EcomLabs

GUÍA PASO A PASO

Búsquedas sin resultados en tu tienda, medidas en GA4

El custom pixel de Shopify que envía a GA4 las búsquedas sin productos, la dimensión y la exploración para el reporte semanal, un script en Node que junta y ordena los términos, y la plantilla para decidir qué corregir cada lunes.

PARA	Tiendas Shopify que miden con GA4 y quieren saber qué buscan sus clientes sin encontrarlo
PROBADO CON	Node 22.22.3 · el pixel en node:vm con un objeto analytics falso · CSV ilustrativos con el formato de exportación de GA4
ACTUALIZADA	19 de noviembre de 2026
DOCUMENTACIÓN	shopify.dev/docs/api/web-pixels-api/standard-events/search_submitted
ARTÍCULO	ecomlabs.dev/g/busqueda
AUTOR	Luis Morales · @luismoralesdev

Índice

Paso 01	El evento: un custom pixel
Paso 02	El reporte en GA4
Paso 03	El script de los diez más comunes
Cada lunes	Plantilla semanal

PASO 01

El evento: un custom pixel

`view_search_results` no dice cuántos resultados hubo; `search_submitted` de Shopify sí. En Shopify: *Settings > Customer events > Add custom pixel*. Pega el código, cambia `G-XXXXXXXXXX` por tu ID de medición, guarda y pulsa *Connect*.

```
// Shopify admin > Settings > Customer events > Add custom pixel
const GA_ID = "G-XXXXXXXXXX"; // your GA4 measurement ID

const script = document.createElement("script");
script.setAttribute("src", `https://www.googletagmanager.com/gtag/js?id=${GA_ID}`);
script.setAttribute("async", "");
document.head.appendChild(script);

window.dataLayer = window.dataLayer || [];
function gtag() { dataLayer.push(arguments); }
gtag("js", new Date());
gtag("config", GA_ID, { send_page_view: false });

analytics.subscribe("search_submitted", (event) => {
  const result = event.data?.searchResult;
  if (!result || result.productVariants.length > 0) return; // only searches with zero products
  let term = (result.query || "").trim().toLowerCase().replace(/\s+/g, " ");
  if (!term) return;
  if (term.includes("@") || term.replace(/\D/g, "").length >= 7) term = "(redacted)"; // no emails or phone
numbers
  const { origin, pathname } = event.context.window.location;
  gtag("event", "search_no_results", {
    zero_result_term: term.slice(0, 100),
    page_location: origin + pathname, // the store URL without ?q=, which holds the raw term
  });
});
```

- Solo envía con cero productos. El término va limpio, cortado a 100 caracteres (el máximo en GA4), y como `(redacted)` si lleva `@` o siete dígitos: Google pide no enviar datos personales.
- `page_location` va sin `?q=`, y `send_page_view: false` evita un segundo `page_view`. El pixel solo corre con los permisos de su configuración (Marketing y Analytics por defecto): que coincidan con tu banner.

Pruébalo con el Pixel Helper y DebugView: lo que existe no envía nada; `zzzz prueba` envía un evento.

PASO 02

El reporte en GA4

La dimensión. *Admin > Data display > Custom definitions > Create custom dimensions* (necesitas rol de Editor):

- *Dimension name*: `Zero result term` (sin guiones: GA4 no los acepta).

- *Scope*: *Event*. *Event parameter*: `zero_result_term`. Ninguno se puede cambiar después.
- Tarda de 24 a 48 horas en aparecer. No registres `search_term`: ya es predefinida.

La exploración. Se arma una vez y se abre cada lunes:

1. *Explore*, exploración en blanco (tabla de forma libre).
2. En *Variables*, importa *Zero result term*, *Event name* y la métrica *Event count*.
3. *Zero result term* en filas, *Event count* en valores.
4. Filtro: *Event name exactly matches* `search_no_results`.
5. Rango: la semana cerrada, de lunes a domingo; el procesamiento tarda hasta 48 horas.
6. *Export data > CSV*: trae todas las filas, no solo las visibles.

