

Universidad Don Bosco
Facultad de Ingeniería
Escuela de Ingeniería en Computación

Asignatura: Lenguajes Interpretados en el Cliente

Ciclo: o2- 2023

Grupo: o1T

Docente: Ing. Guillermo Calderón.

TAREA 1.

Elaborado por:

Bonilla Sorto, Oscar Alexander	BS111203
Rodríguez Campos, Jonathan Alexander	RC181256
Romero López, César Armando	RL152226

Fecha de Entrega: 09 de septiembre de 2023

1. Flexbox

Flexbox (Flexible Box Layout) es un módulo de diseño CSS que permite distribuir el espacio entre los elementos de un contenedor de manera eficiente, incluso cuando los tamaños son desconocidos o dinámicos.

Flexbox es un modelo de diseño unidimensional que se utiliza para organizar elementos en filas o columnas. Los conceptos clave de Flexbox son el eje principal (main axis) y el eje secundario (cross axis).

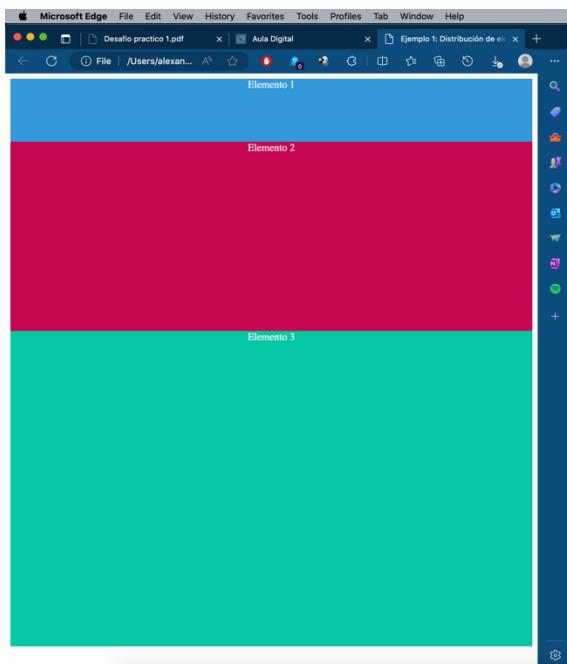
El eje principal se define mediante la propiedad `flex-direction` y establece la dirección en la que se colocan los elementos. El eje secundario es perpendicular al eje principal y se utiliza para alinear los elementos. Si el eje principal es horizontal, el secundario es vertical, y viceversa.

Entre las propiedades principales de Flexbox se tiene:

- `display`: establece el elemento como un contenedor flexible.
- `flex-direction`: establece la dirección del eje principal.
- `justify-content`: alinea los elementos a lo largo del eje principal.
- `align-items`: alinea los elementos a lo largo del eje secundario.
- `flex-wrap`: establece si los elementos deben envolverse o no en varias líneas.
- `flex-grow`: establece la capacidad de crecimiento de un elemento flexible.
- `flex-shrink`: establece la capacidad de reducción de un elemento flexible.
- `flex-basis`: establece el tamaño base de un elemento flexible.

Ejemplos de uso de Flexbox

Ejemplo 1:



