EcomLabs

GUÍA PASO A PASO

CLS: el banner de cookies que no mueve la página

Cómo medir el CLS que provoca tu banner de cookies en Chrome DevTools y con un script de consola, el CSS y el JavaScript de un banner fijo abajo que no empuja nada ni tapa el foco del teclado, y la checklist para comprobarlo antes y después.

PARA	Cualquier tienda en línea con un banner de cookies o de consentimiento
PROBADO CON	Playwright 1.62.0 · Chromium 151 · axe-core 4.12.1 · Node 22.22.3 · Windows 11
ACTUALIZADA	11 de noviembre de 2026
DOCUMENTACIÓN	web.dev/articles/cls
ARTÍCULO	ecomlabs.dev/g/cls
AUTOR	Luis Morales · @luismoralesdev

Índice

Antes de empezar	Qué vas a lograr
Paso 01	Encuentra el salto en Chrome DevTools
Paso 02	Mide con un script de consola
Paso 03	El CSS del banner que no empuja
Paso 04	El JavaScript que lo muestra
Paso 05	Si prefieres reservar espacio
Paso 06	Lo que medimos
Al terminar	Checklist

ANTES DE EMPEZAR

Qué vas a lograr

Medir cuánto mueve tu página el banner de cookies, pasarlo a un banner fijo abajo que aparece encima del contenido sin empujar nada, y comprobar el cambio. Todo el código está copiado de las páginas de prueba que medimos en local el 11 de noviembre de 2026 con Playwright 1.62.0 y Chromium 151. `saveConsent` era un stub: no se llamó a ninguna plataforma de consentimiento.

Umbrales de CLS, en el percentil 75 y por separado en celular y en computadora: 0.1 o menos es bueno; más de 0.1 y hasta 0.25 necesita mejorar; más de 0.25 es malo.

PASO 01

Encuentra el salto en Chrome DevTools

Abre una página de producto en una ventana de incógnito, para que no haya consentimiento guardado y el banner salga. Emula un celular: el mismo banner nos dio 0.1611 a 390 × 844 y 0.0332 a 1366 × 768.

- **Performance:** al abrir el panel ves tu LCP y tu CLS locales y, en la pestaña **Layout shifts**, una tabla de saltos.
- **Record and reload:** activa **Screenshots** y graba. DevTools pasa por `about:blank` y graba la carga completa.
- **Track Layout shifts:** cada rombo morado es un salto. Haz clic en él: **Summary** muestra tiempos, puntaje, elementos y posibles culpables.
- **Insights › Layout shift culprits:** la mejor hipótesis de la causa (imágenes sin tamaño, iframes inyectados, animaciones, fuentes).
- **Rendering › Layout Shift Regions:** `Ctrl+Shift+P`, escribe `rendering`, elige **Show Rendering** y marca la casilla. Al recargar, lo que se mueve se pinta de morado.

PASO 02

Mide con un script de consola

Pégalo en la consola cuando el banner ya apareció; `buffered: true` recupera los saltos anteriores:

```
let cls = 0, windowValue = 0, windowStart = 0, lastShift = -Infinity;

new PerformanceObserver((list) => {
  for (const entry of list.getEntries()) {
    if (entry.hadRecentInput) continue;
    const t = entry.startTime;
    if (t - lastShift >= 1000 || t - windowStart >= 5000) {
      windowValue = 0;
      windowStart = t;
    }
    windowValue += entry.value;
    lastShift = t;
    cls = Math.max(cls, windowValue);
    console.log("shift", entry.value.toFixed(4), "CLS", cls.toFixed(4),
```

```
      entry.sources.map((s) => s.node));
  }
}).observe({ type: "layout-shift", buffered: true });
```

Descarta los saltos de los 500 ms siguientes a un toque, clic o tecla (`hadRecentInput`), aplica las ventanas de sesión y se queda con la peor: ese es el CLS. `sources` lista los elementos que se movieron, no la causa: con el banner arriba salen el header y el main. MDN marca la API LayoutShift como experimental y fuera de Baseline, así que mide en Chrome.

PASO 03

El CSS del banner que no empuja

Fijo abajo y fuera del flujo, con el espacio de la página ajustado a su altura:

```
.cookie-banner {
  position: fixed;
  inset: auto 0 0 0;
  z-index: 100;
  display: flex;
  flex-wrap: wrap;
  align-items: center;
  gap: 0.75rem 1.5rem;
  max-height: 50vh;
  overflow-y: auto;
  padding: 1rem 1rem calc(1rem + env(safe-area-inset-bottom));
  background: #14171a;
  color: #f4f5f6;
}
.cookie-banner p { flex: 1 1 32ch; margin: 0; }
.cookie-banner a { color: inherit; }
.cookie-banner .actions { display: flex; gap: 0.5rem; }
.cookie-banner button {
  min-height: 44px;
  padding: 0 1.25rem;
  font: inherit;
  color: inherit;
  background: transparent;
  border: 1px solid currentColor;
  border-radius: 4px;
}
.cookie-banner :focus-visible { outline: 3px solid #ffd166; outline-offset: 2px; }
html { scroll-padding-bottom: var(--cookie-banner-h, 0px); }
body { padding-bottom: var(--cookie-banner-h, 0px); }
@media (prefers-reduced-motion: no-preference) {
  .cookie-banner { animation: cookie-banner-in 200ms ease-out; }
}
@keyframes cookie-banner-in { from { transform: translateY(100%); } }
```

