Kostenloses eBook

LERNEN nginx

Free unaffiliated eBook created from
Stack Overflow contributors.

#nginx

Inhaltsverzeichnis

Über

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: nginx

It is an unofficial and free nginx ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official nginx.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Kapitel 1: Erste Schritte mit Nginx

Bemerkungen

NGINX wird wie "Engine x" ausgesprochen und wird üblicherweise als Hochleistungsserver für die Protokolle HTTP, HTTPS, SMTP, POP3 und IMAP verwendet. Es kann als Reverse-Proxy-Server, HTTP-Cache oder Lastverteilung verwendet werden.

Es ist ein Open Source-Projekt, dessen Quellcode hier verfügbar ist .

Versionen

Ausführung	Ursprüngliches Veröffentlichungsdatum	Letzte Version	Status	Veröffentlichungsdatum
0,5	2006-12-04	0,5,38	Erbe	2009-09-14
0,6	2007-06-14	0,6,39	Erbe	2009-09-14
0,7	2008-05-19	0,7,69	Erbe	2011-07-19
0,8	2009-06-02	0,8,55	Erbe	2011-07-19
1,0	2011-04-12	1.0.15	Erbe	2012-04-12
1.2	2012-04-23	1.2.9	Erbe	2013-05-13
1.4	2013-04-24	1.4.7	Erbe	2014-03-18
1.6	2014-04-24	1.6.3	Erbe	2015-04-07
1.8	2015-04-21	1.8.1	Erbe	2016-01-26
1,9	2015-04-28	1.9.15	Erbe	2016-04-19
1.10	2016-04-26	1.10.3	Stabil	2016-05-31
1.11	2016-05-24	1.11.9	Hauptleitung	2016-07-26

Examples

Installation und Einrichtung

Nginx ist ein Webserver, der zur Übermittlung von HTTP-Anforderungen über das Internet verwendet wird.

Nginx ist unter Linux, Windows und anderen Betriebssystemen als direkter Download verfügbar und kann auch über die Quelle erstellt werden. Detaillierte Anweisungen finden Sie in der offiziellen Nginx-Referenz.

Ubuntu / Debian

Die stabile Version von nginx ist in offiziellem Repo erhältlich und kann mit installiert werden

```
sudo apt-get install nginx
```

Es werden Systemstartdateien installiert und konfiguriert. Wenn Sie jedoch die neueste Version benötigen, müssen Sie möglicherweise das offizielle PPA hinzufügen.

```
sudo add-apt-repository ppa:nginx/stable
sudo apt-get update
sudo apt-get install nginx
```

Die obigen Anweisungen installieren die neueste stabile Edition.

Nginx-Installation unter Debian und Debian-basierten Distributionen wie Ubuntu

Führen Sie den folgenden Befehl aus, um nginx zu installieren.

```
sudo apt-get install nginx
```

Standardmäßig wird Nginx nach der Installation automatisch gestartet. Sie können auf die Standard-Landingpage von Nginx zugreifen, um zu überprüfen, ob die Software ordnungsgemäß ausgeführt wird, indem Sie den Domänennamen oder die öffentliche IP-Adresse Ihres Servers in Ihrem Webbrowser aufrufen.

Wenn Sie jedoch die neueste Version benötigen, müssen Sie möglicherweise das offizielle PPA hinzufügen.

```
sudo add-apt-repository ppa:nginx/stable
sudo apt-get update
sudo apt-get install nginx
```

Starten Sie NGINX neu

Als root-Benutzer:

```
nginx -s restart
```

Ubuntu-Beispiel

```
sudo service nginx restart
```

Laden Sie die NGINX-Konfigurationsdatei erneut

Als root-Benutzer:

```
sudo nginx –s reload
```

Ubuntu 14.04 Beispiel

```
sudo service nginx reload
```

Ubuntu 16.04 Beispiel

```
sudo systemctl reload nginx
```

Vor dem erneuten Laden sollten Sie die config auf Syntaxfehler überprüfen:

```
sudo nginx –t
```

Oder

```
sudo service nginx configtest
```

NGINX herunterfahren

Als root-Benutzer ausführen.

