Unità di apprendimento 1

Architettura di rete e formati per lo scambio dei dati

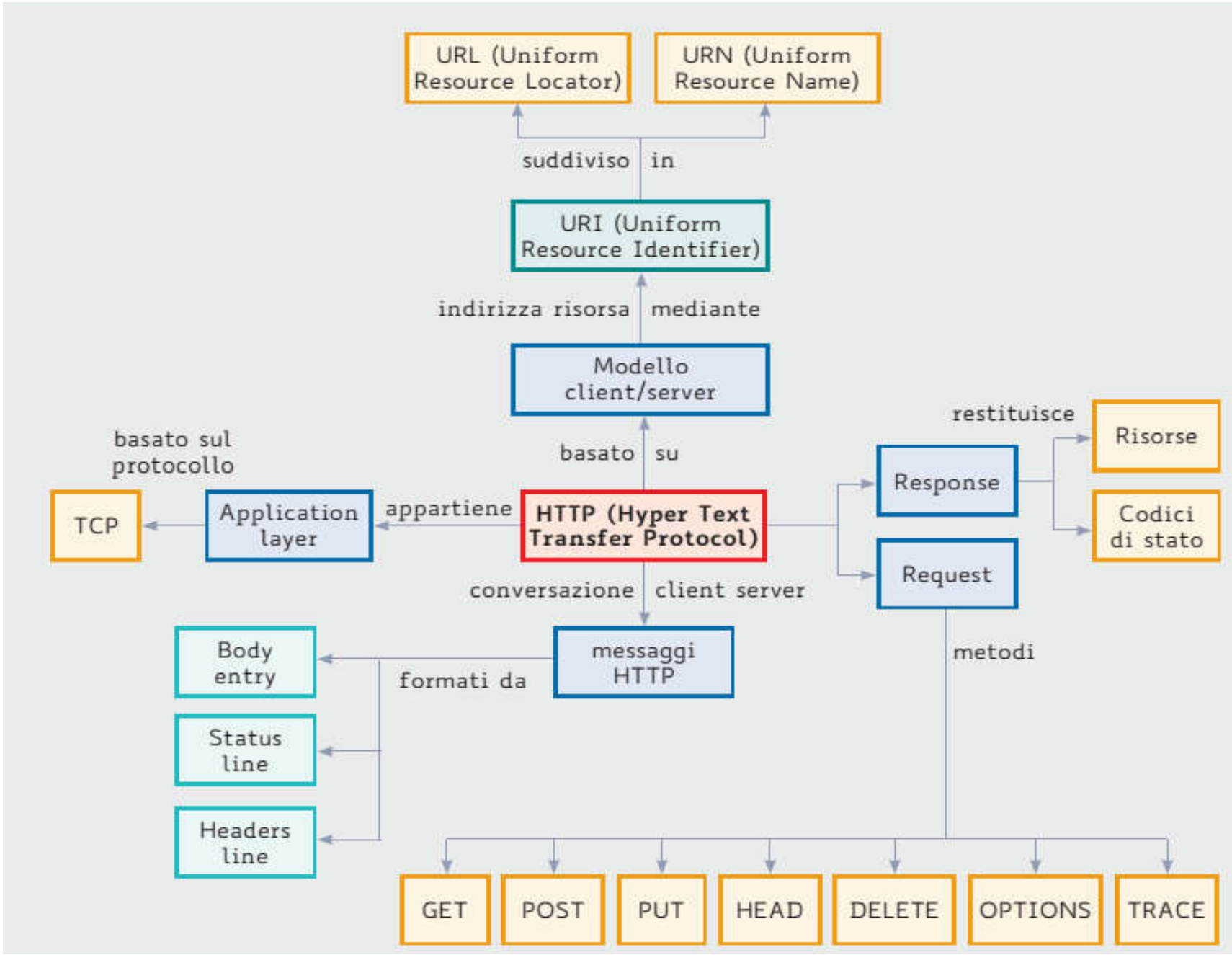
HOEPLI Scuola.it

Unità di apprendimento 1
Lezione 3

La comunicazione nel Web
con
protocollo http

In questa lezione impareremo:

- le caratteristiche del protocollo HTTP
- la struttura dei messaggi HTTP
- I codici di stato


URL (Uniform Resource Locator)
URN (Uniform Resource Name)
suddiviso in
URI (Uniform Resource Identifier)
indirizza risorsa mediante
Modello client/server
basato su
basato sul protocollo
TCP
Application layer
appartiene
HTTP (Hyper Text Transfer Protocol)
restituisce
Response
Risorse
Codici di stato
Request
conversazione client server
Body entry
messaggi HTTP
formati da
metodi
Status line
Headers line
GET
POST
PUT
HEAD
DELETE
OPTIONS
TRACE

HTTP e il modello client/server

- HTTP è un protocollo usato per trasmettere risorse, non solo file.
- Una risorsa è identificata da un URI o URL.
- Il tipo più comune di risorsa è un file, ma può anche essere il risultato di una query generato dinamicamente, l'uscita di uno script CGI, un documento disponibile in diversi linguaggi, o altro ancora.

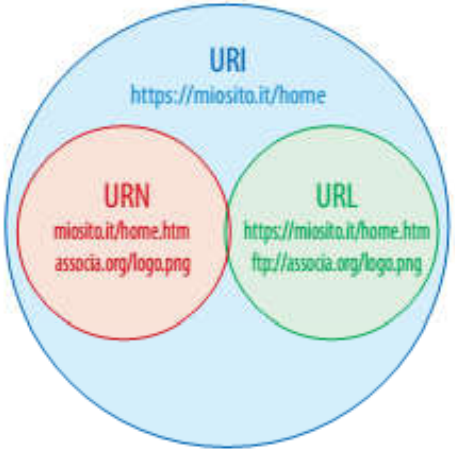
HTTP e il modello client/server

- Vediamo le definizioni del W3C (World Wide Web Consortium):
  - URI (Uniform Resource Identifier) : *“set generico di nomi o indirizzi che rappresentano stringhe assegnate alle risorse”;*
  - URL(Uniform Resource Locator) : *“termine informale, utilizzato solo nelle specifiche tecniche, associato a popolari protocolli quali HTTP,FTP, mailto, ... ecc.”.*

