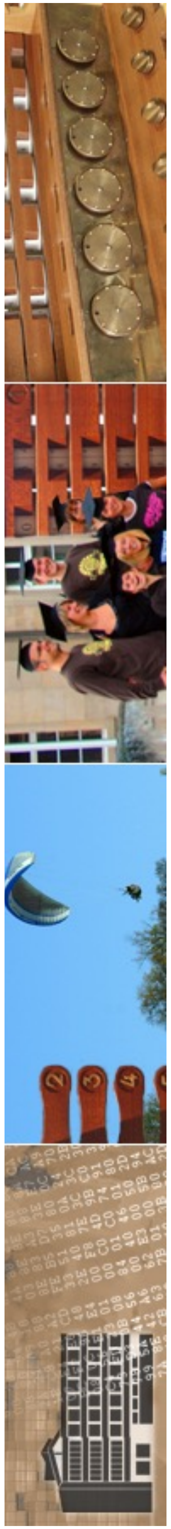


EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

MATHEMATISCH-
NATURWISSENSCHAFTLICHE FAKULTÄT
FACHBEREICH INFORMATIK
INFORMATIONSDIENSTE

Grundlagen der Web-Entwicklung

INF3172

Performante Webserver: NGINX und Caddy

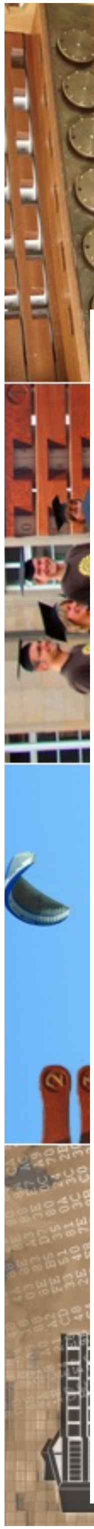
Thomas Walter

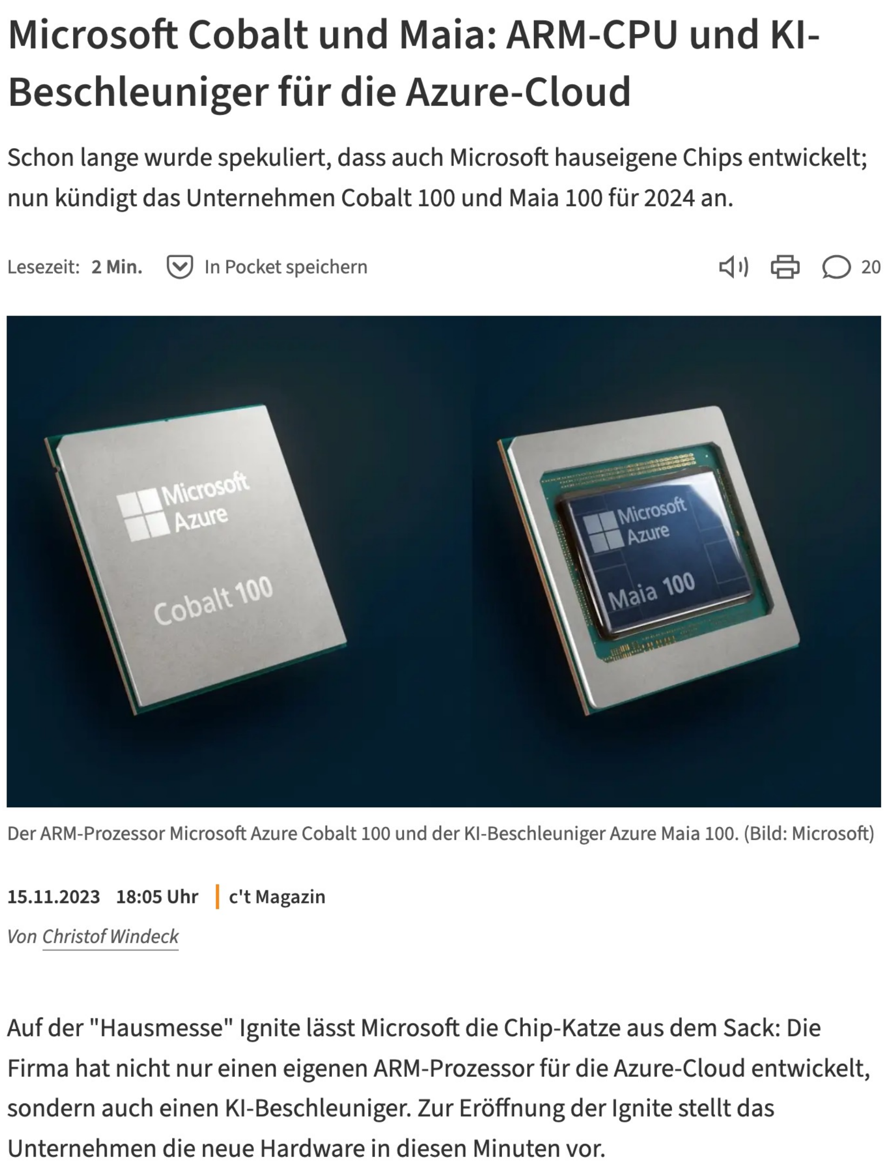
23.11.2023

Version 1.0

NGINX

aktuelles





Microsoft Cobalt und Maia: ARM-CPU und KI-Beschleuniger für die Azure-Cloud

Schon lange wurde spekuliert, dass auch Microsoft hauseigene Chips entwickelt; nun kündigt das Unternehmen Cobalt 100 und Maia 100 für 2024 an.

Lesezeit: **2 Min.** In Pocket speichern 20

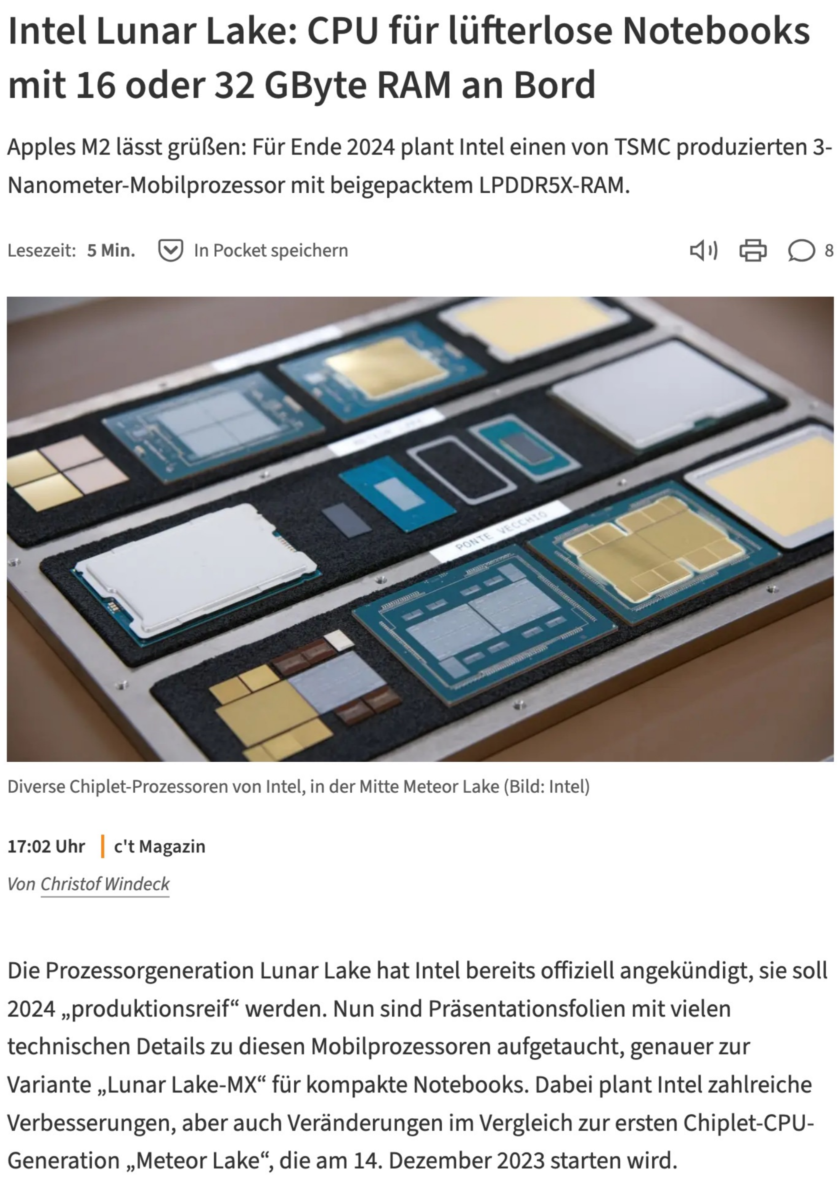
Der ARM-Prozessor Microsoft Azure Cobalt 100 und der KI-Beschleuniger Azure Maia 100. (Bild: Microsoft)

15.11.2023 18:05 Uhr | **c't Magazin**

Von Christof Windeck

Auf der "Hausmesse" Ignite lässt Microsoft die Chip-Katze aus dem Sack: Die Firma hat nicht nur einen eigenen ARM-Prozessor für die Azure-Cloud entwickelt, sondern auch einen KI-Beschleuniger. Zur Eröffnung der Ignite stellt das Unternehmen die neue Hardware in diesen Minuten vor.

llagen
rseme



Intel Lunar Lake: CPU für lüfterlose Notebooks mit 16 oder 32 GByte RAM an Bord

Apples M2 lässt grüßen: Für Ende 2024 plant Intel einen von TSMC produzierten 3-Nanometer-Mobilprozessor mit beigepacktem LPDDR5X-RAM.

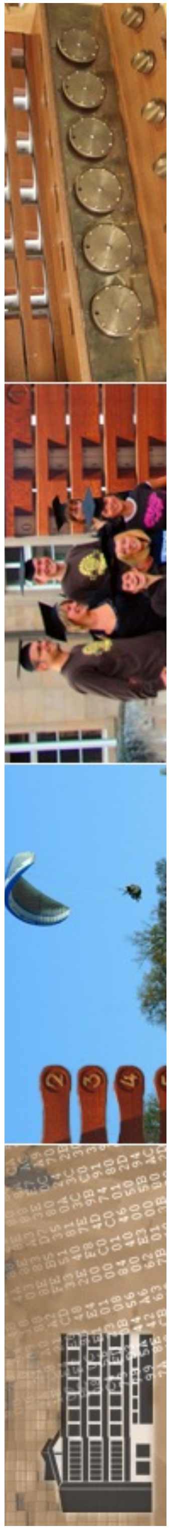
Lesezeit: **5 Min.** In Pocket speichern 8

Diverse Chiplet-Prozessoren von Intel, in der Mitte Meteor Lake (Bild: Intel)

17:02 Uhr | **c't Magazin**

Von Christof Windeck

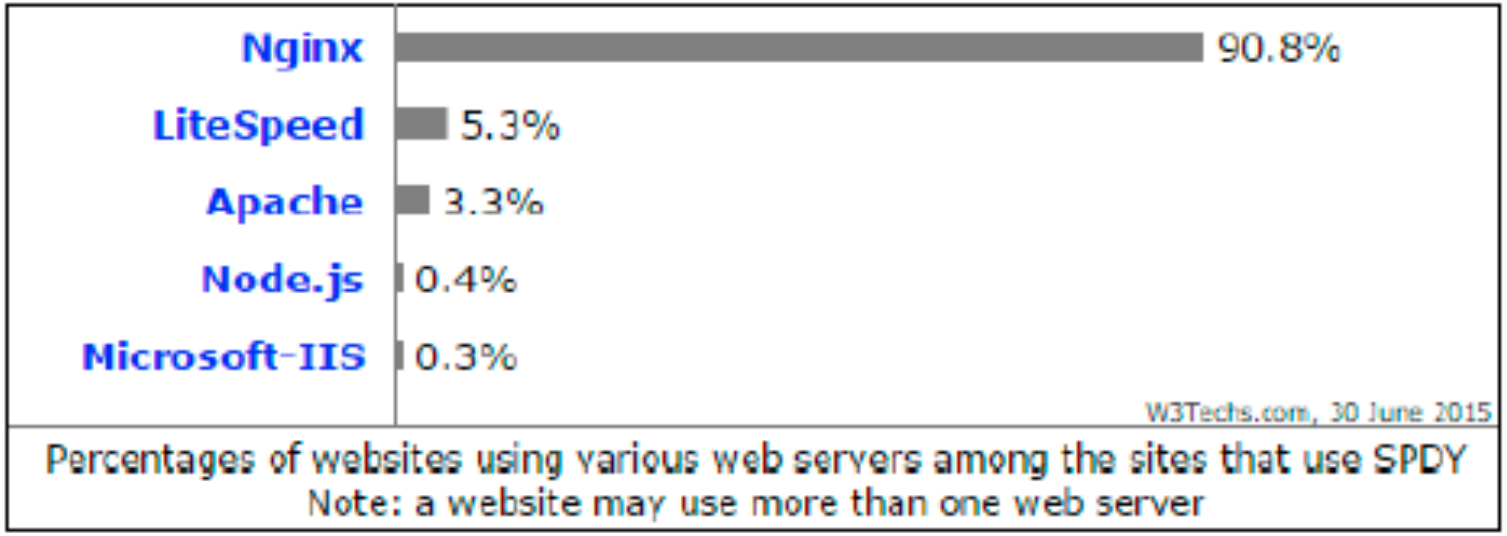
Die Prozessorgeneration Lunar Lake hat Intel bereits offiziell angekündigt, sie soll 2024 „produktionsreif" werden. Nun sind Präsentationsfolien mit vielen technischen Details zu diesen Mobilprozessoren aufgetaucht, genauer zur Variante „Lunar Lake-MX" für kompakte Notebooks. Dabei plant Intel zahlreiche Verbesserungen, aber auch Veränderungen im Vergleich zur ersten Chiplet-CPU-Generation „Meteor Lake", die am 14. Dezember 2023 starten wird.



NGINX

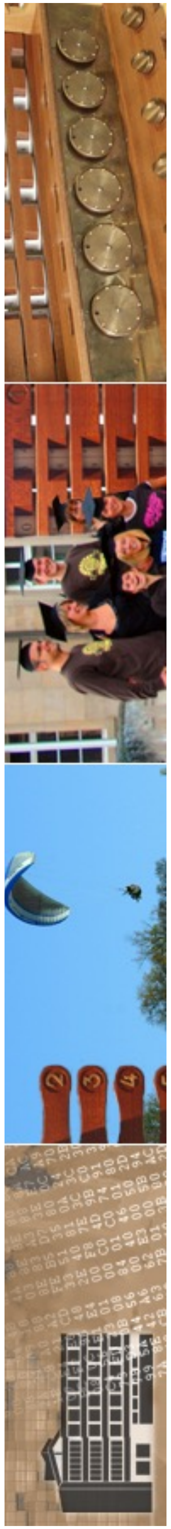
- Web Server & Reverse/Mail Proxy
- nginx.org (nginx.com existiert auch)

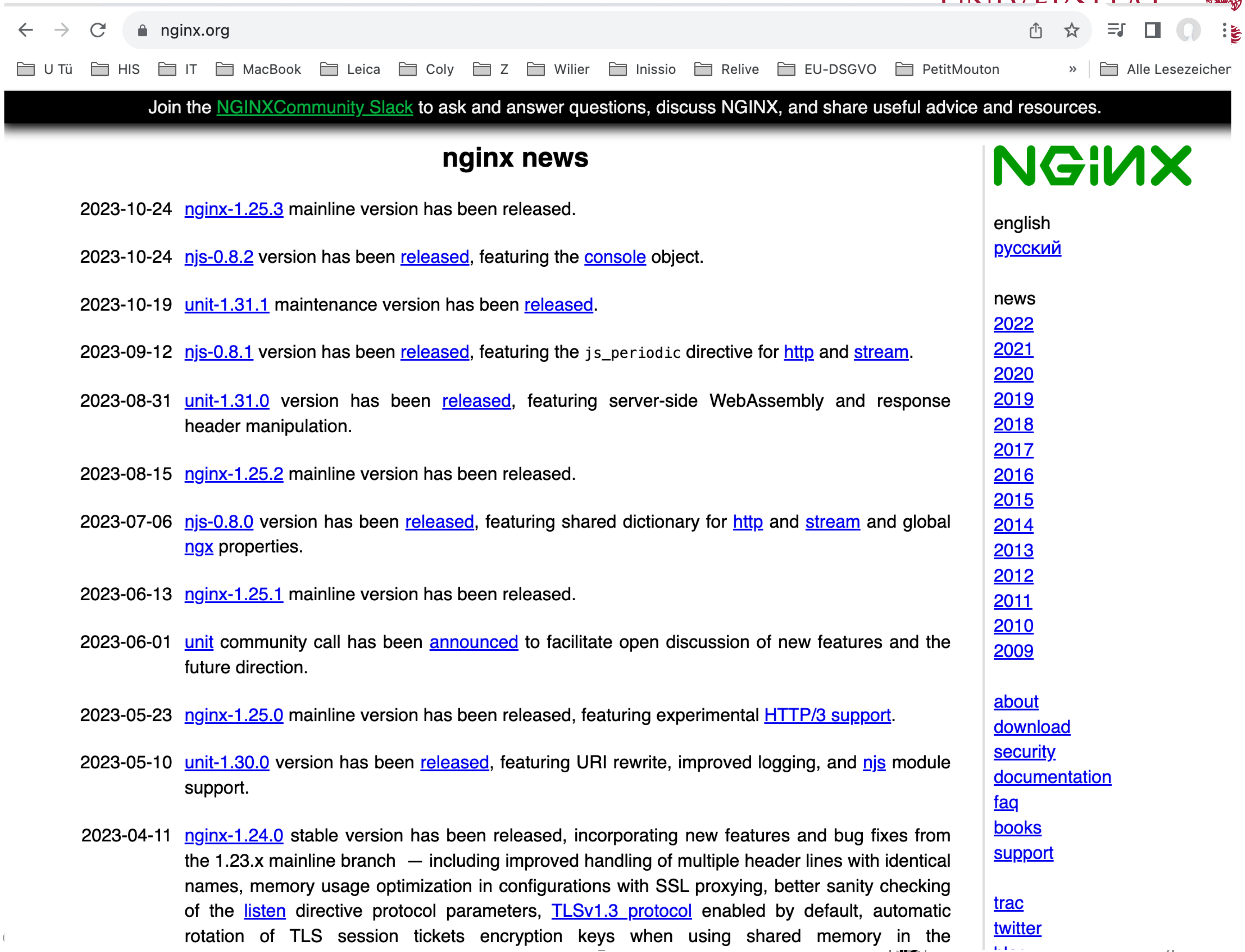
- Optimiert auf Performance



W3Techs.com, 30 June 2015

Percentages of websites using various web servers among the sites that use SPDY
Note: a website may use more than one web server





Join the NGINXCommunity Slack to ask and answer questions, discuss NGINX, and share useful advice and resources.

nginx news

2023-10-24 nginx-1.25.3 mainline version has been released.

2023-10-24 njs-0.8.2 version has been released, featuring the console object.

2023-10-19 unit-1.31.1 maintenance version has been released.

2023-09-12 njs-0.8.1 version has been released, featuring the `js_periodic` directive for http and stream.

2023-08-31 unit-1.31.0 version has been released, featuring server-side WebAssembly and response header manipulation.

2023-08-15 nginx-1.25.2 mainline version has been released.

2023-07-06 njs-0.8.0 version has been released, featuring shared dictionary for http and stream and global ngx properties.

2023-06-13 nginx-1.25.1 mainline version has been released.

2023-06-01 unit community call has been announced to facilitate open discussion of new features and the future direction.

2023-05-23 nginx-1.25.0 mainline version has been released, featuring experimental HTTP/3 support.

2023-05-10 unit-1.30.0 version has been released, featuring URI rewrite, improved logging, and njs module support.

