

---

# A webes kommunikáció fejlődése és modern protokollok működése

HTTP/1.1 → HTTP/2 → HTTP/3

---

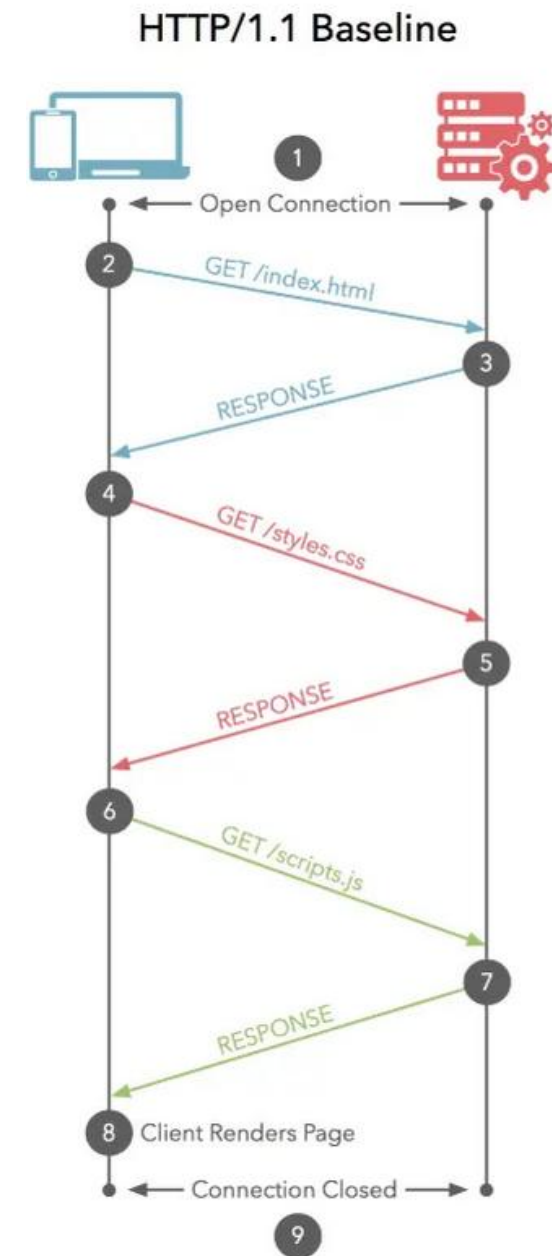
---

# Miért kellett fejlődni a HTTP-nek?

- Weboldalak **komplexebbé** váltak
- **Sok erőforrás:** JS, CSS, képek, API hívások
- **Mobil kliensek:** nagy késleltetés, instabil hálózat
- HTTP/1.1 eredeti tervezési céljai már nem voltak elegendők

# HTTP/1.1 működése röviden

- TCP átviteli protokollra épül
- **Sorrendkényszeres** protokoll
- **Persistent Connections** (Keep-Alive)
- Pipelining (papíron létezett, de gyakorlatban nem használták)
- **Head-of-Line Blocking** (HOL) az alkalmazási rétegben
- 6 kapcsolatos limit böngészőnként egy domain-re (domain sharding)



---

# HTTP/1.1 teljesítménykorlátai

- **Sorrendkényszer** miatt nagyon érzékeny a csomagvesztésre
- **Több TCP kapcsolat** → nagy overhead
- **HOL blocking** problémája nem kerülhető meg
- A protokoll nem tudja priorizálni a kéréseket

