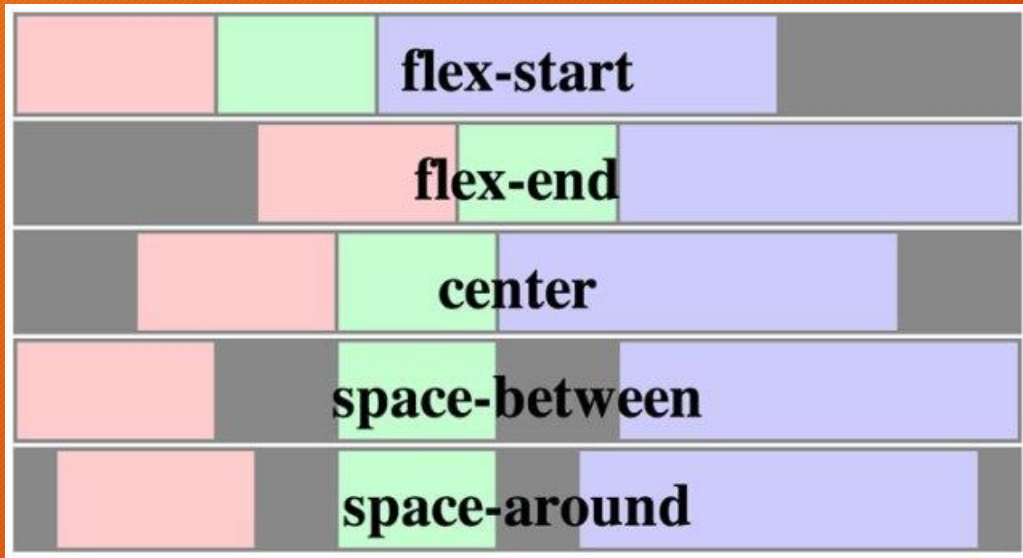
En la tabla se pueden consultar las propiedades CSS que pueden asignarse a los elementos que están dentro del contenedor.

Propiedad	Descripción
order	Para establecer el orden en el que aparecen los elementos de una caja flexible. Por defecto es 0.
flex-grow	Especifica el factor de crecimiento, es decir, cuánto crece un elemento en relación con los demás cuando existe espacio disponible en el contenedor. Por defecto es 0.
flex-shrink	Especifica el factor de reducción, es decir, cuánto decrecerá un elemento en relación con los demás cuando no haya espacio disponible en el contenedor. Por defecto es 1.
flex-basis	Toma el mismo valor que la propiedad width. Especifica el tamaño inicial del elemento antes de distribuir el espacio libre con las propiedades anteriores. Por defecto es <code>main-size(auto)</code> .
flex	Propiedad abreviada de las anteriores.

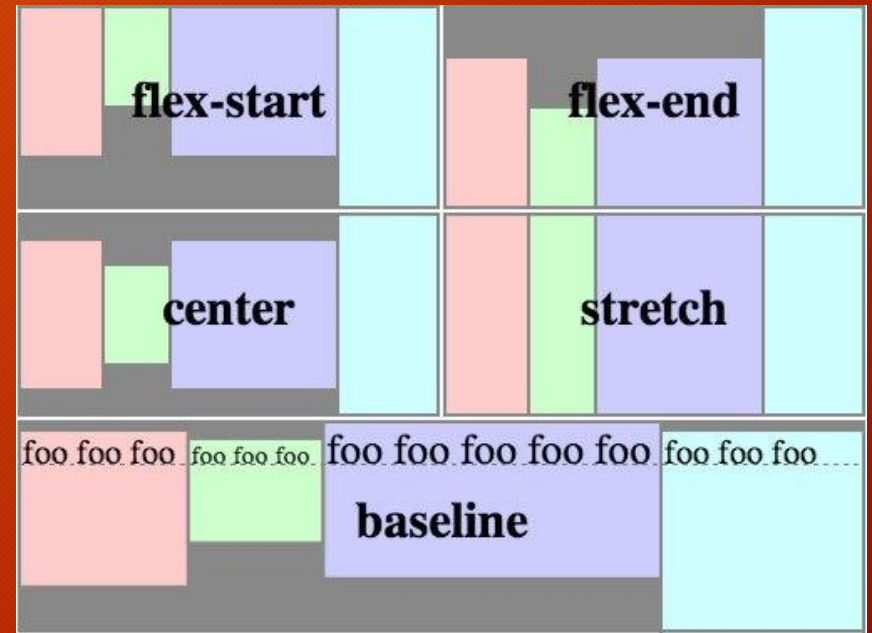
Las propiedades para alinear los elementos flexibles se pueden consultar en la siguiente tabla

