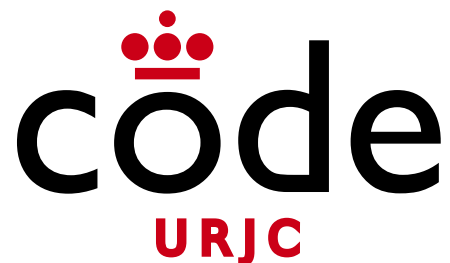


Fundamentos de la Web

Bloque II: Tecnologías de cliente web

Tema 2.2: CSS



©2024

Micael Gallego, Francisco Gortázar, Michel Maes, Óscar Soto, Iván Chicano

Algunos derechos reservados

Este documento se distribuye bajo la licencia
“Atribución-CompartirIgual 4.0 Internacional”
de Creative Commons Disponible en
<https://creativecommons.org/licenses/by-sa/4.0/deed.es>

- Hojas de estilo en cascada
- *Cascading Style Sheets* (CSS)
- Es un lenguaje utilizado para dar **estilo** a contenido estructurado
- Se aplica principalmente a documentos **HTML**
- Se puede usar con otros documentos como **SVG, XML**, etc..

- Con **CSS** se pueden especificar
 - **Colores:** principal, de fondo, degradados...
 - **Tipografía:** familia, tamaños, estilos...
 - **Layout:** Disposición de los elementos en el documento
 - **Efectos:** sombras, esquinas redondeadas...
 - **Animaciones:** en el cambio de los estilos

- **Separación de Presentación y Contenido**
 - Aunque se puede indicar el estilo incluido en el elemento HTML (atributo `style`), es mejor separar el estilo del contenido
 - **Ventajas:**
 - Esto facilita la coordinación entre diseñadores/maquetadores y programadores
 - Se pueden usar distintos CSSs en función del tipo/tamaño del dispositivo
 - La página es más accesible porque el contenido está separado de la presentación del mismo

- **Uso de CSS para dar estilo a un HTML**
 - **A)** Incluido en el elemento (atributo style)
 - **B)** Incluido en el HTML (en el HEAD)
 - **C)** En un fichero independiente (.css)

- Uso de CSS para dar estilo a un HTML
 - A) Incluido en el elemento (atributo style)

```
<!DOCTYPE html>
<html>
<head>
    <title></title>
</head>
<body>
    <p style="color:red">Red font!</p>
</body>
</html>
```

Red font!

- Uso de CSS para dar estilo a un HTML
 - B) Incluido en el HTML (en el HEAD)

```
<!DOCTYPE html>
<html>
<head>
  <style>
    p {
      color: purple;
    }
  </style>
  <title>Result</title>
</head>
<body>
  <p>I'm purple!</p>
</body>
</html>
```

I'm purple!

- Uso de CSS para dar estilo a un HTML
 - C) En un fichero independiente (.css)

```
<!DOCTYPE html>
<html>
<head>
  <title>Result</title>
  <link type="text/css" rel="stylesheet" href="style.css"/>
</head>
<body>
  <p>I'm blue</p>
</body>
</html>
```

style.css

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

I'm blue!

- **Uso de CSS para dar estilo a un HTML**
 - **C) En un fichero independiente (.css)**
 - Es la forma **recomendada**
 - Un mismo fichero .css se puede aplicar a **páginas HTML diferentes**
 - Ese fichero puede estar **cacheado** por el navegador para que no se **descargue** repetidamente
 - Se pueden incluir **varios ficheros css** asociados al mismo HTML


- Un documento CSS tiene el siguiente formato:

Selector

Elemento al que aplicar el estilo identificado por las propiedades

Propiedad

Aspecto del estilo de un elemento. Fuente, color, tamaño...



```
selector {  
    property: value;  
    property2: value;  
    property3: value;  
}  
selector2 {  
    property1: value;  
}
```

```
<h3>What's CSS for?</h3>
<p>CSS is for styling HTML pages!</p>
<h3>Why use it?</h3>
<p>It makes webpages look <span>really rad</span>.</p>
<h3>What do I think of it?</h3>
<p>It's awesome!</p>
```

```
p {
  font-family: Arial;
  color: blue;
  font-size: 12px;
}
h3 {
  color: red;
}
span {
  background-color: yellow;
}
```

What's CSS for?

CSS is for styling HTML pages!

Why use it?

It makes webpages look really rad.

What do I think of it?

It's awesome!

- **Hojas de estilos en “cascada”**
 - Se dice que son en **cascada** porque a los elementos se les aplican algunas de las propiedades del elemento **padre** (las heredan)
 - Si no se quieren heredar, se puede indicar un estilo particular para dicho elemento
 - Para saber si la propiedad se hereda o no hay que consultar la documentación

- Hojas de estilos en “cascada”
 - Como un **estilo está compuesto** de varias **propiedades** (color, tamaño, tipo de letra...), es posible que se declaren algunas nuevas pero otras se sigan “**heredando**” del elemento **padre**
 - En el ejemplo anterior, el elemento `` “hereda” el color azul y demás propiedades del elemento `<p>`

- Cursos online gratuitos sobre tecnologías web
 - <http://www.codecademy.com>
 - <https://lenguajecss.com/css/>



































*Algunos de los ejemplos de este material están basados en Codecademy

- Existen varias formas de especificar el valor de la propiedad **color** en CSS:
 - Utilizando el **nombre del color**
 - Utilizando sus componentes **RGB en formato hexadecimal**
 - Utilizando sus componente en **RGB o RGBA**

Colores

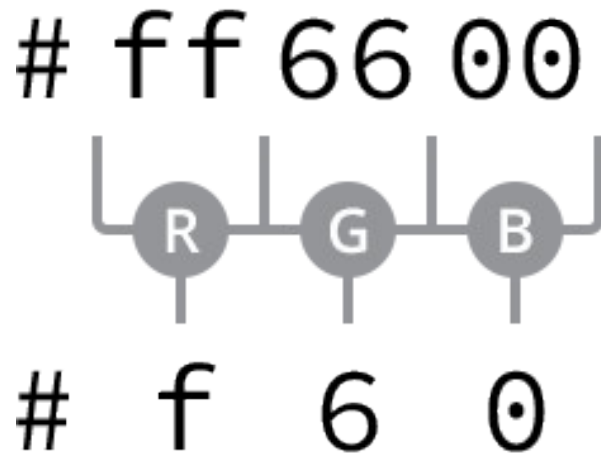
- Nombre

```
selector {  
  color: red;  
}
```

Named	Numeric	Color name	Hex rgb	Decimal
		<i>black</i>	#000000	0,0,0
		<i>silver</i>	#C0C0C0	192,192,192
		<i>gray</i>	#808080	128,128,128
		<i>white</i>	#FFFFFF	255,255,255
		<i>maroon</i>	#800000	128,0,0
		<i>red</i>	#FF0000	255,0,0
		<i>purple</i>	#800080	128,0,128
		<i>fuchsia</i>	#FF00FF	255,0,255
		<i>green</i>	#008000	0,128,0
		<i>lime</i>	#00FF00	0,255,0
		<i>olive</i>	#808000	128,128,0
		<i>yellow</i>	#FFFF00	255,255,0
		<i>navy</i>	#000080	0,0,128
		<i>blue</i>	#0000FF	0,0,255
		<i>teal</i>	#008080	0,128,128
		<i>aqua</i>	#00FFFF	0,255,255

- Valor hexadecimal

```
em { color: #f00; } /* #rgb */  
em { color: #ff0000; } /* #rrggbb */
```



- Valor rgb/rgba

```
em{ color: rgb(255,0,0) }  
em{ color: rgb(100%,0%,0%) }  
em{ color: rgba(255,0,0,1) } /* 1 = SOLID */  
em{ color: rgba(100%,0%,0%,0.5) } /* 50% transparent */  
em{ color: rgba(255,0,0,0.1) } /* 90% transparent */
```

- Generadores de paletas de colores
 - <http://paletton.com/>
 - <https://color.adobe.com/es/create/color-wheel>
 - Integrado en editores de código
- Más información
 - https://devdocs.io/css/color_value

Tamaño de texto

Tamaño absoluto (px)

```
div{ width: 100px; }  
.title{ height: 50px; }
```

Tamaño relativo a su contenedor (%)

```
div{ width: 100%; }  
.title{ height: 50%; }
```

Tamaño de texto

Tamaño absoluto (para impresión) (pt)

```
p{ font-size: 16pt;}  
.title{ font-size: 18pt;}
```

Tamaño relativo a la fuente del elemento (em)

```
p{ font-size: 0.85em;}  
.title{ font-size: 1.5em;}
```

- En general es recomendable usar **medidas relativas** en el tamaño del texto (**font-size**)
- Esto permite adaptar la página a diferentes dispositivos y resoluciones

<https://devdocs.io/css/length>

- El tipo de letra (**font-family**) que se puede asociar a un texto depende de los tipos de letra del **sistema** o los que se pueden descargar de **Internet**
- Como no se puede saber qué tipos de letra están instalados o se pueden descargar, hay que indicar una **lista de tipos de letra**
- Es recomendable que el último sea un tipo de letra genérico que siempre existe: **serif, sans-serif, monospace, cursive, fantasy**

<https://devdocs.io/css/font-family>

Tipo de letra

```
font-family: Times, "Times New Roman", Georgia, serif;  
font-family: Verdana, Arial, Helvetica, sans-serif;  
font-family: "Lucida Console", Courier, monospace;  
font-family: cursive;  
font-family: fantasy;
```

This is an example of a serif font.

This is an example of a sans-serif font.

This is an example of a monospace font.

