ClickGo Tracking SDK

A setup guide for campaign & landing-page tracking

This guide explains what the ClickGo Tracking SDK is, why you'd use it, and exactly how to add it to an advertiser's pages to track clicks, conversions, and events — including events fired when a visitor clicks something like an **Add to Cart** button.

1. What the Tracking SDK is

The Tracking SDK is a tiny script you (or the advertiser) add to the campaign's landing page — and/or the order-confirmation / "thank you" page. Once it's on the page, it records the click and any conversions **directly from the visitor's browser**, reliably, even when the visitor lands on the advertiser's site without going through a ClickGo redirect.

It stores a small first-party value called the **ClickID** in a cookie on the advertiser's own website. That ClickID is what ties "this visitor" to "this publisher's click," so conversions get credited to the right campaign and publisher.

2. Why use it

- **Redirectless links** — the publisher's link can go **straight to the advertiser's page** instead of bouncing through a tracking redirect. Faster for the visitor, and no redirect step to lose.
- **More reliable tracking** — the ClickID is a **first-party** cookie on the advertiser's own domain, so it holds up far better against Safari/Firefox tracking prevention than older third-party pixels.
- **Browser-based conversions** — you can fire a conversion from the page itself, without the advertiser having to build a server-to-server postback.
- **Automatic capture** — it picks up the referrer and the publisher's sub IDs (s1–s7) for you.
- **Publisher pixels fire automatically** — if the campaign uses a conversion container, the SDK fires the publisher's pixels on conversion with no extra work.
- **Works alongside standard links** — a campaign can offer both the redirectless link and the normal redirect link; the SDK handles either.

3. The easy way: the in-app SDK Builder

Open the campaign, go to **Conversion Tracking Options → Enhanced Tracking**, and the **SDK Builder** generates the exact snippets for your tracking domain — the landing snippet, the conversion snippet, and event snippets — ready to copy. The rest of this guide explains what those snippets do so you can place them correctly and adjust them with confidence.

4. Step 1 — Install the SDK on the landing page

Add these two lines to the campaign's **landing page** (the page the visitor first arrives on). Put them just before the closing `</body>` tag, or in the `<head>`. Replace **YOUR-TRACKING-DOMAIN** with your tracking domain.

```
<script src="https://YOUR-TRACKING-DOMAIN/sdk.js"></script>
<script>
  ClickGo.track();
</script>
```

That's it. `ClickGo.track()` is smart about what to do:

- If the visitor already has a ClickID (they came through a standard redirect link, or visited before), it **keeps** it — no duplicate click.
- If they arrived on a **redirectless** link, it **creates** the click right there and stores the ClickID — the publisher and creative are read from the link.
- If it can't find anything to track, it quietly does nothing — it never breaks the page.

Important: keep the two `<script>` tags separate and **don't add `async`** to the first one. The SDK file needs to load before the line that calls `ClickGo.track()` runs.

5. Step 2 — Track the conversion

On the **order-confirmation / thank-you page** (the page shown after the visitor buys or signs up), add this. The SDK automatically reads the ClickID it stored earlier — you don't pass it in.

```
<script src="https://YOUR-TRACKING-DOMAIN/sdk.js"></script>
<script>
  ClickGo.trackConversion({
    orderId: "ORDER-12345",
    orderTotal: "99.00",
    includePublisherPixels: true
  });
</script>
```

What the fields mean (all optional, but recommended):

- **orderId** — the advertiser's order/transaction number, so you can reconcile later.
- **orderTotal** — the sale amount (used for reporting and revenue-share payouts).
- **cpa** — only if you pay a fixed amount per conversion and want to send it explicitly.
- **s1-s7** — Advertiser level sub IDs (s1-s7) back for your own reconciliation (maybe storing order emails, plan details, billing methods, etc here.) These are not the publishers subIDs sent on click.

If the campaign uses a conversion container, the publisher's pixels fire automatically when this runs with the `includePublisherPixels: true` — nothing extra to add.

6. Event tracking

“Events” are extra conversions beyond the first one — a presell, upsell, a second step, or an action the visitor takes on the page. You fire them with `ClickGo.trackEvent()`, which works just like a conversion but requires an **Event ID**.

Where the Event ID comes from: set the event up on the campaign first (the campaign's Events section), then use the ID shown there — the SDK Builder's event picker lists them for you. In the examples below it's shown as `eventId: 12`; replace `12` with your real event's ID.

a) An event fired on a page (upsell, second step, etc.)

Place this on the page where the event happens — same idea as a conversion, just with `trackEvent` and the event ID:

```
<script src="https://YOUR-TRACKING-DOMAIN/sdk.js"></script>
<script>
  ClickGo.trackEvent({
    eventId: 12,
    orderId: "UPSELL-987",
    orderTotal: "29.00"
  });
</script>
```

b) An event fired when the visitor clicks something (e.g. Add to Cart)

You can fire an event the moment a visitor does something — clicking **Add to Cart**, starting checkout, submitting a newsletter form, and so on. The SDK uses the ClickID it already stored on the landing page, so as long as `ClickGo.track()` ran when they arrived, the event is credited correctly.

The simplest way — put it right on the button:

```
<button onclick="ClickGo.trackEvent({ eventId: 12 })">
  Add to Cart
</button>
```

Or attach it in a script (handy when you can't edit the button tag directly). This example fires when an element with `id="add-to-cart"` is clicked:

```
<script>
  document.querySelector("#add-to-cart").addEventListener("click", function () {
    ClickGo.trackEvent({ eventId: 12 });
  });
</script>
```

Same pattern works for any interaction — a “Start Free Trial” button, a form's submit, a “Continue” step — just point it at the matching event's ID.