2023-04-11 nginx-1.24.0 stable version has been released, incorporating new features and bug fixes from the 1.23.x mainline branch — including improved handling of multiple header lines with identical names, memory usage optimization in configurations with SSL proxying, better sanity checking of the listen directive protocol parameters, TLSv1.3 protocol enabled by default, automatic rotation of TLS session tickets encryption keys when using shared memory in the

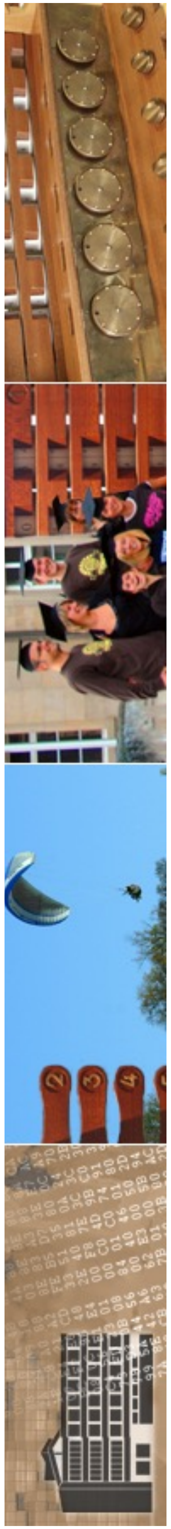
NGINX

english
русский

news
2022
2021
2020
2019
2018
2017
2016
2015
2014
2013
2012
2011
2010
2009

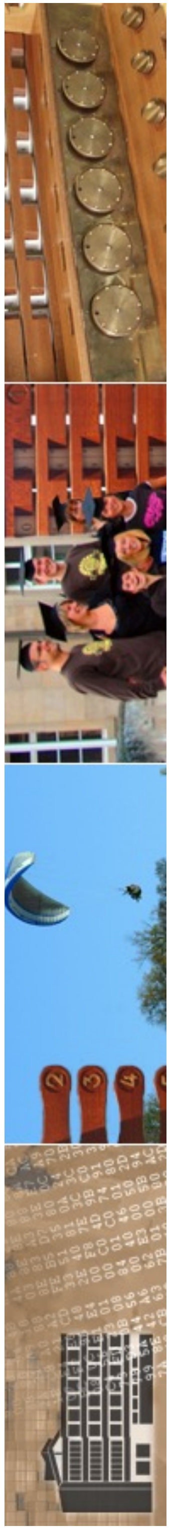
about
download
security
documentation
faq
books
support

trac
twitter



Eigenschaften

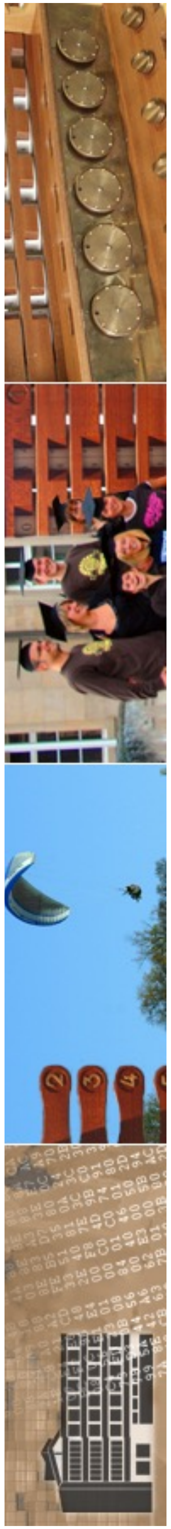
- Auslieferung von statischem Inhalt
- Reverse Proxy mit Caching
- Load Balancing
- TLS (SSL)
- FastCGI, CGI
- Streaming
- HTTP 1.1/2.0, SPDY
- Websockets
- Embedded Perl Scripting



Genese

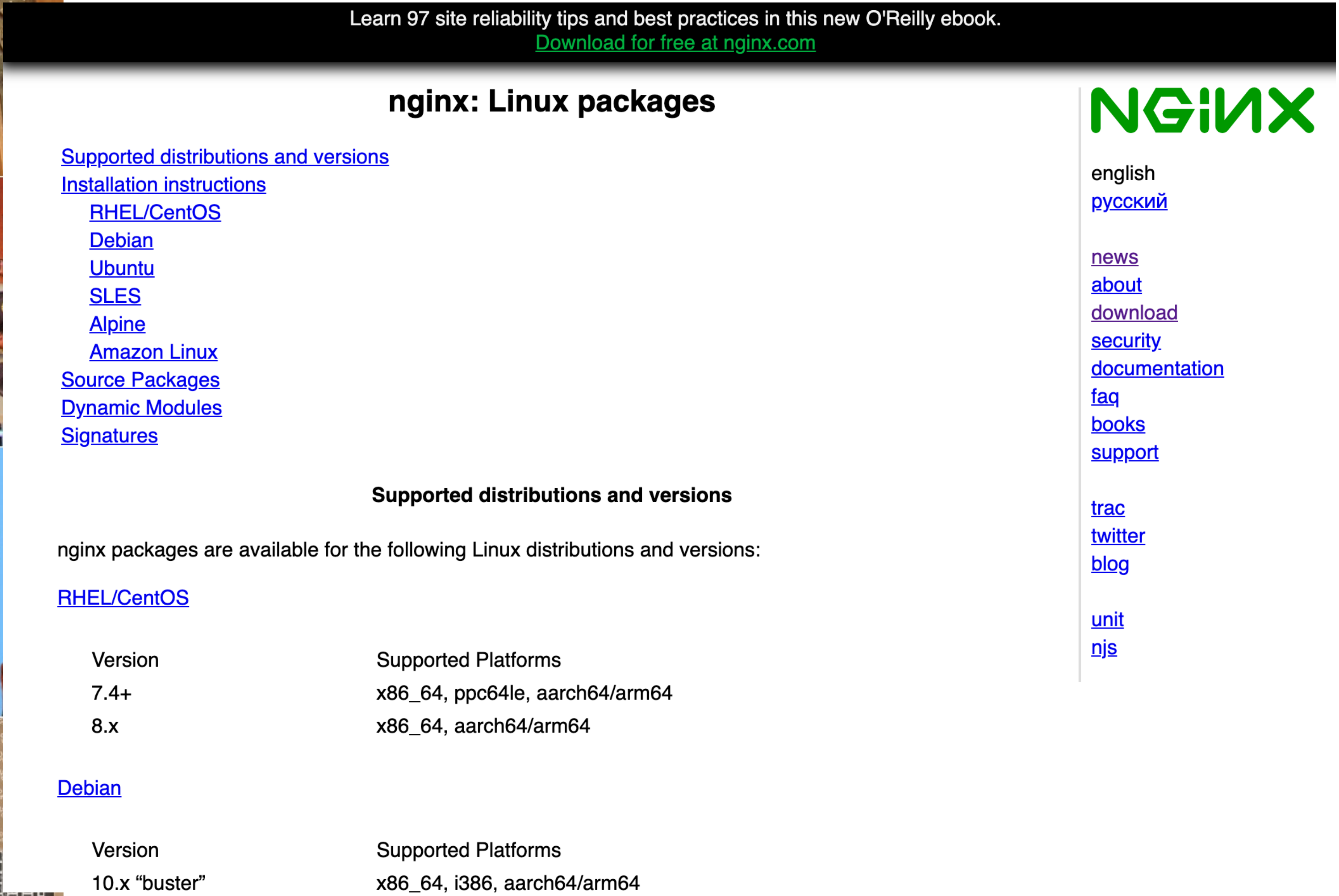
- 2002: Entwicklungsbeginn durch Igor Sysoev
- 2004: Veröffentlichung von Version 0.1.0
- 2011: Veröffentlichung von Version 1.0
- 2012 Gründung von NGINX Inc.
- 2013 NGINX plus (kommerzielle Version)

- Oktober 2023: NGINX 1.25.3



NGINX in Details

- Architektur
  - Modular
  - C
- Betriebssysteme
  - alle gängigen
    - Linux, OIS-X, FreeBSD
    - Windows nicht so stark performanced-optimiert



Learn 97 site reliability tips and best practices in this new O'Reilly ebook.
Download for free at nginx.com

nginx: Linux packages

- Supported distributions and versions
- Installation instructions
  - RHEL/CentOS
  - Debian
  - Ubuntu
  - SLES
  - Alpine
  - Amazon Linux
- Source Packages
- Dynamic Modules
- Signatures

Supported distributions and versions

nginx packages are available for the following Linux distributions and versions:

RHEL/CentOS

Version	Supported Platforms
7.4+	x86_64, ppc64le, aarch64/arm64
8.x	x86_64, aarch64/arm64

Debian

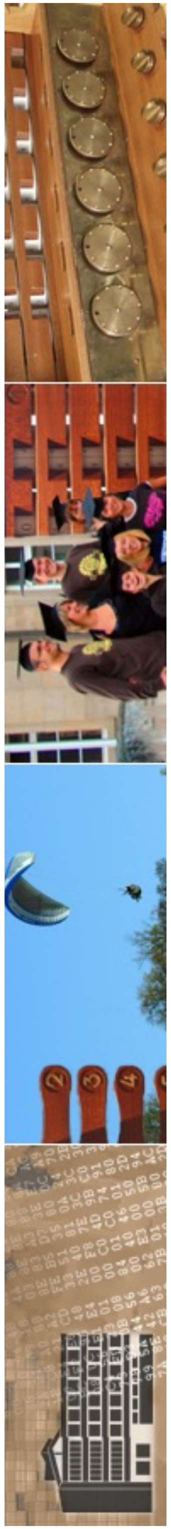
Version	Supported Platforms
10.x “buster”	x86_64, i386, aarch64/arm64

english
русский

news
about
download
security
documentation
faq
books
support

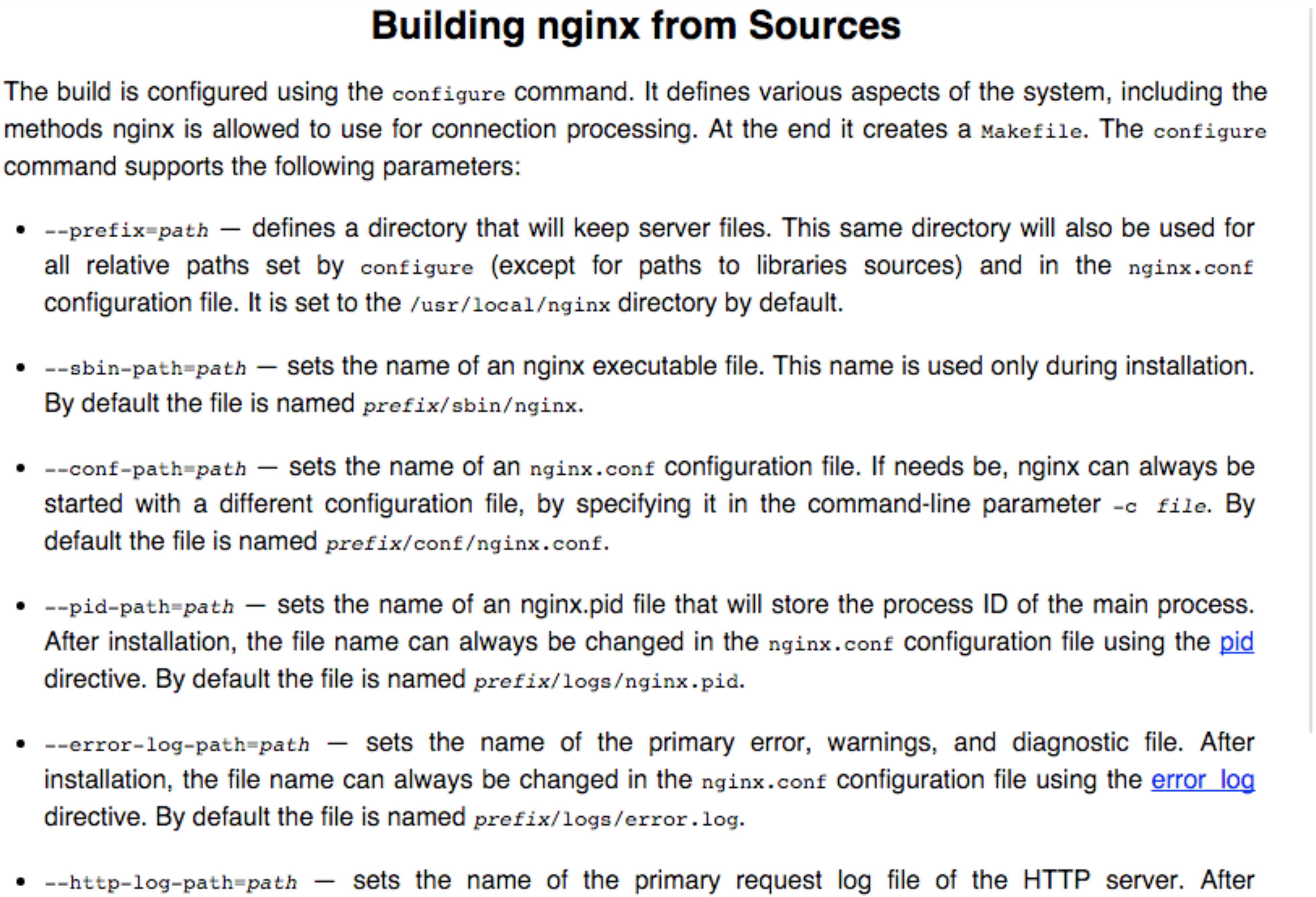
trac
twitter
blog

unit
njs



manuelle Installation

- ähnlich zu Apache
  - (als fertiges Linux-Paket)
  - aus den Source-Files mit
    - configure
    - make
    - make install



Building nginx from Sources

The build is configured using the `configure` command. It defines various aspects of the system, including the methods nginx is allowed to use for connection processing. At the end it creates a `Makefile`. The `configure` command supports the following parameters:

- `--prefix=`*`path`* — defines a directory that will keep server files. This same directory will also be used for all relative paths set by `configure` (except for paths to libraries sources) and in the `nginx.conf` configuration file. It is set to the `/usr/local/nginx` directory by default.

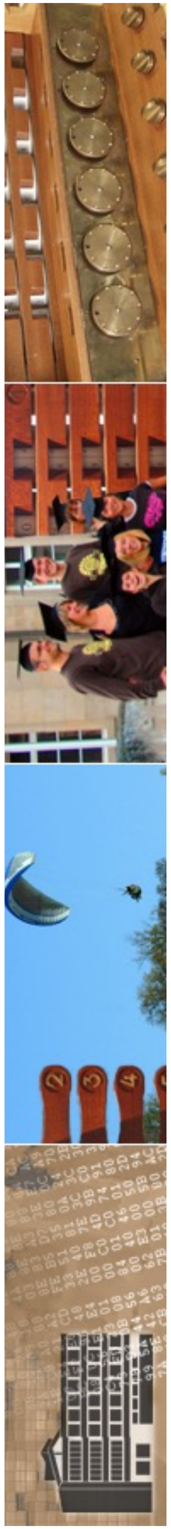
- `--sbin-path=`*`path`* — sets the name of an nginx executable file. This name is used only during installation. By default the file is named *`prefix`*`/sbin/nginx`.

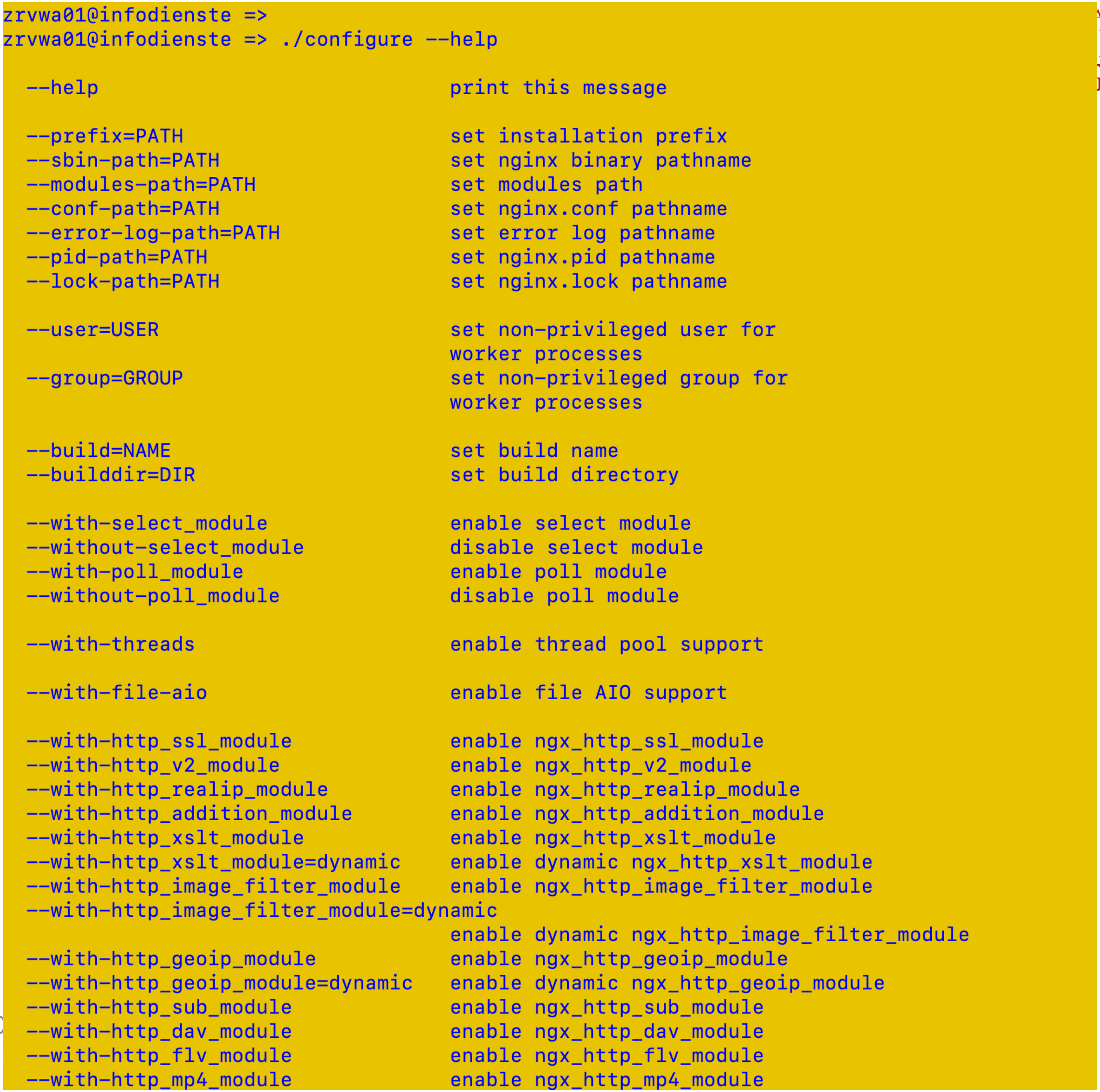
- `--conf-path=`*`path`* — sets the name of an `nginx.conf` configuration file. If needs be, nginx can always be started with a different configuration file, by specifying it in the command-line parameter `-c` *`file`*. By default the file is named *`prefix`*`/conf/nginx.conf`.

- `--pid-path=`*`path`* — sets the name of an nginx.pid file that will store the process ID of the main process. After installation, the file name can always be changed in the `nginx.conf` configuration file using the pid directive. By default the file is named *`prefix`*`/logs/nginx.pid`.

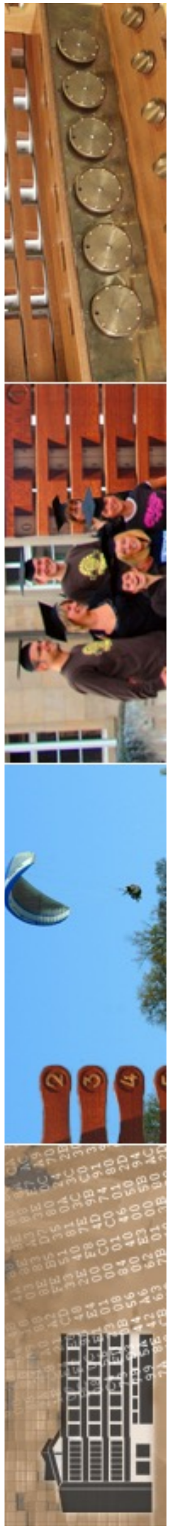
- `--error-log-path=`*`path`* — sets the name of the primary error, warnings, and diagnostic file. After installation, the file name can always be changed in the `nginx.conf` configuration file using the error_log directive. By default the file is named *`prefix`*`/logs/error.log`.

