EXERCISES QUE TRABAJAREMOS EN LA CUE

- EXERCISE 1: MI PRIMER PROYECTO CON FLEXBOX.

EXERCISE 1: MI PRIMER PROYECTO CON FLEXBOX.

En el siguiente ejercicio, aprenderemos a implementar y utilizar flexbox en la creación de una página web.

Primero, crearemos nuestro proyecto, el cual consistirá en un documento HTML que contenga su estructura base; y de un archivo CSS, el cual debe ser invocado en el head de dicho documento HTML.

Posterior a esto, procederemos a agregar un div con clase `container` y 6 ítems (`div`) en su interior.

```
<!DOCTYPE html>
<html lang="es">
   <head>
      <meta charset="UTF-8">
      <meta http-equiv="X-UA-Compatible" content="IE=edge">
      <meta name="viewport" content="width=device-width, initial-
scale=1.0">
      <title>Flexbox</title>
      <link rel="stylesheet" href="assets/css/style.css">
   </head>
   <body>
      <div class="container">
         <div class="item item1">A</div>
         <div class="item item2">B</div>
         <div class="item item3">C</div>
         <div class="item item4">D</div>
         <div class="item item5">E</div>
         <div class="item item6">F</div>
      </div>
   </body>
</html>
```

Ahora, empezaremos a agregar estilo a este contenedor y a los ítems que se encuentran en su interior.

1º Agregaremos la propiedad `display: flex` al contenedor, y definiremos la orientación del eje principal, además, un color de fondo.

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

Si revisamos el adelanto de nuestro trabajo en el navegador, recibiremos el siguiente resultado:

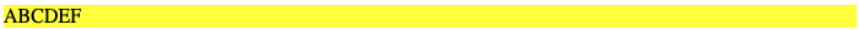

Imagen 1. Vista en el navegador con color de fondo y eje principal fila.

2º Ahora, contrastaremos el resultado anterior, con el que se obtendrá al definir: `flex-direction: column`.

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

Imagen 2. Vista en el navegador con color de fondo y eje principal columna.

3º Agregaremos un color de fondo a los ítems, para así poder contrastar de mejor manera el resultado, además de agregarles un borde para distinguir donde termina uno y donde comienza el siguiente.

```
.item {
        background-color: blueviolet;
        color: white;
        border: 5px solid white;
}
```

Con eso, obtendremos el siguiente resultado:

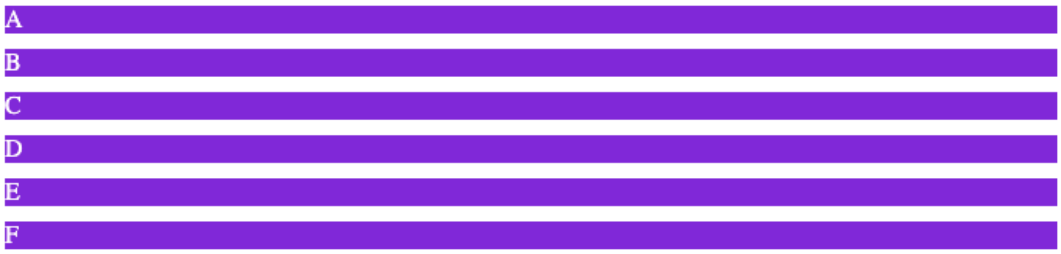

Imagen 3. Resultado luego de agregar color de fondo y bordes a los ítems.

4º Como se puede observar, no existe una separación entre un ítem y el siguiente, por lo que agregaremos un margen a cada uno de estos.

```
.item {
        background-color: blueviolet;
        color: white;
        border: 5px solid white;
        margin: 5px;
}
```

Imagen 4. Resultado luego de agregar margen a los ítems.

5º Ya teniendo nuestra base lista, empezaremos a practicar con otras propiedades de flexbox.

Primero, cambiaremos el valor por defecto de la propiedad `flex-wrap (flex-wrap: nowrap)` por `wrap`, y reduciremos el ancho de los ítems al `40%`.

```
.container {
        display: flex;
        flex-direction: column;
        background-color: yellow;
        flex-wrap: wrap;
}
.item {
        background-color: blueviolet;
        color: white;
        border: 5px solid white;
        margin: 5px;
        width: 40%;
}
```

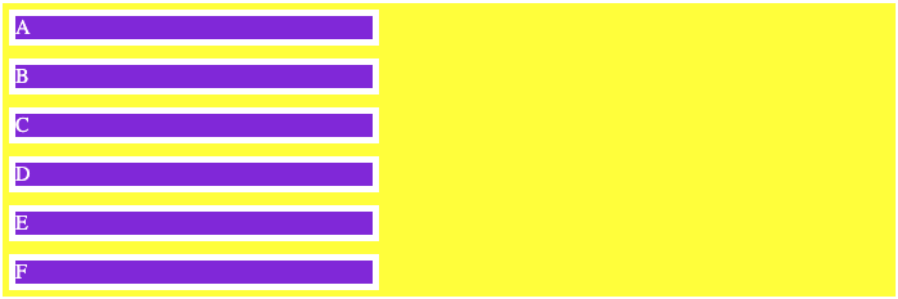

Imagen 5. Resultado luego de agregar `flex-wrap: wrap`, y de reducir el ancho de los ítems al `40%`.

