Informatica A e B
Antonio Lieto

Parte IV
Il Web

ARGOMENTI DI QUESTO GRUPPO DI LUCIDI

- **WEB**
  - Il Web come applicazione Internet
  - Il Web da ipertesto distribuito a sistema di risorse, applicazioni software, servizi e dati
  - URL (Uniform Resource Locator)
  - Il protocollo HTTP
  - I cookies
  - Il browser
  - Pagine Web: considerazioni generali

IL WEB: un'applicazione Internet

- Come detto, il World Wide Web (o WWW) o semplicemente Web è un'applicazione Internet, specificata dal protocollo HTTP (HyperText Transfer Protocol)
- E' realizzato secondo il modello client/server e usa TCP come servizio di trasporto
- il ruolo del client è giocato da specifici software detti semplicemente **HTTP client**,quello del server da applicazioni software dette **Web server**
- **N.B. I browser (descritti in seguito) contengono sempre al proprio interno un HTTP client!**

BREVISSIMA STORIA DEL WEB

Il Web nasce nel 1990 al CERN (European Particle Physics Laboratory) di Ginevra, come mezzo per facilitare la collaborazione scientifica tra centri di ricerca di fisica delle particelle. Il suo ideatore è Tim Berners-Lee (assieme a Robert Cailliau)

La prima interfaccia grafica per il WWW (il browser Mosaic) uscì all'inizio del 1993, per opera di Marc Andreessen, che un anno dopo fondò la Netscape Communication Corp.

Nel 1994, CERN e MIT stipularono un accordo per costituire il Consorzio WWW (W3C), dedicato alla standardizzazione e allo sviluppo di protocolli e linguaggi per il Web. Al W3C hanno in seguito aderito centinaia di altri enti e resta il principale punto di riferimento per tutto ciò che riguarda il Web (www.w3.org)

IL CONCETTO DI IPERTESTO

Un ipertesto è un documento (o un insieme di documenti) costituito da un insieme di nodi (i singoli documenti) e da un insieme di archi che collegano i vari nodi

Un ipertesto può essere visto come una rete, dentro la quale si può navigare, passando da un nodo all'altro percorrendo gli archi. Ogni arco è associato ad un elemento di un nodo (es. una parola), "attraverso" il quale è possibile accedere ad un altro nodo

IL WEB: un enorme ipertesto multimediale...ma non solo!

Il Web si afferma inizialmente come un enorme ipertesto distribuito: i nodi (pagine Web) e gli oggetti che le costituiscono risiedono su macchine sparse in tutto il mondo e collegate tra loro tramite Internet.

Il WWW è stato sin da subito un ipertesto "aperto" perché la sua struttura non è definita a priori, ma può cambiare nel tempo attraverso l'aggiunta di nuovi nodi (pagine Web) e nuovi archi (collegamenti)

In un tale ipertesto, percorrere un arco o collegamento ipertestuale (detto hyperlink) significa attivare una connessione tra un HTTP client (eventualmente interno ad un browser) e un Web server; il dialogo tra client e server è basato sul HTTP e corrisponde alla richiesta di una da parte del client di una pagina Web (o, anche, come vedremo, di un contenuto multimediale) che (se esiste e non sussistono altri impedimenti) viene inviata dal server al client

IL WEB: un enorme ipertesto multimediale...ma non solo!

Nel corso del tempo, il Web si evolve in varie direzioni:

1. l'informazione disponibile su Web acquisisce presto carattere di **multimedialità**: non solo testi, ma anche immagini, suoni e filmati

2. successivamente, accanto alle risorse informative, vengono esposti su Web veri e propri **servizi**, realizzati da applicazioni software (potenzialmente complesse), dette "applicazioni Web".
   Troviamo su Web servizi **fruibili da parte di persone** (es., siti di commercio elettronico, servizi di prenotazione, applicazioni di supporto al Customer Relationship Management, ecc., ecc.), ma anche servizi, detti **Web Services**, **fruibili da parte di applicazioni software**. Spesso, uno stesso servizio è esposto su Web sia con interfaccia rivolta alle persone, sia come Web service