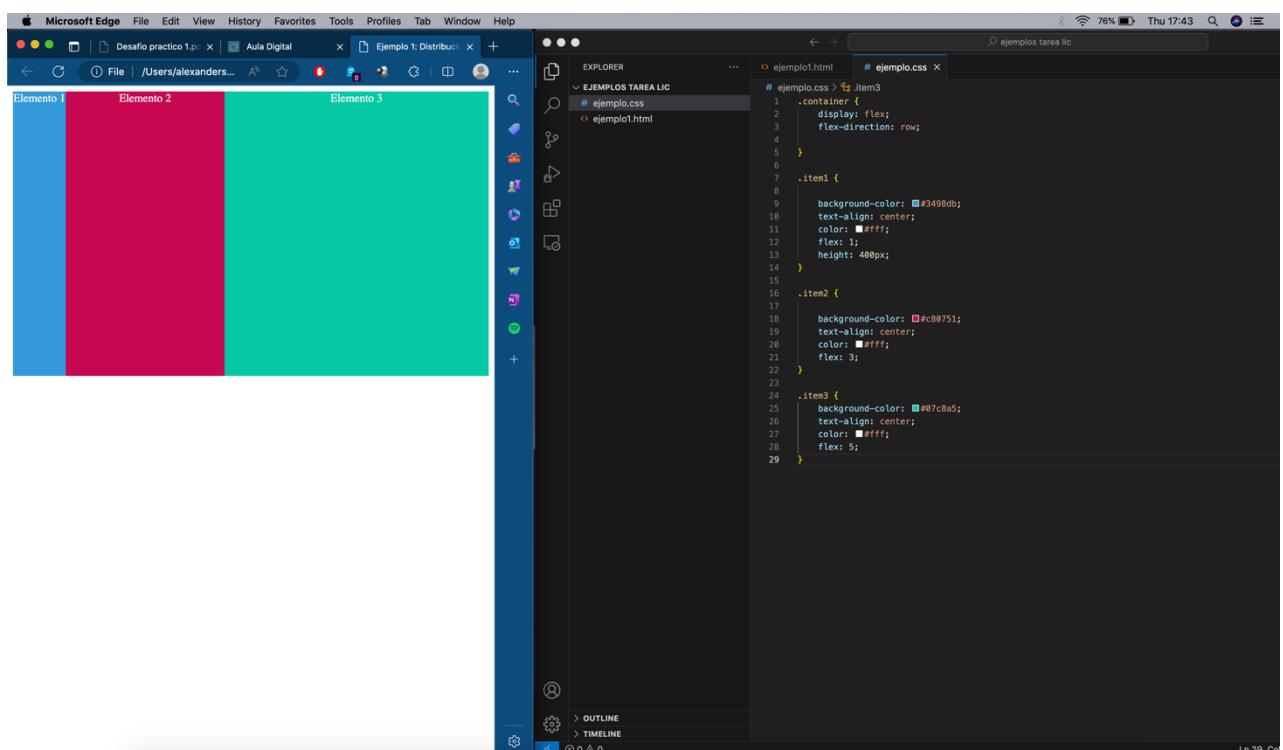
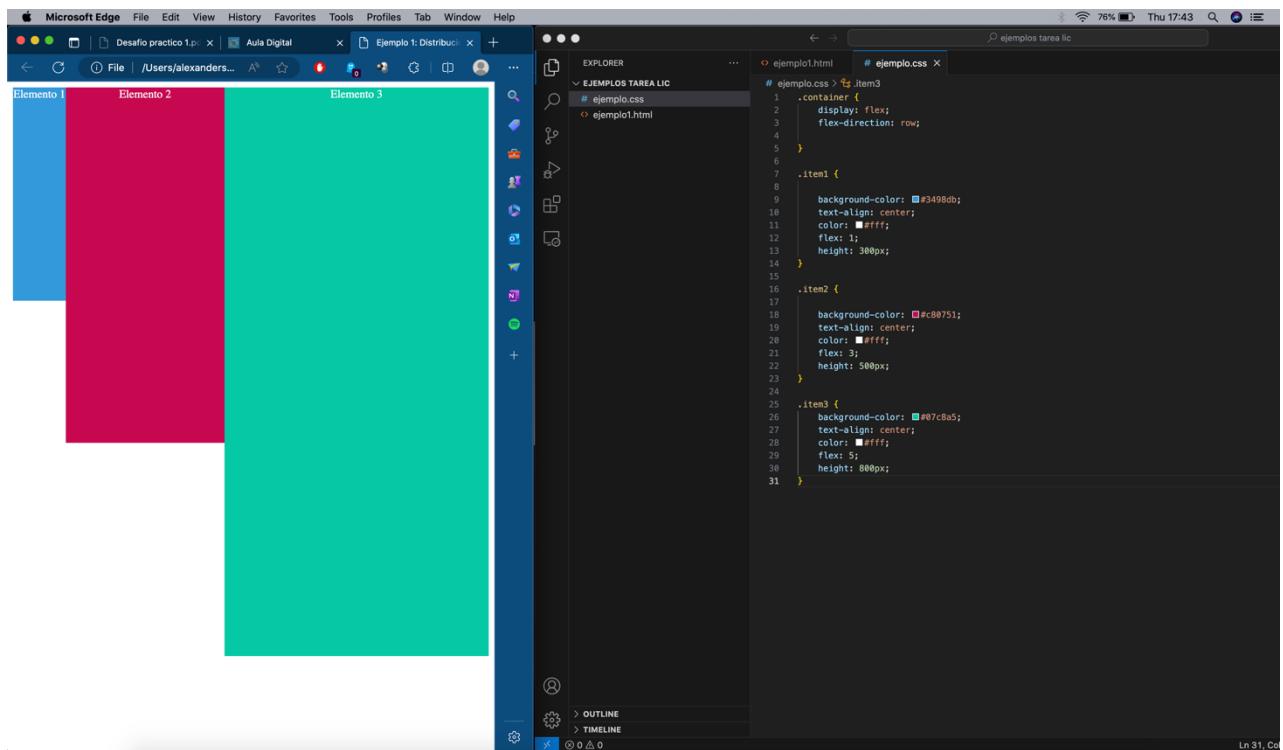
The screenshot shows a Microsoft Edge browser window with the title "Ejemplo 1: Distribución de espacio". The page displays three colored rectangular elements labeled "Elemento 1", "Elemento 2", and "Elemento 3" from top to bottom. Element 1 is blue (#3498db), Element 2 is red (#e64a89), and Element 3 is green (#27c7a5). The browser's developer tools are open, showing the CSS code for the example. The CSS file contains the following code:

