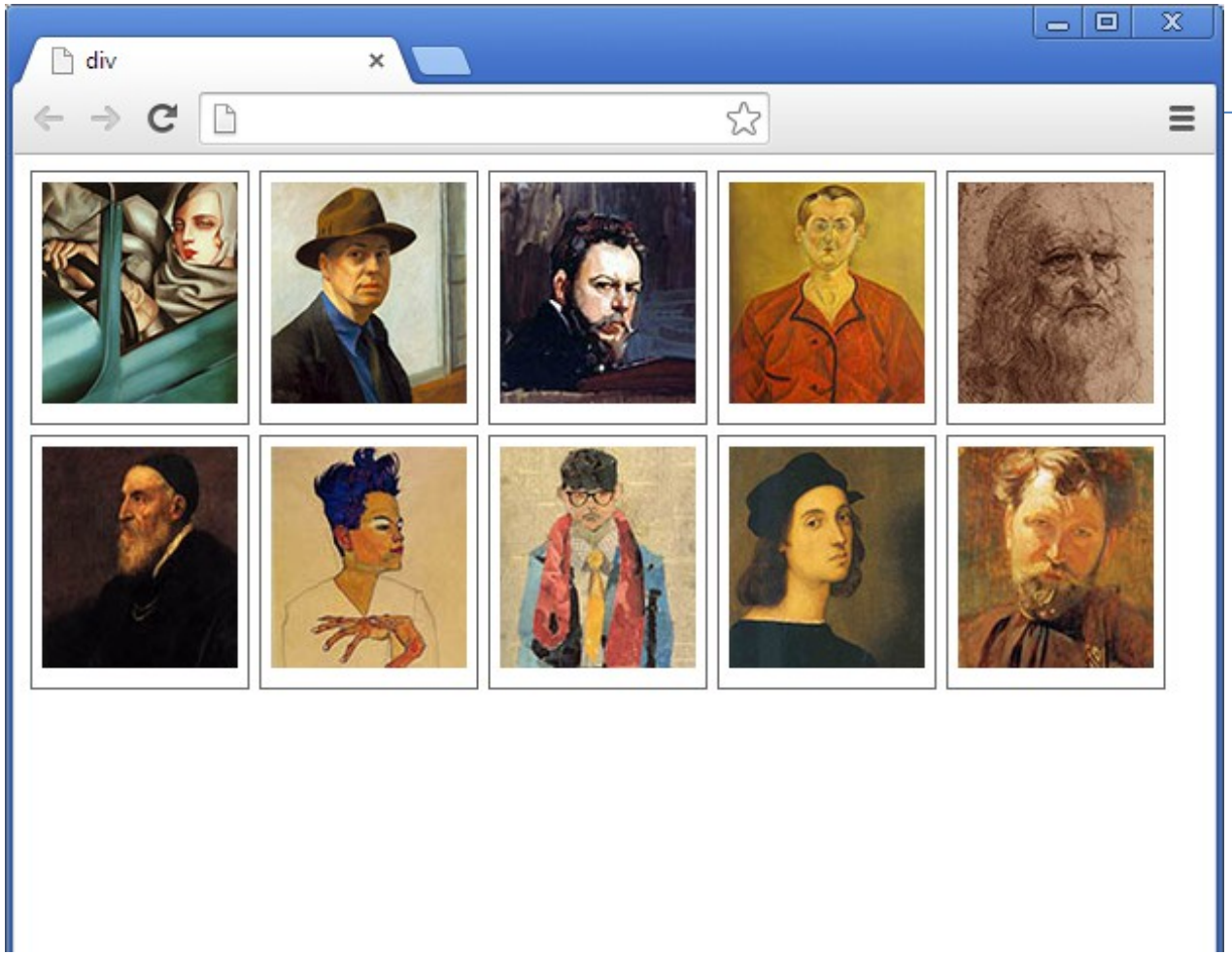
CSS

Estructura - CSS

e/s/d/
madrid

Estructura CSS - float

CSS3 permite posicionar los **div** en la página, **float**



CSS

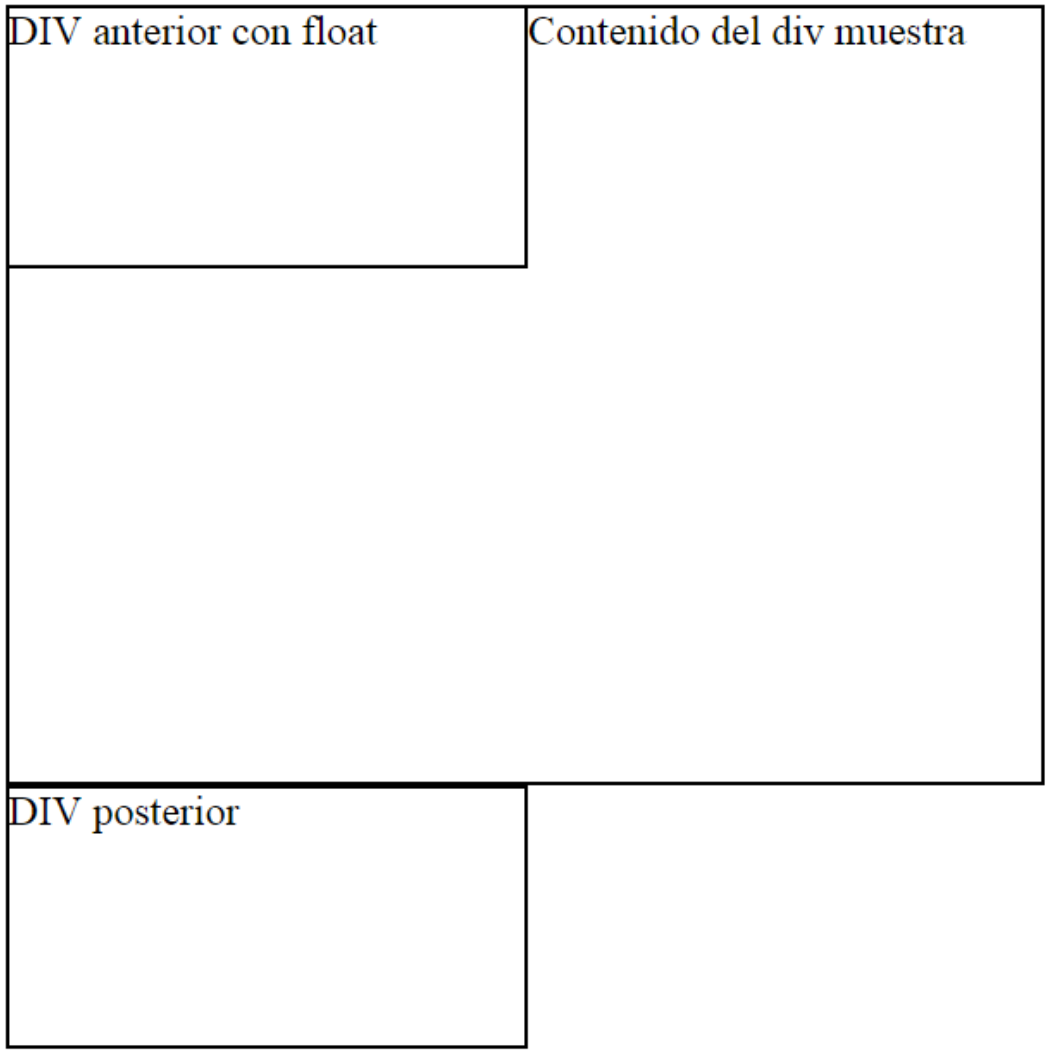
```
div {
  float: left;
  padding: 5px;
  margin-right: 5px;
  margin-bottom: 5px;
  border: 1px solid #666;
}
</style>
```