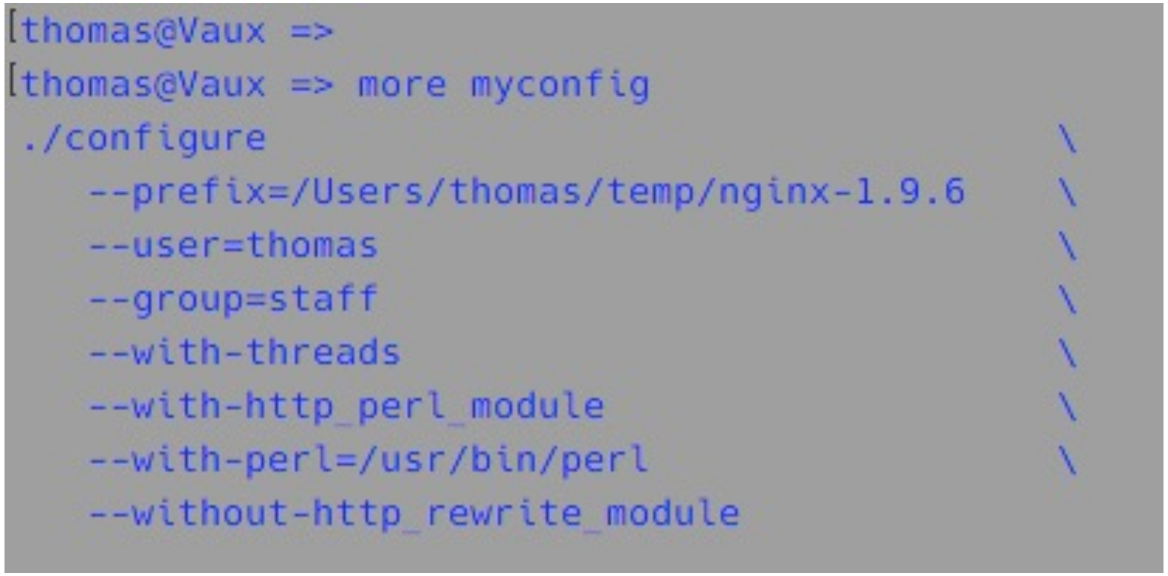
- `--http-log-path=`*`path`* — sets the name of the primary request log file of the HTTP server. After



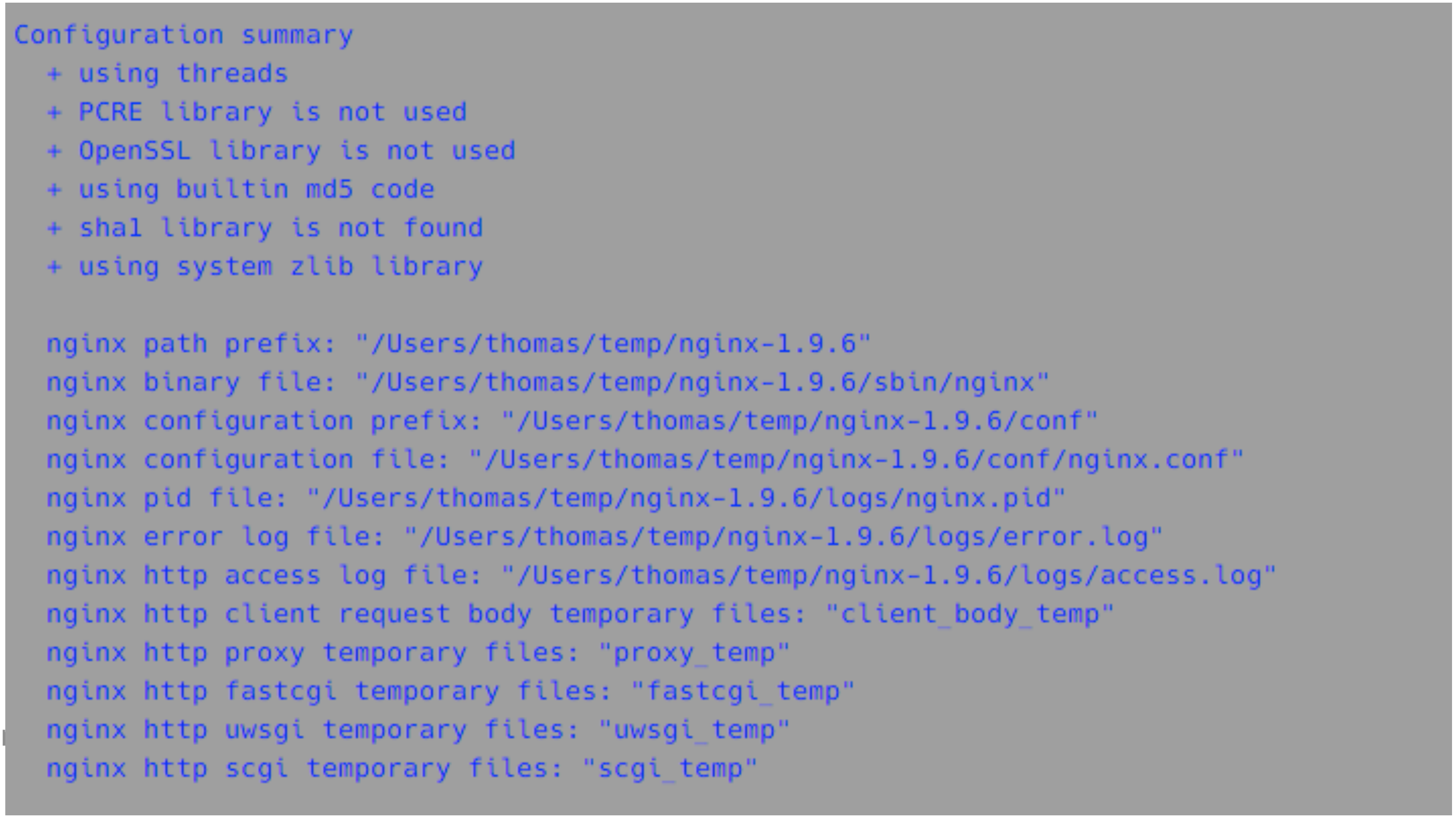


```
zrvwa01@infodienste =>
zrvwa01@infodienste => ./configure --help

  --help                             print this message

  --prefix=PATH                      set installation prefix
  --sbin-path=PATH                   set nginx binary pathname
  --modules-path=PATH                set modules path
  --conf-path=PATH                   set nginx.conf pathname
  --error-log-path=PATH              set error log pathname
  --pid-path=PATH                    set nginx.pid pathname
  --lock-path=PATH                   set nginx.lock pathname

  --user=USER                        set non-privileged user for
                                     worker processes
  --group=GROUP                      set non-privileged group for
                                     worker processes

  --build=NAME                       set build name
  --builddir=DIR                     set build directory

  --with-select_module               enable select module
  --without-select_module            disable select module
  --with-poll_module                 enable poll module
  --without-poll_module              disable poll module

  --with-threads                     enable thread pool support

  --with-file-aio                    enable file AIO support

  --with-http_ssl_module             enable ngx_http_ssl_module
  --with-http_v2_module              enable ngx_http_v2_module
  --with-http_realip_module          enable ngx_http_realip_module
  --with-http_addition_module        enable ngx_http_addition_module
  --with-http_xslt_module            enable ngx_http_xslt_module
  --with-http_xslt_module=dynamic    enable dynamic ngx_http_xslt_module
  --with-http_image_filter_module    enable ngx_http_image_filter_module
  --with-http_image_filter_module=dynamic
                                     enable dynamic ngx_http_image_filter_module
  --with-http_geoip_module           enable ngx_http_geoip_module
  --with-http_geoip_module=dynamic   enable dynamic ngx_http_geoip_module
  --with-http_sub_module             enable ngx_http_sub_module
  --with-http_dav_module             enable ngx_http_dav_module
  --with-http_flv_module             enable ngx_http_flv_module
  --with-http_mp4_module             enable ngx_http_mp4_module
```

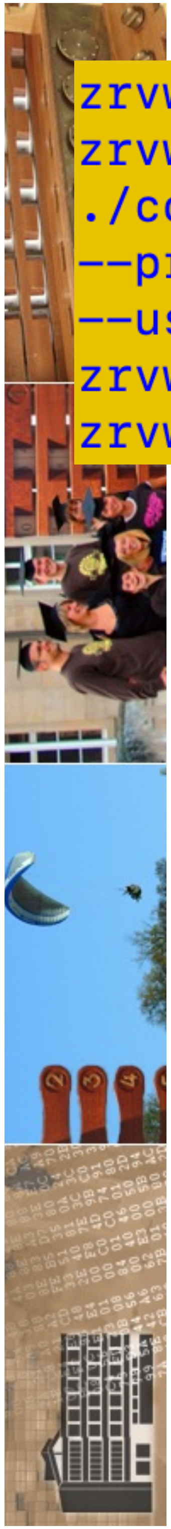


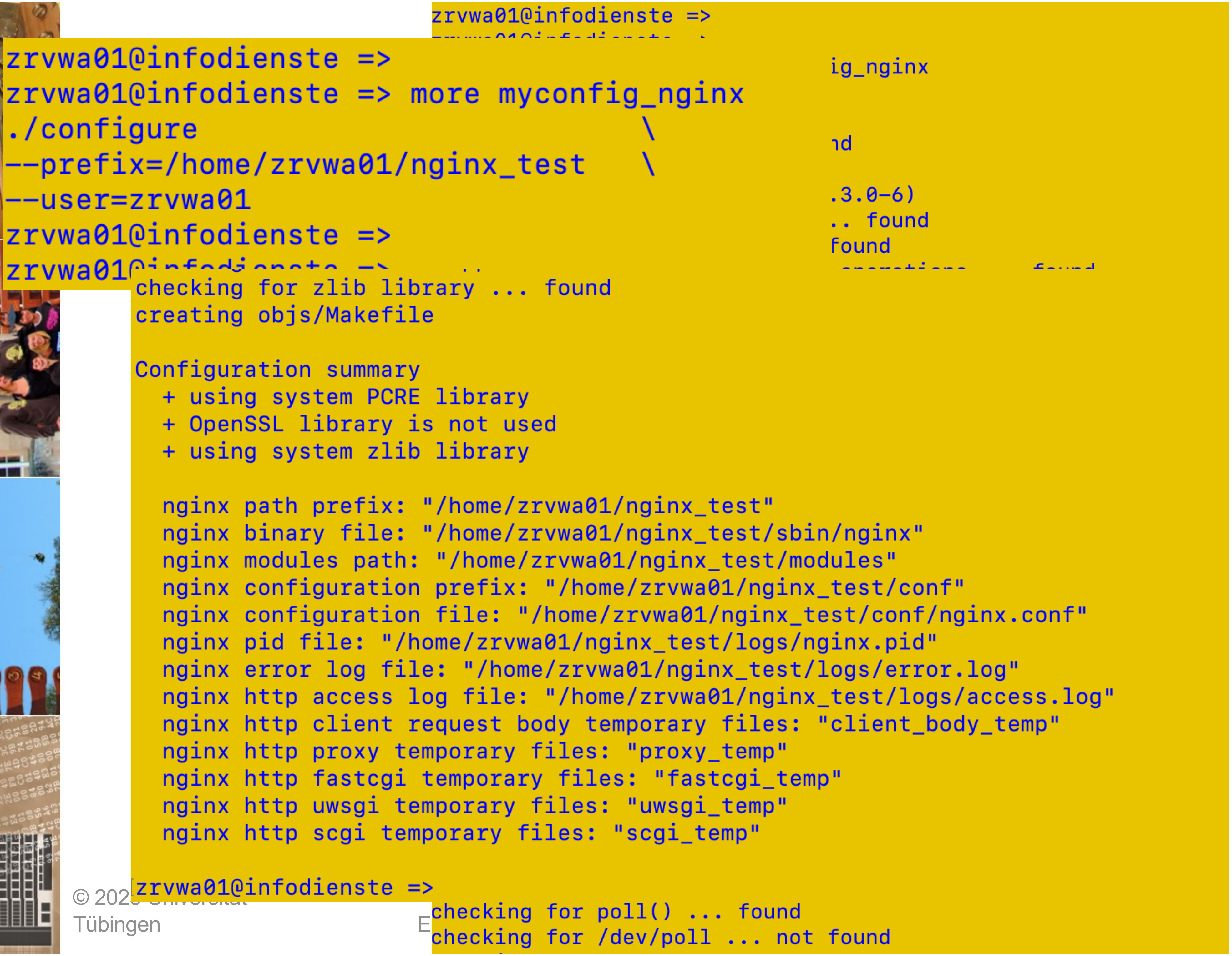


```
[thomas@Vaux =>
[thomas@Vaux => more myconfig
 ./configure                                     \
    --prefix=/Users/thomas/temp/nginx-1.9.6     \
    --user=thomas                                \
    --group=staff                                \
    --with-threads                               \
    --with-http_perl_module                      \
    --with-perl=/usr/bin/perl                    \
    --without-http_rewrite_module
```



```
Configuration summary
  + using threads
  + PCRE library is not used
  + OpenSSL library is not used
  + using builtin md5 code
  + sha1 library is not found
  + using system zlib library

  nginx path prefix: "/Users/thomas/temp/nginx-1.9.6"
  nginx binary file: "/Users/thomas/temp/nginx-1.9.6/sbin/nginx"
  nginx configuration prefix: "/Users/thomas/temp/nginx-1.9.6/conf"
  nginx configuration file: "/Users/thomas/temp/nginx-1.9.6/conf/nginx.conf"
  nginx pid file: "/Users/thomas/temp/nginx-1.9.6/logs/nginx.pid"
  nginx error log file: "/Users/thomas/temp/nginx-1.9.6/logs/error.log"
  nginx http access log file: "/Users/thomas/temp/nginx-1.9.6/logs/access.log"
  nginx http client request body temporary files: "client_body_temp"
  nginx http proxy temporary files: "proxy_temp"
  nginx http fastcgi temporary files: "fastcgi_temp"
  nginx http uwsgi temporary files: "uwsgi_temp"
  nginx http scgi temporary files: "scgi_temp"
```





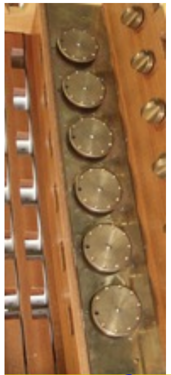
```
zrvwa01@infodienste =>
```

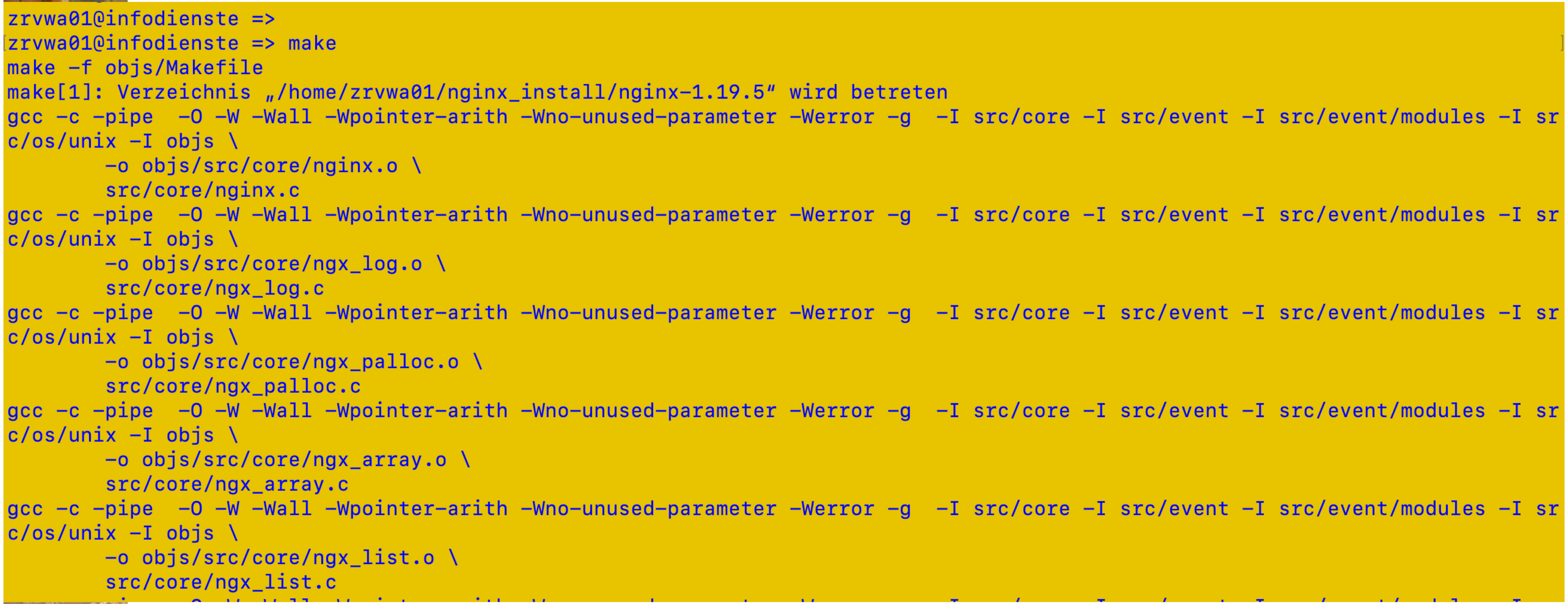
```
ig_nginx

nd

.3.0-6)
.. found
found
```

```
zrvwa01@infodienste =>
zrvwa01@infodienste => more myconfig_nginx
./configure                             \
--prefix=/home/zrvwa01/nginx_test       \
--user=zrvwa01
zrvwa01@infodienste =>
```

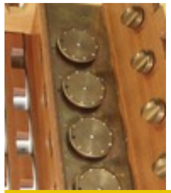
```
checking for zlib library ... found
creating objs/Makefile

Configuration summary
  + using system PCRE library
  + OpenSSL library is not used
  + using system zlib library

  nginx path prefix: "/home/zrvwa01/nginx_test"
  nginx binary file: "/home/zrvwa01/nginx_test/sbin/nginx"
  nginx modules path: "/home/zrvwa01/nginx_test/modules"
  nginx configuration prefix: "/home/zrvwa01/nginx_test/conf"
  nginx configuration file: "/home/zrvwa01/nginx_test/conf/nginx.conf"
  nginx pid file: "/home/zrvwa01/nginx_test/logs/nginx.pid"
  nginx error log file: "/home/zrvwa01/nginx_test/logs/error.log"
  nginx http access log file: "/home/zrvwa01/nginx_test/logs/access.log"
  nginx http client request body temporary files: "client_body_temp"
  nginx http proxy temporary files: "proxy_temp"
  nginx http fastcgi temporary files: "fastcgi_temp"
  nginx http uwsgi temporary files: "uwsgi_temp"
  nginx http scgi temporary files: "scgi_temp"

zrvwa01@infodienste =>
```

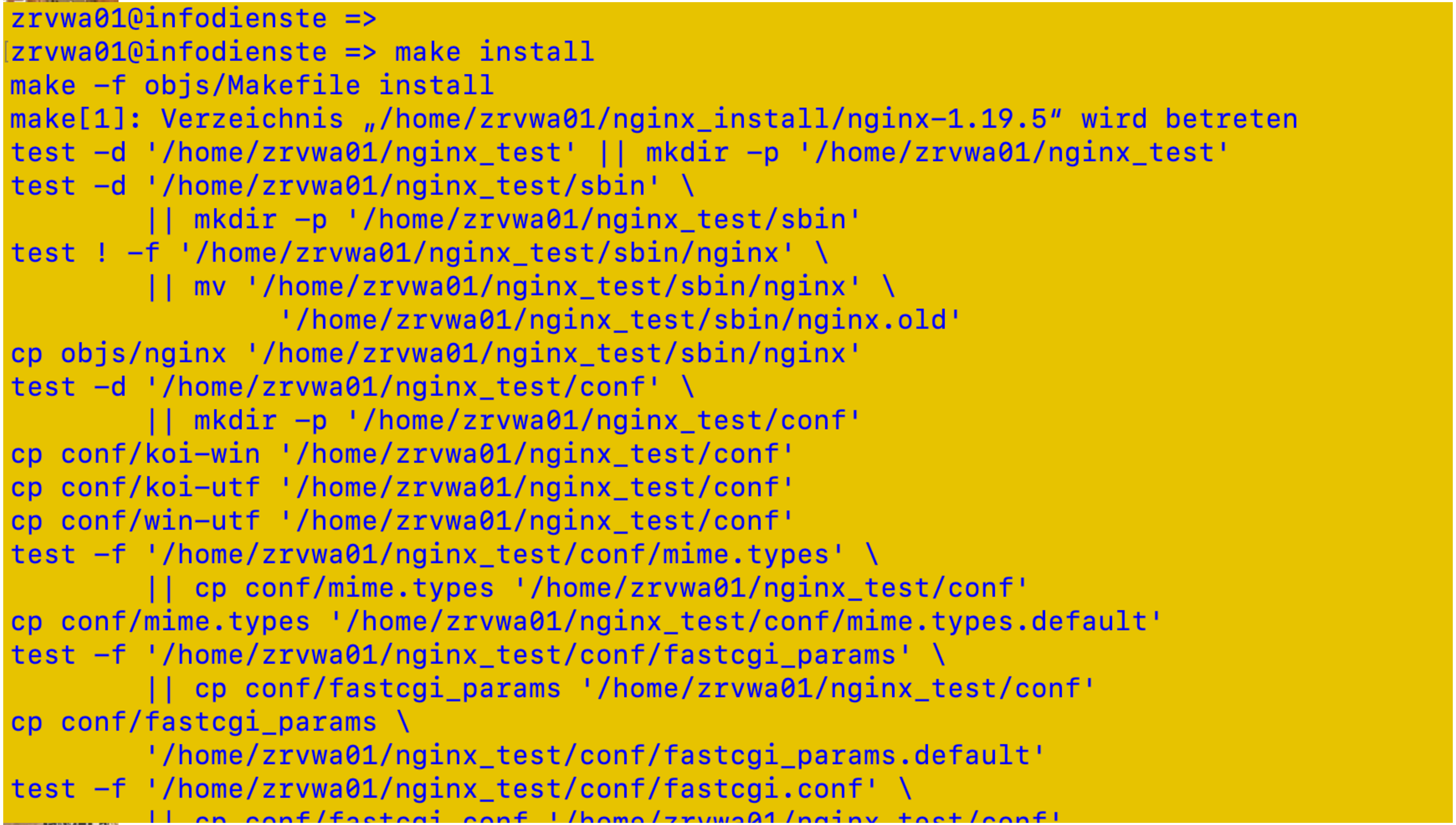
```
checking for poll() ... found
checking for /dev/poll ... not found
```



