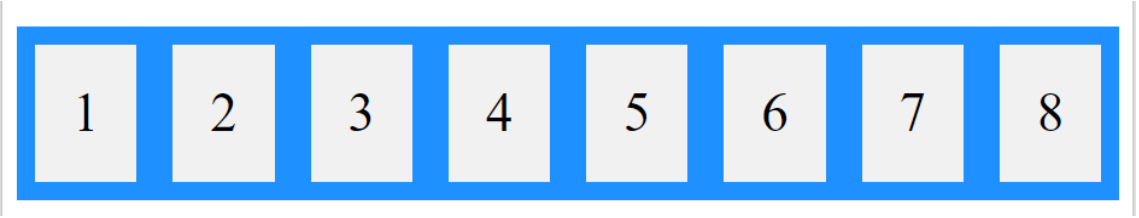
CSS Flexbox

Para comenzar a utilizar el modelo Flexbox, primero debe definir un contenedor flexible.

Ejemplo

Un contenedor flexible con tres elementos flexibles:

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




```
.flex-container {
  display: flex;
  flex-wrap: nowrap;
  background-color: DodgerBlue;
}

.flex-container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
```

Elemento principal (contenedor)

El contenedor se vuelve un contenedor flexbox estableciendo la propiedad `display:flex`

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

Las propiedades de un contenedor flexbox son:

- flex-direction
- flex-wrap
- flex-flow
- justify-content
- align-items

La propiedad flex-direction

La propiedad `flex-direction` define en qué dirección el contenedor quiere apilar los elementos flexibles.

Sus valores pueden ser:

- `column` - apila los elementos flexibles verticalmente (de arriba a abajo):

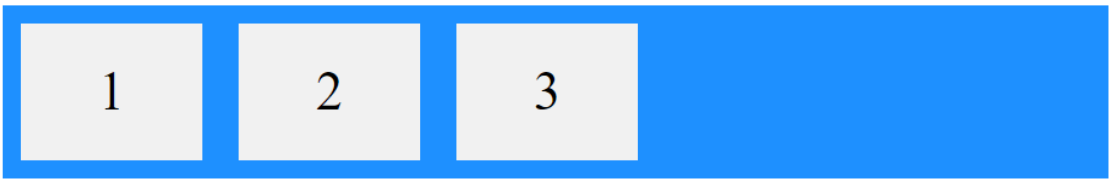
```
.flex-container {
  display: flex;
  flex-direction: column;
}
```

- `column-reverse` apila los elementos verticalmente (pero de abajo hacia arriba):

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

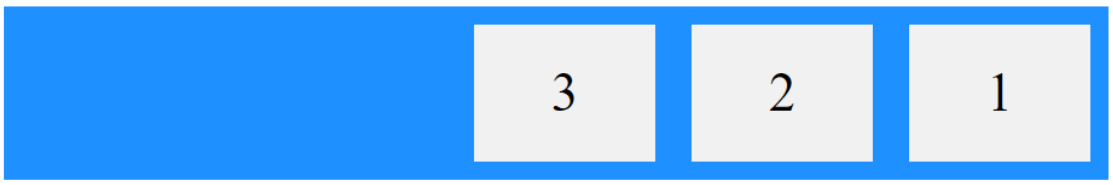
- `row -` apila los elementos flexibles horizontalmente (de izquierda a derecha):

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



- `row-reverse -` apila los elementos flexibles horizontalmente (pero de derecha a izquierda):