e/s/d/ madrid

Estructura CSS - float y clear

CSS3 permite posicionar los **div** en la página, **float** y **clear**

Con **float** el div “flota” a una posición relativa.

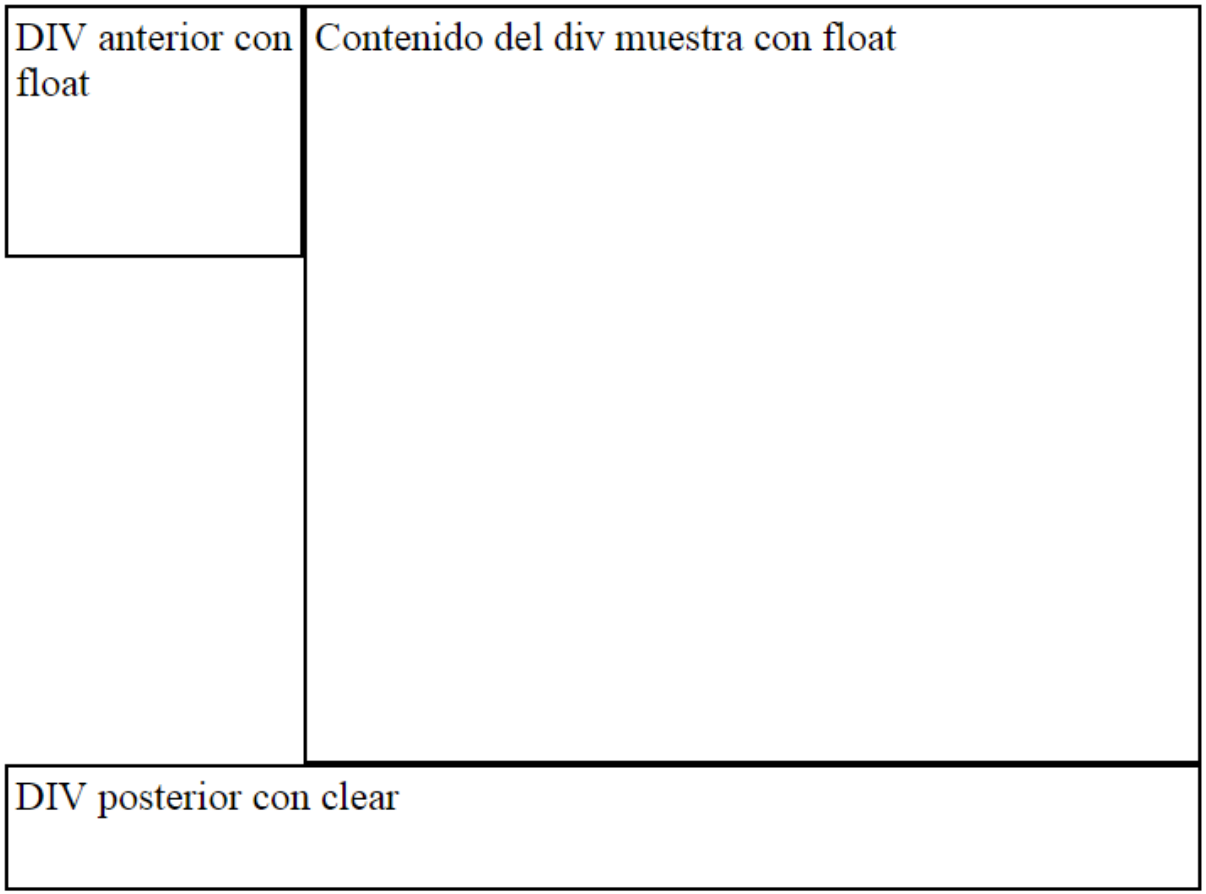


CSS

```
.muestra {
        height: 300px;
        width: 400px;
        border:solid 1px black;
}
.anterior {
        height: 100px;
        width: 200px;
        float: left;
        border:solid 1px black;
}
.posterior {
        height: 100px;
        width: 200px;
        border:solid 1px black;
}
```

Estructura CSS - float y clear

CSS3 permite posicionar los **div** en la página, **float** y **clear**
Con **clear** rompe el esquema del float.



