26 CSS: Propiedad `visibility`

La propiedad `visibility` en CSS es otra forma de controlar si un elemento es visible o no en la página. Aunque a primera vista pueda parecer similar a `display: none;`, tiene una diferencia fundamental en cómo afecta al diseño (layout) de la página.

Propósito:

La propiedad `visibility` determina si la caja de un elemento es renderizada (visible) o no.

Valores Principales:

1. `visible`:
   - Este es el valor **predeterminado**.
   - El elemento se muestra normalmente en la página.
   - **Ejemplo:**

```
.elemento-visible {
  visibility: visible; /* (Generalmente no necesario, es el default) */
  background-color: lightblue;
  padding: 10px;
}
```

2. `hidden`:
   - El elemento se vuelve **invisible**, pero **sigue ocupando su espacio** en el layout. Es como si el elemento fuera transparente y no interactuable, pero su tamaño y posición todavía afectan a los elementos circundantes.
   - Los elementos descendientes también se ocultan por herencia, pero pueden hacerse visibles explícitamente si se les aplica `visibility: visible;`.
   - **Ejemplo:**
     - **HTML:**

```
<div class="contenedor">
  <div class="caja caja1">Caja 1</div>
  <div class="caja caja2-oculta">Caja 2 (Oculta con visibility)</div>
  <div class="caja caja3">Caja 3</div>
</div>
```

     - **CSS:**

```
.caja {
  width: 100px;
  height: 100px;
  border: 1px solid black;
  margin: 5px;
  display: inline-block; /* Para ponerlas en línea */
```

```
}
.caja1 { background-color: lightcoral; }
.caja3 { background-color: lightseagreen; }

.caja2-oculta {
  visibility: hidden;
  background-color: lightgoldenrodyellow; /* El fondo no se verá */
}
```

- **Resultado:** Verás la Caja 1 y la Caja 3. Entre ellas habrá un espacio vacío del mismo tamaño que tendría la Caja 2 (100×100 píxeles más márgenes). La Caja 3 no se moverá para ocupar el espacio de la Caja 2. La Caja 2 está presente en el layout pero no es visible.

Diferencia Clave: `visibility: hidden` vs. `display: none`

Esta es una distinción crucial:

- `visibility: hidden`:
  - Oculta el elemento visualmente.
  - **Conserva el espacio** que el elemento ocuparía en el layout.
  - Afecta al renderizado visual, pero no al flujo del documento.
  - Generalmente, los elementos ocultos con `visibility: hidden` no responden a eventos del usuario (como clics), aunque pueden participar en transiciones o animaciones.
- `display: none`:
  - Oculta el elemento visualmente.
  - **Elimina completamente el elemento del flujo** del documento. No ocupa ningún espacio.
  - Es como si el elemento no existiera en el HTML para propósitos de layout.
  - Los elementos con `display: none` no responden a eventos y no participan en transiciones/animaciones de la misma manera (a menudo se usan para mostrar/ocultar sin animación o como estado final).

Ejemplo Comparativo:

- **HTML:**

```
<p>Texto antes.</p>
<div style="height: 50px; border: 1px dashed red; visibility: hidden;">
  Div oculto con visibility.
</div>
<p>Texto después (con visibility).</p>

<hr>

<p>Texto antes.</p>
<div style="height: 50px; border: 1px dashed blue; display: none;">
  Div oculto con display.
```

```
</div>
<p>Texto después (con display).</p>
```

- **Resultado:** En el primer caso (`visibility: hidden`), verás "Texto antes", luego un espacio vacío de 50px de alto (con el borde rojo discontinuo invisible pero ocupando espacio), y luego "Texto después (con visibility)". En el segundo caso (`display: none`), verás "Texto antes" inmediatamente seguido por "Texto después (con display)", sin ningún espacio entre ellos ocupado por el div azul.

El Valor `collapse`:

- Existe un tercer valor, `collapse`.
- Su comportamiento principal está definido para elementos de **tablas HTML** (`<table>`, `<tr>`, `<td>`, `<colgroup>`, `<col>`). En filas o columnas de tabla, `collapse` las oculta y **elimina su espacio**, similar a `display: none` pero optimizado para tablas (permite que el layout de la tabla se recalcule eficientemente).
- Para otros elementos (como `div`, `span`, etc.), `collapse` generalmente se comporta **igual que** `hidden` (oculta pero conserva el espacio). Sin embargo, el soporte puede variar ligeramente entre navegadores para elementos no tabulares, por lo que `hidden` es más predecible fuera del contexto de tablas.

Herencia:

La propiedad `visibility` se hereda. Si un elemento padre tiene `visibility: hidden`, sus hijos también serán invisibles. Sin embargo, un hijo puede anular esto estableciendo explícitamente `visibility: visible;`.

- **Ejemplo:**

```
<div style="visibility: hidden; border: 1px solid red; padding: 10px;">
 Texto padre oculto.
 <p style="visibility: visible; color: green;">
  Este párrafo hijo SÍ es visible.
 </p>
 Otro texto padre oculto.
</div>
```

- **Resultado:** Verás solo el texto "Este párrafo hijo SÍ es visible." dentro de un espacio vacío que corresponde al `div` padre.

Cuándo Usar `visibility: hidden`:

- Cuando necesitas ocultar un elemento pero **mantener intacto el layout** de la página, evitando que otros elementos se muevan para llenar el espacio vacío.
- En algunas técnicas de animación o transición donde el elemento debe ocupar espacio incluso cuando no es visible.
- Para hacer visible un elemento hijo dentro de un padre invisible.