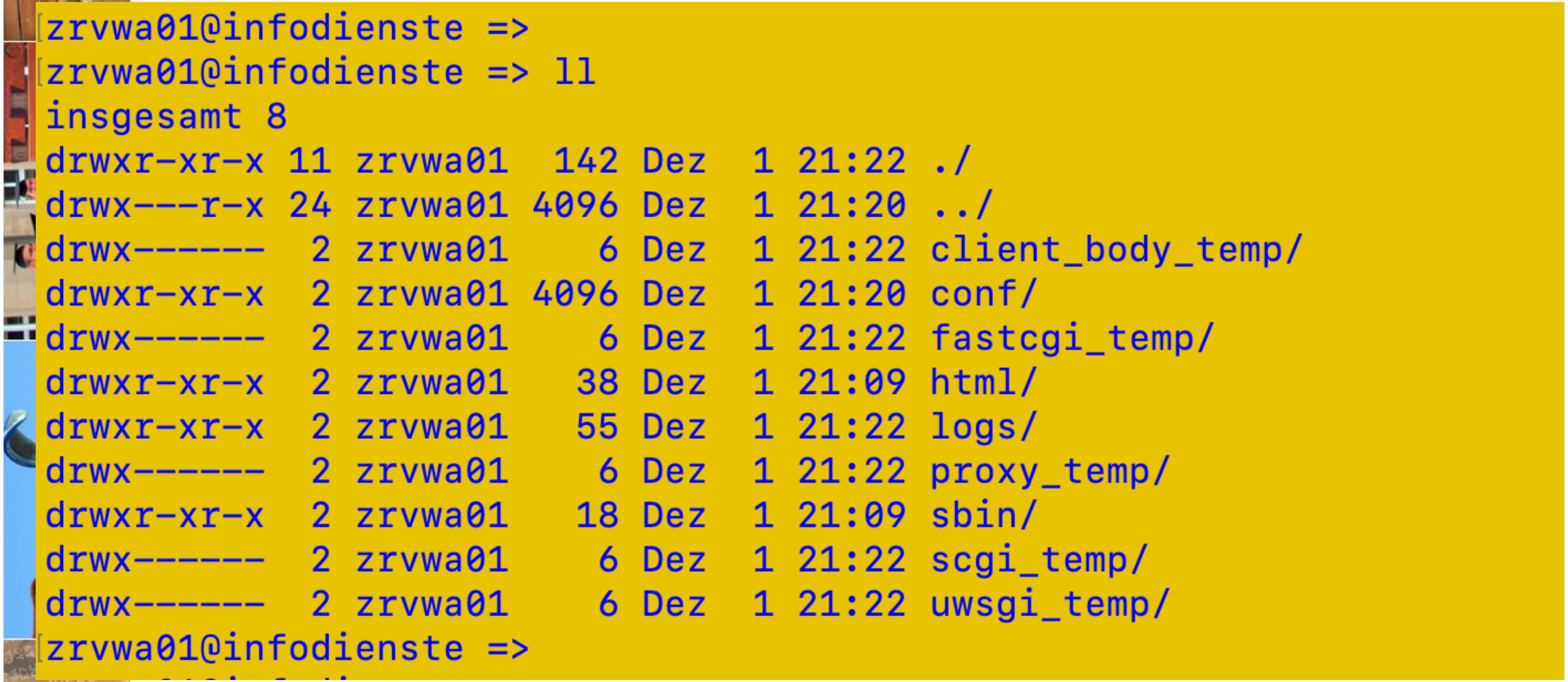


```
zrvwa01@infodienste =>
zrvwa01@infodienste => make
make -f objs/Makefile
make[1]: Verzeichnis „/home/zrvwa01/nginx_install/nginx-1.19.5“ wird betreten
gcc -c -pipe  -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g  -I src/core -I src/event -I src/event/modules -I sr
c/os/unix -I objs \
        -o objs/src/core/nginx.o \
        src/core/nginx.c
gcc -c -pipe  -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g  -I src/core -I src/event -I src/event/modules -I sr
c/os/unix -I objs \
        -o objs/src/core/ngx_log.o \
        src/core/ngx_log.c
gcc -c -pipe  -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g  -I src/core -I src/event -I src/event/modules -I sr
c/os/unix -I objs \
        -o objs/src/core/ngx_palloc.o \
        src/core/ngx_palloc.c
gcc -c -pipe  -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g  -I src/core -I src/event -I src/event/modules -I sr
c/os/unix -I objs \
        -o objs/src/core/ngx_array.o \
        src/core/ngx_array.c
gcc -c -pipe  -O -W -Wall -Wpointer-arith -Wno-unused-parameter -Werror -g  -I src/core -I src/event -I src/event/modules -I sr
c/os/unix -I objs \
        -o objs/src/core/ngx_list.o \
        src/core/ngx_list.c
```

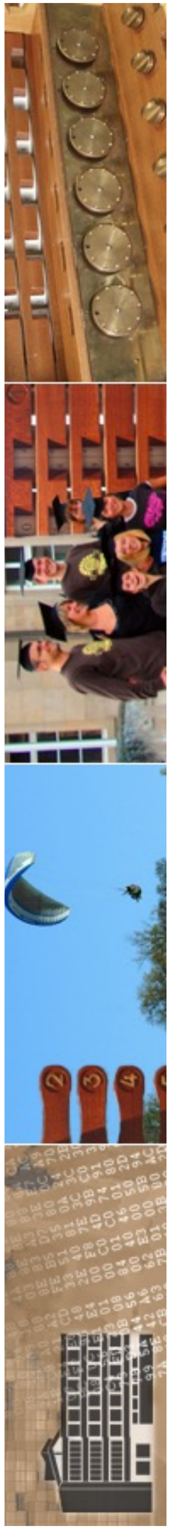


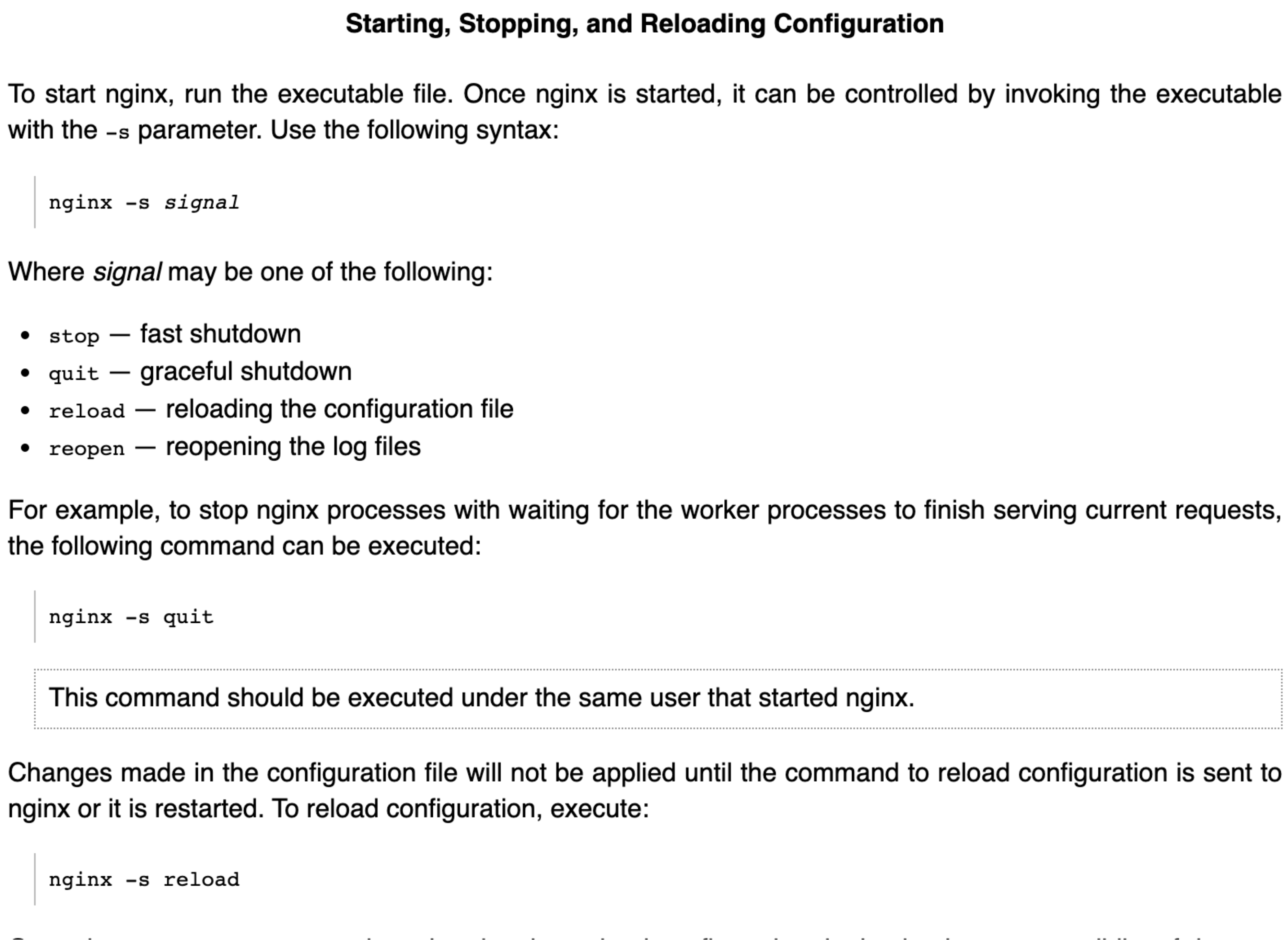


```
zrvwa01@infodienste =>
[zrvwa01@infodienste => make install
make -f objs/Makefile install
make[1]: Verzeichnis „/home/zrvwa01/nginx_install/nginx-1.19.5“ wird betreten
test -d '/home/zrvwa01/nginx_test' || mkdir -p '/home/zrvwa01/nginx_test'
test -d '/home/zrvwa01/nginx_test/sbin' \
        || mkdir -p '/home/zrvwa01/nginx_test/sbin'
test ! -f '/home/zrvwa01/nginx_test/sbin/nginx' \
        || mv '/home/zrvwa01/nginx_test/sbin/nginx' \
                '/home/zrvwa01/nginx_test/sbin/nginx.old'
cp objs/nginx '/home/zrvwa01/nginx_test/sbin/nginx'
test -d '/home/zrvwa01/nginx_test/conf' \
        || mkdir -p '/home/zrvwa01/nginx_test/conf'
cp conf/koi-win '/home/zrvwa01/nginx_test/conf'
cp conf/koi-utf '/home/zrvwa01/nginx_test/conf'
cp conf/win-utf '/home/zrvwa01/nginx_test/conf'
test -f '/home/zrvwa01/nginx_test/conf/mime.types' \
        || cp conf/mime.types '/home/zrvwa01/nginx_test/conf'
cp conf/mime.types '/home/zrvwa01/nginx_test/conf/mime.types.default'
test -f '/home/zrvwa01/nginx_test/conf/fastcgi_params' \
        || cp conf/fastcgi_params '/home/zrvwa01/nginx_test/conf'
cp conf/fastcgi_params \
        '/home/zrvwa01/nginx_test/conf/fastcgi_params.default'
test -f '/home/zrvwa01/nginx_test/conf/fastcgi.conf' \
        || cp conf/fastcgi.conf '/home/zrvwa01/nginx_test/conf'
```



```
[zrvwa01@infodienste =>
[zrvwa01@infodienste => ll
insgesamt 8
drwxr-xr-x 11 zrvwa01  142 Dez  1 21:22 ./
drwx---r-x 24 zrvwa01 4096 Dez  1 21:20 ../
drwx------  2 zrvwa01    6 Dez  1 21:22 client_body_temp/
drwxr-xr-x  2 zrvwa01 4096 Dez  1 21:20 conf/
drwx------  2 zrvwa01    6 Dez  1 21:22 fastcgi_temp/
drwxr-xr-x  2 zrvwa01   38 Dez  1 21:09 html/
drwxr-xr-x  2 zrvwa01   55 Dez  1 21:22 logs/
drwx------  2 zrvwa01    6 Dez  1 21:22 proxy_temp/
drwxr-xr-x  2 zrvwa01   18 Dez  1 21:09 sbin/
drwx------  2 zrvwa01    6 Dez  1 21:22 scgi_temp/
drwx------  2 zrvwa01    6 Dez  1 21:22 uwsgi_temp/
[zrvwa01@infodienste =>
```





Starting, Stopping, and Reloading Configuration

To start nginx, run the executable file. Once nginx is started, it can be controlled by invoking the executable with the `-s` parameter. Use the following syntax:

```
nginx -s signal
```

Where *signal* may be one of the following:

- `stop` — fast shutdown
- `quit` — graceful shutdown
- `reload` — reloading the configuration file
- `reopen` — reopening the log files

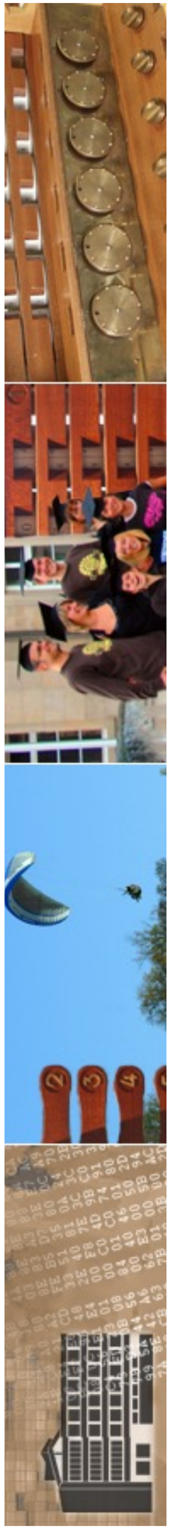
For example, to stop nginx processes with waiting for the worker processes to finish serving current requests, the following command can be executed:

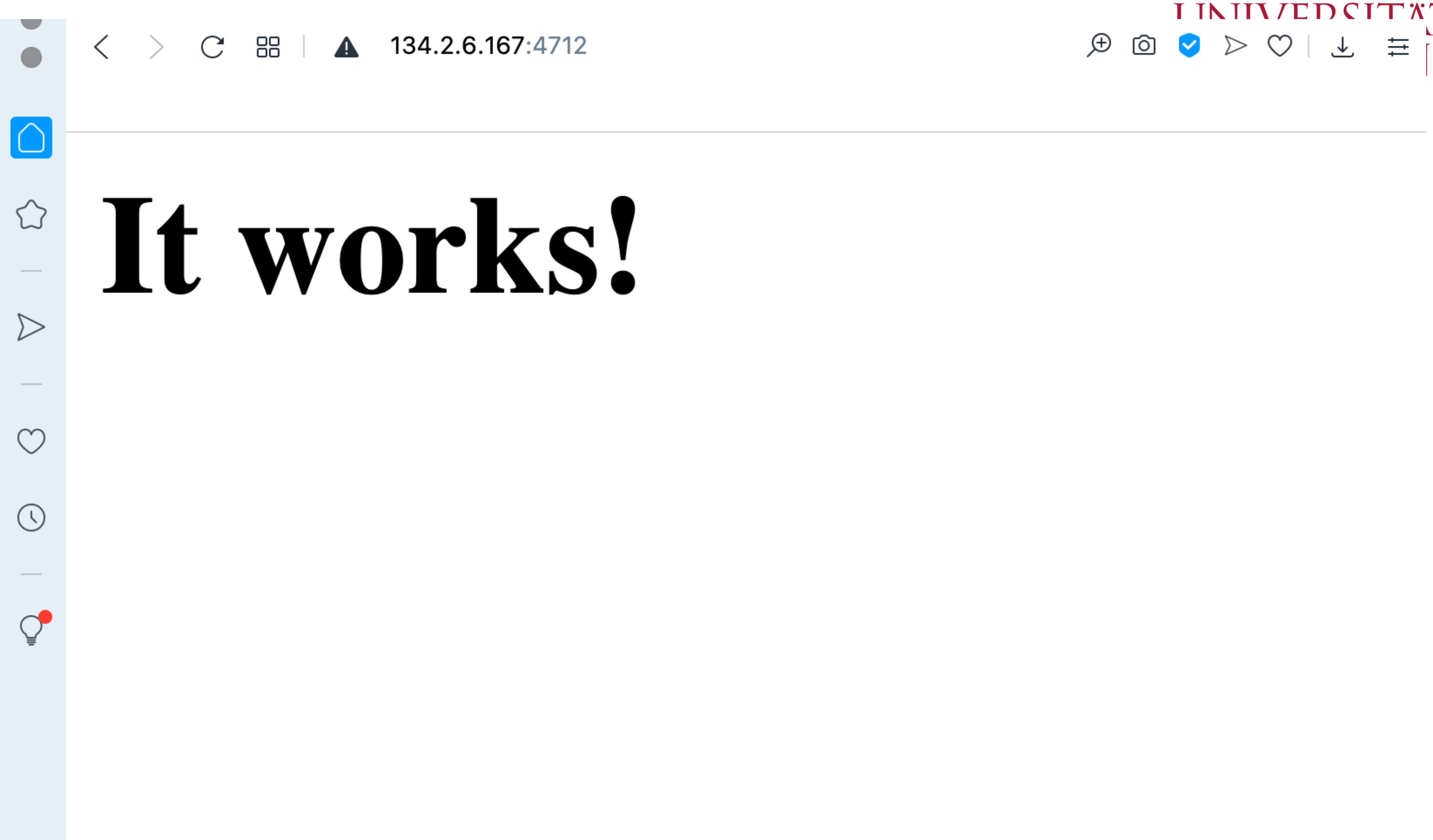
```
nginx -s quit
```

> This command should be executed under the same user that started nginx.