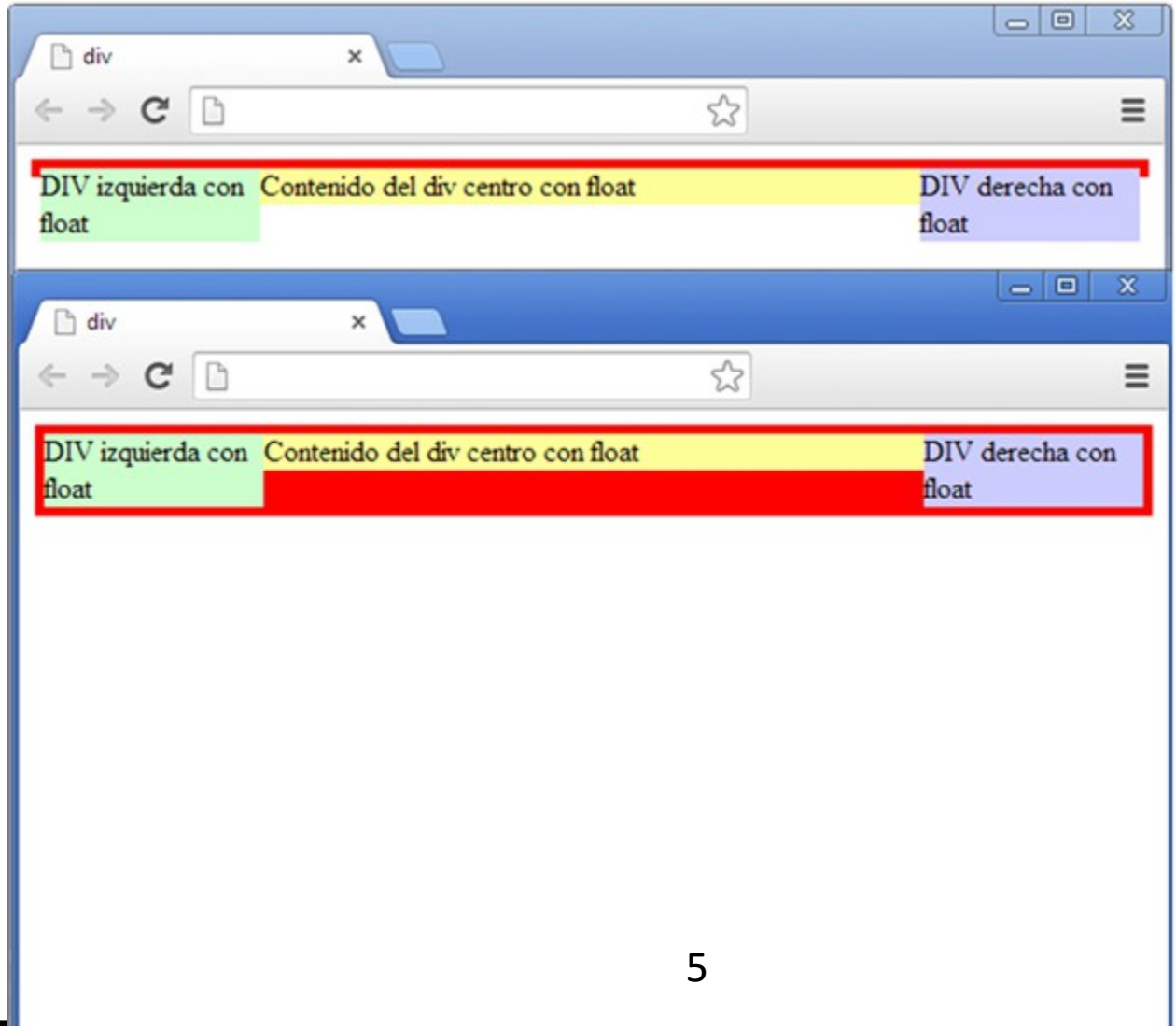
CSS

```
div {
        border:solid 1px black;
        box-sizing:border-box;
        padding:0.2em;
}
.muestra {
        height: 300px;
        width: 60%;
        float: left;
}
.anterior {
        height: 100px;
        width: 20%;
        float: left;
}
.posterior {
        height: 50px;
        width: 80%;
        clear: both;
}
```

Overflow:hidden

Uso de **float** y **overflow:hidden** para que la caja contenedora conozca la altura de sus bloques internos.

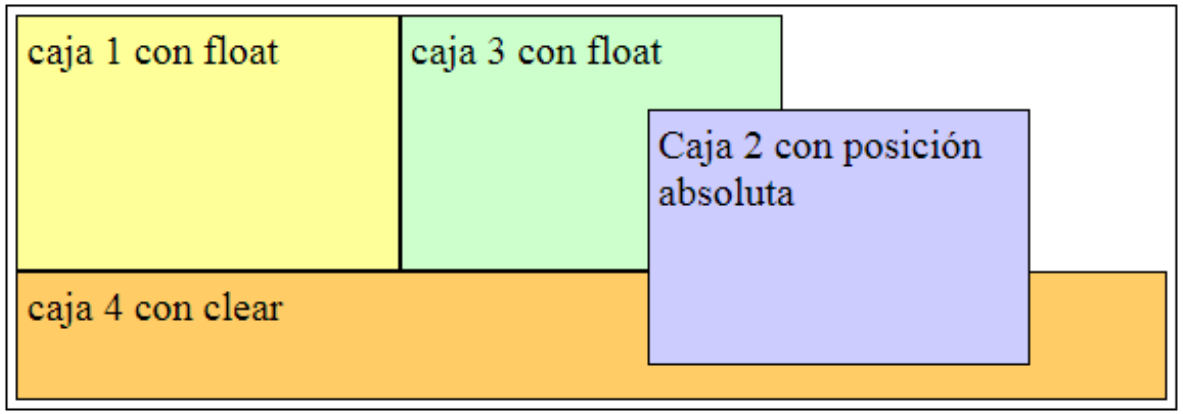
```
.caja{
        background-color: #F00;
                padding: 5px;
        overflow: hidden;
}
.caja .centro {
        width: 60%;
                background-color: #FF9;
        float: left;
}
.caja .izq {
        width: 20%;
        float: left;
        background-color: #CFC;
}
.caja .dcha {
        width: 20%;
        background-color: #CCF;
        float: left;
}
```



Estructura CSS - position

CSS3 permite posicionar los **div** en la página, mediante position, absoluta, relativa o fija.
Con posición absoluta el div sale del flujo del contenido.

