Layout con Flexbox y Layout con Grid

Julian F. Latorre

October 3, 2024

Contenido

Introducción

- Flexbox: Modelo de diseño unidimensional
- Grid: Sistema de diseño bidimensional
- Ambos permiten crear diseños complejos y responsivos
- Menos código, mayor flexibilidad

Conceptos básicos de Flexbox

- Contenedor flex (flex container)
- Elementos flex (flex items)
- Eje principal y eje transversal

```
.container {
    display: flex;
}
```

Propiedades del contenedor flex

- `flex-direction`
- `justify-content`
- `align-items`
- `flex-wrap`
- `align-content`

Ejemplo de propiedades del contenedor flex

```
.container {
    display: flex;
    flex-direction: row;
    justify-content: space-between;
    align-items: center;
    flex-wrap: wrap;
}
```

Propiedades de los elementos flex

- `flex-grow`
- `flex-shrink`
- `flex-basis`
- `flex` (atajo)
- `align-self`
- `order`

Ejemplo de propiedades de los elementos flex

```
.item {
    flex: 1 0 200px;
    align-self: flex-start;
    order: 2;
}
```

Ejemplo Flexbox (HTML)

```html
<nav class="navbar">
    <div class="logo">Logo</div>
    <ul class="nav-links">
        <li><a href="#">Inicio</a></li>
        <li><a href="#">Servicios</a></li>
        <li><a href="#">Contacto</a></li>
    </ul>
</nav>
```

Ejemplo Flexbox (CSS)

```css
.navbar {
    display: flex;
    justify-content: space-between;
    align-items: center;
    padding: 1rem;
    background-color: #f8f9fa;
}

.nav-links {
    display: flex;
    list-style: none;
    gap: 1rem;
}
```

Conceptos básicos de Grid

- Contenedor grid (grid container)
- Elementos grid (grid items)
- Líneas de grid
- Celdas de grid
- Áreas de grid

```
.container {
    display: grid;
}
```

Propiedades del contenedor grid

- `grid-template-columns`
- `grid-template-rows`
- `grid-template-areas`
- `grid-gap`
- `justify-items`
- `align-items`

Ejemplo de propiedades del contenedor grid

```
.container {
    display: grid;
    grid-template-columns: repeat(3, 1fr);
    grid-template-rows: auto 1fr auto;
    grid-gap: 20px;
    justify-items: center;
    align-items: center;
}
```

Propiedades de los elementos grid

- `grid-column`
- `grid-row`
- `grid-area`
- `justify-self`
- `align-self`

Ejemplo de propiedades de los elementos grid

```
.item {
    grid-column: 1 / 3;
    grid-row: 2 / 4;
    justify-self: start;
    align-self: end;
}
```

Ejercicio práctico con Grid

```
<div class="grid-container">
    <header>Encabezado</header>
    <nav>Navegaci n</nav>
    <main>Contenido principal</main>
    <aside>Barra lateral</aside>
    <footer>Pie de p gina</footer>
</div>
```

Ejercicio práctico con Grid (CSS)

```css
.grid-container {
    display: grid;
    grid-template-areas:
        "header header header"
        "nav main aside"
        "footer footer footer";
    grid-template-columns: 200px 1fr 200px;
    grid-template-rows: auto 1fr auto;
    min-height: 100vh;
}

header { grid-area: header; }
nav { grid-area: nav; }
main { grid-area: main; }
aside { grid-area: aside; }
footer { grid-area: footer; }
```

Conclusión y comparación

- Flexbox: ideal para diseños unidimensionales
- Grid: perfecto para diseños bidimensionales
- Flexbox: mayor control sobre alineación y distribución
- Grid: diseños complejos con menos código
- Ambos pueden usarse juntos para diseños más poderosos

¡Gracias!

¿Preguntas?