EcomLabs

STEP-BY-STEP GUIDE

GA4 ecommerce events

Working code for view_item, add_to_cart, begin_checkout and purchase, with gtag and with the dataLayer, ready to adapt to your store. Then the order to validate it in the console, Tag Assistant and DebugView before you trust a report.

FOR	Stores running gtag.js or Google Tag Manager
TESTED WITH	Chromium 151 (Playwright) · local dataLayer · Google traffic blocked
UPDATED	October 30, 2026
DOCS	developers.google.com/analytics/devguides/collection/ga4/ecommerce
ARTICLE	ecomlabs.dev/g/ga4-en
AUTHOR	Luis Morales · @luismoralesdev

Contents

Before you start	What you will set up
Step 01	view_item: a product is viewed
Step 02	add_to_cart: it goes in the cart
Step 03	begin_checkout: payment starts
Step 04	purchase: the order is placed
Step 05	Set up Tag Manager
Step 06	Validate in this order
When you finish	Checklist

BEFORE YOU START

What you will set up

Four events cover the main path of a purchase: `view_item` (a product is viewed), `add_to_cart` (it is added), `begin_checkout` (payment starts) and `purchase` (the order is placed). With them, GA4 can show you which step people leave at.

Use one way of sending per site. If you load `gtag.js` directly, call `gtag("event", …)`. If you run Google Tag Manager, push to the `dataLayer` and let a GA4 tag send it. This guide uses the dataLayer; products and prices are illustrative: change the values, never the field names.

STEP 01

view_item: a product is viewed

On the product page, with the data of the product on screen:

```
window.dataLayer = window.dataLayer || [];
window.dataLayer.push({ ecommerce: null });
window.dataLayer.push({
  event: "view_item",
  ecommerce: {
    currency: "USD",
    value: 24,
    items: [{ item_id: "MUG-001", item_name: "Ceramic mug", price: 24, quantity: 1 }],
  },
});
```

The first line creates the data layer without wiping what is there (never reassign it with `= []`), and `{ ecommerce: null }` clears the previous event so it does not mix with this one.

STEP 02

add_to_cart: it goes in the cart

With the dataLayer, once the cart confirms the product went in:

```
window.dataLayer = window.dataLayer || [];
window.dataLayer.push({ ecommerce: null });
window.dataLayer.push({
  event: "add_to_cart",
  ecommerce: {
    currency: "USD",
    value: 48,
    items: [{ item_id: "MUG-001", item_name: "Ceramic mug", price: 24, quantity: 2 }],
  },
});
```

With gtag.js, the same `currency`, `value` and `items` go straight into `gtag("event", "add_to_cart", { … })`, with no `ecommerce` object and no clearing line. The full example is in the article.

> **CAUTION**
> Fire add_to_cart when your cart confirms the product went in (your API's or theme's response), not on the button click. If the product is not added because of stock or a network error, it must not count.

STEP 03

begin_checkout: payment starts

With the full cart. `value` is the sum of every line: 24 × 2 + 13 × 1 = 61.

```
window.dataLayer = window.dataLayer || [];
window.dataLayer.push({ ecommerce: null });
window.dataLayer.push({
  event: "begin_checkout",
  ecommerce: {
    currency: "USD",
    value: 61,
    items: [
      { item_id: "MUG-001", item_name: "Ceramic mug", price: 24, quantity: 2 },
      { item_id: "BOWL-002", item_name: "Deep bowl", price: 13, quantity: 1 },
    ],
  },
});
```

STEP 04

purchase: the order is placed

On the confirmation page, once per order:

```
window.dataLayer = window.dataLayer || [];
window.dataLayer.push({ ecommerce: null });
window.dataLayer.push({
  event: "purchase",
  ecommerce: {
    transaction_id: "T-1001",
    currency: "USD",
    value: 61,
    shipping: 5,
    items: [
      { item_id: "MUG-001", item_name: "Ceramic mug", price: 24, quantity: 2 },
      { item_id: "BOWL-002", item_name: "Deep bowl", price: 13, quantity: 1 },
    ],
  },
});
```

- `transaction_id` is required: use your store's real order number.
- `shipping` goes on its own. The reference asks for `value` without shipping or tax.

> **CAUTION**
>
> If your prices already include sales tax or VAT, pick one rule (with or without tax) and use it the same way in `price` and `value` across all four events. If each event follows a different rule, GA4's totals will not match each other.

STEP 05

Set up Tag Manager

With the dataLayer, every event needs a tag and a trigger in Tag Manager:

WHERE	WHAT TO SET
Tag type	Google Analytics: GA4 Event
Event name	`add_to_cart` (or the matching event)
More Settings › Ecommerce	Tick *Send Ecommerce data* and pick *Data Layer* as the source
Trigger	*Custom Event* with the same name: `add_to_cart`

Repeat for `view_item`, `begin_checkout` and `purchase`. The trigger name and the pushed name must match character for character: GA4 and Tag Manager are case sensitive.

STEP 06

Validate in this order

1. In the console

Add a product to the cart and paste this into the browser console. It finds both dataLayer pushes and gtag calls:

```
dataLayer.filter((e) => e.event === "add_to_cart" || e[1] === "add_to_cart");
```

An empty array means the problem is on your site. Do not move on to GA4 until the event shows up here.

2. In Tag Assistant

Open your site through `tagassistant.google.com` or Tag Manager's preview mode. Confirm the GA4 tag fires on the event and carries `currency`, `value` and `items`.

3. In DebugView

Tag Assistant and preview mode already turn on debug mode for your device. With gtag you can also turn it on like this:

```
gtag("config", "G-XXXXXXXXXX", { debug_mode: true });
```

In GA4, go to *Admin › Data display › DebugView*, click the event and check its parameters.

> **CAUTION**
> To turn debug mode off, remove `debug_mode`. Setting it to `false` does not disable it. And if your site uses consent mode, accept the Analytics cookies while testing or the events will not show in DebugView.

4. In the reports

Realtime takes a few minutes. Standard reports can take 24 to 48 hours, so do not judge the change on the day you ship it.

WHEN YOU FINISH

Checklist

- [ ] All four names are lowercase and copied from the reference: `view_item`, `add_to_cart`, `begin_checkout`, `purchase`.
- [ ] Every ecommerce push is preceded by `{ ecommerce: null }`.
- [ ] `currency` is a three-letter ISO code and is present whenever `value` is.
- [ ] `value` is a number equal to the sum of `price × quantity`, without shipping or tax.
- [ ] Every item has `item_id` or `item_name` (ideally both).
- [ ] `purchase` carries the real `transaction_id` and is sent once per order.
- [ ] add_to_cart fires when the cart confirms, not on the click.
- [ ] The console finds the event and Tag Assistant shows the tag firing.
- [ ] DebugView shows all four events with their parameters.
- [ ] No other integration (an app or a sales channel) sends the same events.
- [ ] You removed `debug_mode` before going live.