3. i contenuti informativi come testi, immagini, suoni e filmati presenti su Web hanno come fruitori principali gli utenti umani; il significato (la semantica) di tali contenuti informativi è infatti immediatamente accessibile alle sole persone. Recenti sviluppi stanno conducendo alla costruzione di un Web semantico o Web di dati contenente (anche) informazioni il cui significato (o un frammento di esso) è immediatamente accessibile anche ad applicazioni software (**Web semantico** o **Web di dati**)

PAGINE WEB, SITI, HOME PAGE

Pagina Web

Una pagina Web è un nodo dell'ipertesto Web. Una pagina Web può contenere testo, immagini, audio, video, ecc...

Sito Web

Un sito Web è un insieme di pagine Web, generalmente residenti sullo stesso server, gestite da un unico Web master e con un contenuto omogeneo

Home page

Una home page è generalmente la pagina di accesso ad un sito Web

URL (Uniform Resource Locator) (I)

- **Gli URL (Uniform Resource Locator) sono stringhe di testo che identificano risorse quali le pagine Web, i file audio, video, immagini, invocazioni di servizi, ecc.[1] e ne specificano l'ubicazione**
- **Esempi di possibili URL:**
  - http://www.di.unito.it/~lieto/teaching.html
  - http://209.136.79.34/attivita/progetti.html
  - http://www.bellazienda.com:8040/index.html
  - http://www.acme.com/people/john_smith.php?par1=234
  - http://www.acme.com/index.html#products
- **Esempi di uso di URL: nella barra indirizzi dei browser**

[1] **Attualmente, gli URL sono utilizzati in ambito Semantic Web anche per identificare entità del mondo reale e concetti astratti...**

URL (Uniform Resource Locator) (II)

Struttura generale di un URL:

nome o indirizzo IP di un host, es. www.di.unito.it, www.acme.com, 209.136.79.34

percorso che individua la risorsa nel sistema di cartelle del server, es. /www.di.unito.it/~lieto/teaching.html, attivita/progetti.html, index.html

http://<host>:<porta>/<percorso>?<richiesta>#<frammento>

porta (si veda la descrizione del livello di trasporto), es. www.bellazienda.com:8040,

Se la porta non è specificata, si intende implicitamente la porta 80

URL (Uniform Resource Locator) (II)

Struttura generale di un URL:

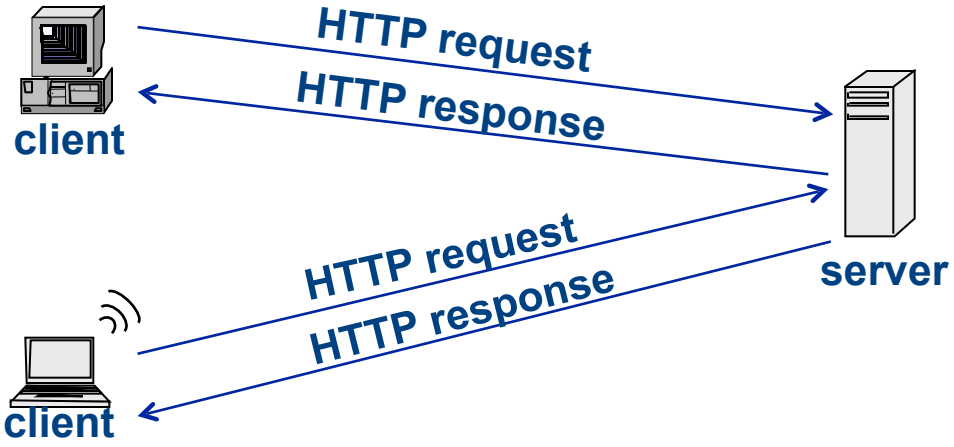
http://<host>:<porta>/<percorso>?<richiesta>#<frammento>

invia al server una serie di dati, nella forma "parametro = valore", es. http://www.acme.com/people/john_smith.php?par1=234, richiede al server la risorsa http://www.acme.com/people/john_smith.php e gli passa anche l'informazione che "par1" vale 234

individua un frammento all'interno della risorsa (cioè una parte della risorsa, come, ad. esempio, una sezione all'interno di un documento Web) , es. http://www.acme.com/index.html#products individua la sezione etichettata "products" all'interno della pagina Web "index.html"