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



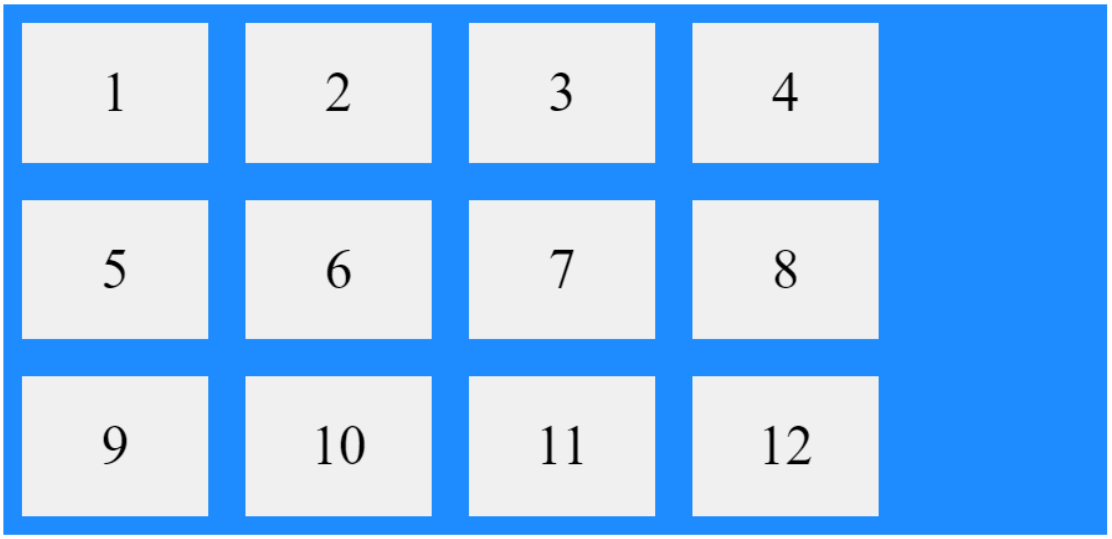
La propiedad flex-wrap

`flex-wrap e`specifica si los elementos deben ajustarse o no.

Ejemplo

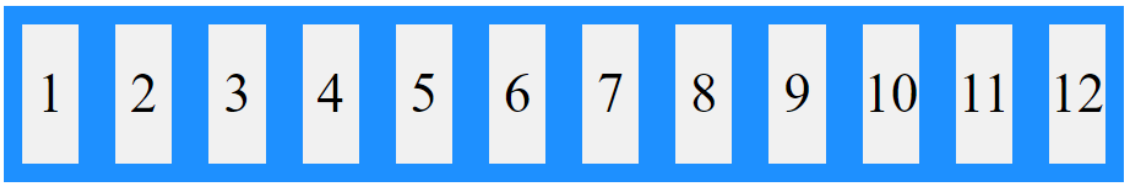
El valor `wrap` especifica que los elementos flexibles se ajustarán si es necesario:

```
.flex-container {
  display: flex;
  flex-wrap: wrap;
}
```



El valor `nowrap` especifica que los elementos flexibles no se ajustarán (esto es el valor predeterminado):

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



La propiedad justify-content

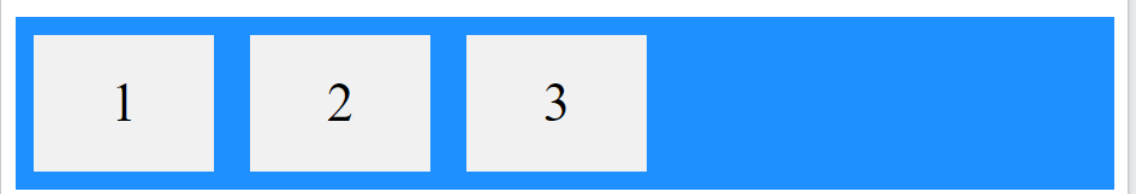
La propiedad `justify-content` se utiliza para alinear los elementos. Sus valores pueden ser:

- `center` - alinea los elementos en el centro de la fila o de la columna (según hayamos especificado la propiedad flex-direction en row o column)

```
.flex-container {
  display: flex;
  justify-content: center;
}
```

