

Inhalt

1	Einleitung	1
1.1	Machen Sie aus Ihrer Website einen Ferrari	1
1.2	Zielgruppe	2
1.3	Beispiele	4
1.4	Warum Sie aus Ihrer Webseite einen Ferrari machen sollten	7
1.4.1	Mehr Pagespeed = mehr Besucher	7
1.4.2	Mehr Pagespeed = weniger Absprünge (Bounce Rate)	7
1.4.3	Mehr Pagespeed = mehr besuchte Seiten	8
1.4.4	Mehr Pagespeed = mehr Umsatz	9
1.5	Vorgehen	11
2	Wie kommt eine Webseite auf unseren Computer?	13
2.1	Anfrage des Browsers an den Webserver	13
2.2	Behandlung des Requests auf dem Server	14
2.3	Übertragung	15
2.4	Anzeigen der Seite im Browser (Rendering)	16
3	Werkzeuge für die Pagespeed-Optimierung	19
3.1	Google PageSpeed Insights	19
3.2	WebPageTest	21
3.3	Test der Internetanbindung eines Webserver	23
3.4	Welche Ladezeit ist gut?	24
3.5	Mit FTP eine Webseite bearbeiten	24
3.6	Mit SSH auf einen Webserver zugreifen	25
3.7	Vorbereitung	27
4	Bilder optimieren	29
4.1	Die richtige Bildgröße	29
4.2	Unterschiedliche Bildgrößen für unterschiedliche Geräte	32
4.3	Welches Bildformat ist das schnellste?	34

4.4	WebP: das richtige Dateiformat	34
4.4.1	Wie kann man WebP-Bilder erstellen?	34
4.4.2	Fallback im CSS	36
4.4.3	Fallback in HTML 5	37
4.4.4	Per .htaccess unterschiedliches Bild ausliefern	37
4.5	Bilder komprimieren	37
4.6	Bildgrößen angeben	42
4.7	CSS statt einem Bild	43
4.8	Image-Map	44
4.9	Seiten mit sehr vielen und sehr großen Bildern	44
4.10	Bilder verzögert laden	44
4.11	Favicon optimieren	45
4.12	Unicode statt Grafiken	46
5	HTML optimieren	49
5.1	Die aktuelle HTML-Version verwenden	49
5.2	Auf Validität des HTML-Codes achten	50
5.3	Keine externen Elemente einbinden	51
5.3.1	Bilder kopieren	51
5.3.2	DNS-Prefetch	51
5.3.3	Externe Elemente verzögert laden	52
5.4	Frames vermeiden	52
5.5	Weiterleitung vermeiden	52
5.6	HTML reduzieren	53
5.7	Sichtbare Inhalte priorisieren	54
5.8	Fehlerhafte Anfragen vermeiden	55
5.9	Schriftarten optimieren	56
5.10	Websichere Schriften verwenden	57
5.11	Cookies vermeiden	58
5.11.1	Was sind Cookies?	58
5.11.2	Ausgleich zwischen Pagespeed und Benutzerfreundlichkeit?	59
6	CSS optimieren	61
6.1	Was ist CSS?	61
6.2	CSS-Dateien reduzieren (minify)	62
6.3	CSS-Dateien zusammenfassen und/oder inline einbinden	64
6.4	Wordpress – Deque CSS	66
6.5	print.css, mobile.css usw. nicht auf allen Seiten mitladen	67
6.6	Ungenutzte CSS-Styles entfernen	68
6.6.1	unused-css.com	68
6.6.2	Dust-Me	69
6.7	CSS-Import vermeiden	71
6.8	CSS nicht immer inline einbinden – aber wenn es sinnvoll ist	71