HTTP e il modello client/server

- Sono state definite due versioni del protocollo HTTP
  - HTTP/1.0 (anno 1996)
  - HTTP/1.1(anno 1999)
- La versione 1.1 permette di specificare una connessione permanente, oltre a consentire il criptaggio di alcuni dati.
- Per connessione permanente si intende una richiesta e la risposta all'interno della stessa connessione, grazie al server che lascia aperta la connessione TCP dopo aver spedito la risposta

HTTP e il modello client/server



- Vediamo un esempio di indirizzo URI:

Il protocollo HTTP

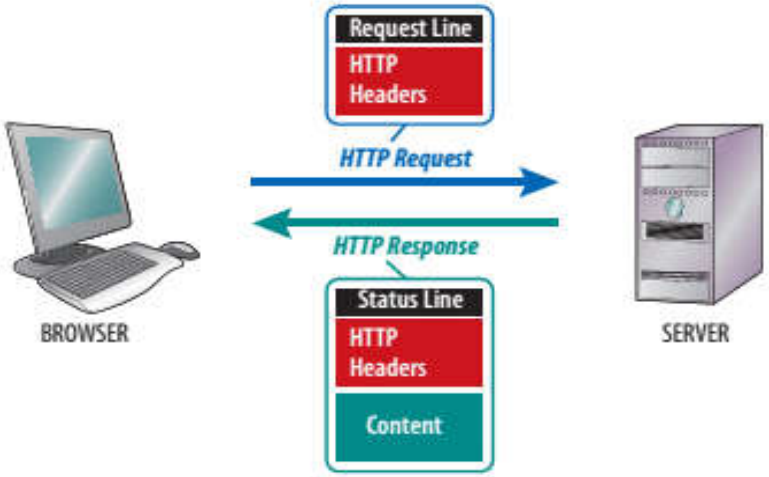
- HTTP è un protocollo di testo che fornisce il livello di trasporto a tutti i protocolli applicativi basati su di esso
- Possiamo dire che il Web (WWW) è nato dall'insieme di diverse tecnologie, le principali sono HTML, URL e HTTP.

Il protocollo HTTP

- Il protocollo HTTP per comunicare utilizza sessioni
  - ciascuna sessione inizia stabilendo per prima cosa una connessione TCP al server
  - utilizza di default la porta TCP 80,
  - effettua poi una richiesta(request) contenente un URL
- In risposta, il server
  - produce e restituisce il file richiesto
  - chiude la connessione TCP immediatamente dopo.

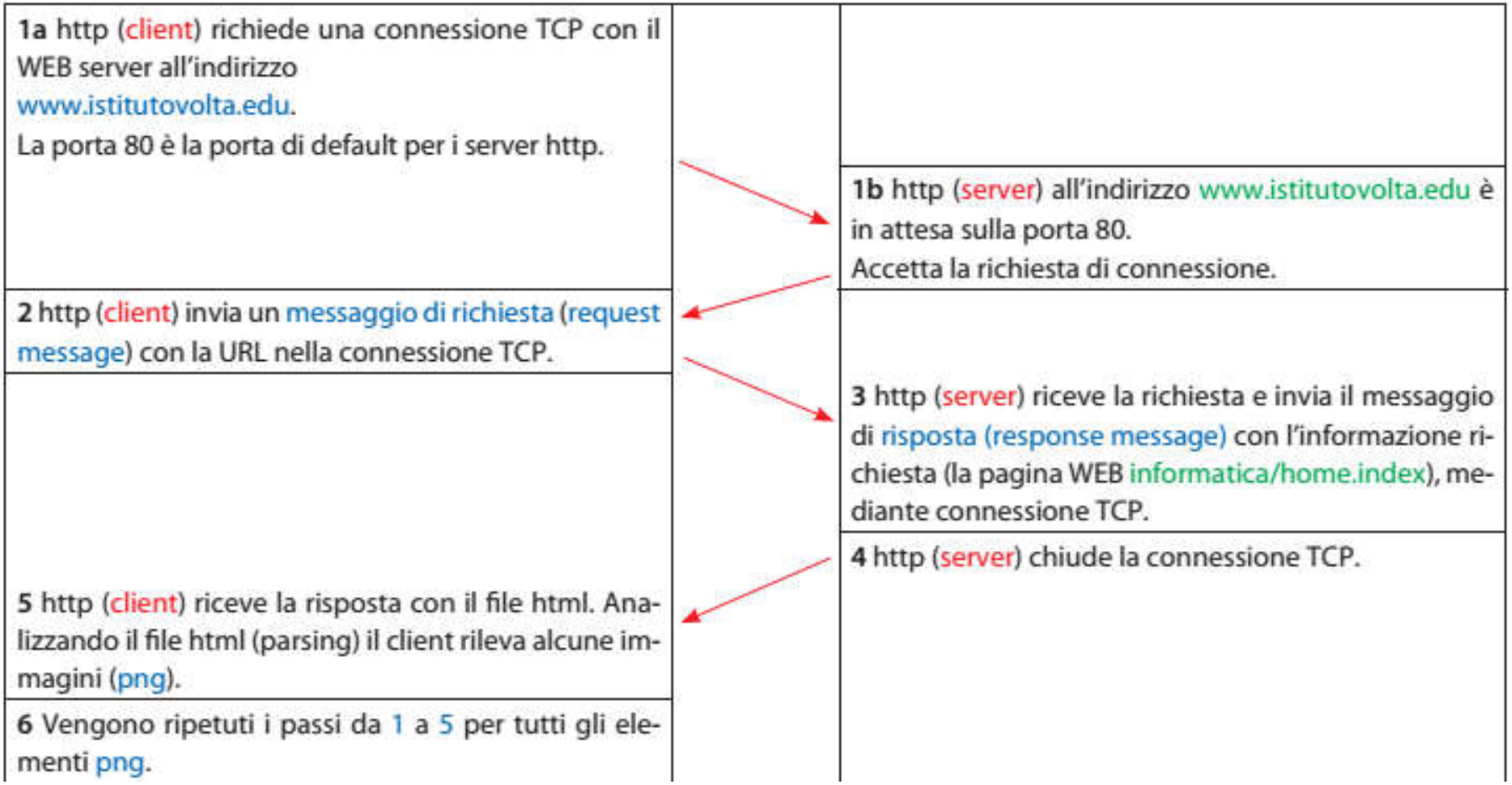
Il protocollo HTTP

- **Conversazione client-server**
  - Ogni conversazione tra cliente server sul Web inizia con una richiesta (request) rappresentata da un messaggio di testo creato dal client in formato HTTP.
  - Il client in via la richiesta al server, quindi attende la risposta(res\ponse).



