Hypertext Transfer Protocol (HTTP) alapok

Jeszenszky Péter

2024. szeptember 20.

Mi a HTTP?

- Állapotnélküli alkalmazásszintű kérés/válasz protokollok egy családja, melyek egy közös általános interfésszel, kiterjeszthető szemantikával és önleíró üzenetekkel teszik lehetővé a rugalmas interakciót hálózati alapú hiperszöveg információs rendszerekkel.
- Fejlesztése kezdetben az IETF és a W3C közötti együttműködés keretében történt.
- Jelenleg az IETF HTTP munkacsoportja fejleszti: https://httpwg.org/

Jellemzők

- **Egységes interfész**: egységes interfészt biztosít erőforrásokkal történő interakcióhoz olyan üzenetek küldése révén, melyek reprezentációkat manipulálnak vagy továbbítanak.
- **A kliens-szerver modellen alapuló kérés/válasz protokoll**: kliensek és szerverek közötti üzenetcserével működik.
- **Állapotnélküli protokoll**: minden egyes kérés jelentése a többiétől külön értelmezhető.
- **Kiterjeszthető**: a HTTP számos általános kiterjesztési pontot határoz meg, melyek révén új egy verzió kiadása nélkül vezethetők be képességek a protokollba, mint például metódusok, állapotkódok vagy mezők.

Aktuális HTTP Verziók

- HTTP/1.1
- HTTP/2
- HTTP/3

Aktuális szabványok

- Roy T. Fielding (ed.), Mark Nottingham (ed.), Julian Reschke (ed.). *RFC 9110: HTTP Semantics*. June 2022. https://www.rfc-editor.org/rfc/rfc9110
- Roy T. Fielding (ed.), Mark Nottingham (ed.), Julian Reschke (ed.). *RFC 9111: HTTP Caching*. June 2022. https://www.rfc-editor.org/rfc/rfc9111
- Roy T. Fielding (ed.), Mark Nottingham (ed.), Julian Reschke (ed.). *RFC 9112: HTTP/1.1*. June 2022. https://www.rfc-editor.org/rfc/rfc9112
- Martin Thomson (ed.), Cory Benfield (ed.). *RFC 9113: HTTP/2*. June 2022. https://www.rfc-editor.org/rfc/rfc9113
- Mike Bishop (ed.). *RFC 9114: HTTP/3*. June 2022. https://www.rfc-editor.org/rfc/rfc9114

Történet (1)

HTTP 0.9:

- A Tim Berners-Lee által 1991-ben írt első dokumentált verzió.
- Nagyon egyszerű, mindössze olyan GET kéréseket támogat, melyekre válaszként egy ASCII karakterekből álló HTML dokumentum kerül visszaküldésre.
- Lásd: https://www.w3.org/Protocols/HTTP/AsImplemented.html

Történet (2)

HTTP/1.0:

- Tim Berners-Lee, Roy T. Fielding, Henrik Frystyk Nielsen. *RFC 1945: Hypertext Transfer Protocol – HTTP/1.0*. May 1996. https://www.rfc-editor.org/rfc/rfc1945
  - Olyan üzenetek használata, melyek a befoglalt tartalomról is tartalmaznak metaadatokat.
  - Már nem csupán HTML dokumentumok, hanem tetszőleges médiatípusok átvitelét támogatja.
  - Többféle metódus támogatása (GET, HEAD, POST, PUT, DELETE, LINK, ULINK).
  - Hitelesítés támogatása (*basic authentication*).

Történet (3)

HTTP/1.1:

- Roy T. Fielding, James Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Tim Berners-Lee. *RFC 2068: Hypertext Transfer Protocol – HTTP/1.1*. January 1997. https://www.rfc-editor.org/rfc/rfc2068
  - Újdonságok: perzisztens kapcsolatok, tartalomegyeztetés, tartományra vonatkozó kérések, ...
- Roy T. Fielding, James Gettys, Jeffrey C. Mogul, Henrik Frystyk Nielsen, Larry Masinter, Paul J. Leach, Tim Berners Lee. *RFC 2616: Hypertext Transfer Protocol – HTTP/1.1*. June 1999. https://www.rfc-editor.org/rfc/rfc2616
  - Az RFC 2068 javítása.

Történet (4)

HTTP/1.1 (folytatás):

- RFC 7230, RFC 7231, RFC 7232, RFC 7233, RFC 7234, RFC 7235
  - A HTTP/1.1 átdolgozása, mely 2014-ben hat RFC-ben került kiadásra.

Történet (5)

HTTP/2:

- A 2015 májusában kiadott RFC 7540 és RFC 7541 írja le.
- A HTTP/1.1-től az üzenetek "keretezésében" különözik: bináris protokoll, nem pedig szöveg-alapú.
- Számos új lehetőséget vezet be, mint például a multiplexelés, szerver *push*, ...

HTTP/3:

- A 2022 júniusában kiadott RFC 9114 és RFC 9204 írja le.
- A lehetőségek tekintetében nagyon hasonló a HTTP/2-höz, azonban már nem a TCP-n alapul, hanem az UDP-re épülő QUIC átviteli protokollon.

Történet (6)

HTTP szemantika:

- 2022 júniusában a HTTP/1.1, HTTP/2 és HTTP/3 specifikációk újrafogalmazásra kerültek a *HTTP Szemantika* specifikációra támaszkodva, mely közös alapokat biztosít az összes HTTP verzió számára.

Történet (7)

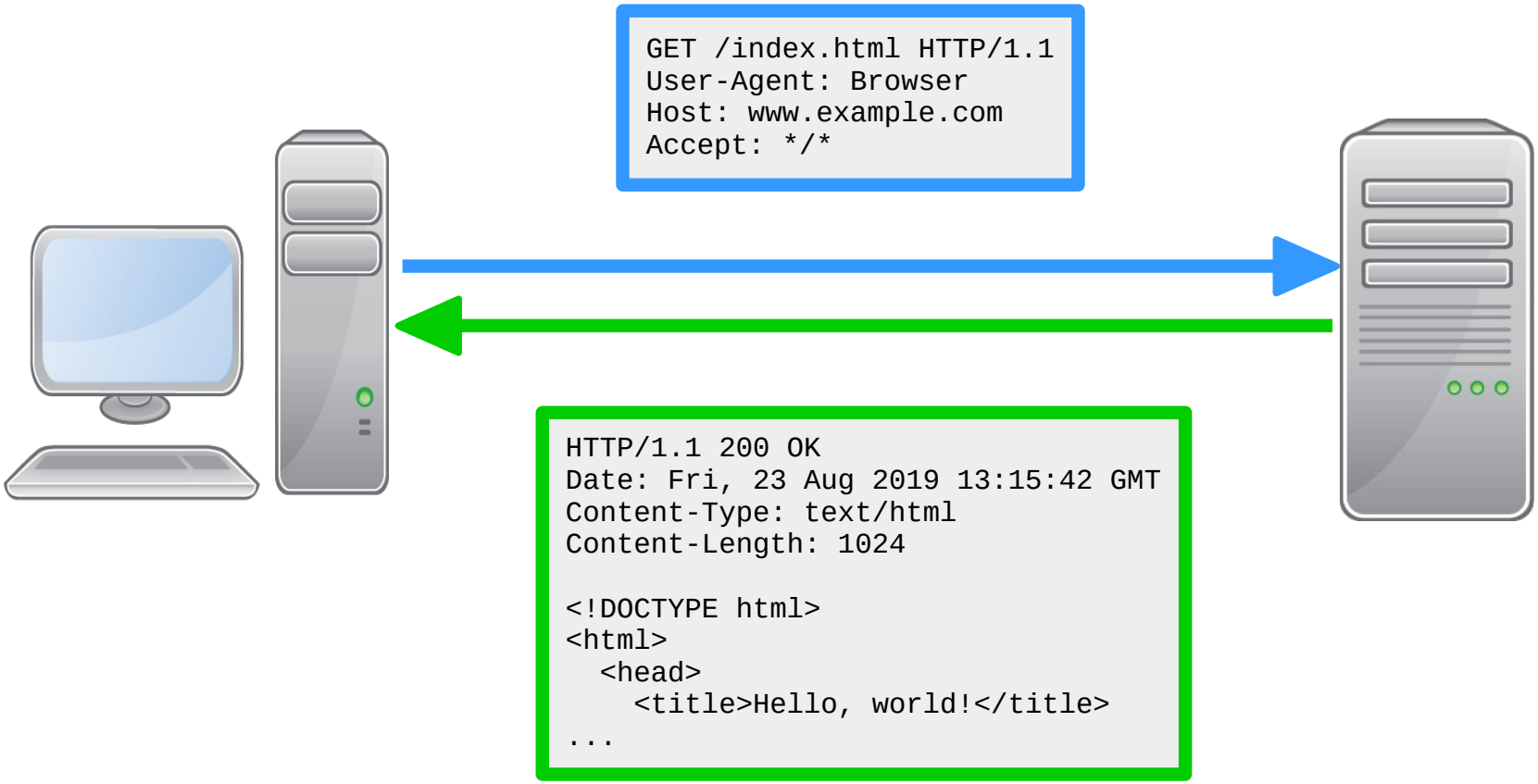
További ajánlott irodalom:

