HTTP szolgáltatások

A HTTP alapjai

A HTTP (HyperText Transfer Protocol) egy alkalmazás réteg protokoll, amely kliens–szerver modellben működik. Elsősorban webes erőforrások (HTML, CSS, JS, képek, API válaszok stb.) továbbítására használják.

Alapvető jellemzők:

- Állapotmentes (stateless)
- Kérés-válasz alapú (request–response)
- TCP (HTTP/1.x, HTTP/2) vagy QUIC/UDP (HTTP/3) felett működik
- Szöveges (HTTP/1.x) illetve bináris (HTTP/2, HTTP/3)

HTTP működési modell

A kommunikáció menete:

Kliens kapcsolatot nyit a szerverhez

Kliens HTTP kérést küld

Szerver HTTP választ küld

Szerver lezárja a kapcsolatot (HTTP ⇐ 1.0) vagy nyitva marad ha szükséges (HTTP >= 1.1)

Egyszerű HTTP 1.0 kérés (https://datatracker.ietf.org/doc/html/rfc1945):

```
```

Egyszerű kérés felépítése:

```
# telnet szerver 80
GET /index.html HTTP/1.1
Host: example.com
User-Agent: curl/8.0 Accept: */*
```

Szerver válasza:

```
 HTTP/1.1 200 OK Content-Type: text/html Content-Length: 1234
<html>...</html>
```

HTTP/0.9 (1991)

A HTTP/0.9 volt a legelső, kísérleti változata a protokollnak, amelyet Tim Berners-Lee fejlesztett ki a

CERN-ben a World Wide Web kezdeti időszakában.

Jellemzői:

Rendkívül egyszerű

Csak egyetlen metódus létezett: GET

Nem léteztek HTTP headerek

Nem volt státuszkód

Nem volt verziószám a kérésben

A válasz kizárólag HTML tartalom volt

Kérés formátuma

```
GET /index.html
```

Fontos:

Nincs HTTP/1.0 vagy HTTP/1.1 jelölés

Nincs Host header

Üres sor sem szükséges

Válasz

A szerver egyszerűen visszaküldi a HTML dokumentumot:

```
<html> <body>Hello World</body> </html>
```

Nincs:

státuszsor

header mező

Content-Type

Content-Length

Korlátok

Nem támogatta a képeket, CSS-t, külön erőforrásokat

Nem volt hibakezelési mechanizmus

Nem alkalmas komplex webalkalmazásokhoz

Jelentősége

A HTTP/0.9 kizárólag dokumentumlekérésre szolgált. Ez a minimalista modell képezte az alapját a későbbi HTTP/1.0 verziónak, amely már bevezette:

státuszkódokat

header mezőket

MIME típusokat

Összegzés

A HTTP/0.9 egy rendkívül egyszerű, csak GET-et támogató, header nélküli protokoll volt, amely kizárólag HTML dokumentum visszaküldésére szolgált.

Ez tekinthető a modern webkommunikáció kiindulópontjának.

HTTP/1.0 (1996)

Jellemzők:

Minden kérés külön TCP kapcsolat

Nincs alapértelmezett persistent connection

Egyszerű szöveges protokoll

Nincs Host header kötelezően

Problémák:

Sok TCP handshake

Magas latency

Nem hatékony párhuzamos erőforrás-letöltés

HTTP/1.1 (1997)

Fejlesztések:

Persistent connection (Keep-Alive)

Host header kötelező

Chunked transfer encoding

Pipeline támogatás

Példa persistent kapcsolatra:

```
Connection: keep-alive
```

Előny:

Kevesebb TCP kapcsolat

Jobb teljesítmény

Probléma:

Head-of-line blocking (pipeline esetén)

HTTP/2 (2015)

Alapja: SPDY protokoll

Fő jellemzők:

Bináris protokoll

Multiplexing (több stream egy TCP kapcsolaton)

Header compression (HPACK)

Server Push

Előny:

Megszűnik az alkalmazás-szintű head-of-line blocking

Jelentősen jobb teljesítmény

Hátrány:

TCP head-of-line blocking továbbra is fennáll csomagszinten

HTTP/3 (2022)

Alapja: QUIC (UDP felett)

Jellemzők:

TCP helyett UDP

TLS 1.3 integrált

Nincs TCP head-of-line blocking

Gyorsabb kapcsolatfelépítés (0-RTT)

Előny:

Jelentősen alacsonyabb latency

Mobil hálózatokon stabilabb

HTTP verziók összehasonlítása

Verzió	Szállítási réteg	Formátum	Multiplexing	Head-of-line blocking
HTTP/1.0	TCP	Szöveges	Nem	Igen
HTTP/1.1	TCP	Szöveges	Korlátozott (pipeline)	Igen
HTTP/2	TCP	Bináris	Igen	TCP szinten igen
HTTP/3	QUIC (UDP)	Bináris	Igen	Nem

Telnet-es HTTP példa (HTTP/1.0)

Kapcsolódás:

```
telnet example.com 80
```

Kérés kézi begépelése:

```
GET / HTTP/1.0
```

(Fontos: az üres sor zárja a header részt.)

Telnet-es HTTP/1.1 példa

```
telnet example.com 80
```

```
GET / HTTP/1.1 Host: example.com Connection: close
```

Megfigyelhető:

Kötelező Host header

Válasz státuszkód

Header mezők

Üres sor

Body

HTTP státuszkódok

Tartomány	Jelentés
1xx	Információ
2xx	Siker
3xx	Átirányítás
4xx	Kliens hiba
5xx	Szerver hiba

Példák:

200 OK

301 Moved Permanently

400 Bad Request

404 Not Found

500 Internal Server Error

HTTPS

A HTTPS nem külön protokoll, hanem:

HTTP + TLS titkosítás

Alapértelmezett port:

80 → HTTP

443 → HTTPS

Telnet helyett HTTPS teszteléshez:

```
openssl s_client -connect example.com:443
```

Összegzés

HTTP/1.x: egyszerű, szöveges, TCP-alapú

HTTP/2: bináris, multiplexelt, gyorsabb

HTTP/3: QUIC-alapú, modern, alacsony latency

Telnet kiváló eszköz a HTTP/1.x működés szemléltetésére

From:
https://wiki.r-l.hu/ - **Reverse-Logic wiki**

Permanent link:
https://wiki.r-l.hu/doku.php?id=webszerver&rev=1771125436

Last update: **2026/02/15 03:17**