Bootcamp Full Stack

Arquitectura Web

Introducción a *Flex*

Un poco de historia: las tablas

¿Sabías que, antes, para encolumnar elementos se trabajaba con tablas? Sí, en los inicios de HTML y CSS, se utilizaban herramientas como *slice,* de Photoshop, para cortar el diseño y llevarlo al HTML en forma de tabla. Era muy complejo realizar esta tarea y, la verdad, nunca quedaba bien.

Luego, llegó la propiedad float, que si bien generó un gran progreso en el maquetado del sitio, seguía generando grandes problemáticas.

Por ejemplo, la necesidad constante de usar clear para no arrastrar un encolumnado que no era compartido por todos los elementos del diseño.

Con la aparición de flex y, más adelante, de grid, estos procesos se simplificaron y se transformaron en una tarea que, por supuesto, amerita conocimiento y atención, pero se muestra radicalmente más sencilla que en los contextos antes mencionados.

¿Qué es *flex?*

Flex surge ante la necesidad de encolumnar elementos. Si bien el mundo del encolumnado ha recorrido diversas posibilidades y sus particularidades serán discutidas en cursos posteriores, **flex es, actualmente, la forma más utilizada de encolumnar elementos**.

Flex, como valor de la propiedad **display**, aplicado sobre un contenedor, permite encolumnar sus elementos.

Por ejemplo, contamos con el main de nuestra interfaz y deseamos generar una estructura como la del ejemplo:

Para lograrlo, generar la siguiente estructura en el **HTML**:

index.html > ...

```
4       <meta charset="UTF-8">
5       <meta http-equiv="X-UA-Compatible" content="IE=edge">
6       <meta name="viewport" content="width=device-width, initial-scale=1.
        0">
7       <link rel="stylesheet" href="css/estilos.css">
8       <title>Selectores</title>
9   </head>
10  <body>
11  <main>
12  <section id="uno">1</section>
13  <section id="dos">2</section>
14  <section id="tres">3</section>
15  </main>
16  </body>
17  </html>
```

Luego, generar el **CSS** de la siguiente manera:

```
3  main{display: flex;}
4  #uno{background-color: #041460}
5  #dos{background-color: #0a2ff1}
6  #tres{background-color: #3957f1}
7
```

Flex: alineación horizontal

La alineación horizontal se trabajará mediante la propiedad **justify-content** con los siguientes posibles valores:

- **Flex-start:** valor por defecto o predeterminado, es decir, sin indicar nada, este será el valor.

```
9   main{display: flex;justify-content: flex-start;}
10  #uno{background-color: #041460}
11  #dos{background-color: #0a2ff1}
12  #tres{background-color: #3957f1}
13
```

El resultado será el siguiente:

- **Center:** los elementos se **centran en el contenedor sin espacios entre sí.**

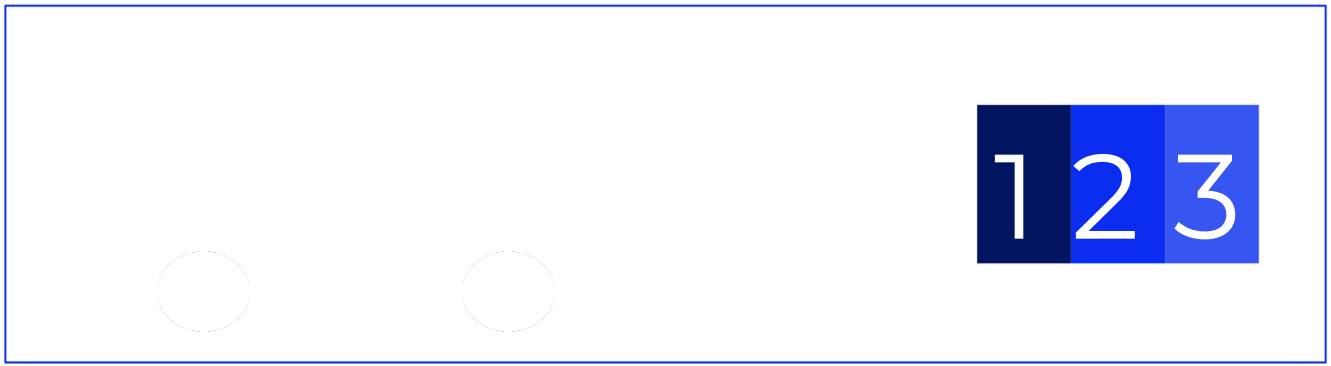
```
15  main{display: flex;justify-content: center;}
16  #uno{background-color: #041460}
17  #dos{background-color: #0a2ff1}
18  #tres{background-color: #3957f1}
19
```