- *Evolution of HTTP* https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Evolution_of_HTTP
- *High Performance Browser Networking – Brief History of HTTP* https://hpbn.co/brief-history-of-http/

HTTP szemantika

- A *HTTP szemantika* specifikáció (azaz az RFC 9110)
  - írja le a HTTP általános architektúráját,
  - közös terminológiát határoz meg,
  - meghatározza a protokoll olyan aspektusait, melyek közösek minden verzióban.
- A HTTP szemantikájának lényege nem változik protokoll verziók között, annak a “dróton történő” kifejezése változhat.

Működés


GET /index.html HTTP/1.1
User-Agent: Browser
Host: www.example.com
Accept: */*
HTTP/1.1 200 OK
Date: Fri, 23 Aug 2019 13:15:42 GMT
Content-Type: text/html
Content-Length: 1024
<!DOCTYPE html>
<html>
<head>
<title>Hello, world!</title>
...

A HTTP kiterjesztése

Általános kiterjesztési pontok:

- Metódusok
- Állapotkódok
- Mezők
- Hitelesítési sémák
- Tartomány egységek
- Tartalomkódolások

Biztonságos HTTP

- A kapcsolatokat a TLS (korábbi nevén SSL) révén biztosítják.
- Lásd:
  - Eric Rescorla. *RFC 8446: The Transport Layer Security (TLS) Protocol Version 1.3*. August 2018. https://www.rfc-editor.org/rfc/rfc8446

curl

Parancssori eszköz (`curl`) és könyvtár (`libcurl`) URL szintaxissal meghatározott adatok átviteléhez.

- Webhely: https://curl.se/
- Tároló: https://github.com/curl/curl
- Programozási nyelv: C
- Platform: Linux, macOS, Windows, ...
- Licenc: *X11 License*

Támogatott protokollok: FTP(S), HTTP(S), POP3(S), SCP, SMTP(S), ...

HTTPie (1)

Parancssori HTTP kliens.

- Webhely: https://httpie.io/
- Tároló: https://github.com/httpie/cli
- Programozási nyelv: Python
- Platform: Linux, macOS, Windows
- Licenc: *New BSD License*

Megkötés: csak a HTTP/1.1-et támogatja.

HTTPie (2)

Jelenleg béta tesztelés alatt:

- HTTPie for Desktop (platform: Linux, macOS, Windows): https://httpie.io/download
- HTTPie for Web: https://httpie.io/app

Böngésző eszközök

- **Chromium**, **Google Chrome**, **Opera**: *Chrome DevTools*
  https://developer.chrome.com/docs/devtools/
- **Firefox**: *Firefox Developer Tools*
  https://firefox-source-docs.mozilla.org/devtools-user/
- **Chromium-alapú Edge**: *Microsoft Edge DevTools documentation*
  https://learn.microsoft.com/en-us/microsoft-edge/devtools-guide-chromium/landing/
- **Safari**: https://developer.apple.com/safari/tools/

REST Client (Visual Studio Code)

Visual Studio Code kiterjesztés, mely lehetővé teszi a felhasználók számára HTTP kérések küldését és a válaszok megtekintését a szerkesztőben.

- Weboldal: https://marketplace.visualstudio.com/items?itemName=humao.rest-client
- Tároló: https://github.com/Huachao/vscode-restclient
- Programozási nyelv: TypeScript
- Platform: Visual Studio Code
- Licenc: *MIT License*

Fogalmak (1)

- **Erőforrás**: Egy HTTP kérés célja, melyet egy URI azonosít.
- **Reprezentáció**:
  - Olyan információ, mely egy adott erőforrás múltbeli, jelenlegi vagy kívánt állapotát hivatott tükrözni.
  - Átvihető a protokollon keresztül.
  - Reprezentáció metaadatokból és reprezentáció adatokból áll.
  - Egy erőforrás több olyan reprezentációval is rendelkezhet vagy pedig képes lehet ilyenek előállítására, melyek az erőforrás jelenlegi állapotát hivatottak tükrözni.

Fogalmak (2)

- **Tartalomegyeztetés** (***content negotiation***): egy mechanizmus egy bizonyos kérés kielégítéséhez a legmegfelelőbb erőforrás reprezentáció kiválasztásához és szolgáltatásához, melyet akkor alkalmaznak, amikor az erőforrásoknak több alternatív reprezentációja lehet.

Fogalmak (3)

- **Kapcsolat**: a HTTP egy kliens/szerver protokoll, mely egy megbízható szállítási vagy viszony rétegbeli kapcsolat fölött működik.
- **Üzenet**: a HTTP egy állapotnélküli kérés/válasz protokoll üzenetek egy kapcsolaton keresztül történő váltásához.
  - Egy üzenet általában egy kérés vagy egy válasz. (Azonban például a HTTP/2 kapcsolatvezérlő üzeneteket is használ.)
- **Küldő/fogadó**: egy adott üzenetet elküldő vagy fogadó tetszőleges implementáció.

Fogalmak (4)

A kliens és a szerver olyan szerepek, melyet egy adott kapcsolatnál töltenek be programok. Ugyanaz a program kliens és szerver is lehet különböző kapcsolatoknál.

- **Kliens**: egy szerverrel egy vagy több HTTP kérés küldése céljából kapcsolatot létrehozó program.
- **Szerver**: egy program, mely kapcsolatokat fogad el abból a célból, hogy HTTP kéréseket szolgáljon ki HTTP válaszok küldésével.

Fogalmak (5)

- **Felhasználói ágens** (***user agent***): egy HTTP kérést kezdeményező program.
  - Például böngésző, keresőrobot, parancssori eszköz (például `curl`, HTTPie), háztartási gép, firmware frissítő szkript, mobil alkalmazás, ...
  - Felhasználói ágensnek lenni nem jelenti azt, hogy egy emberi felhasználó kezeli a programot a kérés időpontjában.
- **Eredet szerver** (***origin server***): egy program, mely hiteles válaszokat tud létrehozni egy adott cél erőforráshoz.

