LEMONCODE

HTML

Grid y Flex avanzado

En este vídeo exploraremos cómo combinar Flexbox y GridLayout en CSS para crear diseños avanzados y responsive. Estas dos herramientas son poderosas y nos permiten crear y controlar la disposición de elementos en una página web.

Flexbox

Flexbox es un modelo de diseño unidimensional que maneja elementos en filas o columnas. Es útil para estructuras pequeñas o para organizar elementos internos de un componente.

Ejemplo Flexbox

```
<!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" />
        <link rel="stylesheet" href="style.css" />
        <title>Flex y Grid Avanzado</title>
    </head>
    <body>
        <div class="flex-container">
            <div class="item">1</div>
            <div class="item">2</div>
            <div class="item">3</div>
        </div>
    </body>
</html>
```

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

.item {
    padding: 20px;
    background-color: lightgreen;
    border: 1px solid black;
}
```

Grid Layout

Grid Layout es un modelo de diseño bidimensional que maneja elementos en filas y columnas. Es ideal para organizar grandes áreas de contenido o la estructura general de un sitio web.

Ejemplo de Grid Layout

na:

```
<!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" />
        <link rel="stylesheet" href="style.css" />
        <title>Flex y Grid Avanzado</title>
    </head>
    <body>
        <div class="grid-container">
            <div class="item">1</div>
            <div class="item">2</div>
            <div class="item">3</div>
            <div class="item">4</div>
            <div class="item">5</div>
            <div class="item">6</div>
        </div>
    </body>
</html>
```

```
.grid-container {
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    gap: 10px;
    background-color: lightblue;
}

.item {
    padding: 20px;
    background-color: lightgreen;
    border: 1px solid black;
}
```

Con estos estilos, los elementos se organizarán en una cuadrícula de 3 columnas y se separarán por un espacio de 10 píxeles.

Combinar Flexbox y GridLayout

La combinación de Flexbox y GridLayout nos permite crear diseños más avanzados y responsive. Podemos aprovechar lo mejor de ambos sistemas para lograr una gran adaptabilidad y complejidad en nuestros diseños web.

Ejemplo de combinación de Flexbox y GridLayout

Imaginemos que queremos diseñar un sitio web con una barra de navegación (usando Flexbox) y un área de contenido principal con una estructura de rejilla (usando GridLayout). Aquí tienes un ejemplo de cómo podríamos hacerlo:

```
<!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" />
        <link rel="stylesheet" href="style.css" />
        <title>Flex y Grid Avanzado</title>
    </head>
    <body>
        <nav class="navbar">
            <a href="#">Inicio</a>
            <a href="#">Productos</a>
            <a href="#">Contacto</a>
        </nav>

        <div class="content">
            <div>Contenido 1</div>
            <div>Contenido 2</div>
            <div>Contenido 3</div>
            <div>Contenido 4</div>
        </div>
    </body>
</html>
```

```
:root {
    --size: 1rem;
}

.navbar {
    display: flex;
    justify-content: space-between;
    background-color: #333;
    color: white;
    padding: var(--size);
}

.navbar a {
    text-decoration: none;
```

```
    padding: 0.5rem var(--size);
    color: white;
}

.content {
    display: grid;
    grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
    gap: var(--size);
    padding: var(--size);
}

.content > div {
    background-color: #f1f1f1;
    padding: var(--size);
    text-align: center;
}
```

Con este ejemplo, hemos combinado Flexbox para la barra de navegación y Grid Layout para el área de contenido principal. De esta manera, hemos creado un diseño más complejo y responsive.

Recuerda que al combinar Flexbox y GridLayout, puedes aprovechar lo mejor de ambos sistemas para crear diseños avanzados y responsive en tus proyectos web. Flexbox es ideal para elementos unidimensionales, mientras que Grid Layout es perfecto para estructuras bidimensionales.