Java

html e http

G. Prencipe
prencipe@di.unipi.it

Introduzione

- Tutte le comunicazioni tra client e server Web avvengono mediate il protocollo *HTTP* (*HyperText Transfer Protocol*, attualmente alla versione 1.1), che è un insieme di regole per richiedere e fornire *risorse* Internet
- *Risorsa* è un termine generale che comprende i file ma non è limitato ad essi
  - È qualunque informazione che possa essere identificata da un URL (la R di URL sta per risorsa)

Introduzione

- La URL (che analizzeremo dopo) è essenzialmente un metodo per dire di quale risorsa il client necessita
  - In altre parole l'indirizzo della risorsa
- Il tipo più comune di risorsa è il file (una pagina ipertestuale, una immagine)
  - Una risorsa può anche essere il risultato di una richiesta, l'output di uno script CGI o altro

HTTP

- Il protocollo HTTP definisce un metodo di interazione client-server ottimizzato per le connessioni brevi e veloci necessarie per le connessioni tra client Web e server Web
- Si tratta di un protocollo
  - Generico
  - Stateless
  - Leggero

HTTP

- **Generico**: facilmente estensibile per comprendere servizi di vario tipo, non solo transazione ipertestuali
- **Stateless**: tra una connessione e l'altra il server non tiene nota della connessione precedente, e quindi tutte le connessioni sono trattate alla stessa maniera, come se si trattasse ogni volta di un nuovo client
- **Leggero**: il client si connette al server solo per il tempo strettamente necessario per trasmettere la risorsa, e quindi chiude la connessione

HTTP

- Si tratta di un funzionamento molto diverso da altri server, per esempio *FTP* (*File Transfer Protocol*)
  - La connessione in FTP è tenuta aperta per tutto il tempo che l'utente desidera, anche se non c'è nessun file da trasferire

URI

- Un *Uniform Resource Identifier* (*URI*) è una stringa di caratteri con una particolare sintassi che identifica una risorsa
  - File, indirizzo email, un libro, il nome di una persona, un host Internet….
- Una URI è composta da uno *schema* per la URI e da una parte *specifica*, separati da due punti

  schema:parte-specifica

URI

- La sintassi della parte specifica dipende dallo schema
- Gli schemi principali sono
  - file
  - ftp
  - http
  - mailto
  - news
  - telnet

URI

- Ci sono due tipi principali di URI
  - Uniform Resource Locator (URL)
    - Puntatore a una particolare risorsa su Internet, che si trova a una particolare locazione
      - *http://www.di.unipi.it/index.html*
  - Uniform Resource Names (URN)
    - Nome per una particolare risorsa ma senza riferimento alla particolare locazione
      - *urn:isbn:17263274389327*

URL HTTP

- L'URL è il modo utilizzato nel protocollo HTTP per indicare la risorsa richiesta
- La forma generale dell'URL è la seguente

  http://server[:port]/[path/][file]
  - dove server è l'indirizzo internet del server HTTP (il nome o l'indirizzo IP),
  - port è il numero della porta a cui connettersi: può essere omesso, e in tal caso prende il valore di default 80
  - path indica al server la directory del file richiesto
  - file è il nome del file richiesto

URL HTTP

- Per esempio l'URL

  http:sbrinz.di.unipi.it:80/Immagini/cherubino.gif
  - chiede di utilizzare il protocollo HTTP per reperire sul server Web il cui nome è sbrinz.di.unipi.it e che risponde alla porta 80 il file cherubino.gif che sta nella directory Immagini
- Più avanti vedremo forme di URL che possono invocare protocolli diversi (per esempio FTP)

Il modello di transazione

- Un client HTTP apre una connessione e spedisce una richiesta al server HTTP
- Il server risponde con un messaggio che normalmente contiene la risorsa richiesta
- Dopo aver risposto il server chiude la connessione
  - HTTP è un protocollo stateless, cioè non conserva informazioni di connessione tra una transazione e l'altra
- Il modello di transazione di HTTP è molto semplice, e questa è la principale ragione per cui si può facilmente estendere

Il modello di transazione

- Una tipica transazione avviene così
  - Il client HTTP (browser) stabilisce una connessione con un server HTTP remoto
  - Le informazioni sul server da contattare e sulla porta da usare sono comprese nel link ipertestuale
  - Il client localizza il server e inizia il processo di connessione.

Il modello di transazione

- Il client spedisce una **richiesta**
- Possono essere fatte richieste diverse che possono comprendere anche informazioni come
  - Tipo di dati il client può trattare
  - Linguaggio naturale che preferisce
  - Tipo di dati spedito al server

Il modello di transazione

- Il server processa la **risposta**
- A questo punto possono succedere cose diverse
  - Molte risposte sono processate dal server stesso (la maggior parte delle volte il processo consiste nel localizzare un file e restituirlo al cliente)
  - Altre vengono passate ad altre applicazioni, soprattutto CGI
    - Queste applicazioni rispondono al server il quale passa le informazioni al client

Il modello di transazione

- Il server **risponde** qualcosa al client
  - Potrebbe essere il file richiesto, una semplice conferma che la richiesta è stata processata, oppure un messaggio di errore
  - In HTTP sono definiti diversi codici per comunicare cosa è successo
- La richiesta potrebbe anche contenere informazioni su cosa il server sa fare e quali tipi di dati sa ritornare al client

Il modello di transazione

- Il server **chiude la connessione**
- I primi protocolli, come *telnet* e *ftp*, erano stati progettati con l'idea che l'utente si collegasse ad un sistema remoto per un periodo di tempo esteso, trasmettendo comandi diversi e tenendo la connessione aperta