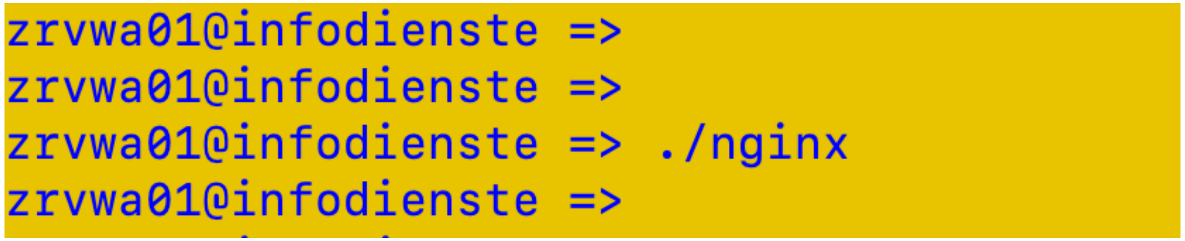
Changes made in the configuration file will not be applied until the command to reload configuration is sent to nginx or it is restarted. To reload configuration, execute:

```
nginx -s reload
```

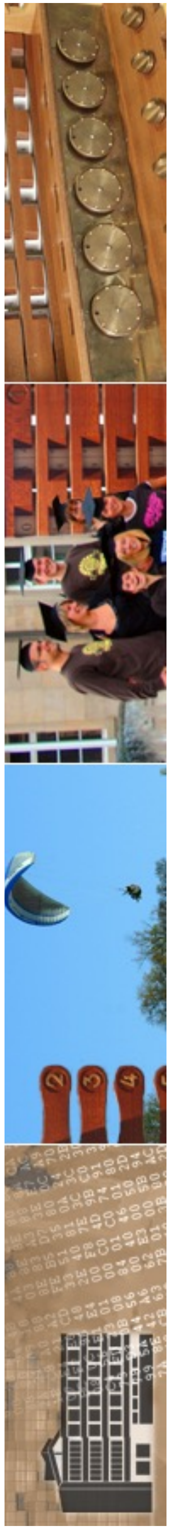




It works!



```
zrvwa01@infodienste =>
zrvwa01@infodienste =>
zrvwa01@infodienste => ./nginx
zrvwa01@infodienste =>
```



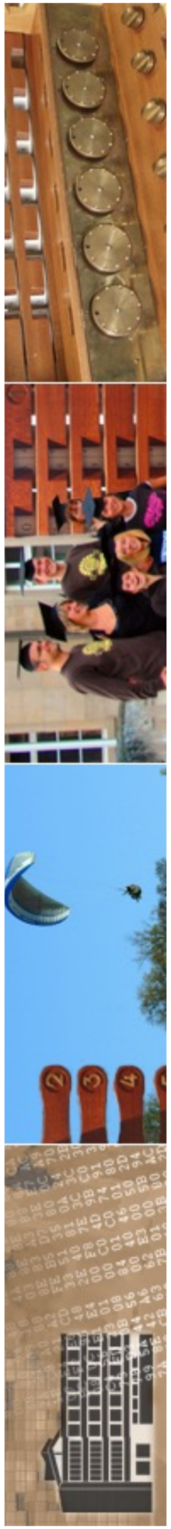
Konfiguration

- zentrale Konfigurationsdatei ist

  `nginx.conf`

  im Unterordner conf

- hat nur **117 Zeilen…**



compilieren: Apache versus nginx

Apache 2.4.51

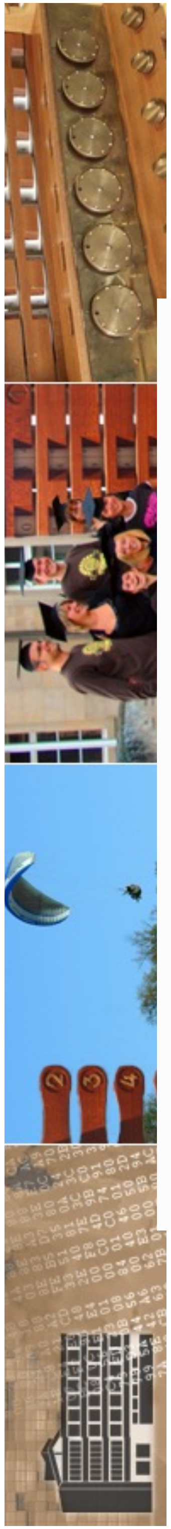
- `make | wc -l`

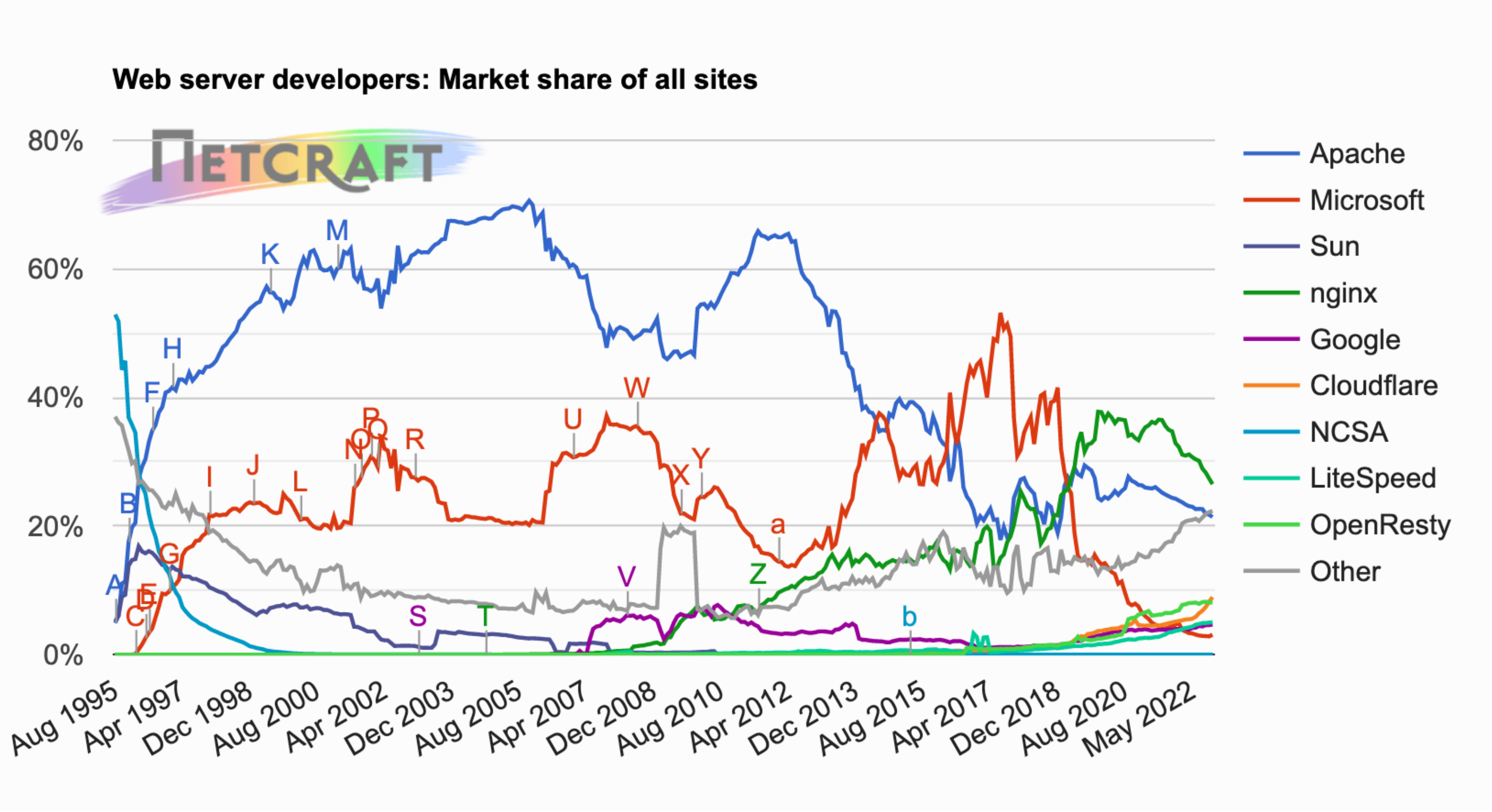
527

nginx 1.22.1

- `make | wc -l`

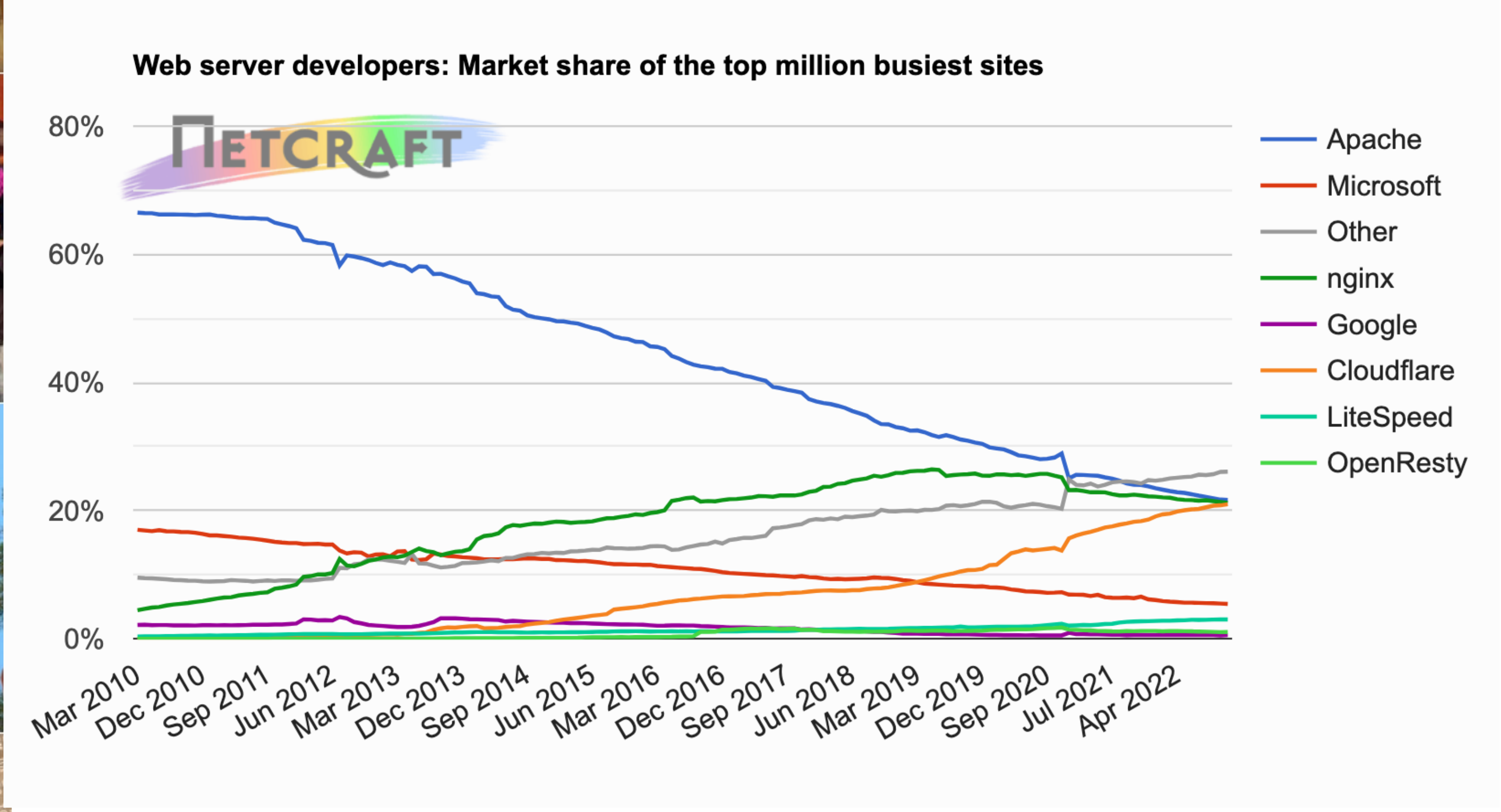
491

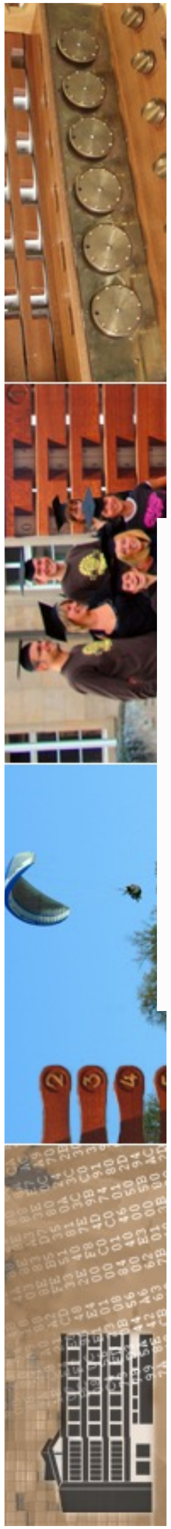


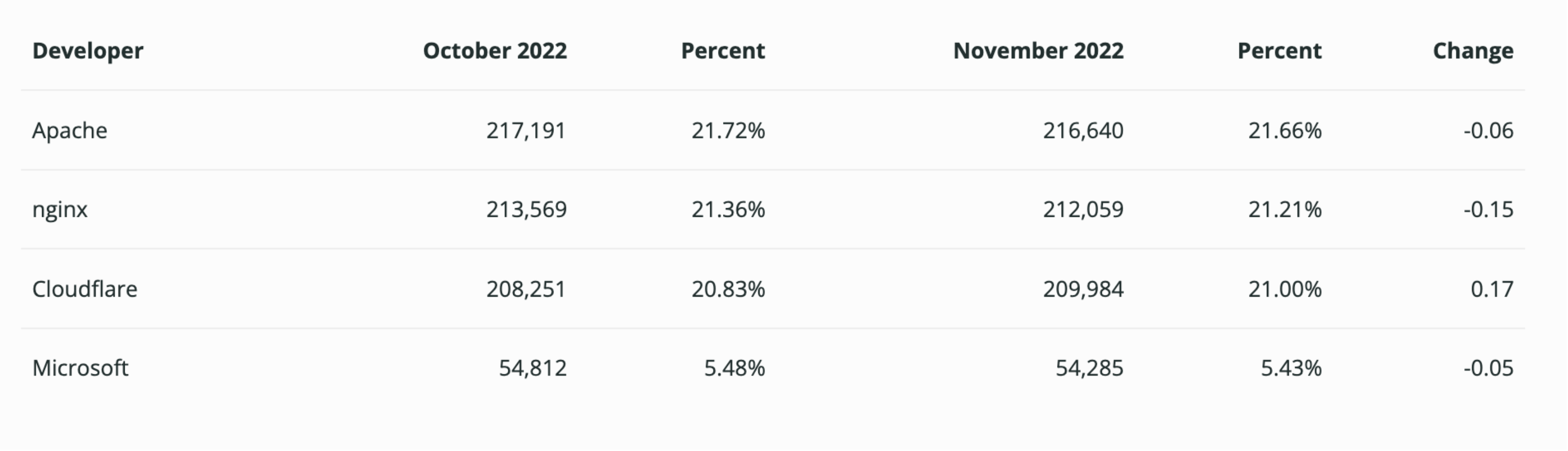


Web server developers: Market share of all sites

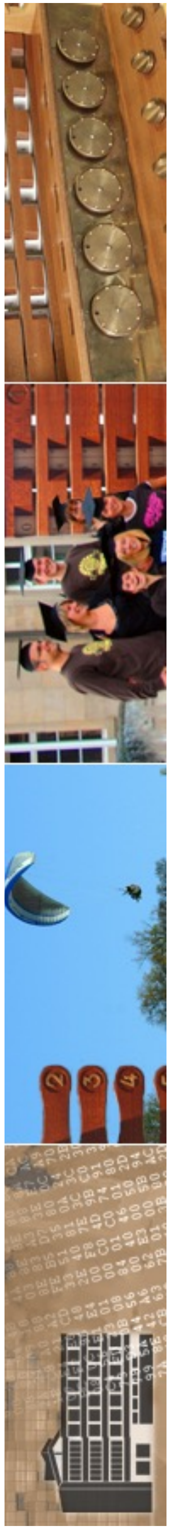
https://news.netcraft.com/archives/category/web-server-survey/

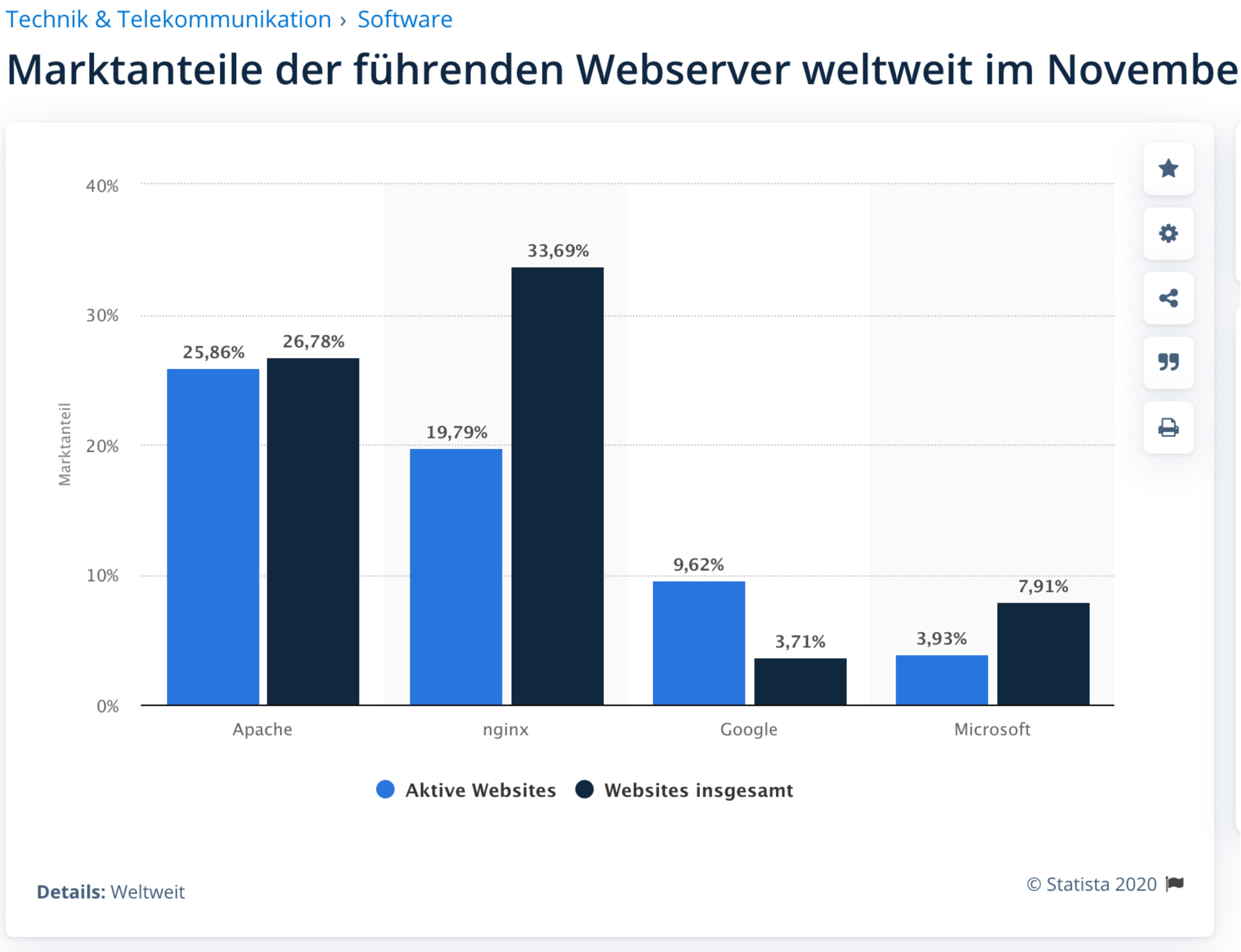

Web server developers: Market share of the top million busiest sites
NETCRAFT
80%
60%
40%
20%
0%
Mar 2010
Dec 2010
Sep 2011
Jun 2012
Mar 2013
Dec 2013
Sep 2014
Jun 2015
Mar 2016
Dec 2016
Sep 2017
Jun 2018
Mar 2019
Dec 2019
Sep 2020
Jul 2021
Apr 2022
Apache
Microsoft
Other
nginx
Google
Cloudflare
LiteSpeed
OpenResty





Developer	October 2022	Percent	November 2022	Percent	Change
Apache	217,191	21.72%	216,640	21.66%	-0.06
nginx	213,569	21.36%	212,059	21.21%	-0.15
Cloudflare	208,251	20.83%	209,984	21.00%	0.17
Microsoft	54,812	5.48%	54,285	5.43%	-0.05