Schnelles Herunterfahren:

```
nginx –s stop
```

Anmutiges Herunterfahren:

```
nginx –s quit
```

Nginx im Inneren

Eine der Hauptattraktionen von Nginx ist der Unterschied in der internen Funktionsweise im Vergleich zu den anderen populären Servern, insbesondere Apache.

Server sind ausgelastete Programme, da sie Anforderungen von mehreren Clients bedienen müssen. Je mehr Anforderungen ein Server pro Sekunde erfolgreich verarbeiten kann, desto besser.

Nginx arbeitet mit einem Parallelitätsparadigma, das als asynchrones IO bezeichnet wird.

In einem herkömmlichen Server ist ein Thread für eine Anforderung bestimmt. Das heißt, sobald ein Thread eine Anfrage aufnimmt, ist er für andere Anfragen effektiv nicht verfügbar. In der Realität könnte ein Thread jedoch viel besser sein, wenn er mehrere Anfragen akzeptiert und gleichzeitig bedient. Asynchrone IO ermöglicht dies.

Nginx kann daher mit seiner asynchronen E / A-Architektur viele Anforderungen innerhalb eines Threads bedienen.

Eine weitere gute Sache bei Nginx ist der relativ schlanke Ressourcenabdruck. Im Vergleich zu Apache ist Nginx weniger ressourcenintensiv und eignet sich daher für Cloud-Server, die nicht besonders leistungsfähig sind.

Es gibt sicherlich andere Async-IO-Server, aber Nginx wird von allen unter Pluginx (auch als Nginx-Modul bezeichnet) am besten unterstützt.

Testen Sie, ob Ihre Änderungen in nginx.config gültig sind

Ubuntu 14.04 Beispiel

```
sudo nginx -t
```

Erste Schritte mit Nginx online lesen: https://riptutorial.com/de/nginx/topic/1121/erste-schritte-mit-nginx

Kapitel 2: Nginx Reverse Proxy

Examples

einfacher Reverse Proxy

```
# Define which servers to include in the load balancing scheme.
# It's best to use the servers' private IPs for better performance and security.

upstream backend {

    ip_hash;
    server 10.10.10.10 slow_start=30s max_fails=3 fail_timeout=15s;
    server 10.10.10.12 slow_start=30s max_fails=3 fail_timeout=15s;

    # Activates the cache for connections to upstream servers.
    keepalive 20;
}

# This server accepts all traffic to port 80 and passes it to the upstream.
# Notice that the upstream name and the proxy_pass need to match.

server {
    listen 80;
    server_name example.com;

    location / {
        proxy_pass http://backend/;
    }
}
```

Nginx Reverse Proxy online lesen: https://riptutorial.com/de/nginx/topic/7431/nginx-reverse-proxy

Kapitel 3: Nginx-Konfigurationen

Examples

Struktur der Konfigurationsdatei

nginx besteht aus Modulen, die von in der Konfigurationsdatei angegebenen Anweisungen gesteuert werden.

Einfache Richtlinien

Eine einfache Direktive besteht aus Namen und Parametern, die durch Leerzeichen getrennt sind, und endet mit einem Semikolon (;).

Blockrichtlinie

Eine Blockdirektive hat dieselbe Struktur wie eine einfache Direktive, endet jedoch anstelle des Semikolons mit einem Satz zusätzlicher Anweisungen, die von geschweiften Klammern ({und}) umgeben sind.

Kontext

Wenn eine Blockanweisung andere Anweisungen in geschweiften Klammern enthalten kann, wird sie als Kontext bezeichnet (Beispiele: Ereignisse, http, Server und Ort).

