Integrate Optimizely with Google Universal Analytics

Prerequisites
1. Enable Google UA integration
2. Create a Google UA custom dimension
3. Create a custom report
4. Confirm your integration
Additional step for gtag.js tracking code

This article will help you:

- Integrate Google Universal Analytics to read data from Optimizely.
- Add Google Universal Analytics integration code to your page.
- Track Google Universal Analytics metrics as experiment goals.
- Create custom dimensions and reports in Google Universal Analytics to track and report on Optimizely data.

If you use Google Universal Analytics (UA) to monitor your site’s traffic, you can view the results of your Optimizely experiments alongside the rest of the data that Google UA collects.

Here are a few reasons to integrate:

- Get a direct view of how Optimizely experiments affect the metrics you track in Google Analytics
- See Google Analytics data for each variation in your experiment (or experience in a campaign)
- Filter your Google Analytics reporting by visitors who were successfully included in an Optimizely experiment and exclude those who weren’t

Looking for a different Google Analytics integration? See our Google Tag Manager article. To learn more, you can also check out these great write-ups by Geoff Kenyon, Trevor Fox, and our Solutions Partner Vertical Nerve.
You can also [download a checklist](#) to help you manage this procedure.

### Prerequisites

First, [enable Google Analytics in your Optimizely project](#) if you haven't already.

Here are a few other details to keep in mind:

- To integrate Optimizely with Google UA, you'll need an available [custom dimension in Google Analytics](#) to populate with Optimizely experiment data.

- Use the number (or *index*) of the custom dimension to configure and send data to your Optimizely experiment. To access your Google UA custom dimensions and create one for Optimizely, scroll down to the [Create a Google UA custom dimension](#) section below.

- Use one custom dimension per concurrently running experiment or campaign to keep experiment names and variations from overwriting each other.

- If your page's Google UA setting has the `setDomainName` function call on your page, make sure that it doesn't use a leading period or "www".

- Your [Optimizely experiment or campaign must be running](#) for Google UA integration to work.

In Optimizely:

- Your Optimizely and Google UA snippets can be placed in any order.

- You do not need to make the `activateUniversalAnalytics` API call.

- Optimizely does not send Optimizely data with the 'pageview'. It sends a separate 'non-interaction event', independent of the 'pageview'.

The reason for having one custom dimension per concurrently running experiment is **data attribution**.

For example, suppose Experiment A and Experiment B both use Custom Dimension 1. If a visitor lands on Experiment A within page 1 and lands on Experiment B on page 2 within the same session, the visitor's actions are attributed only to Experiment B from that point on.

In the same scenario, if Experiment A and Experiment B have their own unique custom dimensions, Experiment A will continue to collect data for the visitor even after they're bucketed into Experiment B.

1. **Enable Google UA integration**

   Optimizely uses Google UA's [custom dimensions](#) to tag your visitors with the experiments and variations to which they've been added. Here are the steps in configuring Optimizely to start sending this information to Google UA:
1. Choose an experiment.

2. In the Manage Experiment menu, click Integrations.

3. In the Google Universal Analytics section, click to select the Tracked checkbox.

4. Select the custom dimension that you want Optimizely to use. Remember, this is a Google UA custom dimension, not the Optimizely feature. Make sure that this custom dimension isn't already in use for another part of your site or a concurrently running Optimizely experiment.

   **Important:**

   After you create your custom dimension in Google UA, Google will prompt you with example JavaScript for your page. However, enabling the Google UA integration means that you don't need to include the UA `setDimension` code on your page. If you include the `ga('set', 'dimension1', dimensionValue);` code on your site, you might overwrite the Optimizely-Google UA integration.

5. If you're using a custom event tracker other than the default (usually named `t0` by Google UA), specify it in the Custom Tracker (optional) field. This changes Optimizely's integration call to use the custom event tracker instead of the default. For example, if your website uses this call:

   ```javascript
   ga('tracker2.send', 'pageview');
   ```

   ...you'll enter tracker2 in the Custom Tracker (optional) field, and Optimizely will integrate with tracker2 instead of the default tracker.

If you are using Google's global site tag `gtag.js` tracking code, read the additional step section about the necessary custom tracker naming convention.

Additionally, if you have input a tracker name into Optimizely's integration and are unable to see the event send
successfully, use Google's Analytics Debugger extension to determine the trackers created on the page. Check if the non-interaction event is being ignored as this would indicate the tracker was never created by Google and is usually the reason the event fails.

2. Create a Google UA custom dimension

To integrate Optimizely with Google UA and view reports based on the integration data, you need a Google UA custom dimension. This custom dimension and the name you give it are used to create reports that leverage your Optimizely integration data for filtering.

