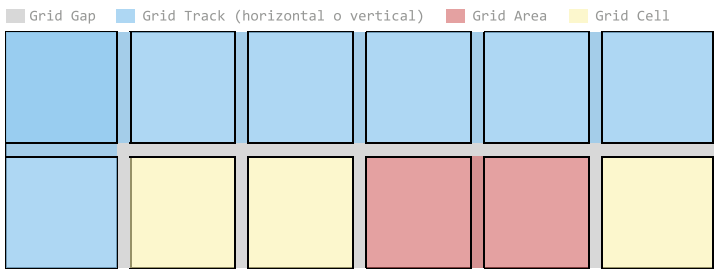


conceptos básicos



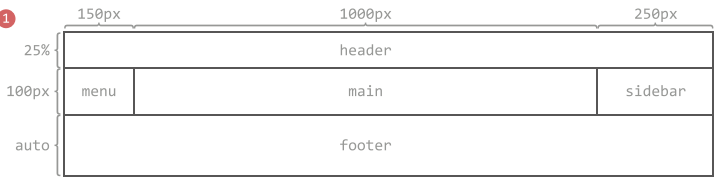
display // Determina que el contenedor va a ser un grid

```
.container {
  display: grid (organizado en bloque) | inline-grid (organizado en línea);
}
```

grid-template // Define el ancho/alto de las filas/columnas del grid

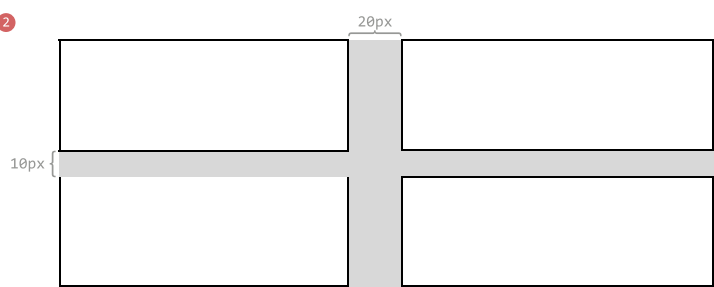
```
1. .container {
  grid-template-columns: 150px 1000px 250px; /* columnas */
  grid-template-rows: 25% 100px auto; /* filas */
  grid-template-areas: "header header header"
    "menu main sidebar"
    "footer footer footer"; /* áreas */
}

/* El valor ancho/alto se puede indicar con diferentes formatos */
grid-template-columns: 25% 50% 25%; /* porcentaje */
grid-template-rows: 1fr 3fr 1fr; /* tracks flexibles */
grid-template-columns: repeat(3, 200px); /* 3 columnas de 200px */
grid-template-rows: min-content max-content; /* relativo al contenido */
grid-template-columns: minmax(150px, 1000px) 1fr; /* rango mínimo y máximo */
```



grid-gap // Define el tamaño del espacio entre las filas/columnas del grid

```
2. .container {
  grid-row-gap: 10px; /* carril de las filas */
  grid-column-gap: 20px; /* carril de las columnas */
  grid-gap: 10px 20px; /* carril de filas y columnas */
}
```

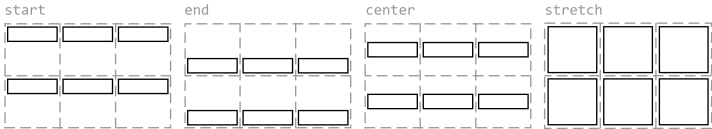


justify-items / align-items // Alinea o justifica las filas/columnas del grid

```
.container {
  justify-items: start | end | center | stretch (default);
}
```



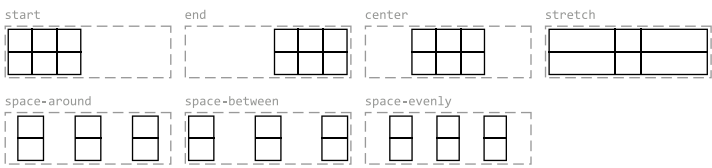
```
.container {
  align-items: start | end | center | stretch (default);
}
```