IL PROTOCOLLO HTTP (I)

- Il protocollo HTTP specifica come deve avvenire l'interazione fra un client HTTP ed un server HTTP
- L'interazione fra un client HTTP ed un server HTTP consiste nello scambio di messaggi di due diversi tipi: messaggi di tipo HTTP request (inviati dal client al server) e messaggi di tipo HTTP response (inviati dal server al client)



- Ogni messaggio HTTP, sia esso di tipo HTTP request o di tipo HTTP response, è strutturato in due parti: **header** (intestazione) **+ body** (contenuto)

IL PROTOCOLLO HTTP (II): esempio di HTTP request

```
GET /~lieto/teaching.html HTTP/1.1
Host: www.di.unito.it
User-agent: Mozilla/4.0
Accept: text/html, image/gif, image/jpeg
Accept-language: it
```

Header

Body (in questo esempio, vuoto)

IL PROTOCOLLO HTTP (IV): formato dei messaggi HTTP request

- Notiamo, innanzi tutto, che i messaggi HTTP request (e anche HTTP response) sono messaggi di tipo testuale
- Riportiamo qui sotto la struttura generale di un HTTP request:

metodo URL versione

nome campo: valore

...

nome campo: valore

body

IL PROTOCOLLO HTTP (IV): formato dei messaggi HTTP request

contiene l'indicazione dell'azione richiesta: sono possibili vari valori; fra questi i più frequenti sono:

GET: esprime la richiesta di una risorsa (indicata nel campo URL). Con GET, il body è vuoto. E' il metodo di uso più frequente

POST: esprime la richiesta di una risorsa (indicata nel campo URL) e invia dati al client, inseriti nel *body*. Un suo uso tipico è per inviare al server i dati inseriti dall'utente in un modulo online

metodo URL versione

nome campo: valore

...

nome campo: valore

body

IL PROTOCOLLO HTTP (IV): formato dei messaggi HTTP request

- Notiamo, innanzi tutto, che i messaggi HTTP request (e anche HTTP response) sono messaggi di tipo testuale (a differenza degli altri tipi di messaggi visti finora, es. datagrammi IP e segmenti UDP e TCP)

identificatore della risorsa richiesta. Spesso specifica un percorso nel file system del server, anche se in alcuni casi esprime un percorso solo virtuale che il server traduce in un percorso reale nel proprio file system

metodo URL versione

nome campo: valore

...

