ItkCriticalCss

Anwenderdokumentation

André Fischer

30.03.2026

Inhaltsverzeichnis

1 Was macht dieses Plugin? 2
1.1 Diagramm: Was beim Laden passiert 2

2 Wann muss Critical CSS neu generiert werden? 3

3 Die Plugin-Konfiguration 3
3.1 Hauptschalter 3
3.2 Service URL (optional) 4
3.3 Preconnect 4
3.4 LCP-Optimierung 4
3.5 Weitere Optimierungen 4

4 Was tun wenn der Shop nach einer Theme-Änderung komisch aussieht? 5

5 Troubleshooting 5
5.1 Die Seite flackert beim Laden (FOUC – Flash of Unstyled Content) 5
5.2 Schriftarten laden verzögert 6
5.3 Das Hero-Bild lädt langsam 6

6 FAQ 6

1 Was macht dieses Plugin?

Google bewertet Webseiten unter anderem danach, wie schnell sie für Benutzer nutzbar sind. Die Messwerte dafür heißen **Core Web Vitals** und haben direkten Einfluss auf das Ranking in den Suchergebnissen. Das Plugin ItkCriticalCss ist dazu da, genau diese Messwerte zu verbessern.

Konkret kümmert es sich um:

Critical CSS (Kritisches CSS): Wenn eine Webseite geladen wird, lädt der Browser erst alle CSS-Dateien herunter, bevor er die Seite anzeigt. Das dauert – und in dieser Zeit sieht der Nutzer entweder gar nichts oder eine unstyled Seite, die dann kurz "aufblitzt". Critical CSS löst das, indem die CSS-Regeln, die für den sofort sichtbaren Bereich der Seite (über dem "Fold", also ohne Scrollen) benötigt werden, direkt in den HTML-Quelltext eingebettet werden. Der Browser kann die Seite dann sofort rendern, ohne auf externe CSS-Dateien warten zu müssen.

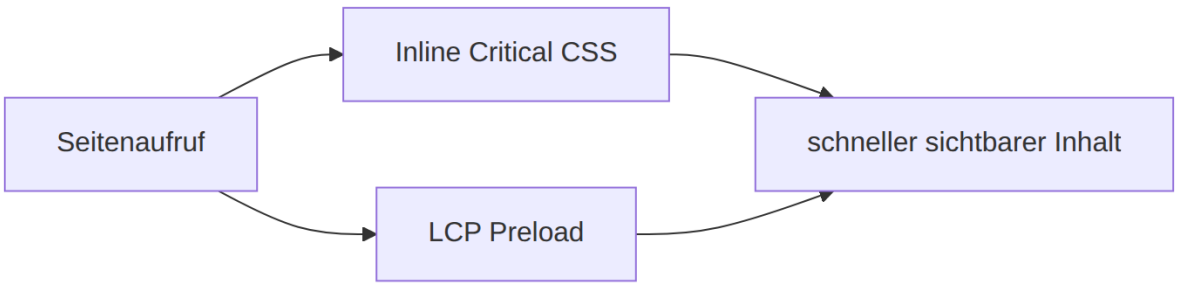
LCP-Optimierung (Largest Contentful Paint): Das ist der Google-Messwert dafür, wie schnell das größte sichtbare Element einer Seite (meist ein Hero-Bild) geladen ist. Das Plugin kann das Hauptbild mit höchster Priorität vorladen, damit es schneller erscheint.

CLS-Prävention (Cumulative Layout Shift): Wenn sich Elemente beim Laden verschieben – z. B. wenn ein Bild zunächst ohne definierte Höhe geladen wird und dann auf einmal Platz einnimmt – zählt das als Layout-Shift und verschlechtert den CLS-Score. Das Plugin kann CSS-Regeln ergänzen, die solche Verschiebungen verhindern.

Ressource-Preloads: Schriftarten und andere wichtige Ressourcen können vorab geladen werden, damit sie sofort verfügbar sind.

Das alles zusammen zielt darauf ab, dass der Shop bei Google PageSpeed Insights bessere Werte bekommt und potenziell besser rankt.

