

Flex

# Flex/Flexbox

El Módulo de Caja Flexible, comúnmente llamado flexbox, fue diseñado como un modelo unidimensional de layout, y como un método que pueda ayudar a distribuir el espacio entre los ítems de una interfaz y mejorar las capacidades de alineación.

La diferencia principal entre Grid y Flex radica en que Flex está diseñado para organizar elementos en una línea (Vertical u horizontal), mientras que Grid fue diseñado para trabajar bajo dos dimensiones.

# Flex



Para comenzar a utilizar Flex, se deberá cambiar el display del contenedor a **display: flex;**

```
.modo-flex{  
  display: flex;  
}
```

```
<div class="modo-flex">  
</div>
```

# Ejercicio

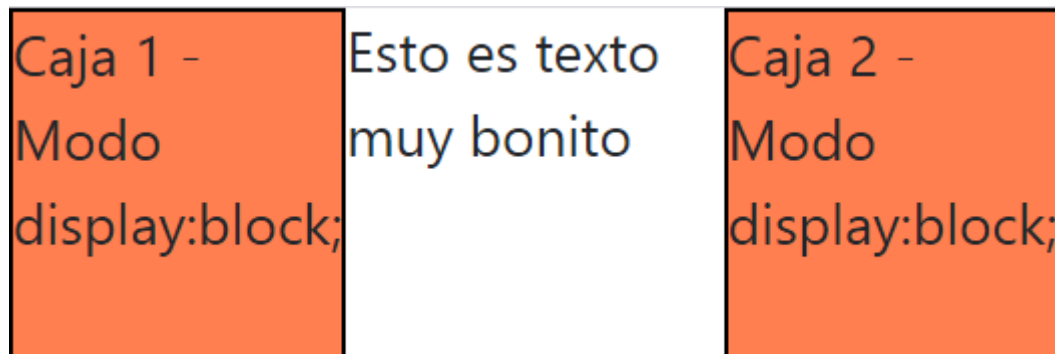
Implemente el siguiente código y observe el siguiente comportamiento.

```
.modo-flex {  
  width: 300px;  
  display: flex;  
}  
  
.modo-flex > div {  
  height: 100px;  
  width: 100px;  
  background-color:  coral;  
  border: 1px solid  black;  
  box-sizing: border-box;  
}
```

```
<div class="modo-flex">  
  <div>Caja 1 - Modo display:block;</div>  
  <span>Esto es texto muy bonito</span>  
  <div>Caja 2 - Modo display:block;</div>  
</div>
```

# Resultado

Sin importar el tipo de display de la etiqueta, este será alineado con el resto de elementos hermanos.



# Flex - Dirección

Una de las cualidades de Flex, es indicar la dirección de alineación en los elementos. En flex, podemos indicar que se comporte como fila o columna.

Propiedad:

**flex-direction**

Valores posibles:

1. **row**: Todos los elementos se alinean de forma horizontal.  
(Valor por defecto)
2. **row-reverse**: Todos los elementos se alinean de forma horizontal y su orden de distribución se encuentra invertido.
3. **column**: Todos los elementos se alinean de forma vertical.
4. **column-reverse**: Todos los elementos se alinean de forma vertical y su orden de distribución se encuentra invertido.

**Observación:** Esta propiedad debe ir en el contenedor.

# Utilizando el ejercicio anterior

Realice las siguientes pruebas

```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: column;  
}
```

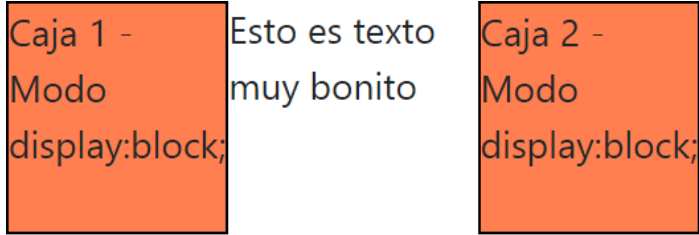
```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: column-reverse;  
}
```

```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: row;  
}
```