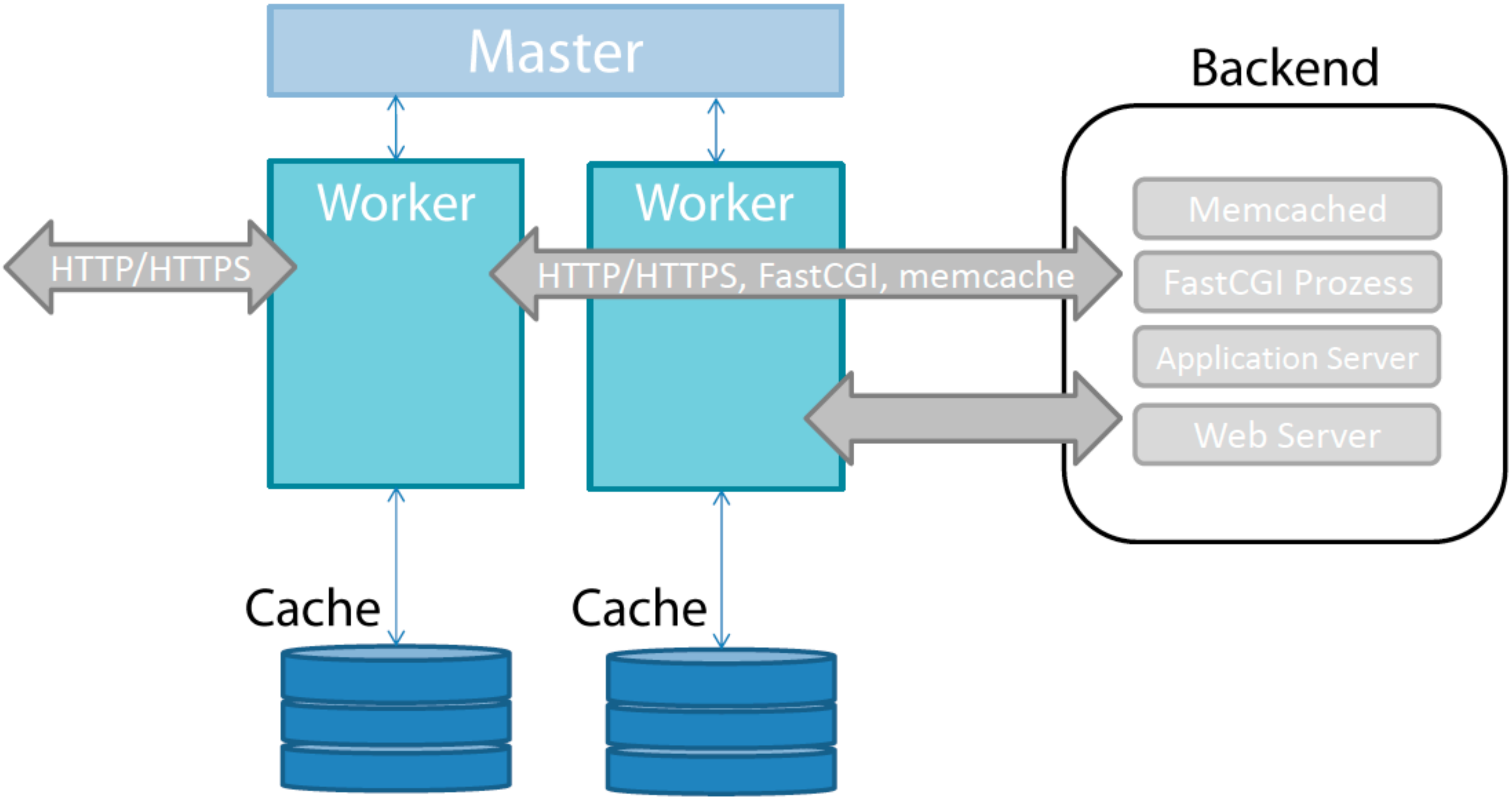
Technik & Telekommunikation › Software

Marktanteile der führenden Webserver weltweit im Novembe

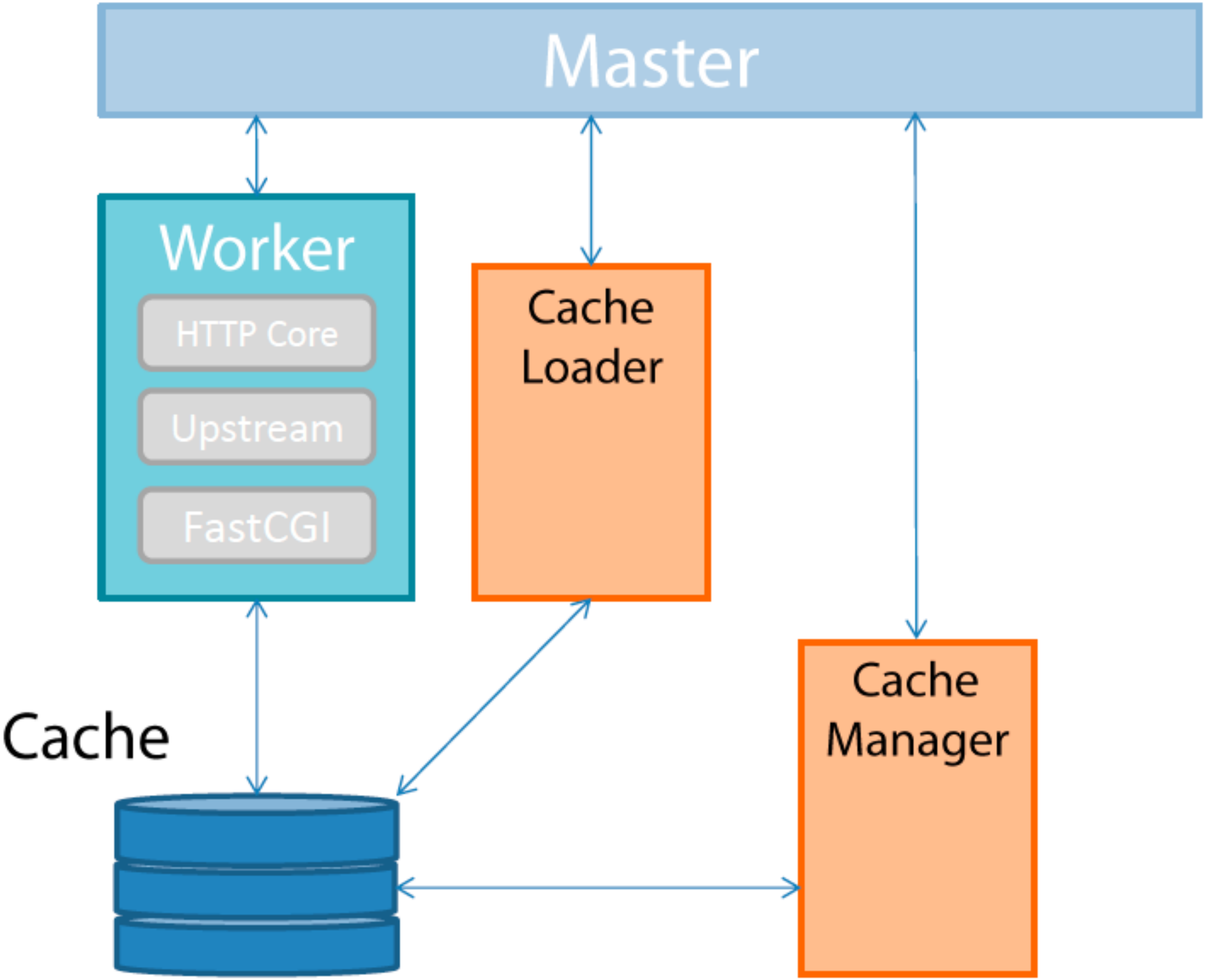
Details: Weltweit

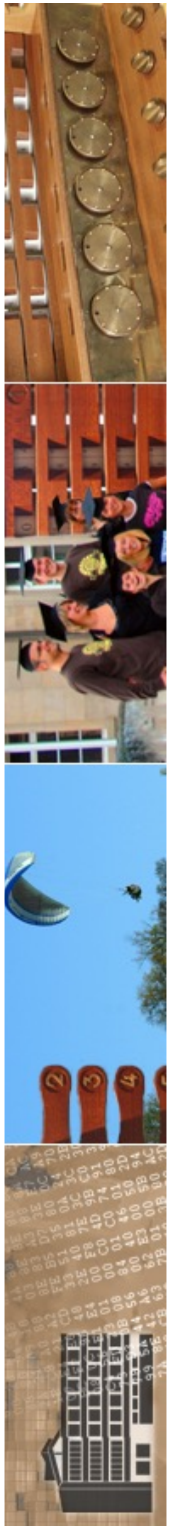
© Statista 2020

Nginx Architektur



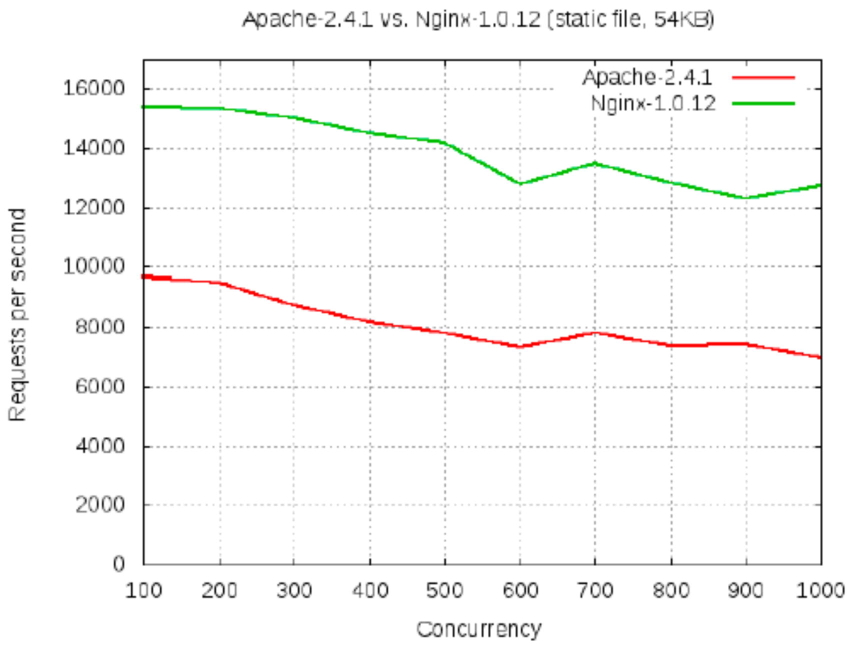
Nginx Architektur Cache



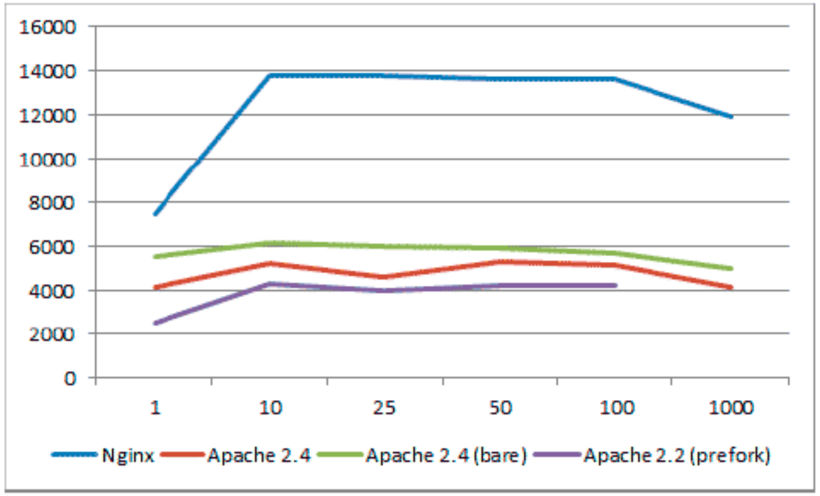


Benchmarks

- Webserver-Benchmakls: schwierig wegen zahlreicher Parameter
- NGINX vorteilhaft bei großer Nutzerzahl und *statischem Inhalt*



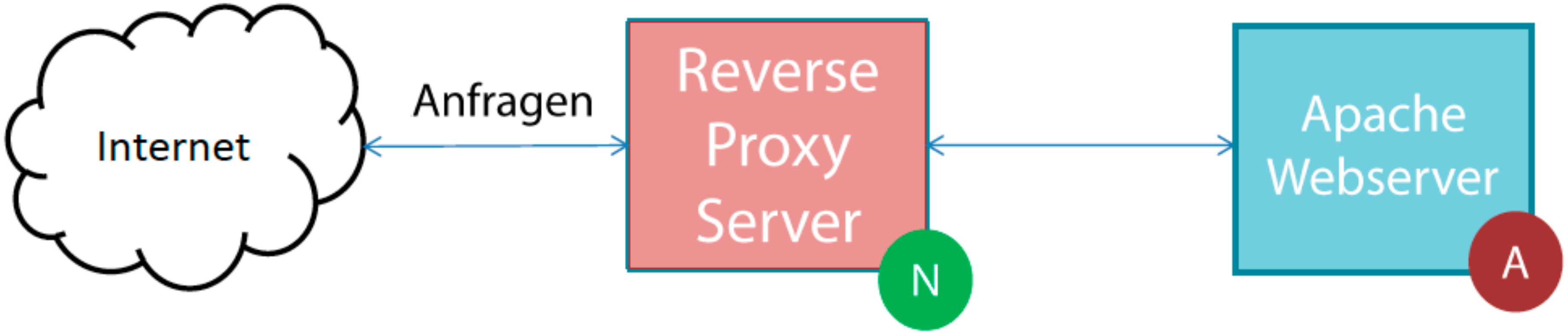
http://tengine.taobao.org/images/benchmark2.png

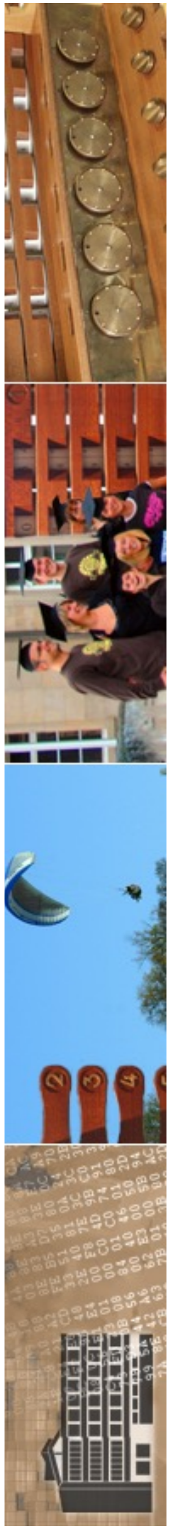


http://www.eschrade.com/page/performance-of-apache-2-4-with-the-event-mpm-compared-to-nginx/

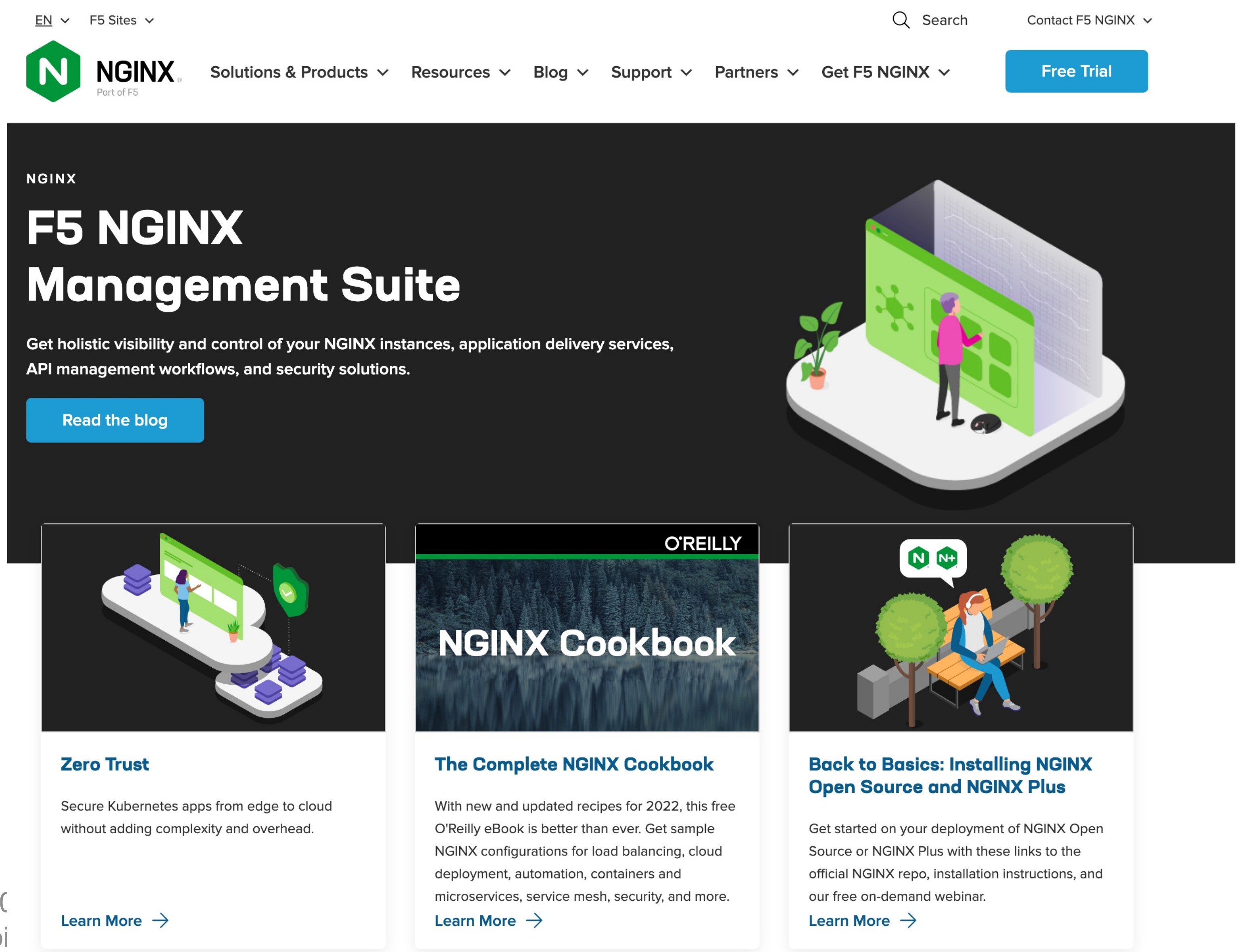
Hybrid-Model: Apache und NGINX

- NGINX: schnelle Auslieferung statischer Inhalte
- Apache: dynamische Inhalte





kommerzielle Version



NGINX

F5 NGINX Management Suite

Get holistic visibility and control of your NGINX instances, application delivery services, API management workflows, and security solutions.

Zero Trust

Secure Kubernetes apps from edge to cloud without adding complexity and overhead.

Learn More →

The Complete NGINX Cookbook

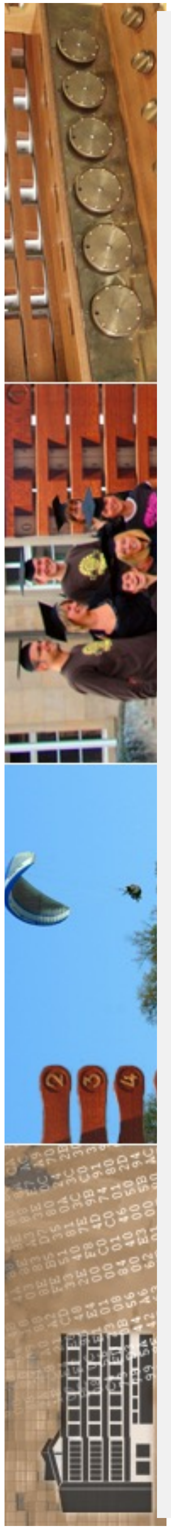
With new and updated recipes for 2022, this free O'Reilly eBook is better than ever. Get sample NGINX configurations for load balancing, cloud deployment, automation, containers and microservices, service mesh, security, and more.

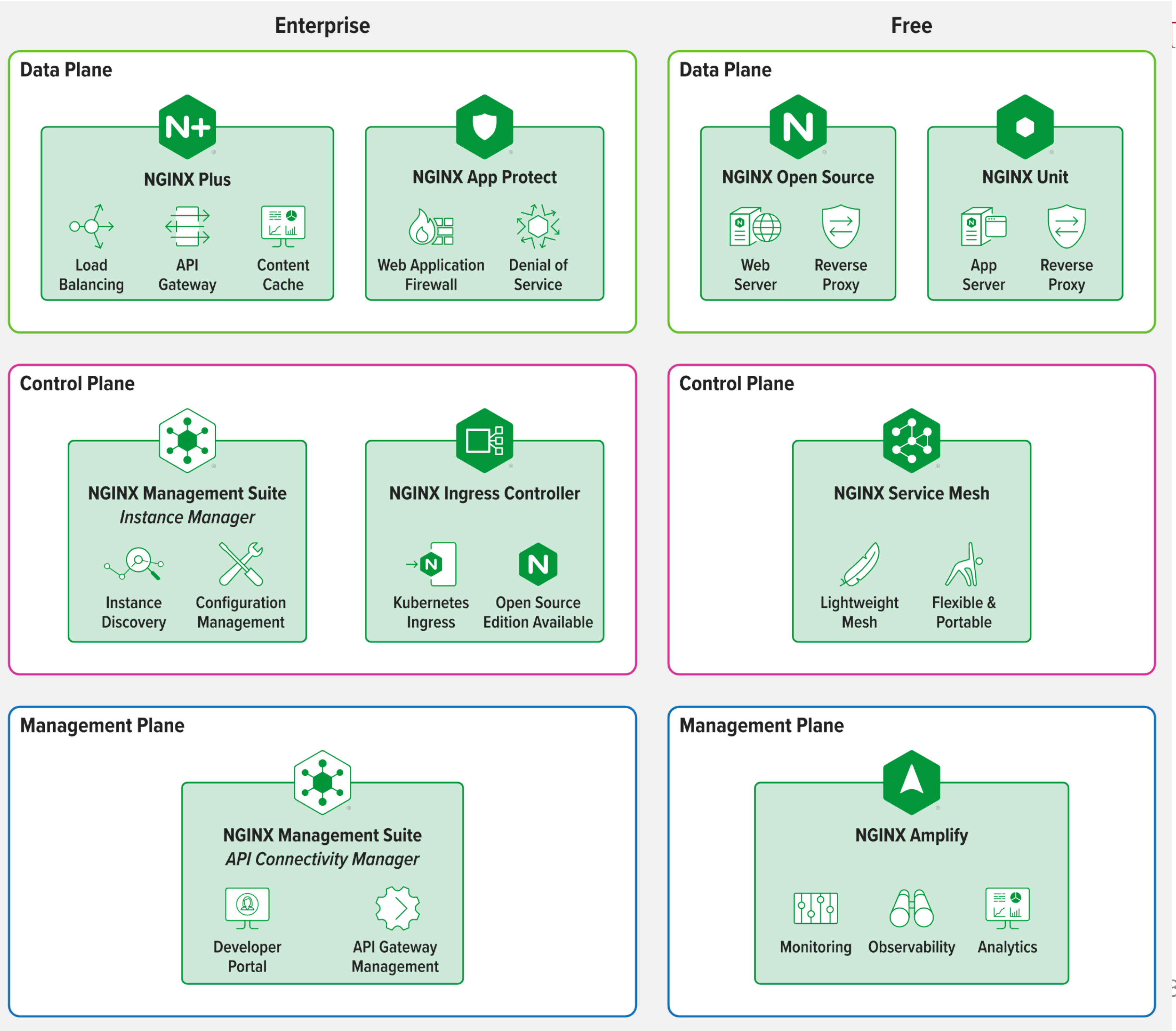
Learn More →

Back to Basics: Installing NGINX Open Source and NGINX Plus

Get started on your deployment of NGINX Open Source or NGINX Plus with these links to the official NGINX repo, installation instructions, and our free on-demand webinar.

