

Razvoj programske podpore za web

**- predavanja -
2021./2022.**

8. HTTP

Creative Commons



- slobodno smijete:
 - **dijeliti** — umnožavati, distribuirati i javnosti priopćavati djelo
 - **prerađivati** djelo
- pod sljedećim uvjetima:
 - **imenovanje:** morate priznati i označiti autorstvo djela na način kako je specificirao autor ili davatelj licence (ali ne način koji bi sugerirao da Vi ili Vaše korištenje njegova djela imate njegovu izravnu podršku).
 - **nekomercijalno:** ovo djelo ne smijete koristiti u komercijalne svrhe.
 - **dijeli pod istim uvjetima:** ako ovo djelo izmijenite, preoblikujete ili stvarate koristeći ga, preradu možete distribuirati samo pod licencom koja je ista ili slična ovoj.

U slučaju daljnjeg korištenja ili distribuiranja morate drugima jasno dati do znanja licencne uvjete ovog djela.

Od svakog od gornjih uvjeta moguće je odstupiti, ako dobijete dopuštenje nositelja autorskog prava.

Ništa u ovoj licenci ne narušava ili ograničava autorova moralna prava.

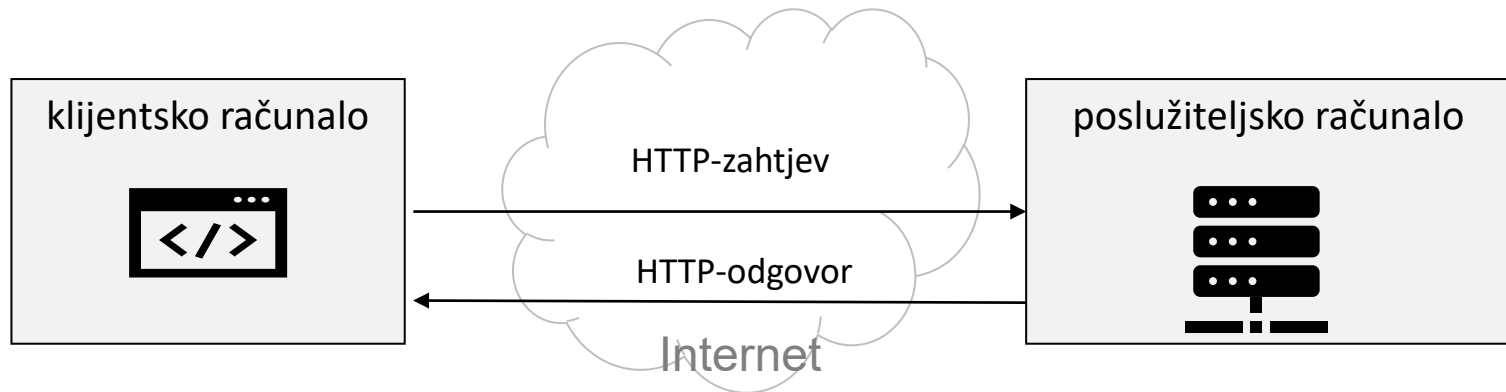
Tekst licence preuzet je s <http://creativecommons.org/>

Sadržaj predavanja

- Uvod
 - Osnove protokola HTTP
 - Web-poslužitelj i web-preglednik
 - Media Type i tijek komunikacije
 - HTTPS
- Identifikacija resursa
 - URI, URL, URN
 - sintaksa URI-ja
 - relativni i apsolutni URI
- Poruke protokola HTTP
 - zahtjev i odgovor
 - metode zahtjeva i kôdovi odgovora
- Priručna spremišta

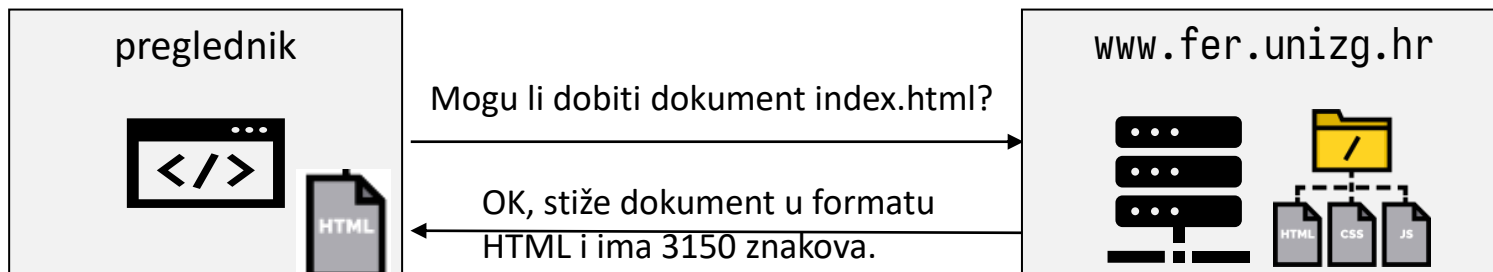
Hypertext Transfer Protocol (HTTP)

- HTTP je **protokol** na aplikacijskom sloju
 - protokol definira format i sadržaj poruka (zahtjeva i odgovora) te očekivano „ponašanje” poslužitelja, tj. pravila za generiranje odgovora na primljeni zahtjev
- **Zahtjev**: definira operaciju (tzv. metodu), oznaku resursa, verziju protokola, itd.
- **Odgovor**: rezultat (uspjeh, neuspjeh, pogreška,..) opisan statusnim kôdom i sadržaj resursa (npr. datoteka HTML, CSS, JPEG...)



Model klijent-poslužitelj

Web-poslužitelj i web-preglednik



■ Web-klijent ili preglednik

(engl. *browser*)

- nudi grafičko korisničko sučelje za prikaz web-sadržaja
- npr. Chrome, Firefox, Microsoft Edge, Opera, itd.



Firefox

Opera

Microsoft
Edge

Vivaldi

■ Web-poslužitelj

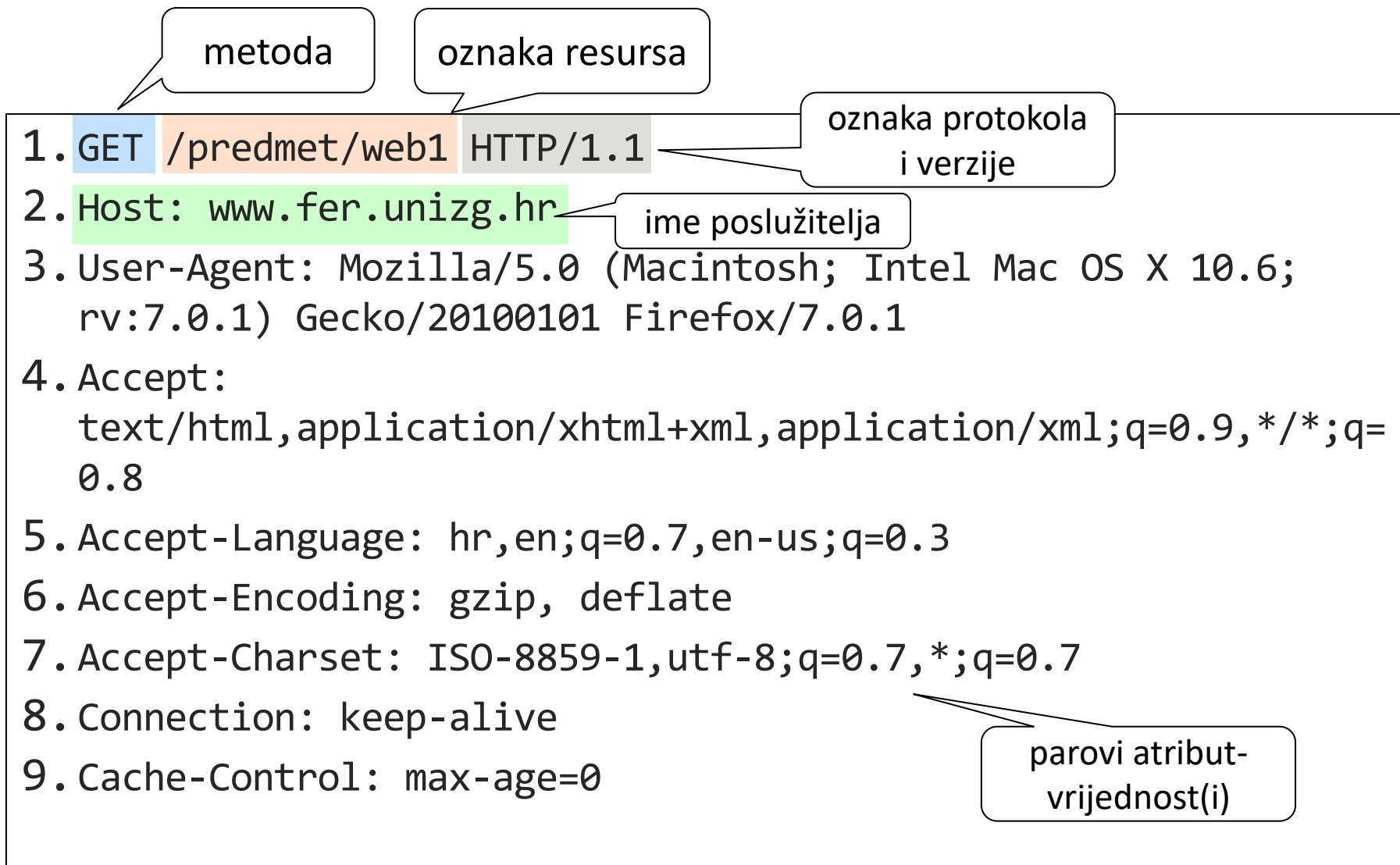
- pohranjuje sadržaj tj. resurse i podržava protokol HTTP
- npr. Apache HTTP Server, Nginx, Microsoft Internet Information Services



Part of F5



Primjer HTTP-zahtjeva



Primjer HTTP-odgovora

1. HTTP/1.1 200 OK statusni kôd
2. Date: Wed, 12 Oct 2011 08:19:32 GMT
3. Server: Apache/2.2.20 (FreeBSD) mod_ssl/2.2.20 OpenSSL/0.9.8q
mod_fcgid/2.3.6
4. Expires: Thu, 19 Nov 2020 08:52:00 GMT
5. Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0
6. Pragma: no-cache
7. P3P: CP="NOI CURa ADMa DEVa TAIa PSAa PSDa IVAa IVDa HISa OTPa OUR BUS IND
UNI COM NAV INT"
8. Set-Cookie: CMS=2p72ge55hqm92itadsfu83pc25; expires=Wed, 19-Oct-2011
20:19:32 GMT; path=/; domain=www.fer.unizg.hr; HttpOnly
9. Vary: Accept-Encoding
10. Transfer-Encoding: chunked
11. Content-Type: text/html; charset=utf-8
12.
13. d9e7
14. <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
15. <html xmlns="http://www.w3.org/1999/xhtml">
16.
17. <head>
18. <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

HTTP – prijenos (hiper)teksta ili resursa?

■ ~~Hypertext~~ Transfer Protocol

- Prenosi **resurs** čiji je sadržaj u raznim formatima (*hypermedia*)

■ Resurs

- datoteka, npr. u formatu HTML ili GIF
- podaci o datoteci, meta-podaci (*meta-data*)
- dio datoteke ili odsječak (*chunk*)
- višedijelni podaci (*multipart data*)
- rezultat obrade (*processing result*)
- ... **bilo što u smislu digitalnog sadržaja** što možemo imenovati, tj. možemo mu pridijeliti jedinstveni identifikator
- definicija iz RFC 2616: “umreženi podatkovni objekt ili usluga identificirana URI-jem”

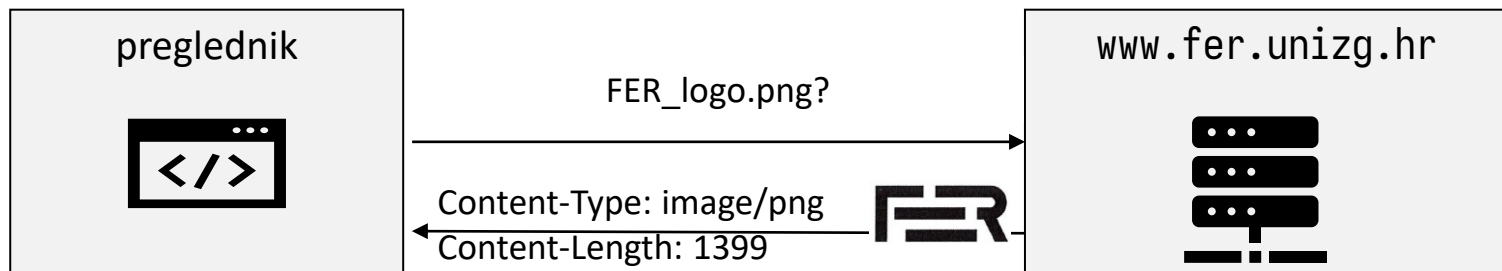
Kojeg je tipa resurs?

- Web-poslužitelj dodaje oznaku tipa svakom resursu, koristi posebnu oznaku u zaglavlju odgovora

Content-Type, npr.

- tekstualni dokument u HTML-formatu: `text/html`
 - slika u JPEG-u ili GIF-u: `image/jpeg`, `image/gif`
 - Microsoft PowerPoint: `application/vnd.ms-powerpoint`
 - video: `video/quicktime`, `video/mp4`
 - JS: `application/javascript`
- Preglednik koristi tip za određivanje vrste medijskog sadržaja resursa (Media Type)

Zašto preglednik treba znati tip/vrstu sadržaja resursa?

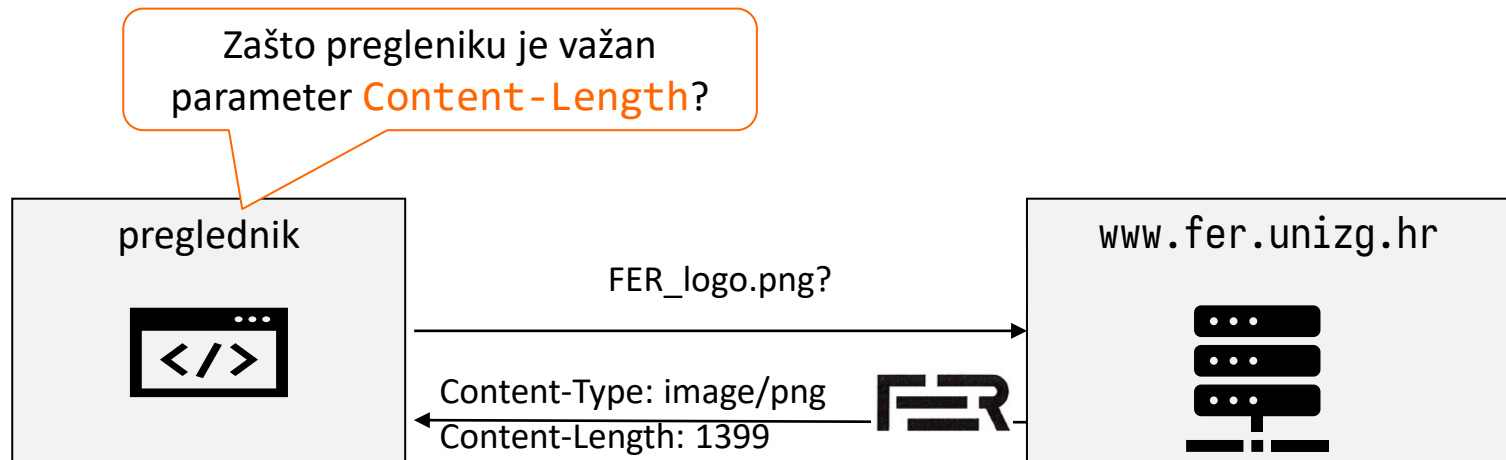


Kolika je duljina resursa?

