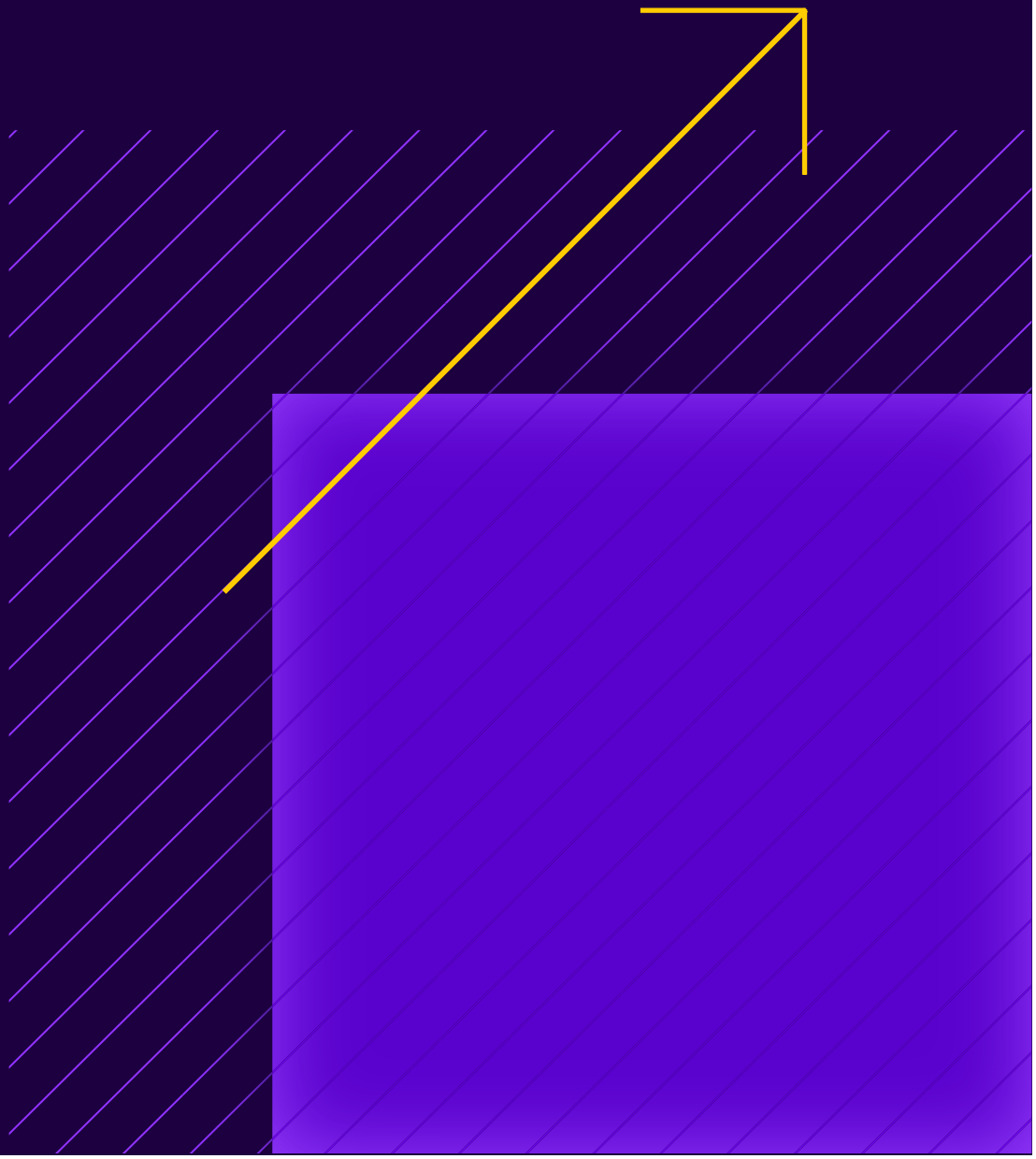
Maximizing Performance

The Guide to Implementing Optimizely Web



Optimizely

Background

Performance has become increasingly important in recent years. In this short guide we aim to highlight some of the most performant ways to implement Optimizely Web on your site today. Your business is unique, and you have the choice and flexibility to implement Optimizely based on your company needs.

THE CHALLENGE

To survive in today's competitive, fast-moving digital market, it's imperative that you drive high-velocity experimentation and keep your site performance as fast as possible. When it comes to running client-side experimentation there is a natural tradeoff that has to be made between performance, user experience (ensuring experimentation is invisible to your customers), and the operational costs of a custom implementation.

THE SOLUTION

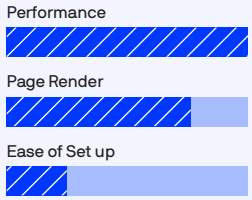
In this guide we highlight a menu of options that profile some of the different configurations for Optimizely Web. Some teams want to maximize performance above anything else, while others care more about ensuring they've eliminated any chance of page flicker. In the pages ahead, we will walk through the benefits of these different approaches and outline all of the best practices you can be doing today to improve your performance.

Introducing a Menu of Options

Optimizely Web provides multiple implementation methods that you can tailor to meet your needs. These are some of the configurations used by our most performance-focused customers.

Fast Performance

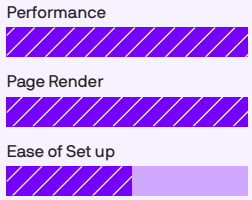
Non-blocking
Self Hosted
Custom Snippets*



Implementing the Optimizely snippet asynchronously allows the rest of the page to continue to render while Optimizely works — masking will also prevent any chance of flashing on the page.