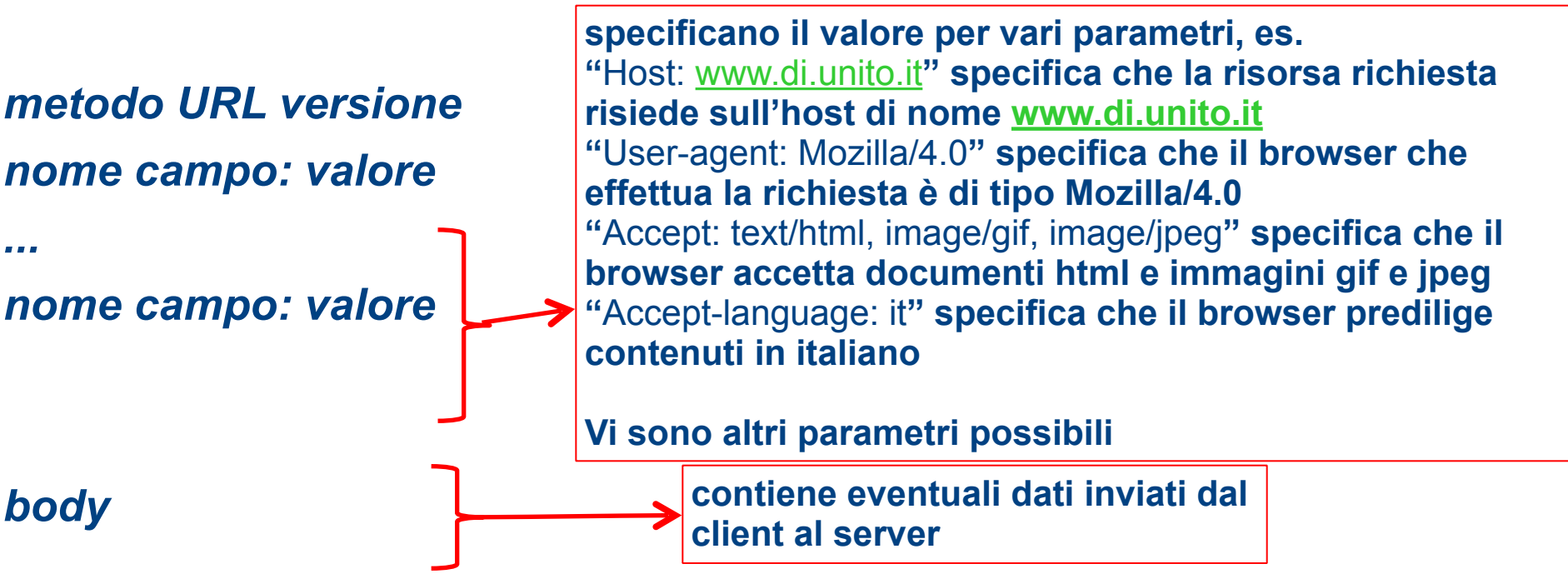
nome campo: valore

body

versione del protocollo HTTP utilizzata

IL PROTOCOLLO HTTP (IV): formato dei messaggi HTTP request

- **Notiamo, innanzi tutto, che i messaggi HTTP request (e anche HTTP response) sono messaggi di tipo testuale Riportiamo qui sotto la struttura generale di un HTTP request:**



IL PROTOCOLLO HTTP (III): esempio di HTTP response

```
HTTP/1.1 200 OK
Date: Thu, 10 Nov 2011 12:00:12 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Fri, 28 Oct 2011 11:00:34 GMT
Content-Type: text/html
…

<HTML>
<HEAD>
…
</HEAD>
  ...
</HTML>
```

Header

Body (in questo esempio, contenente la risorsa richiesta)

IL PROTOCOLLO HTTP (V): formato dei messaggi HTTP response

- **Riportiamo qui sotto la struttura generale di un HTTP response:**

versione codice_di_stato frase

nome campo: valore

...

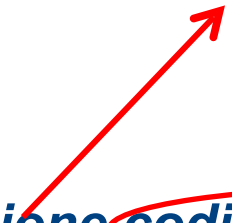
nome campo: valore

body

IL PROTOCOLLO HTTP (V): formato dei messaggi HTTP response

- **Riportiamo qui sotto la struttura generale di un HTTP response:**

versione del protocollo HTTP utilizzata



versione codice_di_stato frase

nome campo: valore

...

nome campo: valore

body

comunicano al client :il risultato della richiesta, es: "200 OK" specifica che la richiesta è stata soddisfatta con successo, "404 Not Found" specifica che la risorsa richiesta non esiste sul server
Vi sono 5 categorie di codici di stato:
1xx: informazione (richiesta ricevuta, l'elaborazione sta continuando)
2xx: successo (l'azione è stata ricevuta con successo, compresa e accettata)
3xx: redirezione (per portare a termine la richiesta sono richieste ulteriori azioni)
4xx: errore del client (la richiesta contiene errori di sintassi o non può essere soddisfatta)
5xx: errore del server (il server non è stato in grado di soddisfare una richiesta apparentemente valida)
[categorie di codici tratte da L.L. Peterson, B. S. Davie, *Reti di calcolatori*, Apogeo, 2004]