Il modello di transazione

- Nel caso del Web invece è probabile che la prossima connessione avvenga in un tempo abbastanza successivo alla precedente, e/o che la richiesta sia diretta ad un altro server
  - In tal caso non c'è vantaggio nel tenere la connessione aperta o mantenere informazioni sul client
  - Quindi HTTP è stato progettato per chiudere la connessione appena ha finito di processare una richiesta

Richieste e risposte

- Client e server HTTP discutono tra loro con il protocollo HTTP
- Una richiesta HTTP, fatta dal client al server, è di questo tipo generale

<METHOD> <URI> "HTTP/1.0" <CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

Richieste e risposte

- Client e server HTTP discutono tra loro con il protocollo HTTP
- Una richiesta HTTP, fatta dal client al server, è di questo tipo generale

<METHOD> <URI> "HTTP/1.0" <CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

METHOD è una singola parola che dice al server che tipo di richiesta viene fatta

Richieste e risposte

- Client e server HTTP discutono tra loro con il protocollo HTTP
- Una richiesta HTTP, fatta dal client al server, è di questo tipo generale

<METHOD> <URI> “HTTP/1.0” <CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

URI è una parte dell’URL che essenzialmente dice il percorso e il nome della risorsa richiesta

Richieste e risposte

- Client e server HTTP discutono tra loro con il protocollo HTTP
- Una richiesta HTTP, fatta dal client al server, è di questo tipo generale

<METHOD> <URI> “HTTP/1.0” <CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

CRLF sono due caratteri ASCII (13 e 10) di a capo e fine linea

Richieste e risposte

- Client e server HTTP discutono tra loro con il protocollo HTTP
- Una richiesta HTTP, fatta dal client al server, è di questo tipo generale

<METHOD> <URI> “HTTP/1.0” <CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

Le righe di Header successive possono comprendere informazioni aggiuntive, come il nome del software client (User-Agent) e i tipi di MIME che il client può trattare

Richieste e risposte

- Client e server HTTP discutono tra loro con il protocollo HTTP
- Una richiesta HTTP, fatta dal client al server, è di questo tipo generale

<METHOD> <URI> “HTTP/1.0” <CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

BODY sono dati opzionali che possono essere aggiunti in qualche richiesta.

Richieste e risposte

- E il server risponde in questo modo

HTTP/1.0 <STATUS_CODE>
<REASON><CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

La risposta comprende la versione di HTTP

Richieste e risposte

- E il server risponde in questo modo

HTTP/1.0 <STATUS_CODE>
<REASON><CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

STATUS_CODE,cioè un codice di tre cifre che indica lo stato della risposta

Richieste e risposte

- E il server risponde in questo modo

HTTP/1.0 <STATUS_CODE>
<REASON><CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

REASON è il motivo della risposta

Richieste e risposte

- E il server risponde in questo modo

HTTP/1.0 <STATUS_CODE>
<REASON><CRLF>
<Header>: <Value>
...
<Header>: <Value>
<CRLF>
<BODY>

BODY contiene i dati che vengono trasferiti al client

Metodi di richiesta

- Nel protocollo HTTP 1.0 sono specificati sette metodi di richiesta (METHOD), cioè sette tipi di transazione
- Solo due o tre di esse sono comunemente usati, e in futuro ne possono essere aggiunti altri

Metodi di richiesta -- GET

- GET è il metodo più usato per chiedere informazioni a un server
- Quando un utente fa clic su un link di una pagina web, il client spedisce una richiesta GET per avere l'URL (cioè la risorsa) specificata nel link
- Il metodo GET può anche essere usato per trasmettere una richiesta ad una applicazione CGI che, come risultato del suo processo, restituisce dati

Metodi di richiesta -- GET

- Più in generale, la richiesta può includere una *query string*, sequenza di informazioni addizionali aggiunte alla fine della URL
- In tal caso i dati in ingresso vengono aggiunti in fondo all'URL (cioè all'indirizzo dell'applicazione) separati da un punto di domanda (?)
  - Es: http://….app?pippo
  - È possibile fare il bookmark della query string
  - Limitata in lunghezza (240 caratteri)
  - Esempio: http://www.whoami.com/Servlet/Search?name=Inigo+Montoya

Metodi di richiesta -- POST

- POST è il metodo usato per spedire ad un server dati che devono essere processati in qualche modo, per esempio da una applicazione CGI
- Le differenze rispetto ad un GET sono queste:
  - Tutte le informazioni addizionali (lunghezza illimitata) passate come parte della richiesta sono
    - Invisibili all'utente
    - Non possono essere ricaricate (reloaded)

Metodi di richiesta -- POST

- Nel body della richiesta c'è un blocco di dati che vengono spediti. Ci sono normalmente degli *header* extra che descrivere il body, come **Content-Type:** e **Content-Length:**
- L'URI richiesto non è un risorsa da recuperare: è normalmente un programma che riceve in input i dati che vengono spediti
- La risposta HTTP è normalmente l'output di un programma, non un file statico

Codici di stato

- La risposta del server HTTP alle domande del client HTTP comprende un codice di stato di tre cifre (STATUS_CODE) che indica lo stato della risposta
- Tutti i codici che iniziano con
  - 1 sono informativi
  - 2 significano successo
  - 3 ridirezione
  - 4 errore del client
  - 5 errore del server

Codici di stato

- Eccone alcuni:
  - 200: OK: la richiesta è stata processata e i dati vengono spediti
  - 204: No Content: la richiesta è stata processata ma non ha avuto risposta
  - 301: Moved Permanently: il file specificato è stato permanentemente spostato in un'altra locazione
  - 403: Forbidden: il client non è autorizzato a ricevere i dati richiesti
  - 404: Not Found: i dati specificati nell'URL non sono stati trovati
  - 500 Internal Server Error: il server è incorso in un errore inaspettato che non ha permesso di completare la richiesta

Java

html e http

fine