

# Webový server Apache

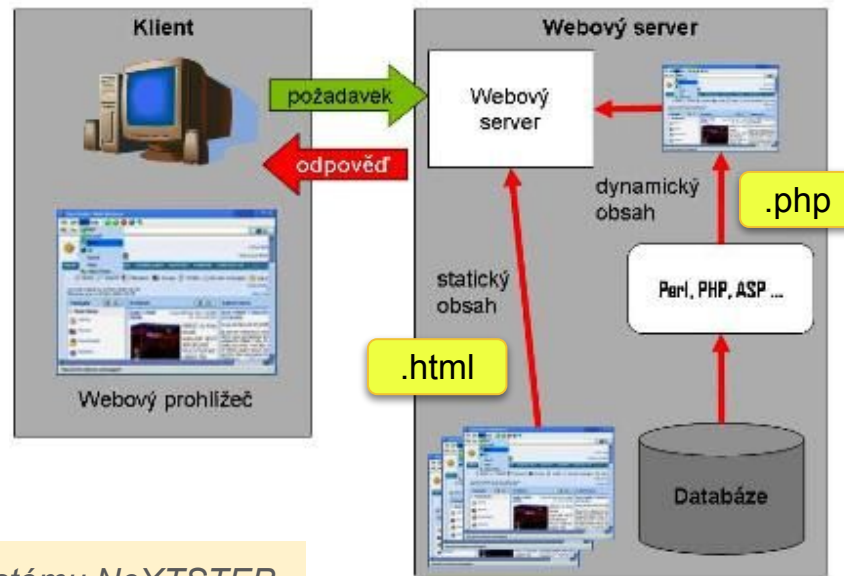
Instalace, kontrola, konfigurace

# Webový server

Počítač připojený k počítačové síti.

Přijímá a vyřizuje požadavky na určených portech ve tvaru HTTP. Počítač, který požadavek vznesl, vrací odpověď opět ve tvaru HTTP:

1. **Hlavičku** obsahující mj. **stavový kód**,
2. **Samotný obsah** (HTML dokument, text, obrázek apod.).



První webový server na světě byl **CERN httpd** a běžel na systému **NeXTSTEP** (předchůdce macOS). Byl spuštěn v roce **1989** a hostoval tento obsah:  
<http://info.cern.ch/hypertext/WWW/TheProject.html>

# Webové servery

Webových serverů je víc, např.

- **Apache HTTP Server**
- **Nginx Web Server**
- **GWS** (Google Web Server)
- **IIS** (Microsoft Internet Information Services)
- **Lighttpd** (Light httpd)
- **SJSAS** (Sun Java System Web Server)
- **Small HTTP server**

a asi 15 [dalších](#).



©sanchit0496

# Apache HTTP Server



TM

My budeme používat [Apache](#), konkrétně jeho **druhou verzi**.

Apache je velmi konzervativní software, kde nové věci přibývají velmi málo.  
Ve verzi 2.4 je už od roku 2012.

Jeho funkcionalita se rozšiřuje **zejména pomocí modulů** (*mods*).

# Apache HTTP Server



Apache HTTP Server je softwarový webový server s [otevřeným kódem](#) pro Linux, BSD, UNIX, macOS i Windows.

Jméno se podle **jedné z verzí** odvozuje od "**a patchy server**" (v překladu "záplatovaný server"). To odráží skutečnost, že Apache původně vznikl jako sada oprav (patchů) a rozšíření pro existující servery HTTPd.

Domovská stránka projektu [httpd.apache.org/](http://httpd.apache.org/)

Vlastnosti:

- bezpečný
- rozšiřitelný pomocí modulů
- konfigurovatelný
- "synonymum" pro webový server

# Instalace

Instalaci webového serveru Apache provedeme příkazem

```
$ sudo apt update
```

```
$ sudo apt install apache2
```

Server Apache se po instalaci automaticky spustí jako **služba** (*démon*).

*Služba je jednoduše aplikace, která v počítači trvale běží.*

# Porty

Webový server naslouchá na portu

- **80** (protokol **HTTP**)
- nebo **443** (protokol **HTTP** s šifrovací podvrstvou SSL čili **HTTPS**). Před odesláním se data zašifrují a teprve poté odešlou protokolem HTTP.

Port 443 si nevyzkoušíme – instalace SSL certifikátu vyžaduje mít na webu funkční doménu (example.com) a tu my nemáme.