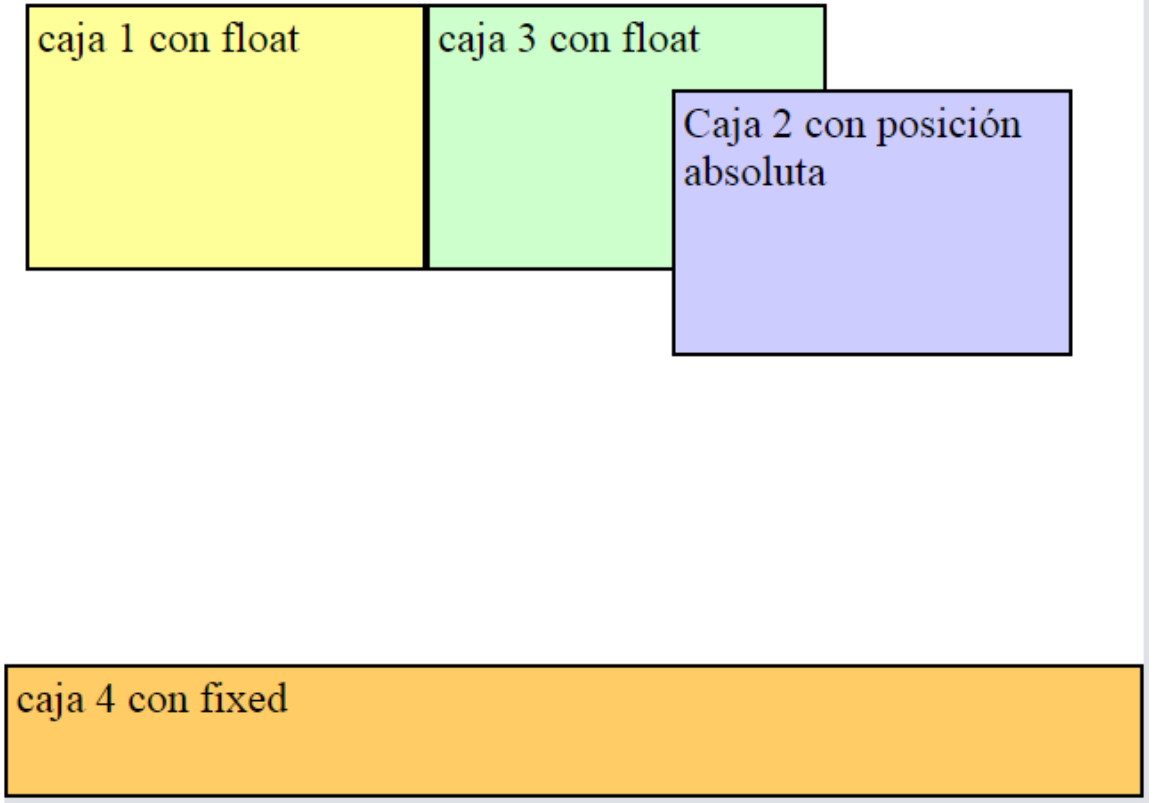
```
.caja {position: absolute;
        left: 100px;
        top: 100px;    }
.caja1 {       height: 100px;
        width: 150px;
        background-color: #FF9;
        float: left;    }
.caja2 {       height: 100px;
        width: 150px;
        background-color: #CCF;
        position: absolute;
        left: 250px;
        top: 40px;    }
.caja3 {       height: 100px;
        width: 150px;
        float: left;
        background-color: #CFC;    }
.caja4 {       height: 50px;
        width: 450px;
        background-color: #FC6;
        clear: both;    }
```



Estructura CSS - position

Con posición **fija** el **div** no cambia de posición en el escalado del navegador o el scroll en la página.

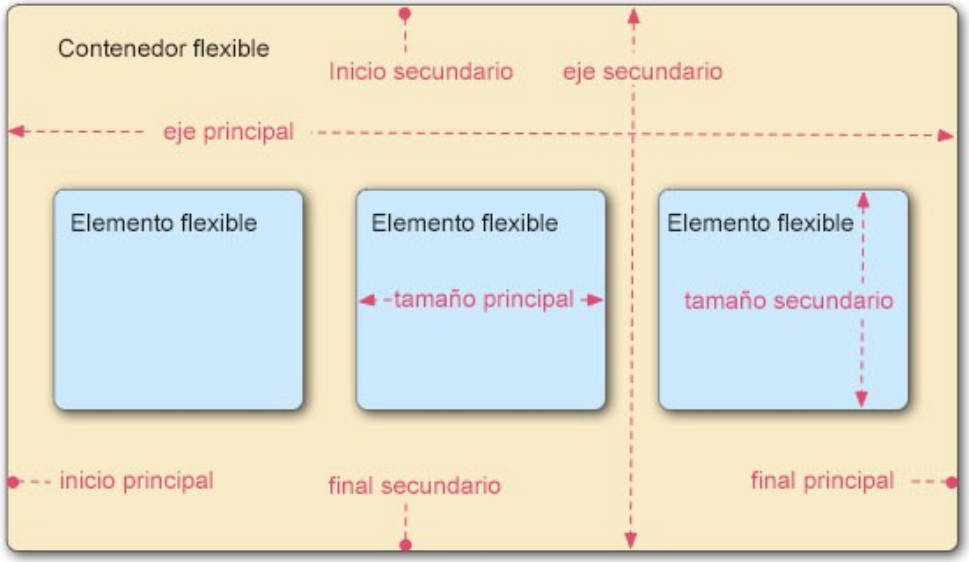
```
.caja1 {      height: 100px;
        width: 150px;
        background-color: #FF9;
        float: left; }
.caja2 {      height: 100px;
        width: 150px;
        background-color: #CCF;
        position: absolute;
        left: 250px;
        top: 40px; }
.caja3 {      height: 100px;
        width: 150px;
        float: left;
        background-color: #CFC;  }
.caja4 {      height: 50px;
        background-color: #FC6;
        clear: both;
        position: fixed;
        bottom: 0px;
        left: 0px;
        width: 100%;  }
```



Estructura CSS - FLEX

Las **cajas flexibles**, se consigue con un nuevo valor de la propiedad display, (display: flex;) de la caja padre.
La orientación se define con flex-direction y puede ser horizontal o vertical, según sea fila o columna.
Los elementos flexibles tienen diferentes formas de alinearse y distribuirse justify-content y align-items.