Direktiven, die sich außerhalb eines Kontexts in der Konfigurationsdatei befinden, gelten als im Hauptkontext. Die Ereignisse und die http-Anweisungen befinden sich im Hauptkontext, Server in http und Ort im Server.

Kommentar

Der Rest einer Zeile nach dem # -Zeichen wird als Kommentar betrachtet.

Testen Sie die NGINX-Konfigurationsdatei

Sie können in einer NGINX-Konfigurationsdatei nach Syntaxfehlern und referenzierten Dateien suchen, bevor Sie diese ausführen:

```
nginx -t
```

Alternativ können Sie ein Service-Skript ausführen

```
service nginx configtest
```

Während dieser beiden Befehle erfahren Sie, ob Ihre neue Nginx-Konfiguration in Ordnung ist (ohne Ihre aktuelle Instanz zu beenden). Configtest verwendet den laufenden Dienst und teilt Ihnen mit, ob die Prüfung erfolgreich ist oder nicht, während nginx -t nicht nur die Konfiguration überprüft, sondern auch alle Informationen, Warnungen und Fehlermeldungen ausgibt.

Quelle: http://devget.net/nginxapache/nginx-configtest-vs-nginx-t/

Geben Sie die zu ladende Konfigurationsdatei an

```
nginx -c <file name>
```

Starten Sie NGINX mit einer expliziten Konfigurationsdatei.

config ändert sich ohne Neustart

```
nginx -s reload
```

Protokollierung in Syslog, inkl. Zuordnung von HTTP-Codes zu Syslog-Schweregraden.

Fügen Sie dieses Snippet irgendwo in den `http {}` -Block ein. oder legen Sie es in eine eigene Datei im Ordner `/etc/nginx/conf.d/` . Beachten Sie auch die offiziellen Dokumente zum Anmelden bei Syslog.

```
#
# Access Log
#
log_format fmt_syslog '[$time_local] $status $remote_addr $http_host "$request"
$body_bytes_sent $request_time "$http_user_agent" $remote_user';
map $status $log_is_error { "~^5\d\d"     1; default 0; }
map $status $log_is_warn  { "~^4[0-8]{2}" 1; default 0; }
map $status $log_is_info  { "~^[1-3]\d\d" 1; default 0; }
access_log
syslog:server=unix:/run/systemd/journal/syslog,nohostname,facility=local2,severity=error
fmt_syslog if=$log_is_error;
access_log
syslog:server=unix:/run/systemd/journal/syslog,nohostname,facility=local2,severity=warn
fmt_syslog if=$log_is_warn;
access_log
syslog:server=unix:/run/systemd/journal/syslog,nohostname,facility=local2,severity=info
fmt_syslog if=$log_is_info;
#
# Error Log
#
error_log syslog:server=unix:/run/systemd/journal/syslog,nohostname,facility=local2 error;
```

In diesem Beispiel wird davon ausgegangen, dass Rsyslog (oder ähnliches) Socket `/run/systemd/journal/syslog` überwacht . Die Standardeinstellung für Debian 8 ist, wenn journal

ForwardToSyslog aktiviert hat . Mit diesem Sockel umgehen Sie Journald. Wenn dieser Socket nicht verfügbar ist, versuchen Sie stattdessen `/dev/log` .

Fühlen Sie sich frei, eine andere Einrichtung anstelle von local2 zu verwenden. Sie können das log_format auch an Ihre Bedürfnisse anpassen .