Propiedad	Descripción
justify-content	Permite alinear los elementos en el eje principal. Toma por valor <code>flex-start</code> , <code>flex-end</code> , <code>center</code> , <code>space-between</code> , <code>space-around</code> .
align-items	Permite alinear todos los elementos en el eje transversal. Toma por valor: <code>auto</code> , <code>flex-start</code> , <code>flex-end</code> , <code>center</code> , <code>baseline</code> , <code>stretch</code> .
align-self	Permite alinear los elementos individuales en el eje transversal; así sobrescribe la propiedad anterior.

## 2. Diseño web adaptativo



Alineamiento sobre el eje principal



Alineamiento sobre el eje transversal



## 2. Diseño web adaptativo

```
body {
  font: 18px Arial ;
  background: #eeeeee ;
}

#main {
  min-height: 450px ;
  margin: 0px ;
  padding: 0px ;
  display: -webkit-flex ;
  display: flex ;
  -webkit-flex-flow: row ;
  flex-flow: row ;
}

#main > article {
  margin: 4px ;
  padding: 5px ;
  border: 1px solid #ccc ;
  border-radius: 2pt ;
  background: #white ; _
  -webkit-flex: 3 1 60% ;
  flex: 3 1 60% ;
  -webkit-Order: 2 ;
  order: 2 ;
}

#main > nav {
  margin: 4px ;
  padding: 5px ;
  border: 1px solid #8888bb ;
  border-radius: 2pt ;
  background: #33b5e5 ;
  -webkit-flex: 1 6 20% ;
  flex: 1 6 20% ;
  -webkit-Order: 1 ;
  orden: 1 ;
}
```