IL PROTOCOLLO HTTP (V): formato dei messaggi HTTP response

- **Riportiamo qui sotto la s**

specificano il valore per vari parametri, es.
"Date: Thu, 10 Nov 2011 12:00:12 GMT" specifica la data di invio della risposta da parte del server
"Server: Apache/1.3.0 (Unix)" specifica che il server è di tipo Apache/1.3.0 per sistema operativo Unix
"Last-Modified: Fri, 28 Oct 2011 11:00:34 GMT" specifica quando la risorsa inviata è stata modificata per l'ultima volta
"Content-Type: text/html" specifica che la risorsa inviata è un documento html

Vi sono altri parametri possibili

versione codice_di_stato f

nome campo: valore

...

nome campo: valore

body

contiene la risorsa inviata. Un caso tipico è quello in cui il body contiene una pagina Web

I COOKIES (I)

- I cookies sono informazioni di tipo testuale create dal server e memorizzate sul client
- Ad esempio, quando un client invia un primo messaggio di richiesta ad un server di un sito di commercio elettronico, il server crea un codice identificativo per il client e lo memorizza nella propria base di dati (assieme a tutte le informazioni utili relative al client, come per esempio, le pagine visitate, i prodotti acquistati, ecc.)
- Il server, invia, sotto forma di cookie, l'identificatore al client (con la data di scadenza e il dominio di validità), chiedendogli implicitamente di memorizzarlo e di comunicarlo allo stesso server nelle interazioni successive. L'invio del cookie è fatto tramite il campo "Set-cookie" nell'HTTP response
- Il client che riceve l'HTTP response con il cookie, memorizza quest'ultimo in una propria cartella
- Quando il client ricontatta lo stesso server (richiedendo risorse che ricadono nel dominio di validità), gli invia il cookie nell'HTTP request, specificandolo nel campo "cookie"

Si noti che i cookie vengono memorizzati in una cartella all'interno del proprio elaboratore e possono essere facilmente rimossi. Inoltre, è possibile disabilitare i cookies sul proprio browser

I COOKIES (II): esempi

Esempio di HTTP response, in cui viene inviato al client un cookie con il codice identificativo creato dal server:

```
HTTP/1.1 200 OK
Date: Thu, 10 Nov 2011 12:00:12 GMT
Server: Apache/1.3.0 (Unix)
Last-Modified: Fri, 12 Oct 2011 11:00:34 GMT
Content-Type: text/html
Set-Cookie: codice_utente=6754; expires= Sat,
   10 Nov 2012 12:00:12 GMT;
   domain=www.acme.com
…

<HTML>
<HEAD>
…
</HEAD>
   …
</HTML>
```

Esempio di HTTP request, in cui il client invia al server il cookie che identifica lo stesso client:

```
GET /products/professional.html  HTTP/1.1
Host: www.acme.com
User-agent: Mozilla/4.0
Accept: text/html, image/gif, image/jpeg
Accept-language: it
Cookie: codice_utente=6754
```

I COOKIES (III)

NOTE:

(1) Per utilizzarli è necessario che il client "accetti" i cookies ⇒ è possibile modificare la configurazione del browser!

(2) I Cookies sono memorizzati nel filesystem locale del sistema su cui gira il client Web

(3) L'utente può cancellarli! ⇒ non sono un metodo tanto affidabile per mantenere informazioni importanti…

I COOKIES DI PRIME E DI TERZE PARTI (I)

- Un cookie di prime parti (first-party cookie) è un cookie che proviene direttamente dal sito che si sta visitando (nel caso di accesso tramite browser, quello indicato nella barra indirizzi del browser), es. visito il sito www.acme.com e il client riceve un cookie in cui "domain = www.acme.com": si tratta si un first-party cookie

- Un cookie di terze parti (third-party cookie) è invece un cookie che ha una provenienza diversa da quella del sito che si sta visitando (cioè, associato ad un dominio diverso da quello del sito che si sta visitando), es. visito il sito www.acme.com e il client riceve un cookie in cui "domain = prodottibelli.com": si tratta si un third-party cookie

I COOKIES DI PRIME E DI TERZE PARTI (II)