- **Sin saltos:** `position: fixed` lo pone encima del contenido y la entrada usa `transform`, que no cuenta como layout shift.
- **Foco a la vista:** `scroll-padding-bottom` evita que el elemento con foco quede debajo (WCAG 2.2, criterio 2.4.11, técnica C43); `padding-bottom` deja leer el final de la página.
- **Contraste:** texto 16.48:1 y contorno de foco 12.48:1 sobre el fondo del banner; botones de 44 px.
- **Celular:** `max-height: 50vh` con scroll interno; nunca tapa más de media pantalla.

PASO 04

El JavaScript que lo muestra

Llama a `showCookieBanner(banner)` desde tu script de consentimiento con el elemento del banner ya creado:

```
function showCookieBanner(banner) {
  const root = document.documentElement;
  const observer = new ResizeObserver(() => {
    root.style.setProperty("--cookie-banner-h", banner.offsetHeight + "px");
  });
  document.body.prepend(banner);
  observer.observe(banner);
  banner.addEventListener("click", (event) => {
    const button = event.target.closest("button[data-consent]");
    if (!button) return;
    saveConsent(button.dataset.consent);
    observer.disconnect();
    banner.remove();
    root.style.removeProperty("--cookie-banner-h");
  });
}
```

El HTML del banner que usamos en la prueba:

```
<section class="cookie-banner" aria-label="Cookies">
  <p>We use cookies to measure visits and remember your cart.
    <a href="/privacy">Privacy policy</a></p>
  <div class="actions">
    <button type="button" data-consent="reject">Reject</button>
    <button type="button" data-consent="accept">Accept</button>
  </div>
</section>
```

`prepend` lo deja primero en el orden del tabulador sin mover nada, `ResizeObserver` mantiene la altura al día si el celular gira y `saveConsent` es tuya: tu plataforma de consentimiento o tu propia cookie.

> **CUIDADO**
>
> Si tu tema tiene una barra fija de «Agregar al carrito», el banner la tapa. Súbela con `translate: 0 calc(-1 * var(--cookie-banner-h, 0px))`; cambiar su `bottom` registró un salto de 0.0187.

PASO 05

Si prefieres reservar espacio

Solo sirve si el servidor sabe si hay consentimiento y la altura del banner es estable: el servidor imprime un `<div class="cookie-slot">` con `min-height` por breakpoint solo cuando la petición no trae la cookie, y el script mete el banner dentro. Mide tu banner en cada ancho: el nuestro mide 136 px hasta 480 px y 76 px desde 767; un solo hueco de 80 px dio 0.0601 en celular, y un hueco que el script quita después, 0.1611.

PASO 06

Lo que medimos

Medido en local en una página de prueba: cada variante servida en memoria por Playwright, Chromium 151, tres corridas iguales por variante.

VARIANTE	CELULAR	COMPUTADORA
Banner arriba, en el flujo	0.1611	0.0332
Banner fijo abajo (este CSS)	0	0
Banner fijo animando `bottom`	0.0143	0.0031
Imagen sin `width`/`height`	0.1835	0.0862
Imagen con `width`/`height`	0	0

Son datos de laboratorio. El dato que cuenta para Core Web Vitals es el de campo: PageSpeed Insights muestra un promedio de 28 días de usuarios reales, así que tu arreglo tarda semanas en verse completo.

AL TERMINAR

Checklist

- [ ] Mediste el CLS en incógnito, con celular emulado y en computadora, y anotaste el valor.
- [ ] El banner está fuera del flujo: `position: fixed` abajo, o en un hueco que imprime el servidor.
- [ ] La entrada usa `transform` y no corre con `prefers-reduced-motion: reduce`.
- [ ] `scroll-padding-bottom` y `padding-bottom` usan la altura real del banner, y con Tab ningún enlace con foco queda debajo.
- [ ] No tapa más de media pantalla ni la barra fija de compra en celular.
- [ ] Texto con contraste de 4.5:1 o más, foco visible, aceptar y rechazar del mismo tamaño.
- [ ] Todas las imágenes llevan `width` y `height`; anuncios e iframes, `min-height`.
- [ ] Volviste a medir, la llegada del banner ya no registra saltos y revisas el dato de campo en las semanas siguientes.