

Progetto Reti di Calcolatori 2018

Bertini Gabriele - matricola 5793664
gabriele.bertini3@stud.unifi.it




8 settembre 2018

Implementazione

Sono stati implementati i livelli da 0 a 6:




● Livello 0

Sviluppo delle classi *MyHTTPRequest* e *MyHTTPReply* che implementano le interfacce *HTTPRequest* e *HTTPReply*. Si occupano di generare rispettivamente richieste e risposte che rispettano le regole di sintassi **HTTP**. È stata creata una classe astratta *MySetUp* per fattorizzare parti comuni di codice e rendere consistente l'implementazione di servizi simili fra le due classi.

- ▶  `MyHTTPReply.java`
- ▶  `MyHTTPRequest.java`
- ▶  `MySetUp.java`



● Livello 1

Sviluppo delle classi *MyHTTPInputStream* e *MyHTTPOutputStream* che estendono le classi astratte *HTTPInputStream* e *HTTPOutputStream*. Servono rispettivamente a leggere/scrivere richieste e risposte **HTTP**.

- ▼  `it.unifi.rc.httpserver.m5793664.streams`
 - ▶  `MyHTTPInputStream.java`
 - ▶  `MyHTTPOutputStream.java`





● Livelli 2 e 3

Sviluppo della classe *MyHTTPHandler1_0* che implementa l'interfaccia *HTTPHandler*, utilizzata per soddisfare le richieste **HTTP/1.0**; sono stati realizzati 2 costruttori, il primo per creare un handler privo di host, mentre il secondo per creare un handler provvisto di host. Questa versione soddisfa richieste di metodi relativi ad **HTTP/1.0**, cioè **GET**, **HEAD** e **POST**.

- ▼  `it.unifi.rc.httpserver.m5793664.handlers`
 - ▶  `MyHTTPHandler1_0.java`




• Livello 4 e 5

Sviluppo della classe *MyHTTPHandler1_1* che estende la classe *MyHTTPHandler1_0*, generando così un handler che accetta richieste sia **HTTP/1.0**, che **HTTP/1.1**. Questa versione soddisfa richieste di metodi relativi ad **HTTP/1.0** e **HTTP/1.1**, cioè **GET**, **HEAD**, **POST**, **PUT** e **DELETE**.

- ▼  `it.unifi.rc.httpserver.m5793664.handlers`
 - ▶  `MyHTTPHandler1_0.java`
 - ▶  `MyHTTPHandler1_1.java`
- ▼  `it.unifi.rc.httpserver.m5793664.http_protocol`
 - ▶  `CheckProtocolException.java`
 - ▶  `MyHTTPProtocolException.java`

• Livello 6

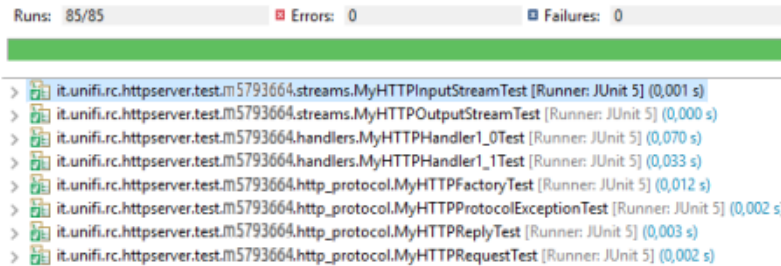
Sviluppo della classe *MyHTTPServer* che implementa l'interfaccia *HTTPServer*, utilizzata per creare un server; delegando in particolare il lavoro alla classe *SingleThreadServer*, la quale si occupa di servire le richieste in arrivo dal server.

- ▼  `it.unifi.rc.httpserver.m5793664.server`
 - ▶  `MyHTTPServer.java`
 - ▶  `SingleThreadServer.java`

Test JUnit 5

Ogni pacchetto contiene un sottopacchetto test. Le classi sono testate principalmente tramite l'utilizzo di una serie di richieste mirate a testare la resistenza e l'aderenza al protocollo.

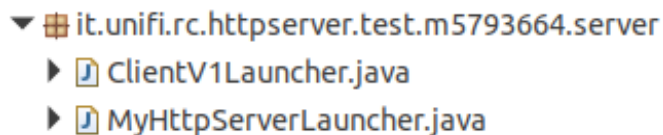
Sono stati realizzati 85 metodi di test unitari utilizzando il framework JUnit 5, al fine di garantire il funzionamento dei metodi chiave di ciascuna classe.



Per la classe *MyHTTPServer* è stato realizzato un test che funge da launcher, il test fallisce se riscontra problemi all'avvio del server.

Per testare il funzionamento del Server, dunque il livello 6 richiesto, è stato creato un semplice script Python, cioè il file *mainClient.py* che si trova in 5793664/Python/.

Per lanciare il server Java realizzato è necessario lanciare il test *MyHttpServerLauncher* presente nel pacchetto *it.unifi.rc.httpserver.test.m5793664.server*.



Questa classe test non fa altro che costruire un oggetto della classe *MyHTTPServer* ed invocare il suo metodo *start()*.

Dal lato dello script Python non fa altro che inoltrare la richiesta al server Java, contattandolo all'indirizzo dell'host locale e sulla stessa porta su cui esso ascolta.

Esempio di funzionamento

1. Esempio di GET lecita

```
GET /get_directory/root_file_html.html HTTP/1.0\r\n
Connection: Keep-Alive\r\n
User-Agent: myBrowser\r\n\r\n
```

```
Creating socket...
Connetting to server...
REQUEST
GET /get_directory/root_file_html.html HTTP/1.0
Connection: Keep-Alive
User-Agent: myBrowser

Sending...
Waiting reply...

From Server:

HTTP/1.0 200 OK
Date: Wed, 11 Jul 2018 21:06:56 +0200
Host: DESKTOP-COBDNU6
Last-Modified: ven, 16 feb 2018 16:41:12 CET

<!DOCTYPE html><html><body><h1>Heading</h1><p>A paragraph.</p></body></html>
```

2. Esempio di GET impossibile da soddisfare

```
GET /get_directory/null HTTP/1.0\r\n
Connection: Keep-Alive\r\n
User-Agent: myBrowser\r\n\r\n
```

```
Creating socket...
Connetting to server...
REQUEST
GET /get_directory/null HTTP/1.0
Connection: Keep-Alive
User-Agent: myBrowser

Sending...
Waiting reply...

From Server:

HTTP/1.0 404 Not Found

null
```