Applicazioni web

Parte 4
http

Alberto Ferrari

Sommario

- http
  - Metodi, intestazioni e codici di stato
    - get
    - post
  - Parametri e cookie

Alberto Ferrari

Http
Hyper Text Transfer Protocol

- Protocollo di livello applicazione per sistemi informativi distribuiti, collaborativi e ipertestuali
- Usato nel World-Wide Web fin dal 1990
  - Semplice & efficiente
  - Supporta molti tipi di dato
  - Senza connessione: una richiesta per ogni connessione
  - Senza stato: lo stato non è conservato tra connessioni diverse
- Descritto in RFC 2616 (HTTP/1.1)
  - Tipi di richiesta che un browser web può inviare
  - Tipi di risposta che un web server può fornire
  - Usa TCP/IP, ma si presume solo un livello di trasporto affidabile
  - Porta di default = 80

Alberto Ferrari

Http - Storia

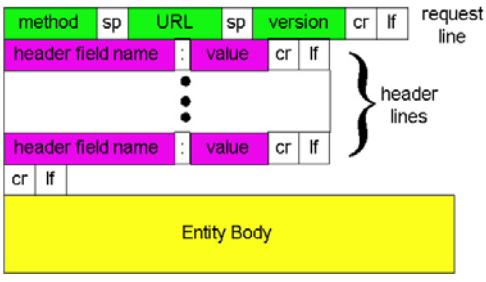
- Http/0.9 – Prima versione del protocollo
  - Semplice protocollo per il trasferimento di dati grezzi su Internet
- Http/1.0 – Miglioramenti, definito in RFC 1945
  - Messaggi in formato simile a quello Mime (e-mail)
  - *Meta-informazioni* sui dati trasferiti
  - Modificatori per il significato delle richeste/risposte
- Http/1.1 – Connessioni persistenti
  - Può usare la stessa connessione per inviare più oggetti in sequenza

Alberto Ferrari

Http - Client e server

- Http è un protocollo di tipo richiesta/risposta (request/response)
- Un client invia una **richiesta** al server, specificando:
  - Metodo, url, versione del protocollo
- Segue un messaggio di tipo mime contenente:
  - Modificatori di richiesta
  - Informazioni sul client
  - Eventuale corpo della richiesta (es. dati form)
- Il server invia in **risposta** una riga di stato, con:
  - Versione del protocollo, codice di successo o errore
- Segue un messaggio di tipo mime contenente:
  - Informazioni sul server
  - Meta-informazioni sulla entità (doc o altra risorsa richiesta)
  - Possibile corpo per contenuto dell'entità

Http
Formato generale richiesta



Http - Messaggi

- Ogni messaggio http consiste di:
  - Una riga di apertura (*start-line*)
  - Zero o più campi di intestazione (*header*)
  - Una *riga vuota* che indica la fine dell'header
  - Un possibile corpo del messaggio (*body*)
- Campi header in formato generico RFC 822 (email)
  - general-, request-, response-, entity-header
- Corpo (se presente) per trasportare il contenuto dell'entità
  - Risorsa associata alla richiesta o risposta

Alberto Ferrari

Http - Metodo get

- Il metodo *get* significa:

  Recupera qualsiasi informazione (in forma di entità) sia identificata dall'url della richiesta
- Se la url indica un processo di produzione dati, questi dati sono restituiti come corpo della risposta
- Semantica modificata in *get condizionale*:
  - *If-Modified-Since*, *If-Unmodified-Since*, ...
  - Per evitare spreco di risorse di rete

Alberto Ferrari

Http - Metodo post

- Il metodo *post* copre diverse **funzioni generali**
  - Invio di messaggi a bulletin board, newsgroup, mailing list, o gruppi di articoli simili
  - Aggiunta di dati ad un database
  - Invio di un blocco di dati, come quelli presenti all'interno di un form, ad un processo che li gestisca
- La url di una richiesta post identifica la risorsa che gestirà l'entità acclusa