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

.item1 {
    background-color: #3498db;
    text-align: center;
    color: #ffff;
    height: 100px;
}

.item2 {
    background-color: #e64a89;
    text-align: center;
    color: #ffff;
    height: 300px;
}

.item3 {
    background-color: #27c7a5;
    text-align: center;
    color: #ffff;
    height: 500px;
}
```



Ejemplo 2

The screenshot shows the Visual Studio Code interface with two panes. The left pane displays the file `ejemplo2.css` containing the following CSS:

```
# ejemplo2.css
.container {
    display: flex;
    justify-content: center;
    align-items: center;
    height: 400px;
}

.item {
    width: 100px;
    height: 400px;
    background-color: #ff5b1f;
    text-align: center;
    color: #ffff;
}

.item2 {
    width: 100px;
    height: 300px;
    background-color: #ffff00;
    text-align: center;
    color: #ffff;
}

.item3 {
    width: 100px;
    height: 100px;
    background-color: #8b1fff;
    text-align: center;
    color: #ffff;
}
```

The right pane shows a browser window displaying three colored boxes (orange, cyan, and purple) arranged vertically in the center of a 400px high container.

The screenshot shows the Visual Studio Code interface with two panes. The left pane displays the file `ejemplo2.css` containing the following CSS:

```
# exemplo2.css
.container {
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    height: 400px;
}

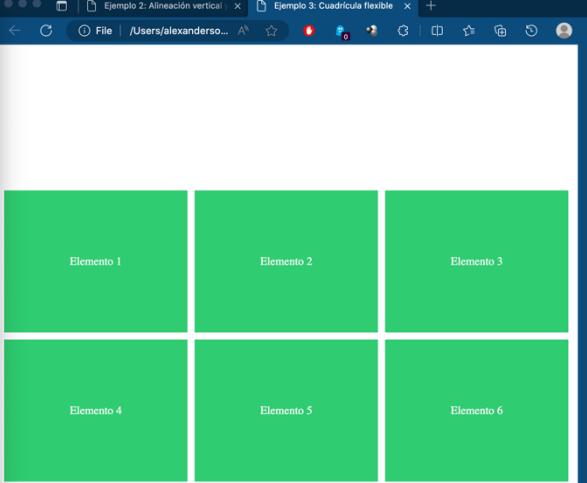
.item {
    width: 400px;
    height: 100px;
    background-color: #ff5b1f;
    text-align: center;
    color: #ffff;
}

.item2 {
    width: 300px;
    height: 100px;
    background-color: #ffff00;
    text-align: center;
    color: #ffff;
}

.item3 {
    width: 100px;
    height: 100px;
    background-color: #8b1fff;
    text-align: center;
    color: #ffff;
}
```

The right pane shows a browser window displaying three colored boxes (orange, cyan, and purple) arranged vertically in the center of a 400px high container.

Ejemplo 3

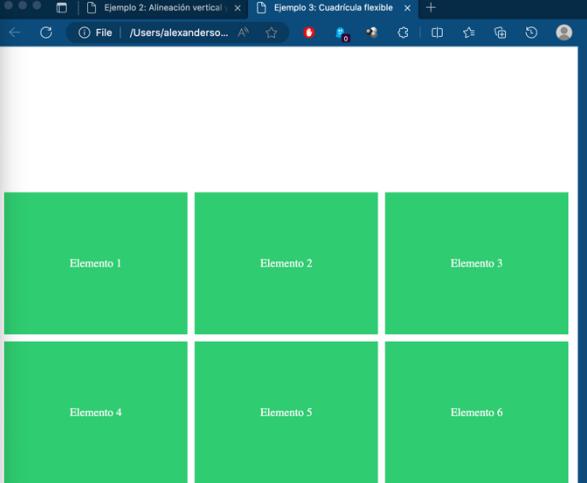


The screenshot shows a 2x3 grid of six green rectangular cards, each labeled with a name: Elemento 1, Elemento 2, Elemento 3 in the top row; and Elemento 4, Elemento 5, Elemento 6 in the bottom row. This visual representation corresponds to the layout defined in the CSS code below.