- Come è possibile che il client riceva cookies di terze parti?
- In questo modo: visito una pagina del sito www.acme.com; tale pagina contiene elementi (l'esempio classico è quello dei banner pubblicitari) provenienti da altri domini, es. prodottibelli.com; per ottenere tali elementi, il browser (meglio: il client HTTP al suo interno) contatta il server di prodottibelli.com e questo gli invia un cookie, associato al proprio dominio
- ...una breve riflessione su quanto sopra, ci consente di capire che la distinzione fra first-party cookie e third-party cookie è solo una distinzione fittizia e "di comodo", basata sul punto di vista di un utente che visita consapevolmente un certo sito Web e che è ignaro di ciò che avviene "dietro le quinte" del proprio browser: in realtà, non sussiste alcuna differenza tecnica o "ontologica" fra cookies di prime-parti e cookies di terze parti: tecnicamente, sono entrambi soltanto cookies

I COOKIES DI PRIME E DI TERZE PARTI (III)

- A che servono i cookies di terze parti?
- Un loro uso tipico è legato alla pubblicità e alla tracciabilità degli utenti
- Es., supponiamo che l'azienda *prodottibelli*, riconducibile al dominio prodottibelli.com, abbia dei banner pubblicitari sia nelle pagine del sito www.acme.com, sia in quelle del sito www.tuttovero.it; quando con un browser visito www.acme.com, il client riceve, assieme al banner pubblicitario di *prodottibelli*, anche un third-party cookie associato a prodottibelli.com. Se successivamente, con il medesimo browser, accedo a www.tuttovero.it, il client, per scaricare il banner di *prodottibelli*, invia al server di prodottibelli.com la richiesta del banner, più il third-party cookie precedentemente ottenuto durante la navigazione nel sito www.acme.com. In questo modo, l'applicazione Web associata a prodottibelli.com può conoscere che quello stesso client si era connesso in precedenza al sito www.acme.com...

IL BROWSER (I)

Principali funzionalità del browser:

- funge da interfaccia utente nell'accesso al Web
- esibisce le funzionalità di HTTP client
- gestisce le risorse ottenute dai server. Nei frequenti casi in cui la risorsa ottenuta è una pagina Web, questo si traduce nella visualizzazione della pagina stessa. Inoltre, se la pagina contiene al proprio interno altre risorse (immagini, audio, ecc.), richiede automaticamente tali risorse ai server opportuni. I browser sono inoltre in grado di interpretare correttamente i linguaggi standard per la descrizione di pagine Web, come HTML ed XHTML (trattati in seguito) e di visualizzare correttamente molti formati per le immagini (JPEG, ecc.)
- per gestire certi tipi di contenuti, i browser si possono avvalere di altri programmi a cui sono "collegati" (es., i cosiddetti "**plugin**"). Esempi sono i programmi per la visualizzazione di contenuti video o per la riproduzione di certi formati audio

IL BROWSER (II)

Principali funzionalità del browser (cont.):

- effettua il caching locale delle risorse caricate, cioè memorizza sul file system locale le risorse caricate, in modo da renderle più velocemente disponibili in caso dovessero essere nuovamente richieste. Questo meccanismo tiene conto dell'eventuale scadenza indicata dal server per le risorse (in modo da evitare, quando possibile, di rendere disponibili risorse obsolete). L'utente può comunque forzare la richiesta al server della risorsa.

IL BROWSER (III)

- **Sono moltissimi i browser attualmente a disposizione; quelli attualmente più diffusi sono:**
  - Firefox
  - Google Chrome
  - Internet Explorer
  - Opera
  - Safari

- **N.B: Una cosa è il browser, altra cosa è il protocollo HTTP: i diversi browser possono differenziarsi anche notevolmente gli uni dagli altri, ma essi, per poter navigare il Web, devono tutti attenersi al protocollo HTTP**

NAVIGARE SUL WEB

Cosa vuol dire "navigare" sul Web?