ESEMPIO

Ipotizziamo di volere richiedere una pagina composta da un file HTML e da alcune immagini in formato png all'indirizzo www.istitutovolta.edu/informatica/home.index.



Tipi di connessioni

- Esistono due tipi di connessione permanente:
  - connessione permanente incanalata
  - connessione permanente non incanalata
- Nella versione non incanalata il client passa una nuova richiesta solo quando la risposta alla precedente è stata ricevuta.
- Nella versione incanalata le richieste vengono via via aggiunte a una coda chiamata pipeline, mentre le risposte vengono processate e inviate nello stesso ordine delle richieste.

I messaggi HTTP

- Durante una connessione e una comunicazione HTTP cliente server si scambiano messaggi di richiesta e di risposta, entrambi formati da:
  - una riga iniziale;
  - un'intestazione (header), anche assente;
  - una riga vuota;
  - un corpo del messaggio (body), anche assente;

Messaggio di richiesta: Request HTTP

- La prima riga della request contiene la versione del protocollo HTTP utilizzato
- nelle righe successive vengono indicate gli header (intestazioni), rappresentate da diversi elementi, ciascuno dei quali composto da un nome, seguito dai due punti (:) e da un valore
- ogni riga rappresenta un distinto header

Messaggio di richiesta: Request HTTP

- Gli header più comuni sono:
  - la versione del browser che prende il nome di User-Agent;
  - l'host presente nell'URL;
  - il parametro l (per i tipi di documento che supportano il tipo MIME

Messaggio di richiesta: Request HTTP

- Una HTTP request è un messaggio testuale inviato dal client al server HTTP ed è formato da tre elementi:
  - riga di richiesta
  - Intestazione HTTP (header)
  - corpo del messaggio(message body)

```
<Method> <URI ><Version>          //riga di richiesta
[Header]                          //intestazione
CRLF                              //riga vuota
[Body]                            //corpo del messaggio
```

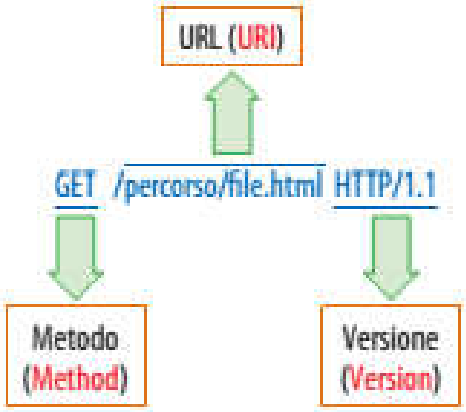
Messaggio di richiesta: Request HTTP

- **Riga di richiesta**
- La riga di richiesta contiene:
  - il metodo (Method) che può essere:
    - GET (versione HTTP 0.9),
    - DELETE, GET, HEAD, LINK, POST, PUT, UNLINK(versione HTTP 1.0),
    - CONNECT, DELETE, GET, HEAD, POST, PUT, OPTIONS, TRACE (versione HTTP 1.1);
  - l'URI (o URL) è l'identificativo di risorsa richiesta al server;
  - la versione (Version) può assumere i valori HTTP/1.0 o HTTP/1.1.

Messaggio di richiesta: Request HTTP

- Un esempio di riga di richiesta HTTP è:

```
GET  /percorso/file.html HTTP/1.1
```



- In questo caso possiamo notare
  - che il metodo richiesto è GET l'URI è rappresentato
- dal path name nel formato /percorso/file.html
  - si tratta di un file in formato html presente nella sottodirectory della radice del server chiamata /percorso

Intestazione HTTP (header)

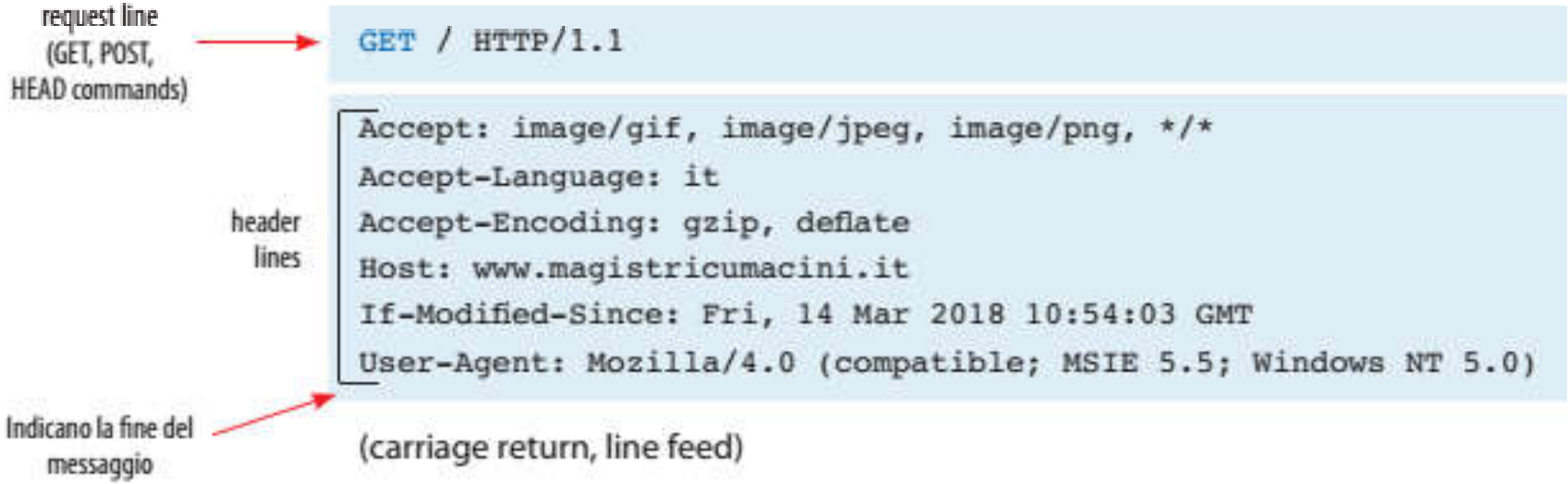
- L'intestazione HTTP (header) contiene tutte le informazioni necessarie per l'identificazione del messaggio.
- È formata da diverse righe, ciascuna delle quali rispetta il formato:

```
nome_header : valoreCRLF
```

