



Foto: Matthias Vietmeier

Drupal

Besseres Google-Ranking

Eine Ankündigung von Google sorgte im April 2010 für Aufregung. Die Leistung einer Website wird zukünftig eine immer größere Rolle bei der Ermittlung der Suchergebnis-Reihenfolge spielen. **Von Ronny Unger**

AUF EINEN BLICK

- Die Ladezeit einer Website ist nicht nur für Besucher entscheidend für die Akzeptanz der Website, sondern sie beeinflusst auch die Ranking-Position bei Google.
- Durch eine Standard-Funktion von Drupal kann die Anzahl der Requests durch CSS und JavaScript-Aggregation verringert werden.
- Mit Caching-Mechanismen, wie zum Beispiel dem Drupal-Page-Cache, einem Reverse-Caching-Proxy und dem Browser-Cache lassen sich die Ladezeiten einer Website reduzieren.
- Die Kombination von Varnish als Reverse-Caching-Proxy und Drupal optimiert die Auslieferung einer Website.

Verschiedene Studien haben bereits gezeigt, dass sich die Erfahrungen und Erwartungen der Website-Besucher stark verändert haben. Noch vor wenigen Jahren galt eine Ladezeit von sechs oder mehr Sekunden als akzeptabel. Das heißt, der Durchschnitt der Besucher war durchaus bereit, so lange zu warten, bis der Ladevorgang der besuchten Website abgeschlossen war. Nach aktuellen Erkenntnissen ist diese Toleranzschwelle mittlerweile auf etwa zwei Sekunden gesunken. Dauert der Ladevorgang länger, empfindet der Besucher die Seite als zu langsam und wird weniger Zeit auf ihr verbringen.

Gerade für komplexere Websites mit viel Interaktivität, wie zum Beispiel Communitys, die gern mit Drupal umgesetzt werden, ist dies eine wichtige Nachricht. Denn hier interagieren viele Nutzer gleichzeitig auf der Seite, wodurch der oder die Webserver stark beansprucht werden und sich die Auslieferungszeiten der einzelnen Seiten erhöhen. Dieser Artikel soll einen Überblick über mögliche Optimierungsschritte für Drupal verschaffen und einige Empfehlungen

geben, wie man auch von Google eine gute Performance bescheinigt bekommt.

Langsame Websites werden von Google zukünftig also schlechter bewertet, schnelle werden bessere Plätze erzielen. Zu Beginn wird der Faktor Geschwindigkeit nur einen kleinen Einfluss haben. Google selbst spricht von etwa ein Prozent aller Suchanfragen, bei denen es tatsächlich zu Änderungen kommen wird. Es kann aber mit großer Sicherheit davon ausgegangen werden, dass die Leistung zukünftig einen größeren Einfluss haben wird. Drupal bietet schon heute eine Vielzahl von Möglichkeiten, um die Leistung einer Website zu erhöhen.

Aber was bedeutet »Leistung einer Website« eigentlich? Hier können wenigstens drei Faktoren identifiziert werden: kurze Ladezeiten, hohe Verfügbarkeit und gute Skalierbarkeit. Für das Ranking bei Google spielt eine kurze Ladezeit die wichtigste Rolle. Sie umfasst die Zeit, welche vom Zeitpunkt einer Anfrage des Besuchers bis zur Anzeige im Browser vergeht. Um beurteilen zu können, ob die eigene Website zu den Schnel-

len oder Langsamen gehört, stellt Google seine Webmaster-Tools zur Verfügung.

Wie in **Bild 1** zu erkennen ist, liegt der Schwellenwert aktuell bei 1,5 Sekunden. Nur wenn die Ladezeit unter diesem Wert liegt, wird die Performance der Website von Google mit gut bewertet. Um die Performance einer Website kurzfristig zu optimieren, muss die Last auf dem oder den Webservern reduziert werden. Dazu bieten sich zwei Möglichkeiten, nämlich die Anfragen an den Webserver oder die Arbeit des Webserverns zu reduzieren.