Quando un utente si connette ad un sito Web, per es. "cliccando" su un link o specificando un URL nell'apposito spazio, tipicamente succedono le seguenti cose (tralasciamo l'eventuale ruolo del proxy server):

- Il browser analizza l'**URL**
- Riceve via DNS l'**indirizzo IP** corrispondente al nome dell'host contenuto nell'URL
- Effettua una **connessione** TCP al server corrispondente all'IP ricevuto e gli invia una **richiesta** per la risorsa indicata nell'URL
- Il server **risponde** inviando la risorsa richiesta
- Il browser **intepreta** la risorsa ricevuta, **visualizzandola** secondo le specifiche in essa contenute

PAGINE WEB (I)

Quando ci connettiamo ad una risorsa in rete, identificata da un URL (digitando l'indirizzo nel browser, cliccando su un link, …):

1. Nel caso più semplice l'URL specifica una pagina (generalmente scritta in HTML o XHTML) precedentemente creata (solitamente da una persona) e salvata in un file sul server, per es:

   *http://www.di.unito.it/~lieto/**teaching.html***

   NB: Se il nome del file non compare, per es:

   http://www.di.unito.it/

   si fa riferimento ad una pagina di default, che generalmente è *index.html* (dipende dai settaggi del sever)

PAGINE WEB (II)

2. l'indirizzo di una **pagina** dinamica (per esempio scritta in ASP, PHP, o JSP) il cui **contenuto viene generato** (selezionato, composto) **in parte o in toto al momento della richiesta**, per es:

   *http://www.di.unito.it/~lieto/**teaching.php***

3. l'indirizzo di un **programma** (per esempio una Java Servlet), per es: *http://www.di.unito.it/~lieto/**didaServlet***

PAGINE WEB (III)

(1) Pagine "statiche" vs pagine "dinamiche“

(2) Pagine realizzate utilizzando tecnologie client-side vs server-side

(3) Linguaggi di mark-up vs linguaggi di scripting e di programmazione

PAGINE WEB (IV)

(1) Pagine "statiche" vs pagine "dinamiche“

- **Pagine "statiche"** = pagine create a-priori (solitamente da una persona) e salvate come file sul server; tipicamente, sono scritte in HTML o XHTML; sono inviate sempre uguali dal server ai vari client che ne fanno richiesta; ogni client che le riceve le presenta all’utente al momento della ricezione; l’utente non ha alcun modo di interagire con essa (può solo accedere ai suoi contenuti tramite la presentazione realizzata dal client e attraversare i collegamenti ipertestuali)

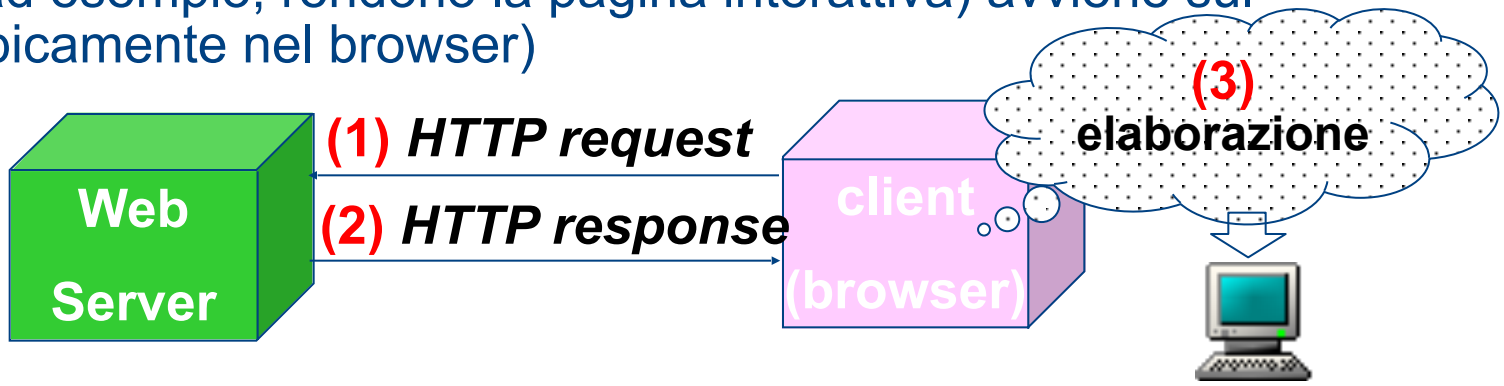
PAGINE WEB (V)