### Archivo CSS

```
#main > aside {
  margin: 4px ;
  padding: 5px ;
  border: 1px solid #8888bb ;
  border-radius: 2pt ;
  background: #33b5e5 ;
  -webkit-flex: 1 6 20% ;
  flex: 1 6 20% ;
  -webkit-Order: 3 ;
  order: 3 ;
}

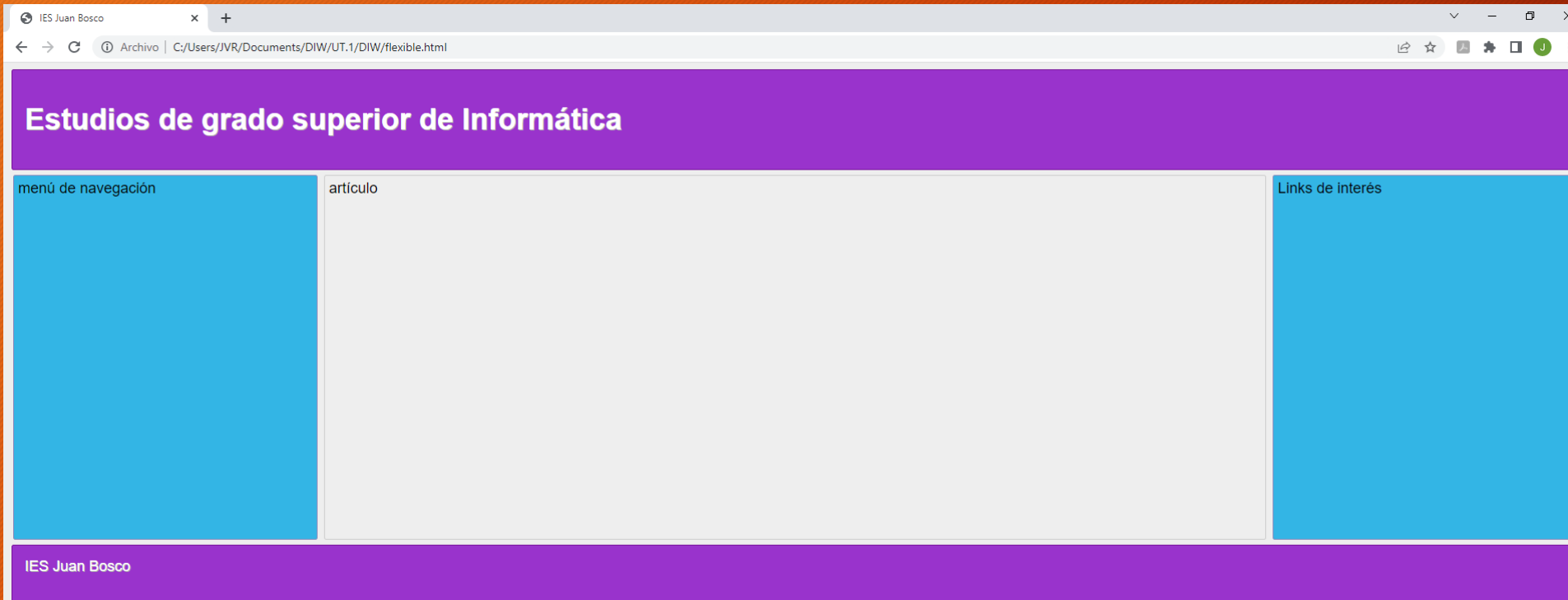
header ,
footer {
  display: block ;
  margin: 2px ;
  padding: 15px ;
  min-height: 50px ;
  border: 1px solid #771199 ;
  border-radius: 2pt ;
  background: #9933cc ;
  color: #ffffff ;
  text-shadow: 1px 1px #aaa ;
}

@media all and (max-width: 640px) {
  #main ,
  # page {
    -webkit-flex-flow: column ;
    flex-flow: column ;
  }
  #main > artículo ,
  #main > nav ,
  #main > aside {
    -webkit-order: 0 ;
    order: 0 ;
  }
  #main > nav ,
  #main > aside ,
  header ,
  footer {
    min-height: 50px ;
  }
}
```

### Archivo HTML

```
<!DOCTYPE html>
<html lang = "es">
<head>
  <meta charset="utf-8" />
  <title> IES Juan Bosco </title>
  <link rel="stylesheet" href="./css/flexible.css" />
</head>
<body>
  <header>
    <h1>Estudios de grado superior de Informática </h1>
  </header>
  <div id = 'main' >
    <article> artículo </article>
    <nav> menú de navegación </nav>
    <aside> Links de interés </aside>
  </div>
  <footer> IES Juan Bosco </footer>
</body>
</html>
```

## 2. Diseño web adaptativo



Has visto en el ejemplo anterior la utilización de las propiedades `display`, `flex` y `order`, y también cómo el contenido de la página en HTML se simplifica muchísimo.

Fíjate también cómo, para asegurar la compatibilidad con otros navegadores, se repite la misma propiedad CSS con su prefijo `-webkit-`.

### 3. Bootstrap 'framework'

En muchas ocasiones, para crear una interfaz web puede partirse de una infraestructura o esqueleto que ya contiene muchos de los elementos necesarios para la implementación de una aplicación web.

Un framework, por lo general, es una estructura conceptual de soporte definida, con módulos concretos de software que pueden ser utilizados como base para la organización y el desarrollo de software .

El objetivo de los frameworks es proporcionar una estructura común para que los desarrolladores no tengan que hacerlo desde cero y puedan reutilizar el código proporcionado.

Actualmente existen varios frameworks para poder desarrollar nuestro sitio web, como pueden ser: Bootstrap ( [getbootstrap.com](https://getbootstrap.com) ), Angular ( [angularjs.org](https://angularjs.org) ), Foundation ( [foundation.zurb.com](https://foundation.zurb.com) ), Leaf ( [getleaf.com](https://getleaf.com) ) o Materialize ( [materializecss.com](https://materializecss.com) ), entre muchos otros.