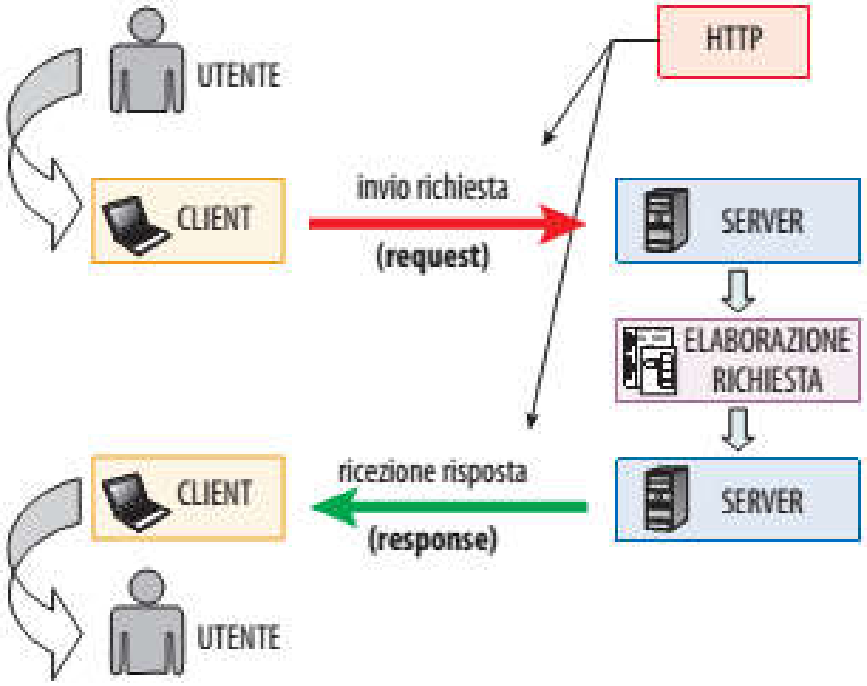
- e sono classificate in:
  - intestazioni generali
  - intestazioni della richiesta
  - intestazioni del corpo dell'entità

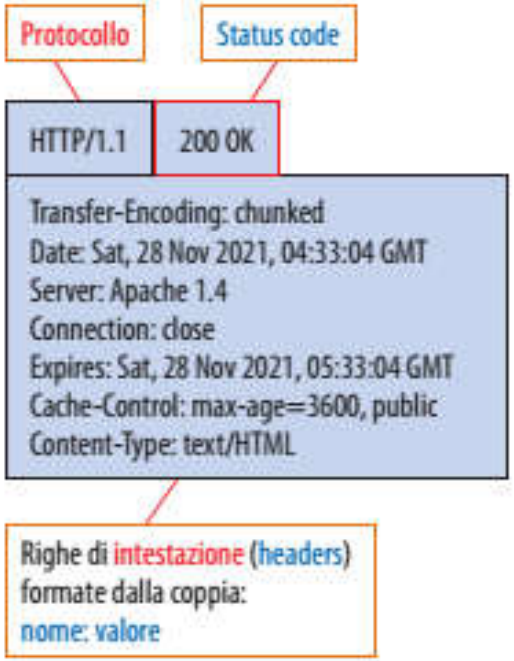
Corpo del messaggio (Message body)

- Il corpo del messaggio (message body) di richiesta contiene i dati trasportati



Messaggio di risposta: Response HTTP





Messaggio di richiesta: Request HTTP

- La response (risposta) HTTP è organizzata in maniera analoga rispetto a una richiesta.
- L'unica differenza è che le risposte iniziano con una riga di stato al posto della riga di richiesta.
- La response è formata da:
  - una riga iniziale conversione protocollo HTTP e stato;
  - un'intestazione(header) facoltativa;
  - un corpo(body) facoltativo.

Header HTTP

- Gli header della response sono formati, come nel caso della richiesta, da insiemi di coppie, formate da un nome separato dal valore dai due punti (:), che specificano le caratteristiche del messaggio.
- Header generali della trasmissione, ad esempio:
  - MIME-Version: la versione MIME usata per la trasmissione, normalmente 1.0;
  - Transfer-Encoding: indica la compressione dei dati trasmessi e si riferisce a tutto il messaggio