- **Pagine "dinamiche“**. Due accezioni del concetto di dinamicità, per cui una pagina dinamica può essere:
  a) una pagina in grado di reagire agli eventi prodotti dall'utente sul client (quindi una pagina interattiva) o almeno in grado di esibire qualche variazione nel contenuto o nella sua visualizzazione (anche indipendente da eventi prodotti dall'utente, es.: movimento autonomo di elementi della pagina), come risultato diretto dell'esecuzione di un qualche programma contenuto nella pagina stessa; spesso, una tale pagina contiene al proprio interno dei programmi scritti in JavaScript
  b) una pagina costruita (in toto o in parte) dinamicamente e in tempo reale sul server, in base alla richiesta che l'utente ha formulato tramite il client e a cui non corrisponde un file statico salvato sul server (quindi può variare ad ogni richiesta); tipicamente, una tale pagina è realizzata tramite linguaggi come PHP, Python, Java, ecc.
  c) una pagina che presenta entrambe le suddette caratteristiche

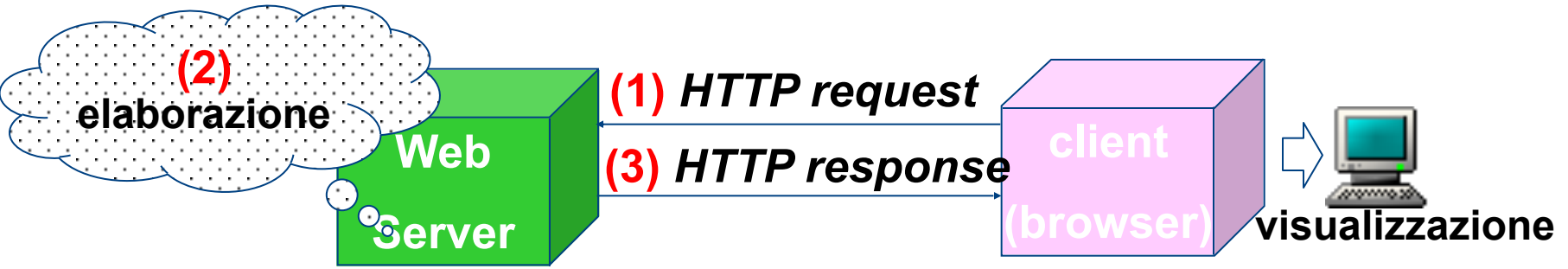
PAGINE WEB (VI)

(2) Tecnologie client-side vs server-side

- **Tecnologie client-side** = tecnologie in cui l'elaborazione e l'esecuzione dei programmi eventualmente contenuti nella pagina (e che, ad esempio, rendono la pagina interattiva) avviene sul client (tipicamente nel browser)



- **Tecnologie server-side** = tecnologie in cui l'esecuzione dei programmi (che, ad esempio, generano in toto o in parte la pagina) avviene sul server (tipicamente nel Web server)



PAGINE WEB (VII)

(3) Linguaggi di mark-up vs di programmazione e di scripting

- **Linguaggi di mark-up** = servono a scrivere **documenti strutturati** : un documento strutturato è un contenuto (spesso testuale) corredato di **meta-informazioni**, espresse tramite *tag* e *attributi*, che **specificano informazioni sul contenuto** (informazioni spesso relative alla **struttura** o alla **visualizzazione**)

  Es: HTML, XML, XHTML

- **Linguaggi di programmazione** = servono a scrivere **programmi**: un programma è una sequenza di **istruzioni, la cui esecuzione produce un qualche effetto**

  Es: Java

  - **Linguaggi di scripting** = linguaggi di programmazione utilizzati per scrivere **script**

  script ="piccolo" programma contenuto in una pagina web[1]

  Es: PHP, JavaScript

[1]**In questo corso, considereremo solo gli script contenuti nelle pagine Web, anche se esistono altri tipi di script**