En este apartado se utilizará el framework Bootstrap 5 . Sus características más relevantes son:

- Es un framework fácil , rápido y gratuito para el desarrollo de la interfaz web.
- Incluye plantillas basadas en HTML y CSS , así como plugins JavaScript .
- Proporciona la capacidad de crear diseños adaptativos.



# 3. Bootstrap 'framework'

## Cómo obtener e instalar Bootstrap

Hay dos formas de iniciar el uso de Bootstrap en nuestro sitio web:

- Descargar Bootstrap en [getbootstrap.com](https://getbootstrap.com) y seguir las instrucciones que se proporcionan.
- Incluir Bootstrap de una CDN (Content Delivery Network).

### Ventaja de usar una CDN

El tiempo para cargar el sitio web es más rápido. Debido a que muchos usuarios ya han descargado Bootstrap de una CDN, cuando se visite nuestro sitio web se cargará desde la caché. Además, cuando el usuario realice su petición, ésta será servida del servidor más próximo a él.

### 3. Bootstrap 'framework'

Una vez descargado, lo descomprimiremos y obtendremos los archivos necesarios para utilizar Bootstrap:

- Carpeta css : aquí se encuentran los archivos bootstrap.css y bootstrap-theme.css. El primero es lo que debemos vincular en nuestros proyectos. Y el segundo es opcional, contiene estilos prefabricados en botones, barras de navegación, etc. Los archivos .min son los mismos; la diferencia es el tamaño y la lectura, ya que en la versión mínima se han quitado las tabulaciones y saltos de línea.
- Carpeta js : aquí están los archivos bootstrap.js y su versión reducida bootstrap.min.js. Este archivo contiene todas las librerías JavaScript, rutinas que hacen que Bootstrap sea más dinámico e interactivo. Por ejemplo, menús desplegable y otras funcionalidades.

### 3. Bootstrap 'framework'

```
<link rel="stylesheet" href="bootstrap/css/bootstrap.min.css">
</head>
.
.
.
.
<script src="bootstrap/js/bootstrap.min.js"></script>
```

Inclusión del 'framework' Bootstrap descargado

```
<link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css">
</head>
.
.
.
.
<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"></script>
```

Inclusión del 'framework' Bootstrap desde una CDN



# 3. Bootstrap 'framework'

## El contenedor

Bootstrap 5 requiere un elemento contenedor para envolver los contenidos del sitio y colocar el grid bootstrap . Podemos elegir dos tipos de contenedores:

- .container : define.
- .container-fluid : define un contenedor adaptable de ancho total, es decir, su anchura se corresponde con el ancho del wiewport .

```
<div class="container-fluid">Caja1</div>  
<div class="container">Caja2</div>
```

Definición de un contenedor fluido

# 3. Bootstrap 'framework'

## Bootstrap grid

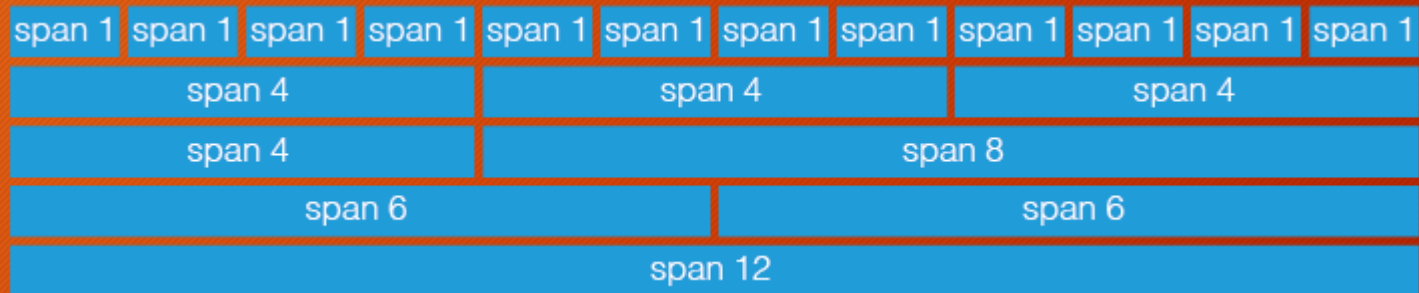
El sistema de columnas, flex , se utiliza para crear diseños de página a través de una serie de filas y columnas que acogen el contenido. Bootstrap incluye un sistema grid responsive fluido y mobile ferts de 12 columnas como máximo.