```
ejemplo3.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Ejemplo 3: Cuadricula flexible</title>
    <link rel="stylesheet" href="ejemplo3.css">
</head>
<body>
    <div class="container">
        <div class="item">Elemento 1</div>
        <div class="item">Elemento 2</div>
        <div class="item">Elemento 3</div>
        <div class="item">Elemento 4</div>
        <div class="item">Elemento 5</div>
        <div class="item">Elemento 6</div>
    </div>
</body>
</html>
```

```
ejecto3.css
.container {
    display: flex;
    flex-wrap: wrap;
    align-items: center;
    height: 400px;
    margin-top: 20px;
}

.item {
    flex: 1;
    min-width: calc(33.33% - 10px); /* Tres columnas con margen */
    margin: 5px;
    background-color: #2ecc71;
    text-align: center;
    color: white;
    align-items: center; /* Centra verticalmente */
    line-height: 20px; /* mantiene la altura de los elementos */
}
```



The screenshot shows the same 2x3 grid of six green rectangular cards, but now they are vertically aligned within their respective columns. This visual effect is achieved by applying the `align-items: center;` property to the `.item` class in the CSS code below.

```
ejemplo3.css
.container {
    display: flex;
    flex-wrap: wrap;
    align-items: center;
    height: 400px;
    margin-top: 20px;
}

.item {
    flex: 1;
    min-width: calc(33.33% - 10px); /* Tres columnas con margen */
    margin: 5px;
    background-color: #2ecc71;
    text-align: center;
    color: white;
    align-items: center; /* Centra verticalmente */
    line-height: 20px; /* mantiene la altura de los elementos */
}
```

2. CSS Grid

CSS Grid es un módulo de diseño CSS que permite crear diseños de cuadrícula avanzados y estructurados. A diferencia de Flexbox, que es un modelo de diseño unidimensional, CSS Grid es un modelo de diseño bidimensional que permite trabajar con filas (rows) y columnas (columns) al mismo tiempo.

Con CSS Grid, puedes controlar la posición de los elementos en ambas direcciones (filas y columnas), lo que permite diseños más complejos. En Flexbox, el control se limita a una sola dirección.

CSS Grid es ideal para crear diseños de cuadrícula complejos, como diseños de tablas, mosaicos y diseños de revistas. Flexbox es más adecuado para la alineación y distribución de elementos en una sola línea.

Propiedades clave de CSS Grid son:

- grid-template-columns: establece el ancho de las columnas de la cuadrícula.
- grid-template-rows: establece la altura de las filas de la cuadrícula.
- grid-gap: establece el espacio entre las celdas de la cuadrícula.
- grid-column-start: establece la posición de inicio de una celda en la cuadrícula.
- grid-column-end: establece la posición final de una celda en la cuadrícula.
- grid-row-start: establece la posición de inicio de una fila en la cuadrícula.
- grid-row-end: establece la posición final de una fila en la cuadrícula.

Para crear diseños de cuadrícula avanzados y estructurados con CSS Grid, se puede utilizar las siguientes propiedades:

- **Para crear una cuadrícula simple**, se puede utilizar la propiedad grid-template-columns para establecer el ancho de las columnas y la propiedad grid-template-rows para establecer la altura de las filas.
- **Para crear una cuadrícula con elementos de diferentes tamaños**, se puede utilizar las propiedades grid-column-start, grid-column-end, grid-row-start y grid-row-end para establecer la posición de cada elemento en la cuadrícula.
- **Para crear una cuadrícula con elementos que se superponen**, se puede utilizar las propiedades z-index y grid-template-areas para establecer el orden de apilamiento de los elementos.

Ejemplo 1

The screenshot shows a code editor interface with two panes. The left pane displays the CSS code for 'ejemplo4.css':