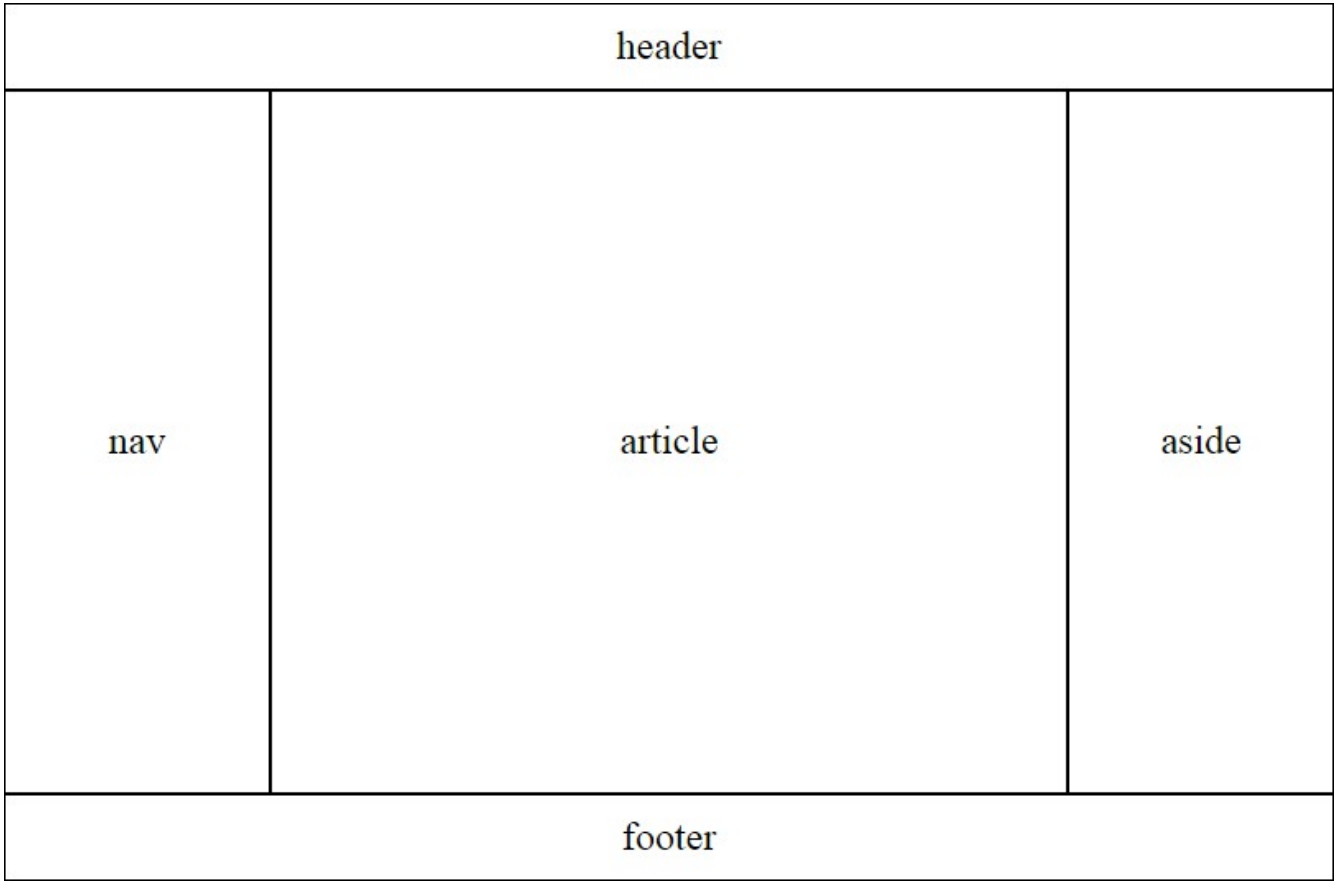
Cada uno de los elementos puede ordenarse o los diferentes modos crecer o Reducirse para ocupar el espacio.



Estructura CSS - FLEX

La propiedad de **cajas flexibles**, es una forma más sencilla de controlar el aspecto de las cajas div.