- Web-poslužitelj dodaje informaciju o duljini resursa, koristi posebnu oznaku u zaglavlju odgovora

Content-Length

- Content-Length označava veličinu tijela poruke, u bajtovima, poslane pregledniku.



Media Type (1/2)

- Potječe iz proširenja elektroničke pošte za označavanje višemedijskih datoteka
 - *MIME Type - Multipurpose Internet Mail Extension*
- Služi za
 - označavanje tipa resursa tj. njegovog sadržaja i kodiranja
 - strukturiranje višedijelnih resursa (onih s više dijelova različite vrste)
- Normirana struktura: **tip/podtip** (*type/subtype*)
 - **application**/javascript, **application**/json, **application**/vnd.ms-excel, **application**/x-latex ...
 - **audio**/mpeg, **audio**/vnd.wave ...
 - **image**/png, **image**/jpeg ...
 - **multipart**/form-data ...
 - **text**/plain, **text**/html, **text**/css ...
 - **video**/mp4, **video**/avi ...

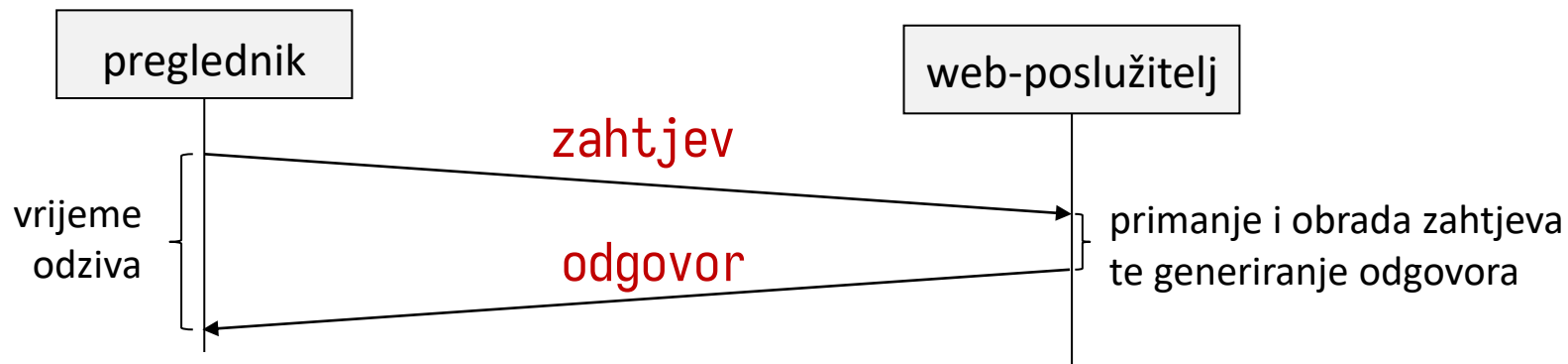
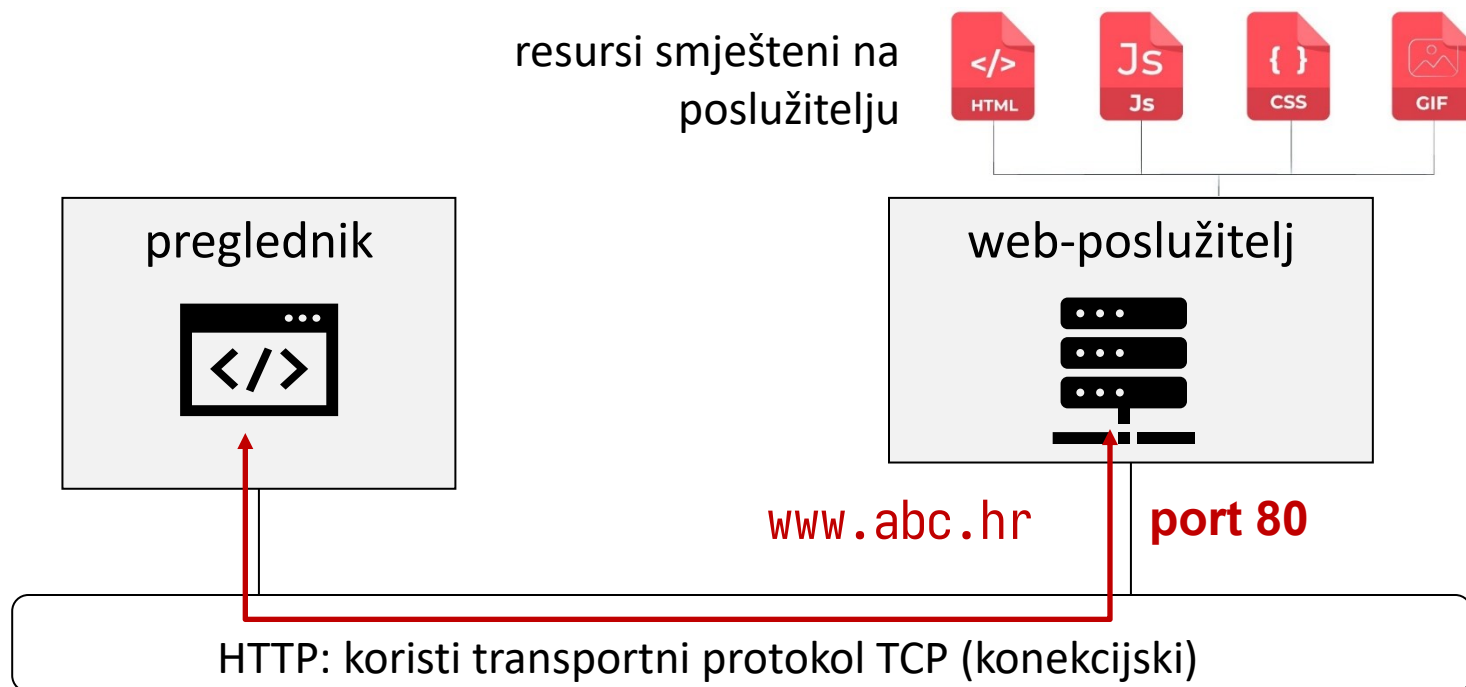
Media Type (2/2)

- Najvažniji standard koji definira Media Type
 - RFC 2046, “MIME: Media Types”, definira tipove i njihovu strukturu
 - Aktuelni popis: <https://www.iana.org/assignments/media-types/media-types.xhtml>
 - ~~text/javascript~~ → application/javascript
- Nazivi su normirani, a mogu se definirati i novi tipovi: o ovome se brine organizacija IANA (Internet Assigned Numbers Authority)
 - RFC 2048, “MIME: Registration Procedures”, definira kako registrirati nove tipove MIME
 - Prefiks podtipa **.vnd** – *vendor tree* (vlasnički proizvodi)
 - Prefiks podtipa **x-** - nenormirani




Osnovna obilježja komunikacije

- Klijent šalje zahtjev (**poruku**) poslužitelju i očekuje odgovor (**poruku**), poruke su **tekstualne**
- Poruka zahtjeva sadrži sve informacije potrebne za ispunjenje zahtjeva i generiranje odgovora od strane web-poslužitelja
- Ciklus zahtjev-odgovor = jedna **konverzacija**
 - slanjem odgovora konverzacija završava (ispunjen je zahtjev)
- Poslužitelj ne čuva stanje između dviju konverzacija s istim klijentom jer je HTTP **protokol bez očuvanja stanja (engl. *stateless protocol*)**
 - postoje posebne tehnike za identifikaciju klijenta i praćenje stanja

Izvedba komunikacije u mreži

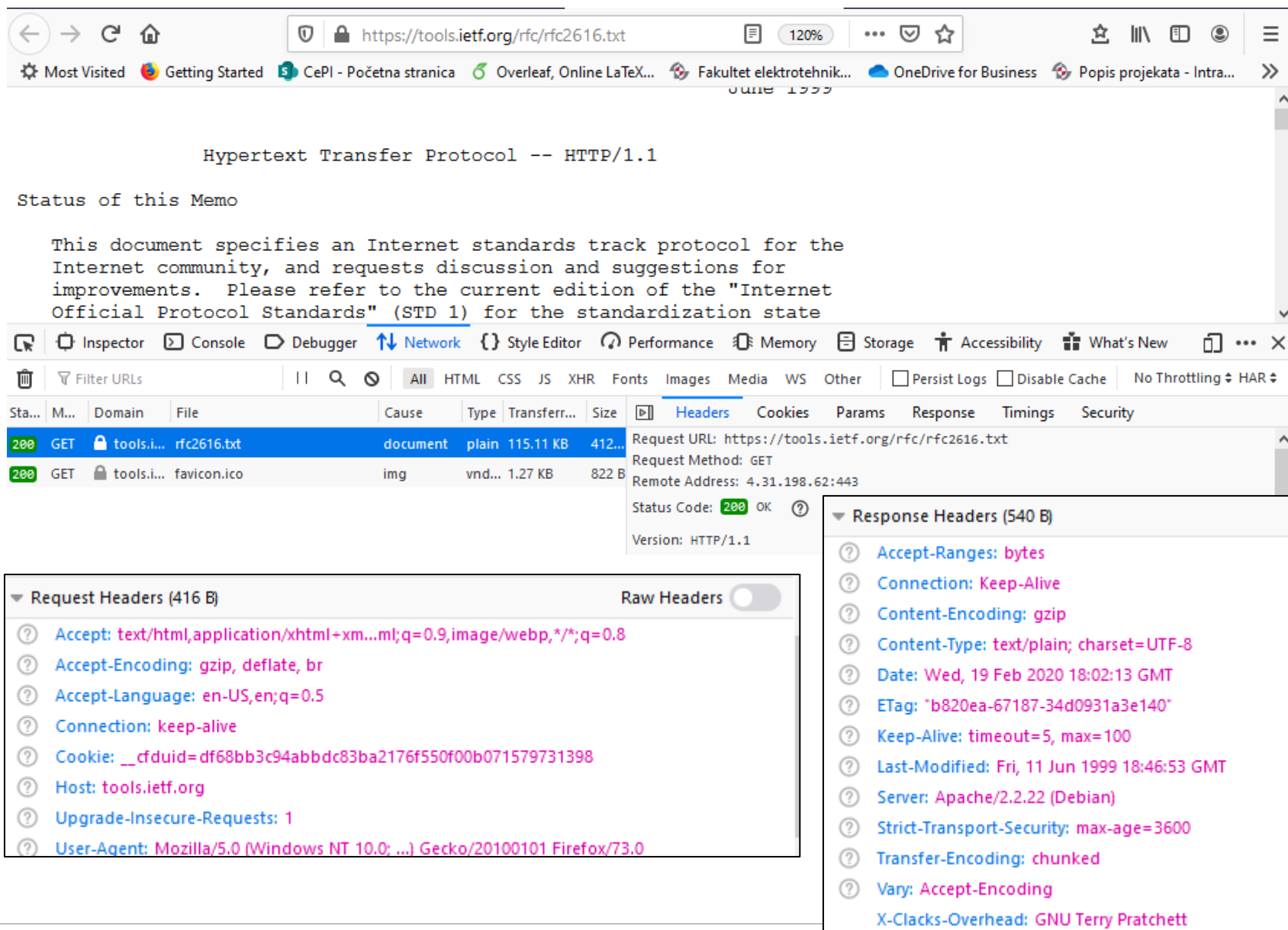


Kako možemo analizirati komunikaciju?

1. Firefox (Chrome) Developer Tools -> Network  
2. Postman 
 - Što možemo provjeriti i mijenjati vezano uz zahtjeve i odgovore?
 - HTTP metodu
 - URI
 - Kôd odgovora
 - Veličina dostavljenog resursa
 - Vrijeme potrebno za dohvat resursa (response time)
 - Zaglavlje zahtjeva
 - Zaglavlje odgovora
 - Cookies
 - itd.

Pažnja, ovo nije isto što i zaglavlje HTML stranice!

Analiza komunikacije 1: Firefox Developer Tools



The screenshot displays the Firefox Developer Tools interface with the Network tab selected. The browser window shows the URL `https://tools.ietf.org/rfc/rfc2616.txt`. The page content is the "Hypertext Transfer Protocol -- HTTP/1.1" document. The Network tab shows a list of requests, with the first request (GET `tools.ietf.org/rfc/rfc2616.txt`) selected. The Request Headers and Response Headers panels are open, showing detailed information about the HTTP transaction.

Request Headers (416 B)

- Accept: text/html,application/xhtml+xml;q=0.9,image/webp,*/*;q=0.8
- Accept-Encoding: gzip, deflate, br
- Accept-Language: en-US,en;q=0.5
- Connection: keep-alive
- Cookie: __cfduid=df68bb3c94abbd83ba2176f550f00b071579731398
- Host: tools.ietf.org
- Upgrade-Insecure-Requests: 1
- User-Agent: Mozilla/5.0 (Windows NT 10.0; ...) Gecko/20100101 Firefox/73.0

Response Headers (540 B)

- Accept-Ranges: bytes
- Connection: Keep-Alive
- Content-Encoding: gzip
- Content-Type: text/plain; charset=UTF-8
- Date: Wed, 19 Feb 2020 18:02:13 GMT
- ETag: "b820ea-67187-34d0931a3e140"
- Keep-Alive: timeout=5, max=100
- Last-Modified: Fri, 11 Jun 1999 18:46:53 GMT
- Server: Apache/2.2.22 (Debian)
- Strict-Transport-Security: max-age=3600
- Transfer-Encoding: chunked
- Vary: Accept-Encoding
- X-Clacks-Overhead: GNU Terry Pratchett

Analiza komunikacije 1: Postman



The screenshot shows the Postman application window. The top bar includes the Postman logo, menu items (File, Edit, View, Help), and workspace controls (My Workspace, Invite). The left sidebar shows a search filter, tabs for History, Collections, and APIs, and a 'Save Responses' toggle. The main area displays an 'Untitled Request' for the URL `https://tools.ietf.org/rfc/rfc2616.txt`. The request method is GET. The 'Body' tab is selected, showing 'This request does not have a body'. The 'Headers' tab is also visible, showing 16 headers. The response status is 200 OK, with a time of 1630ms and a size of 412.91 KB. The response body is not visible in the screenshot.