```
# ejemplo4.css
# ejemplo4.html > <html> > <head> > <link href="ejemplo4.css" rel="stylesheet">
.item {
    grid-template-columns: repeat(3, 1fr); /* Crea 3 columnas de igual ancho */
    grid-gap: 10px; /* Espacio entre las celdas */
    margin-top: 30px;
}
.item {
    background-color: #dbd334;
    text-align: center;
    color: #0010ff;
    padding: 10px;
}
```

The right pane shows a 3x3 grid layout with yellow cells labeled 'Celda 1' through 'Celda 9'. The grid has 3 columns and 3 rows.

La unidad "fr" en CSS Grid significa "fracción de espacio disponible" o "fracción de fila/columna". Es una unidad que se utiliza para especificar proporciones de espacio en una cuadrícula de CSS Grid.

The screenshot shows a code editor interface with two panes. The left pane displays the generated HTML code from 'ejemplo4.css':

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>CSS Grid Ejemplo 1</title>
<link rel="stylesheet" href="ejemplo4.css">
</head>
<body>
<div class="grid-container">
<div class="item">Celda 1</div>
<div class="item">Celda 2</div>
<div class="item">Celda 3</div>
<div class="item">Celda 4</div>
<div class="item">Celda 5</div>
<div class="item">Celda 6</div>
<div class="item">Celda 7</div>
<div class="item">Celda 8</div>
<div class="item">Celda 9</div>
</div>
</body>
</html>
```

The right pane shows the same 3x3 grid layout with yellow cells labeled 'Celda 1' through 'Celda 9'.

Ejemplo 2

The screenshot shows the VS Code interface with two panes. The left pane displays the file `ejemplo5.css` containing the following CSS:

```
ejemplo5.css
# ejemplo5.html > ...
# ejemplo5.css
# ejemplo1.html
# exemplo2.css
# exemplo3.css
# exemplo4.css
# exemplo5.css
# exemplo6.css
# exemplo6.html

.grid-container {
    display: grid;
    grid-template-columns: 1fr 2fr; /* Define dos columnas con proporciones */
    grid-template-rows: auto auto; /* Las filas se ajustarán automáticamente */
    grid-gap: 10px; /* Espacio entre las celdas */
    justify-content: center; /* Centra horizontalmente en el contenido */
    align-content: center; /* Centra verticalmente en el contenido */
    height: 400px; /* Altura fija para el contenedor */
}

.item {
    background-color: #3498db;
    text-align: center;
    color: white;
    padding: 10px;
}

/* Estilos específicos para cada celda */
.item:nth-child(1) {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}

.item:nth-child(2) {
    grid-column: span 2; /* Ocupa dos columnas */
    grid-row: span 1; /* Ocupa una fila */
}

.item:nth-child(3) {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}

.item:nth-child(4) {
    grid-column: span 2; /* Ocupa dos columnas */
    grid-row: span 1; /* Ocupa una fila */
}
```

The right pane shows the rendered output in a browser window, displaying four blue rectangular cells labeled "Celda 1", "Celda 2", "Celda 3", and "Celda 4" from top to bottom. Each cell is centered both horizontally and vertically within its respective grid column or row.

The screenshot shows the VS Code interface with two panes. The left pane displays the file `ejemplo5.css` containing the following CSS:

```
ejemplo5.css
# ejemplo5.html > ...
# exemplo5.css
# exemplo1.html
# exemplo2.css
# exemplo3.css
# exemplo4.css
# exemplo5.css
# exemplo6.css
# exemplo6.html

.grid-container {
    display: grid;
    grid-template-columns: 1fr 2fr; /* Define dos columnas con proporciones */
    grid-template-rows: auto auto; /* Las filas se ajustarán automáticamente */
    grid-gap: 10px; /* Espacio entre las celdas */
    justify-content: center; /* Centra horizontalmente en el contenido */
    align-content: center; /* Centra verticalmente en el contenido */
    height: 400px; /* Altura fija para el contenedor */
}

.item {
    background-color: #3498db;
    text-align: center;
    color: white;
    padding: 10px;
}

/* Estilos específicos para cada celda */
.item:nth-child(1) {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}

.item:nth-child(2) {
    grid-column: span 2; /* Ocupa dos columnas */
    grid-row: span 1; /* Ocupa una fila */
}

.item:nth-child(3) {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}

.item:nth-child(4) {
    grid-column: span 2; /* Ocupa dos columnas */
    grid-row: span 1; /* Ocupa una fila */
}
```

The right pane shows the rendered output in a browser window, displaying four blue rectangular cells labeled "Celda 1", "Celda 2", "Celda 3", and "Celda 4" from top to bottom. Each cell is centered both horizontally and vertically within its respective grid column or row.