Fastest Performance + Seamless Page Render

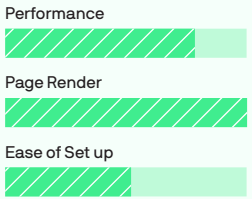
Performance Edge*



Performance Edge works with Optimizely Web to move processing from the browser to the Edge for dramatically faster A/B test performance. The implementation is simple and can be flexible, depending on your experimentation needs.

Seamless Page Render

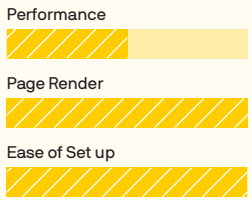
Synchronous
Self-Hosted
Custom Snippets



This approach uses the experimentation best practice of loading the snippet synchronously but allows you to improve performance from the standard implementation by “self-hosting” the snippet. Requires lengtheir implementation up-front.

Minimize Set-up

Synchronous
Optimizely Hosted
Standard Snippet



This is the “out-of-the-box” Optimizely Web configuration. Implementing the snippet in the head of your page synchronously will ensure there is no flicker or impact onyour customers’ experience.

Custom Snippets*
This is an Enterprise feature that allows customers to use multiple snippets for each project as a way to reduce snippet size and improve performance.

Performance Edge*
This is an add-on that allows customers to use Optimizely’s Experiment Network Delivery. With the Edge snippet, Experiments can run in less than 50ms.

DID YOU KNOW? Full Stack is Optimizely’s server-side experimentation product that integrates with your stack via SDKs. If you want to remove the performance trade offs above, and are staffed with engineers or product development teams that don’t require a visual editor, you should consider Optimizely Full Stack.

Performance Best Practices

(regardless of the menu choice)

Reduce the Snippet Download Time

You can prioritize the download of the snippet over other assets on the page by adding what's called a preload tag for the snippet URL. Additionally, adding a preconnect tag will tell the browser that sometime in the session, you'll be sending events to Optimizely — so opening that connection up-front can ensure your site doesn't have to wait for a response later. It's also worth highlighting actions such as CDN warming and cache expiration, which also impact download time measurements.

Manage Your Projects

The size of the Optimizely Snippet grows based on usage. The more JavaScript customers add to their snippet (by way of custom code, Pages, and Experiments), the longer it takes the browser to execute Optimizely specific code. With this in mind, you should be archiving experiments and pages that aren't being used. You can also create multiple projects for different areas of your site to minimize snippet bloat.

Upgrade to Custom Snippets

Customers with a significant number of testing touchpoints should utilize Custom Snippets to ensure they're only including what's needed to test in different places. Custom snippets does require an Enterprise software license.

Eliminate Redundant jQuery

(If your site uses jQuery)

By default, Optimizely X doesn't use jQuery. Check with your team to see if jQuery is already on your site. If you need jQuery you can choose to include it in the snippet, or you can choose to use Optimizely with your own copy of jQuery — but be sure you're not using two copies of jQuery.

Experiment Smarter

If you're doing a full page redesign or implementing a heavy change that requires more than 200 lines of code, consider running this as a "redirect test" in Optimizely to eliminate the extra weight on the snippet. You can also apply individual variation changes asynchronously — regardless of your snippet choice — so that if an element appears beneath the fold it can be applied non-blocking to improve performance.

Increase the Cache Expiration

Increase the Cache Expiration of the snippet to a time that makes sense for the pace of experimentation, such as 10 or 20 minutes. Longer cache expiration times improve site performance — the default setting is 2 minutes.

Implement The Right Optimizely For You

Not all sites should implement Optimizely the same way. So, it is critical to consider your available configurations and how each element contributes to improving performance under various conditions. Put simply, addressing performance requires a thoughtful and deliberate approach to implementation. In spite of all the checks and controls, there are times that snippet performance can still be a concern, depending on the snippet implementation. Below is the complete list of decisions you can make to implement Optimizely, complete with how it affects important tradeoffs such as site performance, ease of implementation, and its ability to reduce content flicker on page load.

As you can see below, Performance Edge is something every team that is prioritizing webpage speed, should consider. Learn more about how to get started.

		Performance	Ease of Implementation	Ease of Use	Flicker
Script Execution	Synchronous	+	+	+	+
	Asynchronous	+	+	+	-
	Async w/Masking	+	-	-	+
	Performance Edge	+	+	+	+
Download Time	Optimizely Hosted	-	+	+	+
	Self-Hosted	+	-	-	+
	Performance Edge	+	+	+	+
Project Size	Standard Snippet	-	+	+	+
	Custom Snippets	+	-	+	+
	Performance Edge	+	+	+	+

Next Steps

Resourcing Performance Needs

Depending on which implementation option you choose and how much focus you want to put towards maximizing performance benefits, this project is something that needs to be resourced properly. While everything here can be done by your in-house team with Optimizely documentation, we also provide professionals services to speed up this process. Please contact your Customer Success Manager to learn more about adding services to your subscription today.

Learn More

To learn more about how to change your implementation today, please visit our Knowledge Base. For more information on self-hosting Optimizely, we have documentation for popular CDNs like Akamai, Fastly, Cloudflare, and Cloudfront. And if you are working with a Customer Success Manager, reach out to learn more about how you can make your site more performant with Optimizely.