1.1 Diagramm: Was beim Laden passiert



2 Wann muss Critical CSS neu generiert werden?

Das ist die wichtigste praktische Information für den täglichen Betrieb: Critical CSS ist kein “einmal einstellen und vergessen”-Feature. Es muss bei bestimmten Änderungen am Shop neu generiert werden.

Critical CSS neu generieren nach:

- Änderungen am Theme oder Theme-Einstellungen (Farben, Fonts, Layout)
- Einem Theme-Kompilierungsvorgang (“Theme neu kompilieren” in Shopware)
- Strukturellen Änderungen am Storefront (neue Plugins die Layout-Blöcke verändern, CMS-Änderungen an der Startseite)
- Wenn das LCP-Hauptbild geändert wurde

Automatik vs. manuell:

- **CLI:** Nach erfolgreichem **bin/console theme:compile** startet **ThemeCompileSubscriber** die Generierung für die gewählten Sales Channels nacheinander.

- **Admin:** Beim Theme-Kompilieren im Backend feuert Shopware **ThemeCompilerConcatenatedStylesEvent** – **ThemeCompileBackendSubscriber** ruft dann `CriticalCssGenerator::generate($salesChannelId)` für den jeweiligen Kanal auf. Ein zusätzlicher Schutz verhindert dabei, dass derselbe Lauf im CLI-Kontext doppelt startet.
  Schlägt die Auto-Generierung fehl (Node fehlt, **serviceUrl** down, …), findest du einen Eintrag im Log – dann `itk:critical:diagnose` und ggf. **itk:critical:generate –sales-channel-id=** manuell. FOUC nach Theme-Änderung = oft noch **altes** Critical CSS.

---

3 Die Plugin-Konfiguration

Unter **Erweiterungen → Meine Erweiterungen** → **Critical CSS Generator** → **Konfiguration** (pro Sales Channel oben wählbar).

3.1 Hauptschalter

Critical CSS einbinden ist der wichtigste Schalter. Wenn er aus ist, macht das Plugin gar nichts – die Seite verhält sich wie ohne Plugin. Das ist praktisch für den Notfall: Wenn Critical CSS zu Problemen

führt, kann man es hier schnell ausschalten und normale Funktion wiederherstellen, ohne das Plugin zu deinstallieren.

3.2 Service URL (optional)

Service URL: Wenn du hier eine URL einträgst, holt sich das Plugin das Critical CSS per **HTTP GET** von diesem Dienst (**url=** wird angehängt). Das ist praktisch, wenn ihr **kein Node.js** auf dem App-Server habt, aber einen zentralen Generator betreibt.

Wenn die Service-URL leer ist, erzeugt der Generator lokal per `Node.js` das Skript `generate-critical.js` (Paket **critical** via **npm install** im Plugin-Ordner). Dafür müssen auf dem Rechner, auf dem du **itk:critical:generate –sales-channel-id=...** ausführst, **Node** und die `node_modules` vorhanden sein – das ist typischerweise eure Build-/Deploy-Pipeline oder der Entwickler-PC, nicht zwingend der Live-Webserver.

3.3 Preconnect

Preconnect-URLs: Hier können Domain-Namen eingetragen werden (kommagetrennt), zu denen der Browser frühzeitig eine Verbindung aufbaut. Das ist nützlich für externe Ressourcen die auf der Seite genutzt werden (z. B. wenn Videos von einem CDN kommen).

Font-Preconnect-URLs: Ähnlich, aber speziell für Font-Quellen, bei denen ein **crossorigin**-Attribut nötig ist. Z. B. `fonts.googleapis.com`, `fonts.gstatic.com` wenn wir Google Fonts nutzen.

3.4 LCP-Optimierung

LCP-Preload-URL: Die exakte URL des Hauptbildes, das auf der Startseite als erstes sichtbares großes Element erscheint (das LCP-Element). Das Plugin lädt dieses Bild mit der höchsten Priorität vor. Die URL muss exakt stimmen – inkl. vollständigem Pfad. Wenn sich das Hauptbild ändert, muss auch diese URL aktualisiert werden.