Anfragen an den Webserver reduzieren

Für den Aufbau einer Website benötigt der Browser meist eine Vielzahl an Requests, um alle Javascript-, Stylesheet-, Flash- und Bilddateien zu laden. Für jeden dieser Requests muss der Browser eine Verbindung zum Server herstellen, die Datei anfordern, herunterladen und temporär speichern. Erst wenn alle Dateien geladen beziehungsweise alle notwendigen Requests abgearbeitet wurden, kann der Browser den Aufbau der Website abschließen.

Das Reduzieren der Anzahl dieser Anfragen ist eine sehr effektive Methode, um die Geschwindigkeit der Website zu verbessern. Wenn der Browser weniger Anfragen an den Webserver schicken und weniger zusätzliche Daten empfangen muss, kann er die Website schneller aufbauen und den Ladevorgang früher abschließen. Weniger Requests entlasten aber auch den Webserver. So hat dieser mehr Ressourcen für die Abarbeitung neuer Anfragen zur Verfügung.

Drupal bringt für diesen Optimierungsschritt zwei sehr gut geeignete Funktionen mit: Javascript- und CSS-Aggregation. Das Prinzip ist bei beiden Verfahren das gleiche: Aus einer Vielzahl von Javascript- oder CSS-Dateien wird eine einzige Datei generiert. Dafür wird beispielsweise der Inhalt aller Javascript-Dateien gesammelt und in eine neue Datei geschrieben. So muss der Browser im besten Fall nur noch eine zusätzliche Anfrage für Javascript und eine für CSS verschicken. Besonders deutlich ist dies zu erkennen, wenn man **Bild 2** und **Bild 3** miteinander vergleicht. Beide Einstellungen können in jeder Drupal-Installation, wie in **Bild 4** zu sehen, unter *Verwalten, Einstellungen, Leistung* vorgenommen werden.

Wenn zusätzlich darauf geachtet wird, dass die Antworten des Webserverns komprimiert zurückgegeben werden, hat man sowohl die Anzahl der Anfragen als auch das übermittelte Datenvolumen reduziert. Drupal bietet die Möglichkeit, die Komprimierung des HTML-Contents mit einem Klick zu aktivieren. Mit dem Drupal-Modul Javascript-Aggregator ist es außerdem möglich, die aggregierte Javascript-Da-



Aufruf der Site-Performance-Option in den Google Webmaster-Tools (**Bild 1**).

+	GET node.css?4	304 Not Modified	drupal-demo
+	GET admin.css?4	304 Not Modified	drupal-demo
+	GET defaults.css?4	304 Not Modified	drupal-demo
+	GET system.css?4	304 Not Modified	drupal-demo
+	GET system-menus.css?4	304 Not Modified	drupal-demo
+	GET user.css?4	304 Not Modified	drupal-demo
+	GET style.css?4	304 Not Modified	drupal-demo
+	GET print.css?4	304 Not Modified	drupal-demo

Notwendige CSS-Dateien vor der Optimierung (**Bild 2**).

+	GET css_c2283dd1e15fec	304 Not Modified	drupal-de
+	GET css_a532dc1ae0c94f	304 Not Modified	drupal-de

Notwendige CSS-Dateien nach der Optimierung (**Bild 3**).

Bandbreiten-Optimierungen

Drupal-Module können eigene CSS- oder Javascript-Dateien enthalten. Jede dieser Dateien wird vom Browser in einem separaten HTTP-Zugriff übertragen. Diese CSS- und Javascript-Dateien können jeweils in einer einzigen Datei zusammengefasst werden. Die CSS-Datei wird zusätzlich komprimiert. Durch die Verringerung der Anzahl (und Größe) der zu übertragenden Dateien wird die Serverlast reduziert, es wird eine geringere Bandbreite benötigt und der Ladevorgang der Website verkürzt sich.

Diese Optionen sind deaktiviert, solange das Dateiverzeichnis nicht eingerichtet ist oder die Download Methode auf privat eingestellt ist.

CSS-Dateien optimieren:

Deaktiviert
 Aktiviert

Es wird empfohlen diese Option nur zu aktivieren, wenn sich die Website in einer Produktionsumgebung befindet, da es die Themenentwicklung beeinträchtigen kann.

JavaScript-Dateien optimieren:

Deaktiviert
 Aktiviert

Es wird empfohlen diese Option nur zu aktivieren, wenn sich die Website in einer Produktionsumgebung befindet, da es die Modulentwicklung beeinträchtigen kann.