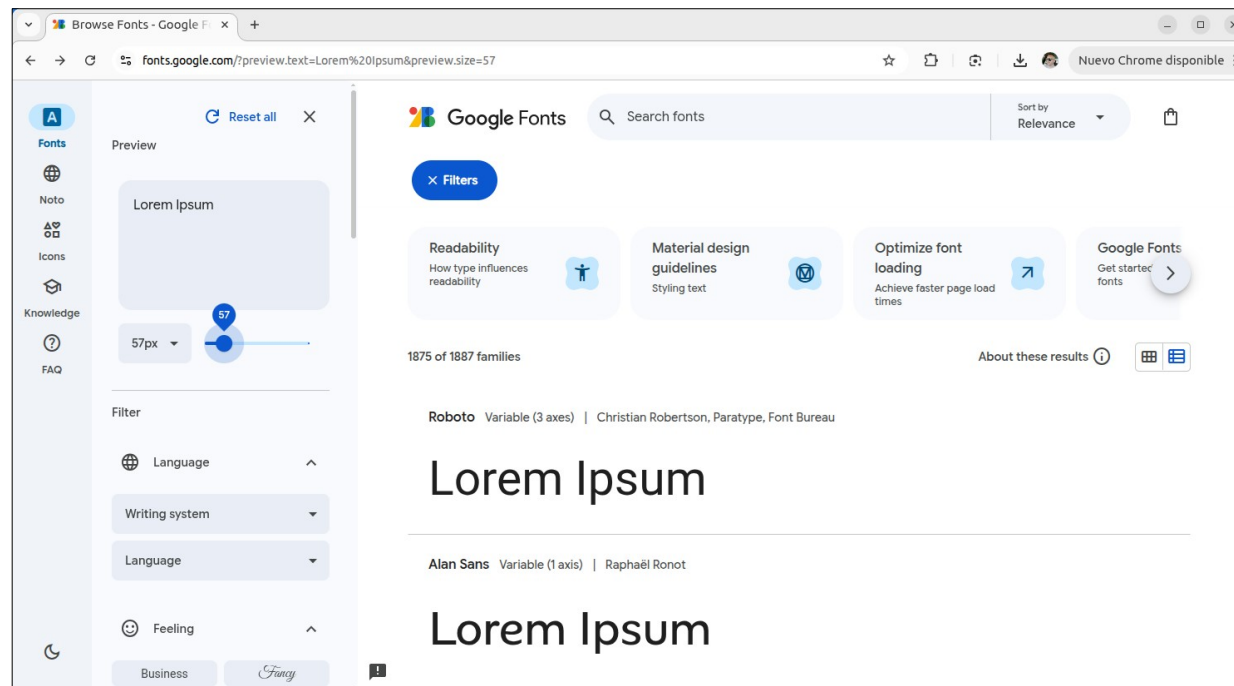
This is an example of a cursive font.

This is an example of a fantasy font.

<https://devdocs.io/css/font-family>

Tipo de letra

- **Google Fonts** proporciona cientos de tipos de letra que se descargan al visitar una web



- Google Fonts

```
<head>
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?
    family=Roboto:ital,wght@0,100..900;1,100..900&display=swap"
    rel="stylesheet">
</head>
```

```
font-family: "Roboto", sans-serif;
font-optical-sizing: auto;
font-weight: 100;
font-style: normal;
Font-variation-settings: "wdth" 100;
```

Normal 400

Grumpy wizards make toxic brew for the evil Queen and Jack.

Otros estilos de texto

- Grosor (**font-weight**)

```
<p>Lorem ipsum...</p>  
<p>Quisque...</p>  
<p>Aliquam eget...</p>
```

```
html{  
    font-weight: bold;  
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris elementum a dolor eu ultricies. Sed aliquet elit justo, non consequat arcu iaculis eu.

Quisque imperdiet tempor arcu. Pellentesque in luctus urna. Phasellus metus tellus, condimentum mollis tincidunt nec, feugiat in diam. Sed feugiat lacus eget odio imperdiet, at vehicula massa congue.

Aliquam eget ornare velit. Pellentesque fringilla convallis odio et pellentesque. Donec id viverra velit.

http://www.w3schools.com/cssref/pr_font_weight.asp

- Interlineado (**line-height**)

```
<p>Lorem ipsum...</p>  
<p>Quisque...</p>  
<p>Aliquam eget...</p>
```

```
html{  
  line-height: 20pt;  
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Mauris elementum a dolor eu ultricies. Sed aliquet elit justo, non consequat arcu iaculis eu.

Quisque imperdiet tempor arcu. Pellentesque in luctus urna. Phasellus metus tellus, condimentum mollis tincidunt nec, feugiat in diam. Sed feugiat lacus eget odio imperdiet, at vehicula massa congue.

Aliquam eget ornare velit. Pellentesque fringilla convallis odio et pellentesque. Donec id viverra velit.

- Alineación (**text-align**)

```
<p>Lorem ipsum...</p>  
<p>Quisque...</p>  
<p>Aliquam eget...</p>
```

```
html{  
  text-align:center;  
}
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Mauris elementum a dolor eu ultricies. Sed aliquet elit
justo, non consequat arcu iaculis eu.

Quisque imperdiet tempor arcu. Pellentesque in luctus
urna. Phasellus metus tellus, condimentum mollis
tincidunt nec, feugiat in diam. Sed feugiat lacus eget
odio imperdiet, at vehicula massa congue.

Aliquam eget ornare velit. Pellentesque fringilla
convallis odio et pellentesque. Donec id viverra velit.

Otros estilos de texto

- Otras propiedades interesantes
 - **text-decoration**
 - **text-shadow**
 - **text-transform**
 - **letter-spacing**
 - **word-spacing**

- Hay muchos elementos en una página HTML que pueden tener bordes
- El borde se configura con la propiedad **border**
- Esta propiedad está formada por 3 elementos:
 - **Ancho del borde:** px, em...
 - **Estilo:** none, hidden, dotted, solid, dashed...
 - **Color**

```
table {  
    border: 1px solid black;  
}
```

<https://devdocs.io/css/border>

Ejemplo: botón básico de CSS

- Para experimentar con todo lo aprendido vamos a hacer un botón usando **CSS**
- El botón se define en **HTML** con un elemento `<div></div>`



Ejemplo: botón básico de CSS

- El botón se define en **HTML** con un elemento `<div></div>`
- Las propiedades CSS del div serán:
 - `height: 50px`
 - `width: 120px`
 - `border-color: #6495ED`
 - `background-color: #BCD2EE`
 - `border-width: 2px`

Ejemplo: botón básico de CSS

- Para que los bordes del botón sean **redondeados**, se usará la propiedad:
 - `border-radius: 5px;`
- **Centramos** el botón en la **página**. Para ello, ponemos el mismo margen a ambos lados:
 - `margin: auto;`
- **Centramos el contenido** del botón con:
 - `text-align: center;`

Ejemplo: botón básico de CSS

- Se incluye un link dentro del div para que sea realmente un botón

```
<div>  
  <a href="link">Friend us on <span>Facebook!</span></a>  
</div>
```

- Se da estilo a los textos:

```
a {  
  text-decoration: none;  
  color: #3D59AB;  
  font-family: Verdana, sans-serif;  
}
```

```
span {  
  font-weight: bold;  
  font-size: 18px;  
  color: #ffffff;  
}
```

Ejemplo: botón básico de CSS

button.css

```
div {
  height: 50px;
  width: 120px;
  border: 5px solid #6495ED;
  background-color: #BCD2EE;
  border-radius: 15px;
  margin: auto;
  text-align: center;
}

a {
  text-decoration: none;
  color: #3D59AB;
  font-family: Verdana, sans-serif;
}

span {
  font-weight: bold;
  font-size: 18px;
  color: #ffffff;
}
```

button.html

```
<!DOCTYPE html>
<html>

<head>
  <title>Facebook button</title>
  <link type="text/css" rel="stylesheet"
        href="button.css" />
</head>

<body>
  <div>
    <a href="link">Friend us on
      <span>Facebook!</span>
    </a>
  </div>
</body>

</html>
```

Ejercicio 1

- Crea una página web con HTML y CSS con el siguiente botón



- Los **selectores** sirven para seleccionar los elementos a los que se les dará el estilo en CSS
- Cuando se usa como selector el nombre de un **elemento**, esas propiedades se aplicarán a todos los elementos de ese tipo en la página

El estilo se aplicará a todos los elementos **a** (links) de la página



```
a {  
    color: #3D59AB;  
}
```

- Existen otras formas de seleccionar elementos

Elementos anidados

- Elementos que están dentro de otros elementos

```
<div>
  <div>
    <p>I like tacos!</p>
  </div>
</div>
```

El estilo se aplicará a todos los elementos **p** que estén dentro de un **div** que a su vez esté dentro de otro **div**



```
div > div > p {
  margin: 10px;
}
```


Elementos anidados

- Podemos seleccionar elementos dentro de otros elementos pero permitiendo que haya elementos entre medias
- Quitamos el símbolo ">"

```
<div>  
  <ul>  
    <li>Leche</li>  
    <li>Galletas</li>  
  </ul>  
</div>
```

El estilo se aplicará a todos los elementos **li** que estén dentro de un **div** aunque no sea directamente

```
div li {  
  margin: 10px;  
}
```

Selector comodín

- Selector `*` para seleccionar cualquier elemento


```
*{  
  border: 2px solid black;  
}
```

- Es útil cuando se quiere dar un estilo a elementos concretos (usando el nombre del elemento como selector) y se quiere configurar un **estilo para todos los demás**

Selector comodín

- Usar el selector comodín * para elementos dentro de otros elementos

Se aplica a
todos los
elementos
dentro de un
párrafo



```
p * {  
    margin: 10px;  
}
```

- ¿Qué ocurre si a un mismo elemento le corresponden varias reglas CCS?

```
ul li p {  
  color: red;  
}
```

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

- Siempre se aplica el **selector más específico**

```
<p>La lista de la compra:</p>  
<ul>  
  <li><p>Leche</p></li>  
  <li><p>Galletas</p></li>  
  <li><p>Azúcar</p></li>  
</ul>
```

La lista de la compra:

- Leche
- Galletas
- Azúcar

Selectores de clase e ID

- Es habitual dar un **nombre a un elemento** concreto para poder darle estilo independientemente de su tipo y su posición en el documento HTML

```
<p id="anuncio_principal">El anuncio</p>
```

- También se puede crear una **clase de elementos**, de forma que a todos los elementos de esa clase (o tipo) se les aplicará el estilo

```
<p class="cita">Lo perfecto es... </p>
```

Selectores de clase e ID

- Definir el estilo del elemento con **id**

```
#anuncio_principal {  
    color: #FF00FF;  
    font-weight: bold;  
}
```

- Definir el estilo de los elementos de una **clase**

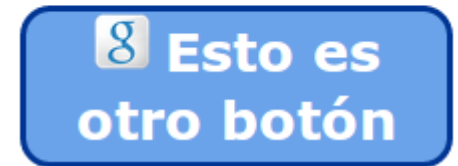
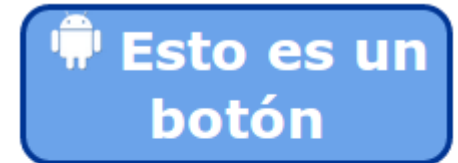
```
.cita {  
    color: red;  
}
```

Selectores de clase e ID

- Usando el atributo **class** podemos reutilizar los estilos de esa clase en varias partes de la página
- Asignaremos el **class** correspondiente al elemento que queramos que tenga el estilo