KEY	VALUE
Date	Wed, 08 Apr 2020 11:31:38 GMT
Server	Apache/2.2.22 (Debian)
Last-Modified	Fri, 11 Jun 1999 18:46:53 GMT
ETag	"b820ea-67187-34d0931a3e140"
Accept-Ranges	bytes
Vary	Accept-Encoding
Content-Encoding	gzip
Strict-Transport-Security	max-age=3600

Za one koji žele naučiti nešto više postoje tutorijali na webu: npr. za početak [Postman API tutorial for beginners](#)

Razvoj protokola HTTP

- HTTP 0.9
 - 1991. g., kasnije dokumentiran u RFC 1945
 - inicijalna inačica, omogućuje prijenos teksta, iznimno jednostavna
- HTTP 1.0 (tzv. World Wide Wait)
 - 1996. g., RFC 1945
 - prva široko uporabljiva inačica, omogućuje prijenos različitih vrsta podataka, svako web-sjedište koristi jedinstvenu IP adresu
- **HTTP 1.1**
 - 1997. g., nadogradnja v1.0 RFC 2068
 - 1999. g., **RFC 2616 (aktualna verzija protokola)**
 - kontrola veze (perzistentne konekcije, zahtjevi i odgovori se prenose istom TCP konekcijom, tzv. „pipelining”), upravljanje priručnim spremištem (engl. *cache*), *virtual host*, itd.
- HTTP/2
 - svibanj 2015. - RFC 7540, fokus na poboljšanje performanci
 - danas je i dalje najraširenija inačica HTTP 1.1

HTTPS

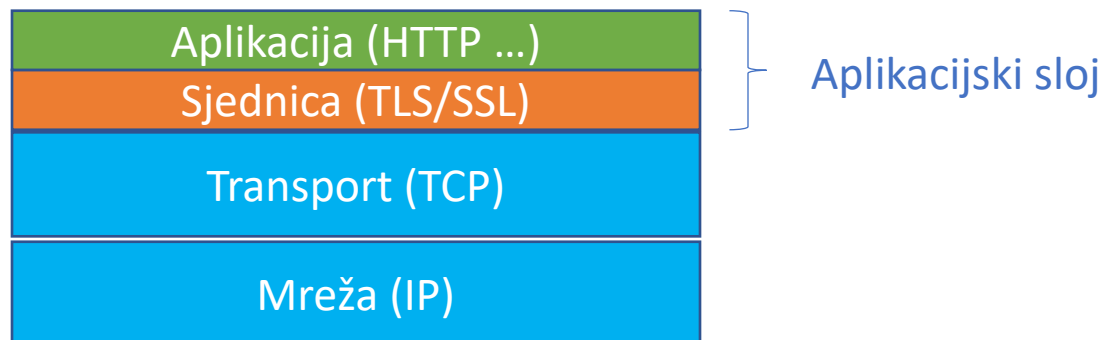


- Hypertext Transfer Protocol Secure (*HTTPS*)
- „sigurna” verzija protokola HTTP
- koriste se isti zahtjevi i odgovori protokola HTTP, ali se poruke šifriraju (može ih čitati samo klijent i poslužitelj)
- koristi protokol TLS (koristi vrata 443), to je evoluirani protokol SSL
- Procjenjuje se da HTTPS danas koristi gotovo 80% web-sjedišta (izvor: <https://w3techs.com/technologies/details/ce-httpsdefault>)

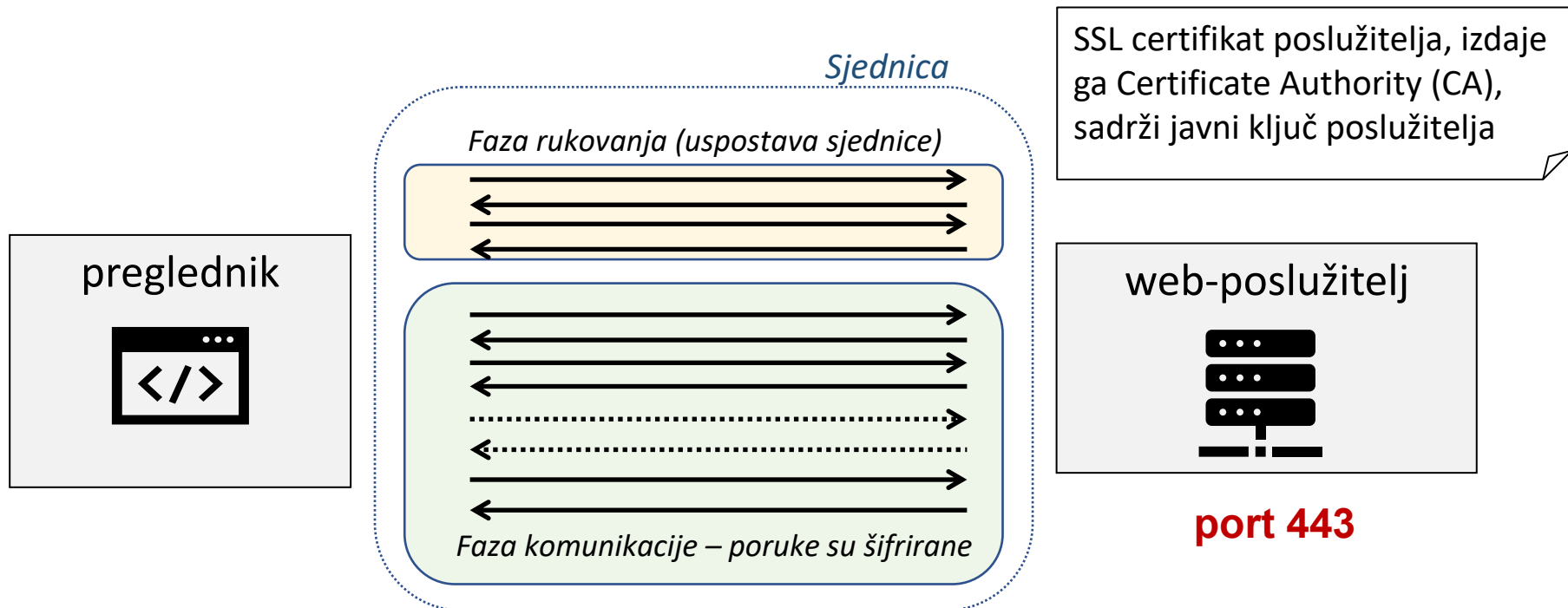
<https://www.fer.unizg.hr>

Svojstva komunikacije ostvarene TLS-om

- Povjerljivost prenesenih podataka (treća strana ne može čitati poruke)
 - simetrična kriptografija za šifriranje poruka
 - generira se **ključ sjednice**: jednokratni simetrični ključ za svaku sjednicu
- Autentičnost strana u komunikaciji (mogućnost provjere identiteta, nema lažnog predstavljanja)
 - temeljena na javnom i tajnom ključu te certifikatima
 - nužna za poslužitelj, opcionalna za klijenta
- Integritet prenesenih podataka (treća strana ne može mijenjati poruke)



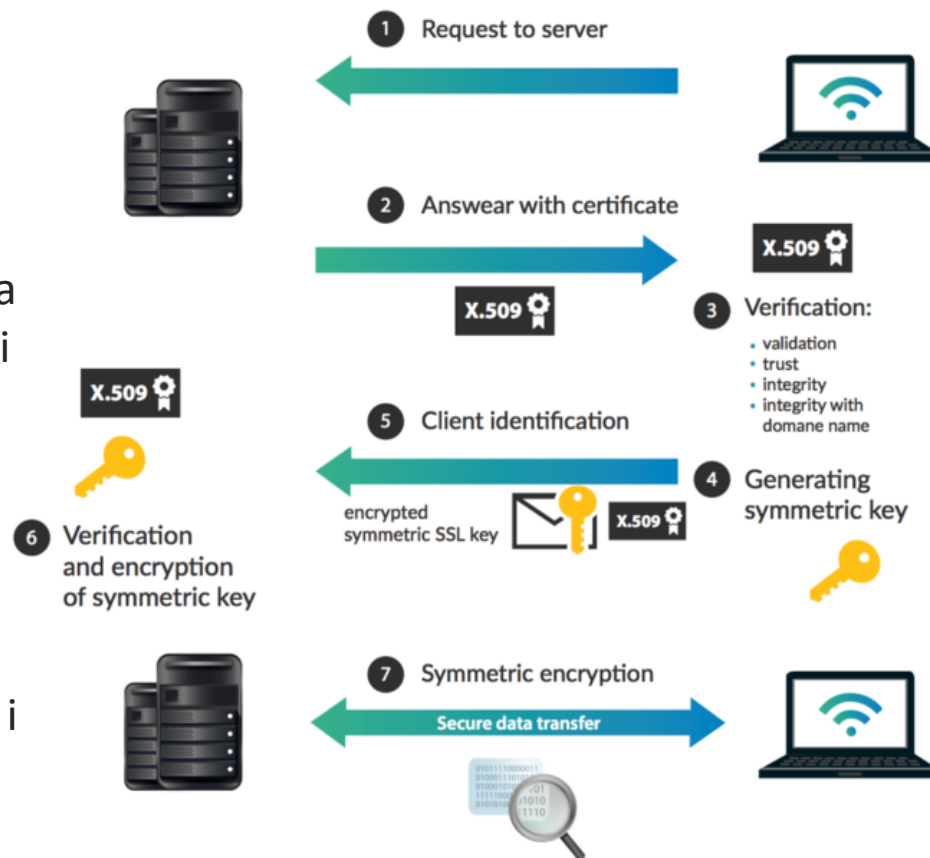
Uspostava sigurne komunikacije korištenjem TLS-a



- Dvije osnovne faze:
 - Faza rukovanja – dogovor o parametrima sigurnog kanala
 - Faza komunikacije – dvosmjerna komunikacija kroz siguran komunikacijski kanal (koristi se **ključ sjednice** za šifriranje poruka)

Detalji o tijeku komunikacije (pojednostavljeno)

1. Preglednik šalje zahtjev poslužitelju
2. Poslužitelj u odgovoru dostavlja svoj javni ključ i certifikat
3. Preglednik provjerava valjanost javnog ključa i certifikata
4. Ako su certifikat i javni ključ poslužitelja valjani, preglednik generira jednokratni simetrični **ključ sjednice**
5. Preglednik šalje u odgovoru **ključ sjednice** šifriran javnim ključem poslužitelja
6. Poslužitelj prima i dešifrira **ključ sjednice** pomoću svog privatnog ključa i šalje u odgovoru **ključ sjednice** šifriran samim sobom
7. **Ključ sjednice** koristi se za daljnje šifriranje poruka između preglednika i poslužitelja



Izvor: <https://faun.pub/ssl-decoded-eee6f2a94f44>

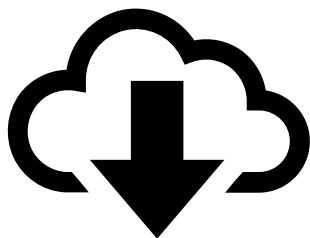
Primjena HTTP-a danas

Raširena primjena u nizu
različitih internetskih
servisa

Youtube koristi Dynamic
Adaptive Streaming over HTTP



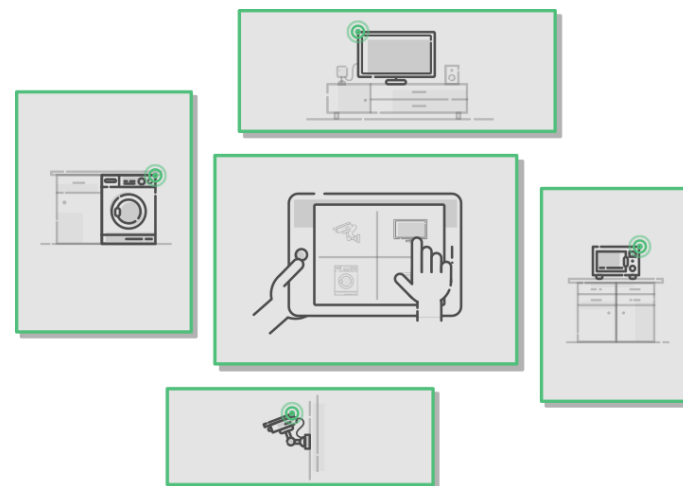
Cloud computing



REST



Internet of Things



Sadržaj predavanja

- Uvod
 - Osnove protokola HTTP
 - Web-poslužitelj i web-preglednik
 - Media Type i tijek komunikacije
 - HTTPS
- Identifikacija resursa
 - URI, URL, URN
 - sintaksa URI-ja
 - relativni i apsolutni URI
- Poruke protokola HTTP
 - zahtjev i odgovor
 - metode zahtjeva i kôdovi odgovora
- Priručna spremišta

Identifikacija resursa

- Lokalna identifikacija resursa
 - Put u datotečnom sustavu lokalnog računala
 - Datotečni sustavi unutar neke lokalne mreže
 - Resurs ima jedinstveno ime unutar lokalnog sustava
- Problemi jednoznačne identifikacije resursa na globalnoj razini
 - Postoji li jedan “**globalni datotečni sustav**” koji bi određivao jedinstvenu identifikaciju resursa?
 - različite vrste resursa i moguće akcije nad njima
 - različite lokacije resursa i načini pristupa resursima
- Potreban je odgovor na sljedeća pitanja:
 - Kako globalno identificirati resurs?
 - Kako pronaći taj resurs?
 - Kako pristupiti resursu?