El resultado será el siguiente:

- **Flex-end:** permite ubicar los **elementos orientados hacia el final del contenedor.**

```
21  main{display: flex;justify-content: flex-end;}
22  #uno{background-color: #041460}
23  #dos{background-color: #0a2ff1}
24  #tres{background-color: #3957f1}
25
```

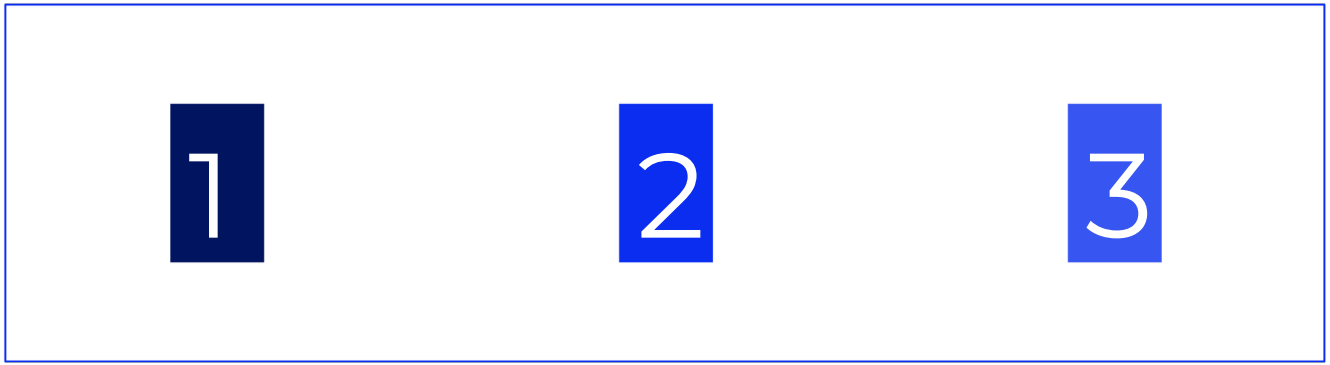
El resultado será el siguiente:



- **`Space-around:`** genera **espacio antes y después** de los elementos.

```
27  main{display: flex;justify-content: space-around;}
28  #uno{background-color: #041460}
29  #dos{background-color: #0a2ff1}
30  #tres{background-color: #3957f1}
31
```

El resultado será el siguiente:

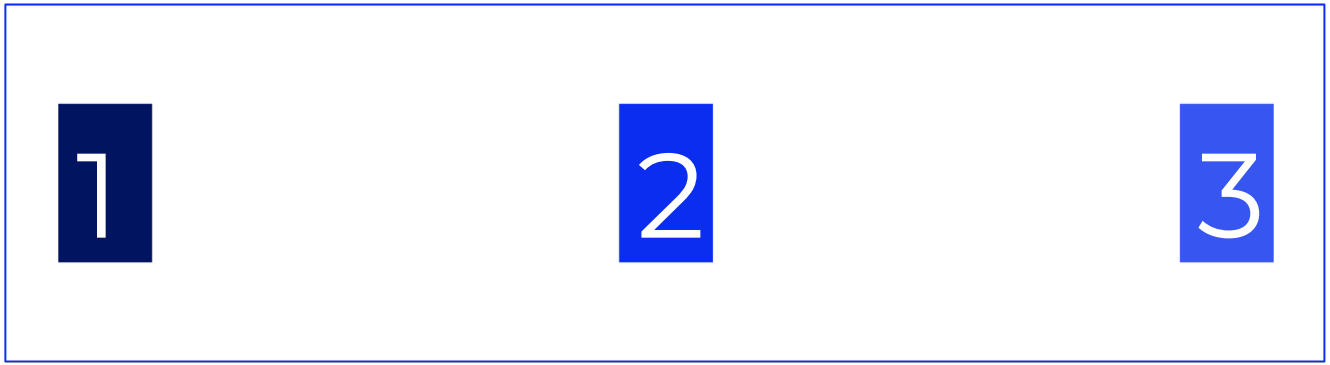


Nota: el espacio entre medio de los elementos es mayor porque se unen el espacio anterior y posterior de los mismos.