Los elementos celda 1 y celda 3 ocupan una columna cada uno y los elementos celda 2 y celda 4 ocupan 2 columnas cada uno. Los elementos se centran horizontal y verticalmente en el contenedor.

Ejemplo 3

The screenshot shows the VS Code interface with two tabs open: 'ejemplo.css' and 'ejemplo6.html'. The 'ejemplo.css' tab contains the following CSS code:

```
# EJEMPLOS TAREA LIC
# ejemplo.css > ejemplos tarea lic
# ejemplo1.html
# ejemplo2.css
# ejemplo2.html
# ejemplo3.css
# ejemplo3.html
# ejemplo4.css
# ejemplo4.html
# ejemplo5.css
# ejemplo5.html
# ejemplo6.css
# ejemplo6.html

# ejemplo6.css > #grid-container
# grid-container {
    display: grid;
    grid-template-columns: repeat(3, 1fr); /* Tres columnas de igual anchura */
    grid-template-rows: repeat(2, 1fr); /* Dos filas de igual altura */
    grid-gap: 10px; /* Espacio entre las celdas */
    align-content: space-between; /* Espaciado verticalmente entre las filas */
    align-items: center; /* Centrado verticalmente en el contenido */
    height: 400px; /* Altura fija para el contenedor */
    margin-top: 240px;
}

.item {
    background-color: #3498db;
    text-align: center;
    color: white;
    padding: 10px;
}

.item1 {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}

.item2 {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 2; /* Ocupa dos filas */
}

.item3 {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}

.item4 {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 2; /* Ocupa dos filas */
}

.item5 {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}

.item6 {
    grid-column: span 1; /* Ocupa una columna */
    grid-row: span 1; /* Ocupa una fila */
}
```

The 'ejemplo6.html' tab shows a grid layout with six items labeled Celda 1 through Celda 6. The layout is as follows:

Celda 1	Celda 2	Celda 3
Celda 4		Celda 5
		Celda 6

The screenshot shows the VS Code interface with two tabs open: 'ejemplo.css' and 'ejemplo6.html'. The 'ejemplo.css' tab contains the same CSS code as the previous screenshot.

The 'ejemplo6.html' tab contains the following HTML code:

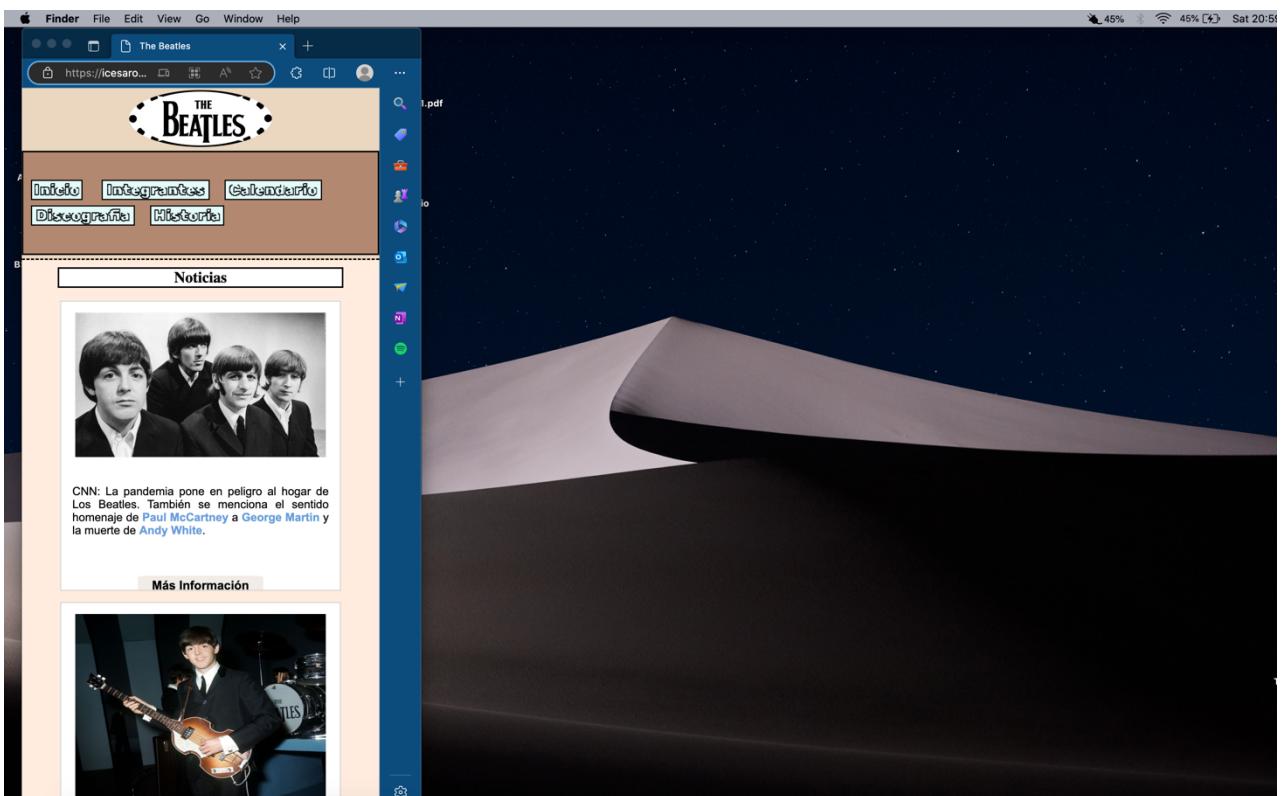
```
# EJEMPLOS TAREA LIC
# ejemplo.css > ejemplos tarea lic
# ejemplo1.html
# ejemplo2.css
# ejemplo2.html
# ejemplo3.css
# ejemplo3.html
# ejemplo4.css
# ejemplo4.html
# ejemplo5.css
# ejemplo5.html
# ejemplo6.css
# ejemplo6.html

# ejemplo6.html > html
html {
    doctype html;
    html lang="en";
    head {
        meta charset="UTF-8";
        meta name="viewport" content="width=device-width, initial-scale=1.0";
        title CSS Grid Ejemplo 3;
        link rel="stylesheet" href="ejemplo6.css";
    }
    body {
        div.grid-container {
            div.item.item1.item1-Celda 1;
            div.item.item2.item2-Celda 2;
            div.item.item3.item3-Celda 3;
            div.item.item4.item4-Celda 4;
            div.item.item5.item5-Celda 5;
            div.item.item6.item6-Celda 6;
        }
    }
}
```

