Custom code, shared code, project Javascript and extensions: when to use each

Skip Ahead

Use custom / shared code and project Javascript
Use extensions
Add custom code, shared code and project Javascript
Add extensions

THIS ARTICLE WILL HELP YOU:

• Understand the differences between custom code, shared code and project Javascript.
• Add custom code, shared code and project Javascript to your variations, experiments and projects.

Optimizely allows you to use custom code, shared code, or project JavaScript to make powerful changes to your site and extend your testing capabilities even beyond Optimizely’s standard toolkit.

For example, you could use them to track certain specific metrics that are not included as default options in the metrics builder, like scroll depth or cart abandonment.

Though they share similar objectives, one significant difference between project Javascript, shared code, and custom code is scope:

• Custom code is used at the variation level;
• Shared code is used at the experiment level;
• Project Javascript runs snippet wide which means it can potentially affect all experiments. Functions and variables you define here will exist within its own special scope but you can also define them at the Window level to be accessible globally.
Project Javascript is the first point of execution in the Optimizely snippet order of execution, which means you don't have to change your native site's code in order to run more advanced experiments.

**Use custom / shared code and project Javascript**

Previously, there were more situations under which using custom code or project Javascript was the best option. However, several of those situations are now best handled by using extensions instead.

Custom code, shared code, and project Javascript are good solutions in situations like these:

- You want to track unique custom events, such as scroll depth and mouse hover.

- You want to add API calls for custom metrics.

- You want to run experiments on single-page apps that execute without requiring the user to reload the page.

- You want to send data from Optimizely to your analytics service. Custom / shared code or project Javascript are acceptable solutions for ad-hoc cases, but if this is a situation that comes up regularly, you should consider the Optimizely analytics extension instead.

You can typically use any of the three options for any of the use cases listed above. Your choice should be determined by whether you are working at the variation level (in which case, use custom code), the experiment level (shared code), or at the project level (project Javascript).

**Use extensions**

You should use extensions instead of custom code, shared code, or project Javascript in cases like these:

- You want to reuse custom modules, like lightboxes or banners, across multiple experiments. Here, you should use editor extensions instead.

- You want to manage multiple analytics integrations on the same page. Optimizely’s analytics extensions accomplish this goal more efficiently than writing your own code from scratch.

- You want to add functions that can be used across multiple variations, and then call them with different parameters in the variation code. Here, the better approach depends on the context. Custom code or project Javascript can certainly be used for ad-hoc development cases, but Optimizely also provides editor extensions that offer more reusable options in cases like this.

**Add custom code, shared code and project Javascript**

To include Project Javascript in your project, navigate to the JavaScript tab in the Settings window.
To include **Shared Code** in an experiment, open the Shared Code dashboard in your experiment and locate the Javascript tab. Then paste your code in the window provided. See this article on [shared code in Optimizely](#) for more detailed instructions.

Lastly, to include **Custom Code** in a variation, open that variation in the Visual Editor, find the Javascript tab in the Variation Code Editor window, and paste your Javascript in the space provided. See this article on [custom code in Optimizely](#) for more detailed instructions.

**Add extensions**

The process of adding extensions is a bit more involved than the procedures for custom code, shared code, and project Javascript. To learn more about how to add extensions, [visit our developer docs](#).