SUBMENÚ SIN USAR JAVASCRIPT, UTILIZANDO SOLO HTML Y CSS

Es posible abrir un submenú sin usar JavaScript, utilizando solo HTML y CSS, aunque con algunas limitaciones en términos de interacción y accesibilidad. A continuación, te explico cómo hacerlo:

---

1. ESTRUCTURA HTML

Utiliza una estructura semántica para el menú y el submenú. Asegúrate de que el submenú esté anidado dentro del elemento del menú principal.

```
<nav aria-label="Menú principal">
  <ul>
    <li>
      <a href="#" aria-haspopup="true" aria-expanded="false">Menú
1</a>
      <ul>
        <li><a href="#">Submenú 1.1</a></li>
        <li><a href="#">Submenú 1.2</a></li>
      </ul>
    </li>
    <li><a href="#">Menú 2</a></li>
  </ul>
</nav>
```

- aria-haspopup="true": Indica que el elemento tiene un submenú.
- aria-expanded="false": Indica que el submenú está cerrado inicialmente.

---

2. ESTILOS CSS

Utiliza CSS para controlar la visibilidad del submenú. Puedes usar la pseudo-clase :focus-within para mostrar el submenú cuando el elemento padre recibe el foco.

```
/* Oculta el submenú inicialmente */
nav ul ul {
  display: none;
}

/* Muestra el submenú cuando el elemento padre está en foco */
nav ul li:focus-within > ul {
  display: block;
}

/* Estilos adicionales para mejorar la usabilidad */
nav ul li a {
  display: block;
  padding: 10px;
  text-decoration: none;
  color: #000;
}
```

```
nav ul li a:hover,
nav ul li a:focus {
  background-color: #f0f0f0;
}
```

- display: none;: Oculta el submenú inicialmente.
- :focus-within: Muestra el submenú cuando el elemento padre recibe el foco (por ejemplo, al navegar con el teclado).

---

3. NAVEGACIÓN CON EL TECLADO

Para que el submenú se abra con el teclado, el usuario debe enfocar el enlace del menú principal (por ejemplo, usando la tecla Tab). Una vez que el enlace está enfocado, el submenú se mostrará automáticamente gracias a :focus-within.

- **Tecla Tab**: Navega entre los elementos del menú.
- **Tecla Enter o Espacio**: Activa el enlace del menú principal (aunque no es necesario para abrir el submenú en este caso).

---

4. LIMITACIONES

- **Accesibilidad**: Sin JavaScript, no es posible cambiar dinámicamente el atributo aria-expanded para indicar si el submenú está abierto o cerrado. Esto puede afectar la experiencia de los usuarios de lectores de pantalla.
- **Compatibilidad**: La pseudo-clase :focus-within no es compatible con versiones muy antiguas de navegadores (como Internet Explorer).

---

5. ALTERNATIVA CON CHECKBOX (HACK CSS)

Otra técnica sin JavaScript es utilizar un checkbox oculto para controlar la visibilidad del submenú. Este enfoque es más interactivo, pero menos semántico.

```
<nav aria-label="Menú principal">
  <ul>
    <li>
      <label for="submenu-toggle">Menú 1</label>
      <input type="checkbox" id="submenu-toggle" hidden>
      <ul>
        <li><a href="#">Submenú 1.1</a></li>
        <li><a href="#">Submenú 1.2</a></li>
      </ul>
    </li>
    <li><a href="#">Menú 2</a></li>
  </ul>
</nav>
Run HTML
css
Copy
/* Oculta el submenú inicialmente */
```

```
nav ul ul {
  display: none;
}

/* Muestra el submenú cuando el checkbox está marcado */
#submenu-toggle:checked ~ ul {
  display: block;
}

/* Estilos adicionales */
label {
  cursor: pointer;
  display: block;
  padding: 10px;
}
```

- **Ventaja**: Funciona sin JavaScript.
- **Desventaja**: No es semántico y puede ser confuso para los lectores de pantalla.

CONCLUSIÓN

Si no puedes usar JavaScript, la mejor opción es utilizar :focus-within en CSS para mostrar el submenú. Sin embargo, ten en cuenta que este enfoque tiene limitaciones en términos de accesibilidad y compatibilidad. Para una experiencia más robusta y accesible, se recomienda utilizar JavaScript junto con HTML y CSS.