Konfiguration der Javascript- und CSS-Aggregation (**Bild 4**).

tei mit Minimize und GZip noch weiter zu verkleinern. **Tabelle 1** zeigt die Größenordnung, in der das Volumen reduziert werden kann.

Nachfolgend noch eine weitere Möglichkeit, die allerdings keinen direkten Bezug zu Drupal hat. Bei aufwendigen Layouts mit vielen Hintergrundbildern ist es sinnvoll, diese zu Sprites zusammenzufassen. Dabei wird aus vielen kleinen Bildern ein großes erzeugt. Mit dem CSS-Attribut *background-position* kann beeinflusst werden, welches Bild angezeigt wird. In **Bild 5** ist zu sehen, wie alle Icons aus dem Google-Reader zu einem Bild zusammengefasst sind. ▶

Arbeit des Webservers reduzieren

Hier lautet das Zauberwort Caching. Ein Cache kann den wiederholten Zugriff auf Daten beschleunigen, indem er die Daten in einen Zwischenspeicher schreibt. Dabei kann es sich um Ergebnisse von komplexen Berechnungen oder um ganze HTML-Seiten handeln. Zwischen dem Absenden der Benutzeranfrage und dem Darstellen der Website im Browser stehen verschiedene Caching-Möglichkeiten zur Verfü-

als Webbeschleuniger konzipiert und entwickelt wurde. Ihm zur Seite steht die Varnish Configuration Language (VCL), mit der das Verhalten des Servers gesteuert werden kann. So lassen sich das Caching, die Requests und Responses beeinflussen. Wie lange ein Cache-Eintrag in einem der Caches verbleibt, wird mit Hilfe von HTTP (Hypertext Transfer Protocol) gesteuert. Dafür stehen verschiedene Header im Protokoll zur Verfügung. Wie diese Header aussehen könnten, zeigt Bild 7. Einer der Wichtigsten ist der Cache-Control-Header. Er wurde mit HTTP 1.1 eingeführt, um die Schwachstellen der bisherigen Cache-Verwaltung zu beseitigen. Teile dieses Headers können sowohl im Request als auch im Response vorkommen.

TABELLE 1: EINSATZ VON MINIMIZE UND GZIP

Filename	Size	Mode
6e13ccb6262e06b9f890414db56d3b1f.js	289558 Bytes	Aggregated by Drupal Core
6e13ccb6262e06b9f890414db56d3b1f.jsmin.js	173243 Bytes	Minified with JSMIn
6e13ccb6262e06b9f890414db56d3b1f.jsmin.js.gz	47618 Bytes	Minified and gzipped

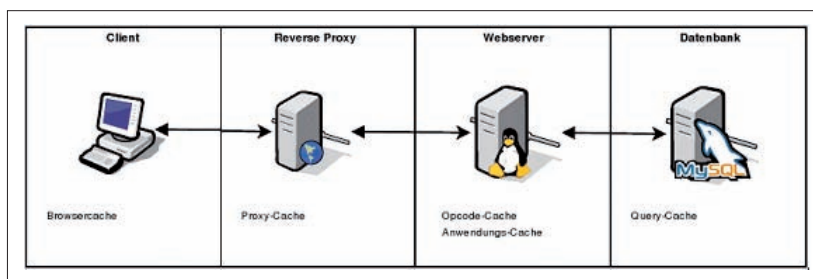
gung. Bild 6 zeigt eine Auswahl der bekanntesten Cache-Arten. Der Einsatz dieser Caches kann sowohl die Ladezeit der Website verkürzen als auch die Anfragen an den Webserver reduzieren. Grundsätzlich gilt: Je näher sich der Cache beim Benutzer befindet, desto größer ist seine Auswirkung. Der Browser-Cache eignet sich besonders für statische Dateien, die sich selten ändern. Das können beispielsweise Bilder, JavaScript- oder Stylesheet-Dateien sein.



Icons in Sprite zusammengefasst (Bild 5).