Como puede observarse, no se evidencian más cambios que el ancho de los elementos; esto es así, ya que la dirección del eje principal está definida como columna, lo que implica que el flujo de los elementos será vertical siempre.

6º Cambiaremos `flex-direction` por `row`.

```
.container {
        display: flex;
        flex-direction: row;
        background-color: yellow;
        flex-wrap: wrap;
}
```

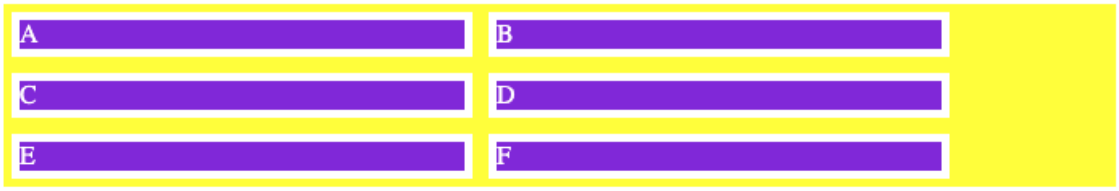

Imagen 6. Resultado luego de cambiar a `flex-direction: row`.

Aquí se puede notar, que los ítems ahora se acomodaron para ocupar el `40%` del ancho del contenedor en múltiples líneas.

7º ¿Qué pasaría si le cambiamos el valor a `flex-wrap` por `nowrap`?

Imagen 7. Propiedad `flex-wrap: nowrap`.

Los ítems ajustan su ancho para entrar todos en una sola línea y, por ende, no desbordar el contenedor.

8º Ahora, modificaremos la alineación de los ítems dentro del contenedor. Para esto, definiremos un ancho menor para ellos, y probaremos cada uno de los valores que puede tomar la propiedad **`justify-content`**.

```
.container {
        display: flex;
        flex-direction: row;
        background-color: yellow;
        flex-wrap: nowrap;
        justify-content: flex-start;
}

.item {
        background-color: blueviolet;
        color: white;
        border: 5px solid white;
        margin: 5px;
        width: 40px;
}
```

Imagen 8. justify-content: flex-start.

Imagen 9. justify-content: flex-end.

Imagen 10. justify-content: center.

Imagen 11. justify-content: space-between.

Imagen 12. justify-content: space-around.

9º Otra de las propiedades de gran relevancia es align-content. Aquí volveremos a definir un ancho de `40%, flex-wrap: wrap`, y le determinaremos una medida de altura al contenedor.

```
.container {
        display: flex;
        flex-direction: row;
        background-color: yellow;
        flex-wrap: wrap;
        height: 200px;
        align-content: flex-start;
}
.item {
        background-color: blueviolet;
        color: white;
        border: 5px solid white;
        margin: 5px;
        width: 40%;
}
```

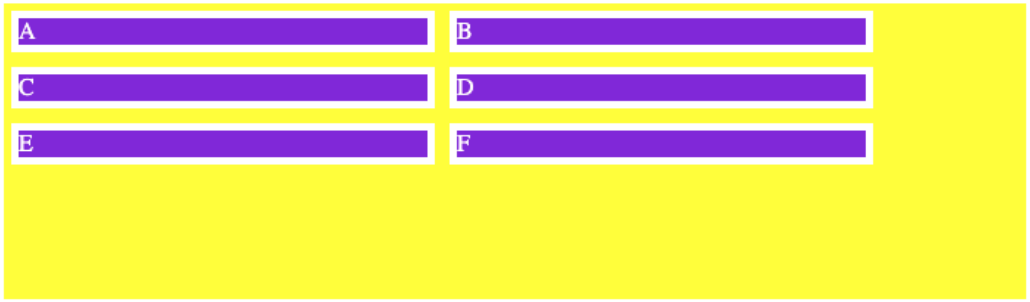

Imagen 13. align-content: flex-start.

Imagen 14. align-content: flex-end.