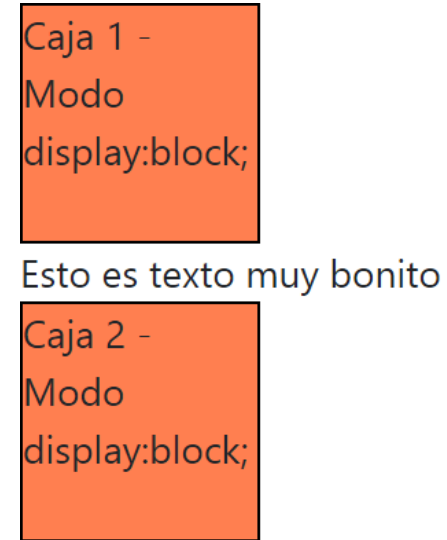
```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: row-reverse;  
}
```

# Resultados

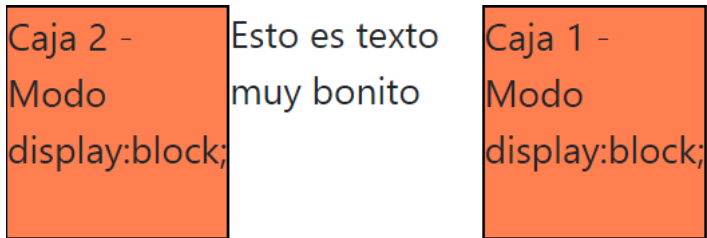
## row



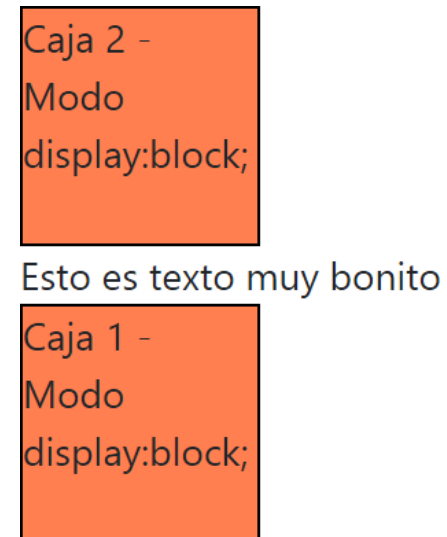
## column



## row-reverse



## column-reverse



Cuando se aplica reverse el orden de los elementos cambia, el primero ahora es el último y el último es ahora el primero.





# Ejercicio

Implemente el siguiente código y observe su comportamiento.

```
<div class="modo-flex">
  <div>Caja 1 </div>
  <div>Caja 2</div>
  <div>Caja 3 - Modo display:block;</div>
  <div>Caja 4</div>
  <div>Caja 5</div>
</div>
```

```
.modo-flex {
  width: 300px;
  display: flex;
  flex-direction: row;
}

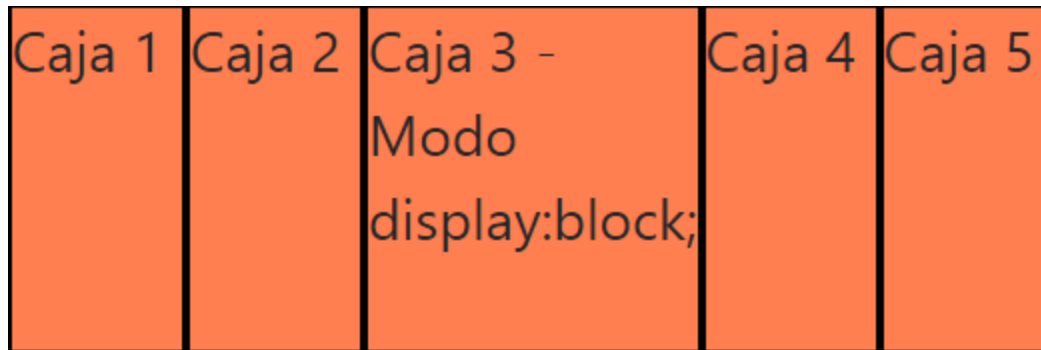
.modo-flex > div {
  height: 100px;
  width: 100px;
  background-color: coral;
  border: 1px solid black;
  box-sizing: border-box;
}
```

Observe que el contenedor flex tiene un **ancho de 300px** y cada caja en su interior tiene **un ancho de 100px**.

**¿Qué ocurrirá?**

# Respuesta

El contenedor flex, comprime todo su contenido y lo fuerza a mantenerse alineado.



# Distribución

En flex, podemos indicar si debe o no forzar la alineación en una única fila, en caso que el contenido no quepa en la fila, se puede indicar que genere una nueva fila.