Learn More →





Enterprise

Data Plane

NGINX Plus

- Load Balancing
- API Gateway
- Content Cache

NGINX App Protect

- Web Application Firewall
- Denial of Service

Control Plane

NGINX Management Suite
Instance Manager

- Instance Discovery
- Configuration Management

NGINX Ingress Controller

- Kubernetes Ingress
- Open Source Edition Available

Management Plane

NGINX Management Suite
API Connectivity Manager

- Developer Portal
- API Gateway Management

Free

Data Plane

NGINX Open Source

- Web Server
- Reverse Proxy

NGINX Unit

- App Server
- Reverse Proxy

Control Plane

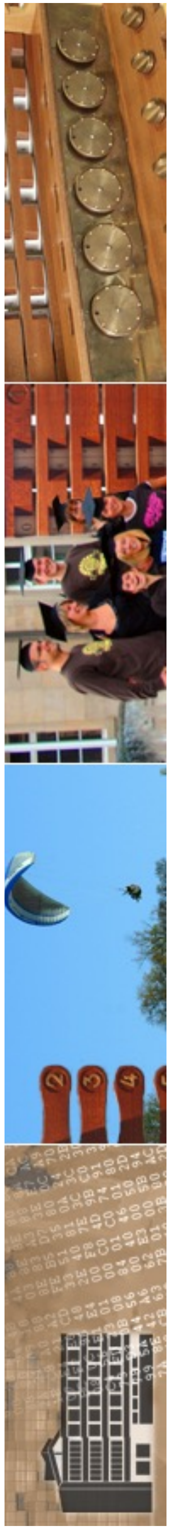
NGINX Service Mesh

- Lightweight Mesh
- Flexible & Portable

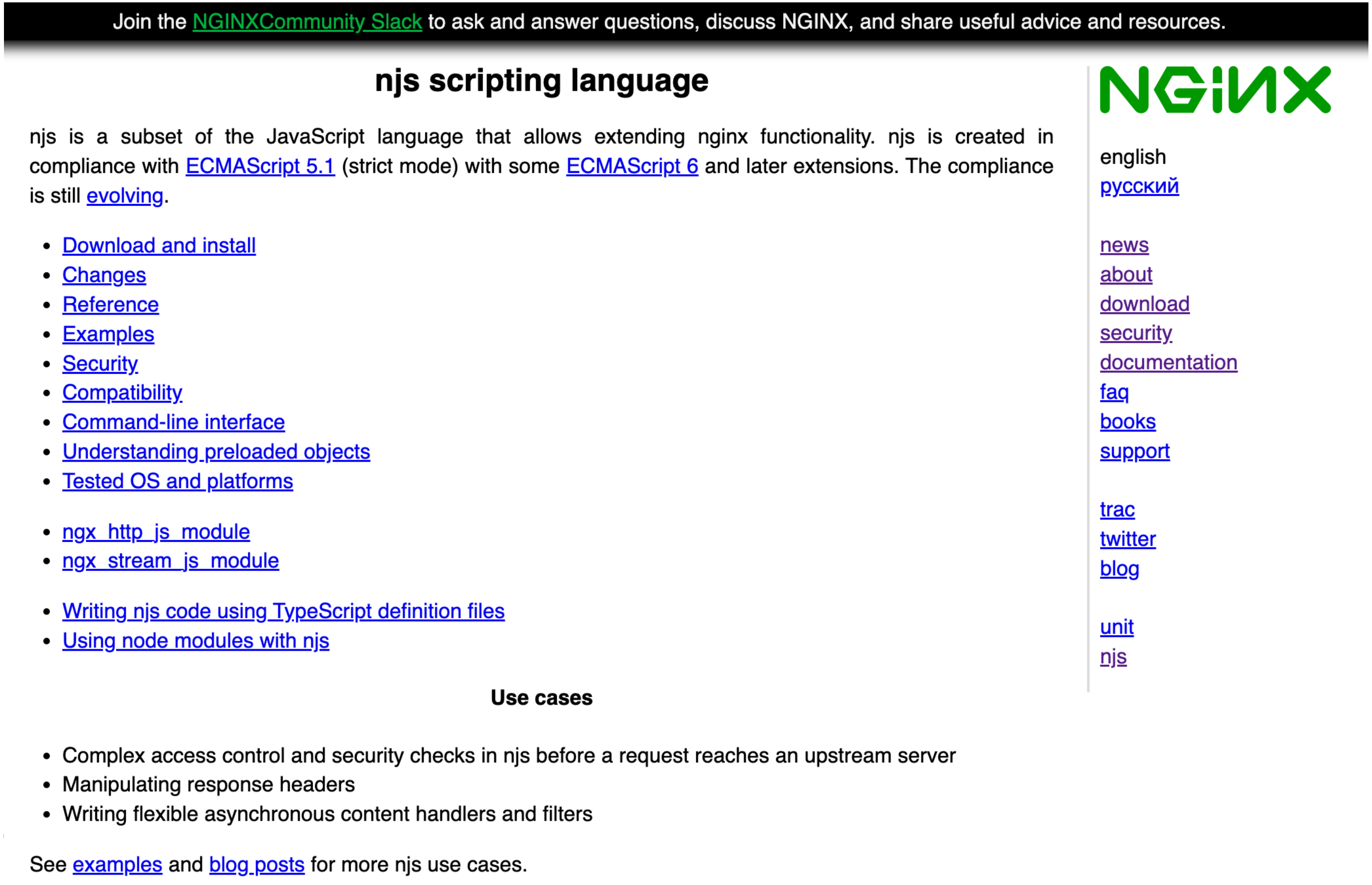
Management Plane

NGINX Amplify

- Monitoring
- Observability
- Analytics



njs scripting language



Join the NGINXCommunity Slack to ask and answer questions, discuss NGINX, and share useful advice and resources.

njs scripting language

njs is a subset of the JavaScript language that allows extending nginx functionality. njs is created in compliance with ECMAScript 5.1 (strict mode) with some ECMAScript 6 and later extensions. The compliance is still evolving.

- Download and install
- Changes
- Reference
- Examples
- Security
- Compatibility
- Command-line interface
- Understanding preloaded objects
- Tested OS and platforms

- ngx_http_js_module
- ngx_stream_js_module

- Writing njs code using TypeScript definition files
- Using node modules with njs

Use cases

- Complex access control and security checks in njs before a request reaches an upstream server
- Manipulating response headers
- Writing flexible asynchronous content handlers and filters

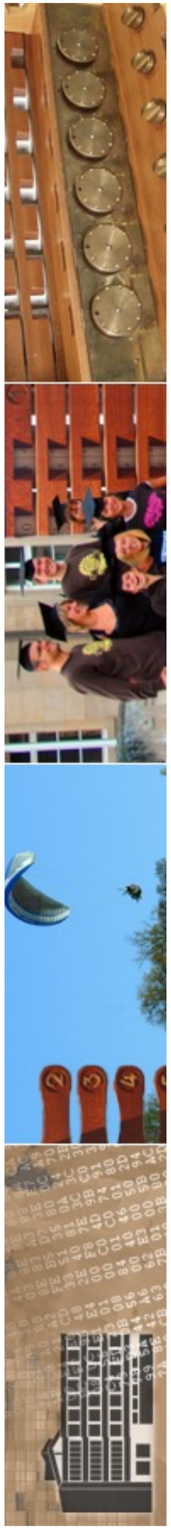
See examples and blog posts for more njs use cases.

english
русский

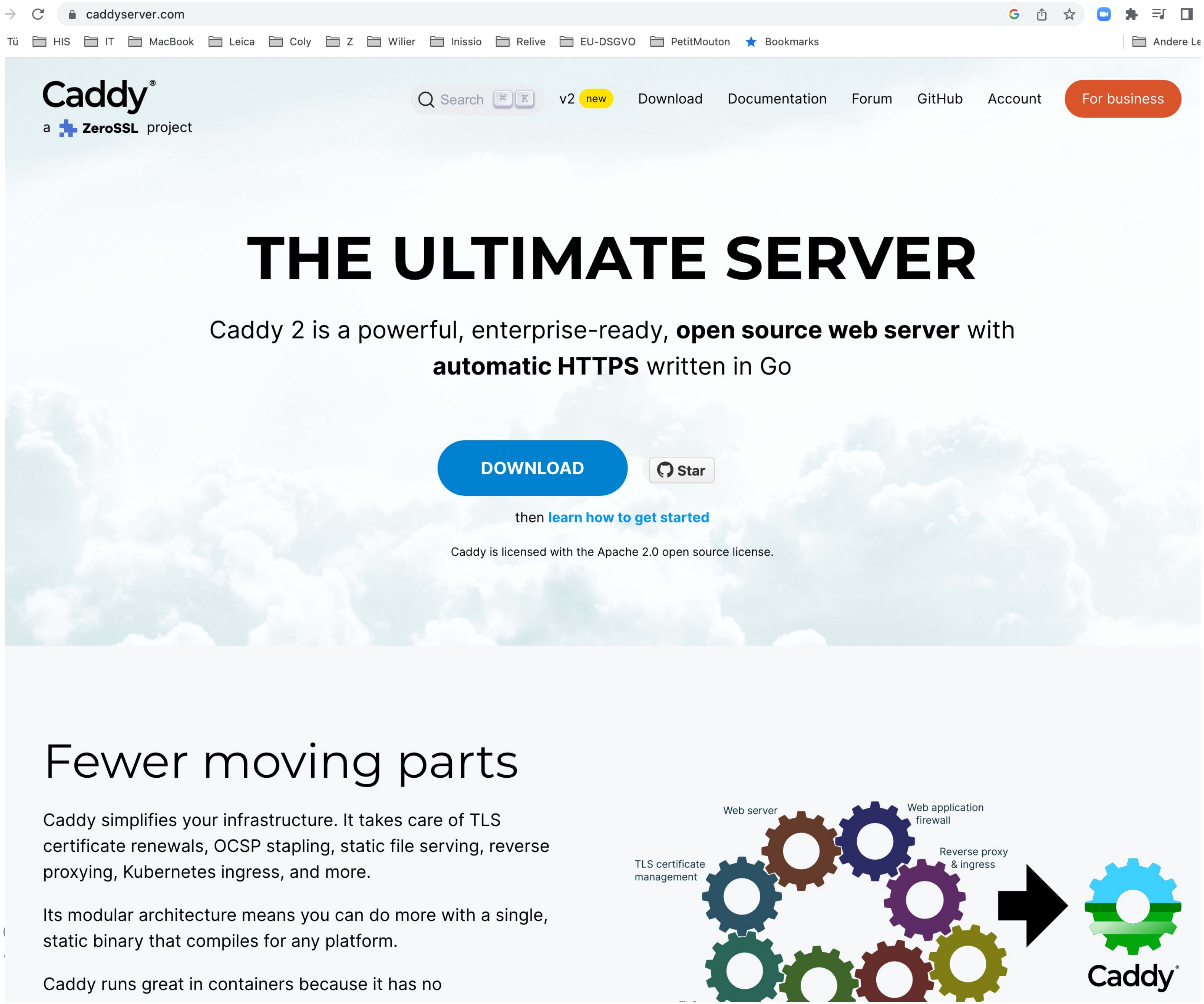
news
about
download
security
documentation
faq
books
support

trac
twitter
blog

unit
njs



Caddy



THE ULTIMATE SERVER

Caddy 2 is a powerful, enterprise-ready, **open source web server** with **automatic HTTPS** written in Go

then **learn how to get started**

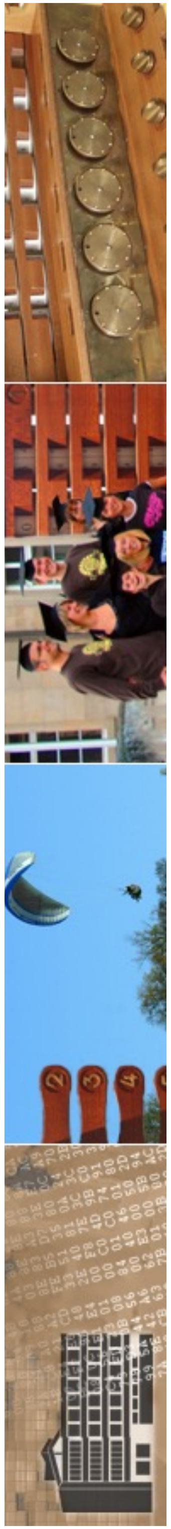
Caddy is licensed with the Apache 2.0 open source license.

Fewer moving parts

Caddy simplifies your infrastructure. It takes care of TLS certificate renewals, OCSP stapling, static file serving, reverse proxying, Kubernetes ingress, and more.

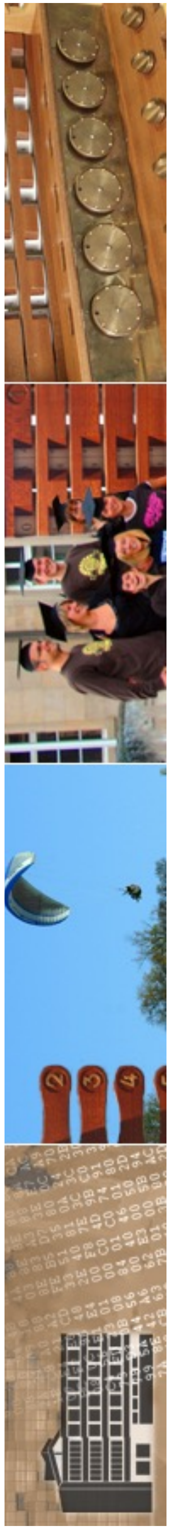
Its modular architecture means you can do more with a single, static binary that compiles for any platform.

Caddy runs great in containers because it has no



Caddy

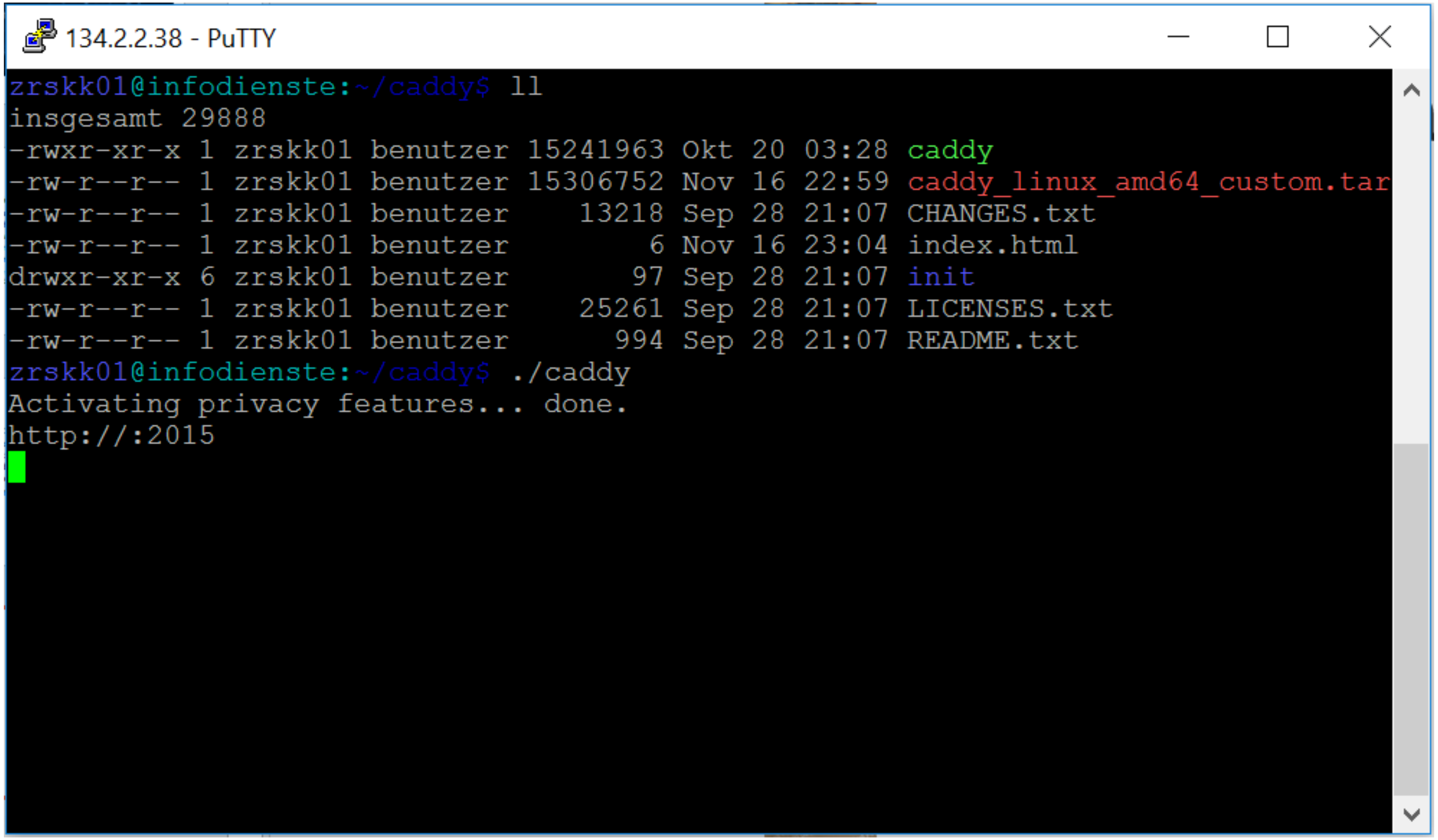
- Hauptsächlich für statische Inhalt
  - Kann auch Dynamik mit fastCGI
  - Einfache Nutzbarkeit
  - Automatische SSL Zertifikate mit Let’s Encrypt
  - Support von IPV6, HTTP/2
  - Performance für meiste Seiten ähnlich zu Nginx