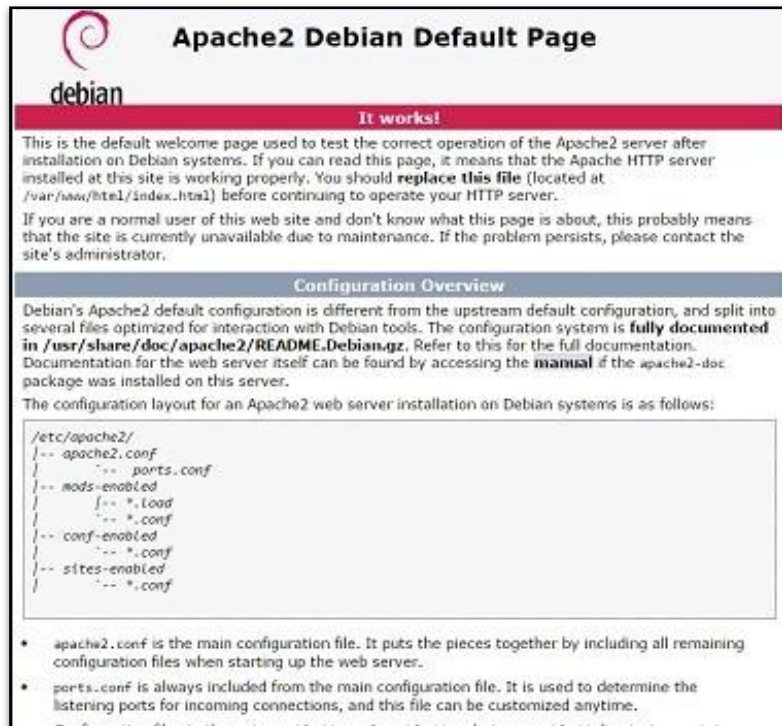
# Adresáře a soubory

V Debianu je výchozím adresářem pro webový obsah:

```
/var/www/html/
```

V něm se nachází soubor `index.html` s ukázkovou webovou stránkou s užitečnými informacemi:

```
/var/www/html/index.html
```



**Apache2 Debian Default Page**

It works!

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Debian systems. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should **replace this file** (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

**Configuration Overview**

Debian's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Debian tools. The configuration system is **fully documented in `/usr/share/doc/apache2/README.Debian.gz`**. Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the **manual** if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Debian systems is as follows:

```
/etc/apache2/  
|-- apache2.conf  
|   |-- ports.conf  
|-- mods-enabled  
|   |-- *.Load  
|   |-- *.conf  
|-- conf-enabled  
|   |-- *.conf  
|-- sites-enabled  
|   |-- *.conf
```

- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.

# Stavové kódy

Stavový kód odpovědi udává, zda byl požadavek vyřízen v pořádku, či zda došlo k nějakým problémům. Kódy jsou trojčíselná čísla a dělí se do pěti skupin:

- **1xx Informační**
- **2xx Success**
  - **200 OK:** Standardní odpověď pro úspěšný HTTP požadavek.
- **3xx Redirect**
- **4xx Client Error**
  - **404 Not Found:** Požadovaný zdroj nebyl nalezen.
  - **403 Forbidden:** Uživatel nemá oprávnění k přístupu k požadovanému zdroji.
- **5xx Server Error**

# Kontrola funkčnosti webového serveru

Jde provést několika způsoby:

**Terminálem** (zjistíme různé informace):

```
$ systemctl status apache2 – vypíše stav služby
```

```
$ curl -I http://192.168.23.nnn – vypíše pouze hlavičky odpovědi
```

```
$ nmap 192.168.23.nnn – vypíše otevřené porty (protokol http je 80)
```

**Webovým prohlížečem:**

```
http://192.168.23.nnn .. zobrazí obsah souboru index.html
```

# Protokolování požadavků

Apache protokoluje přijímané požadavky a taktéž zaznamenává případné chyby. To pomáhá správcům vytvářet statistiky a podle typu a množství požadavků optimalizovat obsah, způsob uložení i způsob prezentace požadovaných dat.

```
$ sudo tail /var/log/apache2/access.log  
$ sudo /var/log/apache2/error.log
```

Instantní výpis posledních 10 řádků (tam jsou ty nejnovější záznamy)

aktivuješ přepínačem `-f`. Zobraz si:

```
$ sudo tail -f /var/log/apache2/access.log
```

a několikrát aktualizuj webovou stránku. Sleduj výpis. Co vše se v něm zobrazuje?  
Výpis ukončíš `Ctrl+C`.

