Relazione Progetto

Reti di Calcolatori

Anno Accademico 2017/2018

ServerHTTP

Daniel Zanchi - daniel.zanchi@stud.unifi.it - 5655418

Funzionalità implementate:

Livello 0:

Classi *MyHTTPRequest* e *MyHTTPReply* che implementano le rispettive interfacce *HTTPRequest* e *HTTPReply*.
Tramite queste due classi vengono generate le richieste e le risposte rispettando le regole generali della sintassi HTTP.

Livello 1:

Classi *MyHTTPInputStream* e *MyHTTPOutputStream* che estendono le rispettive classi astratte *HTTPInputStream* e *HTTPOutputStream.*
Servono rispettivamente a leggere e scrivere una richiesta e risposta HTTP nell'apposito stream.

Livello 2 e 3:

Classe *MyHTTPHandler1_0* che implementa l'interfaccia *HTTPHandler1_0* utilizzata per le richieste HTTP/1.0.
Sono stati realizzati due costruttori in cui uno non contiene *host* mentre l'altro ne è provvisto.
Questo handler è in grado di soddisfare i metodi richiesti da HTTP/1.0: *GET, HEAD, POST.*

Livello 4-5:

Classe *MyHTTPHandler1_1* che estende la classe *MyHTTPHandler1_0* dove viene generato un handler in grado di accettare richieste sia HTTP/1.0 sia HTTP/1.1.
Infatti questa versione soddisfa i seguenti metodi: *GET, HEAD, POST, PUT, DELETE.*

É stata creata una classe *MyHTTPProtocolException* che implementa *HTTPProtocolException* grazie alla quale vengono generati messaggi di eventuali errori del protocollo HTTP.

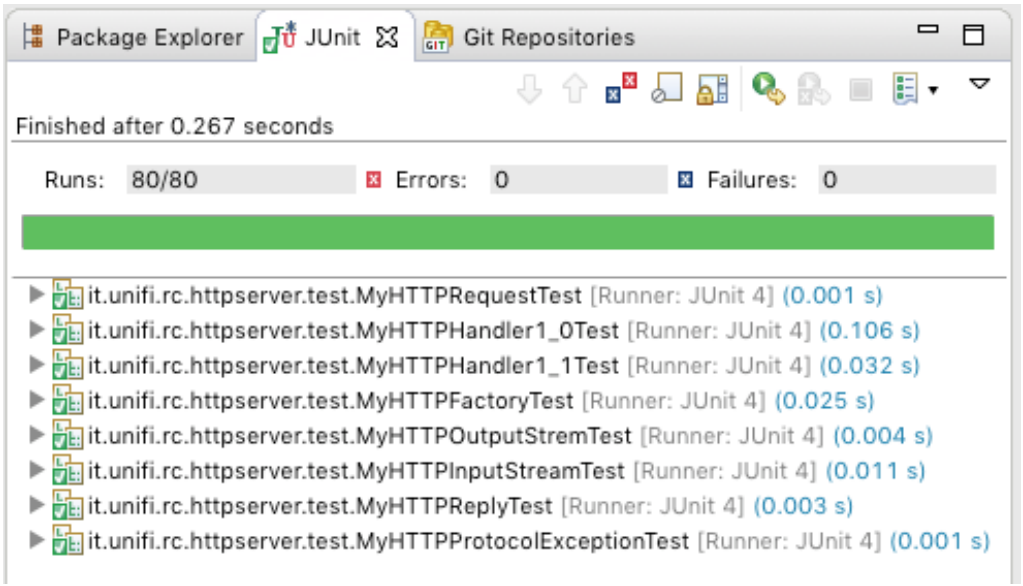
É presente una descrizione più dettagliata del codice sotto forma di *Javadoc.*

Il progetto non comprende i livelli sufficienti a garantire la funzionalità completa del server. Le classi sopra descritte sono state eseguite tramite i test.

Test:

La funzionalità del progetto realizzato è stata testata tramite JUnit4.
Sono stati testati i metodi confrontando i risultati prodotti con quelli aspettati.
Ogni classe è stata testata principalmente per l'uso primario e successivamente con l'utilizzo di test mirati a verificare la resistenza del protocollo garantendo anche l'esecuzione del codice in caso di eventuali errori.
É stato verificato ad esempio che le richieste prodotte rispettassero le sintassi HTTP anche nelle sue forme anomale.

I risultati dei test positivi assicurano il funzionamento delle classi implementate:



Per rendere il codice più pulito e facile da leggere è stata inclusa una classe *TestObj* che funziona da helper per creare le varie richieste e risposte nei diversi formati.

Tutti i test si trovano all'interno del pacchetto: *it.unifi.rc.httpserver.test*

La copertura del codice:

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
HTTPServer	91.7 %	3,124	281	3,405
tests	90.3 %	1,655	177	1,832
src	93.4 %	1,469	104	1,573
it.unifi.rc.httpserver.m5655418	93.7 %	1,453	97	1,550
MyHTTPHandler1_0.java	89.4 %	353	42	395
MyHTTPHandler1_1.java	95.1 %	372	19	391
MyHTTPInputStream.java	91.2 %	124	12	136
MyHTTPOutputStream.java	94.7 %	178	10	188
MyHTTPReply.java	96.5 %	195	7	202
MyHTTPRequest.java	95.6 %	152	7	159
MyHTTPFactory.java	100.0 %	39	0	39
MyHTTPProtocolException.java	100.0 %	40	0	40
it.unifi.rc.httpserver	69.6 %	16	7	23