Propiedad

**flex-wrap**

Valores posibles:

**nowrap**: Todo debe ser alineado en una fila. (Valor por defecto)

**wrap**: En el caso que no haya espacio se crea una nueva fila.

**wrap-reverse**: En caso que no haya espacio se crea una nueva fila y el **orden entre filas** es invertido.

**Observación:** Esta propiedad debe ir en el contenedor.

# Ejercicio

Utilizando el ejercicio anterior, utilice la nueva propiedad y observe su comportamiento.

```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
}
```

```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap-reverse;  
}
```

```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: nowrap;  
}
```

# Resultado

## nowrap

|        |        |                                    |        |        |
|--------|--------|------------------------------------|--------|--------|
| Caja 1 | Caja 2 | Caja 3 -<br>Modo<br>display:block; | Caja 4 | Caja 5 |
|--------|--------|------------------------------------|--------|--------|

## wrap

|        |        |                                    |
|--------|--------|------------------------------------|
| Caja 1 | Caja 2 | Caja 3 -<br>Modo<br>display:block; |
| Caja 4 | Caja 5 |                                    |

## wrap-reverse

|        |        |                                    |
|--------|--------|------------------------------------|
| Caja 4 | Caja 5 |                                    |
| Caja 1 | Caja 2 | Caja 3 -<br>Modo<br>display:block; |

# flex-flow

Propiedad que unifica en una sola línea las propiedades flex-direction y flex-wrap.

`flex-flow: flex-direction flex-wrap;`

```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
}
```



```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-flow: row wrap;  
}
```

# flex-grow

Define la capacidad de un elemento flexible para crecer si es necesario. Acepta un valor **sin unidad** que sirve como proporción. Esta proporción está acotada al espacio disponible, en el caso que no haya espacio no se notará ningún efecto.

Si todos los elementos tiene un flex-grow en 1, significará que el espacio sobrante se distribuirá por igual para todos.

# Ejercicio

Implemente el siguiente código y observe su comportamiento.

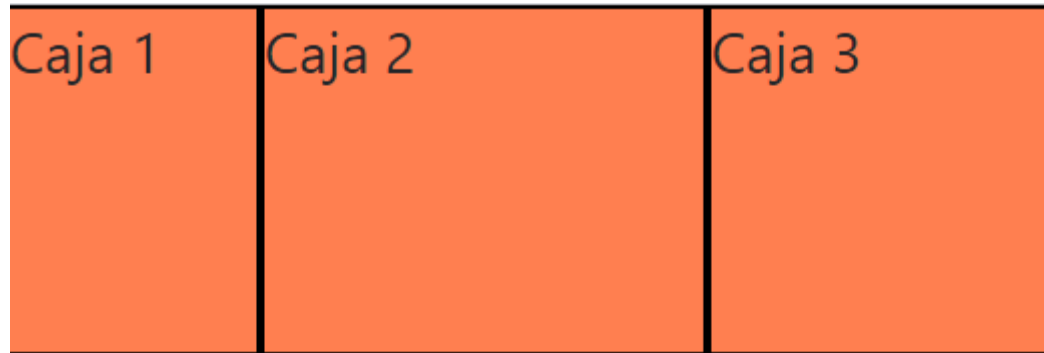
```
.modo-flex {  
  width: 300px;  
  display: flex;  
  flex-flow: row wrap;  
}  
  
.modo-flex > div {  
  height: 100px;  
  background-color: coral;  
  border: 1px solid black;  
  box-sizing: border-box;  
}  
  
.w-1 { flex-grow: 1; }  
.w-2 { flex-grow: 2; }  
.w-3 { flex-grow: 3; }
```

```
<div class="modo-flex">  
  <div class="w-1">Caja 1</div>  
  <div class="w-3">Caja 2</div>  
  <div class="w-2">Caja 3</div>  
</div>
```



# Respuesta

La caja 2 es 3 veces el tamaño de la caja 1 y la caja 3 es 2 veces el tamaño de la caja 1.



Recordar que esta propiedad distribuye el espacio sobrante en cada elemento. Si no hay espacio sobrante, no habrá ningún efecto.

# justify-content

Define la alineación a lo largo del eje principal. Ayuda a distribuir los elementos entre el espacio sobrante.