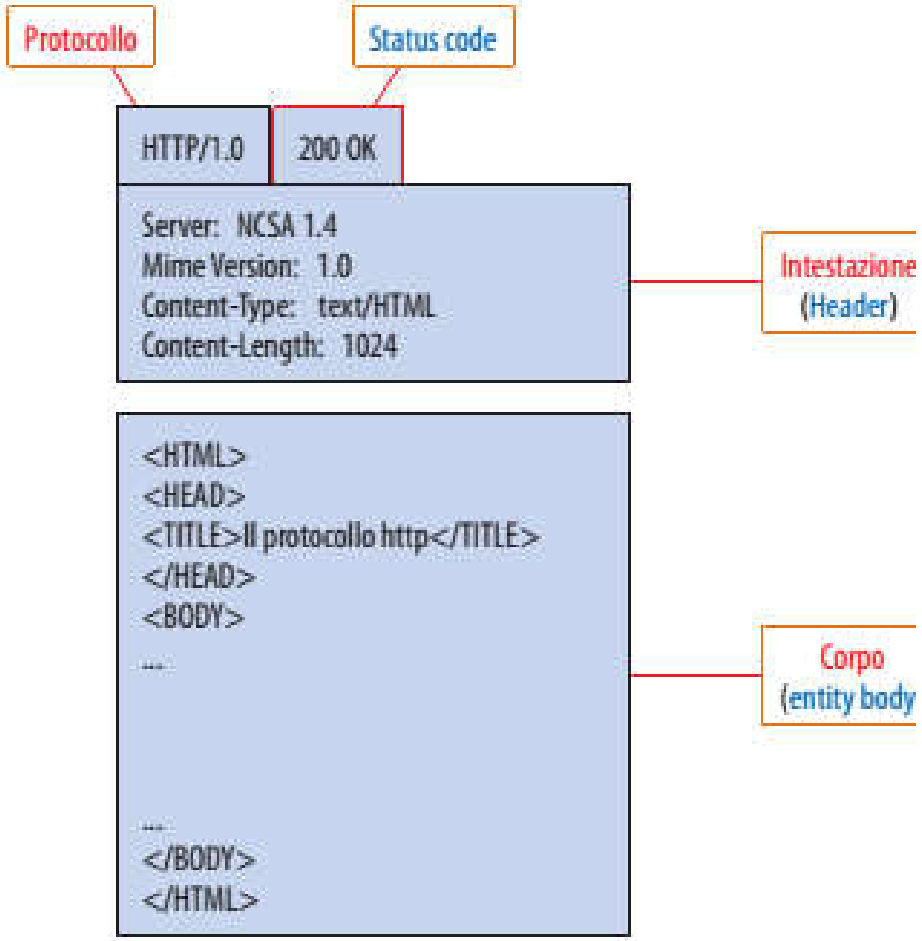
Header HTTP

- Header relativi all'entità trasmessa:
  - Content-Type: indica il MIME e specifica se si tratta di un testo, un'immagine PNG, un suono WAV, un MPG ecc.;
  - Expires: indica la data di validità della risorsa.
- Header riguardo la richiesta effettuata, ad esempio:
  - User-Agent: indica la versione e il tipo di browser oltre alla versione e al sistema operativo del client;
  - Host: indica il nome di dominio del server e la porta TCP sulla quale il server è in ascolto.

Header HTTP

- Header della risposta generata, come per esempio:
  - Server: una stringa che descrive il server: tipo, sistema operativo e versione;
  - WWW-Authenticate: indica il Tipo di autenticazione e le credenziali per autenticare un utente con il server.
- Il protocollo HTTP utilizza messaggi in formato ASCII.

Body della risposta



Metodi (verbi) HTTP

- Nella conversazione HTTP, la prima riga di intestazione di una request contiene un elemento chiamato metodo o verbo HTTP

```
GET / HTTP/1.1
```

- In questo secondo caso viene invece richiesto il metodo DELETE:

```
DELETE /prove/risorsa HTTP/1.1
```

METODO	VERSIONE HTTP	MODIFICHE IN HTTP/1.1	DESCRIZIONE
GET	1.0 e 1.1	Richiesta di parti di entità	Richiede un file al server. La risposta è composta da vari header separati con due CR (Carriage Return) dal path effettivo della risorsa richiesta, in genere rappresentato dal file HTML che contiene la pagina richiesta. Tra i dati della risposta si troverà il protocollo usato (HTTP/1.1) e lo status code che indica l'esito della richiesta. Possiamo anche specificare alcuni parametri, rappresentati da una coppia: `attributo=valore&attributo=valore ... ecc.`
HEAD	1.0 e 1.1	Uso di header di richiesta condizionali	Richiede solo l'header, senza alcun percorso di file. Utilizzato solo in ambito di fase di testing per pagine Web dinamiche.
POST	1.0 e 1.1	Gestione di connessione, trasmissione di messaggio	Invia informazioni all'URL specificato. Le informazioni sono presenti in coppie: `attributo=valore&attributo=valore ... ecc.` e vengono elaborate dal server.
PUT	1.1 (1.0 non standard)		Esegue l'upload di un file sul server, creandolo o riscrivendolo, con il nome indicato e i contenuti specificati nella parte che segue gli header.
DELETE	1.1 (1.0 non standard)		Cancella una risorsa, generalmente rappresentata da un file, sul server. L'utente con cui è in esecuzione il Web server deve poter avere permessi in scrittura sul file indicato e il server deve essere configurato opportunamente (permesso di cancellazione).
OPTIONS	1.1	Estensibilità	Richiede l'elenco dei metodi concessi dal server, per esempio: `OPTIONS * HTTP/1.0`
TRACE	1.1	Estensibilità	Traccia una richiesta, visualizzando come viene trattata dal server, per esempio `TRACE * HTTP/1.0`

Metodi (verbi) HTTP

- I più importanti metodi HTTP necessari per la conversazione attraverso applicazioni Web server side (API) di tipo RESTful sono:
  - GET, POST, PUT e DELETE.
- Le API per HTTP consentono di eseguire operazioni sui dati presenti sui server.

Metodi (verbi) HTTP

- Le applicazioni RESTful (REpresentational State Transfer) consentono di effettuare operazioni da remoto sui dati presenti nei server, che lo permettono, sfruttando i metodi del protocollo HTTP.
- Le operazioni del tipo CRUD(Create, Retrieve, Update, Delete), cioè crea, recupera, aggiorna e cancella, possono essere così descritte:
  - chiedere dati al server (GET)
  - creare dati sul server (POST)
  - modificare o sostituire i dati sul server (PUT)
  - cancellare un oggetto sul server (DELETE)

Header HTTP

- La codifica URL viene applicata anche nel caso in cui la richiesta venga generata, per esempio, da un link HTML (tag <a href>) e non solo da campi di un modulo (tag <form>).
- Vediamo i passaggi che devono essere effettuati per codificare la stringa da inviare.
- La stringa da inviare è sempre formata da una serie di coppie nome campo e valore
- Possiamo paragonare il nome campo a una specie di variabile e il valore al dato in essa contenuto

Header HTTP

- le operazioni da fare sono le seguenti:
  - 1 convertire tutti i caratteri non sicuri presenti nella stringa in simboli formati dalla percentuale e dal codice ASCII relativo: %xx,dove xxè il valore ASCII del carattere, in esadecimale. I caratteri non sicuri includono =, &, %, @, spazio, +, e altri caratteri non stampabili;
  - 2 eliminare gli spazi vuoti sostituendoli con il carattere %20;
  - 3 unire i nomi e valori con uguale (=) e separare i nomi con ampersand (&), per esempio

```
nome1=valore1&nome2=valore2&nome3=valore3 ... ecc.
```