```
.button {  
    ...  
}  
.button a {  
    ...  
}  
.button img {  
    ...  
}
```

```
<div class="button">  
      
    <a href="">Un botón</a>  
</div>  
  
<div class="button">  
      
    <a href="">Otro botón</a>  
</div>
```



- El sistema de selectores permite seleccionar elementos del HTML
- También se pueden usar los selectores para aplicar estilos diferentes dependiendo del estado de un elemento (**pseudo-class**)

```
selector:pseudo-class_selector {  
    property: value;  
}
```


- Por ejemplo, se puede cambiar el estilo de los links cuando el usuario pasa encima de ellos con el ratón

```
a:hover {  
    color: #cc0000;  
    font-weight: bold;  
    text-decoration: none;  
}
```

- Otros pseudo-clases de link:
 - a:link: Enlace no visitado
 - a:visited: Enlace visitado

- Otras pseudo-clases interesantes:
 - `elem:first-child`: Se aplica el primer elemento hijo

```
p:first-child { color: red; }
```

- `elem:nth-child(n)`: Hijo n

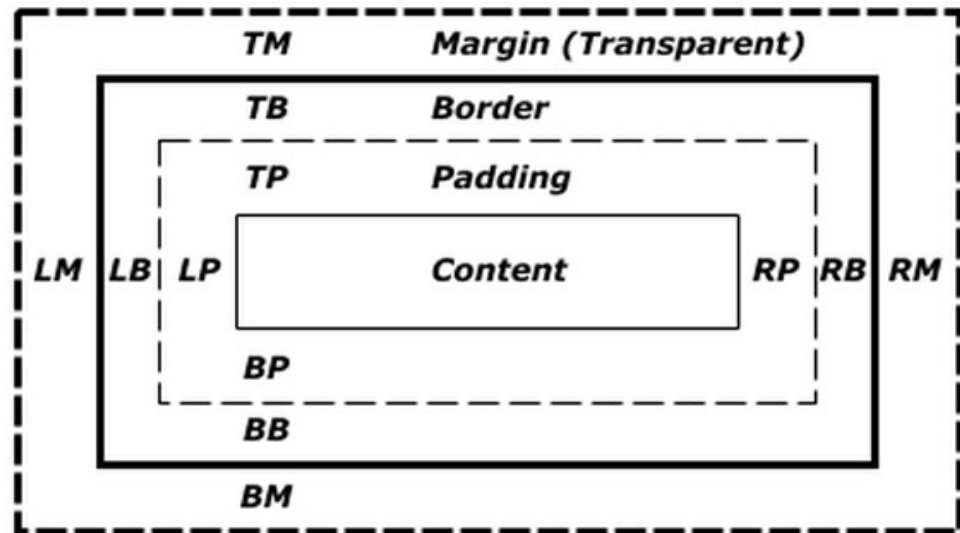
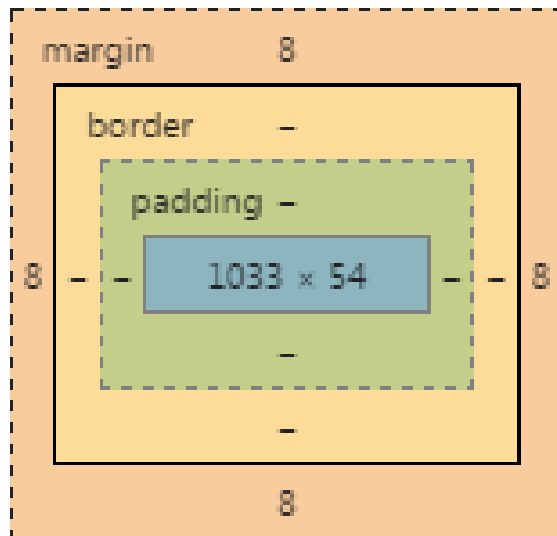
```
p:nth-child(2) { color: red; }
```

<https://developer.mozilla.org/en-US/docs/Web/CSS/Pseudo-classes>

Tamaño, margen y posición

- Por defecto, los elementos de una página se comportan como **palabras en un texto**
- Pero su tamaño, margen y posición es altamente configurable en base al **box model**
- Cada elemento está en una caja que tiene:
 - **Margen:** Espacio transparente alrededor del elemento
 - **Borde:** Línea que rodea el elemento
 - **Relleno (padding):** Relleno del elemento desde su contenido hasta el borde
 - **Contenido:** Información del propio elemento

Tamaño, margen y posición



- Margin edge
- Border edge
- - - Padding edge
- Content edge

Tamaño, margen y posición

- Por defecto, los elementos `<div>` ocupan todo el ancho de la página

```
#one { background-color: red }  
#two { background-color: blue }  
#three { background-color: yellow }  
#four { background-color: green }
```

```
<div id="one">A</div>  
<div id="two">B</div>  
<div id="three">C</div>  
<div id="four">D</div>
```

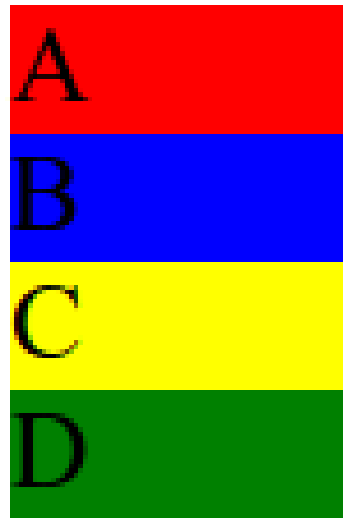


Tamaño, margen y posición

- Propiedad **display**:
 - **display: block**
 - No permite otros elementos en la misma línea (por defecto)
 - Se puede cambiar su alto y ancho
 - **display: inline-block**
 - Permite otros elementos en la misma línea
 - Se puede cambiar su alto y ancho
 - **display: inline**
 - Permite otros elementos en la misma línea
 - **NO** se puede cambiar su alto y ancho
 - **display: none**
 - Oculto

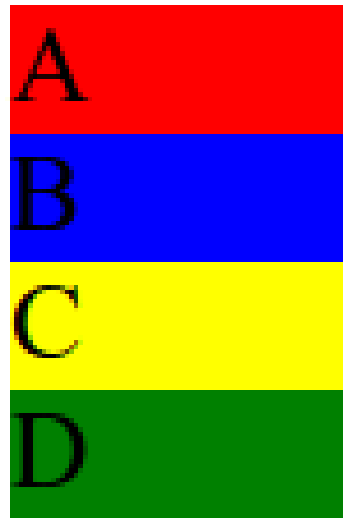
Tamaño, margen y posición

```
div {  
    width: 50px;  
}
```



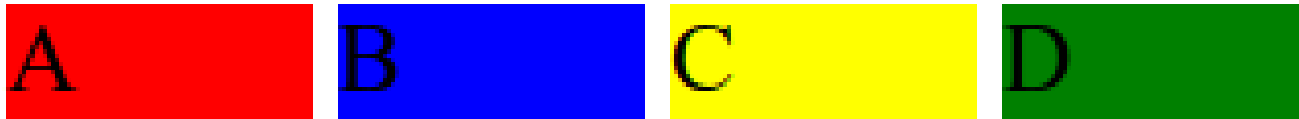
Tamaño, margen y posición

```
div {  
  width: 50px;  
  display: block;  
}
```



Tamaño, margen y posición

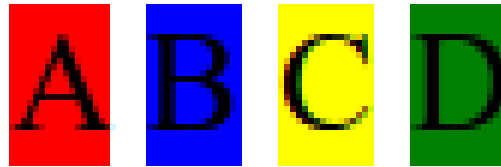
```
div {  
  width: 50px;  
  display: inline-block;  
}
```



Este espacio se puede
evitar con margin: -4px

Tamaño, margen y posición