Propiedad

**justify-content**

Valores más usados:

- **start/flex-start**: Los elementos se posicionan en el inicio de la dirección.
- **center**: Los elementos están centrados a lo largo de la línea.
- **end/flex-end**: Los elementos se posicionan en el final de la dirección
- **space-around**: Los elementos distribuyen el espacio sobrante a su alrededor.
- **space-between**: los elementos se distribuyen uniformemente en la línea; el primer elemento está en la línea de inicio y el último elemento en la línea final.

**Observación 1:** Las palabras start y end son una estandarización de flex-start y flex-end, por lo que algunos navegadores como Chrome aún no reconocen la versión actualizada(start/end).

**Observación 2:** Esta propiedad debe ser aplicada al contenedor.

# Ejercicio

Implemente el siguiente código y utilice la propiedad justify-content.

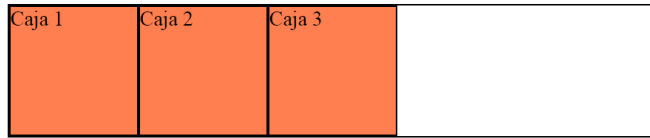
**start(flex-start)/center/end(flex-end)/  
space-around/space-between**

```
.modo-flex {  
  width: 500px;  
  display: flex;  
  flex-flow: row wrap;  
  border: 1px solid black;  
}  
  
.modo-flex > div {  
  width: 100px;  
  height: 100px;  
  background-color: coral;  
  border: 1px solid black;  
  box-sizing: border-box;  
}
```

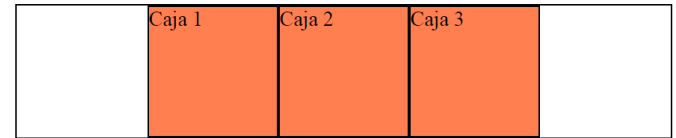
```
<div class="modo-flex">  
  <div>Caja 1</div>  
  <div>Caja 2</div>  
  <div>Caja 3</div>  
</div>
```

# Solución

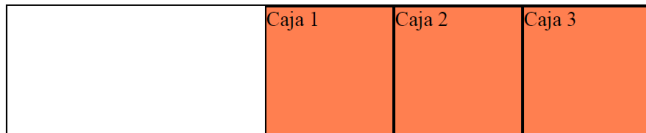
## Start/flex-start (Por Defecto)



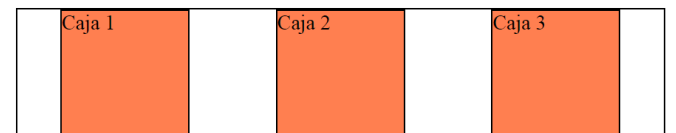
## center



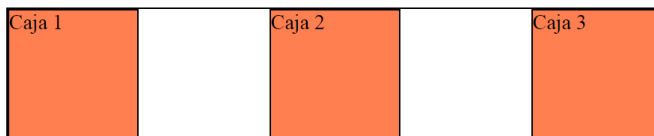
## end/flex-end



## space-around



## space-between



# align-items

Define el comportamiento predeterminado de cómo se distribuyen los elementos flexibles a lo largo del eje vertical.

Propiedad

**align-items**

Valores más usados:

- **start/flex-start**: Los elementos se posicionan en el inicio.
- **center**: Los elementos están centrados a lo largo de la línea vertical.
- **end/flex-end**: Los elementos se posicionan en la parte inferior.
- **stretch(Por defecto)**: Los elementos ocupan todo el alto posible. (Los elementos hijos no deben tener height fijo)

**Observación 1:** Las palabras start y end son una estandarización de flex-start y flex-end, por lo que algunos navegadores como Chrome aún no reconocen la versión actualizada(start/end).

**Observación 2:** Esta propiedad debe ser aplicada al contenedor.

# Ejercicio

Implemente el siguiente código y utilice la propiedad align-items.

**start(flex-start)/center/end(flex-end)/stretch**

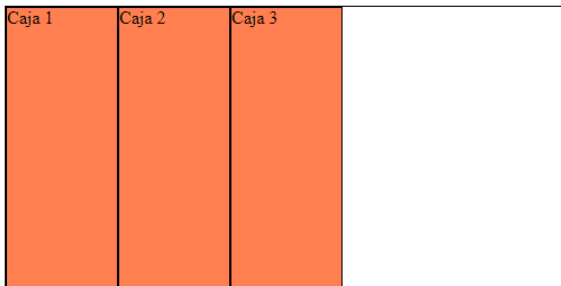
```
.modo-flex {  
  width: 500px;  
  height: 250px;  
  display: flex;  
  flex-flow: row wrap;  
  border: 1px solid black;  
}
```

```
.modo-flex > div {  
  width: 100px;  
  min-height: 100px;  
  background-color: coral;  
  border: 1px solid black;  
  box-sizing: border-box;  
}
```