Begrenzen Sie Anforderungsmethoden

Eine gewöhnliche Website benötigt nur 3 HTTP-Methoden: `GET` , `HEAD` und `POST` . Blockieren Sie alle anderen Methoden mithilfe von limit_except :

```
location / {
    [...]
    # Note: GET includes HEAD
    limit_except GET POST {
        deny  all;
    }
    [...]
}
```

Nginx-Konfigurationen online lesen: https://riptutorial.com/de/nginx/topic/2550/nginx-konfigurationen

Kapitel 4: Nützliche Weiterleitungen

Examples

HTTPS-Weiterleitung

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;
    server_name example.com www.example.com;
    return 307 https://$host$request_uri;
}
```

Eine 301-Weiterleitung ist ebenfalls eine Alternative. Wenn jedoch ein POST an eine 301 gesendet wird, senden viele Clients die Anforderung erneut als GET - was fast immer serverseitig ausfällt. Ein 307 verlangt den gleichen Anforderungstyp.

Anfragen an einen anderen Server umleiten

```
server {
   server_name example.com;
   return 301 $scheme://example.net$request_uri;
}
```

Mobile Site-Umleitung

Nginx-Konfiguration zum Erkennen von Anforderungen vom mobilen Benutzeragenten und zur Weiterleitung an die mobile Site.

```
location / {

    #mobile site handling as per user agent
    set $mobile_rewrite do_not_perform;     // variable to store action. default set to not
perform redirection to mobile site.

    if ($http_user_agent ~*
"(android|bb\d+|meego).+mobile|avantgo|bada\/|blackberry|blazer|compal|elaine|fennec|hiptop|iemobile|ip
|maemo|midp|mmp|mobile.+firefox|netfront|opera m(ob|in)i|palm(
os)?|phone|p(ixi|re)\/|plucker|pocket|psp|series(4|6)0|symbian|treo|up\.(browser|link)|vodafone|wap|win
ce|xda|xiino") {
        set $mobile_rewrite perform;
    }

    if ($http_user_agent ~* "^(1207|6310|6590|3gso|4thp|50[1-6]i|770s|802s|a
wa|abac|ac(er|oo|s\-
)|ai(ko|rn)|al(av|ca|co)|amoi|an(ex|ny|yw)|aptu|ar(ch|go)|as(te|us)|attw|au(di|\-m|r |s
)|avan|be(ck|ll|nq)|bi(lb|rd)|bl(ac|az)|br(e|v)w|bumb|bw\-(n|u)|c55\/|capi|ccwa|cdm\-
|cell|chtm|cldc|cmd\-|co(mp|nd)|craw|da(it|ll|ng)|dbte|dc\-s|devi|dica|dmob|do(c|p)o|ds(12|\-
d)|el(49|ai)|em(l2|ul)|er(ic|k0)|esl8|ez([4-7]0|os|wa|ze)|fetc|fly(\-|_)|g1 u|g560|gene|gf\-
5|g\-mo|go(\.w|od)|gr(ad|un)|haie|hcit|hd\-(m|p|t)|hei\-|hi(pt|ta)|hp( i|ip)|hs\-c|ht(c(\-|
|_|a|g|p|s|t)|tp)|hu(aw|tc)|i\-(20|go|ma)|i230|iac( |\-
```