Uniform Resource Identifier (URI)

URI - **U**niform **R**esource **I**dentifier

(uniformni identifikator resursa)

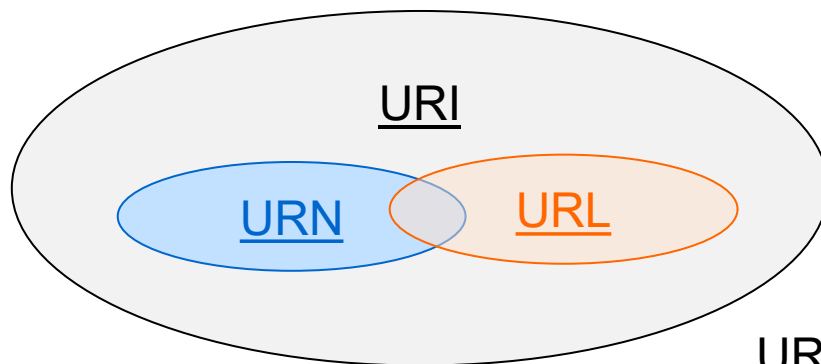
- **uniformni**: jednoobrazni način zapisa, tj. propisana je struktura zapisa
- **identifikator**: sadrži informaciju nužnu za razlikovanje identificiranog resursa od svih ostalih (\neq identitet!)
- **resurs**: informacijski izvor; “*bilo što*” što se može identificirati URI-jem

*Pojam URI-ja je središnji pojam u arhitekturi WWW-a.
World Wide Web Consortium (W3C) definira WWW kao “informacijski prostor u kojem su objekti od interesa identificirani URI-jima”.*

URI, URL, URN

■ URN – Uniform Resource Name

- definira **naziv** resursa koji garantira **jedinstvenost** i **trajnost** identifikacije
- slabo se koristi u praksi
- npr. *urn:ietf:rfc:2616* (jedinstveni naziv IETF-ovog RFC-a 2616)



■ URL – Uniform Resource Locator

- sadrži informaciju o **lokaciji** resursa
- najčešći oblik URI-ja
- **NE** garantira jedinstvenost i trajnost identifikacije
- neformalno se koristi kao sinonim za „adresu” resursa
- npr. URL RFC-a 2616

<http://www.ietf.org/rfc/rfc2616.txt>

URL i URN su podskupovi URI-ja

Primjeri URI-ja

- `ftp://ftp.is.co.za/rfc/rfc1808.txt`
- `http://www.ietf.org/rfc/rfc2396.txt`
- `ldap://[2001:db8::7]/c=GB?objectClass?one`
- `mailto:John.Doe@example.com`
- `news:comp.infosystems.www.servers.unix`
- `tel:+1-816-555-1212`
- `telnet://192.0.2.16:80/`
- `urn:oasis:names:specification:docbook:dtd:xml:4.1.2`

Preuzeto iz [IETF-specifikacije URI-ja](#)

RFC 3986

Analiza URI-ja HTTP-a (1/2)

`http://www.fer.unizg.hr/predmet/web1`

shema

naziv poslužitelja
(*host name*)

oznaka resursa na
web-poslužitelju

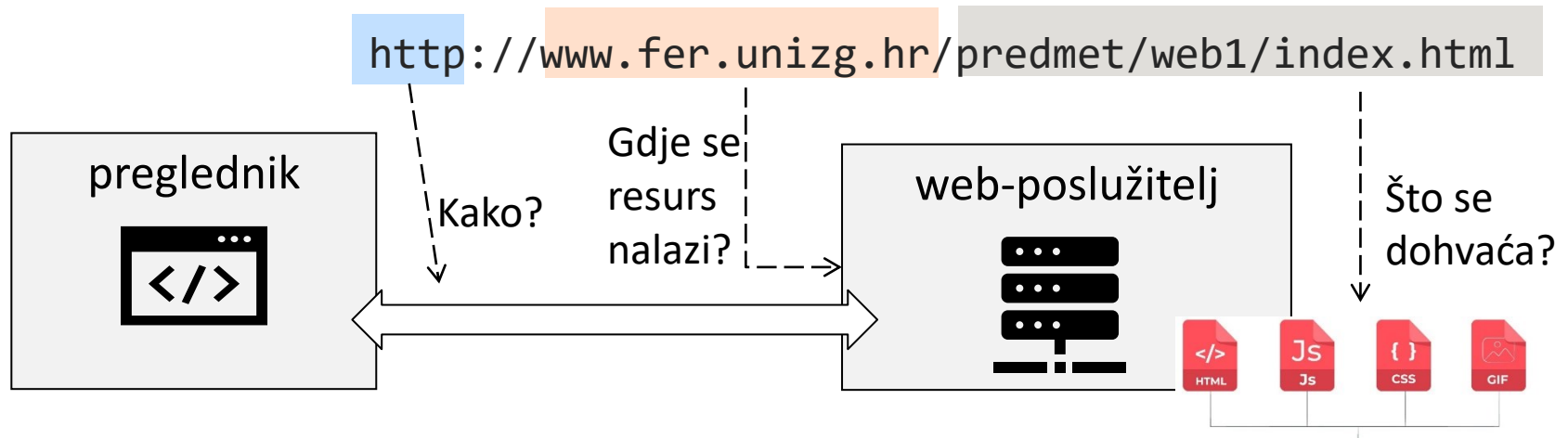
shema URI-ja definira način pristupa resursu (npr. protokol HTTP)

put i oznaka resursa - analizira ga web-poslužitelj kako bi dohvatio zadani resurs

host name - može sadržavati potpuno ime ili IP adresu poslužitelja

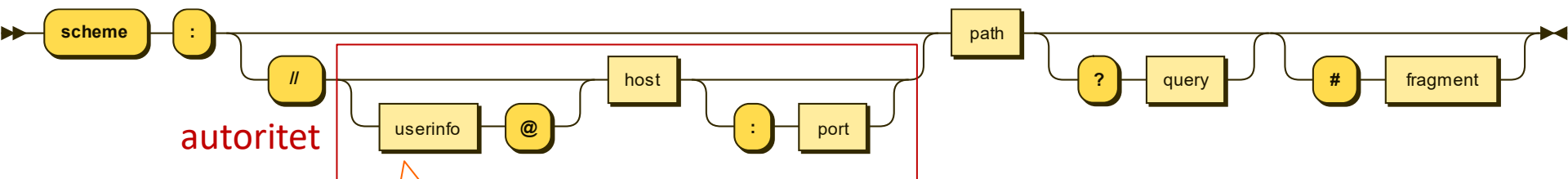
Analiza URI-ja HTTP-a (2/2)

- Shema URI-ja: koristi klijent, definira kako pristupiti resursu (protokol)
- *Host name*: koristi klijent, definira gdje je resurs smješten (poslužitelj)
- Put: koristi poslužitelj, definira koji je lokalni resurs zatražen



Opća sintaksa URI-ja (RFC 3986)

- Sastoji se od hijerarhijskog niza komponenti:
 - **shema** (engl. *scheme*)
 - **autoritet** (engl. *authority*)
 - **put** (engl. *path*)
 - **upit** (engl. *query*)
 - **fragment** (engl. *fragment*)



Obično se ne koristi

Izvor: OmenBreeze - Own work, CC BY-SA 4.0,
<https://commons.wikimedia.org/w/index.php?curid=82827943>

Shema URI-ja

- svaki **apsolutan URI** je složen ovako:

<shema>: <dio specifičan za shemu>

- **shema** je zaseban potprostor imena
 - npr. “**http**”, “**ftp**”, “**mailto**”, “**urn**”, “**file**”, “**news**”
 - neki su dobili ime po protokolu, ali to **ne znači da je shema = protokol**
- sintaksa ostatka URI-ja ovisi o shemi, drugačija za svaku shemu

Uobičajen ustroj URI-ja

- većina shema niz *<dio specifičan za shemu>* ustrojava ovako:

// <*autoritet*> <*put*> ? <*upit*>

- **autoritet** - predstavlja (logički) poslužitelj
- **put** - put kroz hijerarhiju (po uzoru na datotečni sustav)
 - počinje kosom crtom i nadalje je sastavljen od segmenata odvojenih kosom crtom, npr:

/

/segment1/2/3

/mark-twain/roman/tom-sawyer

- **upit** - popis parametara u proizvoljnom redoslijedu

Shema *http*

- shema *http* definira autoritet ovako:

$\langle \textcolor{red}{autoritet} \rangle = \langle \textcolor{purple}{host} \rangle : \langle \textcolor{gray}{vrata} \rangle$

- primjeri:

`http://www.fer.unizg.hr/?@=1d2w9#news_8980`

`http://www.google.com:81/search?q=web`

- broj **vrata** je neobavezan dio autoriteta (podrazumijeva se tcp/80)
- **put** je neobavezan
- **upit** se pojavljuje iza prvog upitnika, neobavezan je
 - upit može imati više segmenata
 - segmenti upita se odvajaju znakom **&**
 - svaki segment je tipično par (ime, vrijednost) odvojen znakom **=**

Fragment

- URI-ju se na kraj može dodati identifikator fragmenta odvojen znakom “#”, npr.

```
http://www.fer.unizg.hr/predmet/web1/materijali/HTTP.html#b3
```

- u HTML dokumentu “HTTP.html” postoji *bookmark* s nazivom “b3” na čiji se vrh pozicionira prozor preglednika
- bookmark se može dodati potencijalno u svaki element, potrebno je definirati globalni atribut **id**
- identifikator fragmenta je smislen samo ako se koristi pri akciji dohvaćanja resursa u kojem se nalazi odgovarajući fragment

```
<h2 id="b1">Poglavlje prvo</h2>
<p> ... </p>

<h2 id="b2">Poglavlje drugo</h2>
<p> ... </p>

<h2 id="b3">Poglavlje treće</h2>
<p> ... </p>
```

Sintaksa - posebni znakovi

- URI izravno koristi znakove iz ograničenog skupa znakova ASCII
- ostali ASCII znakovi se predstavljaju u posebnom “*escaped*” obliku: znak “%” i dvoznamenkasti heksadecimalni kôd

povećanje od +30%

- “ć” ima kôd E6, razmak 20, a “%” 25

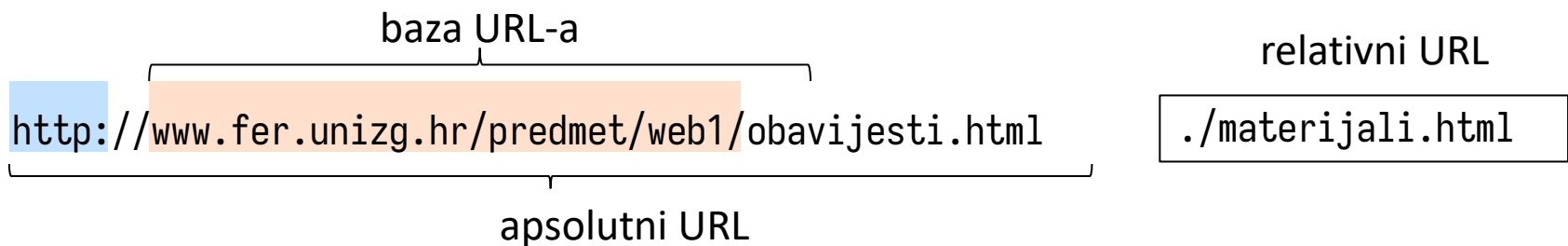
pove%E6anje%20od%20+30%25

- U URI-ju se razmak kodira i znakom + (**prisjetite se obrazaca**)
- U JavaScriptu se može koristiti **encodeURIComponent()** za kodiranje URI-a.

Referenca za kodiranje znakova: [HTML URL Encoding Reference](#)

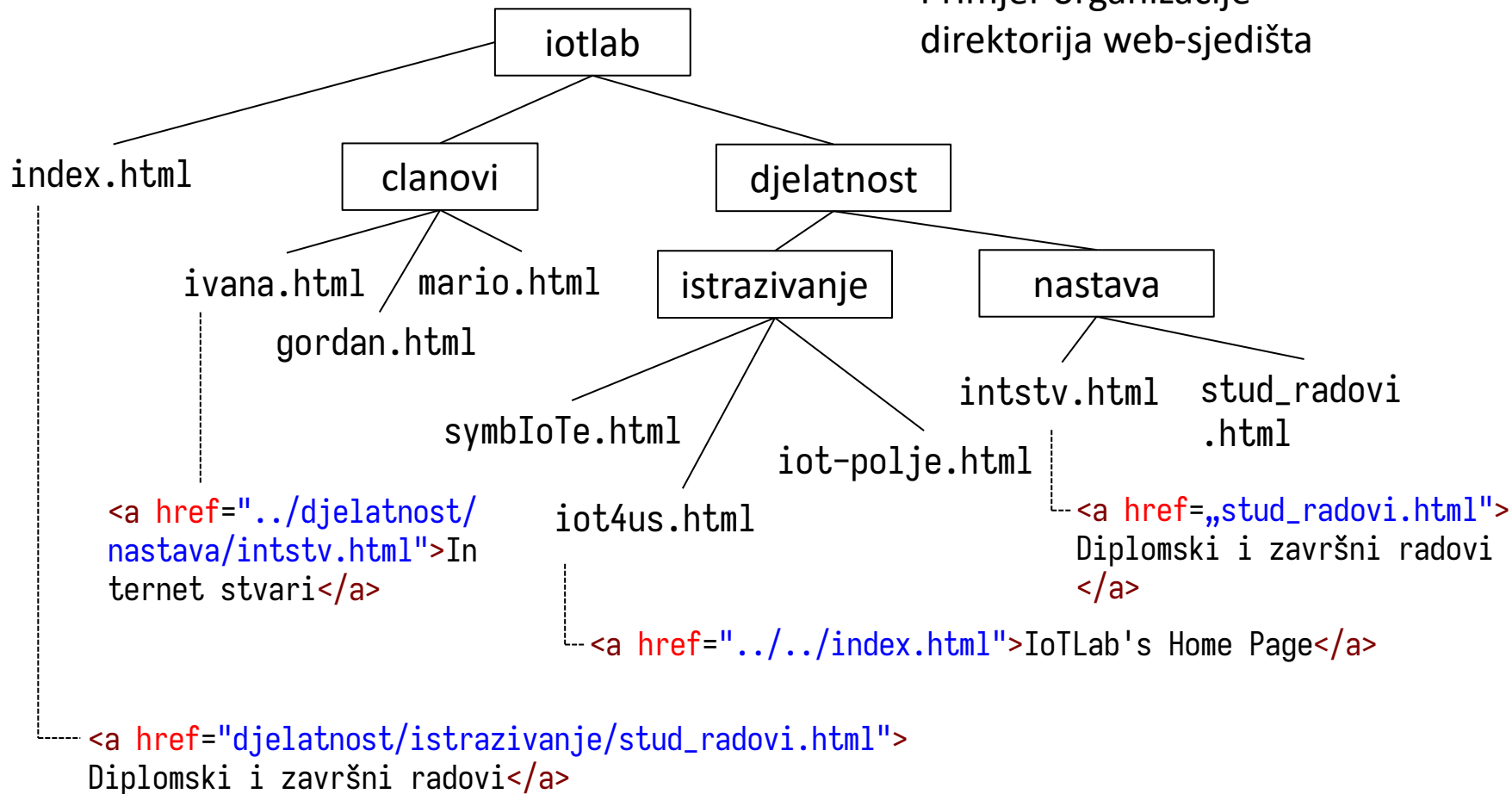
Relativni i apsolutni URL-ovi

- **Apsolutni URL**: prenosi potpunu informaciju za pristup resursu, uvijek počinje sa < *shema* > :
- **Relativni URL** je nepotpun, koristi se kao pogodan skraćeni zapis URL-a
- Baza URL-a se zaključuje iz URL-a dokumenta u kome se pojavljuje relativni URL ili je definirana u elementu zaglavlja <base>
 - Oznake "." i ".." (engl. dot-segments), relativne u odnosu na put