Los preprocesadores SASS y LESS son herramientas que permiten escribir pseudocódigo CSS (variables, condiciones, bucles, funciones) que será convertido en CSS real.

### 3. Bootstrap 'framework'

Este sistema funciona de la siguiente manera:

- Las filas, rows , deben ir dentro de un contenedor de ancho fijo o fluido. De esta forma, se establece su alineamiento y relleno.
- El contenido se coloca dentro de las columnas.



Para obtener más información sobre los estilos que conforman el grid de Bootstrap puede consultar la web: [getbootstrap.com/docs/5.2/layout/grid/](https://getbootstrap.com/docs/5.2/layout/grid/).



### 3. Bootstrap 'framework'

El sistema Bootstrap grid ofrece seis tamaños de cuadrícula para los distintos tipos de dispositivos. Así pues, la clase que se utiliza para definir cada fila de la cuadrícula es `.row`, y las clases que se utilizan para definir las columnas son:

- `.col-xs-(num_columnas)`: (<576 px).
- `.col-sm-(num_columnas)`: (>=576 px).
- `.col-md-(num_columnas)`: (>=768 px).
- `.col-lg-(num_columnas)`: (>=992 px)
- `.col-xl-(num_columnas)`: (>=1.200 px)
- `.col-xxl-(num_columnas)`: (>=1.400 px)

# 3. Bootstrap 'framework'

```
<div class="container">
  <div class="row">
    <div class="col-xs-12">
      <p>Dispositivos muy pequeños (anchura mas pequeña de 576px)</p>
    </div>
  </div>
  <div class="row">
    <div class="col-6 bg-primary">col-6</div>
    <div class="col-6 bg-danger">col-6</div>
  </div>

  <div class="row">
    <div class="col-sm-12">
      <p>Dispositivo pequeños (telefonos, anchura mas grande o igual a 576px)</p>
    </div>
  </div>
  <div class="row">
    <div class="col-sm-6 bg-primary">col-sm-6</div>
    <div class="col-sm-6 bg-danger">col-sm-6</div>
  </div>

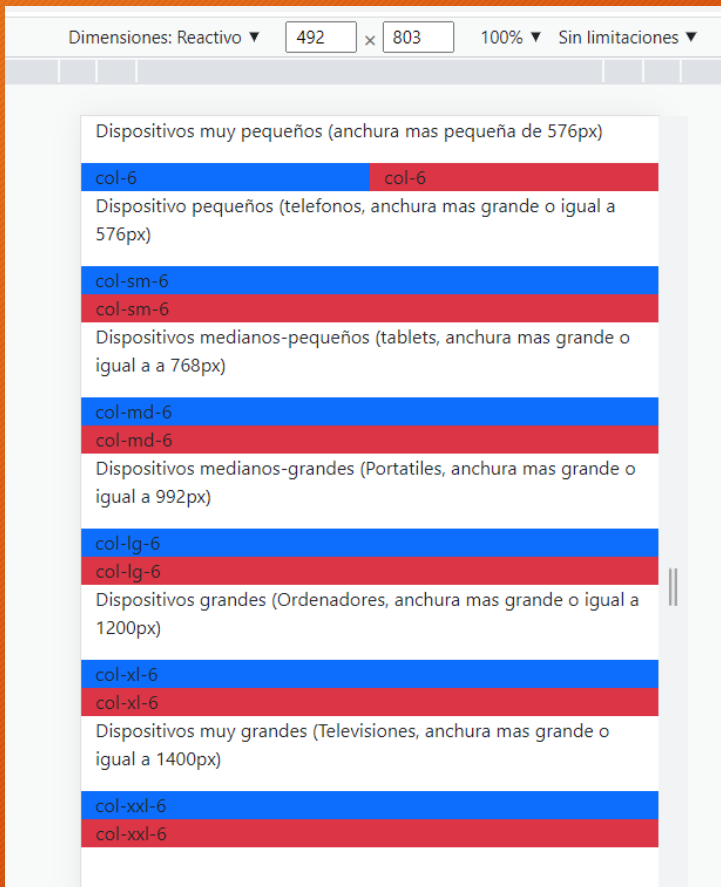
  <div class="row">
    <div class="col-md-12">
      <p>Dispositivos medianos-pequeños (tablets, anchura mas grande o igual a a 768px)</p>
    </div>
  </div>
  <div class="row">
    <div class="col-md-6 bg-primary">col-md-6</div>
    <div class="col-md-6 bg-danger">col-md-6</div>
  </div>
```

```
<div class="row">
  <div class="col-lg-12">
    <p>Dispositivos medianos-grandes (Portatiles, anchura mas grande o igual a 992px)</p>
  </div>
</div>
<div class="row">
  <div class="col-lg-6 bg-primary">col-lg-6</div>
  <div class="col-lg-6 bg-danger">col-lg-6</div>
</div>

<div class="row">
  <div class="col-xl-12">
    <p>Dispositivos grandes (Ordenadores, anchura mas grande o igual a 1200px)</p>
  </div>
</div>
<div class="row">
  <div class="col-xl-6 bg-primary">col-xl-6</div>
  <div class="col-xl-6 bg-danger">col-xl-6</div>
</div>

<div class="row">
  <div class="col-xxl-12">
    <p>Dispositivos muy grandes (Televisiones, anchura mas grande o igual a 1400px)</p>
  </div>
</div>
<div class="row">
  <div class="col-xxl-6 bg-primary">col-xxl-6</div>
  <div class="col-xxl-6 bg-danger">col-xxl-6</div>
</div>
</div>
```