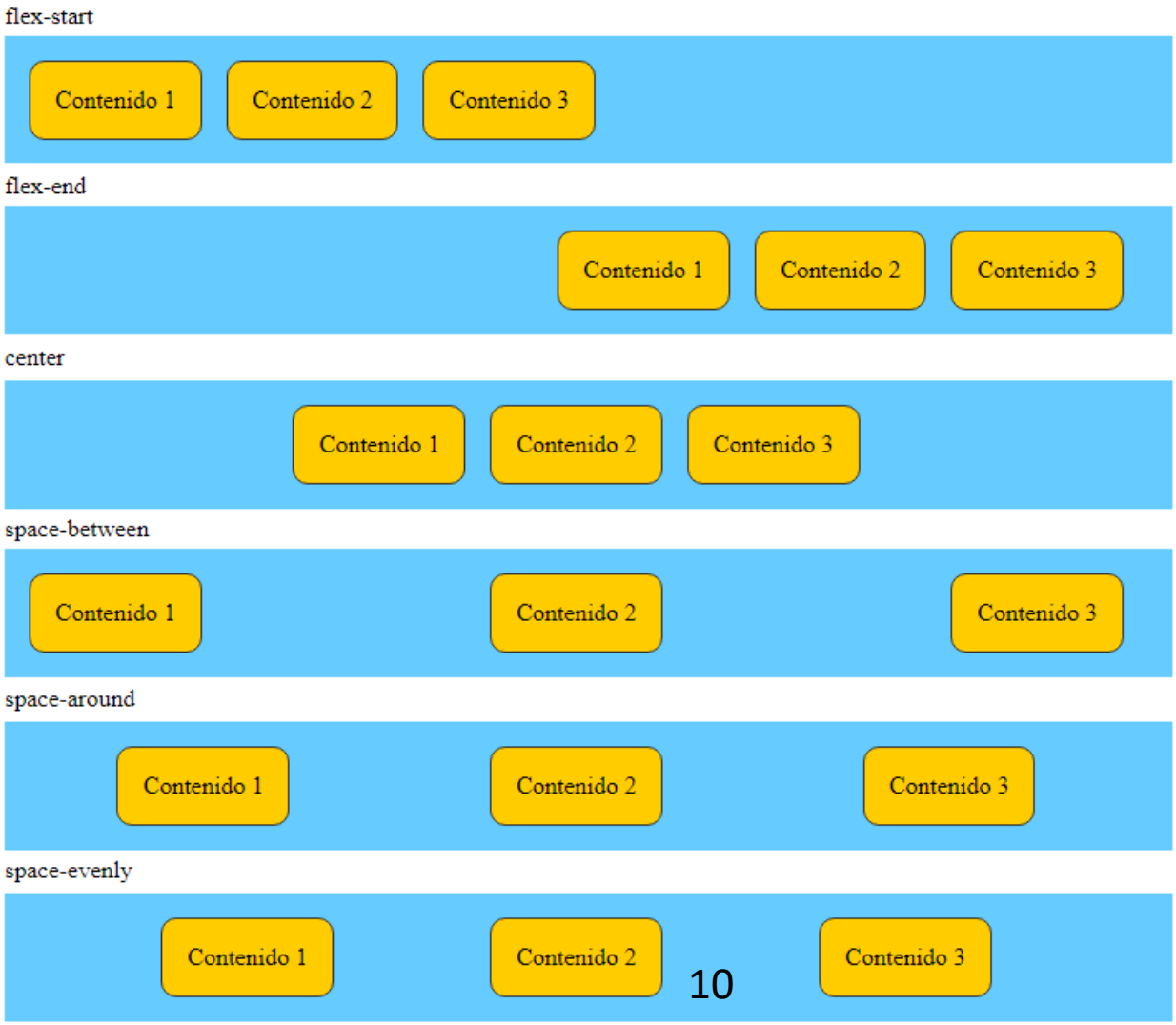
```
#main {
        height: 80%;
         display: flex;
         flex-flow: row;
}
article, nav, aside, header, footer  {
        display: flex;
        justify-content: center;
        align-items: center;
        font-size:1.5em;
        border:solid 1px black;
        box-sizing:border-box;
}
#main article  {
        flex: 3 1 60%;
        order: 2;
}
#main nav  {
        flex: 1 6 20%;
        order: 1;
}
#main aside  {
        flex: 1 6 20%;
        order: 3;
}
header, footer  {
        min-height: 10%;
}
```



Estructura CSS - FLEX – justify-content

Flex, facilita la alineación de los objetos en horizontal con **justify-content** y en vertical con **align-items**

```
.flex {  display: flex;
         background-color: #6CF;
         padding:1em;
}
.flex > div {     background-color: #FC0;
         border: 1px solid #333;
         margin-right:1em;
         padding:1em;
         border-radius:10px;
}
.start { justify-content: flex-start;
}
.end {  justify-content: flex-end;
}
.center {        justify-content: center;
}
.between {     justify-content: space-between;
}
.around {       justify-content: space-around;
}
.evenly {       justify-content: space-evenly;
}
```



Estructura CSS - FLEX – align-item

Flex, facilita la alineación de los objetos en horizontal con **justify-content** y en vertical con **align-items**

```
.flex {
        display: flex;
        background-color: #6CF;
        padding:1em;
}
.flex > div {
        background-color: #FC0;
        border: 1px solid #333;
        margin-right:1em;
        padding:1em;
        border-radius:10px;
}
img {   vertical-align:bottom;
}
.start { align-items: flex-start;
}
.end {  align-items: flex-end;
}
.center {       align-items: center;
}
.baseline {     align-items: baseline;
}
.stretch {      align-items: stretch;
}
```

Recursos CSS FLEX

MDN Usando las cajas flexibles CSS
https://developer.mozilla.org/es/docs/Web/Guide/CSS/Cajas_flexibles

CSS-TRICKS A Complete Guide to Flexbox
http://css-tricks.com/snippets/css/a-guide-to-flexbox/

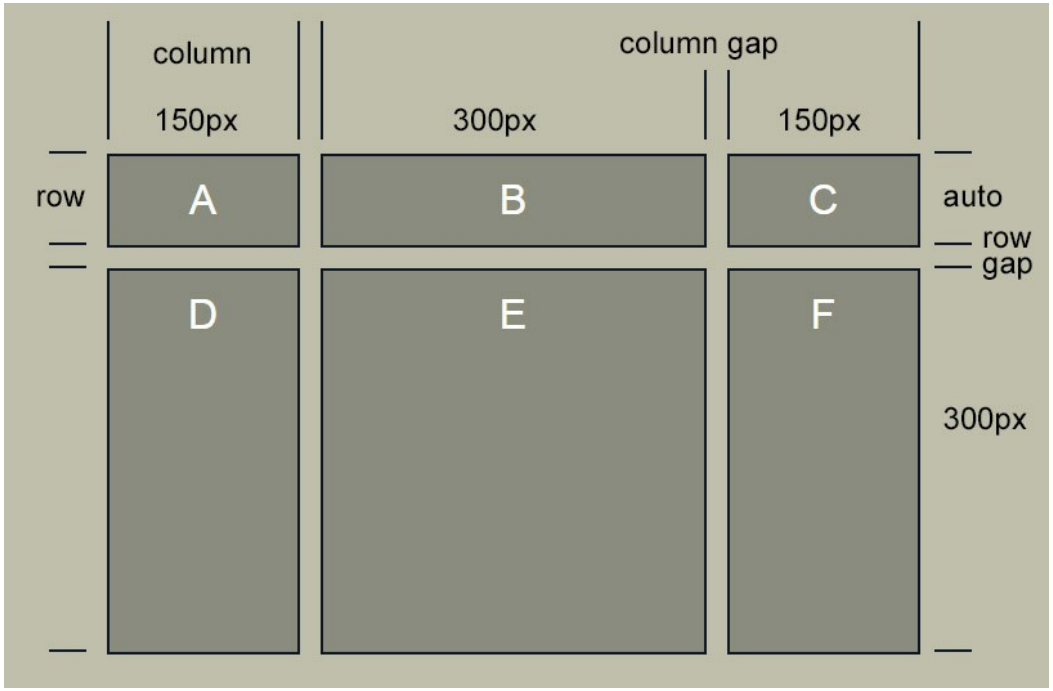
Flexbox Froggy Un juego para aprender CSS flexbox
http://flexboxfroggy.com/#es

Estructura CSS

La nueva etiqueta de display **Grid**, es el mejor sistema para estructurar la Web mediante la creación de rejillas de composición, muchos framework ya utilizan sistemas de grid, esta etiqueta CSS tiene importantes mejoras.