Fogalmak (6)

- **Közvetítő** (**intermediary**): lehetővé teszik kérések kapcsolatok egy láncán keresztül történő kiszolgálását.
  - A közvetítőknek a következő három szokásos formája van: proxy, átjáró, alagút.

Fogalmak (7)

Példa:

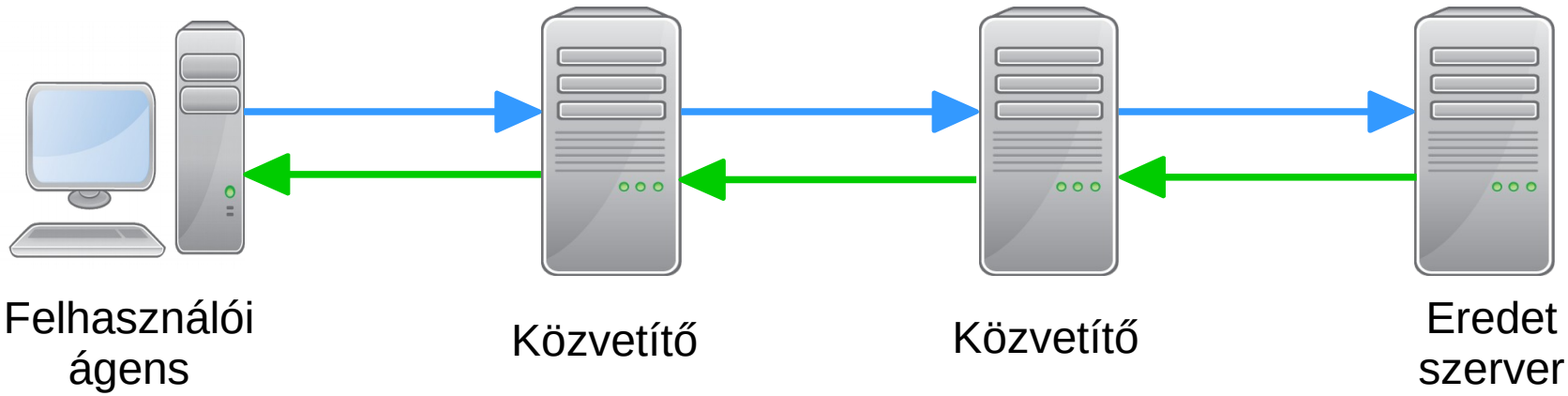


Figure 1: Három különböző kapcsolaton megy át a teljes láncon áthaladó kérés vagy válasz

A `http` and `https` URI sémák

- A HTTP definiálja a `http` és `https` URI sémákat.
  - Egy `http` vagy `https` URI eredet szerverét a hoszt azonosító és az opcionális portszám határozza meg.
  - Az útvonal komponens és az opcionális lekérdezés komponens egy lehetséges cél erőforrást azonosít az eredet szerver névterében.
- Egy `http` vagy `https` URI előfordulása nem jelenti azt, hogy mindig egy HTTP szerver figyel az URI által meghatározott eredet szerveren.
- A `https` sémán keresztül elérhető erőforrásoknak nincs közös identitása a `http` sémával.

`http` URI séma

- Célja, hogy lehetővé tegye erőforrások azonosítását egy olyan potenciális eredet szerveren, mely egy adott porton vár TCP kapcsolatokra.
- URI szintaxis:

  `"http://" host [":" port] [útvonal] ["?" lekérdezés]`

  - Ha nincs megadva vagy üres a port alkomponens, akkor a 80-as TCP port az alapértelmezés.

`https` URI séma

- Célja, hogy lehetővé tegye erőforrások azonosítását egy olyan potenciális eredet szerveren, mely egy adott porton vár TCP kapcsolatokra és mely képes HTTP kommunikációhoz biztonságossá tett TLS kapcsolat létrehozására.
- URI szintaxis:

  `"https://" host [":" port] [útvonal] ["?" lekérdezés]`

  - Ha nincs megadva vagy üres a port alkomponens, akkor a 443-as TCP port az alapértelmezés.

`http` és `https` URI összehasonlítás (1)

- Az üres útvonal ekvivalens a `"/"` útvonallal.
- A séma és a host komponensek kisbetű-nagybetű érzéketlenek és megadásuk rendszerint kisbetűkkel történik. A többi komponens összehasonlítása kisbetű-nagybetű érzékeny módon történik.
- A nem fenntartott karakterek ekvivalensek a százalékos kódolással megadott formájukkal.

`http` és `https` URI összehasonlítás (2)

- Például ekvivalensek az alábbi URI-k:
  - http://www.inf.unideb.hu/, http://www.inf.unideb.hu:80/, http://www.inf.unideb.hu, http://www.inf.unideb.hu:80
  - https://web.unideb.hu/~jeszy/, https://web.unideb.hu/%7ejeszy/, https://WEB.UNIDEB.HU/%7Ejeszy/
- Ekvivalens HTTP URI-król felételezhető, hogy ugyanazt az erőforrást azonosítják.

Üzenet keretezés

- A HTTP minden egyes főverziója saját szintaxist határoz meg az üzenetváltáshoz, melyet **keretezési mechanizmusnak** (***framing mechanism***) is neveznek.

Üzenet absztrakció

- Az RFC 9110 az üzenetek egy olyan absztakcióját határozza meg, mely szerint egy üzenet a következőkből áll:
  - Vezérlő adatok
  - Fejléc szakasz
  - Tartalom
  - Lezáró szakasz
- Ez az üzenet absztrakció egy több HTTP verziót átfogó általánosítás, melynek vannak olyan lehetőségei, melyek bizonyos verziókban nem találhatók meg.
- Az üzeneteket önleírónak szánják, azaz minden, amit egy fogadónak tudnia kell az üzenetről, megállapítható a (dekódolt) üzenet vizsgálatával.

Üzenet absztrakció: vezérlő adatok

- Az üzenetek az elsődleges céljukat leíró vezérlő adatokkal kezdődnek.
  - Kérésekben a vezérlő adatok a metódust, a kérést célt és a protokoll verziót foglalják magukban.
  - Válaszokban a vezérlő adatok az állapotkódot, az opcionális indok frázist és a protokoll verziót foglalják magukban.
- Minden HTTP üzenetnek van protokoll verziója.

Üzenet absztrakció: fejléc szakasz

- A tartalom előtt küldött vagy fogadott mezőket **fejlécmezőknek** (vagy informálisan egyszerűen **fejléceknek**) nevezzük.
- Egy üzenet fejléc szakasza fejléc mezősorok egy sorozatából áll.

Üzenet absztrakció: tartalom (1)

- A HTTP üzenetek egy teljes vagy részleges reprezentációt hordozhatnak az üzenet tartalmaként.
- A tartalom átvitele egy bájtfolyamként történik a fejléc szakasz után.
- A `Content-Type` és a `Content-Encoding` fejlécmezők által meghatározott formátumban és kódolásban van.

Üzenet absztrakció: tartalom (2)

Tartalom szemantika:

- A tartalom célját egy kérésben a metódus szemantika határozza meg.
  - Például egy reprezentáció egy POST kérés tartalmaként a cél erőforrás által feldolgozandó információkat ábrázol.
- Egy válaszban a tartalom célját a kérés metódusa, az állapotkód és a tartalmat leíró válasz mezők határozzák meg.
  - Például egy GET kérésre adott 200 (OK) állapotkódú válasz a cél erőforrás az üzenet létrehozásának pillanatában megfigyelhető aktuális állapotát ábrázolja.