# Další příkazy pro řízení běhu

```
$ sudo systemctl stop|start|restart|reload apache2
```

- stop – Zastaví webový server.
- start – Spustí webový server.
- restart – Restartuje webový server (hard restart).
- reload – Načte změněnou konfiguraci (soft restart).

Pro úplnost: Ve výchozím nastavení je Apache nakonfigurovaný na automatické spuštění po startu serveru. Toto chování jde změnit:

- disable – Po restartu nespustí webový server.
- enable – Zpět na automatické spuštění.

# Konfigurace modulů a webů

# VirtualHost

VirtualHost je konfigurace v Apache, která umožňuje **hostování (provozu) více webových stránek na jednom serveru.**

Každý virtuální host je přiřazen k určité doméně nebo IP adrese a specifikuje, jaký obsah se má server při návštěvě této domény nebo adresy načíst.

# Modelový příklad

mějme dvě domény [hrnce.cz](http://hrnce.cz) a [poklicky.cz](http://poklicky.cz). V konfiguraci Apache vytvoříme pro každou doménu **samostatný VirtualHost**, který bude směřovat na **jiný adresář**, i když oba weby běží na stejné IP adrese. Weby budou dostupné na

- [hrnce.cz](http://hrnce.cz) / [www.hrnce.cz](http://www.hrnce.cz)
- [poklicky.cz](http://poklicky.cz) / [www.poklicky.cz](http://www.poklicky.cz)

Takto to zcela **běžně provozují poskytovatelé webhostingů**. Bylo by naivní se domnívat, že každý web má svůj dedikovaný (vyhrazený) server.

Existuje i opačný případ: jeden web je provozován na mnoha serverech (facebook.com, instagram.com, google.com...)

# Konfigurace pro hrnce.cz

Soubor `/etc/apache2/sites-available/hrnce.conf`

```
<VirtualHost *:80>
    ServerName hrnce.cz
    ServerAlias www.hrnce.cz
    ServerAdmin webmaster@hrnce.cz
    DocumentRoot /var/www/hrnce

    <Directory /var/www/hrnce>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/hrnce_error.log
    CustomLog ${APACHE_LOG_DIR}/hrnce_access.log combined
</VirtualHost>
```

## Options

- Určuje, co je v daném adresáři povoleno.
- **Indexes** – pokud není nalezen výchozí soubor (např. index.html), Apache zobrazí seznam souborů v adresáři.
- **FollowSymLinks** – Apache může následovat symbolické odkazy (symlinky) a zpřístupnit tak jejich cíle.

## AllowOverride

- Povoluje používání soubor **.htaccess** v adresáři.
- **All** znamená, že všechny typy nastavení (např. přesměrování, přepisování URL, přístupová práva) mohou být v **.htaccess** souboru měněny.
- Pokud by byl **None**, **.htaccess** by se ignoroval.

## Require all granted

- Přístup k danému adresáři je povolen všem klientům (bez omezení IP nebo oprávnění).

## Proměnná `${APACHE_LOG_DIR}`

- definuje cestu `/var/log/apache2`. Proměnná je definována v hlavním konfig. souboru Apache: `/etc/apache2/envvars`.

# Konfigurace pro poklicky.cz

Soubor `/etc/apache2/sites-available/poklicky.conf`

```
<VirtualHost *:80>
    ServerName poklicky.cz
    ServerAlias www.poklicky.cz
    ServerAdmin webmaster@poklicky.cz
    DocumentRoot /var/www/poklicky

    <Directory /var/www/poklicky>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/poklicky_error.log
    CustomLog ${APACHE_LOG_DIR}/poklicky_access.log combined
</VirtualHost>
```

# Záznamy v DNS

⚠ Ostrý provoz virtualhostů by vyžadoval **vlastnictví veřejné IP adresy se záznamem v DNS**. Ten by vypadal nějak takto:

```
hrnce.cz      A      210.147.113.98
www          CNAME  hrnce.cz
hrnce.cz     AAAA   2001:0db8:85a3::8a2e:0370:7334

poklicky.cz  A      210.147.113.98
www          CNAME  poklicky.cz
poklicky.cz  AAAA   2001:0db8:85a3::8a2e:0370:7334
```