PASO 03

El script de los diez más comunes

GA4 agrupa por texto exacto (`café` y `cafe` son dos filas). `top-terms.mjs` las junta y muestra las diez primeras; probado con Node 22:

```
// Usage: node top-terms.mjs ga4-export.csv [catalog.txt]
// ga4-export.csv: the exploration exported as CSV (term column, then Event count).
// catalog.txt (optional): words from your product titles, one per line.
import { readFileSync } from "node:fs";

const [csvFile, catalogFile] = process.argv.slice(2);

// One CSV line into cells; handles "quoted, cells" and "" inside quotes.
function cells(line) {
  const out = [];
  let cell = "", quoted = false;
  for (let i = 0; i < line.length; i++) {
    const c = line[i];
    if (quoted && c === '"' && line[i + 1] === '"') { cell += '"'; i++; }
    else if (c === '"') quoted = !quoted;
    else if (c === "," && !quoted) { out.push(cell); cell = ""; }
    else cell += c;
  }
  return [...out, cell];
}

// "Tenis  Blancos" and "tenis blancos" count as the same search.
const norm = (s) => s.normalize("NFD").replace(/\p{M}/gu, "").toLowerCase().replace(/\s+/g, " ").trim();

// Letters you have to change, add or remove to turn a into b.
function distance(a, b) {
  let prev = Array.from({ length: b.length + 1 }, (_, j) => j);
  for (let i = 1; i <= a.length; i++) {
    const row = [i];
    for (let j = 1; j <= b.length; j++) {
      row[j] = Math.min(prev[j] + 1, row[j - 1] + 1, prev[j - 1] + (a[i - 1] === b[j - 1] ? 0 : 1));
```

```
    }
    prev = row;
  }
  return prev[b.length];
}

// Skip the # lines GA4 puts before the header, the header itself and any totals row.
const lines = readFileSync(csvFile, "utf8").replace(/^/, "").split(/\r?\n/);
const rows = lines.filter((l) => l.trim() && !l.startsWith("#")).slice(1).map(cells);
const totals = new Map();
for (const [term, count] of rows) {
  const n = Number(String(count).replace(/,/g, ""));
  const key = norm(term ?? "");
  if (!key || !Number.isFinite(n) || /^(grand )?totals?$/.test(key)) continue;
  totals.set(key, (totals.get(key) ?? 0) + n);
}

const catalog = catalogFile ? readFileSync(catalogFile, "utf8").split(/\r?\n/).map(norm).filter(Boolean) : [];
function hint(term) {
  for (const word of term.split(/[^\p{L}\p{N}]+/u)) {
    if (word.length < 5 || catalog.includes(word)) continue; // short words give false hints
    const near = catalog.find((w) => distance(word, w) <= 2);
    if (near) return `typo? ${word} -> ${near}`;
  }
  return "";
}

const top = [...totals].sort((a, b) => b[1] - a[1] || a[0].localeCompare(b[0])).slice(0, 10);
console.log("  #  searches  term".padEnd(39) + "hint");
top.forEach(([term, n], i) => {
  console.log(`${String(i + 1).padStart(3)}  ${String(n).padStart(8)}  ${term.padEnd(24)}${hint(term)}`);
});
```

Corre `node top-terms.mjs ga4-export.csv catalog.txt`. `catalog.txt`, opcional, lleva una palabra de tus títulos y etiquetas por línea; con él, el script marca posibles errores de dedo.

CADA LUNES

Plantilla semanal

SEMANA	TÉRMINO	BÚSQUEDAS	CAUSA	ARREGLO	¿SALIÓ?
9–15 nov (ilustrativo)	volza de piel	22	Error de dedo	Etiqueta «volza» en bolsas	24 nov

1. Exporta la semana cerrada, corre el script, pega los diez términos y ponle a cada uno causa y arreglo: sinónimo o error de dedo (la palabra como etiqueta), campo que no se busca (pásalo a título, tipo o etiquetas), producto que falta (anótalo) o ruido.
2. Arregla primero lo más buscado que menos cuesta. El lunes siguiente, marca qué términos corregidos ya no aparecen: si uno sigue, el arreglo no funcionó.