The 'ejemplo6.html' tab shows the same grid layout with six items labeled Celda 1 through Celda 6, matching the visual representation in the first screenshot.

El valor `span` se utiliza en CSS Grid para especificar cuántas columnas o filas debe abarcar un elemento en una cuadrícula. La palabra clave `span` seguida de un número indica cuántas unidades de columnas o filas debe ocupar el elemento, comenzando desde su ubicación actual en la cuadrícula.

3. CAPTURAS FUNCIONAMIENTO SITIO WEB GRUPO MUSICAL

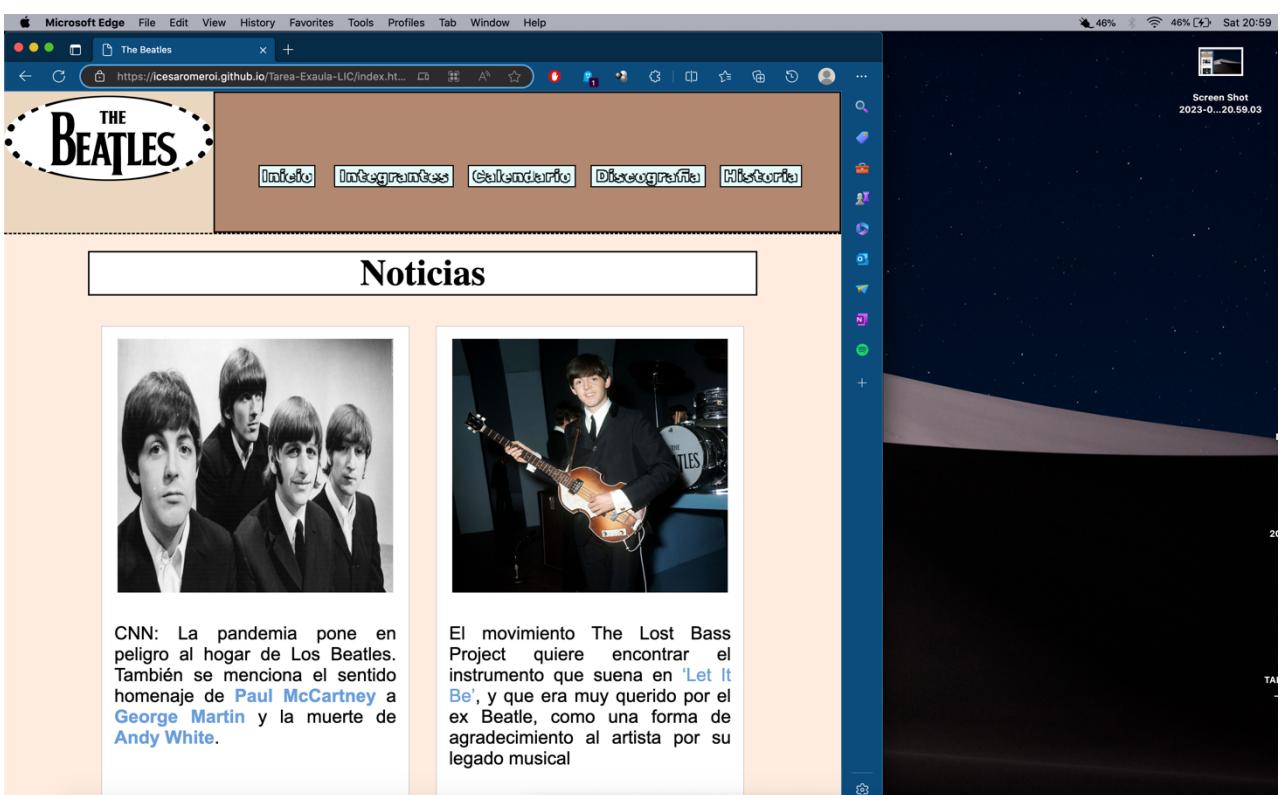


The Beatles

Noticias

CNN: La pandemia pone en peligro al hogar de Los Beatles. También se menciona el sentido homenaje de Paul McCartney a George Martin y la muerte de Andy White.

Más Información



The Beatles

Noticias

CNN: La pandemia pone en peligro al hogar de Los Beatles. También se menciona el sentido homenaje de Paul McCartney a George Martin y la muerte de Andy White.

El movimiento The Lost Bass Project quiere encontrar el instrumento que suena en 'Let It Be', y que era muy querido por el ex Beatle, como una forma de agradecimiento al artista por su legado musical

The Beatles

https://icesaromeroi.github.io/Tarea-Exaula-LIC/index.html

Inicio Integrantes Calendario Discografía Historia

Noticias





CNN: La pandemia pone en peligro al hogar de Los Beatles. También se menciona el sentido homenaje de [Paul McCartney](#) a [George Martin](#) y la muerte de [Andy White](#).

El movimiento The Lost Bass Project quiere encontrar el instrumento que suena en ['Let It Be'](#), y que era muy querido por el ex Beatle, como una forma de homenaje al artista por su legado musical.