- **`Space-between:`** genera **espacio entre los elementos**.

```
33  main{display: flex;justify-content: space-between;}
34  #uno{background-color: #041460}
35  #dos{background-color: #0a2ff1}
36  #tres{background-color: #3957f1}
37
```

El resultado será el siguiente:

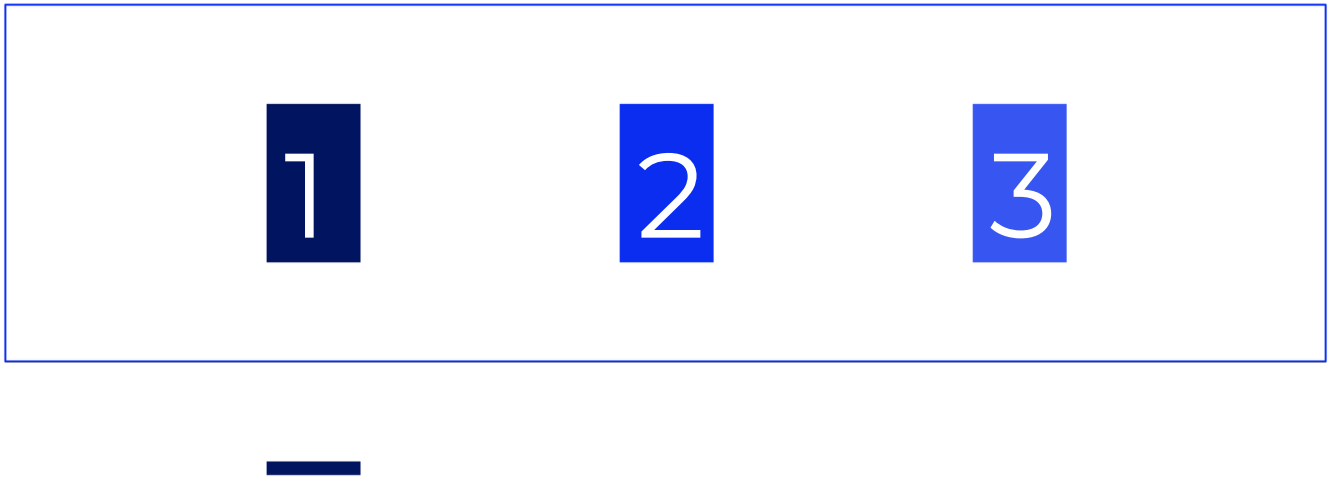


Nota: los elementos de los extremos no cuentan con espacio inicial (en el caso del primero) ni posterior (en el caso del último de la fila).

- **Space-evenly:** genera **espacio antes y después** de los elementos, **pero siempre igual,** por lo tanto, no hay espacios mayores a otros.

```
39  main{display: flex;justify-content: space-evenly;}
40  #uno{background-color: #041460}
41  #dos{background-color: #0a2ff1}
42  #tres{background-color: #3957f1}
43
```

El resultado será el siguiente:



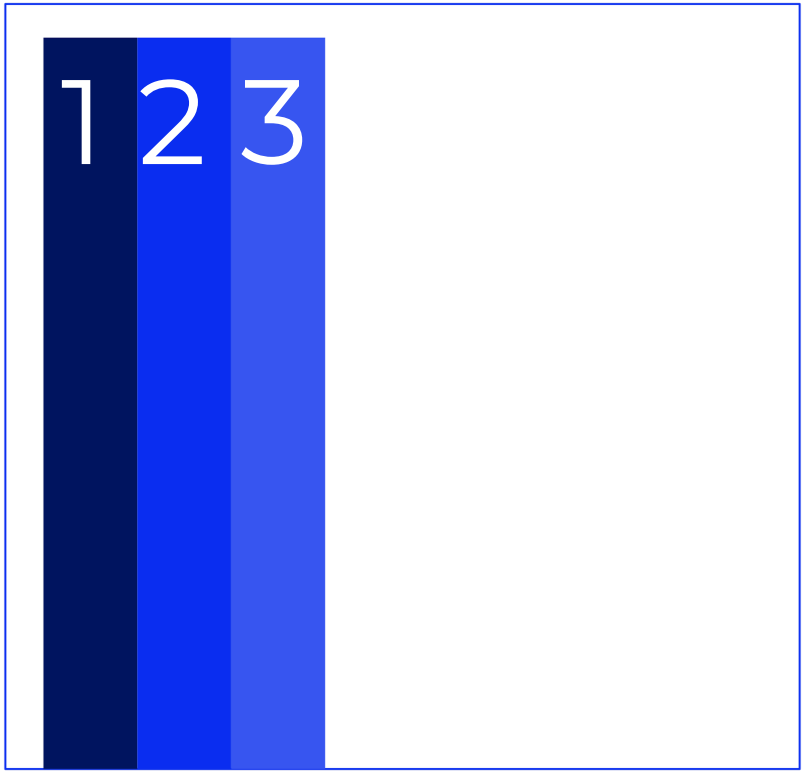
Flex: alineación vertical

La alineación vertical se trabajará con la propiedad **align-items** con los siguientes valores:

