REALIZZAZIONE DI UN WEB SERVER

Si tratta di realizzare un server che si comporti come previsto dal protocollo HTTP:

- **il client invia al server una richiesta sotto forma di una riga ASCII, nel formato**

  `GET` *`nomefile`* `HTTP/`*`versione`*

  Ad esempio, **`GET /index.htm HTTP/1.1`**
- **il server risponde inviando al client uno stream di caratteri ASCII, *secondo la sintassi HTML*; ad es:**

```
"<html><head></head><body>" +
"<h1> File non trovato </h1>" +
"</body></html>"
```

REALIZZAZIONE DI UN WEB SERVER

Alcune note:

- **Il protocollo HTTP è privo di stato: una volta inviato il file richiesto, il server chiude la connessione.**
- **Di norma, il server è in ascolto sulla porta 80**
- **È comunque possibile indicare al client (un qualunque browser) di *usare una diversa porta* mediante la notazione:**

  `http://`*`nomehost`*`:`*`porta`*`/`*`nomefile`

  (ad es., `http://localhost:888/index.html`)

REALIZZAZIONE DI UN WEB SERVER

Realizzazione del server in Java

- **Versione (volutamente) *minimale***
  - **single-threaded, non parametrica**
- **Si presta a essere estesa per divenire:**
  - **parametrica rispetto alla *porta di ascolto***
  - **parametrica rispetto alla *directory di base***
  - **multi-threaded**
  - **...**

  (i parametri potrebbero essere letti dalla riga di comando oppure da un *file di configurazione*)

REALIZZAZIONE DI UN WEB SERVER

Scelta base di progetto

- **usare stream di byte..** (non di caratteri)
  - **così si trattano anche immagini, file ZIP, etc**
- **..tranne che per la riga iniziale** (`GET ...`)

Conseguenze:

- **letture da `InputStream` fatte *byte per byte*** (il valore -1 indica la fine dello stream di input)
- **scritture di stringhe su `DataOutputStream` fatte con `writeBytes()`** (converte una String in sequenza di byte)

REALIZZAZIONE DI UN WEB SERVER

```
import java.net.*; import java.io.*;
public class ServerHTTP {
 public static void main(String args[]) {
  final String FILENOTFOUND =
    "<html> <body> <h1> File non trovato </h1>" +
    "</body> </html>" ;
  int port = 888;
  String baseHREF = "/Home/";
  String defaultFile = "index.html";
  switch(args.length) {
      case 2: baseHREF = args[1];
      case 1: port = Integer.parseInt(args[0]);
  }
  ...
```

REALIZZAZIONE DI UN WEB SERVER

```
...
try {
 ServerSocket socket= new ServerSocket(port);
 System.out.println("Porta " + port );
 String line = null, url = null;
 while (true) {
  Socket cs = socket.accept();
  System.out.println("Connessione da " + cs);
  DataOutputStream os =
   new DataOutputStream(cs.getOutputStream());
  InputStream is = cs.getInputStream();
  ...
```

REALIZZAZIONE DI UN WEB SERVER

reader sintetizzato "al volo", solo per leggere la prima riga

```
...
line = new BufferedReader(
  new InputStreamReader(is)).readLine();
// GET file HTTP/1.1
int spazio1 = line.indexOf(' ');
int spazio2 = line.indexOf(' ',spazio1+1));
url = line.substring(spazio1+2, spazio2);
// estrae nomefile, eliminando la barra '/'
if (url.equals("")) url = defaultFile;
System.out.println("Richiesto file " + url);
File fd = new File(BaseHREF + url);
if (!fd.exists()) os.writeBytes(FILENOTFOUND);
else {
 ...
```

REALIZZAZIONE DI UN WEB SERVER

```
  … // il file richiesto esiste certamente
  FileInputStream fin = new FileInputStream(fd);
  int x;
  while ((x=fin.read())!=-1) os.write(x);
  } // endif
  cs.close();
  } // end main loop
 } catch (Exception e){ System.err.println(e); }
 } // end main
}
```

REALIZZAZIONE DI UN WEB SERVER

Esempio d'uso:

```
java HTTPServer 88 /Home/Java/
```

Log su console:

```
Porta 88
Connessione da Socket [addr=...,port=1607,localport=88]
Ricevuta riga GET /Finestra.html HTTP/1.1
Richiesto file Finestra.html
Ricerca file /Home/Java/Finestra.html
Connessione da Socket [addr=...,port=1612,localport=88]
Ricevuta riga GET / HTTP/1.1
Richiesto file index.html
Ricerca file /Home/Java/index.html
...
```