Einer der bekanntesten Drupal-Caches ist sicherlich der Page-Cache. In ihm speichert Drupal das gesamte HTML für eine Website ab. Nur in wenigen Fällen kann es sinnvoll sein, diese Option nicht zu nutzen. Daher kann grundsätzlich empfohlen werden, den Drupal-Page-Cache zu aktivieren. Auch die nachfolgend vorgestellte

Der Browser-Cache und der Reverse-Caching-Proxy sind auf korrekt gesetzte HTTP-Header (beispielsweise Expires oder Cache-Control) im Response des Webservers angewiesen. Nur so kann deren korrekte Funktionsweise sichergestellt werden. Der Original-Drupal-Core ist in der aktuellen Version 6 leider noch nicht in der Lage, alle Caching-Möglichkeiten zu nutzen. Es werden beispielsweise keine sinnvollen HTTP-Expire-Header gesetzt, so dass die Nutzung des Browser-Caches oder eines Reverse-Caching-Proxys nicht möglich ist. Pressflow oder der Drupal-Core von der Cocomore AG schaffen hier Abhilfe und setzen die HTTP-Header korrekt um. So wird der Einsatz eines Proxy-Servers erst möglich. Da beide kompatibel zum Original-Drupal-Core sind, kann dieser vollständig durch eine der beiden Varianten ersetzt werden.



Verschiedene Cache-Arten (Bild 6).

te Optimierung ist nur bei aktiviertem Page-Cache möglich.

Der Reverse-Caching-Proxy wird vor einem Server-System eingerichtet. Über ihn wird der gesamte Datenstrom zu und von den Servern geleitet. So ist es möglich, dass der Proxy häufig angeforderte Seiten in seinem Zwischenspeicher ablegt und direkt aus diesem ausliefern kann, ohne den Webserver nochmals mit der gleichen Anfrage zu belasten. Ein Vertreter, der in letzter Zeit immer mehr an Bedeutung gewonnen hat, ist Varnish. Varnish ist ein Proxy-Server, der rein

Folgendes Beispiel soll den Ablauf verdeutlichen und stellt ein grundlegendes Problem beim Einsatz externer Caches dar: Stellen wir uns vor, wir haben auf unserer Website einen neuen Beitrag erstellt und freigegeben. Wird der Beitrag nun vom ersten Besucher abgerufen, erhält Varnish diese Anfrage. Da Varnish bisher keinerlei Cache-Eintrag zu dieser Anfrage finden kann, wird der Request an den Webserver weitergeleitet. Drupal generiert das HTML für die Antwort, setzt die richtigen HTTP-Header mit einer Gültigkeitsdauer von 24 Stunden und gibt die gesamte Antwort zurück an Varnish. Dort werden die HTTP-Header ausgewertet und entschieden, dass diese Antwort für 24 Stunden zwischengespeichert werden darf. Im Anschluss leitet Varnish die Antwort an den Besucher der Website weiter. Für jeden folgenden Besucher des neuen Beitrags kann Varnish die passende Antwort nun direkt aus dem Cache ausliefern.

Nun fällt uns nach drei Stunden allerdings auf, dass sich ein wirklich schlimmer Fehler im neuen Beitrag eingeschlichen hat. Es fällt uns

```
HTTP/1.1 200 OK
Date: Fri, 30 Oct 1998 13:19:41 GMT
Server: Apache/1.3.3 (Unix)
Cache-Control: max-age=3600, must-revalidate
Expires: Fri, 30 Oct 1998 14:19:41 GMT
Last-Modified: Mon, 29 Jun 1998 02:28:12 GMT
ETag: "3e86-410-3596fbbc"
Content-Length: 1040
Content-Type: text/html
```

Der Cache-Control-Header (Bild 7).

nicht schwer, den Fehler im Text zu beheben und den Beitrag zu speichern. Anschließend kann Drupal die korrigierte Fassung ausliefern. Allerdings werden aktuell alle Anfragen bezüglich des neuen Beitrags direkt von Varnish beantwortet, der nur die alte und fehlerhafte Version des Beitrags kennt.