```
|\/)|ibro|idea|ig01|ikom|im1k|inno|ipaq|iris|ja(t|v)a|jbro|jemu|jigs|kddi|keji|kgt(
|\/)|klon|kpt |kwc\-|kyo(c|k)|le(no|xi)|lg( g|\/(k|l|u)|50|54|\-[a-w])|libw|lynx|m1\-
w|m3ga|m50\/|ma(te|ui|xo)|mc(01|21|ca)|m\-
cr|me(rc|ri)|mi(o8|oa|ts)|mmef|mo(01|02|bi|de|do|t(\-| |o|v)|zz)|mt(50|p1|v )|mwbp|mywa|n10[0-
2]|n20[2-3]|n30(0|2)|n50(0|2|5)|n7(0(0|1)|10)|ne((c|m)\-
|on|tf|wf|wg|wt)|nok(6|i)|nzph|o2im|op(ti|wv)|oran|owg1|p800|pan(a|d|t)|pdxg|pg(13|\-([1-
8]|c))|phil|pire|pl(ay|uc)|pn\-2|po(ck|rt|se)|prox|psio|pt\-g|qa\-a|qc(07|12|21|32|60|\-[2-
7]|i\-)|qtek|r380|r600|raks|rim9|ro(ve|zo)|s55\/|sa(ge|ma|mm|ms|ny|va)|sc(01|h\-|oo|p\-
)|sdk\/|se(c(\-|0|1)|47|mc|nd|ri)|sgh\-|shar|sie(\-|m)|sk\-
0|sl(45|id)|sm(al|ar|b3|it|t5)|so(ft|ny)|sp(01|h\-|v\-|v
)|sy(01|mb)|t2(18|50)|t6(00|10|18)|ta(gt|lk)|tcl\-|tdg\-|tel(i|m)|tim\-|t\-
mo|to(pl|sh)|ts(70|m\-|m3|m5)|tx\-9|up(\.b|g1|si)|utst|v400|v750|veri|vi(rg|te)|vk(40|5[0-
3]|\-v)|vm40|voda|vulc|vx(52|53|60|61|70|80|81|83|85|98)|w3c(\-| )|webc|whit|wi(g
|nc|nw)|wmlb|wonu|x700|yas\-|your|zeto|zte\-)") {
        set $mobile_rewrite perform;
    }

    #google bot mobile handling
    if ($http_user_agent ~* "(googlebot-mobile)") {
        set $mobile_rewrite perform;
    }

    if ($mobile_rewrite = perform) {
        proxy_pass http://www.mobile-domain.com:$port;
    }
}
```

HTTP-Speicherort auf dem HTTPS-Server

HTTPS-Server mit http-Speicherort:

```
server {
    listen 443;
    root /var/www/
    location / {
       ...
    }
    location /http {
        rewrite ^ http://$host$request_uri? permanent;
    }
}
```

HTTP-Server leitet HTTPS um, außer an einem Ort

```
server {
    root /var/www/
    location / {
        rewrite ^ https://$host$request_uri? permanent;
    }
    location /http {
       ...
    }
}
```

Ablehnen von Anfragen basierend auf dem Host- oder Ländercode

Verwenden Sie die Nginx-Map , um Felder zu analysieren und Anforderungen abzulehnen.

```
# Allowed hosts
map $http_host $name {
    hostnames;

    default       no;

    example.com   yes;
    *.example.com yes;
    example.org   yes;
    *.example.org yes;
    .example.net  yes;
    wap.*         yes;
}

# Allowed countries
map $geoip_country_code $allowed_country {
    default no;
    country_code_1 yes;
    country_code_2 yes;
}
```

```
server {
    listen 80 default_server;
    listen [::]:80 default_server;

    # Disallow access based on hostname
    if ($name = no) {
        return 444;
    }

    # Disallow access based on GeoIP
    if ($allowed_country = no) {
        return 444;
    }
...
```

Nützliche Weiterleitungen online lesen: https://riptutorial.com/de/nginx/topic/3631/nutzliche-weiterleitungen

Kapitel 5: Protokollierung

Examples

Grundlegendes Beispiel

Syntax

```
Syntax:  log_format name string ...;
Syntax:  access_log path [format [buffer=size] [gzip[=level]] [flush=time] [if=condition]];
access_log off;
```

Sie zusammen verwenden

```
log_format compression '$remote_addr - $remote_user [$time_local] '
                       '"$request" $status $bytes_sent '
                       '"$http_referer" "$http_user_agent" "$gzip_ratio"';

access_log /spool/logs/nginx-access.log compression buffer=32k;
error_log /spool/logs/nginx-error.log;
```

Öffnen Sie die Protokolldateien erneut

Als root-Benutzer ausführen:

```
nginx -s reopen
```

Vermeiden Sie die Protokollierung für favicon.ico und robots.txt