- **Stretch:** valor por defecto o predeterminado, es decir, sin indicar nada este será el valor.

```
3  main{display: flex; align-items: stretch;}
4  #uno{background-color: #041460}
5  #dos{background-color: #0a2ff1}
6  #tres{background-color: #3957f1}
7
```

El resultado será el siguiente. Los **elementos se estiran hasta cubrir todo el alto del contenedor,** (para lograr la siguiente referencia le hemos implementado al mail un height de 300px).



- **Flex-start:** los elementos **se ubican en la parte superior del contenedor** sin importar su height o alto.

```
 9  main{display: flex; align-items: flex-start; height: 300px;}
10  #uno{background-color: #041460}
11  #dos{background-color: #0a2ff1}
12  #tres{background-color: #3957f1}
13
```

El resultado será el siguiente:

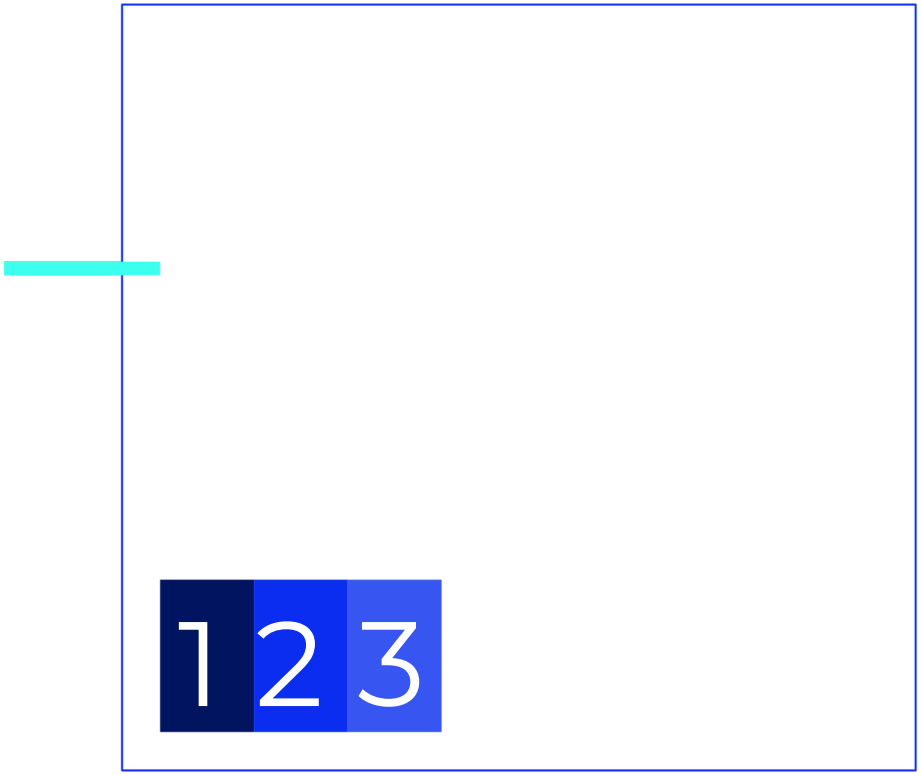
```
align-items: flex-start;
```

1 2 3

- **Flex-end:** los elementos **se ubican en la parte inferior del contenedor** sin importar su height o alto.

```
15  main{display: flex; align-items: flex-end; height: 300px;}
16  #uno{background-color: #041460}
17  #dos{background-color: #0a2ff1}
18  #tres{background-color: #3957f1}
19
```

El resultado será el siguiente.:



- **Center:** permite ubicar los elementos **hacia el centro del contenedor.**

```
21  main{display: flex; align-items: center;; height: 300px;}
22  #uno{background-color: #041460}
23  #dos{background-color: #0a2ff1}
24  #tres{background-color: #3957f1}
25
```

El resultado será el siguiente.:

```
align-items: center;
```