Grundsätzlich ist es schwierig, die Daten in einem externen Cache wie Varnish vor der Ablaufzeit zu aktualisieren. Dafür ist ein Rückkanal zum Reverse-Caching-Proxy nötig, der den Cache-Eintrag explizit invalidiert und dem System damit mitteilt, dass es seinen vorhandenen Cache-Eintrag löschen und eine aktuelle Version anfordern muss. Drupal bietet mit dem Modul *varnish* glücklicherweise bereits eine Lösung an. Nach dem Speichern einer geänderten Fassung des Beitrags sorgt es automatisch dafür, dass Varnish informiert und der vorhandene Cache-Eintrag gelöscht wird. Fordert nun ein Besucher unseren neuen Beitrag an, so speichert Varnish die aktualisierte Version im Cache und kann diese anschließend wieder an alle Besucher ausliefern.

Fazit

»Reduzieren auf das Wesentliche«, heißt also die Devise. Drupal bietet zahlreiche Möglichkeiten, die Performance der eigenen Website ohne Programmierkenntnisse zu steigern. Mit den Stan-

dard-Einstellungen, wie der CSS und Javascript-Aggregation sowie den Drupal-Caches kann die Auslastung des Webservers reduziert werden. Darüber hinaus gibt es für Drupal noch weitere Möglichkeiten, die Last zu reduzieren. Wird dann noch ein Reverse-Caching-Proxy vor den Webserver geschaltet, so können die Auslieferungzeiten von häufig aufgerufenen Seiten deutlich vermindert werden. Je schneller die Seite übermittelt ist, desto positiver wirkt sich das auf das Ranking bei Google aus.

Natürlich konnte im Rahmen dieses Artikels nur ein kleiner Teil aller Möglichkeiten beleuchtet werden. Die gezeigten Optimierungen wirken sich allerdings massiv auf die Ladezeiten der Website aus. **[mb]**

LINKS ZUM THEMA

Google-Ankündigung vom April 2010

▶ googlewebmastercentral.blogspot.com/2010/04/using-site-speed-in-web-search-ranking.html

Google Webmaster-Tools

▶ www.google.com/webmasters/tools

Drupal-Modul Javascript-Aggregator

▶ drupal.org/project/javascript_aggregator

Proxy-Server Varnish

▶ www.varnish-cache.org

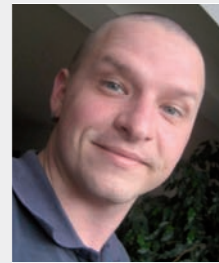
Drupal-Core von Cocomore

▶ drupal.cocomore.com

Drupal-Modul varnish

▶ drupal.org/project/varnish

AUTOR



Ronny Unger verfügt über langjährige Erfahrung mit dem Content-Management-System Drupal. Besonders intensiv beschäftigt er sich dabei mit den Optimierungspotenzialen von Drupal. Aktuell ist er bei der Cocomore AG in Frankfurt beschäftigt und verantwortet dort als Head of IT unter anderem die Umsetzung zahlreicher Drupal-Projekte.

▶ www.cocomore.com

COCOMORE

Die Cocomore AG ist eine gut 90 Mitarbeiter starke Multimedia-Agentur mit Büros in Frankfurt/Main und Königswinter (Köln/Bonn).

Mit den Standbeinen Kommunikation und Technologie entwickelt und betreut Cocomore Kommunikations- und Customer-Relationship-Management-Lösungen. Zu den Kunden zählen die Metro-Gruppe, Nestlé, Procter & Gamble, RTL, SCA und Sanofi-Aventis. Langjährige Kundenbeziehungen beruhen auf nachweisbaren Erfolgen (zum Beispiel Effie-Finalist 2008 und 2009). Cocomore hat Projekte in über 30 Ländern realisiert und ist Mitglied im Bundesverband Digitale Wirtschaft.



Cocomore realisiert mit einem Kernteam von 15 festangestellten Drupal-Entwicklern seit über vier Jahren komplexe Websites und große Portale mit Drupal. Neben einer eigenen Drupal-Distribution inklusive angepasstem Core stellt Cocomore auch Maintainer für Module der Drupal-Community zur Verfügung. Darüber hinaus ist Cocomore Mitglied der Drupal Association und Partner von Acquia, dem Anbieter professioneller Drupal-Hosting- und -Support-Leistungen des Drupal-Erfinders Dries Buytaert.

Unter www.cocomore.de und drupal.cocomore.com/de erfahren Sie mehr über Cocomore und die Produkte und Leistungen rund um Drupal.