Here’s how to create a Google UA custom dimension:

1. Log into your Google UA account as an Admin and click the Admin tab.

2. Click Custom Definitions and select the Custom Dimensions sub-item.
If you see the custom dimension you want to integrate your Optimizely experiment with, note its name and index number—no need to add a new one.

If you want to integrate with a custom dimension that isn’t listed, you do need to create a new one. Click *New Custom Dimension*, and give the new dimension a name and a scope.

3. Click *Create*.

Your custom dimension is now available for integrating with Optimizely experiments.

Although we recommend integration scope at the session level, you can configure your integration scope at the user level. Optimizely’s default integration uses session-level scope to minimize the possibility of overwriting your experiment data sent to Google UA. But if you’re tracking complex conversions that span multiple sessions, you might prefer a user-level scope. Choose the scope that’s better for the types of conversions you want to track.
For example, you might want a **session-level** scope if the experiment is running site-wide or if a visitor will always go back through the experiment pages before converting. But you’d want a **user-level** scope if it’s possible for a user to leave the site, then come back and convert without seeing the experiment again.

If you do this, be cautious of reusing custom dimensions in Google UA. For consistency between Google UA and Optimizely’s Results page, set the Google UA reporting range to the same duration as your experiment.

Check out [Google’s documentation on scoping](https://developers.google.com/analytics/devguides/collection/analyticsjs/scoping) for details.

### 3. Create a custom report

You’ll want to see reports on your experiments within Google UA, so you’ll need a custom report for each experiment. Here’s how to create custom reports:

1. Log into your Google UA account and click the *Customization* tab.

![Customization tab in Google Analytics](image)

2. Set up a custom report for each experiment with which you have integrated Google UA: click *New Custom Report*, give the report a title, and add the metric groups you want to view in the report.

3. To filter the report for only your Optimizely experiment, choose the custom dimension you set up as one of the *Dimension Drilldowns*.

4. Add the custom dimension in the *Filters* section.

5. Enter the *regular expression* ("Regex") match for the **experiment ID** or **campaign ID** you want to filter for. Use [Experiment IDs](https://developers.google.com/analytics/devguides/collection/analyticsjs/variables#variable-experimentid) for Web Experiments and [Campaign IDs](https://developers.google.com/analytics/devguides/collection/analyticsjs/variables#variable-campaignid) for Optimizely Personalization Campaigns. This will ensure that your report shows only the Google UA data for that particular experiment.
Important:

Leave the fields in the Filters section empty when you're creating a custom report for an iOS experiment.

6. Click Save.

Now you can access your report and update the date filter as needed!

4. Confirm your integration

Google UA's custom dimensions contain visitor-specific information that can be sent along with events and pageviews. It can take up to 24 hours to populate that data into Google UA, but the custom dimension is set immediately when you integrate Google UA and Optimizely. This means you can confirm that the custom dimension is set correctly with the Google Analytics’ debugger tool.

Using the debugger tool, look for the variable `cd[x]`, where `x` is the slot that corresponds to the Optimizely experiment. The variable is set with the Optimizely experiment, followed by the variation you were bucketed into. Here's an example:

```
dimension7 (&cd7) Optimizely_Onboarding (1645002506): Variant#1
```
If you would like to learn more on our naming convention, please take a look at our "Naming Conventions for Analytics Integrations" article.

Troubleshooting your Google UA integration or seeing discrepancies in your reports? See our GA troubleshooting article.

When a decision event is made, Optimizely sends that event to Google UA as soon as possible, polling for availability of the Google UA object.

If the Google UA object isn't ready when a decision is made, Optimizely checks back every 200 ms for 10 seconds see if Optimizely can send the UA string. This helps with delayed activation experiments (and if they fall under delayed activation, it helps with geo-targeted experiments too).

---

**Additional step for gtag.js tracking code**

Google Analytics offers two different tracking code blocks: the traditional analytics.js and the global site tag gtag.js. If you are using gtag.js you will need to specify a custom tracker name (per step 1-5 above). This tracker is generated by Google and follows the naming convention of gtag_UA_XXXX_X where UA_XXXX_X is the tracking id. For example if your tracking id is UA-12345-1, your custom tracker should be gtag_UA_12345_1:

![Google Universal Analytics](image)

If sent successfully, you should be able to use the Google Debugger extension to confirm that the GA command with the analytics string was sent to Google Analytics. That command may look like the following:

```javascript
Running command: ga("gtag_UA_12345_1.send", "event", "Optimizely", "Assigned To Campaign", {nonInteraction: true, dimension1: "experimentName(experimentId):variationName(variationId)"})
```