6.9	CSS nicht innerhalb von style-Attributen verwenden	72
6.10	CSS Sprites	73
7	JavaScript optimieren	75
7.1	Was ist JavaScript (JS)?	75
7.2	JavaScript reduzieren	75
7.2.1	Wie müssen Sie vorgehen?	76
7.2.2	Externe Dateien	77
7.3	Nicht benötigtes JavaScript entfernen	78
7.4	jQuery optimieren	79
7.5	JavaScript zusammenfassen	80
7.6	JS-Optimierung automatisieren mit GruntJS	81
7.7	JavaScript verzögert laden	83
7.8	JavaScript vs. CSS	84
7.9	Ladezeit von Benutzern ermitteln	85
8	Komprimieren	87
8.1	Was ist Datenkomprimierung?	87
8.2	Wie können wir mit Komprimierung unsere Webseite beschleunigen?	88
8.3	gzip auf Apache Webservern	88
8.3.1	Komprimierung mit mod_gzip	89
8.3.2	Komprimierung mit mod_deflate	89
8.4	gzip funktioniert nicht	90
8.4.1	Shared Hosting	90
8.4.2	gzip und deflate installieren	90
8.5	gzip auf Windows-Servern	92
9	Critical Rendering Path	93
9.1	Was ist der Critical Rendering Path?	93
9.2	Wie baut sich eine Seite auf? (Rendering)	95
9.3	CSS in Kritisch und Nichtkritisch unterteilen	95
9.4	Prefetch und Prerender	97
9.4.1	DNS-Prefetch	98
9.4.2	Prefetch	98
9.4.3	Prerender	99
10	Zwischenspeichern (Cache)	101
10.1	Was ist ein Cache?	101
10.2	Browser-Cache	101
10.3	Server-Cache	105
10.4	Wie können wir den Server-Cache nutzen?	106
10.4.1	WP Super Cache	106
10.4.2	W3 Total Cache	109
10.4.3	Externer Server-Cache	111

10.4.4	Caching mit Varnish	111
10.4.5	Statische Seite	116
11	CDN – Content Delivery Networks	121
11.1	Was ist ein CDN?	121
11.2	Wie richtet man ein CDN ein?	123
12	Webserver optimieren	125
12.1	Sie brauchen keinen teuren Server!	125
12.2	Hardware	126
12.2.1	Arbeitsspeicher (RAM)	126
12.2.2	Prozessoren und Prozessorkerne	127
12.2.3	Festplatten	127
12.3	Software	128
12.3.1	Betriebssystem	128
12.3.2	Webserver-Software	128
12.4	Webserver-Performance steigern	129
12.4.1	Shared-Hosting	130
12.4.2	Ein eigener Server?	131
12.4.3	Webserver auf Nginx wechseln	133
12.4.4	Webserver auf Litespeed wechseln	136
12.4.5	Keep Alive	138
12.5	Unnötige Anfragen blockieren	139
12.5.1	Wie kann man erkennen, dass eine Seite von vielen Crawlern besucht wird?	139
12.5.2	Wie kann man unerwünschte Crawler und Spam-Bots fernhalten?	141
13	Responsive Design	143
13.1	Warum responsive Design?	143
13.2	Webseite responsive machen	144
13.3	Abhängigkeiten vermeiden	144
13.4	Lösungsansätze	145
13.4.1	Elternelement ausblenden	145
13.4.2	Weiterleitung auf eine mobile Seite	145
13.4.3	Mit JavaScript Bilder erkennen	146
13.5	Viewport	147
13.6	Bilder anpassen	148
14	Ladezeit von HTTPS-Seiten optimieren	149
14.1	Was ist HTTPS?	149
14.1.1	Identifizierung	150
14.1.2	Verschlüsselung	151
14.2	SSL beschleunigen	152
14.2.1	HTTPS nicht auf allen Seiten verwenden	152
14.2.2	HTTPS-Handshake auf einem anderen Server durchführen	153

14.2.3	Server updaten	153
14.2.4	False Start	155
14.2.5	Weiterleitungen vermeiden	156
14.2.6	HSTS	156
15	Blick in die Zukunft	159
15.1	BPG-Bildformat	159
15.2	Google Transcode	160
15.3	Accelerated Mobile Pages	161
16	Fazit	163
Index		165