CARATTERE	CODIFICA URL ENCODE	CARATTERE	CODIFICA URL ENCODE	CARATTERE	CODIFICA URL ENCODE
	%20	#	%23	%	%25
"	%22	$	%24	&	%26
*	%2A	/	%2F		
0	%30	1	%31	2	%32
3	%33	4	%34	5	%35
6	%36	7	%37	8	%38
9	%39	<	%3C	>	%3E
@	%40	[	%5B	]	%5D

Le rappresentazioni HTTP

- Come abbiamo visto client e server HTTP si scambiano informazioni riguardanti risorse identificate da un indirizzo (URL o URI)
- sia la richiesta sia la risposta contengono una rappresentazione della risorsa
- I due elementi cardine della rappresentazione sono:
  - l'intestazione (header) , che contiene i metadati
  - il corpo (body) del messaggio, che contiene informazioni che rappresentano dati in qualunque formato

```
Content/Type: application/json
```

I codici di stato

- I codici di stato, o status code, sono dei codici restituiti dai server HTTP per indicare al client l'esito di una richiesta.
- Sono stati definiti dall'IETF e sono circa 50 e sono stati suddivisi per categoria:
  - 1 codici da 100-199 (Information)
  - 2 codici da 200-299 (Successful)
  - 3 codici da 300-399 (Redirection)
  - 4 codici da 400-499 (Client error)
  - 5 codici da 500-599 (Server error)

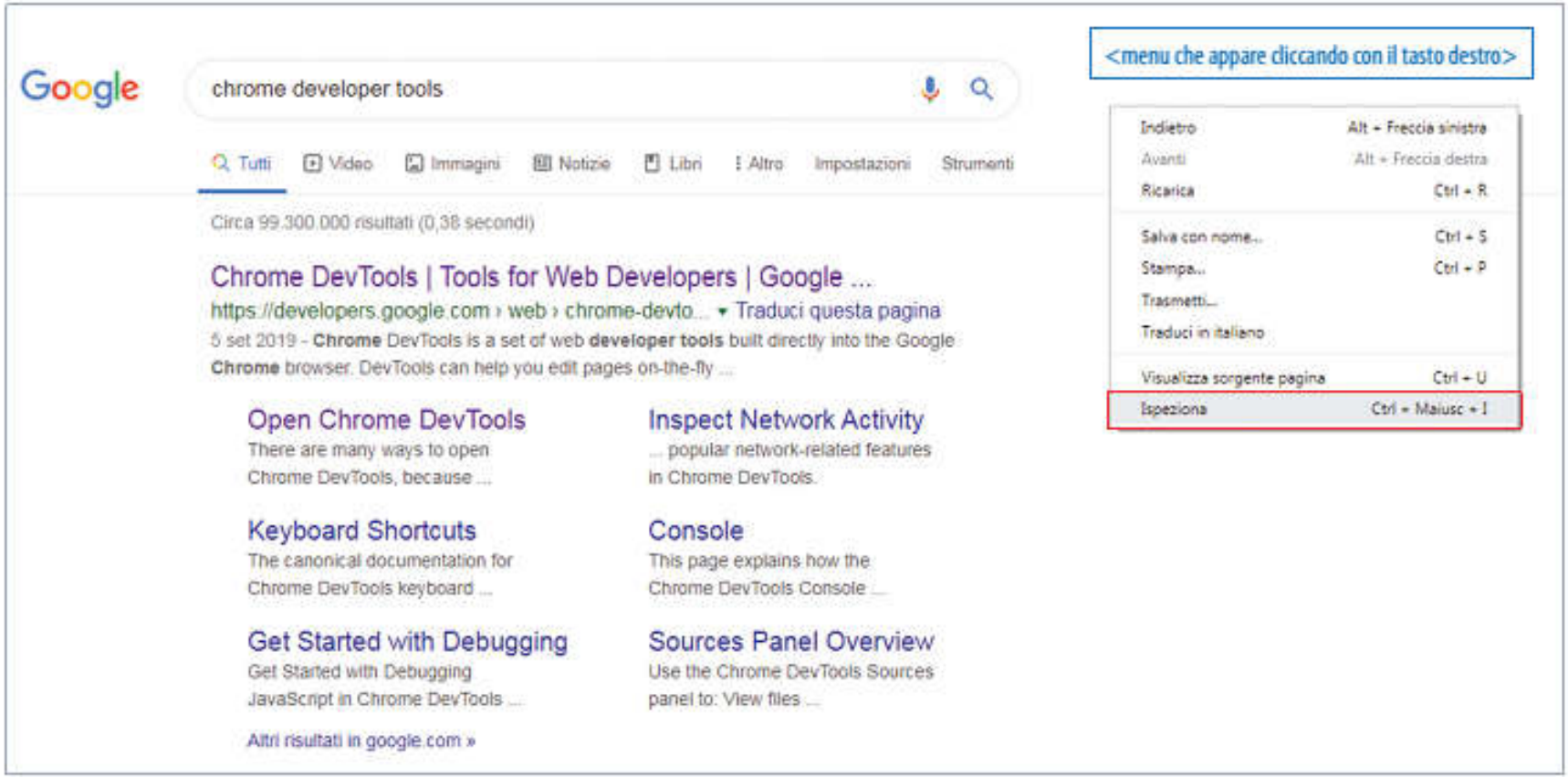
I codici di stato – esempi

```
100 Continue

200 OK

201 Created

204 No content

206 Partial Content

301 Moved Permanently
302 Found
303 See Other

400 Bad request

401 Unauthorized

403 Forbidden

404 non trovato

405 metodo non permesso

500 errore interno al server
```