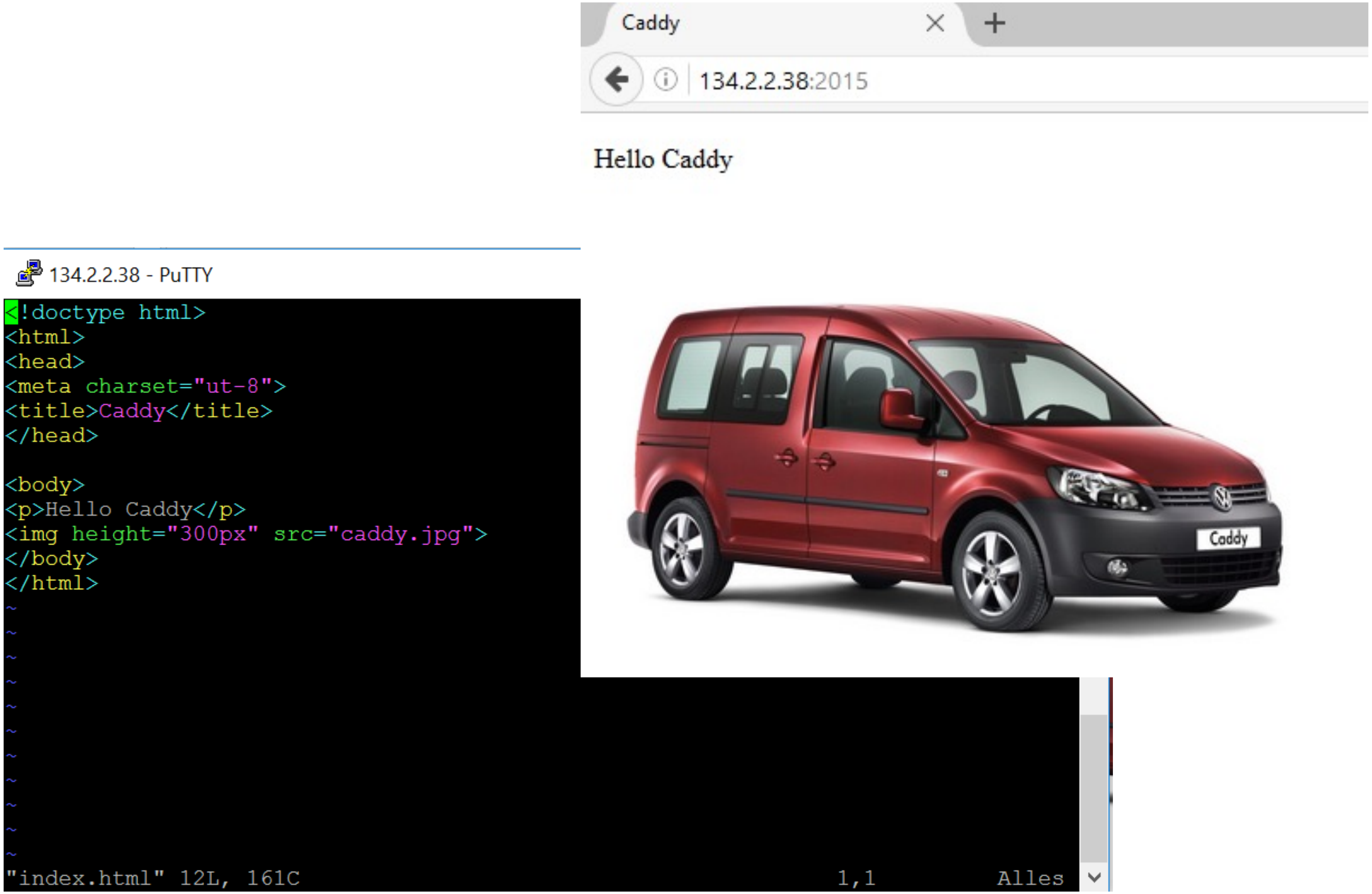
Caddy

- Installation maximal einfach
  - Download
  - Entpacken
  - ./caddy ausführen
  - Aufrufen der Seiten über localhost:2015
- Erweiterungen aktuell in Entwicklung

Caddy



```
zrskk01@infodienste:~/caddy$ ll
insgesamt 29888
-rwxr-xr-x 1 zrskk01 benutzer 15241963 Okt 20 03:28 caddy
-rw-r--r-- 1 zrskk01 benutzer 15306752 Nov 16 22:59 caddy_linux_amd64_custom.tar
-rw-r--r-- 1 zrskk01 benutzer    13218 Sep 28 21:07 CHANGES.txt
-rw-r--r-- 1 zrskk01 benutzer        6 Nov 16 23:04 index.html
drwxr-xr-x 6 zrskk01 benutzer       97 Sep 28 21:07 init
-rw-r--r-- 1 zrskk01 benutzer    25261 Sep 28 21:07 LICENSES.txt
-rw-r--r-- 1 zrskk01 benutzer      994 Sep 28 21:07 README.txt
zrskk01@infodienste:~/caddy$ ./caddy
Activating privacy features... done.
http://:2015
```

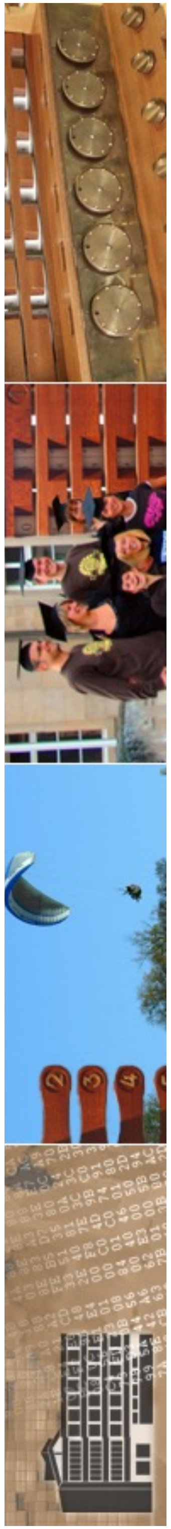


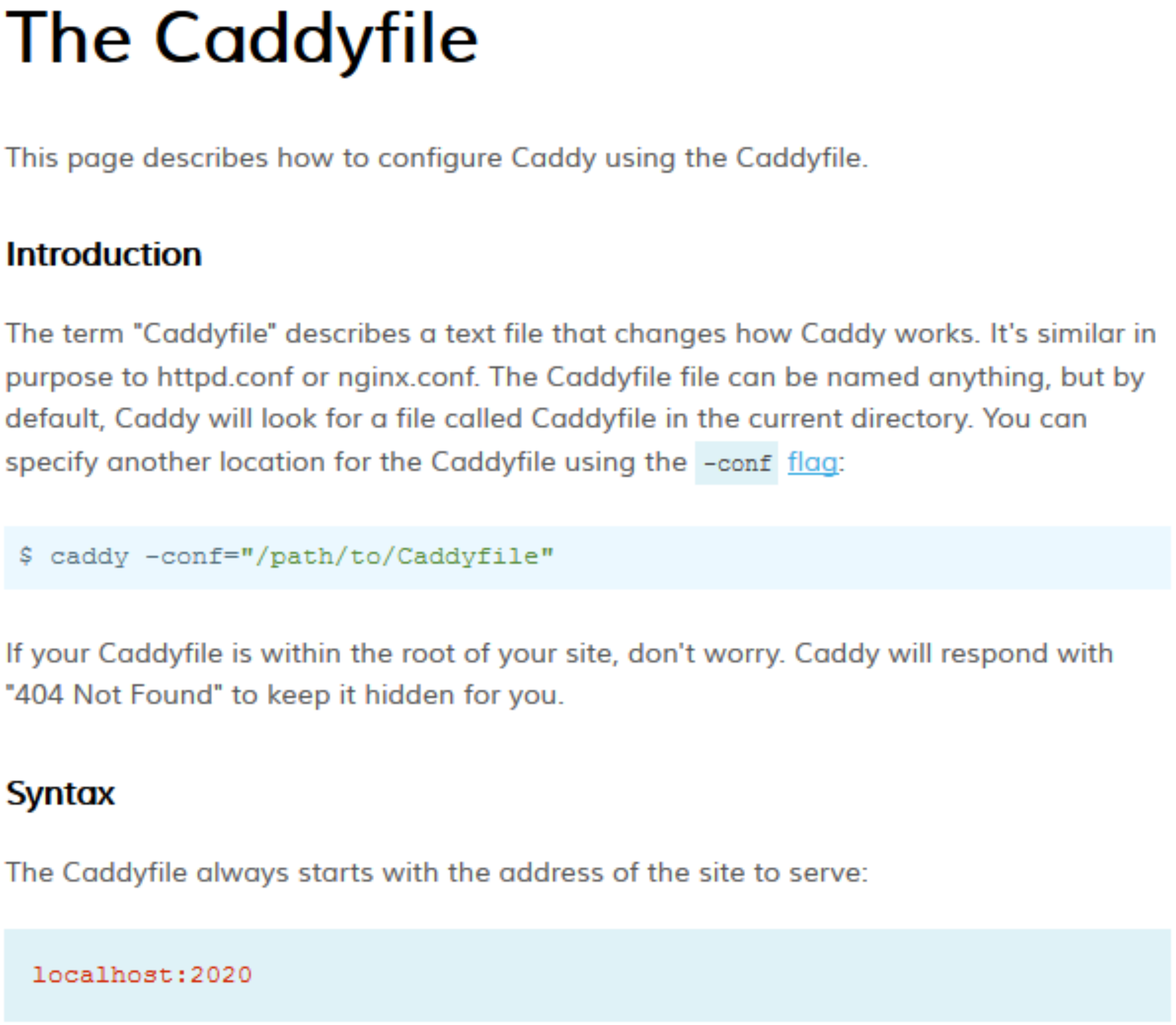
134.2.2.38:2015

Hello Caddy

134.2.2.38 - PuTTY

```
<!doctype html>
<html>
<head>
<meta charset="ut-8">
<title>Caddy</title>
</head>

<body>
<p>Hello Caddy</p>
<img height="300px" src="caddy.jpg">
</body>
</html>
~
~
~
~
~
~
~
~
~
~
~
"index.html" 12L, 161C                                        1,1          Alles
```





The Caddyfile

This page describes how to configure Caddy using the Caddyfile.

Introduction

The term "Caddyfile" describes a text file that changes how Caddy works. It's similar in purpose to httpd.conf or nginx.conf. The Caddyfile file can be named anything, but by default, Caddy will look for a file called Caddyfile in the current directory. You can specify another location for the Caddyfile using the `-conf` flag:

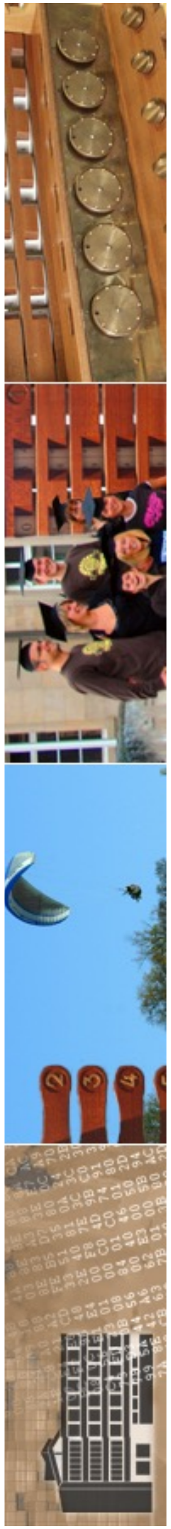
```
$ caddy -conf="/path/to/Caddyfile"
```

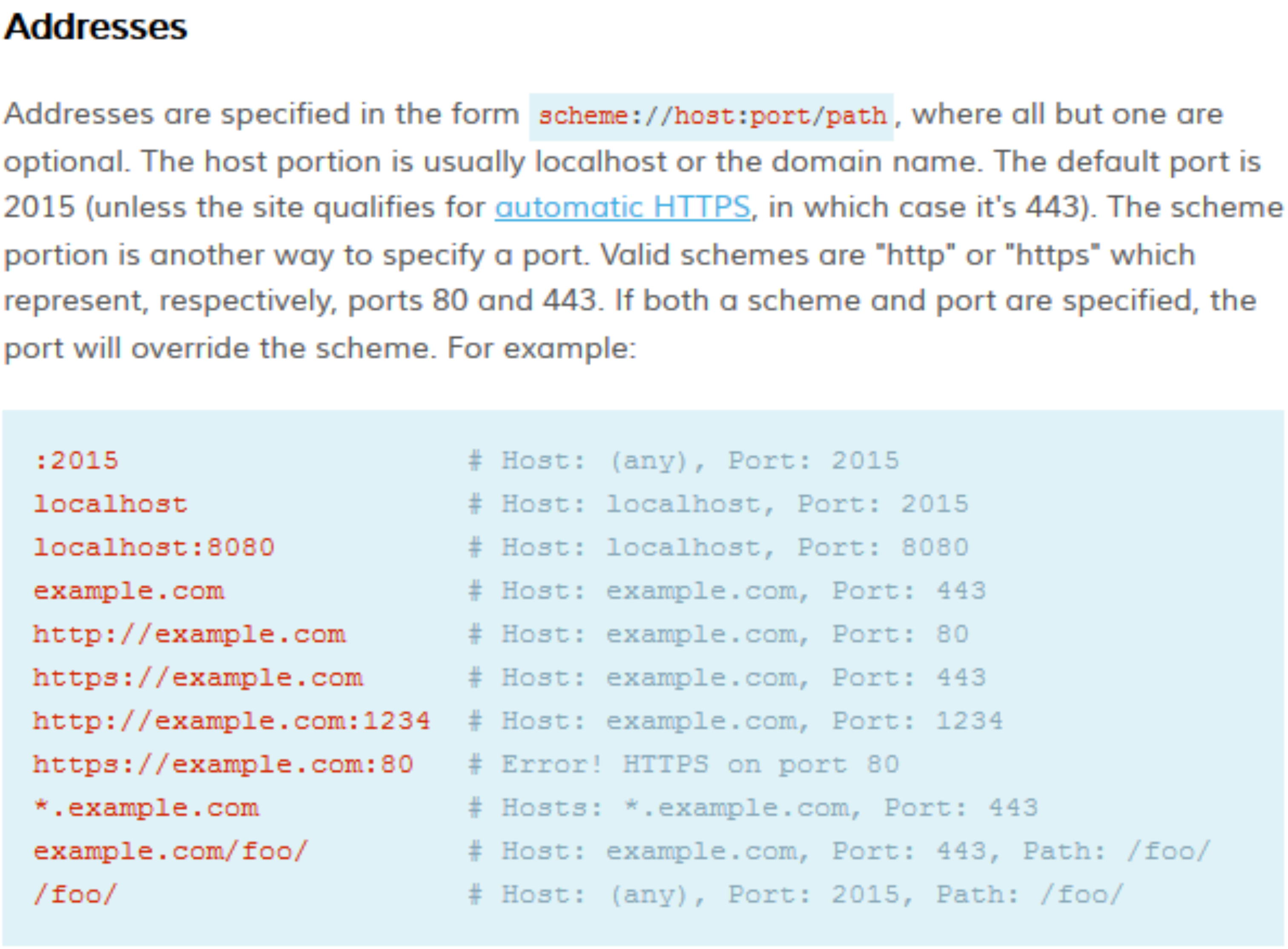
If your Caddyfile is within the root of your site, don't worry. Caddy will respond with "404 Not Found" to keep it hidden for you.

Syntax

The Caddyfile always starts with the address of the site to serve:

```
localhost:2020
```

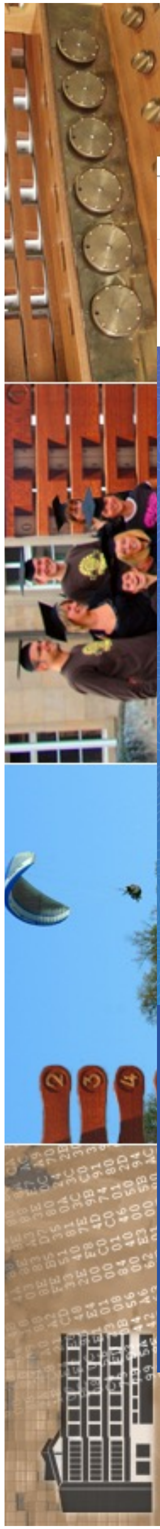


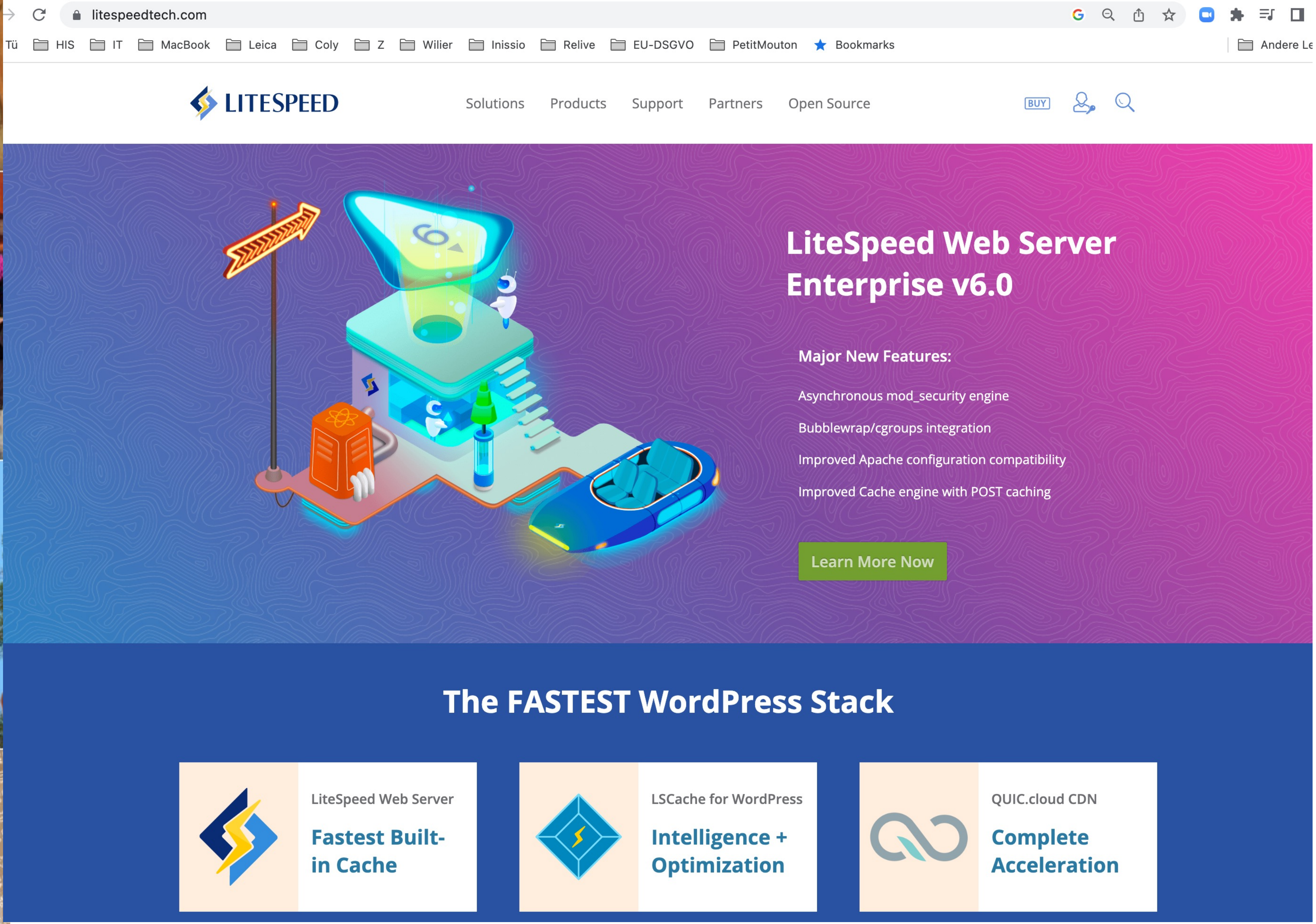


Addresses

Addresses are specified in the form `scheme://host:port/path`, where all but one are optional. The host portion is usually localhost or the domain name. The default port is 2015 (unless the site qualifies for automatic HTTPS, in which case it's 443). The scheme portion is another way to specify a port. Valid schemes are "http" or "https" which represent, respectively, ports 80 and 443. If both a scheme and port are specified, the port will override the scheme. For example:

```
:2015                    # Host: (any), Port: 2015
localhost                # Host: localhost, Port: 2015
localhost:8080           # Host: localhost, Port: 8080
example.com              # Host: example.com, Port: 443
http://example.com       # Host: example.com, Port: 80
https://example.com      # Host: example.com, Port: 443
http://example.com:1234  # Host: example.com, Port: 1234
https://example.com:80   # Error! HTTPS on port 80
*.example.com            # Hosts: *.example.com, Port: 443
example.com/foo/         # Host: example.com, Port: 443, Path: /foo/
/foo/                    # Host: (any), Port: 2015, Path: /foo/
```





LiteSpeed Web Server Enterprise v6.0

Major New Features:

Asynchronous mod_security engine

Bubblewrap/cgroups integration

Improved Apache configuration compatibility

Improved Cache engine with POST caching

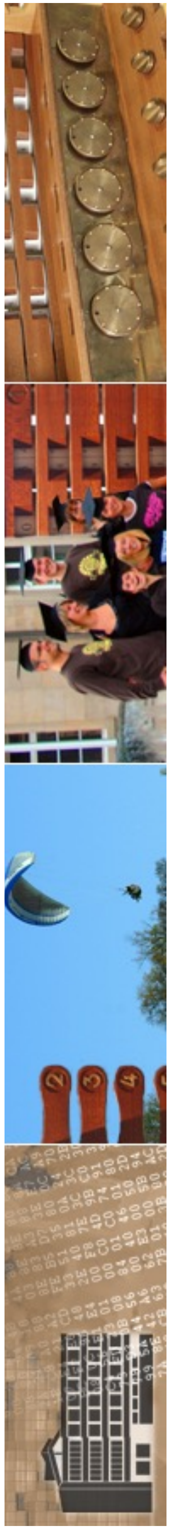
Learn More Now

The FASTEST WordPress Stack

LiteSpeed Web Server
Fastest Built-in Cache

LSCache for WordPress
Intelligence + Optimization

QUIC.cloud CDN
Complete Acceleration



…und nun…

- haben wir zwei aktuelle, sehr performante Web-Server kennen gelernt:
  nginx und Caddy

- als nächstes:
  fastCGI und ServerSideIncludes