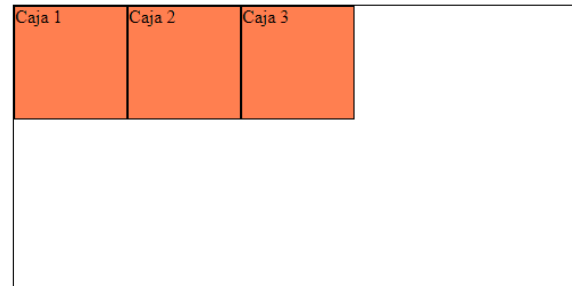
```
<div class="modo-flex">  
  <div>Caja 1</div>  
  <div>Caja 2</div>  
  <div>Caja 3</div>  
</div>
```

# Solución

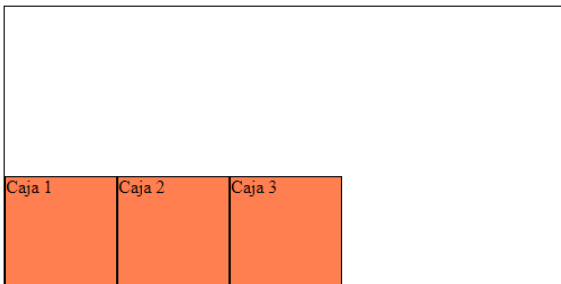
**stretch**



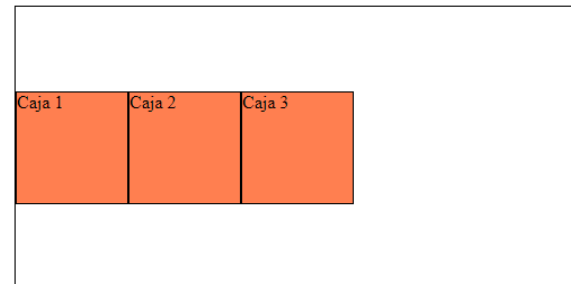
**start/flex-start**



**end/flex-end**



**center**



# align-self

En flex incorpora la propiedad para alinear de forma vertical cada elemento por separado.

Valores más usados:

- **start/flex-start**: Los elementos se posicionan en el inicio.
- **center**: Los elementos están centrados a lo largo de la línea vertical.
- **end/flex-end**: Los elementos se posicionan en la parte inferior.
- **stretch(Por defecto)**: Los elementos ocupan todo el alto posible. (Los elementos hijos no deben tener height fijo)

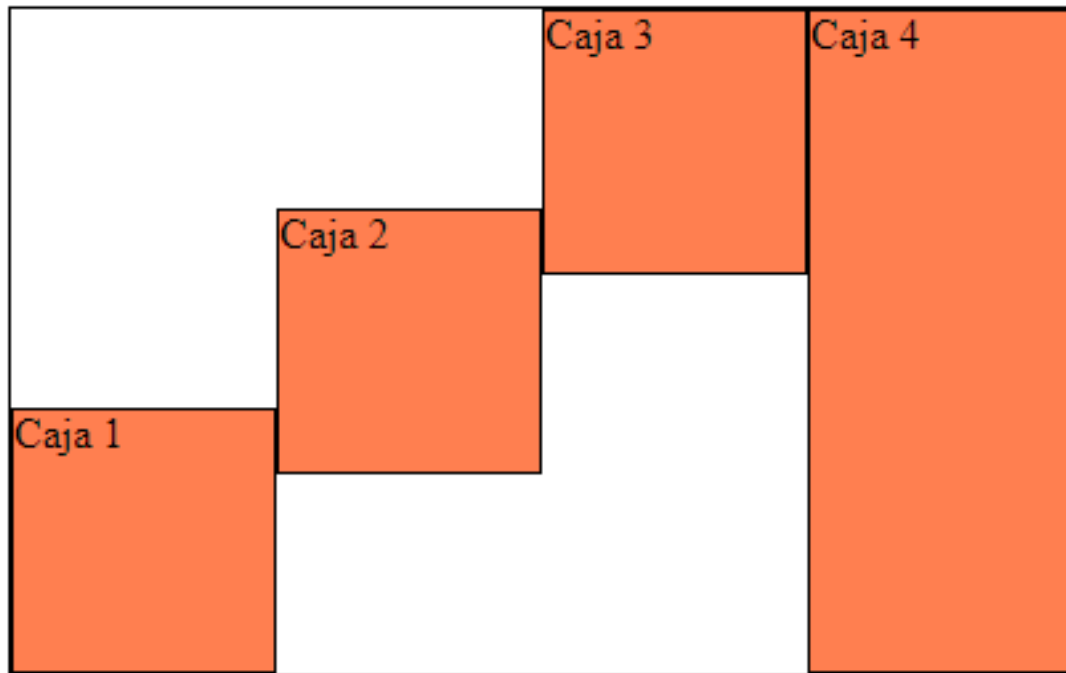
**Observación 1:** Las palabras start y end son una estandarización de flex-start y flex-end, por lo que algunos navegadores como Chrome aún no reconocen la versión actualizada(start/end).