### 3. Bootstrap 'framework'



En la imagen se puede ver cómo todas las filas se parten en dos columnas horizontales, pero cuando modificamos el tamaño del navegador cada fila se acomoda verticalmente en función de la clase y de la resolución. Así pues, los divs `col-sm-4` y `col-md-4` mantendrán su estructura en dos columnas alineadas horizontalmente, pero cuando el tamaño del viewport sea menor a 576 px y 768 px respectivamente éstas se acomodarán verticalmente. Igualmente ocurre con las demás.

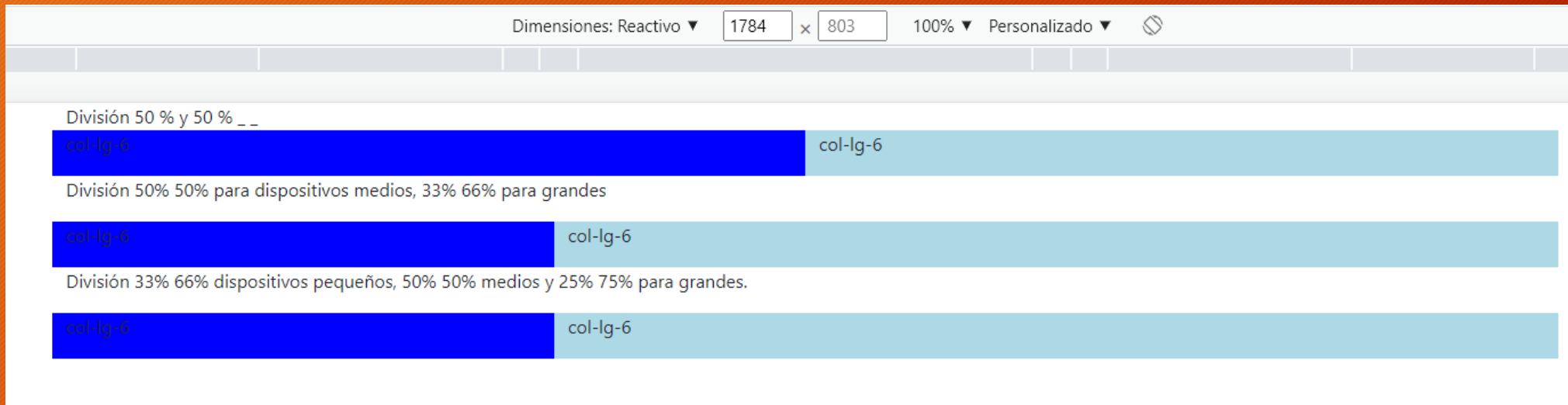


### 3. Bootstrap 'framework'

Además, Bootstrap proporciona columnas mixtas de ancho. Supongamos que queremos distribuir el espacio en dos columnas con el 50% cada una, de forma que las dos columnas no ocupen siempre el 50%, sino que en dispositivos mayores de 1.200 px estos divs ocupen el primero un 33% y el segundo un 66%. Haremos:

```
<body>
<div class = "container">
  <div class = "row">
    <div class = "col-lg-12">
      División 50 % y 50 % _ _
    </div>
  </div>
  <div class = "row">
    <div class = "col-lg-6 bg-primary" > col-lg-6 </div>
    <div class = "col-lg-6 bg-info" > col-lg-6 </div>
  </div>
  <div class = "row">
    <div class = "col-lg-12">
      <p> División 50% 50% para dispositivos medios, 33% 66% para grandes </p>
    </div>
  </div>
  <div class = "row">
    <div class = "col-lg-6 col-xl-4 bg-primary" > col-lg-6 </div>
    <div class = "col-lg-6 col-xl-8 bg-info" > col-lg-6 </div>
  </div>
  <div class = "row">
    <div class = "col-lg-12">
      <p> División 33% 66% dispositivos pequeños, 50% 50% medios y 25% 75% para grandes. </p>
    </div>
  </div>
  <div class = "row">