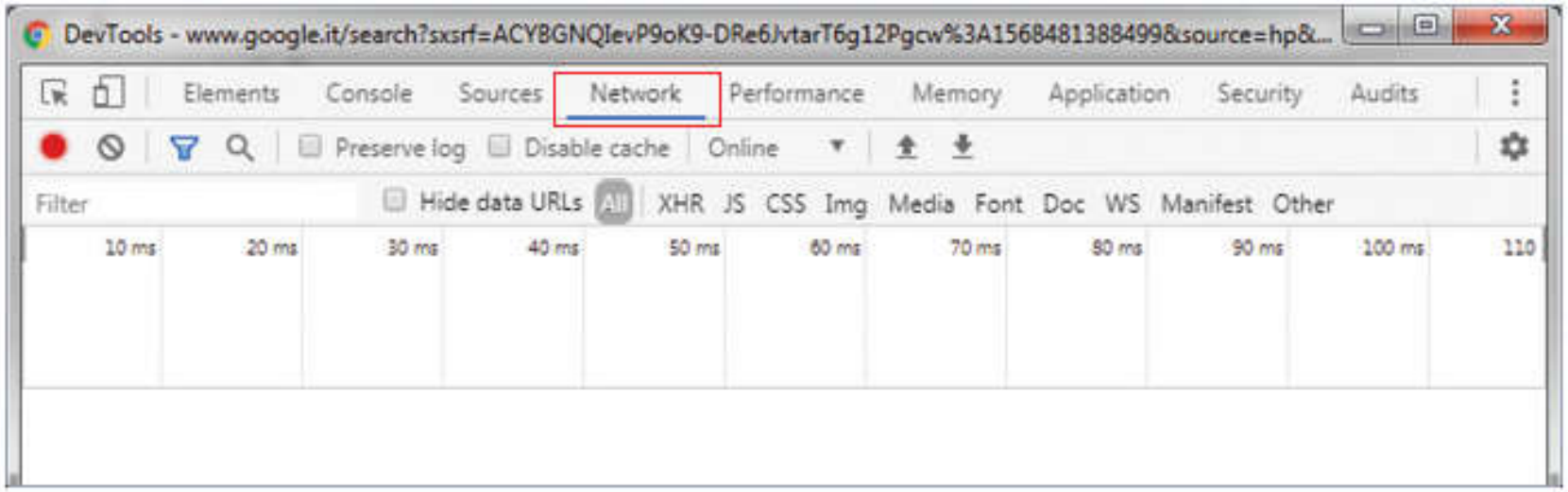
Come vedere il funzionamento di HTTP

- Se usate Chrome e i suoi strumenti, è disponibile Chrome Developer Tools che nella sua versione completa permette:
  - la navigazione all'interno del DOM delle pagine Web;
  - l'analisi e la modifica dell'HTML e dei fogli di stile in tempo reale;
  - il monitoraggio, il debug e le modifiche del codice JavaScript in tempo reale.