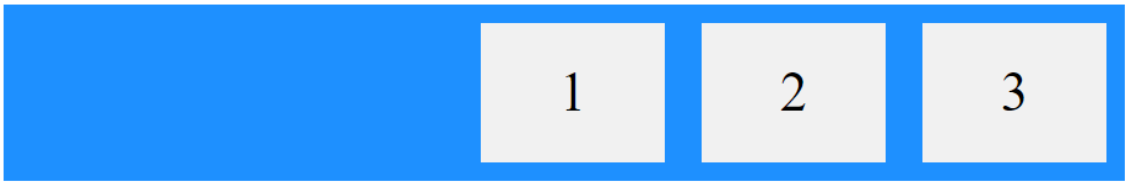
- `flex-start` - alinea los elementos al principio del contenedor (esto es el valor predeterminado):

```
.flex-container {
  display: flex;
  justify-content: flex-start;
}
```



- `flex-end` - alinea los elementos al final:

```
.flex-container {
  display: flex;
  justify-content: flex-end;
}
```

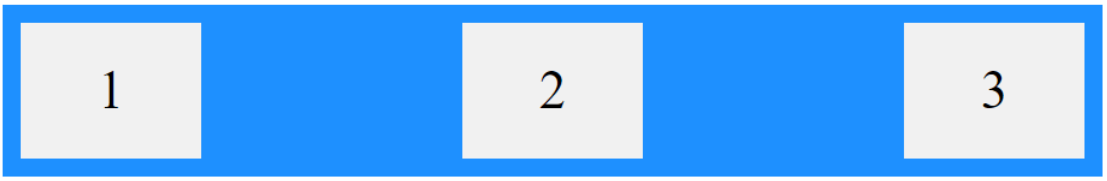


- `space-around` - muestra los elementos con espacio antes, entre y después:

```
.flex-container {
  display: flex;
  justify-content: space-around;
}
```

- `space-between` - muestra los elementos flexibles con espacio entre líneas:

```
.flex-container {
  display: flex;
  justify-content: space-between;
}
```



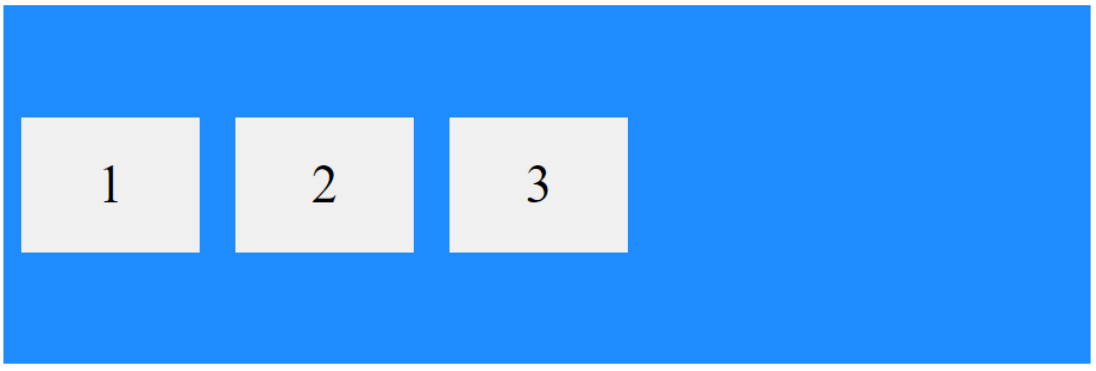
La propiedad align-items

La propiedad `align-items` se utiliza para alinear los elementos en el contenedor.

En estos ejemplos utilizamos un contenedor de 200 píxeles de alto para demostrar mejor esta propiedad.