LCP-Bildoptimierung: Wenn aktiviert, setzt das Plugin an relevanten Bild-Elementen die Attribute **loading="eager"** und **fetchpriority="high"**, damit der Browser sie sofort und mit Priorität lädt.

3.5 Weitere Optimierungen

Theme-CSS vorladen (empfohlen aktiv): Das Theme-CSS wird mit einem Preload-Tag eingebunden, damit der Browser es früher anfordert.

Async Stylesheets: Laut Hilfetext in der Config ist das Feld vor allem zur **Kompatibilität** da; bei aktivem Critical CSS wird das Theme-CSS ohnehin **non-blocking** geladen – du musst hier meist nichts ändern.

Thumbnail: lazy als Standard: Wenn an, bekommen Bilder **ohne** eigenes **loading**-Attribut standardmäßig **loading="lazy"** (gut für Listen). **LCP-Kandidaten** mit hoher Priorität bleiben laut Konzept **eager** – trotzdem nach Deploy kurz in PageSpeed prüfen.

CLS-Prävention: Das Plugin hängt **seitentypspezifische** Regeln an (Startseite, Produkt, Kategorie, …) und ergänzt sie ggf. um das generierte Critical CSS. Wenn etwas „zu fest" wirkt, kannst du die Option testweise ausstellen.

---

4 Was tun wenn der Shop nach einer Theme-Änderung komisch aussieht?

Das ist ein klassisches Critical-CSS-Problem. Das zuvor generierte Critical CSS enthält CSS-Regeln vom alten Theme. Das neue Theme hat möglicherweise andere Klassen, andere Farben, andere Abstände. Das veraltete Critical CSS wird dann im **head** inline eingebettet und überlagert kurz das neue Theme-CSS – das führt zu visuellem Flackern oder falschen Stilen.

Sofortlösung: "Critical CSS einbinden" in der Plugin-Konfiguration deaktivieren. Damit ist das Critical-CSS-Problem sofort weg, die Seite lädt etwas langsamer, aber korrekt.

Richtige Lösung: Critical CSS **pro betroffenem Sales Channel** neu generieren lassen: **php bin/console itk:critical:generate –sales-channel-id=** (oder `theme:compile` per CLI, siehe oben).

---

5 Troubleshooting

5.1 Die Seite flackert beim Laden (FOUC – Flash of Unstyled Content)

Das deutet entweder auf veraltetes Critical CSS (Theme wurde geändert, CSS nicht neu generiert) oder auf ein Problem beim Generieren hin. Als erste Maßnahme Critical CSS deaktivieren und das Team informieren.

5.2 Schriftarten laden verzögert

Prüfen ob die Font-Quellen-Domains in den Preconnect-URLs eingetragen sind. Wenn Google Fonts genutzt werden: `fonts.googleapis.com` und `fonts.gstatic.com` dort eintragen.

5.3 Das Hero-Bild lädt langsam

Prüfen ob die exakte Bild-URL in **LCP-Preload-URL** eingetragen ist. Die URL muss identisch mit der src-URL des Bildes im HTML sein. Falls sich die URL je nach Viewport ändert (responsive Images), ist das schwieriger zu lösen – Entwicklungs-Team fragen.

---

6 FAQ

Wo liegen die generierten Critical-CSS-Dateien?
Im Plugin unter `custom/plugins/ItkCriticalCss/.../storefront/critical/` (pro Kanal z. B. `critical_<salesChannelId>.css` und optional `critical_product_…`). Details in der Entwicklerdoku.

Warum sieht die Produktseite anders aus als die Startseite?
Es gibt **zwei** generierte Dateien (Home + Produktdetail). Wenn die Produktdetail-Generierung fehlschlägt, greift ein Fallback auf die Homepage-CSS.

Muss ich nach Theme-Compile immer `itk:critical:generate` laufen?
In der Regel **nein**: nach `theme:compile` und nach **Admin-Kompilieren** versucht das Plugin die Generierung selbst (siehe oben). Nur bei **Fehlern** oder wenn ihr gezielt einen Kanal ohne erneutes Kompilieren neu erzeugen wollt: **itk:critical:generate –sales-channel-id=…**; zur Analyse `itk:critical:diagnose`.