```
div {  
  width: 50px;  
  display: inline;  
}
```



¿Y el none?

Este espacio se puede
evitar con margin: -2px

Tamaño, margen y posición

- Propiedad **margin**:
 - Controla el espacio alrededor de un elemento
 - **margin: auto**: Pone el mismo margen a la izquierda y a la derecha. Es decir, centra el elemento horizontalmente
 - **margin-top, margin-right, margin-bottom, margin-left**: Permiten cambiar el margen individualmente a cada lado
 - **margin: 1px 2px 3px 4px**: Forma compacta de cambiar todos los márgenes a la vez (top right bottom left)
 - **margin: 10px**: El mismo margen a cada lado

Tamaño, margen y posición

- Propiedad **border**:
 - Controla el borde del elemento
 - **border-width**: Ancho del borde
 - **border-color**: Color del borde
 - **border-style**: Estilo del borde: solid, dashed, dotted, groove...
 - **border: 4px solid #FF0000**: Forma compacta de cambiar todas las propiedades a la vez

Tamaño, margen y posición

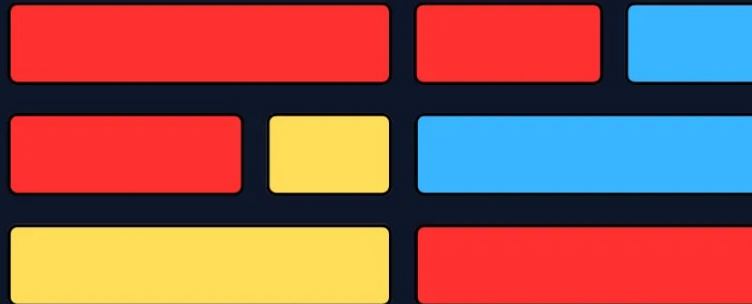
- Propiedad **padding**:
 - Rellena entre el contenido y el borde
 - **padding-top, padding-right, padding-bottom, padding-left**: Permiten cambiar el relleno individualmente a cada lado
 - **padding: 1px 2px 3px 4px**: Forma compacta de cambiar todos los rellenos a la vez
 - **padding: 10x**: Forma compacta de poner el mismo padding en todos los lados

- **Posición de los elementos**
 - Antes se usaban las propiedades **position** y **float**
 - Eran muy limitadas y complejas de usar
 - Se han creado dos nuevos sistemas de posicionamiento:
 - **Flexbox**
 - **Grid**

Tamaño, margen y posición

- Posición de los elementos
 - **Flexbox**
 - Elementos en una dimensión: Filas o columnas
 - Barras de navegación, alineación de botones..
 - **Grid**
 - Elementos en dos dimensiones: Filas y columnas
 - Layouts de páginas completos, Galerías de imágenes

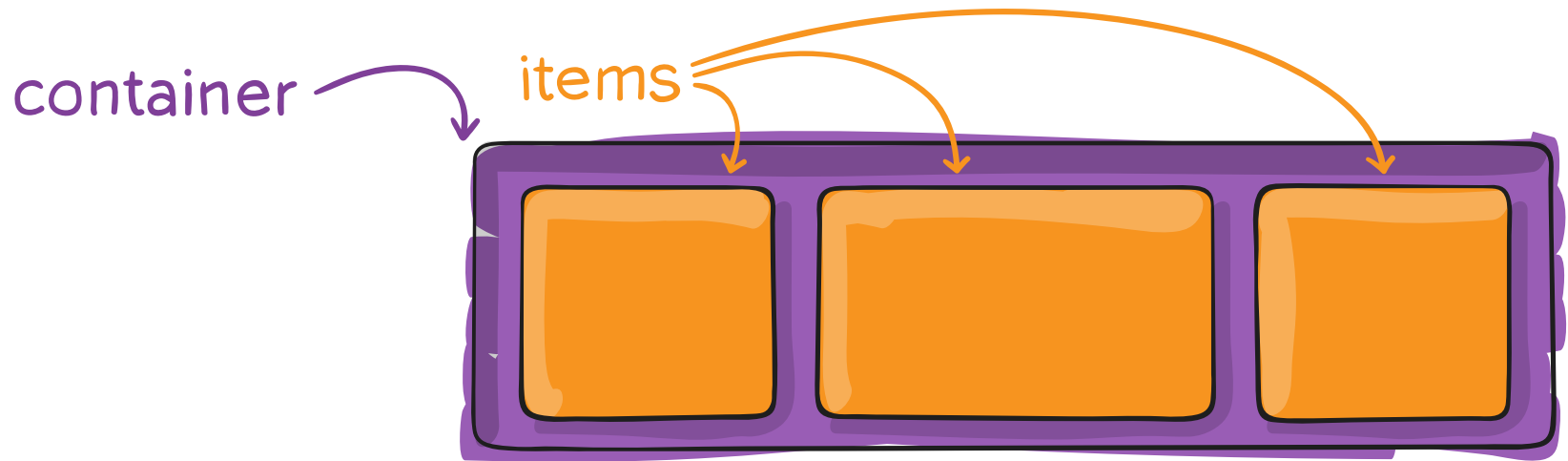
FLEXBOX



Material basado en

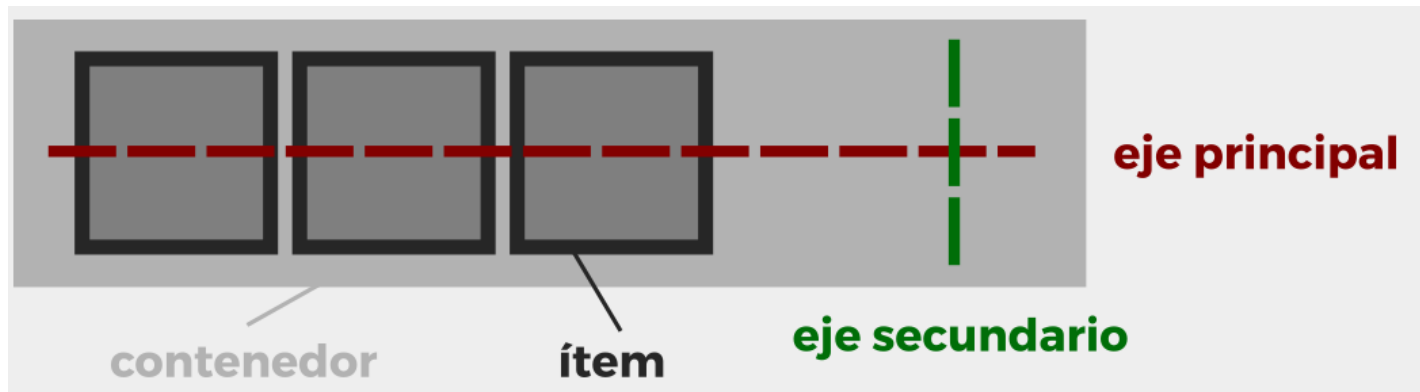
<https://css-tricks.com/snippets/css/a-guide-to-flexbox/>

Flexbox



- **Elementos**

- **Contenedor:** Es el elemento padre que tendrá en su interior cada uno de los ítems flexibles.
- **Eje principal:** Los contenedores flexibles tendrán una orientación principal específica. Por defecto, el eje principal del contenedor flexbox es en horizontal (en fila).



- **Eje secundario:** De la misma forma, los contenedores flexibles tendrán una orientación secundaria, perpendicular a la principal. Si la principal es en horizontal, la secundaria será en vertical (y viceversa).
- **Ítem:** Cada uno de los hijos que tendrá el contenedor en su interior.

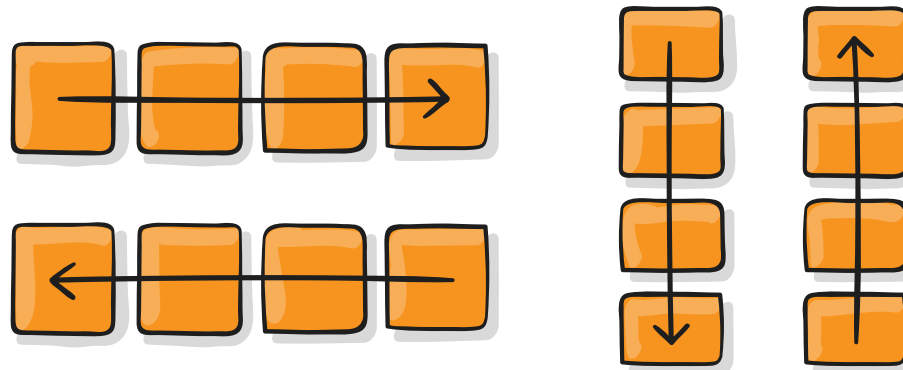
<https://lenguajecss.com/css/maquetacion-y-colocacion/flexbox/>

- Propiedad **display**
 - **inline-flex**: Establece un contenedor en línea (ocupa solo el contenido).
 - **flex**: Establece un contenedor en bloque (ocupa todo el ancho del padre).

```
.container {  
    display: flex; /* or inline-flex */  
}
```

flex-direction

- Establece el eje principal (horizontal o vertical) y la dirección

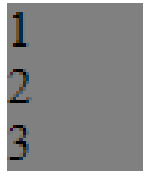


```
.container {  
  display: flex;  
  flex-direction: row | row-reverse | column | column-reverse;  
}
```

Flexbox

```
<div class="container"> <!-- Flex container -->
  <div class="item item-1">1</div> <!-- Flex items -->
  <div class="item item-2">2</div>
  <div class="item item-3">3</div>
</div>
```

```
.container {
  display: flex;
  flex-direction: column;
  background: steelblue;
}
.item {
  background: grey;
}
```



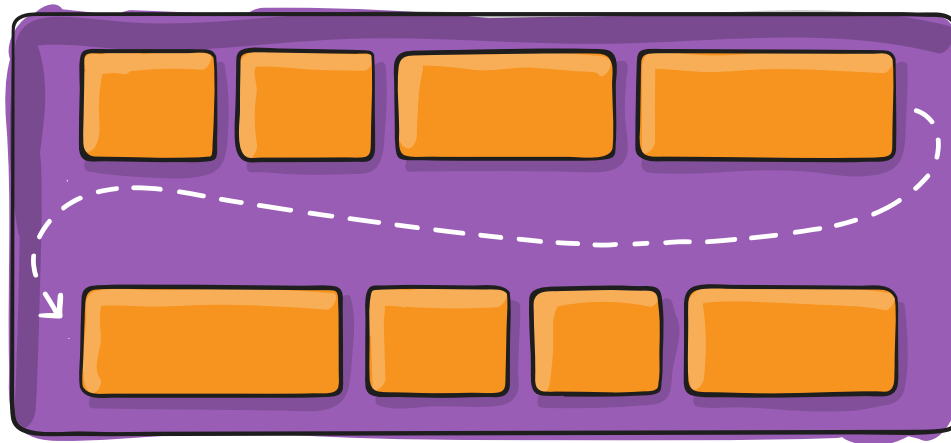
Flex-direction: column



Flex-direction: row

flex-wrap

- Evita o permite el desbordamiento (multilinea).



```
.container {  
  display: flex;  
  flex-wrap: nowrap | wrap | wrap-reverse;  
}
```

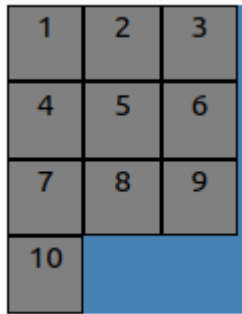
flex-wrap

- Evita o permite el desbordamiento (multilinea).
 - **nowrap**: Establece los ítems en una sola línea (no permite que se desborde el contenedor).
 - **wrap**: Establece los ítems en modo multilínea (permite que se desborde el contenedor).
 - **wrap-reverse**: Establece los ítems en modo multilínea, pero en dirección inversa.

Flexbox

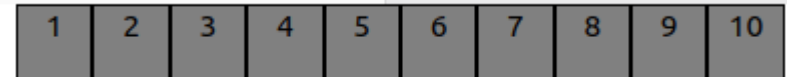
```
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
  <div class="item">6</div>
  <div class="item">7</div>
  <div class="item">8</div>
  <div class="item">9</div>
  <div class="item">10</div>
</div>
```

```
.container {
  display: flex;
  flex-wrap: wrap; //no-wrap
  background: steelblue;
  width: 100px;
}
.item {
  background: grey;
  min-height: 30px;
  min-width: 30px;
  border: 1px solid black;
  text-align: center;
}
```



1	2	3
4	5	6
7	8	9
10		

flex-wrap: wrap;



1	2	3	4	5	6	7	8	9	10
---	---	---	---	---	---	---	---	---	----

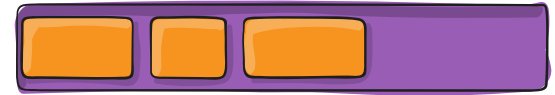
flex-wrap: nowrap;

justify-content

- Se utiliza para alinear los ítems del eje principal