Relativni URL (primjer)

Primjer organizacije
direktorija web-sjedišta



Sadržaj predavanja

- Uvod
 - Osnove protokola HTTP
 - Web-poslužitelj i web-preglednik
 - Media Type i tijek komunikacije
 - HTTPS
- Identifikacija resursa
 - URI, URL, URN
 - sintaksa URI-ja
 - relativni i apsolutni URI
- Poruke protokola HTTP
 - zahtjev i odgovor
 - metode zahtjeva i kôdovi odgovora
- Priručna spremišta

Poruke protokola HTTP

zahtjev

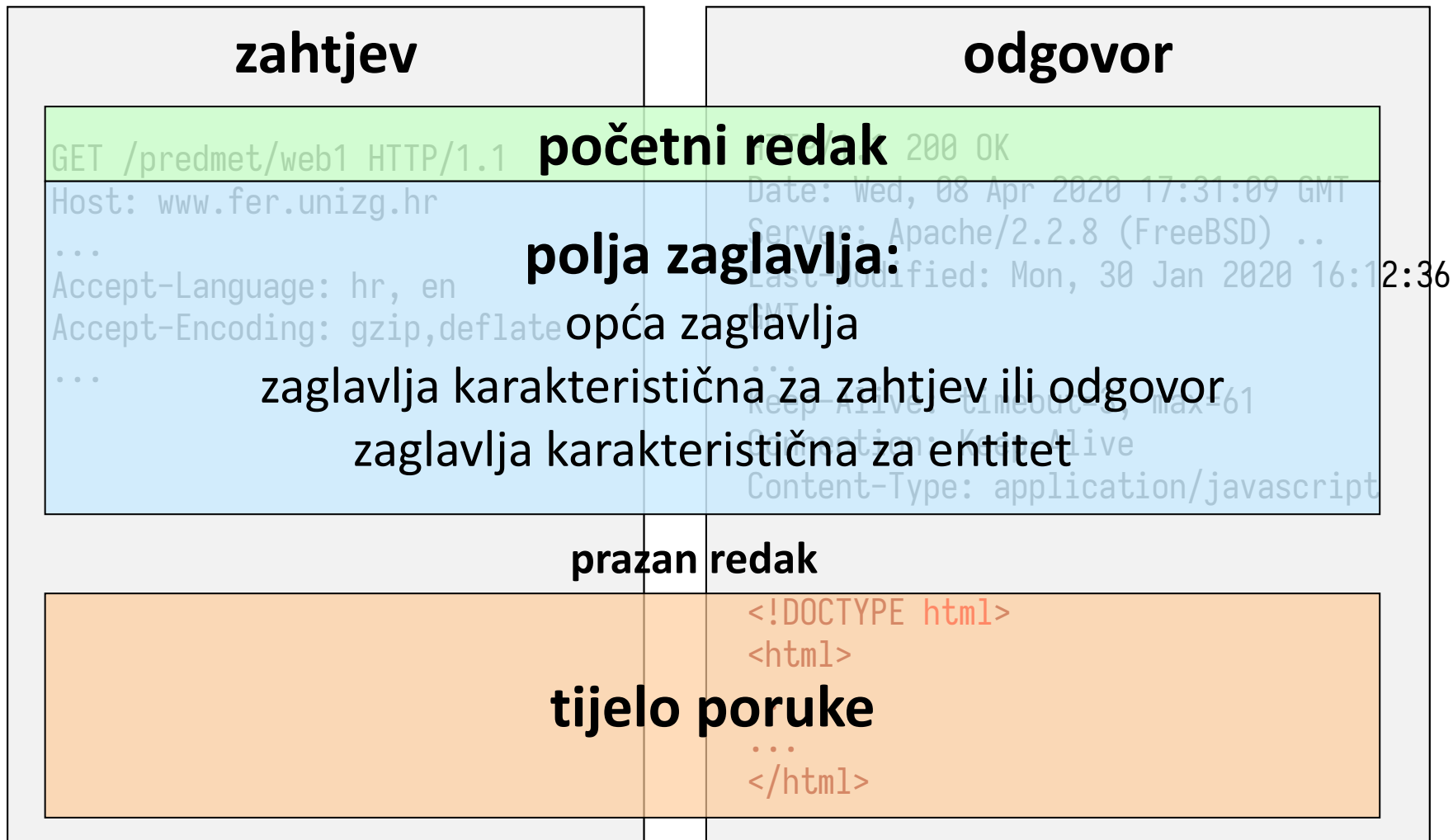
```
GET /predmet/web1 HTTP/1.1
Host: www.fer.unizg.hr
...
Accept-Language: hr, en
Accept-Encoding: gzip,deflate
...
```

odgovor

```
HTTP/1.1 200 OK
Date: Wed, 08 Apr 2020 17:31:09 GMT
Server: Apache/2.2.8 (FreeBSD) ..
Last-Modified: Mon, 30 Jan 2020 16:12:36 GMT
...
Keep-Alive: timeout=3, max=61
Connection: Keep-Alive
Content-Type: application/javascript

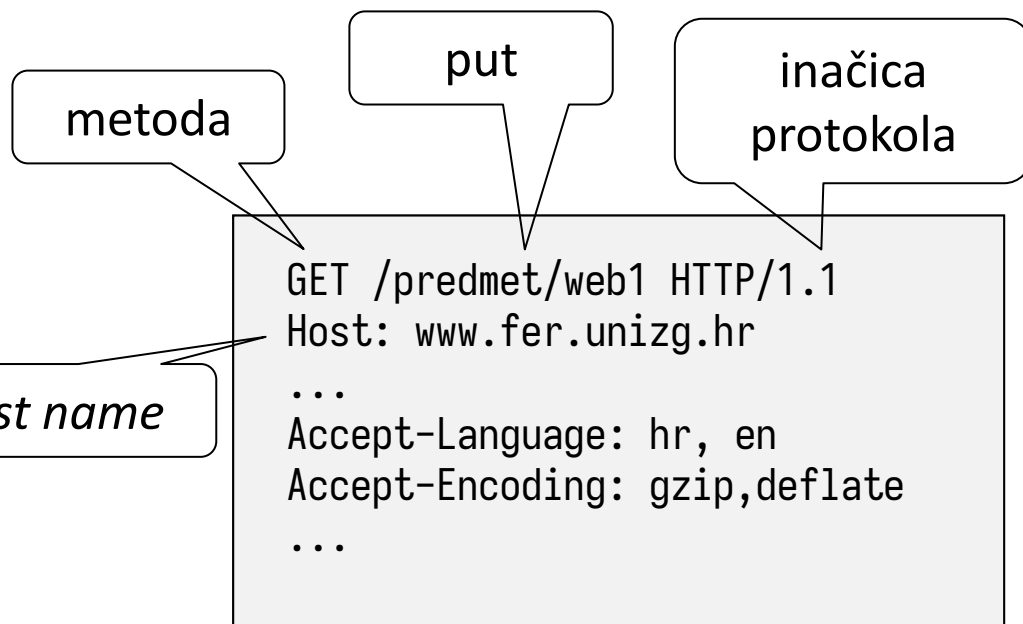
<!DOCTYPE html>
<html>
...
...
</html>
```


Poruke protokola HTTP

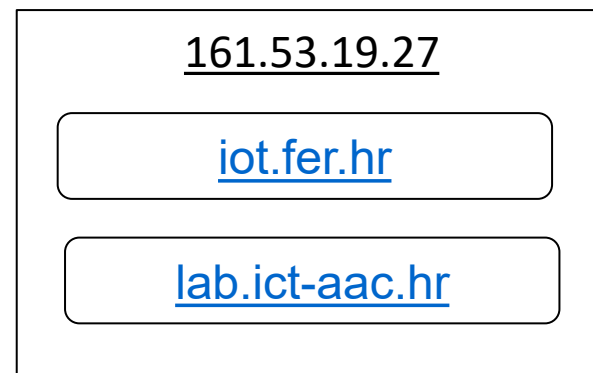


Zahtjev HTTP-a

- Sadrži **metodu HTTP-a**
 - definira operaciju nad resursom
- Definira **URI** resursa
 - jednoznačno određuje resurs: *host* + put
- Dodatna oznaka **inačice protokola**
 - različiti zahtjevi na strukturu poruke za različite inačice protokola
 - kompatibilnost metoda prema starijim inačicama



Virtual host: više web-sjedišta na jednom računalu



Metode zahtjeva

- HTTP/1.1 definira 8 metoda i omogućuje dodavanje novih (*extensions*):
 - OPTIONS - informiranje o mogućnostima poslužitelja i komunikacijskim zahtjevima povezanim s resursom
 - GET - za dohvaćanje sadržaja resursa (najčešća metoda)
 - HEAD - za dohvaćanje podataka o resursu (npr. provjera da postoji, veličina)
 - POST - aktiviranje resursa (npr. slanje podataka obrazaca)
 - PUT - postavljanje entiteta (npr. promjena podataka - kod REST-a)
 - DELETE - brisanje resursa (npr. kod REST-a)
 - TRACE - za dijagnostiku
 - CONNECT - pokreće dvosmjernu komunikaciju s traženim resursom putem proxy-ja prema web-sjedištu koje koristi HTTPS

Odgovor HTTP-a

- Struktura:
 - *Response-Line = HTTP-Version Status-Code Reason-Phrase CRLF*
- Određuje **rezultat zahtjeva** HTTP-a
 - zadan numerički (kôd rezultata, *Status-Code*)
i tekstualno (opis, *Reason-Phrase*)
- Oznaka inačice protokola
 - može (ali ne mora) biti ista kao inačica protokola zahtjeva
- Primjer:

• HTTP/1.1 404 Not Found

Inačica
protokola

Kôd
rezultata

Opis
rezultata

Kôd odgovora

- Troznamenkasti broj
- Grupe prema prvoj znamenki:
 - **1xx: informativni** (*Informational*)
 - zahtjev primljen, nastavak rada
 - **2xx: uspjeh** (*Success*)
 - zahtjev je uspješno primljen, protumačen i prihvaćen
 - **3xx: preusmjerenje** (*Redirect*)
 - za ispunjenje zahtjeva potrebne su daljnje akcije
 - **4xx: greška klijenta** (*Client Error*)
 - zahtjev nije ispravno oblikovan ili se ne može ispuniti
 - **5xx: greška poslužitelja** (*Server Error*)
 - poslužitelj nije uspio ispuniti inače ispravan zahtjev

Kôd odgovora - primjeri

- 100: Continue
- 200: OK
- 301: Moved Permanently (tj. *permanent redirect*)
- 400: Bad Request
- 401: Unauthorized
- 403: Forbidden
- 404: Not Found
- 500: Internal Server Error
- 501: Not Implemented
- 503: Service Unavailable
- 505: HTTP Version not supported

Metoda GET

- **GET** - metoda za dohvaćanje resursa
 - znači “pribavi reprezentaciju tog resursa”
 - aktivira se prilikom upisivanja adrese u preglednik ili odabira poveznice (engl. *link*)
- ako poslužitelj ima zahtijevani resurs, vraća ga u tijelu odgovora; inače vraća pogrešku
- moguće je koristiti djelomični GET (*partial GET*) - dohvaća se samo dio datoteke (u zaglavlju zahtjeva definira se raspon *Range*)

Analizirajmo primjer zahtjeva GET

GET

<https://tools.ietf.org/rfc/rfc2616.txt>

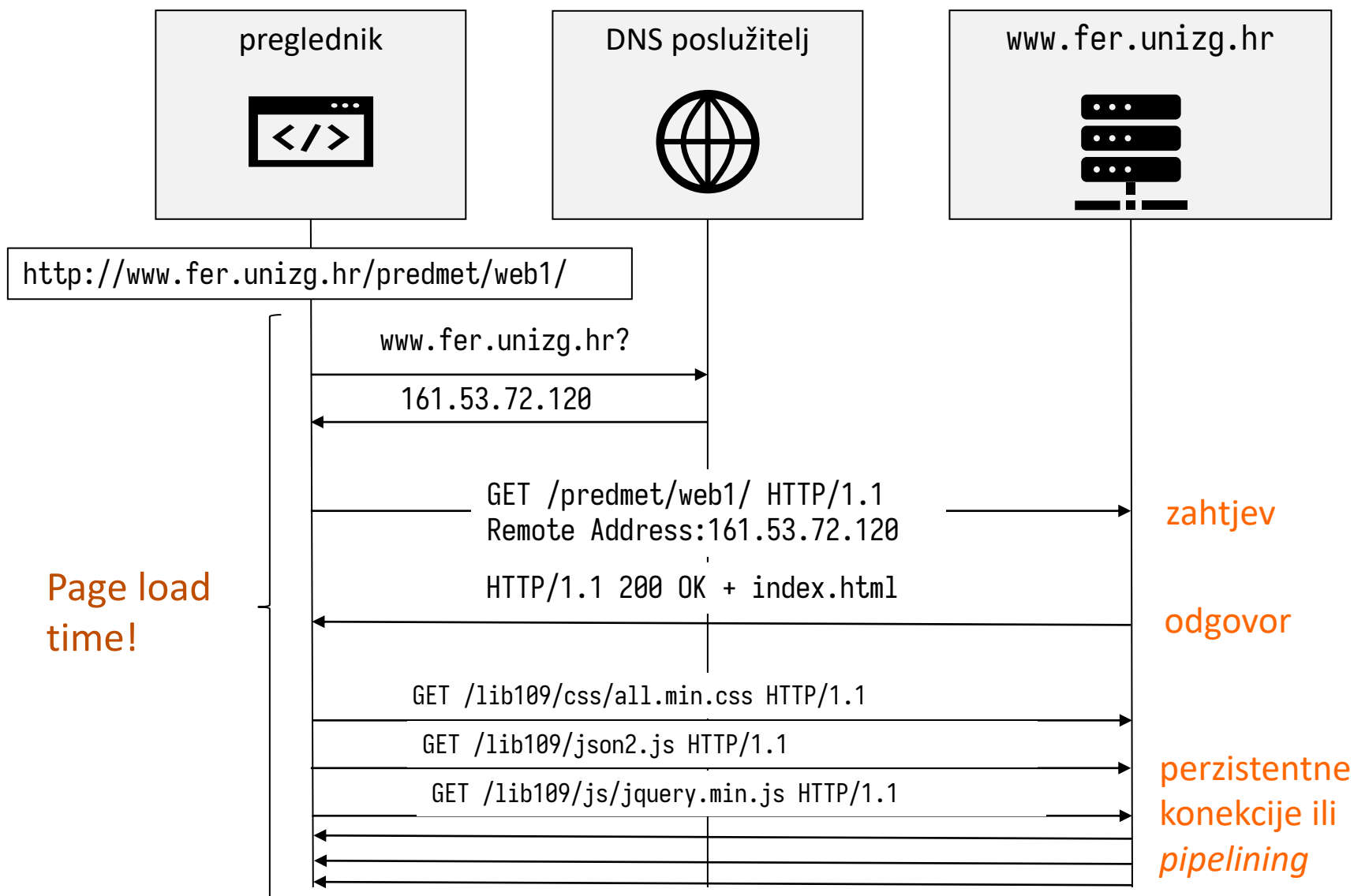
1. GET /rfc/rfc2616.txt HTTP/1.1
2. Host: tools.ietf.org
3. User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:74.0) Gecko/20100101 Firefox/74.0
4. Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
5. Accept-Language: en-US,en;q=0.5
6. Accept-Encoding: gzip, deflate, br
7. Connection: keep-alive
8. Upgrade-Insecure-Requests: 1

Analizirajmo primjer odgovora na GET

1. HTTP/1.1 200 OK
2. Date: Wed, 08 Apr 2020 13:47:34 GMT
3. Server: Apache/2.2.22 (Debian)
4. Last-Modified: Fri, 11 Jun 1999 18:46:53 GMT
5. ETag: "b820ea-67187-34d0931a3e140"
6. Accept-Ranges: bytes
7. Vary: Accept-Encoding
8. Content-Encoding: gzip
9. Strict-Transport-Security: max-age=3600
10. X-Frame-Options: SAMEORIGIN
11. X-Xss-Protection: 1; mode=block
12. X-Content-Type-Options: nosniff
13. X-Clacks-Overhead: GNU Terry Pratchett
14. Keep-Alive: timeout=5, max=100
15. Connection: Keep-Alive
16. Transfer-Encoding: chunked
17. Content-Type: text/plain; charset=UTF-8

18. <!DOCTYPE html>
...HEAD + BODY

Primjer komunikacije sa složenijim web-sjedištem



Koliko se zahtjeva generira prilikom pristupa stranici predmeta?