**Observación 2:** Esta propiedad debe ser aplicada en el elemento hijo del contenedor.



# Ejercicio

Escriba el código necesario para obtener un resultado similar.



# Solución

```
.modo-flex {  
  width: 400px;  
  height: 250px;  
  display: flex;  
  flex-flow: row wrap;  
  border: 1px solid black;  
}  
  
.modo-flex > div {  
  width: 100px;  
  min-height: 100px;  
  background-color: coral;  
  border: 1px solid black;  
  box-sizing: border-box;  
}  
  
.stretch { align-self: stretch; }  
.start { align-self: flex-start; }  
.end { align-self: flex-end; }  
.center { align-self: center; }
```

```
<div class="modo-flex">  
  <div class="end">Caja 1</div>  
  <div class="center">Caja 2</div>  
  <div class="start">Caja 3</div>  
  <div class="stretch">Caja 4</div>  
</div>
```

# align-content

Distribuye las filas alrededor del espacio disponible.

Esta propiedad no tiene efecto cuando solo hay una línea de elementos flexibles.

- **start/flex-start**: Las filas se posicionan en el inicio.
- **center**: Las filas están centradas.
- **end/flex-end**: Las filas se posicionan en el final.
- **space-around**: Las filas distribuyen el espacio sobrante a su alrededor.
- **space-between**: Las filas se distribuyen uniformemente; la primera fila está en la línea de inicio y la última fila en la línea final.

# Ejercicio

Implemente el siguiente código y utilice la propiedad align-content.

**start(flex-start)/center/end(flex-end)/  
space-around/space-between**

```
.modo-flex {  
  display: flex;  
  flex-wrap: wrap;  
  height: 300px;  
  width: 200px;  
  border: 1px solid black;  
  align-content: flex-start;  
}  
  
.modo-flex > div {  
  height: 50px;  
  width: 50px;  
  background-color: rgb(0, 132, 194);  
  border: 1px solid rgb(0, 78, 114);  
  box-sizing: border-box;  
}
```

```
<div class="modo-flex">  
  <div>1</div>  
  <div>2</div>  
  <div>3</div>  
  <div>4</div>  
  <div>5</div>  
  <div>6</div>  
  <div>7</div>  
  <div>8</div>  
  <div>9</div>  
  <div>10</div>  
  <div>11</div>  
  <div>12</div>  
</div>
```

# Soluciones

**start/flex-start**

|   |    |    |    |
|---|----|----|----|
| 1 | 2  | 3  | 4  |
| 5 | 6  | 7  | 8  |
| 9 | 10 | 11 | 12 |
|   |    |    |    |

**end/flex-end**

|   |    |    |    |
|---|----|----|----|
|   |    |    |    |
| 1 | 2  | 3  | 4  |
| 5 | 6  | 7  | 8  |
| 9 | 10 | 11 | 12 |

**stretch**

**Elimine el height de los  
divs contenidos en flex**

|   |    |    |    |
|---|----|----|----|
| 1 | 2  | 3  | 4  |
| 5 | 6  | 7  | 8  |
| 9 | 10 | 11 | 12 |

**center**

|   |    |    |    |
|---|----|----|----|
|   |    |    |    |
| 1 | 2  | 3  | 4  |
| 5 | 6  | 7  | 8  |
| 9 | 10 | 11 | 12 |
|   |    |    |    |