Microsoft Edge

File Edit View History Favorites Tools Profiles Tab Window Help

Calendario

https://icesaro...

THE BEATLES

Inicio Integrantes Calendario Discografía Historia

Calendario de Conciertos



Fecha: 9 de febrero de 1964

Lugar:
The Ed Sullivan Show en Nueva York, Estados Unidos.



Fecha: 15 de agosto de 1965

Lugar:
Shea Stadium en Nueva York, Estados Unidos.



Microsoft Edge File Edit View History Favorites Tools Profiles Tab Window Help

Calendario https://icesaromeroi.github.io/Tarea-Exaula-LIC/Calendario.html

THE BEATLES

Inicio Integrantes Calendario Discografia Historia

Calendario de Conciertos

	
Fecha: 9 de febrero de 1964	Fecha: 15 de agosto de 1965
Lugar: The Ed Sullivan Show en Nueva York, Estados Unidos.	Lugar: Shea Stadium en Nueva York, Estados Unidos.

20 20 20 20

Microsoft Edge File Edit View History Favorites Tools Profiles Tab Window Help

Calendario https://icesaromeroi.github.io/Tarea-Exaula-LIC/Calendario.html

THE BEATLES

Inicio Integrantes Calendario Discografia Historia

Calendario de Conciertos

		
Fecha: 9 de febrero de 1964	Fecha: 15 de agosto de 1965	Fecha: 30 de enero de 1969
Lugar: The Ed Sullivan Show en Nueva York, Estados Unidos.	Lugar: Shea Stadium en Nueva York, Estados Unidos.	Lugar: Techo de Apple Corps en Londres, Reino Unido (concierto improvisado conocido como el "concierto en la azotea").

4. ENLACES

- **Repositorio**

<https://github.com/ICesaRomeroI/Tarea-Exaula-LIC>

- **GitHub Pages**

<https://icesaromeroi.github.io/Tarea-Exaula-LIC/index.html>