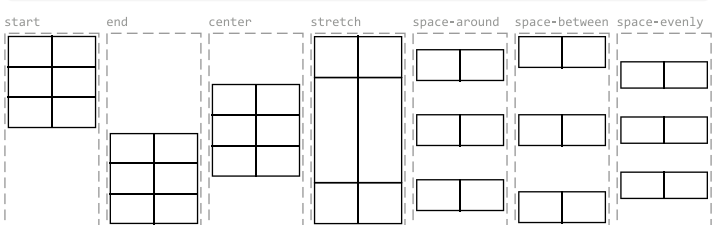
justify-content / align-content

// Alinea o justifica las filas/columnas del grid

```
.container {
  justify-content: start | end | center | stretch (default)
  space-around | space-between | space-evenly;
}
```



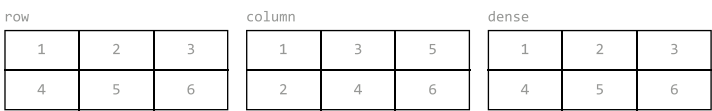
```
.container {
  align-content: start | end | center | stretch (default)
  space-around | space-between | space-evenly;
}
```



grid-auto-flow

// Coloca automáticamente elementos que no están explícitamente ubicados

```
.container {
  grid-auto-flow: row | column | dense;
}
```



grid-row / grid-column / grid-column + grid-row

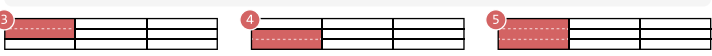
// Indica la ubicación basada en columnas/filas dentro del grid

```
/* Teniendo en cuenta un grid de 3 columnas y 3 filas */
```

```
3. .item-1 {
  /* columna 1 - filas 1 y 2 */
  grid-row-start: span 2;
}
```

```
4. .item-2 {
  /* columna 1 - filas 2 y 3 */
  grid-row-start: 2;
  grid-row-end: 4;
}
```

```
5. .item-3 {
  /* columna 1 - filas 1, 2 y 3 */
  grid-row: 1/4 | 1 / span 3;
}
```

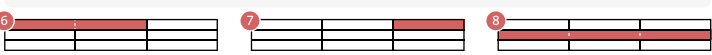


```
/* Teniendo en cuenta un grid de 3 columnas y 3 filas */
```

```
6. .item-1 {
  /* columnas 1 y 2 - fila 1 */
  grid-column-start: span 2;
}
```

```
7. .item-2 {
  /* columna 3 - fila 1 */
  grid-column-start: 3;
  grid-column-end: 4;
}
```

```
8. .item-3 {
  /* columnas 1, 2 y 3 - fila 1 */
  grid-row: 2 / 4 | 2 / span 3;
}
```



```
/* Teniendo en cuenta un grid de 3 columnas y 3 filas */
```

```
.item {
  /* columnas 1, 2 y 3 - filas 2 y 3 */
  grid-column: 1 / span 3;
  grid-row: 2 / span 2;
}
```



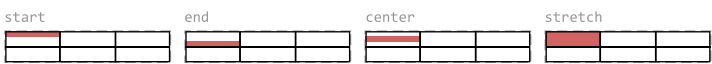
justify-self / align-self

// Alinea vertical u horizontalmente el contenido de una cuadrícula

```
.item-a {
  justify-self: start | end | center | stretch (default);
}
```



```
.item-a {
  align-self: start | end | center | stretch (default);
}
```

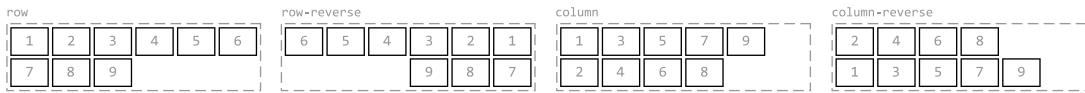


display // Define un contenedor flexible, permitiendo alinear sus hijos directos de la forma que determinemos

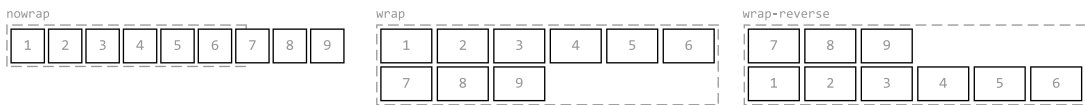
```
.container {
  display: flex | inline-flex;
}
```

flex-direction / flex-wrap

```
.container{
  /* Dirección de los items en fila o columna */
  flex-direction: row | row-reverse | column | column-reverse;
}
```



```
.container{
  /* Determina si los items se ajustan en una línea o no */
  flex-wrap: nowrap | wrap | wrap-reverse;
}
```