Razvoj programske potpore za web

Naslovica / Razvoj programske potpore za web

Razvoj programske potpore za web

Razvoj programske potpore za web

Anketa

Na ovoj stranici trenutno nije odabrana niti jedna anketa!

Razvoj programske potpore za web

Opis predmeta

Sadržaj predmeta obuhvaća osnove web-arhitektura, protokola i standarda, te programiranje u jezicima HTML5, JavaScript i CSS. Studenti će naučiti osnovne koncepte i tehnologije razvoja programske potpore za web te će razumjeti obilježja i način rada protokola HTTP i kako ga programski koristi. Studenti će naučiti oblikovati i implementirati cjelovitu dinamičku web-aplikaciju koja uključuje klijentski i poslužiteljski dio te će moći identificirati i ocijeniti obilježja razvijenog rješenja.

Oblici nastave

Predavanja
Nastava na predmetu organizirana je u dva nastavna ciklusa. Prvi nastavni ciklus sastoji se od 7 tjedana nastave i međuispita dok drugi ciklus sadrži 6 tjedana nastave i završni ispit. Nastava se provodi kroz 15 tjedana s tjednim opterećenjem od 3 školska sata.

Laboratorij
Studenti samostalno rješavaju 4 praktična zadatka kroz laboratorijske vježbe. HTML, CSS, JavaScript, Node.js.

Način ocjenjivanja

Vrsta provjere	Kontinuirana nastava		Ispitni rok	
	Prag	Udio u ocjeni	Prag	Udio u ocjeni
Laboratorijske vježbe	15 %	32 %	15 %	32 %
Međuispit: Pismeni	0 %	30 %	0 %	
Završni ispit: Pismeni	40 %	38 %		
Ispit: Pismeni			40 %	68 %

Tjedni plan nastave

1. Uvod, HTML

2. HTML

3. CSS

Za studente

Vaš komentar

Izvedba

ID 229838

Ljetni semestar

5 ECTS

R0 Engleski jezik

R2 E-učenje

45 Predavanja

0 Seminar

0 Auditorne vježbe

15 Laboratorijske vježbe

0 Konstruktivske vježbe

Ocjenjivanje

87,5 izvrstan

75 vrlo dobar

62,5 dobar

50 dovoljan

Sličan predmet

Web Lab: A Web Programming Class and Competition, MIT

Web Engineering, ETH Zurich

64 requests

954 kB transferred

2,7 MB resources

Finish: 1,8 s

DOMContentLoaded: 1,72 s

Load: 1,8 s

Metoda HEAD

- **HEAD** - metoda za dohvaćanje podataka o resursu
 - razlika u odnosu na GET: poslužitelj **ne** vraća sadržaj resursa u tijelu odgovora
 - najčešće se koristi kako bi se provjerilo postoji li entitet na poslužitelju
- druge uporabe:
 - provjera veličine datoteke prije dohvaćanja
 - pribavljanje metapodataka o entitetu
- na poslužitelju se zahtjev HEAD obrađuje jednako kao i GET

Metoda POST

- **POST** - metoda za “aktiviranje” resursa
 - znači “pomoću adresiranog resursa obradi podatke koje šaljem”
 - obično se aktivira pritiskom gumba u obrascu
- može se koristiti za ispunjavanje Web obrasca
 - podaci koje je upisao korisnik prenose se metodom POST na poslužitelj i tamo se obrađuju
 - npr:
 - registracija i otvaranje korisničkog računa
 - zahtjev za provedbom narudžbe
 - dodavanje komentara tekstu
 - prijavljivanje u termin laboratorijskih vježbi

Prisjetimo se obrazaca HTML-a

- Podaci se iz obrasca šalju poslužitelju weba
 - šalju se odabranom resursu na strani poslužitelja (definirani URI)
 - obrađuju se od strane tog resursa (poslužitelja ili nekog vanjskog programa, skripte itd.)
 - rezultat obrade se vraća u odgovoru poslužitelja (npr. kao nova HTML stranica ili neki drugi tip resursa, binarna datoteka)
- Element **<form>**: definira kako se podaci šalju poslužitelju
- Kod slanja podataka iz obrasca na poslužitelj, preglednik formira niz parova *name-value*

name1=value1&name2=value2&name3=value3 ...

Element **<form>**

URI resursa kojem se
prosljeđuju podaci
(relativni ili apsolutni)

cilj rezultata obrade
(isti prozor, novi prozor,
imenovani prozor ...)

metoda za
prosljeđivanja sadržaja
iz obrasca (GET ili
POST)

```
<form action="post-example.php" target="_self" method="POST">
  <label for="first">First name:</label><br>
  <input type="text" id="first" name="first" required><br>
  <label for="last">Last name:</label><br>
  <input type="text" id="last" name="last" required><br>
  <label for="username">Username:</label><br>
  <input type="text" id="username" name="username" value=""
placeholder="enter your username" required><br><br>
  <input type="submit" value="Submit">
</form>
```

Obrazac i metoda POST (1/2)

Obrazac

localhost:3000/pr01.s54.formPOST.html

Učimo obrasce!

GET ili POST?

First name:

Last name:

Username:
enter your username

Submit

Obrazac

localhost:3000/pr01.s54.formPOST.html

Učimo obrasce!

GET ili POST?

First name:
John

Last name:
Doe

Username:
jdoe

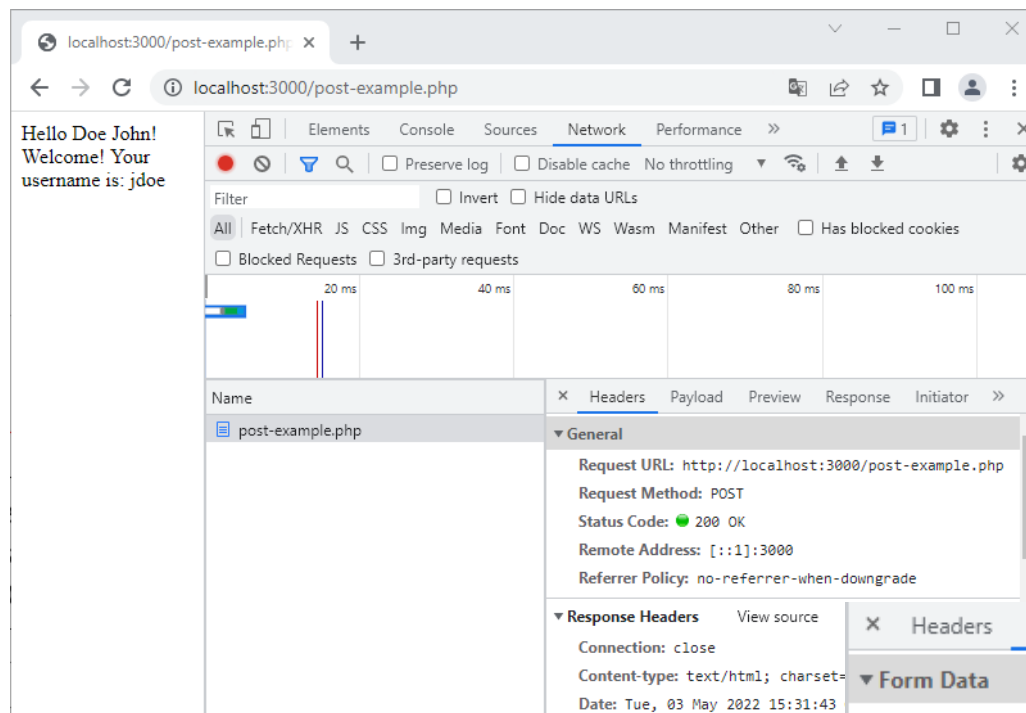
Submit

Nisu definirani tipova podataka, sve su stringovi!

`first=John&second=Doe&username=jdoe`

Kako se prenose podaci do poslužitelja kada se koristi metoda POST?

Obrazac i metoda POST (2/2)



Podaci se prenose
u tijelu zahtjeva
POST

```
<?php
// The global $_POST variable allows you to access
the data sent with the POST method by name
$first = htmlspecialchars($_POST['first']);
$last = htmlspecialchars($_POST['last']);
$un = htmlspecialchars($_POST['username']);
echo 'Hello ', $last, ' ', $first, '! Welcome!';
echo 'Your username is: ', $un;
?>
```

Prisjetimo se kodiranja podataka iz obrasca

- Problem prenošenja podataka tekstnim protokolima
 - US-ASCII, ...
 - Binarne datoteke ?!?

`<form enctype="tip kodiranja" action="/processForm.php" method="GET">`

- Tri metode kodiranja u upotrebi (norma HTML 5):
 - *application/x-www-form-urlencoded* (podrazumijevana metoda)
 - *multipart/form-data* (ako se prenose binarne datoteke) -> primjer na sljedećem slajdu
 - *text/plain* (uveden u HTML5)

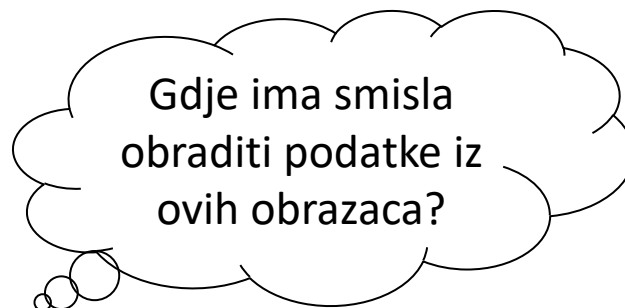
Jednostavni primjer prijena datoteke

```
<form method="post" enctype="multipart/form-data">
  <div>
    <label for="file">Choose file to upload</label>
    <input type="file" id="file" name="file" multiple accept=
".jpg, .jpeg, .png">
    <div>
      <div>
        <button>Submit</button>
      </div>
    </div>
  </form>
```

- Za prijenos datoteka se koristi obavezno POST
- Moguće je koristiti i Base64 kodiranje (*binary-to-text encoding scheme*), kodira binarne podatke u ASCII znakove
- Za više informacija: [https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics of HTTP/Data URLs#encoding data into base64 format](https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Data_URLs#encoding_data_into_base64_format), <https://base64.guru/learn/what-is-base64>
- Link na koder: <https://base64.guru/converter/encode>

Obrada obrasca na strani klijenta ili poslužitelja

Temperature =
Celsius Kelvin
Formula $15^{\circ}\text{C} + 273.15 = 288.15\text{K}$



FER - projekti

0. OSNOVNO O PROJEKTU

Kratika projekta ☐

IoT-polje

Naziv projekta (na hrvatskom jeziku) * ☐

Ekosustav umreženih uređaja i usluga za Internet stvari s primjenom u poljoprivredi

Naziv projekta (na engleskom jeziku) ☐

An Ecosystem of Networked Devices and Services for IoT Solutions Applied in Agriculture