```
.container {  
  display: flex;  
  justify-content: flex-start | flex-end | center |  
    space-between | space-around | space-evenly | ...  
}
```

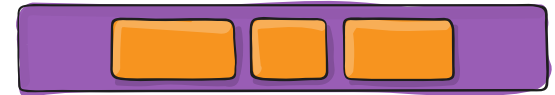
flex-start



flex-end



center



space-between



space-around



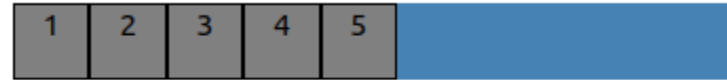
space-evenly



Flexbox

```
<div class="container">  
  <div class="item">1</div>  
  <div class="item">2</div>  
  <div class="item">3</div>  
  <div class="item">4</div>  
  <div class="item">5</div>  
</div>
```

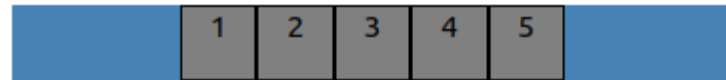
```
.container {  
  display: flex;  
  justify-content: flex-start;  
  background: steelblue;  
  width: 300px;  
}  
.item {  
  background: grey;  
  min-height: 30px;  
  min-width: 30px;  
  border: 1px solid black;  
  text-align: center;  
}
```



justify-content: flex-start;



justify-content: flex-end;



justify-content: center;



justify-content: space-between;



justify-content: space-around;

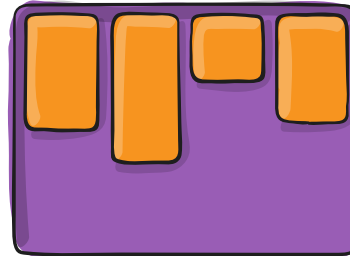
Flexbox

align-items

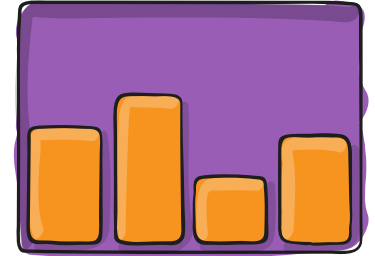
- Usada para alinear los ítems del eje secundario

```
.container {  
  display: flex;  
  align-items: stretch | flex-start | flex-end |  
    center | baseline | first baseline | ...  
}
```

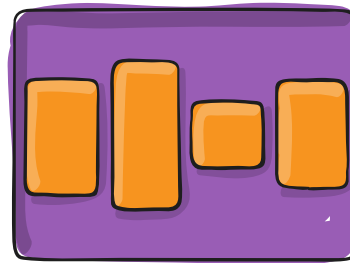
flex-start



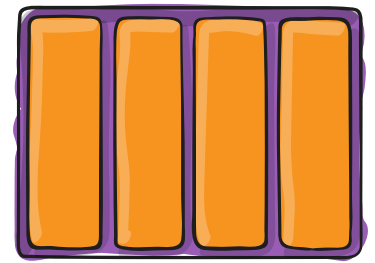
flex-end



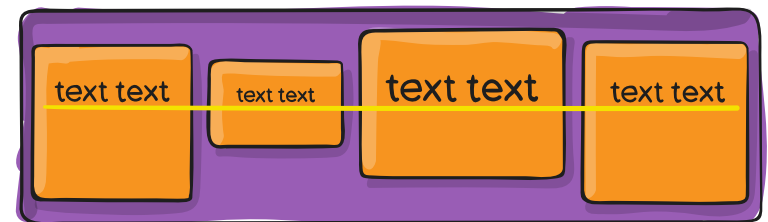
center



stretch



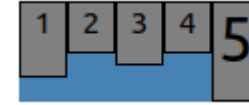
baseline



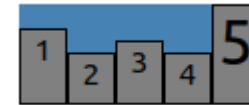
Flexbox

```
<div class="container">
  <div class="item">1</div>
  <div class="item">2</div>
  <div class="item">3</div>
  <div class="item">4</div>
  <div class="item">5</div>
</div>
```

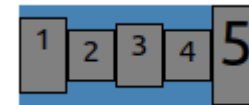
```
.container {
  display: flex;
  align-items: flex-end;
  background: steelblue;
  width: 100px;
}
.item {
  background: grey;
  min-height: 20px;
  min-width: 20px;
  border: 1px solid black;
  text-align: center;
}
.item:nth-child(1) { min-height: 30px; }
.item:nth-child(3) { min-height: 25px; }
.item:nth-child(5) { min-height: 40px; height: 40px; }
```



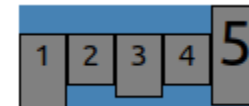
`justify-content: flex-start;`



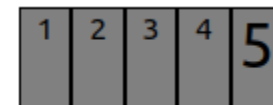
`justify-content: flex-end;`



`justify-content: center;`



`justify-content: baseline;`



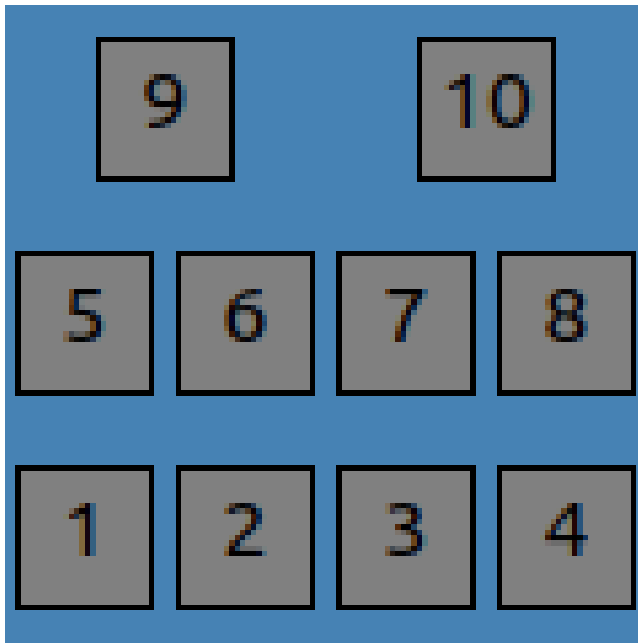
`justify-content: stretch;`

Propiedades comunes

- **flex-end:** Agrupa los ítems al final del eje principal.
- **center:** Agrupa los ítems al centro del eje principal.
- **space-between:** Distribuye los ítems dejando el máximo espacio para separarlos.
- **space-around:** Distribuye los ítems dejando el mismo espacio alrededor de ellos (izq/dcha).
- **space-evenly:** Distribuye los ítems dejando el mismo espacio (solapado) a izquierda y derecha.
- **stretch:** Estira los ítems para ocupar de forma equitativa todo el espacio.

Ejercicio 2

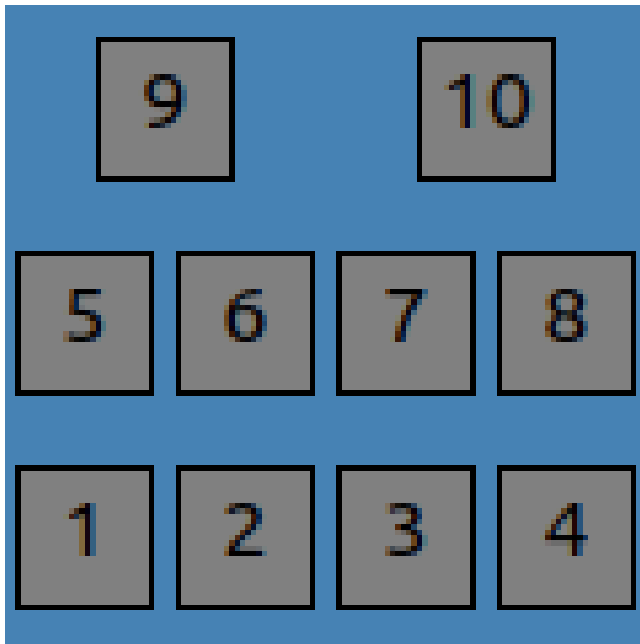
- Trata de replicar la siguiente figura utilizando Flexbox



```
<div class="container">  
  <div class="item">1</div>  
  <div class="item">2</div>  
  <div class="item">3</div>  
  <div class="item">4</div>  
  <div class="item">5</div>  
  <div class="item">6</div>  
  <div class="item">7</div>  
  <div class="item">8</div>  
  <div class="item">9</div>  
  <div class="item">10</div>  
</div>
```

Ejercicio - Solución

- Trata de replicar la siguiente figura utilizando Flexbox



```
body{
  padding-top: 1em;
}
.container {
  display: flex;
  align-items: center;
  justify-content: space-around;
  flex-wrap: wrap-reverse;
  background: steelblue;
  width: 100px;
  height: 100px;
}
.item {
  background: grey;
  min-height: 20px;
  min-width: 20px;
  border: 1px solid black;
  text-align: center;
}
```


- Herramienta para aprender *Flexbox*

FLEXBOX FROGGY

Nivel 1 de 24 ▾


Bienvenido a Flexbox Froggy, un juego donde ayudarás a Froggy y a sus amigos escribiendo código CSS. Guía a esta rana hacia la hoja de lirio en la derecha, usando la propiedad `justify-content`, la cual alinea elementos horizontalmente y acepta los siguientes valores:

- `flex-start`: Alinea elementos al lado izquierdo del contenedor.
- `flex-end`: Alinea elementos al lado derecho del contenedor.
- `center`: Alinea elementos en el centro del contenedor.
- `space-between`: Muestra elementos con la misma distancia entre ellos.
- `space-around`: Muestra elementos con la misma separación alrededor de ellos.

Por ejemplo, `justify-content: flex-end;` moverá la rana a la derecha.