HTML

```
<div class="container">
  <div>A</div>
  <div>B</div>
  <div>C</div>
  <div>D</div>
  <div>E</div>
  <div>F</div>
</div>
```



CSS

```
.container {
  display: grid;
  grid-template-columns: 150px 300px 150px;
  grid-template-rows: auto 300px;
  grid-gap: 1rem;
}
```


e/s/d/ madrid

Estructura CSS

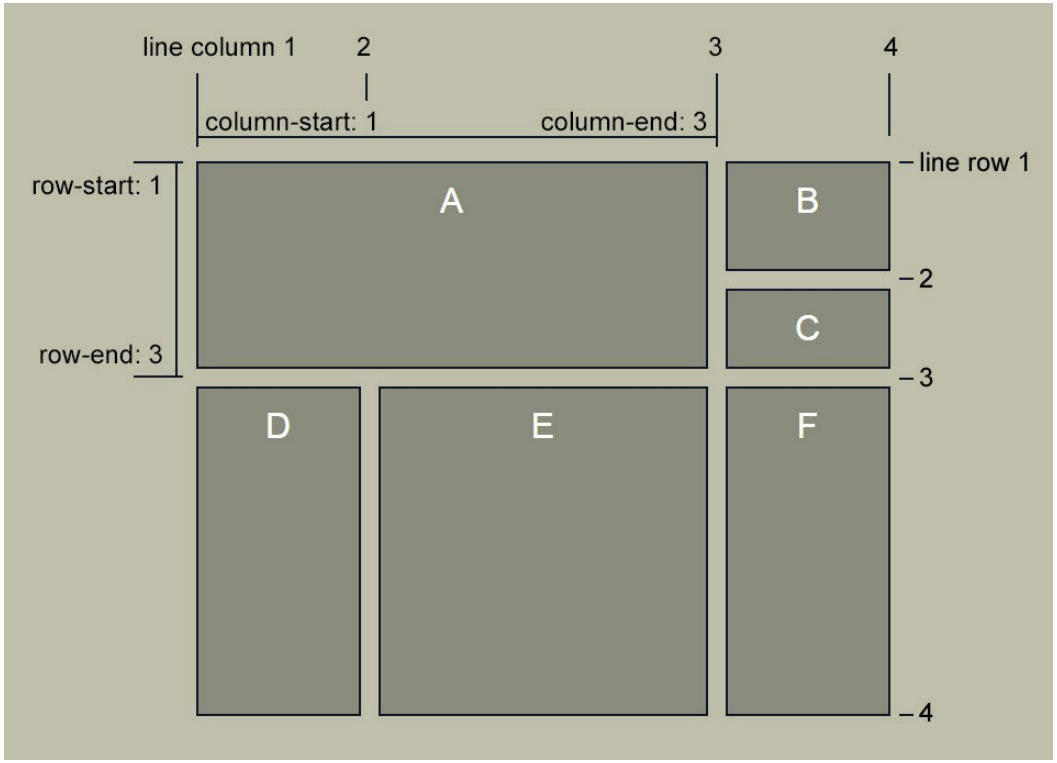
La línea es el separador horizontal (row) y vertical (column), se puede emplear para indicar el principio y final de una celda.

HTML

```
<div class="container">
  <div class="item1">A</div>
  <div>B</div>
  <div>C</div>
  <div>D</div>
  <div>E</div>
  <div>F</div>
</div>
```

CSS

```
.container {
display: grid;
grid-template-columns: 150px 300px 150px;
grid-template-rows: 100px auto 300px;
grid-gap: 1rem;
}
.item1 {
grid-row-start: 1;
grid-row-end: 3;
grid-column-start: 1;
grid-column-end: 3;
}
```



Estructura CSS

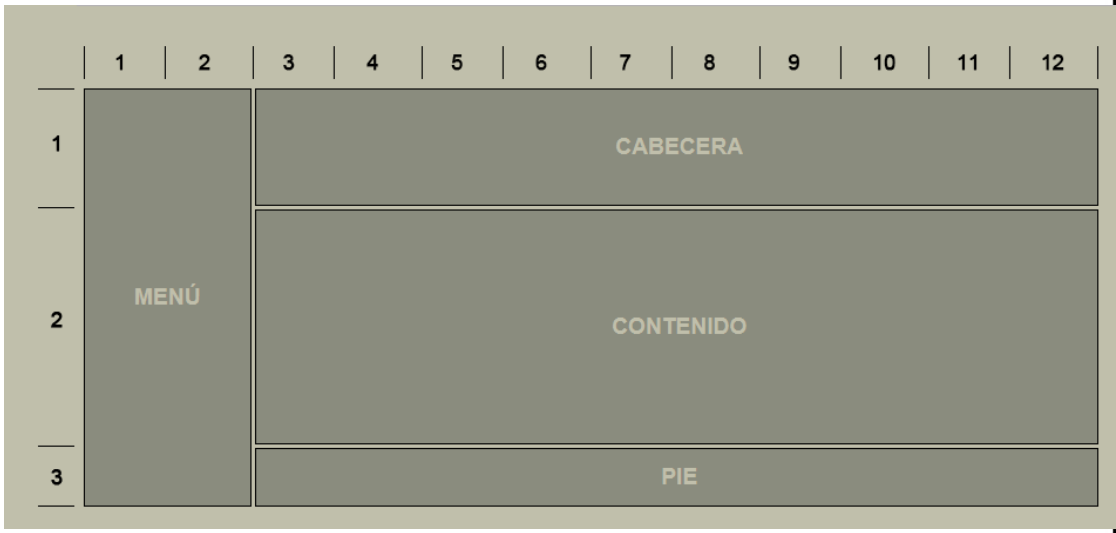
Podemos trabajar con una cuadrícula y asociar cada zona a un área, que puede abarcar varias celdas.