7. Sub IDs (s1–s7)

Sub IDs are the publisher's own tracking slots — s1 through s7. You normally don't touch them:

- **On the landing page**, the SDK **automatically captures** any s1–s7 the publisher added to their link — exactly like a normal redirect click would.
- **On a conversion or event**, you can also send **separate** Advertiser level sub IDs (s1-s7) back for your own reconciliation (maybe storing order emails, plan details, billing methods, etc here.) These are **not** taken from the page automatically (the page's own address isn't the click). If you need to attach one for your own reconciliation, pass it explicitly:

```
ClickGo.trackConversion({
  orderId: "ORDER-12345",
  orderTotal: "99.00",
  s1: "winter-sale"
});
```

8. Advanced configuration

If you need to change defaults, call `ClickGo.configure({ ... })` **once, before** `ClickGo.track()`. All settings are optional.

```
<script src="https://YOUR-TRACKING-DOMAIN/sdk.js"></script>
<script>
  ClickGo.configure({
    cookieDomain: ".advertiser.com",
    cookieDuration: 60,
    debug: true
  });
  ClickGo.track();
</script>
```

- **cookieDomain** — set this to a domain like `.advertiser.com` when the landing page and the thank-you page are on **different subdomains** (e.g. `www.advertiser.com` → `shop.advertiser.com`). It lets the ClickID carry across them. Leave it out if everything is on one host.
- **cookieDuration** — how many days the ClickID is remembered (your attribution window). Defaults to **90** days.
- **organic** — a “house default” publisher/creative for **untagged** visits (people who reach the page without a tracked link). See the organic example below.

- **debug** — set to `true` while you're testing to print helpful messages to the browser console. Turn it off for live pages.
- **apiBase** — only needed if you self-host the SDK on a different origin. Almost never required; leave it out and the SDK uses the domain it was loaded from.

Organic / house-traffic example

To attribute untagged visitors to a house publisher and creative:

```
<script src="https://YOUR-TRACKING-DOMAIN/sdk.js"></script>
<script>
  ClickGo.configure({
    organic: { creativeId: 34, publisherId: 100 }
  });
  ClickGo.track();
</script>
```

9. Helper functions

- **ClickGo.getClickId()** — returns the stored ClickID. Useful if you want to pass it into the advertiser's own systems or a server-to-server feed.

```
  var clickId = ClickGo.getClickId();
```

- **ClickGo.urlParam("name")** — reads a value from the page's web address (query string). Handy for pulling a value the advertiser put in the URL to send along with a conversion.

10. Full examples

Landing page (complete)

```
<!doctype html>
<html>
  <head><title>Special Offer</title></head>
  <body>
    <!-- your landing page content -->

    <script src="https://YOUR-TRACKING-DOMAIN/sdk.js"></script>
    <script>
      ClickGo.track();
    </script>
  </body>
</html>
```

Thank-you page with an Add-to-Cart event (complete)

```
<!doctype html>
<html>
  <head><title>Thank You</title></head>
  <body>
    <h1>Thanks for your order!</h1>
    <button id="add-to-order">Add 1-year warranty to your order - $29</button>

    <script src="https://YOUR-TRACKING-DOMAIN/sdk.js"></script>
    <script>
      // record the purchase
```

```
      ClickGo.trackConversion({
        orderId: "ORDER-12345",
        orderTotal: "99.00"
      });

      // record the upsell when they click the button
     document.querySelector("#add-to-order").addEventListener("click", function () {
        ClickGo.trackEvent({ eventId: 12, orderTotal: "29.00" });
      });
    </script>
  </body>
</html>
```

11. Redirectless vs. standard links (quick primer)

When **Allow Redirectless** is on, a campaign offers two link styles — publishers can use either:

- **Redirectless link** — the advertiser's destination URL itself (with a small `cgid` value the SDK reads). The visitor lands straight on the page; the SDK on that page creates the click. This is the link that **requires** the SDK to be installed.
- **Standard link** — the usual tracking-domain redirect link. It works with or without the SDK; if the SDK is present it simply "captures" the ClickID the redirect already created.

Tip: keep the `{CLICK_ID}` token in your destination URL. It costs nothing on a redirectless link (it's dropped automatically) and keeps the standard link fully working alongside the SDK.

12. Testing & troubleshooting

- **Turn on debug** — add `ClickGo.configure({ debug: true })` and watch the browser's developer console for `[ClickGo]` messages telling you what happened.
- **Check the cookie** — after landing, a cookie named `_cg_click` should appear for the advertiser's site. No cookie usually means no ClickID was created or captured.
- **"Nothing to track"** — if a visitor reaches the page with no ClickID and no link identifiers (and no organic default set), the SDK does nothing by design. Make sure the link is a real tracked link, or configure an organic default.
- **Conversions not crediting** — the conversion page reads the ClickID from the cookie, so the SDK must have run on the landing page first, and (for separate subdomains) `cookieDomain` must be set.
- **Missing referrer** — some browsers set a strict privacy policy that hides the referring page; that's a browser limitation, not a setup error.
- **Redirectless link not tracking at all** — the SDK must be installed on the landing page. Without it, a redirectless link has nothing to create the click.

13. FAQ

Will it slow down the page? No — the script is tiny, runs quietly, and is built to never block the page or throw errors.

What if the SDK isn't installed? Standard redirect links still track normally. Only redirectless links depend on the SDK being on the page.

Is it privacy-friendly? The ClickID is a first-party value stored on the advertiser's own site for attribution — the same purpose a redirect already serves, just more reliably.