Come vedere il funzionamento di HTTP

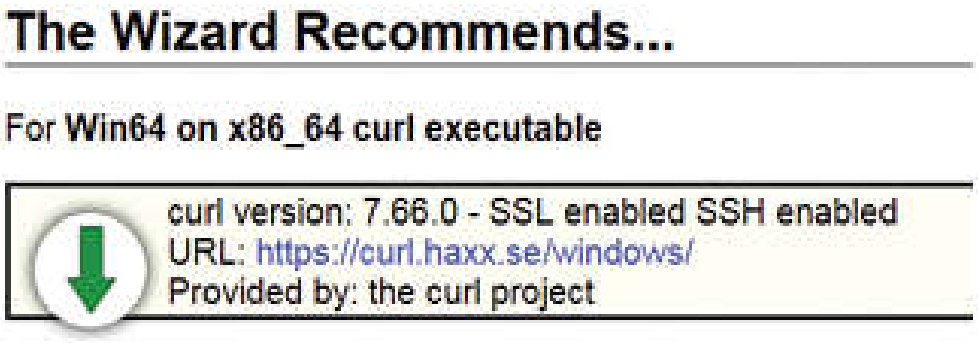


I codici di stato



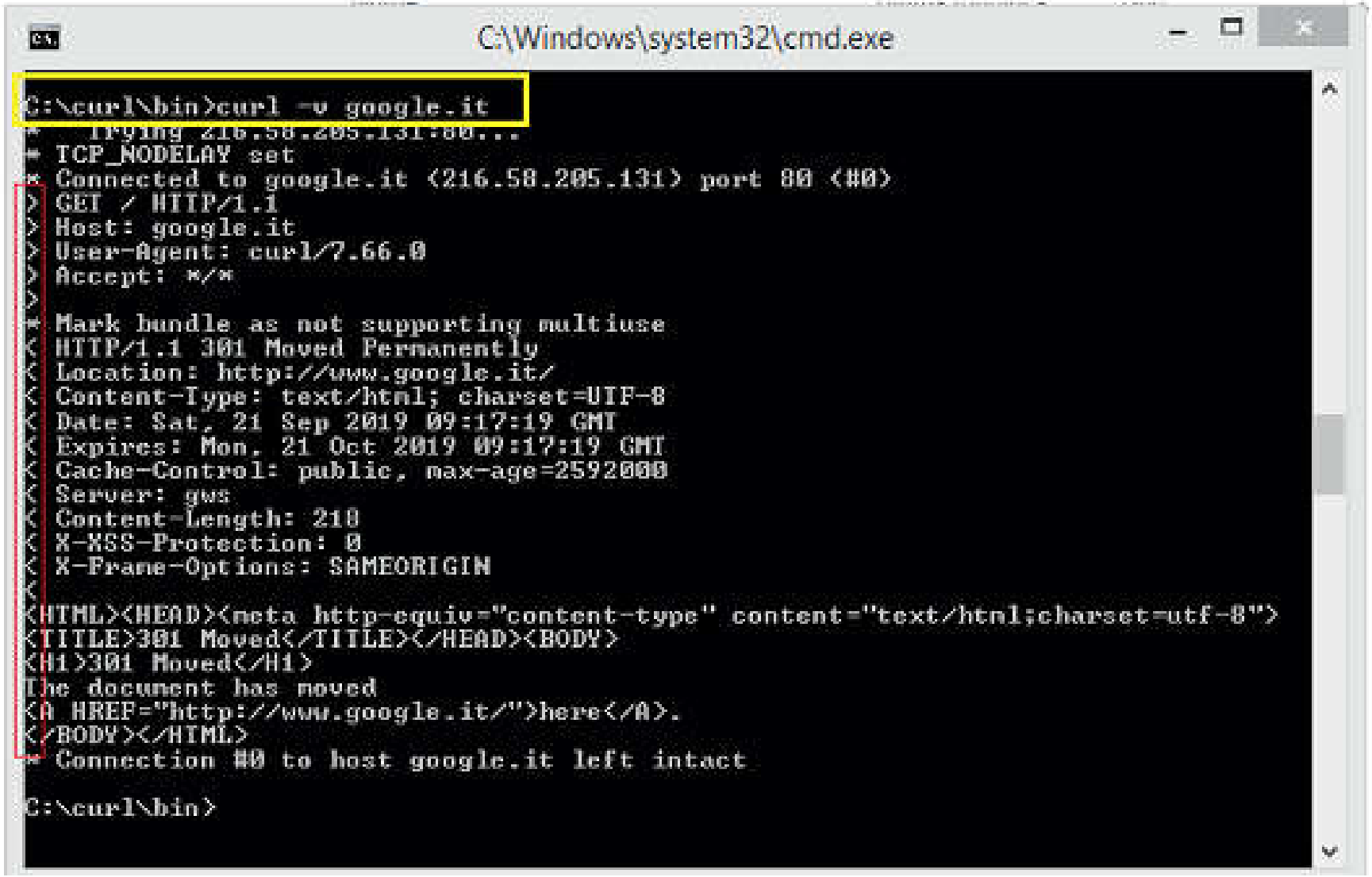
cURL

- Un altro utile metodo per prendere confidenza con l'HTTP è di usare un client dedicato
- Noi utilizzeremo **cURL**



I codici di stato

- cURL è un software gratuito e open source definito dai sui stessi autori come “command line tool and library for transferring data with URLs”.
- cURL viene utilizzato nelle righe di comando o negli script per trasferire i dati in tutti i tipi di dispositivi, dal PC al tablet, dai televisori alle stampanti, dai router alle apparecchiature Audio EDD.
- cURL viene utilizzato per effettuare i test dei siti Web, sia semplici come quelli composti da poche pagine che noi realizzeremo, sia estremamente complessi come Facebook o YouTube.
- cURL è un software da riga di comando per Shell/DOS che, oltre a HTTP, interagisce con svariati protocolli, come FTP e TELNET.


C:\Windows\system32\cmd.exe
C:\curl\bin>curl -v google.it
*   Trying 216.58.205.131:80...
* TCP_NODELAY set
* Connected to google.it (216.58.205.131) port 80 (#0)
> GET / HTTP/1.1
> Host: google.it
> User-Agent: curl/7.66.0
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 301 Moved Permanently
< Location: http://www.google.it/
< Content-Type: text/html; charset=UTF-8
< Date: Sat, 21 Sep 2019 09:17:19 GMT
< Expires: Mon, 21 Oct 2019 09:17:19 GMT
< Cache-Control: public, max-age=2592000
< Server: gws
< Content-Length: 218
< X-XSS-Protection: 0
< X-Frame-Options: SAMEORIGIN
<
<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.it/">here</A>.
</BODY></HTML>
* Connection #0 to host google.it left intact
C:\curl\bin>

SCELTA MULTIPLA

1 Quale tra le seguenti richieste del metodo GET non è permessa:

a richiesta assoluta
b richiesta di test
c richiesta parziale
d richiesta condizionale

2 La codifica URL del carattere "e commerciale" (ampersand o "&") è:

a non può essere codificata in URL encode
b %20
c %AC
d %26

3 Indica quale campo non è contenuto della seguente query string: pippo.php?uno=b%2A&esse=ciao%3E&A=a%26b

a E
b uno
c esse
d A

4 Quale tra i seguenti è un argomento valido del metodo PUT?

a Risorsa da ottenere nella response del server
b Risorsa da ottenere facendo un GET in seguito
c Risorsa da ottenere facendo un POST in seguito
d Un header di test della connessione

5 Qual è il limite della dimensione dei campi di una richiesta POST, in HTTP/1.1?

a 1024 byte
b Nessun limite
c 1024 bit
d 256 kB

6 I campi GET vengono inviati al server:

a memorizzati nel body message
b memorizzati nell'intestazione
c non possono essere inviati campi
d dalla successiva richiesta GET

7 I campi POST vengono inviati al server:

a memorizzati nel body message
b memorizzati nell'intestazione
c non possono essere inviati campi
d dalla successiva richiesta GET

VERO/FALSO

1 I client sono elementi passivi. V F
2 Un URL identifica sempre una risorsa. V F
3 Il protocollo TCP utilizza HTTP come protocollo di trasporto. V F
4 Le righe di header HTTP sono formate da un nome, seguito dai due punti (:) e da un valore. V F
5 La versione del browser prende il nome di User-Agent. V F
6 Nei messaggi HTTP gli header si concludono con una riga vuota che è facoltativa. V F
7 Una riga di richiesta HTTP può contenere anche un pathname di una risorsa. V F
8 La prima riga di una request HTTP contiene solo la versione http. V F
9 Il metodo TRACE può essere usato per API di applicazioni RESTful. V F
10 La codifica URL si applica solo ai campi form. V F
11 Il metodo PUT non può essere utilizzato per effettuare un upload di un file. V F
12 Nel metodo POST non esistono limiti di lunghezza nei parametri della richiesta. V F

COMPLETAMENTO

GET – versione – header – RESTful – PUT – client – body – server – POST – MIME

1 Nella prima riga di una HTTP request vi è la _____ del protocollo HTTP utilizzato, mentre nelle righe successive vengono indicate le _____.
2 Il tipo _____ è un dato formato da due componenti che identifica la classe e il formato di ogni tipo di file. Per esempio text/plain o audio/mpeg sono alcuni dei formati disponibili.
3 Nella sezione _____ della risposta del server viene indicato il codice HTML della pagina richiesta.
4 Le applicazioni _____ consentono di effettuare operazioni di tipo CRUD sui dati presenti nei server sfruttando i metodi _____, POST, _____, DELETE del protocollo HTTP.
5 Attraverso il metodo GET inizia lo scambio di due messaggi: il _____ inizia mandando un messaggio al _____ che identifica l'oggetto richiesto tramite il campo URI.