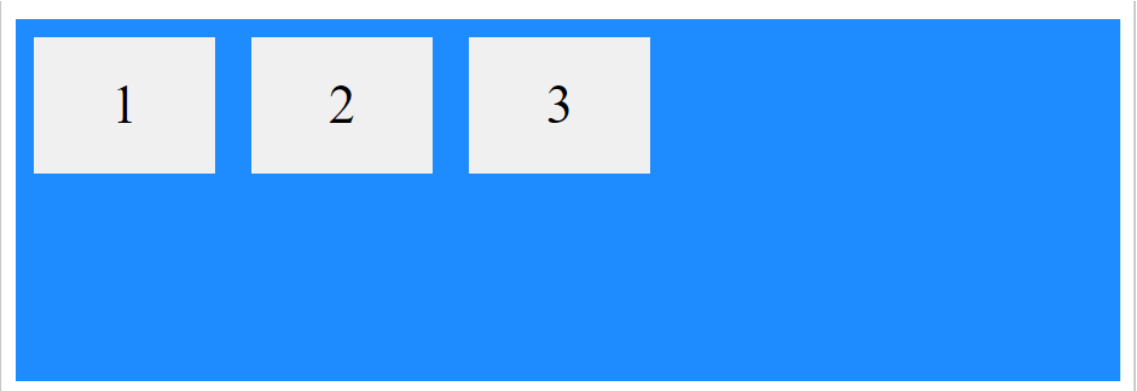
- `Center` - alinea los elementos flexibles en el medio del contenedor:

```
.flex-container {
  display: flex;
  height: 200px;
  align-items: center;
}
```



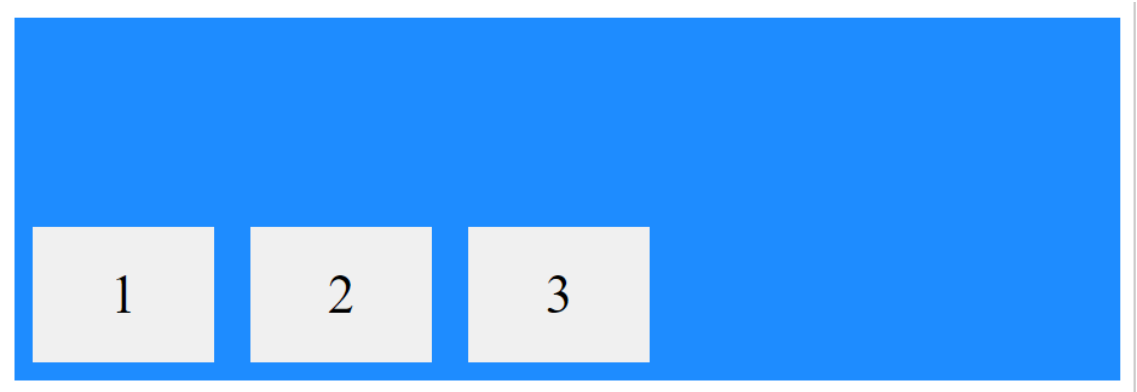
- `flex-start` - alinea los elementos flexibles en la parte superior del contenedor:

```
.flex-container {
  display: flex;
  height: 200px;
  align-items: flex-start;
}
```



- `flex-end` - alinea los elementos flexibles en la parte inferior del contenedor:

```
.flex-container {
  display: flex;
  height: 200px;
  align-items: flex-end;
}
```

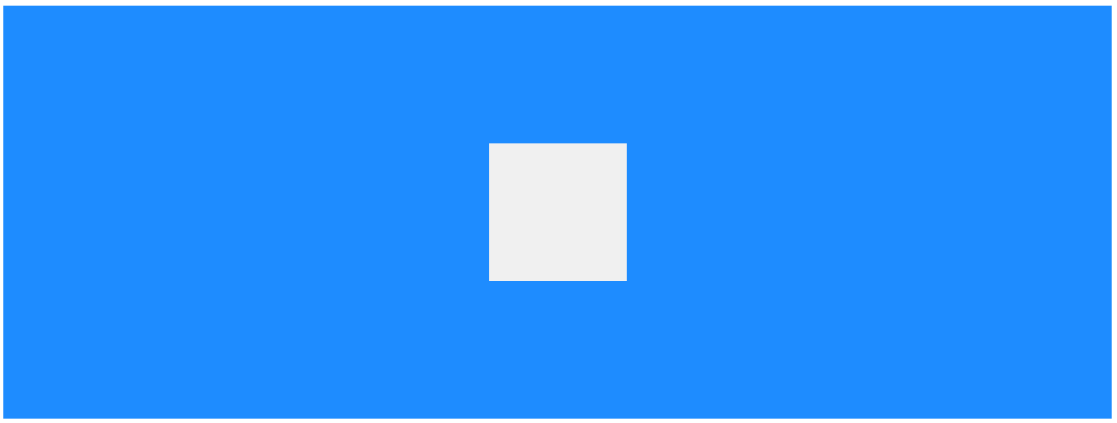


Centrado perfecto

En el siguiente ejemplo resolveremos un problema de estilo muy común: el centrado perfecto.