```
1 #pond {
2   display: flex;
3   justify-content: flex-end;
4 }
5
6
7
8
9
10
```

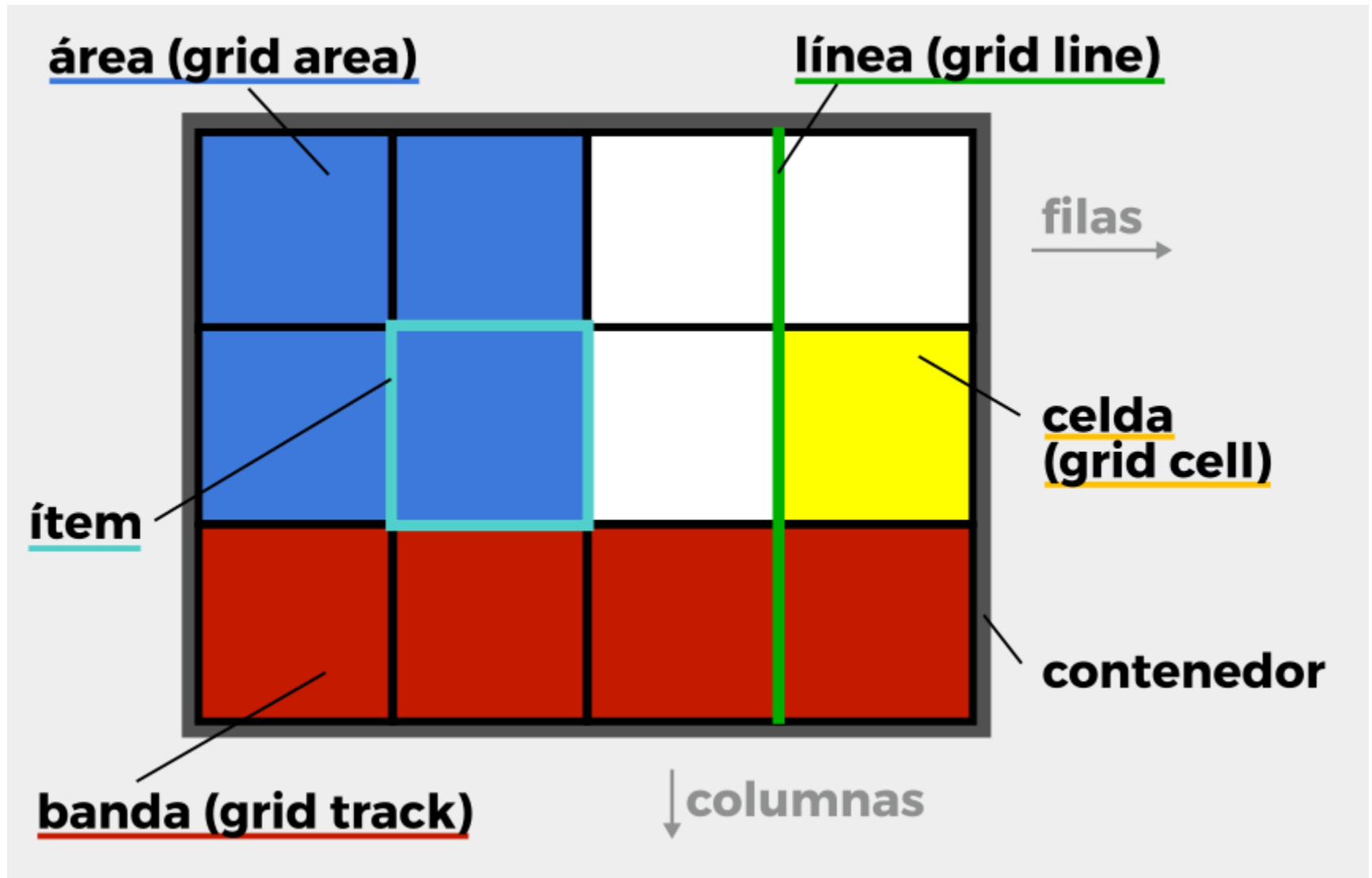
Siguiente



<https://flexboxfroggy.com/#es>



Grid



- **Elementos**

- **Contenedor:** El elemento padre contenedor que definirá la cuadrícula o rejilla.
- **Ítem:** Cada uno de los hijos que contiene la cuadrícula (elemento contenedor).
- **Celda (grid cell):** Cada uno de los cuadritos (unidad mínima) de la cuadrícula.
- **Area (grid area):** Región o conjunto de celdas de la cuadrícula.
- **Banda (grid track):** Banda horizontal o vertical de celdas de la cuadrícula.
- **Línea (grid line):** Separador horizontal o vertical de las celdas de la cuadrícula.

Propiedades

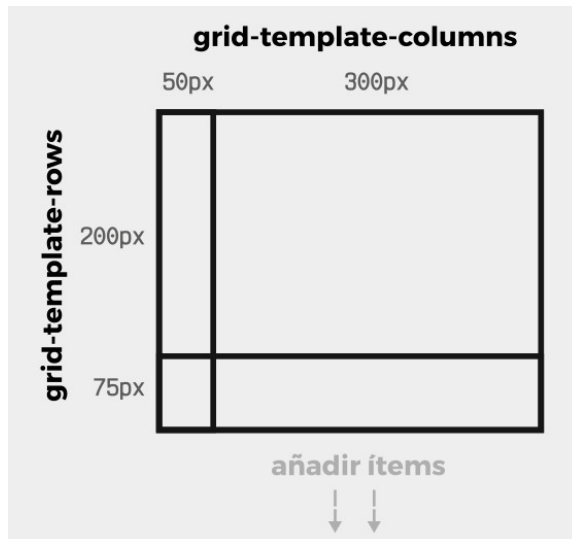
- **Propiedades**
 - **inline-grid:** Establece una cuadrícula con ítems en línea
 - **grid:** Establece una cuadrícula con ítems en bloque
 - **grid-template-columns:** Establece el tamaño de cada columna (col 1, col 2...).
 - **grid-template-rows:** Establece el tamaño de cada fila (fila 1, fila 2...).

Grid

- Dos columnas, una con 50 px y otra con 300 px de ancho
- Dos filas, una con 200 px y otra con 75 px de largo

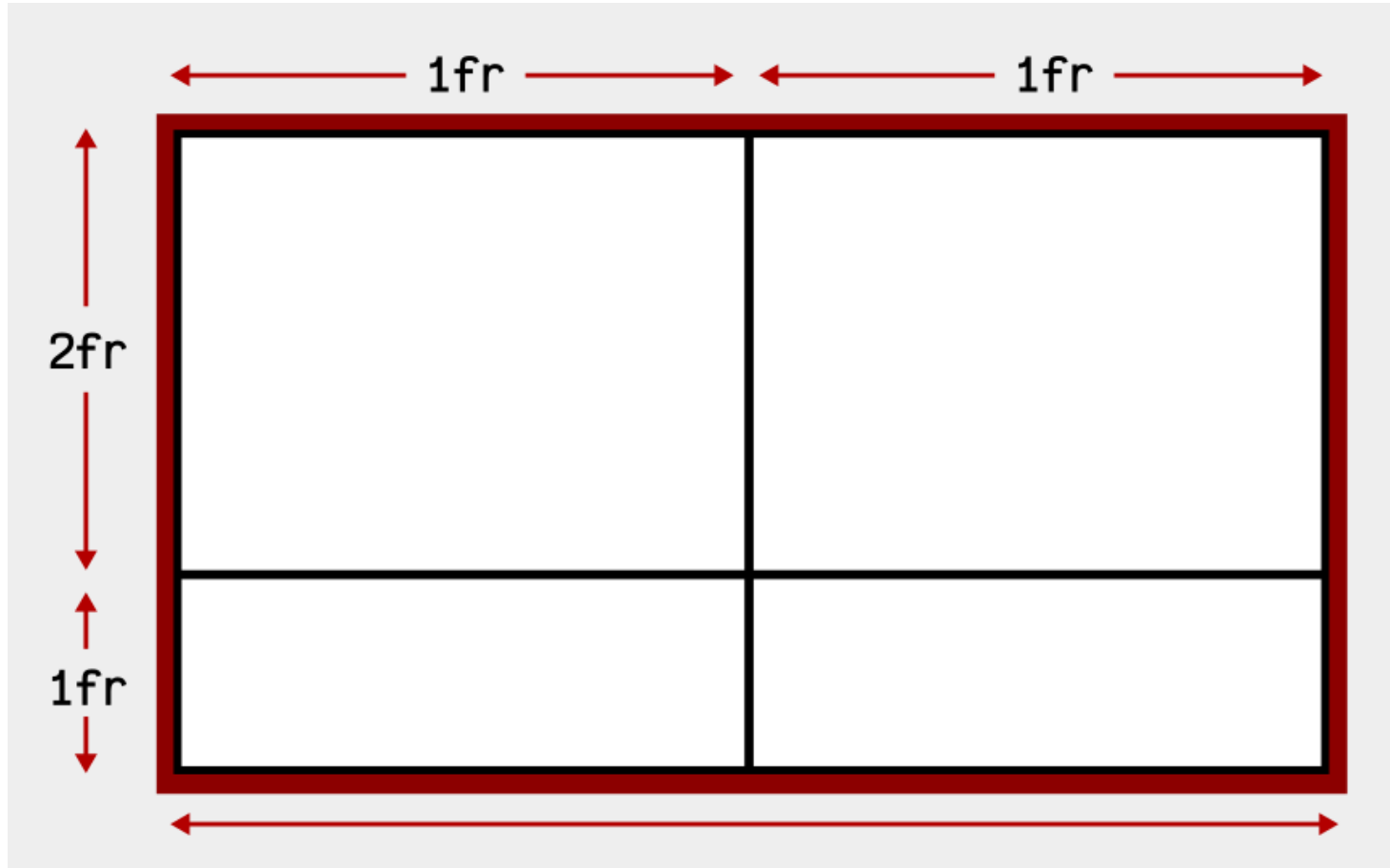
```
<div class="grid">
  <div class="item a">Item 1</div>
  <div class="item b">Item 2</div>
  <div class="item c">Item 3</div>
  <div class="item d">Item 4</div>
</div>
```

```
.grid {
  display: grid;
  grid-template-columns: 50px 300px;
  grid-template-rows: 200px 75px;
}
.item { color: white; }
.a { background: blue; }
.b { background: red; }
.c { background: green; }
.d { background: orange; }
```



Grid

Unidad de fracción restante



- Se puede combinar con otras unidades

Filas y columnas repetitivas

- Se pueden hacer repetitivas las filas y las columnas

```
.grid {  
  display: grid;  
  grid-template-columns: 100px repeat(2, 50px) 200px;  
  grid-template-rows: repeat(2, 50px 100px);  
}
```

- Si añadimos nuevos items, se crearán:
 - Columnas: 100 px, 50px, 50px, 200px
 - Filas: 50 px, 100 px, 50 px, 100 px

Filas y columnas repetitivas

- Se pueden hacer repetitivas las filas y las columnas