# Výchozí VirtualHost

Soubor: `/etc/apache2/sites-available/000-default.conf`

Obsah souboru:

```
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

## Praktický příklad

V této variantě nepoužíváme VirtualHosty ani aliasy v konfiguraci Apache. Každý „web“ je pouze podadresář ve výchozím DocumentRoot: `/var/www/html/`

Tím vznikne jednoduchý způsob, jak provozovat více webů:

`http://192.168.21.nnn/bylinky.cz`

`http://192.168.21.nnn/caje.cz`

Každý z nich odpovídá obsahu adresáře:

`/var/www/html/bylinky.cz/`

`/var/www/html/caje.cz/`

# Jak Apache zpracuje URL

Apache má výchozí nastavení:

```
DocumentRoot /var/www/html
```

**To znamená:** Název adresáře (`bylinky.cz`, `caje.cz`) nemá žádnou speciální funkci – je to pouze cesta v URL. Stejně tak nemá "doména" `.cz` žádný význam.

To znamená:

`http://192.168.21.nnn/` → `/var/www/html/`

`http://192.168.21.nnn/bylinky.cz` → `/var/www/html/bylinky.cz/`

`http://192.168.21.nnn/caje.cz` → `/var/www/html/caje.cz/`

# Jak Apache zpracuje URL

Název adresáře (`bylinky.cz`, `caje.cz`) nemá žádnou speciální funkci – je to pouze cesta v URL.

- **Apache neřeší doménu**, protože žádný VirtualHost není definován.
- Vše jde do jednoho **defaultního hostu** a rozhoduje **jen cesta** (path).

Příklad požadavku:

```
GET /bylinky.cz HTTP/1.1
Host: 192.168.21.nnn
```

Apache tedy hledá:

```
/var/www/html/bylinky.cz/
```

- Pokud obsah existuje → zobrazí se.
- Pokud ne → 404 Not Found.

# Adresářová struktura webů

Chceme mít provozované tyto „weby“:

```
http://192.168.21.nnn/bylinky.cz
```

```
http://192.168.21.nnn/caje.cz
```

Pak adresáře musí být:

```
/var/www/html/bylinky.cz
```

```
/var/www/html/caje.cz
```

Uvnitř každého musí být alespoň `index.html`, např.:

```
<h1>bylinky.cz</h1>
```

```
<p>Vítejte na webu bylinky.cz</p>
```

# Oprávnění

Adresáře i jejich obsah musí být **čitelný pro Apache**, vlastníkem bude **uživatel**.

-R = rekurzivně

```
$ sudo chown -R user:www-data /var/www/html
```

# Skript

Naklonuj Git repozitář se dvěma skripty:

```
$ cd  
$ git clone https://github.com/edumach/site  
$ cd site  
$ chmod +x *.sh  
$ ls -l
```

# Vytvářecí skript `create_site.sh`

Skript `create_site.sh` automatizuje:

1. vytvoření adresáře
2. vytvoření souboru `index.html`
3. nastavení správných oprávnění
4. vypsání URL, kde bude web dostupný

Nepoužívá se žádné nastavování VirtualHostů, žádné aliasy ani úpravy Apache.

```
$ sudo ./create_site.sh
```

# Mazací skript **delete\_site.sh**

Skript `delete_site.sh`:

1. smaže vybraný podadresář webu
2. nesahá na žádné konfigurační soubory Apache
3. Apache není nutné reloadovat (nic se nezměnilo v konfiguraci)

Skript pouze odstraní požadovaný adresář:

```
/var/www/html/<web>
```

# Shrnutí

- Nepoužíváme VirtualHosty, každý web je **pouze adresář** ve `/var/www/html`.
- Cesta v URL přesně odpovídá adresáři na disku.
- Konfigurace Apache se nemění.
- Tento koncept se běžně používá, viz např. <https://edumach.cz/cokolada/>

`http://<server>/bylinky.cz` → `/var/www/html/bylinky.cz/`

`http://<server>/caje.cz` → `/var/www/html/caje.cz/`

Jde o nejjednodušší možné řešení pro naše školní prostředí při kterém nepotřebujeme žádné zásahy do Apache.

`.cz` je **jen součást názvu**. Web `bylinky.cz` bude fungovat úplně stejně pod název `bylinky`.