Sažetak projekta (na hrvatskom jeziku) ☐

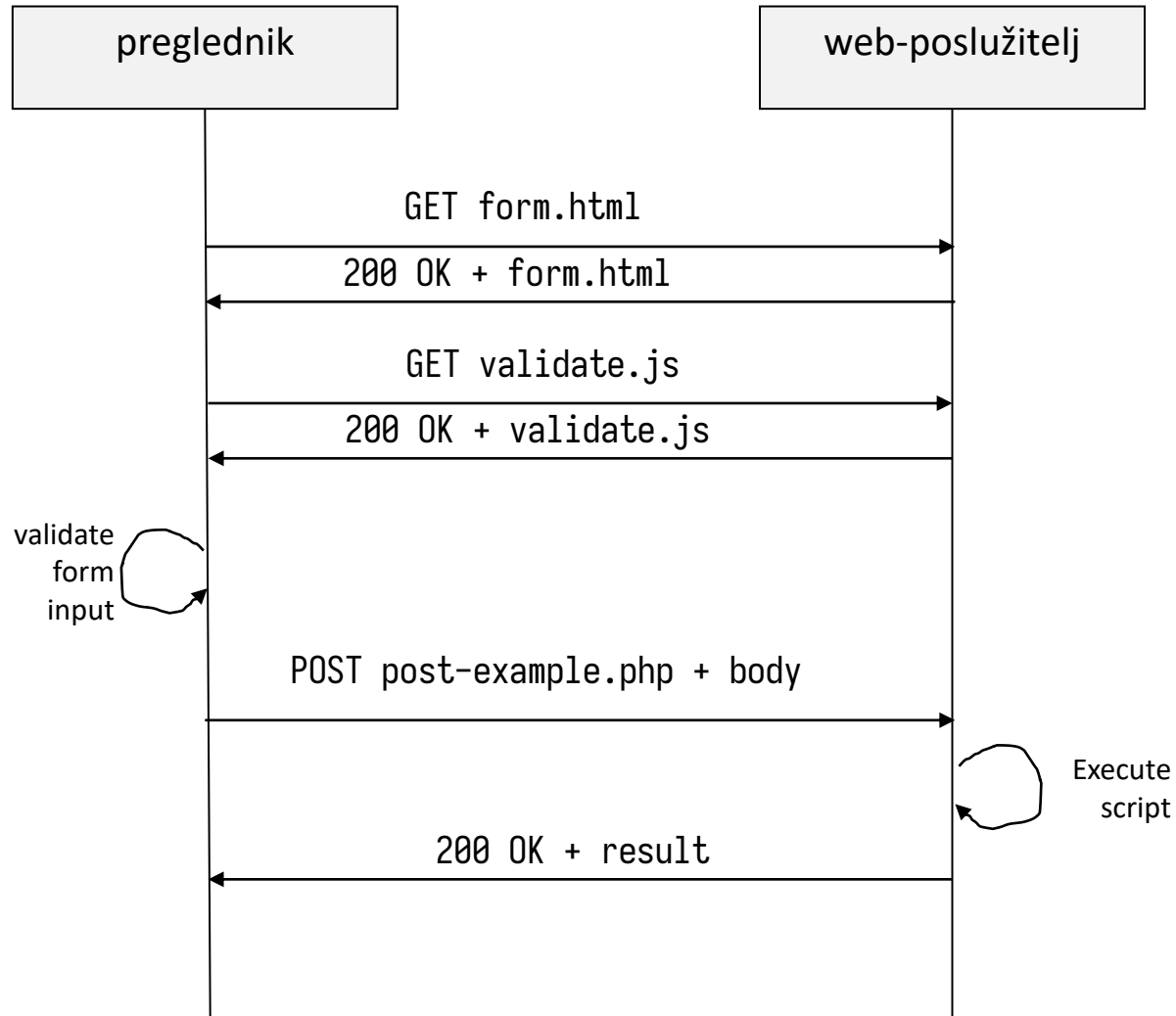
Cilj projekta je povećati tržišno orijentirane IRI aktivnosti u područjima Interneta stvari i biljnih znanosti za uspostavu ekosustava umreženih uređaja i inovativnih usluga s primjenom u poljoprivredi. Projektne aktivnosti su usmjerene na istraživanje i razvoj interoperabilnih i sigurnih tehničkih rješenja ekosustava za prikupljanje i naprednu obradu stvarnovremenskih mikroklimatskih i agronomskih podataka, radi unaprjeđenja biljne proizvodnje u RH. Projekt će omogućiti prijenos znanja i tehnologije u hrvatska ICT-poduzeća i povećanje konkurentnosti obiteljskih poljoprivrednih gospodarstava.

Sažetak projekta (na engleskom jeziku) ☐

The project goal is to increase market-oriented R&D activities in the areas of Internet of Things and Plant Sciences to establish an ecosystem of networked devices and innovative services with application in agriculture. Project activities are focused on the research and development of interoperable and secure technical solutions for an ecosystem enabling the collection and advanced processing of real-time microclimate and agronomic data to improve plant production in the Republic of Croatia. The project will enable transfer of knowledge and technology to Croatian ICT companies and increase the competitiveness of family farms.

https://www.w3schools.com/js/js_validation.asp

Validacija podataka (dijagram)



Ostale metode HTTP-a

- **OPTIONS**

- informiranje o mogućnostima resursa i poslužitelja (URI = *)

- **PUT**

- postavljanje entiteta uključenog u tijelu zahtjeva na zadani URI
 - POST navodi resurs koji mora obraditi podatke koji se šalju u tijelu zahtjeva
 - PUT navodi naziv resursa u kojeg treba pohraniti podatke koji se šalju u tijelu
 - najčešće se ne koristi za Web jer predstavlja sigurnosni rizik
 - koristi se za REST: promjena podataka

- **DELETE** - brisanje odabranog resursa (ne koristi se)

- **TRACE** - klijent dobiva kopiju zahtjeva kojeg je uputio poslužitelju, služi za dijagnostiku

- **CONNECT** - za buduću uporabu, ne implementira se

Zaglavlja odgovora

HTTP/1.0
HTTP/1.1

```
HTTP-Version Status-Code Reason-Phrase
parametar1: vrijednost
parametar2: vrijednost
parametar3: vrijednost
....
<prazna linija>
<html>
- tijelo dokumenta -
</html>
```

general_header

Cache-Control:
Connection:
Date:
Pragma:
Trailer:
Transfer-Encoding:
Upgrade:
Warning:

response_header

Accept-Ranges:
Age:
ETag:
Location:
Proxy-Authenticate:
Retry-After:
Server:
Vary:
WWW-Authenticate:

entity_header

Allow:
Content-Encoding:
Content-Language:
Content-Length:
Content-Location:
Content-MD5:
Content-Range:
Content-Type:
Expires:
Last-Modified:

Informacija

100 Continue

Uspjeh

200 OK

Preusmjerenje

300 Multiple Choices

301 Moved
permanently

302 Found

304 Not Modified

Pogreška na klijentu

400 Bad Request

401 Unauthorized

403 Forbidden

404 Not Found

Pogreška na poslužitelju

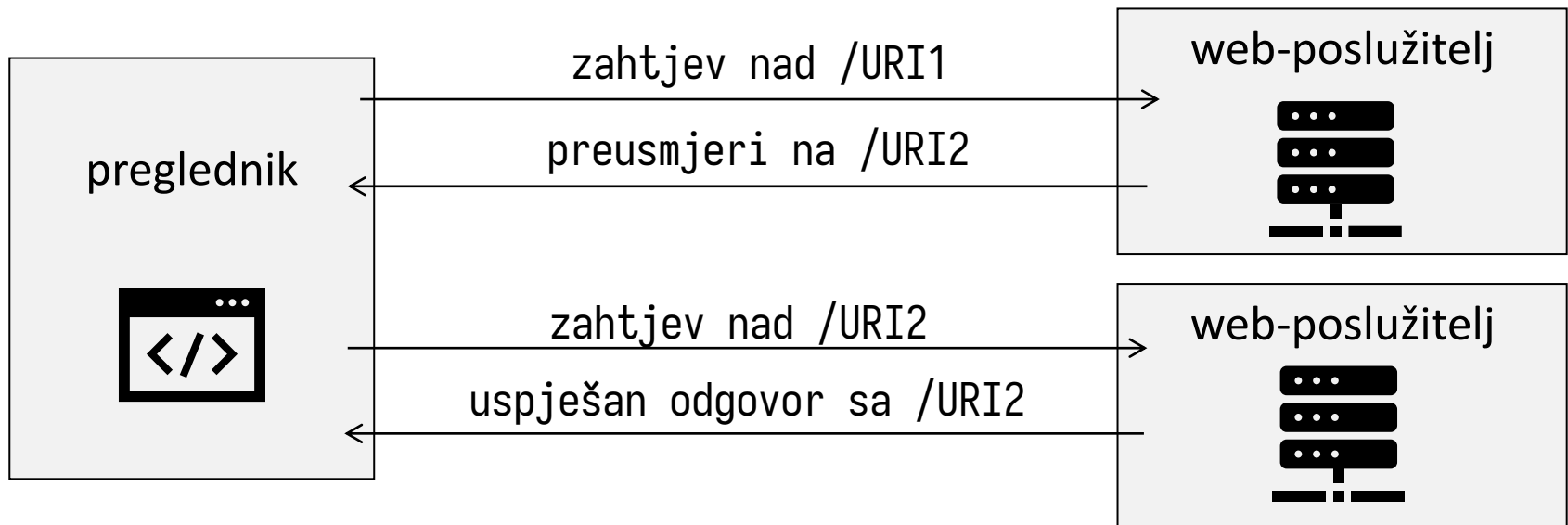
500 Internal Server
Error

Uspješan odgovor (2xx)

- poslužitelj je uspješno primio, razumio i ispunio zahtjev
- najčešći kôd **200** (*OK*)
 - za zahtjev **GET** znači da je entitet dostavljen u tijelu odgovora sadržaj traženog resursa
 - za **POST** znači da je resurs primio podatke i dostavljeni entitet opisuje ishod akcije
- najčešći odgovor na Webu

Preusmjeravanje (3xx)

- klijent treba poduzeti dodatne korake kako bi ispunio izvorni zahtjev (301 *permanent*, 302 *temporal redirect*)
- novi URI se nalazi u zaglavlju odgovora **Location:**
- sigurne metode se mogu izvršiti bez sudjelovanja korisnika
 - sigurne: GET, HEAD, OPTIONS, TRACE



Pogreške na klijentu (4xx)

- namijenjena za slučajeve kad se čini da je pogreška nastupila na strani klijenta
- odgovor treba sadržavati poruku namijenjenu korisniku koja opisuje situaciju i nudi moguće rješenje, npr.
- **400** (*Bad Request*)
 - pogreška u sintaksi zahtjeva
- **401** (*Unauthorized*)
 - zahtjevu nedostaje autorizacija korisnika
- **404** (*Not Found*)
 - resurs nije dostupan, ali se ne ulazi u detalje zašto
 - obično je pogreška prilikom zadavanja URI-ja

Pogreške na poslužitelju (5xx)

- kod ovakvih odgovora poslužitelj je “svjestan” da zahtjev nije ispunjen zbog njegove pogreške
- **500** (*Internal Server Error*)
 - obično programska pogreška u resursu
- **503** (*Service Unavailable*)
 - poslužitelj je preopterećen
 - odgovor može sadržavati polje **Retry-After:**

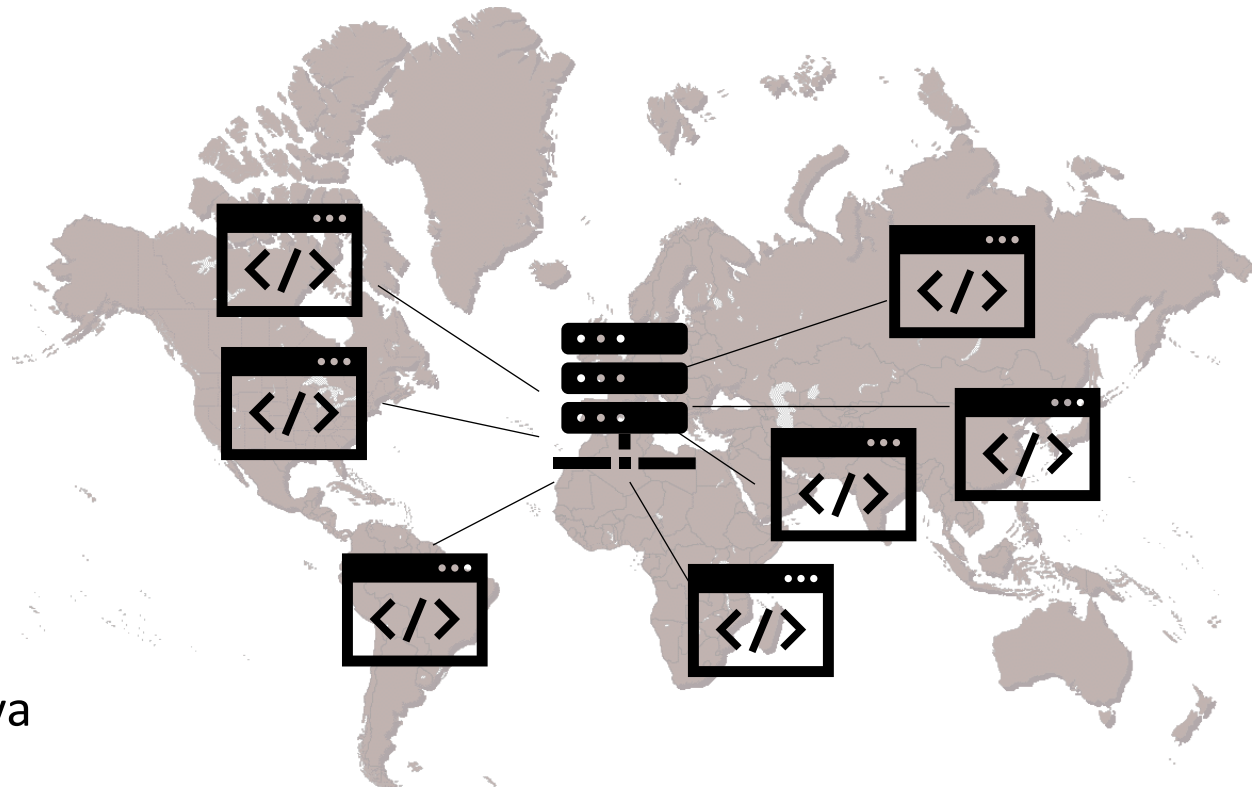
Sadržaj predavanja

- Uvod
 - Osnove protokola HTTP
 - Web-poslužitelj i web-preglednik
 - Media Type i tijek komunikacije
 - HTTPS
- Identifikacija resursa
 - URI, URL, URN
 - sintaksa URI-ja
 - relativni i apsolutni URI
- Poruke protokola HTTP
 - zahtjev i odgovor
 - metode zahtjeva i kôdovi odgovora
- Priručna spremišta

Web je ipak centraliziran!