SOLUCIÓN: Establezca las propiedades `justify-content` y `align-items` en `center` y el elemento flexible quedará perfectamente centrado.

```css
.flex-container {
  display: flex;
  height: 300px;
  justify-content: center;
  align-items: center;
}
```



```css
/* Esta regla ajusta automáticamente las dimensiones de los elementos, de forma que no tengamos que ajustar su tamaño cada vez que cambiamos el padding o el border
concretamente:

* es el selector universal, y actúa como un comodín; significa que esta regla se aplica a TODOS los
  demás elementos en el documento HTML

  box-sizing: define cómo se deben calcular las dimensiones de un elemento; su valor por defecto (si no lo ponemos) es content-box, que significa que las propiedades width y height no incluyen el padding ni el border (aquí te lo explican:
https://www.w3schools.com/css/css_boxmodel.asp).

  border-box: hace que las dimensiones del elemento incluyan el contenido + padding + border, y así no te tienes que preocupar por que las cajas "se te salgan" si añades padding y/o border
*/
* {
    box-sizing: border-box;
  }

  body {
    margin: 0;
    font-family: sans-serif;
    background: #f5f6f7;
  }

  .header {
    text-align: center;
```

```
    text-transform: uppercase;
    padding: 32px;
    background-color: #0a0a23;
    color: #fff;
    border-bottom: 4px solid #fdb347;
  }

  /*
  la clase gallery es el contenedor flexbox (display:flex), así que todo lo que caiga dentro seguirá las reglas de de flexbox. Si no tienes una regla display:flex ya puedes poner justify-content, align-items, etc. que te va a dar igual, no van a funcionar.
  */

  .gallery {
    /* indicamos que gallery es un contenedor flex */
    display: flex;
    /* flex-direction: indicamos si los elementos se alinean en el eje horizontal (row) o en el eje vertical (column); además podemos indicar que te ponga los elementos en orden inverso (row-reverse o column-reverse)
    */
    flex-direction:row;
    /*
     flex-wrap (wrap significa "envolver") hace que, si no caben los elementos en una línea, se acomoden saltando a la segunda línea los que no quepan en la primera (poniendo el valor wrap). Si no queremos este salto de línea ponemos nowrap o no ponemos nada porque este es el valor por defecto.
    */
    flex-wrap: wrap;
    /*
    justify-content: pone los elementos al principio de la linea (flex-start), al final (flex-end), en el centro (center) o se distribuyen dejando el mismo espacio entre ellos (space-between) o dejando la mitad del espacio entre dos elementos adyacentes al principio y al final (space-around)
    */
    justify-content: center;
    /*
    align-items: indican dónde se ubican los elementos EN EL CONTENEDOR. Tiene los mismos valores que justify-content; la diferencia estriba en que justify-content indica cómo se distribuyen los elementos en la fila (si flex-direction es row) on en la columna (si flex-direction es column).
    */
    align-items: center;
    /* indica que habrá un espacio de 16px entre las filas y las columnas*/
    gap: 16px;
    max-width: 1400px;
    margin: 0 auto;
```

```
    padding: 20px 10px;
  }

  .gallery img {
    width: 100%;
    max-width: 350px;
    height: 300px;
    object-fit: cover;
    border-radius: 10px;
  }

  /* ::after es un pseudoelemento: añade un contenido después de
gallery*/
  .gallery::after {
    content: "";
    width: 350px;
  }
```

Referencias

https://www.w3schools.com/css/css3_flexbox.asp

https://www.freecodecamp.org/learn/2022/responsive-web-design/