```
.grid {  
  display: grid;  
  grid-template-columns: 100px repeat(2, 50px) 200px;  
  grid-template-rows: repeat(2, 50px 100px);  
}
```



Grid

Propiedades - elementos

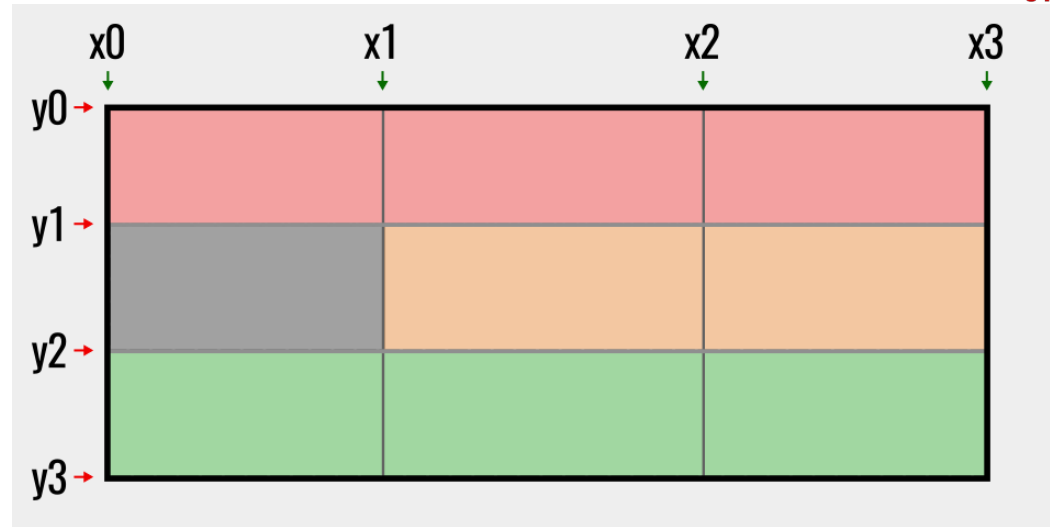


```
<div class="grid">  
  <div class="header">Header</div>  
  <div class="sidebar">Sidebar</div>  
  <div class="content">Content</div>  
  <div class="footer">Footer</div>  
</div>
```

```
.grid {  
  display: grid;  
  grid-template-columns: [x0] 1fr [x1] 1fr [x2] 1fr [x3];  
  grid-template-rows: [y0] 1fr [y1] 1fr [y2] 1fr [y3];  
}
```

Grid

```
.header {  
  background: darkred;  
  grid-column-start: x0;  
  grid-column-end: x3;  
}  
.sidebar {  
  background: black;  
  grid-row: y1 / y2;  
  color: white;  
}  
.content {  
  background: orange;  
  grid-column: x1 / x3;  
  grid-row: y1 / y3;  
}  
.footer {  
  background: green;  
  grid-column: x0 / x3;  
  grid-row: y2;  
}
```



- Zona **.header** desde la columna x0 a x3.
- Zona **.sidebar** desde la fila y1 a y2.
- Zona **.content** desde la columna x1 a x3 y desde la fila y1 a y3.
- Zona **.footer** desde la columna x0 a x3 en la fila y2.

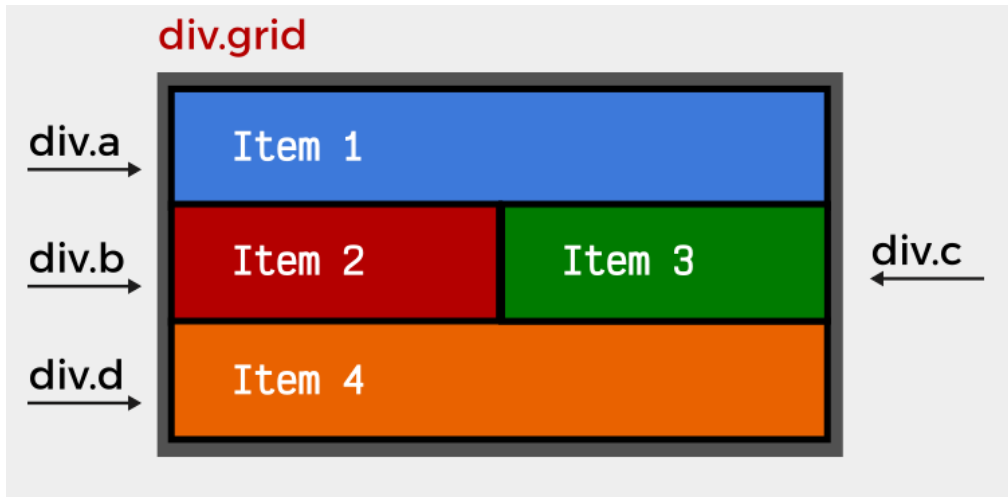


Grid

Por áreas

```
.grid {  
  display: grid;  
  grid-template-areas: "head head"  
                      "menu main"  
                      "foot foot";  
}  
.a { grid-area: head; background: blue }  
.b { grid-area: menu; background: red }  
.c { grid-area: main; background: green }  
.d { grid-area: foot; background: orange }
```

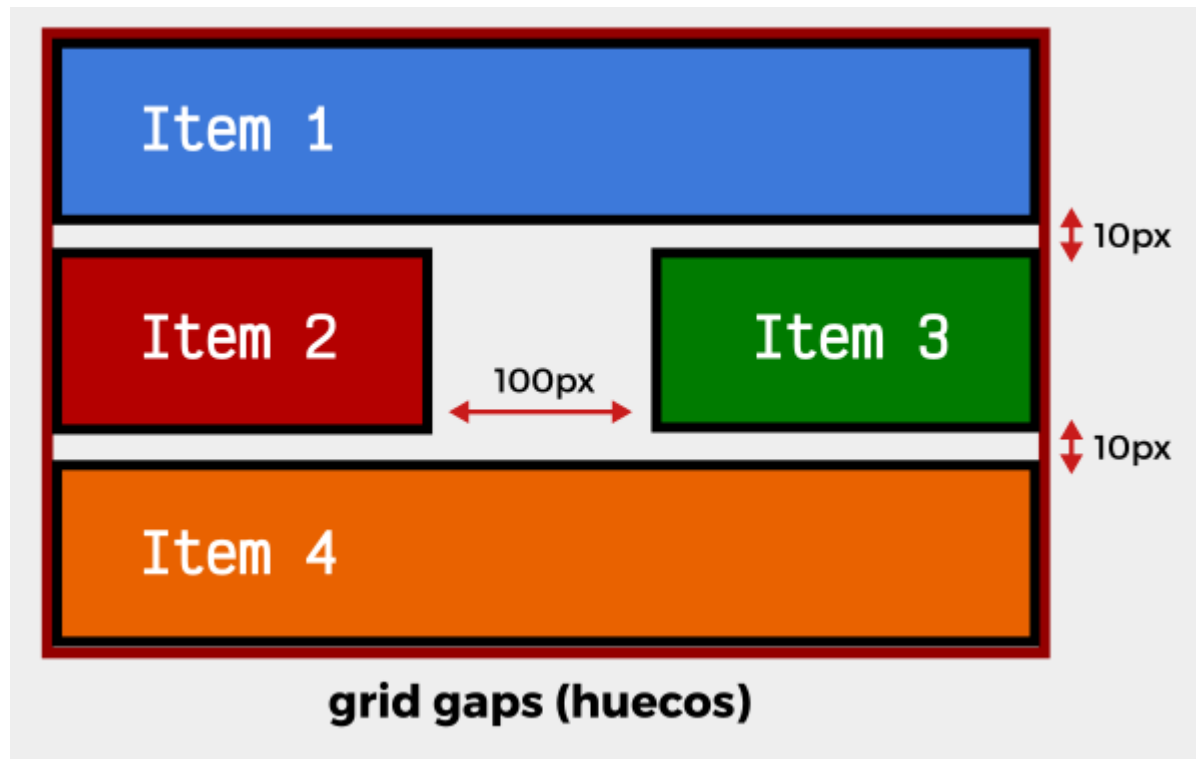
```
<div class="grid">  
  <div class="a">Header</div>  
  <div class="b">Sidebar</div>  
  <div class="c">Content</div>  
  <div class="d">Footer</div>  
</div>
```



Grid

Huecos