Http
Connessioni usa-e-getta

1. Il browser apre una connessione al computer e alla porta specificati nella url desiderata
2. Il browser trasmette la parola "GET" seguita da uno spazio, un percorso per la risorsa, un ritorno a capo
   - Es. comando per chiedere la home page del server:
   - `GET /`
3. Il server risponde con una riga di stato, intestazioni varie, una riga vuota e il contenuto del file richiesto
4. Il server poi chiude la connessione (one-shot)

Per ricevere immagini, video ecc.
il client deve aprire altre connessioni:
una per ogni oggetto necessario

Http Esempio di richiesta

```
POST /beta.jsp HTTP/1.1
Referer: http://www.alpha.com/alpha.jsp (the address (URI) of
  the resource from which the Request-URI was obtained)
Connection: Keep-Alive
User-Agent: Mozilla/4.61
Host: www.alpha.com:80
Cookie: name=value
Accept: image/gif, image/jpeg, */*
Accept-Language: en
[blank line here]
selected-item=1234&action=show+details
```

```
GET /beta.jsp?selected-item=1234&action=show+details HTTP/1.1
Referer: http://www.alpha.com/alpha.jsp (the address (URI) of
  the resource from which the Request-URI was obtained)
[...]
If-modified-since: Mon, 10 Jul 2000 22:55:23 GMT
[blank line here]
```

Alberto Ferrari

Http - Esempio di risposta

```
HTTP/1.1 200 OK
Date: Fri, 12 Nov 2001 11:46:53 GMT
Server: Apache/1.3.3 (Unix)
Last-Modified: Mon, 12 Jul 2000 22:55:23 GMT
Accept-Ranges: bytes
Content-Length: 234
Content-Type: text/html
[blank line here]
<html>
  <head><title>Beta page</title></head>
  <body>
      <h1>Beta page</h1>...
```

Alberto Ferrari

Http - Codici di stato

- 1xx – Informational
  - Richiesta ricevuta, elaborazione in corso
- 2xx – Success
  - Richiesta ricevuta, interpretata e accettata
- 3xx – Redirection
  - Necessario intraprendere azioni ulteriori
  - *304* – Not modified
- 4xx – Client Error
  - Richiesta con sintassi scorretta o bloccata
  - *400* – Richiesta scorretta; *401* – Non autorizzato; *402* – Richiesto pagamento; *403* – Vietato; *404* – Non trovato
- 5xx - Server Error
  - Il server non riesce a soddisfare una richiesta apparentemente valida

Http - Cookie

I cookie sono un meccanismo generale
per memorizzare e recuperare informazioni
lato client di una connessione http

1. Quando un server invia una risposta, può inviare anche informazione di stato da memorizzare sul client
   - Specificato il **range di url** per cui lo stato è valido
   - Può essere specificata una **scadenza**
2. Per ogni futura richiesta fatta dal client in quel range, sarà ritrasmessa al server anche l'informazione di stato

Http - Cookie

- Si può inviare un cookie al client includendo un header *Set-Cookie* nella **risposta** http
  - `Set-Cookie: NAME=VALUE; expires=DATE; path=PATH; domain=DOMAIN_NAME; secure`
- Quando richiede una url, il browser la confronta con tutti i cookie (con il rispettivo range di url)
- Per ciascun cookie che corrisponde, alla **richiesta** http viene aggiunta una coppia chiave/valore
  - `Cookie: NAME1=VALUE1; NAME2=VALUE2 ...`

Alberto Ferrari

Http - Cookie

- L'aggiunta di informazione di stato, semplice e persistente, sul lato client estende in modo significativo le capacità di applicazioni web
- Semplice carrello per commercio elettronico: elenco prodotti selezionati memorizzato in cookie
- Spesso al client è inviato solo un *session-id*
  - Usato dal server come chiave per trovare, in un repository locale, l'informazione di stato per il client

Alberto Ferrari