“The traditional Web model comes of age”

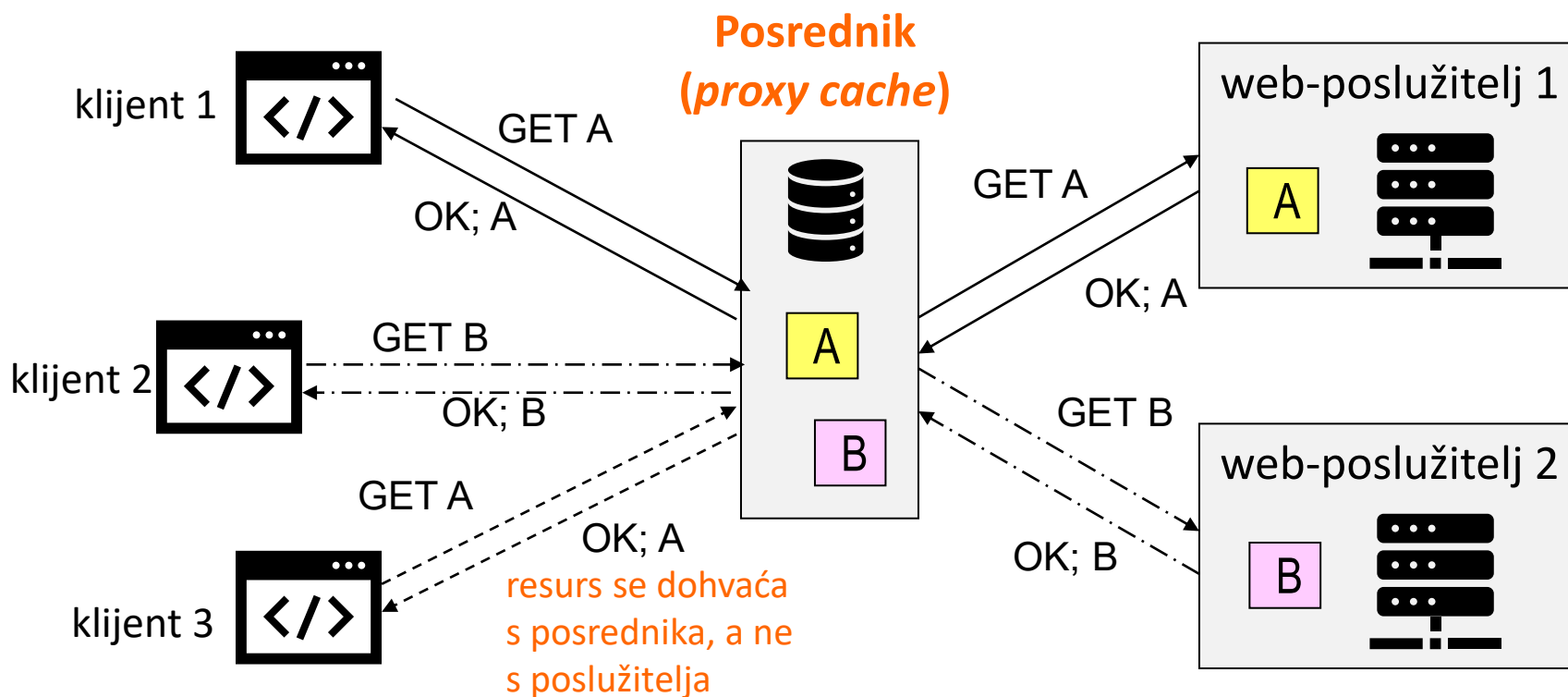
- Tradicionalni model klijent-poslužitelj pati od problema skalabilnosti
 - preopterećenost poslužitelja
 - povećan i nejednolik intenzitet klijentskih zahtjeva (broj zahtjeva u jedinici vremena) stiže poslužitelju



Web i priručna spremišta

- Cilj: „preseliti” resurse u blizinu korisnika, tj. pohraniti kopiju resursa u priručno spremište
- Priručno spremište
 - pohranjuje podatke na lokaciji bliskoj onoj na kojoj se ti podaci koriste
 - neki sadržaji su “popularniji”, tj. koriste se češće od drugih (poznato iz statistike pristupa), pa ih ima smisla pohraniti u priručno spremište
 - popularne sadržaje je moguće replicirati na lokacijama s povoljnim pristupom (na klijentskom računalu, u lokanoj mreži u mreži ISP-a, na strani poslužitelja, u mreži priručnih spremišta tzv. *Content Delivery Network*)

Ideja posredničkog poslužitelja s priručnim spremištem



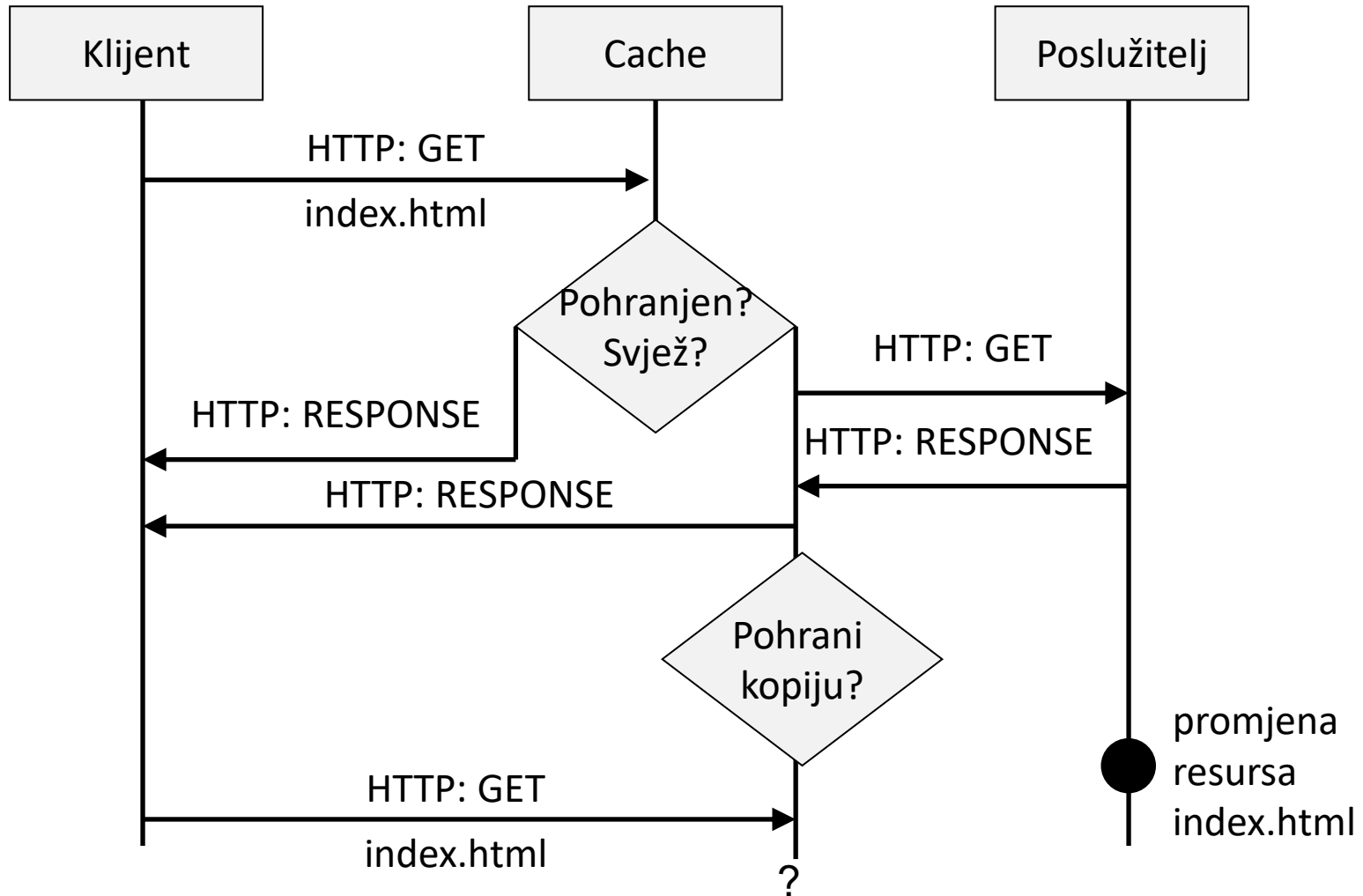
Osnovni ciljevi primjene priručnog spremišta

- **smanjiti vrijeme odziva** web-poslužitelja
 - ako se resurs dohvaća iz priručnog spremišta koje je mrežno bliže klijentu, onda se vrijeme odziva smanjuje
 - povoljno za korisnika jer je ovo osnovna mjera performanci web-poslužitelja
 - vrijeme se smanjuje samo ako priručno spremište pohranjuje traženi resurs
- **smanjiti internetski promet**
 - priručno spremište je mrežno bliže klijentu i zahtjevi/odgovori se ne prenose Internetom do web-poslužitelja
- **smanjiti opterećenje poslužitelja**
 - priručna spremišta preuzimaju dio klijentskih zahtjeva umjesto web-poslužitelja

Koji resurs se smije pohraniti u priručno spremište?

- Statični sadržaj koji se često ne mijenja, npr. slike
- Pravilo definira poslužitelj (tj. admin)
- Osobni i dinamički generirani sadržaji se ne pohranjuju (npr. broj kreditne kartice, personalizirani odgovor na upit)
- Klijent također može definirati prihvaća li ili ne sadržaj iz priručnog spremišta

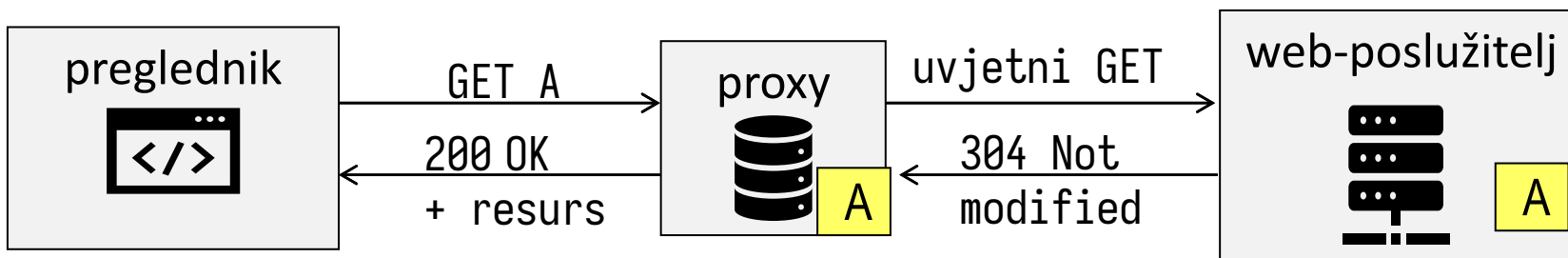
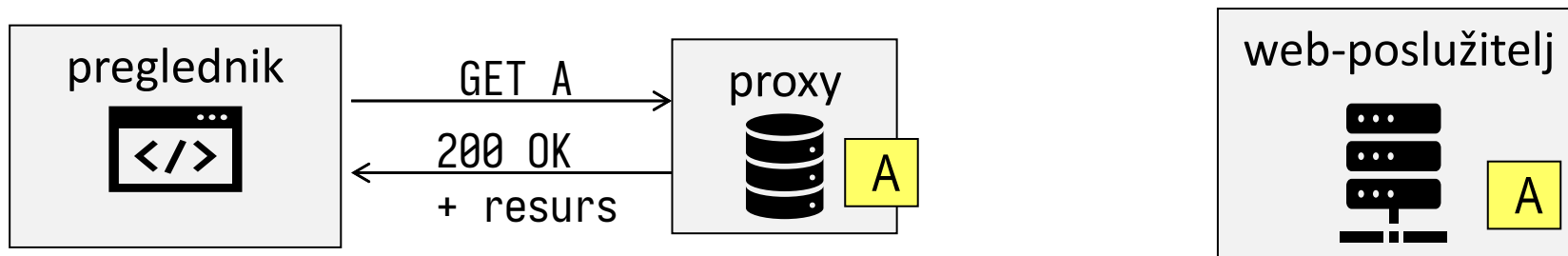
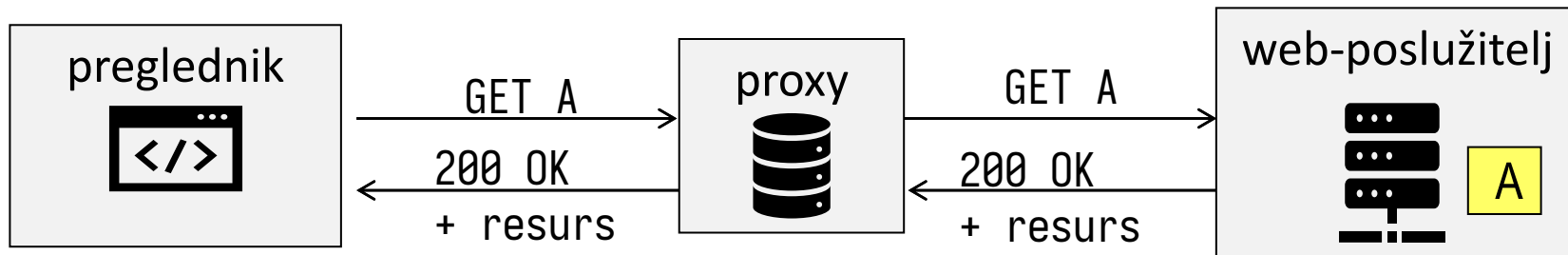
Osnovni tijek komunikacije



Uvjetni (*conditional*) GET

- ponašanje metode GET se mijenja ako se koristi uvjetni GET
- Koristi sljedeća polja u zaglavlju GET zahtjeva
 - If-Modified-Since: / If-Unmodified-Since:
 - If-Match: / If-None-Match: / If-Range:
- Odgovor na uvjetni GET može biti: 304 (Not Modified) ili 200 OK
 - Ako je 304, traženi resurs nije se mijenjao na poslužitelju od zadnjeg zahtjeva i kopija iz priručnog spremišta nije zastarjela
- Za upravljanje ponašanjem priručnog spremišta koristi se polje Cache-Control (pojavljuje se u zahtjevu i odgovoru)

Moguća interakcija klijenta, priručnog spremišta i poslužitelja



Provjera je li došlo do promjene resursa!

Primjer za 304 Not Modified

The screenshot shows a web browser window with the address bar displaying `https://tools.ietf.org/rfc/rfc2616`. The page content includes the text "Network Working Group", "Request for Comments: 2616", and "Obsoletes: 2068". The browser's developer tools are open, showing the Network tab. A table of network requests is visible, with the first request (304 GET) selected. The Headers panel for this request shows the status code 304 Not Modified and the response headers.

Sta...	Me...	Domain	File	Cause	Type	Transferred	Siz
304	GET	tools.ietf...	rfc2616	document	html	cached	501
200	GET	tools.ietf...	favicon.ico	img	vn...	cached	822

Request Headers (485 B)

```
Host: tools.ietf.org
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:75.0) Gecko/20100101 Firefox/75.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Connection: keep-alive
Upgrade-Insecure-Requests: 1
If-Modified-Since: Thu, 02 May 2019 21:13:44 GMT
If-None-Match: "c5e5fc-7d5a8-587ee1bec7200;5a2ec60a59840"
Cache-Control: max-age=0
```

Response Headers (276 B)

```
HTTP/1.1 304 Not Modified
Date: Fri, 10 Apr 2020 22:53:29 GMT
Server: Apache/2.2.22 (Debian)
Connection: Keep-Alive
Keep-Alive: timeout=5, max=100
ETag: "c5e5fc-7d5a8-587ee1bec7200;5a2ec60a59840"
Content-Location: rfc2616.html
Vary: negotiate,accept,Accept-Encoding
```

Preporučena literatura

1. <https://developer.mozilla.org/en-US/docs/Web/HTTP>
 - Uvodno o HTTP-u, <https://developer.mozilla.org/en-US/docs/Web/HTTP/Overview>
 - Poruke HTTP-a, <https://developer.mozilla.org/en-US/docs/Web/HTTP/Messages>
 - Za više informacija o performancama (TCP u kontekstu HTTP-a, slajd 46), Connection Management in HTTP 1.1
[https://developer.mozilla.org/en-US/docs/Web/HTTP/Connection management in HTTP 1.x](https://developer.mozilla.org/en-US/docs/Web/HTTP/Connection_management_in_HTTP_1.x)
2. Validacija podataka u obrascima,
https://www.w3schools.com/js/js_validation.asp

Slike korištene u prezentaciji

Icons made by [Freepik](https://www.flaticon.com) from www.flaticon.com (slajdovi 5 i 18)
REST icon <https://icon-library.net/icon/rest-api-icon-23.html> (slajd 22)