JavaScript rendering or blocked resources: diagnosing indexing issues in Google Search

Why some pages never make it into Google's index

Every SEO professional has faced the same frustration: a page that looks perfect in a browser but never appears in search results. The reasons often come down to two interconnected problems, how Google handles JavaScript rendering and how blocked resources affect crawl budget. Understanding the difference between these two issues is essential for anyone working on indexing issue diagnosis. Without that clarity, you risk chasing the wrong fix while the real cause stays hidden.

Google has become much better at rendering JavaScript over the years, but it is still not perfect. The search engine's crawler, Googlebot, queues pages for rendering after it finishes crawling the raw HTML. That rendering process takes time and resources. If your site relies heavily on JavaScript to load content, links, or metadata, you introduce a point of failure. Google may render the page partially or not at all, depending on how complex the JavaScript is and how much crawl budget the site has left after processing other resources.

How JavaScript rendering works in Google's pipeline

When Googlebot first visits a URL, it downloads the HTML. It then parses that HTML for links and resource hints, such as stylesheets and JavaScript files. The crawler queues the URL for rendering in a separate, more resource-intensive stage. During rendering, Googlebot executes JavaScript in a headless browser. This step can reveal content that was hidden behind JavaScript logic, such as product descriptions loaded via an API call or navigation links injected by a script.

The problem is that rendering is not guaranteed. Google has a limited rendering budget for each site. If a page requires many JavaScript files, or if those files are slow to load, the rendering process may time out. When that happens, Google indexes only the raw HTML, which might be a skeleton page with little useful content. This is a common source of indexing gaps for sites built with modern JavaScript frameworks like React, Angular, or Vue.



Signs your JavaScript rendering is failing

You can spot JavaScript rendering problems by comparing what Google sees with what a user sees. Use the URL Inspection Tool in Google Search Console. If the inspected page shows a different version than what appears in a browser, rendering is likely incomplete. Another clue is a sudden drop in organic traffic for pages that rely on JavaScript for core content, such as category pages that load products via an API call. Also check the "crawled as" date in Search Console. If Google last crawled the page weeks ago but your site changes daily, the rendering queue may be too slow.

Blocked resources and their impact on crawl budget

Blocked resources are files that Googlebot cannot access because of robots.txt directives, noindex tags, or authentication walls. Every time Googlebot tries to fetch a resource and gets blocked, it wastes part of its crawl budget. Crawl budget is the number of URLs Google will crawl on your site within a given time frame. When a significant portion of that budget is spent on blocked resources, fewer pages get crawled and indexed.

Common blocked resources include CSS files, JavaScript files, images, and fonts. If you accidentally block a JavaScript file that is essential for rendering, Google may not be able to execute the page's JavaScript at all. That creates a double problem: the resource is blocked, so rendering fails, and the crawl budget is wasted on repeated attempts to access it. Blocked resources also affect how Google understands the page structure. For example, if your CSS is

blocked, Google may not see the page layout correctly, which can influence its assessment of mobile usability.

How to check for blocked resources

Start with the robots.txt file. Look for rules that disallow access to common resource directories like /js, /css, or /fonts. Use the robots.txt Tester in Google Search Console to see whether Googlebot can reach specific URLs. Also review your server logs for 403 or 404 responses on resource files. A high ratio of blocked or missing resources relative to total crawled URLs is a clear signal that crawl budget is being wasted.

Comparing the two issues: rendering vs blocked resources

JavaScript rendering problems and blocked resources share a common symptom, pages that do not appear in search results, but they have different root causes and different fixes. Rendering problems occur when Google can execute JavaScript but runs out of time or capacity. Blocked resources occur when Google is prevented from even downloading the files it needs. In practice, the two often overlap. A site that blocks its JavaScript files will also suffer from poor rendering, because Google cannot run scripts it cannot access.

The key difference lies in how you diagnose each issue. For rendering problems, you inspect the rendered page output and the timing of JavaScript execution. For blocked resources, you check server logs and robots.txt rules. A thorough indexing issue diagnosis should cover both angles. If you only look at rendering, you might miss a simple robots.txt mistake that is starving your site of crawl budget. If you only look at blocked resources, you might overlook a slow-loading JavaScript file that causes rendering timeouts.



Practical steps for diagnosing each problem

For JavaScript rendering issues, use the following approach. First, test a few key pages with the URL Inspection Tool. Compare the "crawled" version with the live version. If they differ, check whether the JavaScript files are being loaded. Use the "View crawled page" option to see what Google actually rendered. If the page is mostly empty, you need to reduce JavaScript dependency or implement server-side rendering. Second, monitor the rendering queue in your server logs. Look for patterns where Googlebot requests many JavaScript files but then stops abruptly. That suggests a timeout.

For blocked resources, start with a crawl of your site using a tool that respects robots.txt. Identify every resource that returns a 403 or 404. Then decide whether each resource should be accessible to Googlebot. For essential CSS and JavaScript files, remove the disallow rule from robots.txt. For resources that are genuinely not needed for indexing, leave them blocked but make sure they are not critical for rendering. Also check for accidental noindex tags on resource directories. These can happen when a developer adds a blanket noindex rule to a subdomain.

A Simon White SEO Fire Fighter technical SEO audit covers both of these areas in depth. The audit examines how Googlebot interacts with your site's resources, identifies rendering bottlenecks, and checks for blocked files that waste crawl budget. That kind of systematic review is often the fastest way to locate the specific cause of an indexing shortfall.

After you have identified the problem, the fix depends on the root cause. For rendering issues, consider moving critical content into the initial HTML, using server-side rendering, or pre-rendering pages on the server. For blocked resources, adjust robots.txt rules and review your site's resource architecture to ensure that essential files are always accessible. In both cases, monitor the results in Search Console over the following weeks. Indexing improvements often take time because Google needs to recrawl and re-render the affected pages.

When to prioritize one over the other

If your site is built with JavaScript frameworks and you see a large number of pages that return only a basic HTML skeleton, focus on JavaScript rendering first. That is likely the primary bottleneck. If your site is more traditional but you notice that Google crawls only a small fraction of your pages each day, blocked resources might be the culprit. Check your crawl stats in Search Console. If the ratio of crawled to indexed pages is low, and many crawled URLs return errors, blocked resources are probably wasting your budget.



For most sites, both issues need attention. A Simon White SEO Fire Fighter technical SEO audit integrates both checks into a single workflow, which saves time and ensures nothing is missed. The goal is to create a clean path for Googlebot to crawl, render, and index every page that deserves to be in search results. When rendering and resource access work smoothly, crawl budget is used efficiently, and your index coverage improves naturally.

Ultimately, indexing issue diagnosis comes down to understanding how Google sees your site. JavaScript rendering and blocked resources are two of the most common barriers. By learning to distinguish them, you can apply the right fix and get your pages into the index faster.