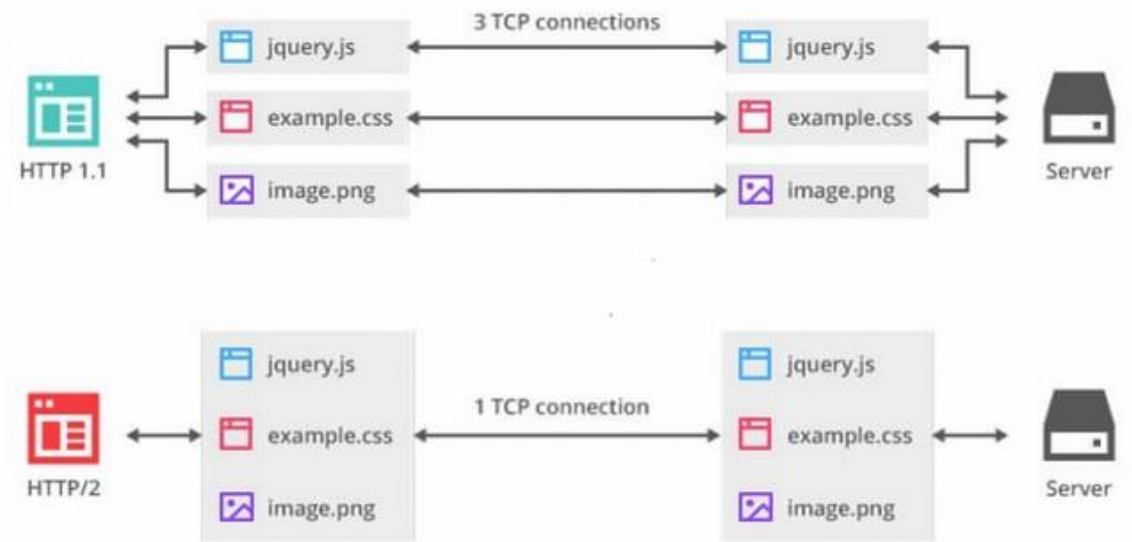
---

# Mi hozta el a HTTP/2 fejlesztését?

- A **multiplexing** iránti igény
- Oldalak **komplexitása**
- Gyorsabb oldalbetöltés
- **Kevesebb (egyetlen) TCP** kapcsolat
- Mobil hálózatok megjelenése
- SPDY („speedy”) Google kísérleti protokollja alapozta meg

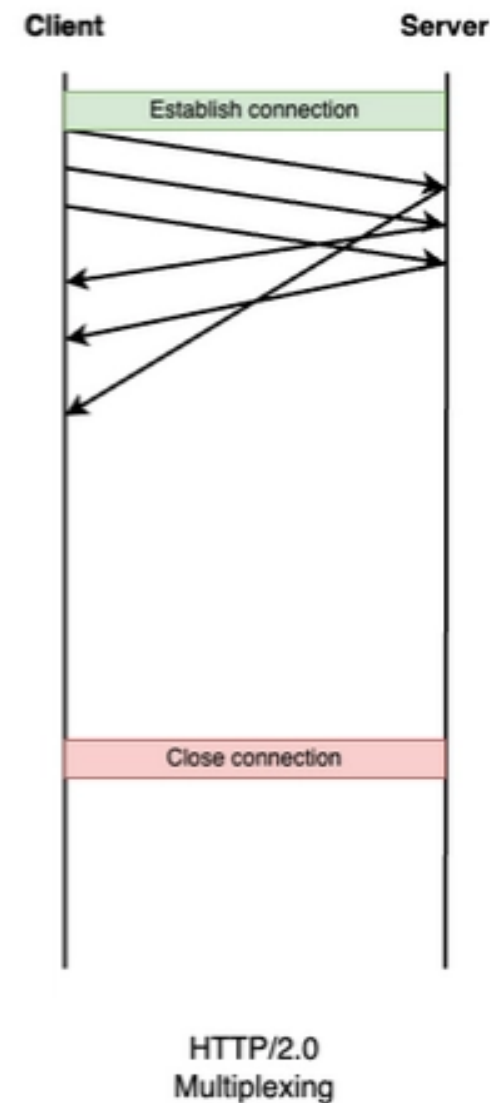
# HTTP/2: Bináris protokoll

- A HTTP/2 **bináris** protokoll
- Kérések és válaszok **keretekre (frames)** oszlanak
- **Stream** alapú kommunikáció
- Külön stream-ek **egyetlen TCP** kapcsolaton