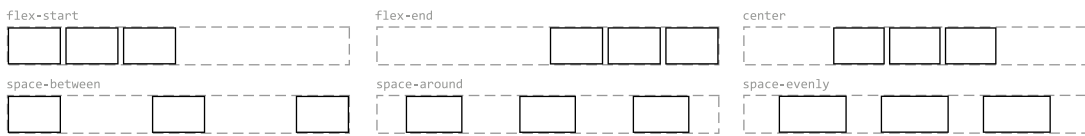


flex-flow (flex-direction + flex-wrap)

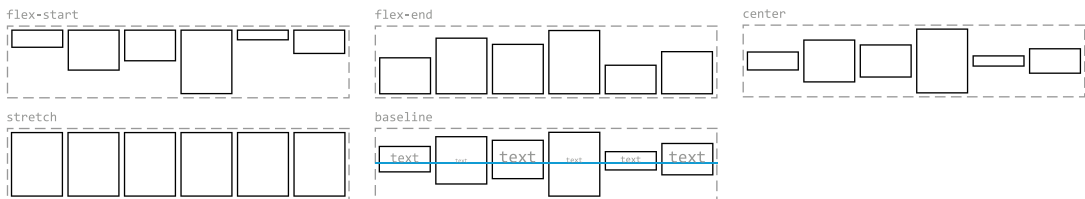
```
.container {
  /* Shorthand */
  flex-flow: row (flex-direction) wrap (flex-wrap);
}
```

justify-content / align-items // Alinea los items en sentido horizontal y vertical

```
.container {
  /* Alineación horizontal de los item */
  justify-content: flex-start | flex-end | center | space-between | space-around | space-evenly;
}
```

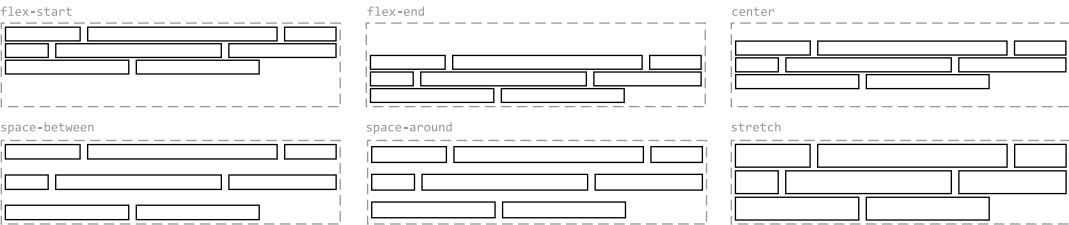


```
.container {
  /* Alineación vertical de los items */
  align-items: flex-start | flex-end | center | stretch | baseline;
}
```



align-content // Alinea los items en sentido vertical y horizontal

```
.container {
  align-content: flex-start | flex-end | center | space-between | space-around | stretch;
}
```



flex-grow / flex-shrink / flex-basis

```
.item:nth-child(4) {
  /* Define en qué proporción crece un item */
  flex-grow: 3 (número); /* por defecto es 0 */
}
```

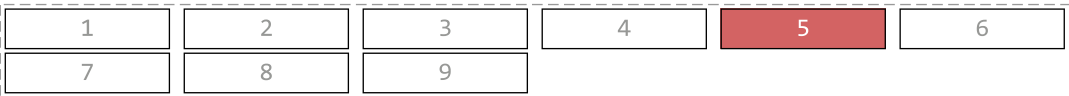


```
.item {
  /* Define la capacidad de un objeto flexible para contraerse si es necesario */
  flex-shrink: 2 (número); /* por defecto es 1 */

  /* Define el tamaño de un item antes de distribuir el espacio restante */
  flex-basis: auto | 10% | 5rem;
}
```

order

```
.item {
  /* Orden de los items, permitiendo colocar un elemento en un orden diferente al que le corresponde en el código */
  order: 5 (número); /* por defecto es 0 */
}
```



align-self // Permite alinear de forma individual un elemento

```
.item:nth-child(3) {
  align-self: auto | flex-start | flex-end | center | baseline | stretch;
}
```