Üzenet absztrakció: lezáró szakasz

- A tartalom után küldött vagy fogadott mezőket **lezáró mezőknek** (vagy informálisan egyszerűen **lezáróknak**) nevezzük.
- A lezáró mezők ellenőrző összegek, digitális aláírások, kézbesítési metrikák vagy utófeldolgozási állapot információk továbbítására használhatók.
- Egy üzenet lezáró szakasza lezáró mezősorok egy sorozatából áll.
- A lezáró mezőket a fejlécmezőktől elkülönítve ajánlott feldolgozni és tárolni.

Mezők (1)

- A HTTP mezőket használ adatok név/értek párok formájában történő szolgáltatásához.
- Mezők szolgálnak a következő információk továbbítására:
  - magát az üzenetet leíró metaadatok kérésekben és válaszokban (például `Date`),
  - reprezentáció metaadatok kérésekben és válaszokban (például `Content-Type`),
  - információk a kliensről kérésekben (például `User-Agent`),
  - információk a szerverről válaszokban (például `Server`),
  - erőforrás metaadatok válaszokban (például `Last-Modified`).

Mezők (2)

- A mezők küldése és fogadása az üzenetek fejléc és lezáró szakaszaiban történik.
  - Egy üzenet fejléc vagy lezáró szakaszában küldött mezőt fejlécmezőnek illetve lezáró mezőnek nevezünk.
  - Bizonyos mezők, mint például az `ETag`, előfordulhatnak fejléc- vagy lezáró mezőként is.

Mezők (3)

Mezőnevek:

- Egy mezőnév egy vagy több karakterből álló olyan sorozat, melyben az US-ASCII karakterkészlet egy részhalmaza használható csak.
- A mezőnevek kisbetű-nagybetű érzéketlenek.

Mezők (4)

Mezőértékek:

- Egy mezőérték egy olyan karaktersorozat, mely egy vagy több nyomtatható US-ASCII karakterből, szóközből és vizszintes tabulátorból áll.
- A vezető és záró *whitespace* karaktereket a felhasználás előtt el kell távolítani.
- A mezőkhöz előírható, hogy egyetlen tagot vagy pedig egy vesszővel elválasztott taglistát hordozzanak.
- Minden egyes mező korlátozhatja a megengedett értékek halmazát.

Mezők (5)

Mezőszakaszok:

- Tetszőleges számú olyan mezősorból állnak, melyek mindegyike egy mezőnevet és egy hozzá tartozó mezősor értéket tartalmaz.
- Amikor egy mezőnév megismétlődik egy szakaszban, az értéke egy olyan lista, melyben a mező a szakaszbeli mezősor értékei az előfordulásuk sorrendjében kerülnek összefűzésre elválasztóként egy vessző karakterrel.
  - A gyakorlatban a `Set-Cookie` fejlécmező gyakran több mezősorban is megjelenik egy válaszban, azonban a `Set-Cookie` mezősor értékek nem kerülnek egyesítésre egyetlen mezőértékké.
- Nem lényeges, hogy milyen sorrendben fordulnak elő különböző mezőnevű mezősorok egy szakaszban.

Mezők (6)

- A HTTP specifikációk sok szabványos mezőt határoznak meg.
- A HTTP által meghatározottakon túl sok más specifikáció definiál fejlécmezőket.
- Eltérő rendelkezés hiányában a mezők az összes HTTP verzióhoz kerülnek meghatározásra.
- Az IANA adminisztrálja a HTTP mezőket.
  - Lásd: *Hypertext Transfer Protocol (HTTP) Field Name Registry* https://www.iana.org/assignments/http-fields/http-fields.xhtml

Reprezentáció metaadatok (1)

- A reprezentáció fejlécmezők a reprezentációról szolgáltatnak metaadatokat.
- Amikor egy üzenet magában foglal tartalmat, a reprezentáció fejlécmezők írják le, hogy hogyan kell azt értelmezni.
- HEAD kérésre adott válaszban a reprezentáció fejlécmezők azokat a reprezentáció adatokat írják le, melyet tartalomként kaptunk volna, ha a kérés egy GET kérés lett volna.

Reprezentáció metaadatok (2)

- A `Content-Type` fejlécmező a reprezentáció médiatípusát jelzi.
- Példák:
  - `Content-Type: text/plain`
  - `Content-Type: image/png`
  - `Content-Type: text/html; charset=UTF-8`

Reprezentáció metaadatok (3)

- A `Content-Encoding` fejlécmező jelzi, hogy mely tartalom kódolások kerültek alkalmazásra a reprezentációra azokon túl, melyek a médiatípus velejárói.
- Így tehát jelzi, hogy így mely dekódolási mechanizmusokat kell alkalmazni ahhoz, hogy megkapjuk a `Content-Type` fejlécmező által hivatkozott médiatípusú adatokat.
- Példák:
  - `Content-Encoding: gzip`
  - `Content-Encoding: br`

Tartalomkódolások (1)

- Lehetővé teszik a reprezentáció a mögöttes médiatípust megőrző és veszteségmentes tömörítését vagy más hasznos átalakítását.
- A reprezentációt gyakran kódolt formában tárolják, közvetlenül továbbítják és csak a végső fogadó dekódolja.
- Az alkalmazott tartalomkódolás(ok) a reprezentáció tulajdonsága(i).
  - A reprezentációt leíró összes többi metaadat a kódolt formára vonatkozik.

Tartalomkódolások (2)

- Az IANA adminisztrálja a tartalomkódolásokat:
  - *Hypertext Transfer Protocol (HTTP) Parameters – HTTP Content Coding Registry* https://www.iana.org/assignments/http-parameters/http-parameters.xhtml#content-coding
- A tartalomkódolás nevek kisbetű-nagybetű érzéketlenek.

Tartalomkódolások (3)

Példák:

- **gzip**: GZIP állományformátum (lásd: RFC 1952)
- **br**: Brotli tömörített adatformátum (lásd: RFC 7932)

A `User-Agent` fejlécmező (1)

- A felhasználói ágensről tartalmaz információkat, ahonnan a kérés származik.
- Felhasználható a válasz testreszabásához vagy pedig böngésző vagy operációs rendszer használatra vonatkozó elemzésekre.
- A felhasználói ágens számára ajánlott minden egyes kérésben elküldeni a fejlécmezőt.
- A mezőérték egy vagy több termékazonosítóból áll, melyek mindegyikét nulla vagy több megjegyzés követi.
  - A termékazonosítók felsorolása a fontosságuk szerinti csökkenő sorrendben történik.
  - Minden egyes termékazonosító egy névből és egy opcionális verziószámból áll.
  - A megjegyzéseket zárójelek határolják.

A `User-Agent` fejlécmező (2)

Példák:

- **curl**: `curl/8.8.0`
- **Firefox**: `Mozilla/5.0 (X11; Linux x86_64; rv:130.0) Gecko/20100101 Firefox/130.0`
- **Google Chrome**: `Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/116.0.0.0 Safari/537.36`
- **Chromium-based Edge**: `Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.114 Safari/537.36 Edg/103.0.1264.51`

Metódusok (1)

- Elsődlegesen a metódusok jelzik a kérések célját.
- A metódus szemantika tovább specializálható a kérésben bizonyos fejlécmezőkkel (lást például a feltételes kérés fejlécmezőket).
- A metódusnevek kisbetű-nagybetű érzékenyek.