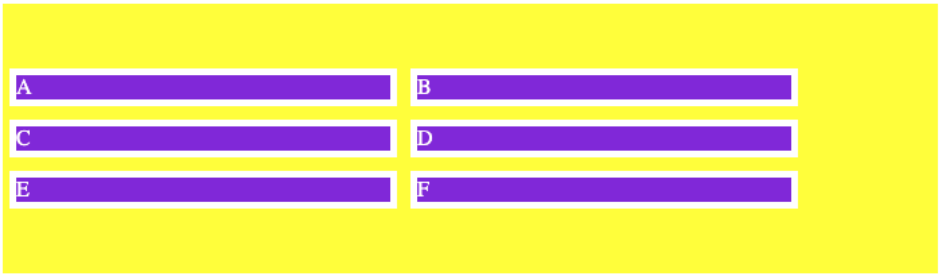

Imagen 15. align-content: center.

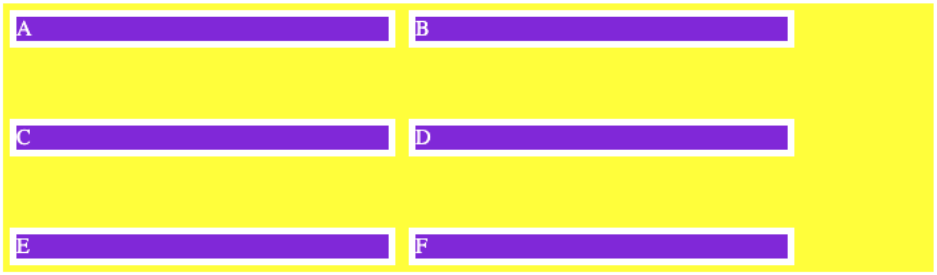

Imagen 16. align-content: space-between.

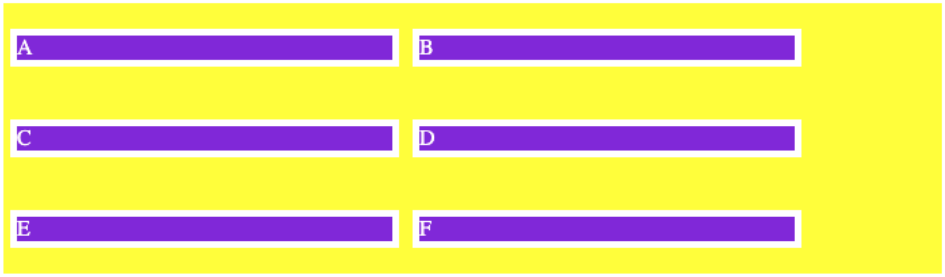

Imagen 17. align-content: space-around.

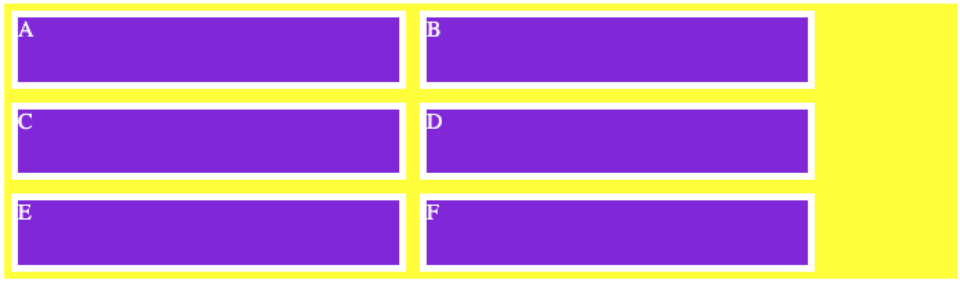

Imagen 18. align-content: stretch.

10º Por último, revisaremos la propiedad `align-items`, la cual se aplica al contenedor de los ítems que se quieren alterar. `align-self`, en cambio, debe aplicarse a cada uno de éstos que quiera modificarse.

Cambiaremos, además, la propiedad `flex-wrap` a `nowrap`, para observar de forma más clara las alteraciones que se hayan realizado.

```
.container {
        display: flex;
        flex-direction: row;
        background-color: yellow;
        flex-wrap: nowrap;
        height: 200px;
        align-items: flex-start;
}

.item {
```

```
        background-color: blueviolet;
        color: white;
        border: 5px solid white;
        margin: 5px;
        width: 40%;
}
```

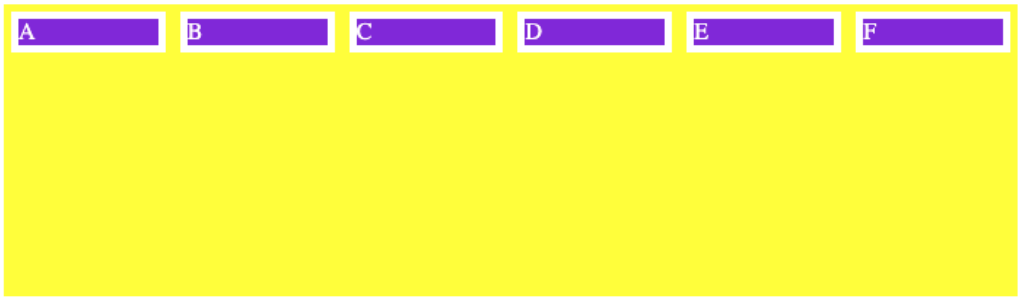

Imagen 19. align-items: flex-start.