```
.grid {  
  column-gap: 100px;  
  row-gap: 10px;  
}
```



Posición

- **justify-items:** Distribuye los elementos en el eje horizontal (start | end | center | stretch)
- **align-items:** Distribuye los elementos en el eje vertical (start | end | center | stretch)

Distribución

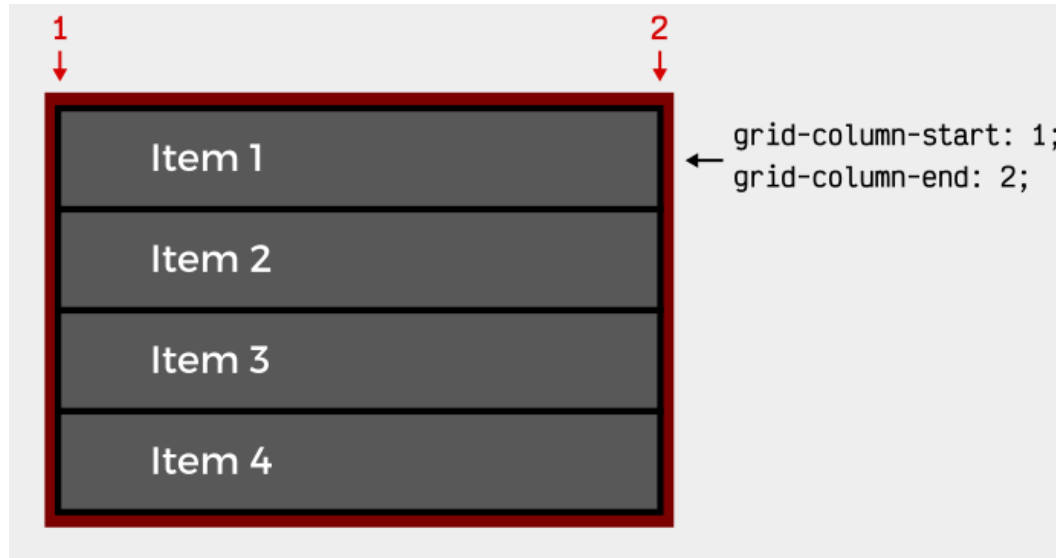
- **justify-content:** start | end | center | stretch | space-around | space-between | space-evenly
- **align-content:** start | end | center | stretch | space-around | space-between | space-evenly

Elementos hijos

- **Grid-column-start:** Indica en que columna empezará el ítem de la cuadrícula.
- **grid-column-end:** Indica en que columna terminará el ítem de la cuadrícula.
- **grid-row-start:** Indica en que fila empezará el ítem de la cuadrícula.
- **grid-row-end:** Indica en que fila terminará el ítem de la cuadrícula.

Grid

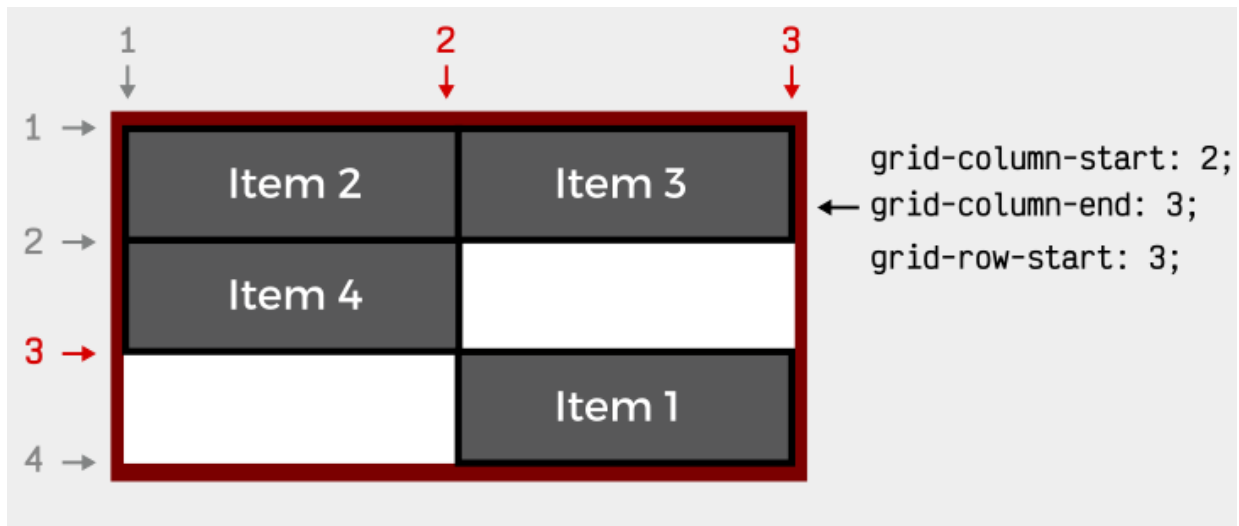
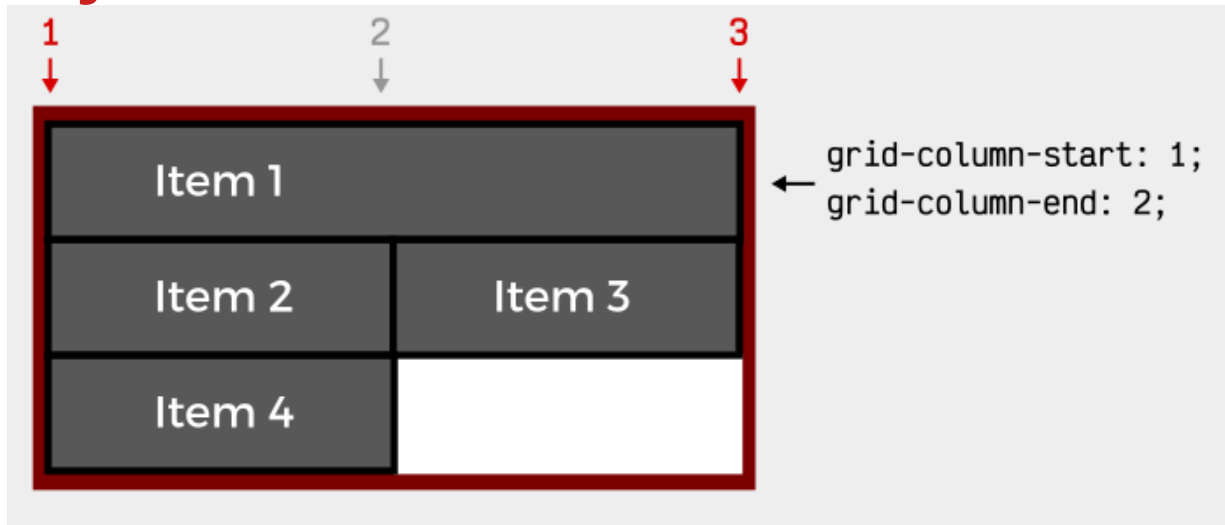
Elementos hijos



```
.grid {  
  display: grid;  
}  
.a {  
  grid-column-start: 1;  
  grid-row-end: 2;  
}
```


Grid

Elementos hijos

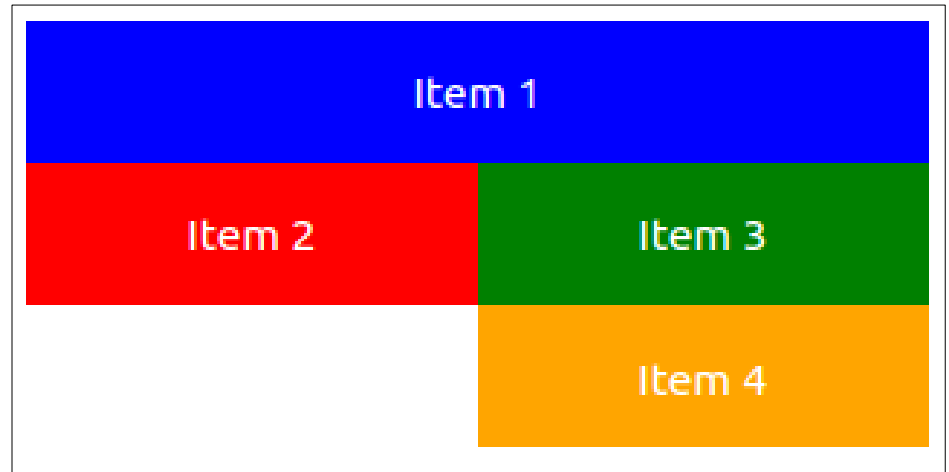


Ejercicio 3

- Trata de replicar la siguiente figura utilizando Grid

```
<div class="grid">  
  <div class="item a">Item 1</div>  
  <div class="item b">Item 2</div>  
  <div class="item c">Item 3</div>  
  <div class="item d">Item 4</div>  
</div>
```

```
.grid {  
  display: grid;  
}  
.item {  
  color: white;  
  padding: 1em;  
  text-align: center;  
}  
.a { background: blue; }  
.b { background: red; }  
.c { background: green; }  
.d { background: orange; }
```

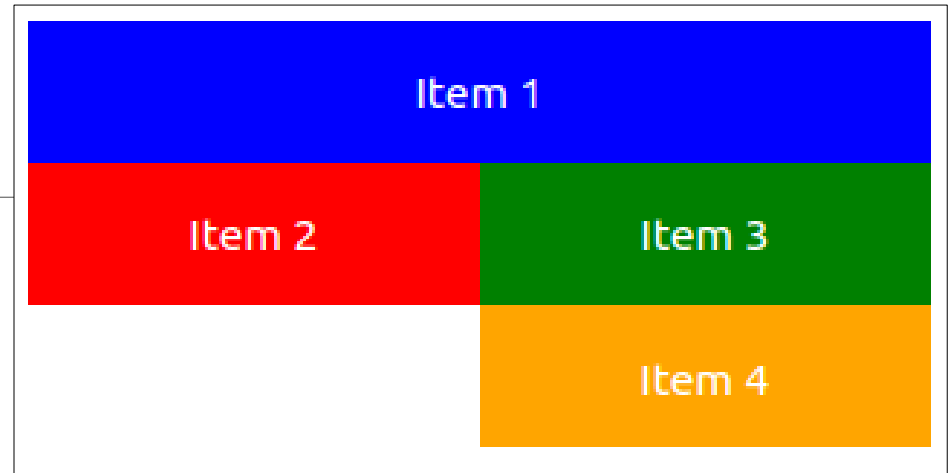


Ejercicio 3 - Solución

- Trata de replicar la siguiente figura utilizando Grid

```
<div class="grid">  
  <div class="item a">Item 1</div>  
  <div class="item b">Item 2</div>  
  <div class="item c">Item 3</div>  
  <div class="item d">Item 4</div>  
</div>
```

```
.grid {  
  display: grid;  
}  
.item {  
  color: white;  
  padding: 1em;  
  text-align: center;  
}  
.a { background: blue; grid-column-start: 1; grid-column-end: 3;}  
.b { background: red; grid-row-start: 2;}  
.c { background: green; grid-row-start: 2; }  
.d { background: orange; grid-row-start: 3; grid-column-start: 2}
```



- Herramienta para aprender *Grid*

GRID GARDEN

◀ Nivel 1 de 28 ▶

¡Bienvenido a Grid Garden, donde escribirás tu código CSS para cultivar tu jardín de zanahorias! Riega solo las áreas que tienen zanahorias usando la propiedad `grid-column-start`.

Por ejemplo, `grid-column-start: 3;` regará el área comenzando por la tercera línea vertical, que es otra manera de decir el 3er borde vertical contando desde la izquierda de la cuadrícula.

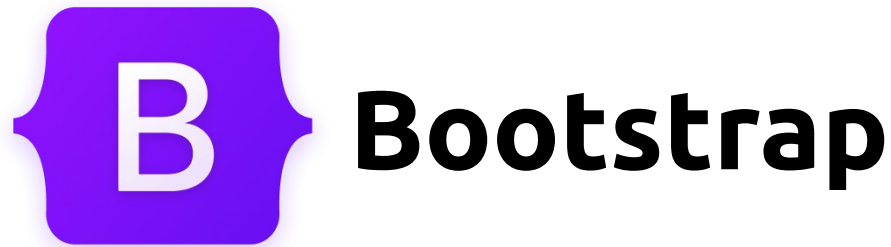
```
1 #garden {
2   display: grid;
3   grid-template-columns: 20% 20% 20% 20% 20%;
4   grid-template-rows: 20% 20% 20% 20% 20%;
5 }
6
7 #water {
8   
9 }
10
11
12
13
14
```

Siguiente

<https://cssgridgarden.com/#es>

- Diseñar una página completa editando el **CSS** desde cero es una tarea de **muy bajo nivel**
- Los diseñadores gráficos se especializan en **maquetación web** para crear **diseños** que los programadores podamos usar
- Existen **librerías de CSS** facilitan la **maquetación web**

- Algunas de las librerías CSS más usadas



<https://getbootstrap.com/>



<https://tailwindcss.com/>

- Páginas con templates HTML+CSS (comerciales y gratuitos)
 - https://www.w3schools.com/w3css/w3css_templates.asp
 - <https://templatemo.com/>
 - <https://htmlrev.com/>
 - <https://colorlib.com/wp/cat/bootstrap-5/>
 - <https://teleporthq.io/static-website-templates>
 - <https://themewagon.com/>