Metódusok (2)

A specifikáció által meghatározott szabványos metódusok:

- CONNECT
- DELETE
- GET
- HEAD
- OPTIONS
- POST
- PUT
- TRACE

Metódusok (3)

- Minden általános célú szervernek támogatnia kell a GET és a HEAD metódusokat, az összes többi metódus opcionális.
- További HTTP metódusok is meghatározásra kerültek.
- Az IANA adminisztrálja a HTTP metódusokat.
  - Lásd: *Hypertext Transfer Protocol (HTTP) Method Registry* https://www.iana.org/assignments/http-methods/http-methods.xhtml

Metódusok (4)

- Egy cél erőforrás által megengedett metódusok az `Allow` fejlécmezőben sorolhatók fel.
- Ha egy kérés metódusát nem ismeri fel vagy nem implementálja egy eredet szerver, akkor az 501 (*Not Implemented*) állapotkóddal ajánlott válaszolnia.
- Ha egy kérés metódusát felismeri és implementálja ugyan egy eredet szerver, de nem engedélyezi a cél erőforrásra, akkor a 405 (*Method Not Allowed*) állapotkóddal ajánlott válaszolnia.

GET metódus

- A cél erőforrás egy aktuális kiválasztott reprezentációjának átvitelét kérelmezi.
- Az információ-visszakeresés elsődleges mechanizmusa.
- Egy kliens úgy módosíthatja a GET jelentését a kérésben a `Range` fejlécmező küldésével, hogy az csak a kiválasztott reprezentáció bizonyos részeinek átvitelét kérelmezi.

HEAD metódus (1)

- Azonos a GET metódussal, azzal a különbséggel, hogy a szerver nem küldhet tartalmat a válaszban.
- Úgy használható metaadatok szerzésére a kiválasztott reprezentációról, hogy reprezentáció adatok nem kerülnek átvitelre.

HEAD metódus (2)

Példa:

```
$ curl --head https://www.r-project.org/
HTTP/1.1 200 OK
Date: Sun, 15 Sep 2024 12:34:32 GMT
Server: Apache
Last-Modified: Fri, 14 Jun 2024 08:10:02 GMT
ETag: "1b27-61ad5247107c3"
Accept-Ranges: bytes
Content-Length: 6951
Vary: Accept-Encoding
Content-Type: text/html
```

POST metódus (1)

- Azt kérelmezi, hogy a cél erőforrás dolgozza fel a kérésben mellékelt reprezentációt a saját szemantikájának megfelelően.
- Tipikus felhasználások:
  - Adatok (például űrlap adatok) küldése egy adatfeldolgozó folyamat számára.
  - Egy üzenet postázása egy hírcsoportba, levelezési listára vagy blogba.
  - Egy új erőforrás létrehozása.
  - Adatok hozzáfűzése egy erőforrás létező reprezentációjához.

POST metódus (2)

Példa: httpbin.org

```
$ http --form https://httpbin.org/post number=42 text="Hello, World!" -v
POST /post HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, br, zstd
Connection: keep-alive
Content-Length: 32
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Host: httpbin.org
User-Agent: HTTPie/3.2.3

number=42&text=Hello%2C+World%21
```

POST metódus (3)

Példa: httpbin.org

```
$ http --form --multipart https://httpbin.org/post \
  number=42 text="Hello, World!" -v
POST /post HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, br, zstd
Connection: keep-alive
Content-Length: 225
Content-Type: multipart/form-data; boundary=f067108d371949f2a8c4fdbb991afac1
Host: httpbin.org
User-Agent: HTTPie/3.2.3

--90f9bc6b7be04d918e70eda8ef004ec8
Content-Disposition: form-data; name="number"

42
--90f9bc6b7be04d918e70eda8ef004ec8
Content-Disposition: form-data; name="text"

Hello, World!
--90f9bc6b7be04d918e70eda8ef004ec8--
```

POST metódus (4)

Példa: Fun Translations API

```
$ http --form https://api.funtranslations.com/translate/yoda.json \
  text="Hello, World!" -v
POST /translate/yoda.json HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate, br, zstd
Connection: keep-alive
Content-Length: 22
Content-Type: application/x-www-form-urlencoded; charset=utf-8
Host: api.funtranslations.com
User-Agent: HTTPie/3.2.3

text=Hello%2C+World%21
```

POST metódus (5)

Példa (folytatás): Fun Translations API

```
HTTP/1.1 200 OK
Connection: keep-alive
Content-Language: en
Content-Length: 183
Content-Type: application/json; charset=utf-8
Date: Sun, 15 Sep 2024 12:39:26 GMT
Server: nginx/1.14.0 (Ubuntu)
X-Powered-By: Luracast Restler v3.0.0rc3
X-RateLimit-Limit: 10 per hour
X-RateLimit-Remaining: 9

{
    "contents": {
        "text": "Hello, World!",
        "translated": "Force be with you,World!",
        "translation": "yoda"
    },
    "success": {
        "total": 1
    }
}
```

PUT metódus (1)

- Azt kérelmezi, hogy a kerüljön létrehozásra vagy helyettesítésre a cél erőforrás állapota a kérésben mellékelt reprezentáció által meghatározott állapottal.
- Egy adott reprezentációt tartalmazó sikeres PUT kérés arra enged következtetni, hogy egy következő GET kérés ugyanarra a cél erőforrásra egy 200 (OK) állapotkódú válaszban elküldött ekvivalens reprezentációt eredményez.

PUT metódus (2)

Példa: transfer.sh

```
$ echo "Hello, World!" | http PUT https://transfer.sh/hello.txt \
  Content-Type:text/plain -v
PUT /hello.txt HTTP/1.1
Accept: application/json, */*;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Content-Length: 14
Content-Type: text/plain
Host: transfer.sh
User-Agent: HTTPie/3.2.2

Hello, World!
```

PUT metódus (3)

Példa (folytatás): transfer.sh

```
HTTP/1.1 200 OK
Content-Length: 40
Content-Type: text/plain
Date: Mon, 07 Aug 2023 10:52:58 GMT
Server: Transfer.sh HTTP Server
X-Made-With: <3 by DutchCoders
X-Served-By: Proudly served by DutchCoders
X-Url-Delete: https://transfer.sh/fUMsV053HK/hello.txt/SYNHIEOxWNFhDsEhKpf7

https://transfer.sh/fUMsV053HK/hello.txt
```

Különbség a PUT és POST között

A POST és a PUT metódus közötti alapvető eltérést a mellékelt reprezentáció eltérő célja hangsúlyozza:

- A cél erőforrás egy POST kérésben a mellékelt reprezentációt hivatott kezelni.
- A mellékelt reprezentáció egy PUT kérésben a cél erőforrás állapotát hivatott helyettesíteni.

DELETE metódus (1)

- Azt kérelmezi, hogy az eredet szerver törölje a cél erőforrás és aktuális funkcionalitása közötti kapcsolatot.
- Ha a cél erőforrásnak egy vagy több aktuális reprezentációja van, akkor ezeket az eredet szerver vagy megsemmisíti, vagy nem, a kapcsolódó tárterület vagy felszabadításra kerül, vagy nem, teljes egészében az erőforrás természetétől és az eredet szerver általi implementációjól függően.
- Viszonylag kevés erőforrás engedi meg a DELETE metódust.

