




Progetto Reti di Calcolatori 2018

Lorenzo Pratesi Mariti 5793319

lorenzo.pratesi@stud.unifi.it

Livelli implementati: 0-1-2-3-4-5-6

• Livello 0




- >  MyHTTPReply.java
- >  MyHTTPRequest.java
- >  MySetUp.java

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.

• Livello 1

- ▼  it.unifi.rc.httpserver.m5793319.streams
 - >  MyHTTPInputStream.java
 - >  MyHTTPOutputStream.java

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







• Livello 2-3

- ▼  it.unifi.rc.httpserver.m5793319.handlers
 - >  MyHTTPHandler1_0.java

Sviluppo della classe *MyHTTPHandler1_0* che implementa l'interfaccia *HTTPHandler*, utilizzata per soddisfare le richieste **HTTP/1.0**; si sono realizzati 2 costruttori, il primo per creare un handler privo di host, mentre il secondo provvisto.

Questa versione soddisfa richieste di metodi relativi ad **HTTP/1.0**, cioè **GET**, **HEAD** e **POST**.




• Livello 4-5

- ▼  it.unifi.rc.httpserver.m5793319.handlers
 - >  MyHTTPHandler1_0.java
 - >  MyHTTPHandler1_1.java
- ▼  it.unifi.rc.httpserver.m5793319.http_protocol
 - >  CheckProtocolException.java
 - >  MyHTTPProtocolException.java

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**.

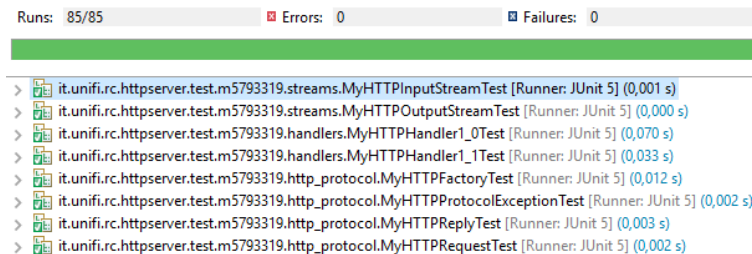
• Livello 6

- ▼  it.unifi.rc.httpserver.m5793319.server
 - >  MyHTTPServer.java
 - >  SingleThreadServer.java

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.

Test JUnit5

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 JUnit5, al fine di garantire il funzionamento dei metodi chiave di ciascuna classe.



Si noti come, data la natura non critica dell'applicazione, non è stata ricercata una test coverage del 100%.

Element	Coverage	Covered Instruction	Missed Instructions	Total Instructions
▼ HttpServer				
▼ src				
> it.unifi.rc.httpserver.m5793319.http_protocol	89,2 %	1.308	158	1.466
> it.unifi.rc.httpserver.m5793319.handlers	82,6 %	313	66	379
> it.unifi.rc.httpserver.m5793319.streams	92,4 %	636	52	688
> it.unifi.rc.httpserver.m5793319	92,6 %	312	25	337
> it.unifi.rc.httpserver	79,5 %	31	8	39
▼ tests				
> it.unifi.rc.httpserver.test.m5793319.http_protocol	69,6 %	16	7	23
> it.unifi.rc.httpserver.test.m5793319.streams	91,0 %	1.596	157	1.753
> it.unifi.rc.httpserver.test.m5793319.handlers	85,9 %	585	96	681
> it.unifi.rc.httpserver.test.m5793319	88,6 %	264	34	298
> it.unifi.rc.httpserver.test.m5793319	96,6 %	633	22	655
> it.unifi.rc.httpserver.test.m5793319	95,8 %	114	5	119

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. Si trova in *5793319/Python/mainClient.py*

Per lanciare il server Java realizzato è necessario lanciare il test *MyHttpServerLauncher* presente nel pacchetto *it.unifi.rc.httpserver.test.m5793319.server*

- ▼ it.unifi.rc.httpserver.test.m5793319.server
 - > ClientV1Launcher.java
 - > MyHttpServerLauncher.java

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

Lato script Python non fa altro che inoltrare la richiesta al server java, contattandolo all'indirizzo dell'host locale e sulla setta porta su cui esso ascolta

Funzionamento

Esempio di una *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>
```

Esempio di una *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
```