**space-around**

|   |    |    |    |
|---|----|----|----|
|   |    |    |    |
| 1 | 2  | 3  | 4  |
|   |    |    |    |
| 5 | 6  | 7  | 8  |
|   |    |    |    |
| 9 | 10 | 11 | 12 |
|   |    |    |    |

**space-between**

|   |    |    |    |
|---|----|----|----|
| 1 | 2  | 3  | 4  |
|   |    |    |    |
| 5 | 6  | 7  | 8  |
|   |    |    |    |
| 9 | 10 | 11 | 12 |

# ¡Importante!

Cuando se cambia la dirección de distribución de elementos con `flex-direction` y se asigna el valor de `column`, las propiedades `align-items` y `justify-content` intercambian sus funcionalidades.

## `flex-direction: row;`

`align-items`: Alinea los elementos en **vertical**.

`justify-content`: Alinea los elementos en **horizontal**.

## `flex-direction: column;`

`align-items`: Alinea los elementos en **horizontal**.

`justify-content`: Alinea los elementos en **vertical**.

# Ejercicio

Ingresa a la siguiente página y completa todos los niveles!

## **Flex:**

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

## **Grid:**

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

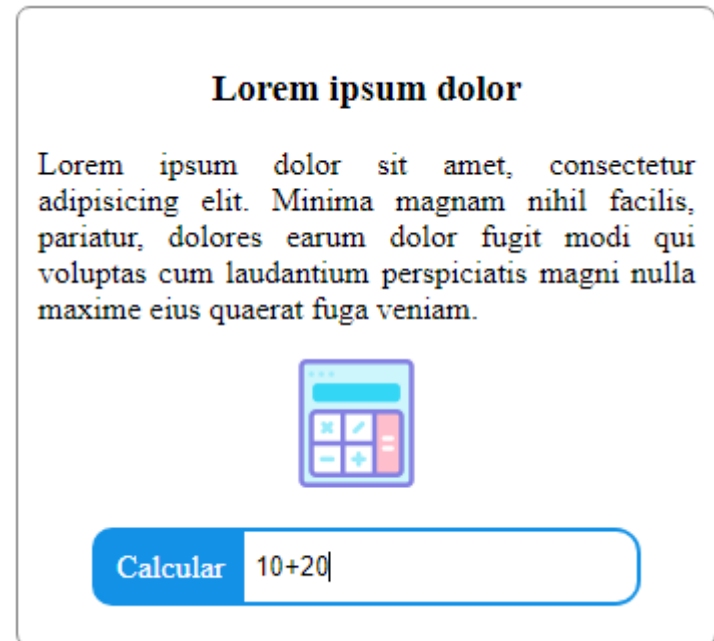
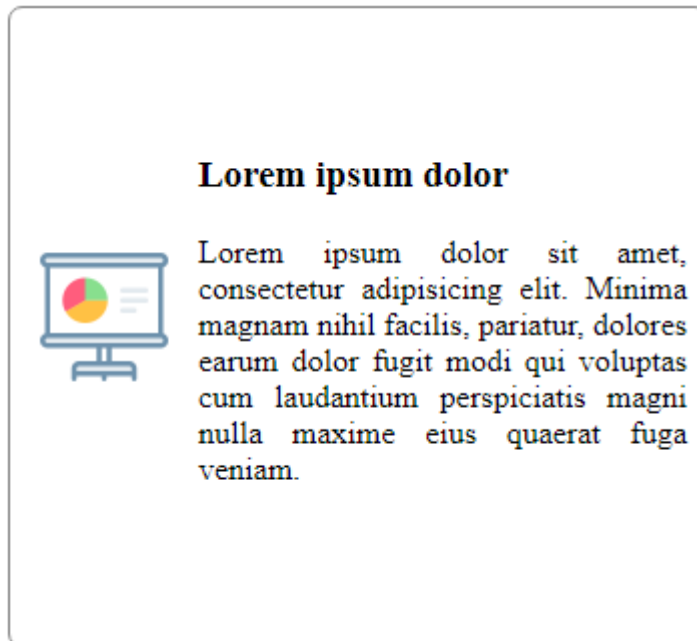
# Ejercicio - Cartas

Escribir el código necesario para representar la siguiente imagen.

Cada caja tiene un ancho de 350px.

Las imágenes tienen una dimensión de 64px.

El botón de calcular está diseñado con un Flex.





# Solución

Revisar solución en la página del curso.