DELETE metódus (2)

Példa: transfer.sh

```
$ http DELETE https://transfer.sh/fUMsV053HK/hello.txt/SYNHIEOxWNFhDsEhKpf7 -v
DELETE /fUMsV053HK/hello.txt/SYNHIEOxWNFhDsEhKpf7 HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Connection: keep-alive
Content-Length: 0
Host: transfer.sh
User-Agent: HTTPie/3.2.2

HTTP/1.1 200 OK
Content-Length: 0
Date: Mon, 07 Aug 2023 10:57:30 GMT
Server: Transfer.sh HTTP Server
X-Made-With: <3 by DutchCoders
X-Served-By: Proudly served by DutchCoders
```

OPTIONS metódus (1)

- A cél erőforrás kommunikációs opcióiról (az eredet szerveren vagy egy közvetítőn) rendelkezésre álló információkat kér.
- Cél erőforrásként * általában a szervert jelenti, nem pedig egy meghatározott erőforrást.
- Egy OPTIONS kérésre adott sikeres válaszban a szerver számára ajánlott minden olyan fejlécmező elküldése, melyek a szerver által implementált és a cél erőforrásra alkalmazható opcionális lehetőségeket jelezhetnek.
  - Például az `Allow` fejlécmező a cél erőforrás által támogatottként hirdetett metódusokat sorolja fel.

OPTIONS metódus (2)

Példa:

```
$ curl -X OPTIONS https://apache.org/foundation/contact.html -v
> OPTIONS /foundation/contact.html HTTP/1.1
> Host: apache.org
> User-Agent: curl/8.8.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Connection: keep-alive
< Content-Type: text/html
< Content-Length: 0
< Server: Apache
< Allow: HEAD,GET,POST,OPTIONS,TRACE
< ...
```

TRACE metódus

- A kérés visszaküldését kérelmezi.
  - A kérés végső fogadója kell, hogy visszaküldje a kapott üzenetet a kliensnek egy 200-as (OK) állapotkódú válasz tartalmaként. Ennek egy módja a `message/http` formátum használata.
  - Az érzékeny adatokat tartalmazó kérés mezőket ki kell zárni.
- A metódus általában biztonsági okokból nem engedélyezett a szervereken.

Állapotkódok (1)

- Egy válasz állapotkódja egy olyan háromjegyű decimális egész szám, mely a kérés eredményét és a válasz jelentését írja le, beleértve azt, hogy sikeres volt-e a kérés és hogy milyen tartalom van benne mellékelve (ha van egyáltalán).
- Az összes érvényes állapotkód a 100 és 599 közötti tartományba esik.

Állapotkódok (2)

Az állapotkód első számjegye a válasz fajtáját határozza meg:

- 1*xx* (informáló): a kapcsolat állapotát vagy a kérés előrelhaladását közlő köztes választ jelez egy végső válasz küldése előtt.
- 2*xx* (siker): azt jelzi, hogy a kérés sikeresen fogadásra, értelmezésre és elfogadásra került.
- 3*xx* (átirányítás): azt jelzi, hogy a felhasználói ágens további műveletet kell, hogy végrehajtson a kérés teljesítéséhez, melyet automatikusan elvégezhet.
- 4*xx* (kliens hiba): azt jelzi, hogy a kérés rossz szintaxisú vagy nem teljesíthető.
- 5*xx* (szerver hiba): azt jelzi, hogy a szerver nem teljesített egy nyilvánvalóan érvényes kérést.

Állapotkódok (3)

- Egy kliensnek nem kell megértenie minden regisztrált állapotkód jelentését.
  - Kötelező azonban az állapotkód fajtájának megértése az első számjegy alapján.
  - Egy nem felismert állapotkódot az *x*00 állapotkóddal ekvivalensként kell kezelni, ahol *x* a nem felismert állapotkód első számjegye.
- HEAD kérésekre adott válaszok kivételével a 4*xx* vagy 5*xx* állapotkódú válaszokban a szerver számára ajánlott egy olyan reprezentáció küldése, mely a hiba magyarázatát tartalmazza, valamint azt, hogy a hibát okozó körülmények átmenetiek vagy állandóak.

Állapotkódok (4)

- Az állapotkódok kiterjeszthetők.
- Az IANA adminisztrálja az állapotkódokat.
  - Lásd: *Hypertext Transfer Protocol (HTTP) Status Code Registry* https://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml

Főbb állapotkódok (1)

Állapotkód	Indok	Leírás
100	Continue	A szerver megkapta a kérés bevezető részét és még nem utasította el. A szerver azután kíván egy végső választ küldeni, ha a teljes kérést megkapta és megfelelően járt el.
101	Switching Protocols	A szerver készen áll a kapcsolaton használt alkalmazási protokoll lecserélésére.

Főbb állapotkódok (2)

Állapotkód	Indok	Leírás
200	OK	A kérés sikeres volt. A válaszban küldött tartalom a metódustól függ. Például egy GET kérésre adott válasz tartalma a cél erőforrás egy reprezentációja.
201	Created	A kérés teljesítésre került, eredményül egy vagy több új erőforrás került létrehozásra.
202	Accepted	A kérés elfogadásra került, de a feldolgozása nem fejeződött be.
204	No Content	A szerver sikeresen teljesítette a kérést, de a válaszban nem kerül küldésre további tartalom.
206	Partial Content	A szerver sikeresen teljesített egy tartományra vonatkozó kérést a kiválasztott reprezentáció egy vagy több részének átvitelével.

Főbb állapotkódok (3)

Állapotkód	Indok	Leírás
300	Multiple Choices	A cél erőforrásnak egynél több reprezentációja van. HEAD kérés kivételével ajánlott a válaszban tartalomként egy reprezentáció metaadatokból és URI hivatkozásokból álló lista elhelyezése, melyből a felhasználói ágens kiválaszthatja a számára legmegfelelőbbet.
301	Moved Permanently	A cél erőforrás URI-ja véglegesen megváltozott, a rá való további hivatkozásokhoz a mellékelt URI-k egyikét kellene használni.
302	Found	A cél erőforrás ideiglenesen egy másik URI alatt található.
303	See Other	A szerver a felhasználói ágenst egy másik erőforrásra irányítja át, melynek célja, hogy egy közvetett választ adjon az eredeti kérésre.
304	Not Modified	A szerver egy feltételes GET vagy HEAD kérést kapott. Nincs szükség arra, hogy elküldje a cél erőforrás reprezentációját, mivel a kérés azt jelzi, hogy a kliens érvényes reprezentációval rendelkezik.

Főbb állapotkódok (4)

Állapotkód	Indok	Leírás
400	Bad Request	A szerver nem tudja vagy nem fogja feldolgozni a kérést valamilyen kliens hiba miatt (például a kérés szintaktikailag hibás).
401	Unauthorized	A kérés nem tartalmaz érvényes hitelesítő adatokat a cél erőforráshoz.
403	Forbidden	A szerver ugyan megértette a kérést, de megtagadja a teljesítését. Ha a kérésben hitelesítő adatok vannak, akkor azokat nem megfelelőnek tekinti a szerver a hozzáféréshez.
404	Not Found	Az eredet szervernek nincs a cél erőforráshoz aktuális reprezentációja vagy nem kívánja azt nyilvánosságra hozni.
405	Method Not Allowed	Az eredet szerver ismeri ugyan a metódust, de a cél erőforrás nem támogatja azt.

Főbb állapotkódok (5)

Állapotkód	Indok	Leírás
500	Internal Server Error	Olyan váratlan körülmény lépett fel a szerveren, mely megakadályozza a kérés teljesítését.
501	Not Implemented	A szerver nem támogatja a kérés teljesítéséhez szükséges funkcionalitást (például metódust).
503	Service Unavailable	A szerver jelenleg nem képes a kérés kezelésére például ideiglenes túlterhelés vagy menetrend szerinti karbantartás miatt.