```
location = /favicon.ico {
    log_not_found off;
    access_log off;
}

location = /robots.txt {
    allow all;
    log_not_found off;
    access_log off;
}
```

Protokollierung online lesen: https://riptutorial.com/de/nginx/topic/2551/protokollierung

Kapitel 6: Verwenden von nginx zur Bereitstellung sauberer Browser-URLs

Examples

Umleitung vs Reverse Proxy

Professionell erstellte Webanwendungen legen dem Benutzer die internen Details der Serverumgebung nicht offen. Wenn Sie bei Ihrem Händler eine Bestellung aufgeben, wird nicht https://mydealer.com:8443/Dealerapp/entryPage.html angezeigt (oder muss eingegeben werden), sondern nur mydealer.com, obwohl der App-Server alle Details benötigt in der längeren URL angegeben. Dies kann erreicht werden durch:

```
if ($scheme = http) {
    return 301 https://$server_name$request_uri;
}
```

Dies führt zu einer **Weiterleitung** . Auch wenn der Client keine sichere Verbindung angefordert hat, wird der Browser sofort darauf umgeleitet. In Ländern, in denen strikte Datenschutzgesetze gelten, sind Sie möglicherweise sogar verpflichtet, dies für jede kommerzielle Website zu tun. Der Umleitungsweg wird gewählt, da der Browser die sichere Verbindung kennen muss. Andernfalls würde er nicht mit dem Server verhandeln, um die Sicherheit zu gewährleisten.

```
location /app/ {
    proxy_pass https://mydealer.com:8443/Dealerapp/entryPage.html;
}
```

Dies ist ein **Reverse Proxy** . Es ist für den Browser transparent. So können Sie vor dem Endbenutzer vollständig verbergen, ob Sie einen oder mehrere App-Server haben, an welchen Ports sie abhören oder wie ihre Anwendungen benannt sind. Andernfalls würden alle Lesezeichen Ihrer Clients ungültig, wenn Ihr Datencenter die App auf einen anderen Server verschieben muss, der 8543 abhört. In diesem Beispiel wird der Benutzer auch zu Ihrer Einstiegsseite geleitet. Dies kann weggelassen werden, wenn Sie die Einstiegsseite index.html nennen. Möglicherweise verfügen Sie jedoch über mehrere Einstiegsseiten in einer App, die der Benutzer mit einem Lesezeichen versehen möchte. Sie sind also flexibler, wenn Sie nicht an index.html gebunden sind.

Ich habe die Firmen-URL mit / app nachfixiert. Dies könnte entfallen, wenn Ihre Domain nur diese App bereitstellt. Falls neben der App auch statischer Inhalt vorhanden ist, z. B. Ihre Firmenbeschreibung, möchten Sie möglicherweise die einfache URL dafür.

Dieser Proxy funktioniert nur für die Einstiegsseite. Normalerweise benötige ich zwei weitere URL-Schemas, eines für statische Inhalte der Web-App, wie JavaScript, CSS und Bilder. Ich habe alle in einer Ordnerstruktur unter einem Ordner namens serverapp abgelegt und schreibe den folgenden Proxy:

```
location /app/serverapp/ {
    proxy_pass https://mydealer.com:8443/Dealerapp/serverapp/;
}
```

Und ein anderer Weg für REST-Dienste; Der Rest-URL-Pfad stimmt hier nicht mit einem Ordner überein, sondern mit dem Pfad eines jax-rs-REST-Services:

```
location /rest/ {
    proxy_pass https://mydealer.com:8443/Dealerapp/rest/;
}
```

Ein weiterer Schritt, selten irgendwo erwähnt, muss unternommen werden. Der App-Server wird unter dem URL-Pfad / Dealerapp ausgeführt, so dass ein Sitzungscookie mit dem Eigenschaftspfad = Dealerapp ausgegeben wird. Der Browser kennt diesen Pfad nicht und ignoriert das Cookie aufgrund seiner Same Origin-Richtlinie. Wir könnten es über Cross-Origin Resource Sharing davon überzeugen, dies zuzulassen, aber es ist wahrscheinlich einfacher, den Cookie-Pfad zu ändern, indem Sie den Pfad auf / setzen und etwas schreiben