    <div class = "col-md-3 col-lg-6 col-xl-4 bg-primary" > col-lg-6 </div>
    <div class = "col-md-9 col-lg-6 col-xl-8 bg-info" > col-lg-6 </div>
  </div>
</div>
```

### 3. Bootstrap 'framework'



Véase en la figura cómo el tamaño de las columnas se redimensiona en función de la anchura y la resolución del viewport .

### 3. Bootstrap 'framework'

De la misma forma, en Bootstrap también disponemos de utilidades para ocultar y mostrar un elemento basándonos en el tamaño del dispositivo y la resolución del viewport .

Así pues, podemos usar las siguientes clases para mostrar elementos div :

- d-(Breakpoint)-none
- d-(Breakpoint)-inline
- d-(Breakpoint)-inline-block
- d-(Breakpoint)-block



# 3. Bootstrap 'framework'

## Desplazar columnas

Utilizamos la clase `-offset-` para desplazar cualquier columna a la derecha. Esta clase aumenta el tamaño del margen izquierdo de la columna en una cantidad equivalente a ese número de columnas. Por ejemplo, la clase `.offset-md-6` desplaza la columna hacia la derecha o da márgenes a la izquierda de un ancho equivalente a seis columnas.

```
<div class = "container">
  <div class="row">
    <div class="col-lg-4 bg-danger"> .col-lg-4 </div>
    <div class="col-lg-4 offset-lg-4 bg-info"> .col-lg-4 .offset-lg-4 </div>
  </div>

  <div class="row">
    <div class="col-lg-3 offset-lg-3 bg-danger"> .col-lg-3 .offset-lg-3 </div>
    <div class="col-lg-3 offset-lg-3 bg-info"> .col-lg-3 .offset-lg-3 </div>
  </div>

  <div class="row">
    <div class="col-lg-6 offset-lg-3 bg-info"> .col-lg-6 .offset-lg-3 </div>
  </div>
</div>
```

.col-lg-4

.col-lg-3 .offset-lg-3

.col-lg-6 .offset-lg-3

.col-lg-4 .offset-lg-4

.col-lg-3 .offset-lg-3