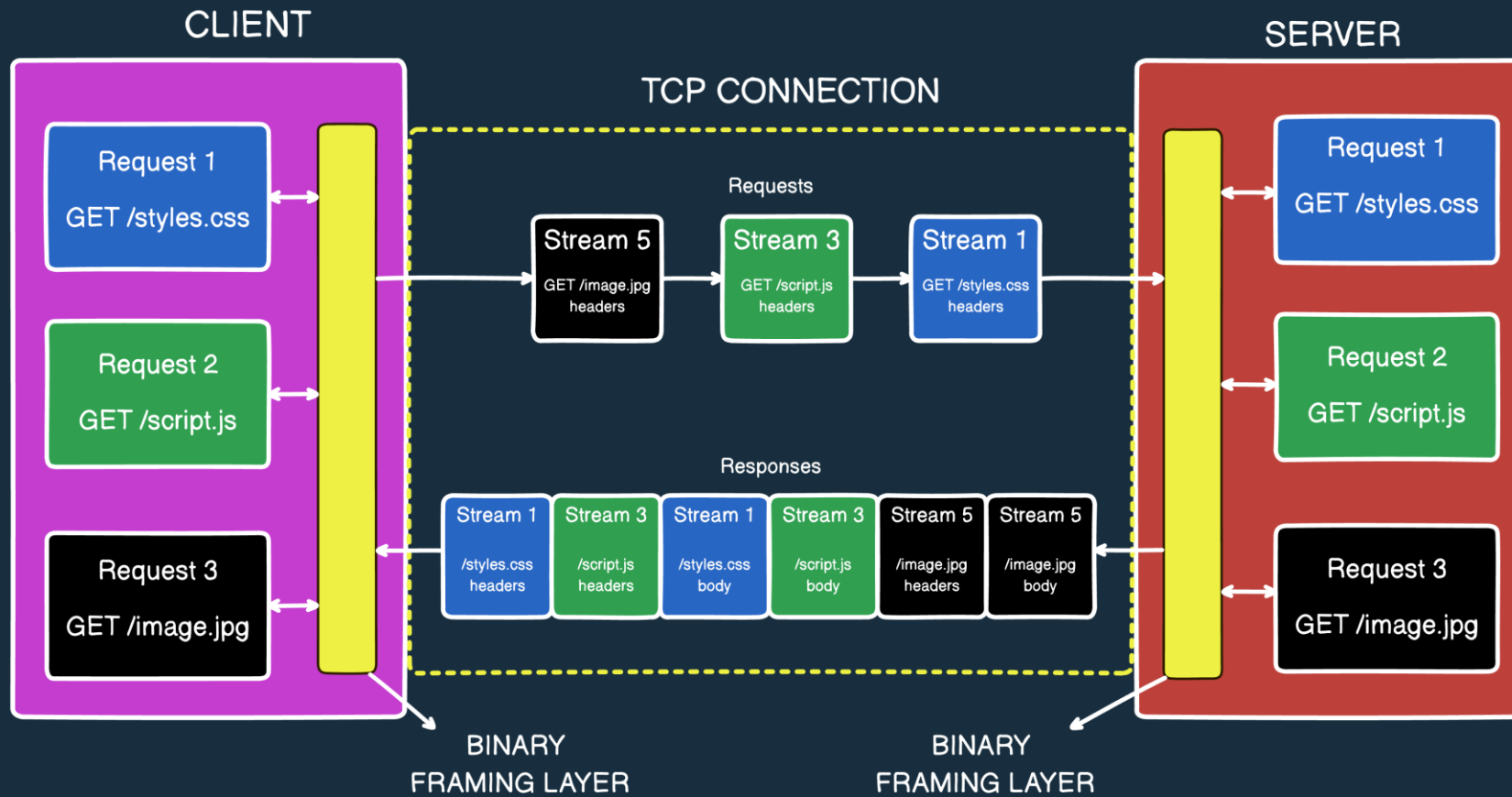


# HTTP/2 Multiplexing

- **Több kérés és válasz** párhuzamosan
- A böngésző **nem blokkolódik**
- **Megszűnik a HOL blocking** a HTTP protokoll szinten
- Nincs szükség több TCP kapcsolatra
- **Egy TCP kapcsolat**, sok stream
- Minden stream kap egy azonosítót (Stream ID)



# Multiplexing in HTTP/2





---

# HTTP/2 egyéb újdonságai

- **HPACK:** Header Compression
- HPACK **tömöríti a fejléceket**, ezáltal jelentős sávszélességcsökkenés
- **Stream prioritások:** böngésző közölheti, mi fontosabb
- **Server Push:** előre küldött erőforrások

---

# HTTP/2 hátránya: TCP HOL blocking

- A HTTP-szintű HOL blocking ugyan megszűnt
- De a **TCP-szintű HOL** még mindig fennáll
- **Csomagvesztés** továbbra is lassítja az egész kapcsolatot
- Egy elveszett csomag → minden stream lassul