```
<session-config>
    <session-timeout>720</session-timeout>
    <cookie-config>
        <name>SZSESSION</name>
        <path>/</path>
        <http-only>true</http-only>
        <secure>true</secure>
    </cookie-config>
</session-config>
```

in unserer web.xml.

Verwenden von nginx zur Bereitstellung sauberer Browser-URLs online lesen: https://riptutorial.com/de/nginx/topic/6270/verwenden-von-nginx-zur-bereitstellung-sauberer-browser-urls

Kapitel 7: Wordpress-Blog-Integration mit Rails-Anwendung mit Nginx

Examples

nginx Serverblock

Nachdem Sie die Einstellungen für php5-fpm und wordpress eingerichtet und konfiguriert haben, können Sie die Datei /etc/nginx/conf/nginx.conf wie folgt konfigurieren.

Sie müssen die Standortblöcke innerhalb des Serverblocks definieren und die URL dort wie definiert umschreiben.

```
server {
listen 443 ssl;
        server_name abc.co.uk;
    root /home/ubuntu/www/abc/current/public;
        try_files $uri/index.html $uri @unicorn;
    ssl on;
    ssl_certificate /etc/nginx/ssl/abc.crt;
    ssl_certificate_key /etc/nginx/ssl/abc.key;
    location /blog/wp-admin/ {
        root /var/www/html/;
        index index.php;
        try_files $uri $uri/ /index.php?$args;
        location ~* \.(js|css|xml|txt|jpg)$ {
             expires 14d;
         root /var/www/html/;
             access_log off;
           }

    location ~ \.php$ {
                try_files $uri $uri/ /index.php;
                fastcgi_pass unix:/var/run/php5-fpm.sock;
                fastcgi_param SCRIPT_FILENAME $request_filename;
                fastcgi_index index.php;
                include /etc/nginx/conf/fastcgi_params;
    }

}
location ^~ /blog {
    root /var/www/html/;
    index index.php;
    try_files $uri $uri/ /index.php?$args;
    rewrite ^/blog/(.*)+$ /blog/index.php?$1;
    location ~* \.(js|css|xml|txt|jpg)$ {
         expires 14d;
     root /var/www/html/;
         access_log off;
       }

    location ~ \.php$ {
                try_files $uri $uri/ /index.php;
                fastcgi_pass unix:/var/run/php5-fpm.sock;
```

```
                fastcgi_param SCRIPT_FILENAME $request_filename;
                fastcgi_index index.php;
                include /etc/nginx/conf/fastcgi_params;
    }

  }



}
```

Wordpress-Blog-Integration mit Rails-Anwendung mit Nginx online lesen: https://riptutorial.com/de/nginx/topic/3891/wordpress-blog-integration-mit-rails-anwendung-mit-nginx

Credits

S. No	Kapitel	Contributors
1	Erste Schritte mit Nginx	Bbak, Community, I Am Batman, James, Marek Skiba, Mark Stosberg, Neo, Przemysław Jagielski, rajarshig, RamenChef, RationalDev, theDrifter, treecoder, Xevaquor
2	Nginx Reverse Proxy	smart-developer
3	Nginx-Konfigurationen	Bbak, James, Pablo Fernandez, RationalDev
4	Nützliche Weiterleitungen	Aleksey Deryagin, Alexandre Maciel, Alexey Ten, Gaurav Kumar, Joshua DeWald, Justin W., Keelan, Muaaz Rafi, smart-developer, timbo
5	Protokollierung	Gustav, RationalDev, timbo
6	Verwenden von nginx zur Bereitstellung sauberer Browser-URLs	TAM
7	Wordpress-Blog-Integration mit Rails-Anwendung mit Nginx	Abid Iqbal