Móka az állapotkódokkal

Egy kis móka:

- 418 (I'm a teapot):
  - Larry Masinter. *RFC 2324: Hyper Text Coffee Pot Control Protocol (HTCPCP/1.0)*. 1 April 1998. https://www.rfc-editor.org/rfc/rfc2324
  - Imran Nazar. *RFC 7168: The Hyper Text Coffee Pot Control Protocol for Tea Efflux Appliances (HTCPCP-TEA)*. 1 April 2014. https://www.rfc-editor.org/rfc/rfc7168
- *HTTP Cats* https://http.cat/
- *HTTP Status Dogs* https://httpstatusdogs.com/

Átirányítás (1)

Példa:

```
$ curl --http1.1 -v http://w3.org/
> GET / HTTP/1.1
> Host: w3.org
> User-Agent: curl/8.8.0
> Accept: */*
>
< HTTP/1.1 301 Moved Permanently
< Location: https://www.w3.org/
< ...
<
```

Átirányítás (2)

Példa:

```
$ curl --http1.1 -v -L http://w3.org/
> Host: w3.org
> User-Agent: curl/8.8.0
> Accept: */*
>
< HTTP/1.1 301 Moved Permanently
< Location: https://www.w3.org/
< ...
<
> GET / HTTP/1.1
> Host: www.w3.org
> User-Agent: curl/8.8.0
> Accept: */*
>
< HTTP/1.1 200 OK
< ...
```

Átirányítás (3)

Példa:

```
$ http https://dbpedia.org/resource/Hungary
HTTP/1.1 303 See Other
Connection: keep-alive
Content-Length: 0
Content-Type: text/html; charset=UTF-8
Date: Sun, 15 Sep 2024 12:55:33 GMT
Location: http://dbpedia.org/page/Hungary
Server: Virtuoso/08.03.3331 (Linux) x86_64-generic-linux-g
...
```

Tartalomegyeztetés

- Amikor a válaszok (akár sikert, akár hibát jelző) tartalmat hordoznak, az eredet szerver gyakran többféle módon is ábrázolhatja a tartalmat, például különböző formátumokban, nyelveken vagy kódolásokkal.
- A különböző felhasználóknak vagy felhasználói ágenseknek ugyancsak eltérő képességei, jellemzői vagy preferenciái lehetnek, melyek befolyásolhatják, hogy a rendelkezésre álló reprezentációk közül melyik lenne számukra a legjobb.
- Ezért biztosít a HTTP a tartalomegyeztetéshez mechanizmusokat.

Tartalomegyeztetési minták

A HTTP két főbb tartalomegyeztetési mintát határoz meg:

- **Proaktív egyeztetés** (***proactive negotiation***): a szerver választja ki a reprezentációt a felhasználói ágens kifejezett preferenciái alapján.
  - **Szerver-vezérelt** (***server-driven***) tartalomegyeztetés néven is ismert.
- **Reaktív egyeztetés** (***reactive negotiation***): a szerver választásra kínálja fel a felhasználói ágensnek a reprezentációk listáját.
  - **Ágens-vezérelt** (***agent-driven***) tartalomegyeztetés néven is ismert.

Más tartalomegyeztetési minták is lehetségesek. Ezek a minták egymást nem kölcsönösen kizáróak.

Proaktív egyeztetés (1)

- Az előnyben részesített reprezentációt egy algoritmussal választja ki az eredet szerver a felhasználói ágens preferenciái alapján.
- A választás a válaszhoz rendelkezésre álló reprezentációk és a kérésben szolgáltatott különféle információk – beleértve az explicit tartalomegyeztetési fejlécmezőket, a `User-Agent` mező részeit és az olyan implicit jellemzőket, mint például a kliens hálózati címe – összehasonlítása alapján történik.
- Proaktív egyeztetésnek alávetett válaszban gyakran kerül küldésre a `Vary` fejlécmező annak jelzésére, hogy a kérés mely részei kerültek felhasználásra a kiválasztási algoritmusban.

Proaktív egyeztetés (2)

Előnyös:

- Amikor nehéz egy felhasználói ágens számára leírni a rendelkezésre álló reprezentációk közüli választás algoritmusát, vagy
- amikor a szerver az első válaszban el kívánja küldeni a felhasználói ágensnek az általa legjobbnak becsült reprezentációt, elkerülendő egy további kérést.

Proaktív egyeztetés (3)

Hátrányok:

- Lehetetlen a szerver számára annak pontos meghatározása, hogy mi lenne a legjobb egy tetszőleges felhasználónak, mivel ehhez kimerítő ismeretek lennének szükségesek a felhasználói ágens képességeiről és a válasz tervezett felhasználási módjáról.
- Nagyon nem hatékony a felhasználói ágens képességeit minden kérésben leírni, ez egyben a felhasználó magánszférájára is egy lehetséges kockázatot jelenthet.
- Bonyolítja az eredet szerver megvalósítását és a kérésekre válaszokat előállító algoritmusokat.
- Korlátozza a válaszok újrafelhasználhatóságát megosztott gyorsítótárazáshoz.

Reaktív egyeztetés (1)

- A tartalom kiválasztását a felhasználói ágens végzi el, miután egy olyan kezdeti választ kap, mely erőforrások egy listáját tartalmazza alternatív reprezentációkhoz.
- A kiválasztást végrehajthatja automatikusan a felhasználói ágens vagy kézzel a felhasználó.

Reaktív egyeztetés (2)

- Előnyös:
  - Amikor a válasz általánosan használt dimenziók (például típus, nyelv vagy kódolás) mentén változik, vagy
  - amikor az eredet szerver nem képes a felhasználói ágens képességeinek megállapítására a kérés vizsgálatából.
- Hátrányok:
  - Miután megkapta az alternatív reprezentációk listáját, a felhasználói ágensnek egy további kérést kell végrehajtania ahhoz, hogy megkapja a kívánt reprezentációt.
  - A HTTP nem határoz meg mechanizmust az automatikus kiválasztás támogatására.

q paraméter

- A tartalomegyeztetési mezők egy q nevű szokásos paramétert használnak ahhoz, hogy egy relatív "súlyt" rendeljenek hozzá az egyes fajta tartalmak preferenciáihoz.
- A súly egy egy 0 és 1 közé normalizált valós szám, ahol 0,001 a legkevésbé preferált, 1 pedig a leginkább preferált alternatívát jelenti, a 0 jelentése pedig "nem elfogadható".
- Ha nem jelenik meg a q paraméter, akkor 1 az alapértelmezés.

Tartalomegyeztetési mezők

Az alábbi fejlécmezőket használhatják a kérésekben a felhasználói ágensek a preferenciáik megadásához:

- **`Accept`**: a válasz médiatípus preferenciák megadására szolgál.
- **`Accept-Charset`**: a szöveges válasz tartalmakban használt karakterkódolási preferenciák jelzésére szolgál.
- **`Accept-Encoding`**: a tartalomkódolási preferenciák jelzéséhez használható.
- **`Accept-Language`**: a válaszban preferált természetes nyelvek jelzésére használható.

Ha egy kérés nem tartalmazza ezen mezők valamelyikét, akkor ez azt jelenti, hogy a küldőnek nincs preferenciája a megfelelő dimenzió tekintetében.