---

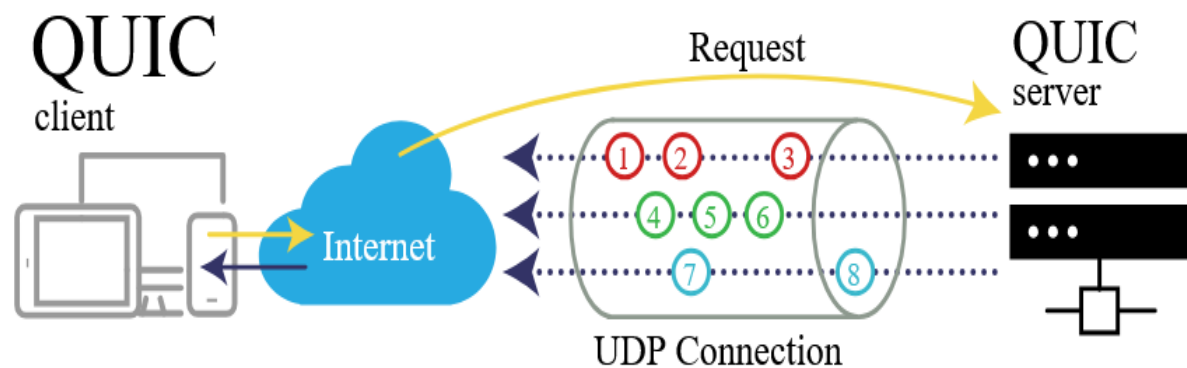
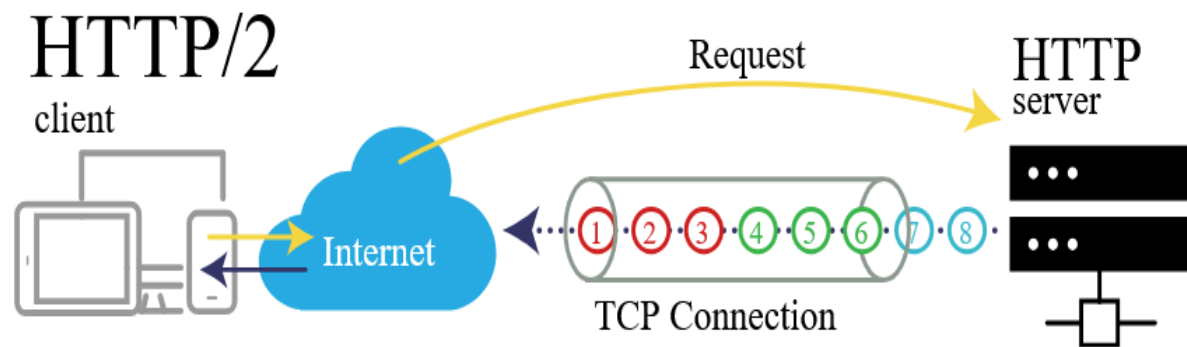
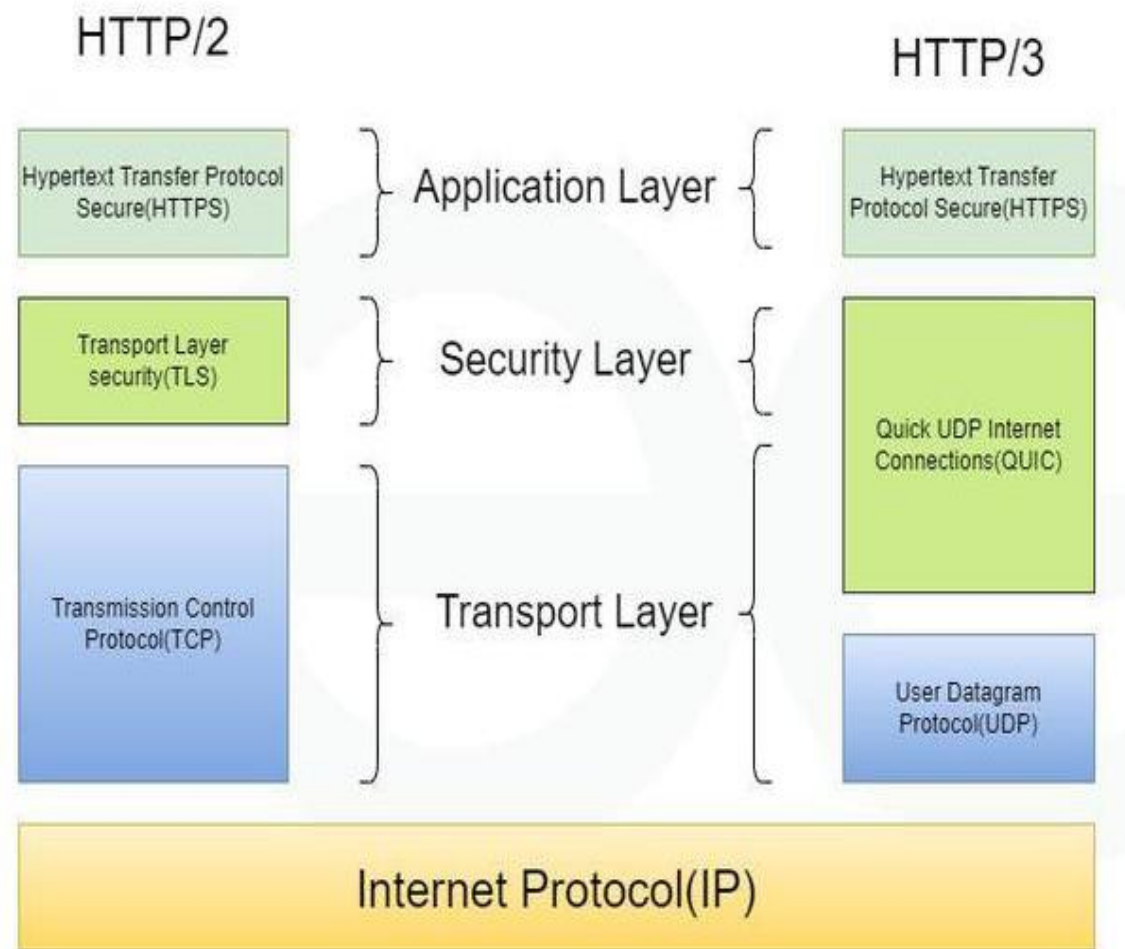
# HTTP/3 és a QUIC protokoll