HTML

```
<div class="container">
 <div class="header">CABECERA</div>
 <div class="menu">MENÚ</div>
 <div class="content">CONTENIDO</div>
 <div class="footer">PIE</div>
</div>
```

CSS

```
.container {
   display: grid;
   grid-gap: 3px;
   grid-template-columns: repeat(12, 1fr);
   grid-template-rows: 100px 200px 50px;
   grid-template-areas:
      "m m h h h h h h h h h h"
      "m m c c c c c c c c c c"
      "m m f f f f f f f f f f";
}
 .header { grid-area: h;}
.menu { grid-area: m;}
.content { grid-area: c;}
.footer { grid-area: f;}
```



e/s/d/ madrid

Estructura CSS

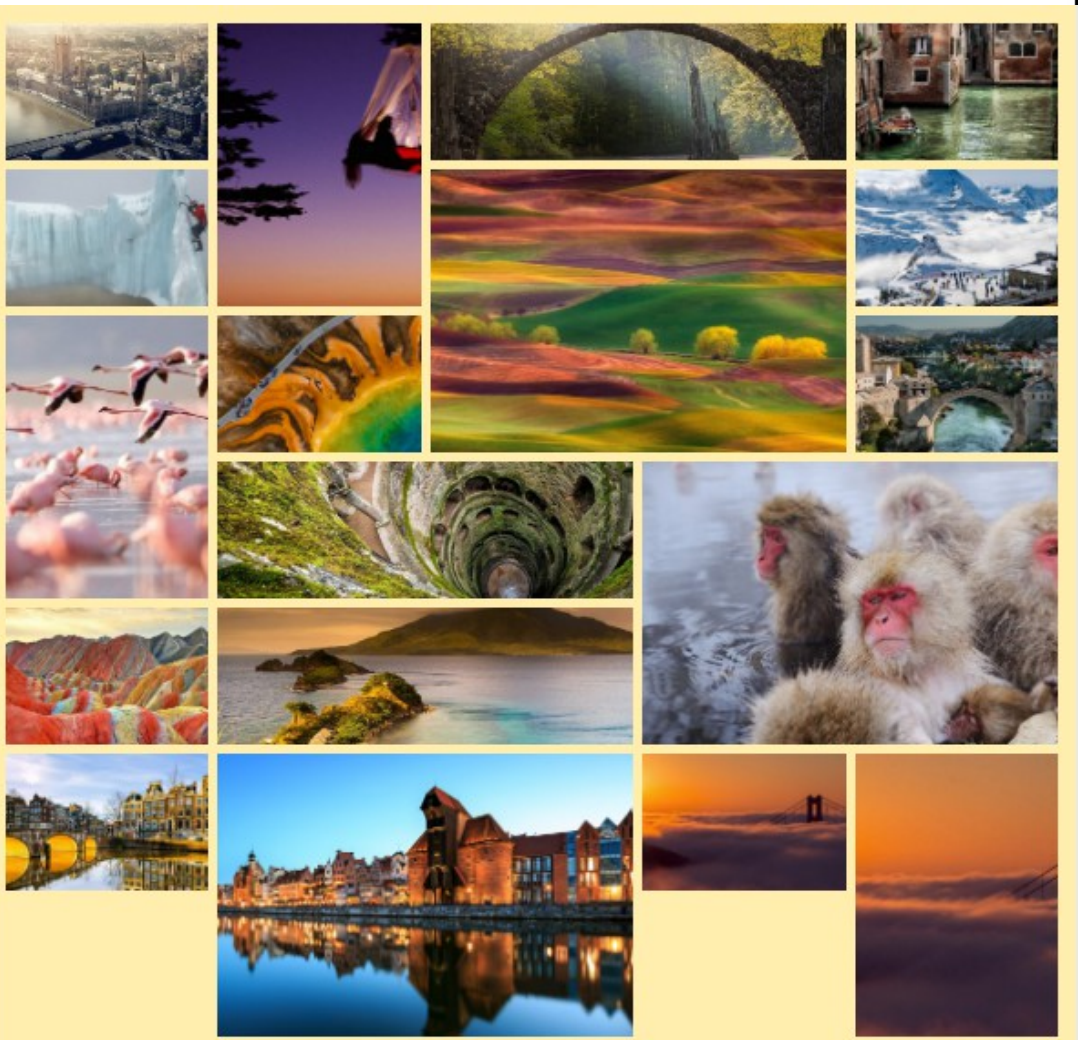
grid-auto-flow, controla la posición automática de las cajas contenidas en un grid.

HTML

```
<div class="container">
<div><img src="img… /></div><div
class="vertical"><img … /></div>
<div class="horizontal"><img… /></div>
<div class="big"><img … /></div>
… …
</div>
```

CSS

```
.container {
display: grid;
grid-gap: 5px;
grid-template-columns: repeat(auto-fit,
minmax(100px, 1fr));
grid-auto-rows: 75px;
grid-auto-flow: dense;
}
.horizontal { grid-column: span 2;  }
.vertical { grid-row: span 2; }
.big {
    grid-column: span 2;
    grid-row: span 2;
}
```



Recursos CSS GRID

CSS-TRICKS A Complete Guide to Grid
https://css-tricks.com/snippets/css/complete-guide-grid/

M | Pavel Laptev: Learning CSS grid layout with the Swiss
https://medium.com/@PavelLaptev/learning-css-grid-with-the-swiss-2

CSS-TRICKS: Auto-Sizing Columns in CSS Grid: `auto-fill` vs `auto-fit`
https://css-tricks.com/auto-sizing-columns-css-grid-auto-fill-vs-auto-fit

GRID GARDEN Un juego para aprender CSS Grid
https://cssgridgarden.com/#es

CSS

Estructura - CSS

e/s/d/
madrid