`Accept` fejlécmező (1)

- Értéke média tartományok egy vesszővel elválasztott listája, melyben minden egyes média tartományt 0 vagy több médiatípus paraméter (például `charset`) valamint egy opcionális relatív súly (azaz `q` paraméter) követhet.
- Média tartomány:
  - `*/*`: az összes médiatípust jelenti
  - *type*`/*`: az adott típus összes altípusát jelenti (például `text/*`)
  - *type*/*subtype*: az adott médiatípust jelenti (például `text/html`)

Accept fejlécmező (2)

- A mező értéke változhat, például eltérő lehet egy, a böngésző címsorában hivatkozott dokumentum vagy egy <img> HTML elemben hivatkozott kép letöltésekor.
  - Lásd: https://developer.mozilla.org/en-US/docs/Web/HTTP/Content_negotiation#the_accept_header
- Alapértelmezett értékek bizonyos böngészőkhöz: https://developer.mozilla.org/en-US/docs/Web/HTTP/Content_negotiation/List_of_default_Accept_values

Accept fejlécmező (3)

Alapértelmezett érték a Firefox 92 számú vagy későbbi verzióiban:

```
Accept: text/html,application/xhtml+xml,
application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
```

Média tartomány	q
text/html	1
application/xhtml+xml	1
image/avif	1
image/webp	1
application/xml	0.9
/	0.8

Tartalomegyeztetés (1)

Példa:

```
$ curl http://www.gnu.org/ -H Accept-Language:de -v -o gnu.de.html
> GET / HTTP/1.1
> Host: www.gnu.org
> User-Agent: curl/8.8.0
> Accept: */*
> Accept-Language:de
>
< HTTP/1.1 200 OK
< Date: Sun, 15 Sep 2024 12:58:07 GMT
< Server: Apache
< Content-Location: home.de.html
< Vary: negotiate,accept-language,Accept-Encoding
< Cache-Control: max-age=0
< Expires: Sun, 15 Sep 2024 12:58:07 GMT
< Content-Length: 32815
< Content-Type: text/html
< Content-Language: de
< ...
```

Tartalomegyeztetés (2)

Példa:

```
$ http https://www.mozilla.org/ Accept-Language:es -v
GET / HTTP/1.1
Accept: */*
Accept-Encoding: gzip, deflate
Accept-Language: es
Connection: keep-alive
Host: www.mozilla.org
User-Agent: HTTPie/3.2.3

HTTP/1.1 302 Moved Temporarily
Connection: keep-alive
Content-Length: 0
Content-Type: text/html; charset=utf-8
Date: Sun, 15 Sep 2024 13:13:40 GMT
Location: /es-ES/
Server: granian
Vary: Accept-Language
...
```

Tartalomegyeztetés (3)

Példa: DBpedia

- Adatok kérése JSON formátumban:

```
$ curl http://dbpedia.org/resource/Hungary \
  -H Accept:application/json -L -o Hungary.json
```

- Adatok kérése XML formátumban:

```
$ curl https://dbpedia.org/resource/Hungary \
  -H Accept:application/rdf+xml -L -o Hungary.xml
```

- Adatok kérése Turtle formátumban:

```
$ curl https://dbpedia.org/resource/Hungary \
  -H Accept:text/turtle -L -o Hungary.ttl
```

HTTP/1.1

- Szöveg-alapú protokoll.
- Egy HTTP/1.1 üzenet kérés vagy válasz.

HTTP/1.1: üzenet formátum (1)

- Az üzenetek egy **kezdősor**ral (***start-line***) kezdődnek, melyet CRLF követ.
- A kezdősort nulla vagy több **fejlécsor** követi, melyeket együttesen **fejlécek**nek vagy **fejléc szakasz**nak neveznek.
- Egy üres sor jelzi a fejléc szakasz végét.
- Opcionálisan szerepelhet az üzenet végén egy **üzenettörzs** (***message body***).

HTTP/1.1: üzenet formátum (2)

- A kérések kezdősorának szintaxisa:

  `<metódus> <kérés-cél> "HTTP/1.1"`

  - A kérés-cél a cél erőforrást azonosítja, melyre a kérés vonatkozik.
    - Leggyakoribb formája: `útvonal ["?" lekérdezés]`.
    - Ha a cél URI útvonal komponense üres, akkor a kliens `"/"`-t kell, hogy küldjön útvonalként.
    - A cél URI host és port komponense a `Host` fejlécmezőben kerül elküldésre.

HTTP/1.1: üzenet formátum (3)

- A válaszok első sorát **állapotsor**nak nevezik és a következő szintaxisa van:

  `"HTTP/1.1" <állapotkód> [indok_frázis]`

  - Az opcionális indok frázis az állapotkód egy szöveges leírása.
  - Egy kliens számára ajánlott az indok fázis figyelmen kívül hagyása.

HTTP/1.1: üzenet formátum (4)

Mezősorok:

- Minden mezősor egy kisbetű-nagybetű érzéketlen mezőnévvel kezdődik, melyet egy pontosvessző (":"), opcionális vezető *whitespace*, egy mezősor érték és záró *whitespace* követ. A mezősorokat egy CRLF zárja.

HTTP/1.1: üzenet formátum (5)

Üzenettörzs:

- Egy HTTP/1.1 üzenet törzse (ha van) szolgál a kérés vagy válasz tartalmának hordozására.
- Az üzenettörzs azonos a tartalommal, feltéve hogy nem kerül alkalmazásra egy átviteli kódolás (például `chunked`).

Webszerver szoftverek (1)

Webszerverek szoftverek piaci részesedése:

- *Netcraft Web Server Survey*
  https://www.netcraft.com/resources/?topic=web-server-survey

Webszerver szoftverek (2)

A legelterjedtebben használt webszerver szoftverek:

Név	Platform	Licenc	Megjegyzés
Apache HTTP Server https://httpd.apache.org/	cross-platform	*Apache License 2.0*	
nginx https://nginx.org/	cross-platform	2-pontos BSD / nem szabad	Kiejtés: *engine x*
Internet Information Services (IIS) https://www.iis.net/	Windows	nem szabad	
Google Web Server (GWS)	Linux	nem szabad	A Google által használt saját fejlesztésű szerver.

Java támogatás

- **Pre-JDK 11**: A `java.net.HttpURLConnection` osztály biztosít korlátozott HTTP kliens funckionalitást: https://docs.oracle.com/en/java/javase/21/docs/api/java.base/java/net/HttpURLConnection.html
- **JDK 11-**: A `java.net.http` csomag biztosít átfogó HTTP kliens funkcionalitást: https://docs.oracle.com/en/java/javase/21/docs/api/java.net.http/java/net/http/package-summary.html

Kliens programkönyvtárak

- Java:
  - Eclipse Jetty HTTP Client (licenc: *Eclipse Public License 2.0*/*Apache License 2.0*) https://jetty.org/ https://github.com/eclipse/jetty.project
  - OkHttp (licenc: *Apache License 2.0*) https://square.github.io/okhttp/ https://github.com/square/okhttp
  - Retrofit (licenc: *Apache License 2.0*) https://square.github.io/retrofit/ https://github.com/square/retrofit
- Python:
  - Requests (licenc: *Apache License 2.0*) https://requests.readthedocs.io/ https://github.com/psf/requests
  - urllib3 (licenc: *MIT License*) https://urllib3.readthedocs.io https://github.com/urllib3/urllib3

További ajánlott irodalom

- *MDN Web Docs – HTTP*
  https://developer.mozilla.org/docs/Web/HTTP
- Ilya Grigorik. *High Performance Browser Networking*. O'Reilly, 2013.
  https://hpbn.co/