- **UDP** alapú protokoll, ami csak „hordozó”
- Google fejlesztése
- **QUIC („quick”)**: User-space (alkalmazás rétegbeli) implementáció
- **Beépített titkosítás** (TLS 1.3), ami kötelező
- **Multiplexing és streaming** támogatása
- QUIC saját bináris transport protokoll, stream alapú átvitel

---

# HTTP/3: QUIC felett

- HTTP/3 = HTTP/2 továbbfejlesztett framing
- Minden **stream** külön kezelhető és **független**
- **Nincs TCP-level HOL blocking**
- HOL blocking teljes megszüntetése
- Csomagvesztés csak az adott streamre hat
- QPACK: Header Compression



---

# HTTP/3: Válasz a modern web problémáira

- Egymástól **független** és **párhuzamos stream-ek**
- **Gyors** kapcsolatfelépítés
- **Kevesebb round-trip**: Nincs külön TCP handshake és TLS handshake
- **Stabil** teljesítmény
- Még **gyorsabb** működés mobil hálózaton

# Összehasonlítás: HTTP/1.1 vs HTTP/2 vs HTTP/3

Tulajdonság	HTTP/1.1	HTTP/2	HTTP/3
Transport	TCP	TCP	UDP + QUIC
Multiplexing	Nincs	Van	Valódi, full-duplex
HOL Blocking	Nagyon nagy	Részben	Gyakorlatilag nincs
Header Compression	Nincs	HPACK	QPACK
Titkosítás	Opcionális	Opcionális	Mindig (beépített)
Prioritás	Nincs	Van	Van

---

# Hol tart ma a HTTP/3 elterjedtsége?

- Google, Facebook, YouTube, Cloudflare → teljes támogatás
- Böngészők: Chrome, Firefox, Edge, Safari támogatják
- Lassú, de stabil terjedés jellemzi



---

# Összefoglalás

- **HTTP/1.1** – soros, lassú, sok TCP kapcsolat
- **HTTP/2** – multiplexing, gyorsabb, de TCP-limitációkkal
- **HTTP/3** – QUIC alapú, UDP felett, modern, gyors, a jövő szabványa