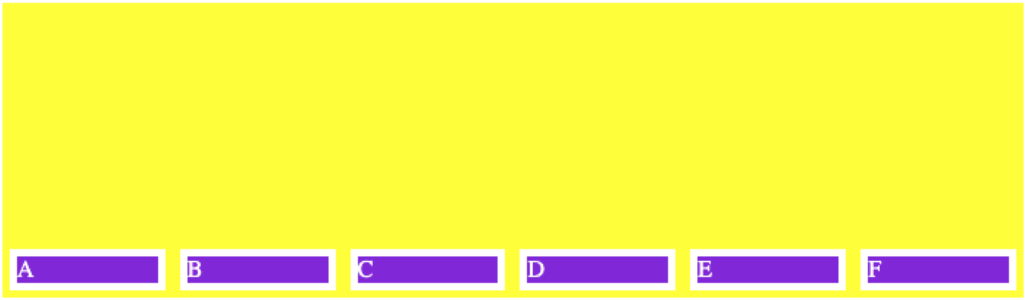

Imagen 20. align-items: flex-end.

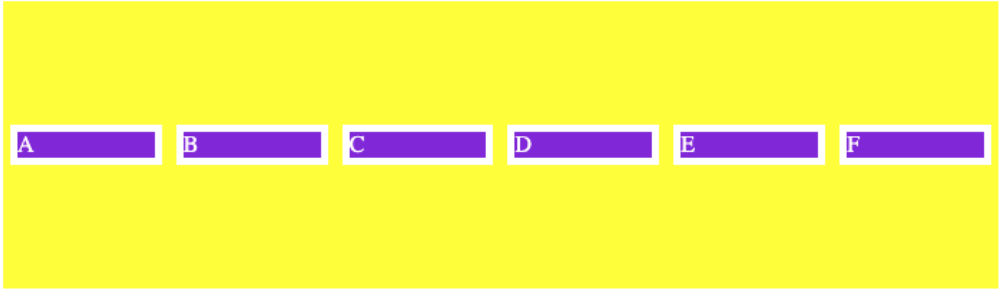

Imagen 21. align-items: center.

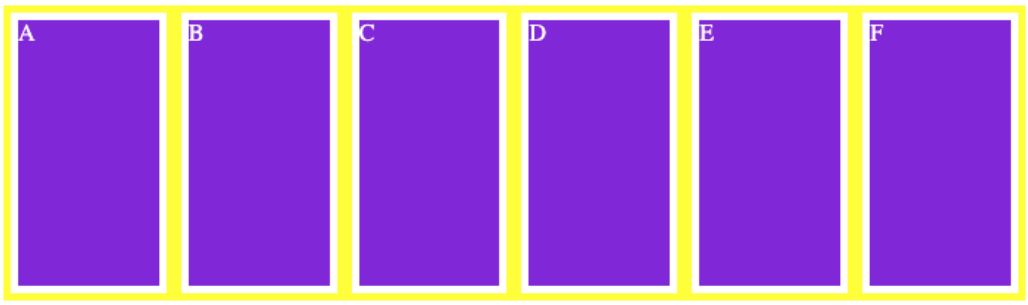

Imagen 22. align-items: stretch.

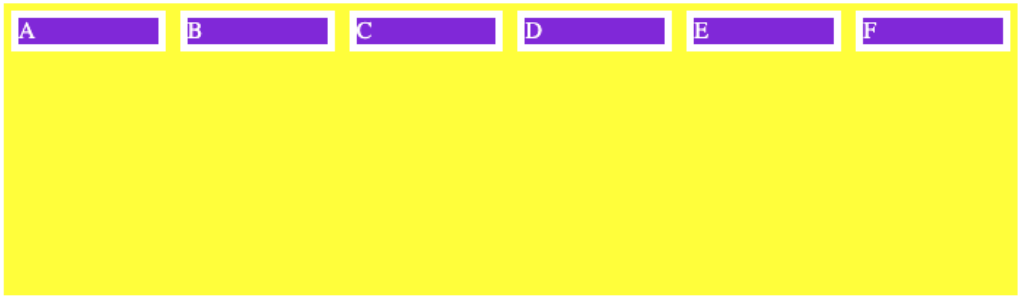

Imagen 23. Align-items: baseline.

En este ejercicio hemos aprendido cómo utilizar algunas propiedades de `flexbox`. Te invitamos a seguir realizando las actividades correspondientes a este CUE, y a continuar avanzando en el curso.