FinanceBot

LaTeX

Area di progetto

Lingua Italiana

Classe 4J

Indice

1 Introduzione

1.1 Chi siamo

Siamo una classe di un istituto tecnico di indirizzo informatico. Insieme abbiamo realizzato un programma che monitora titoli NASDAQ attraverso API di Yahoo finance. I dati elaborati verrano poi processati da un bot di telegram.

2 Protocolli di rete

I protocolli di rete comprendono 5 protocolli, i più importanti sono i protocolli IP, ARP e ICMP. Il livello di trasporto permette a delle applicazioni che girano su terminali remoti di comunicare. Contiene due protocolli che permettono alle due applicazioni di scambiare dei dati indipendentemente dai livelli inferiori. Internet Protocol (IP), è un protocollo di rete, che si occupa diindirizzamento/instradamento, appartenente alla suite di protocolli Internet TCP/IP su cui è basato il funzionamento della rete Internet.I moderni protocolli per le reti informatiche generalmente usano la tecnologia del packetswitching per inviare e ricevere messaggi sotto forma di pacchetti, che sono messaggi suddivisi in piccoli pezzi che sono raccolti e riassemblati a destinazione.

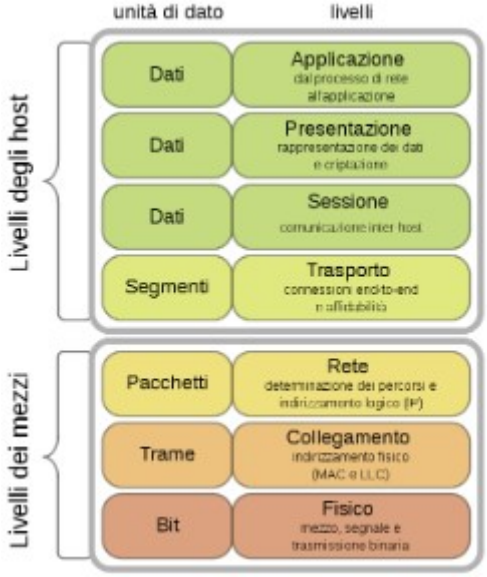

Figura 1: protocolli di rete

3 Modello ISO/OSI

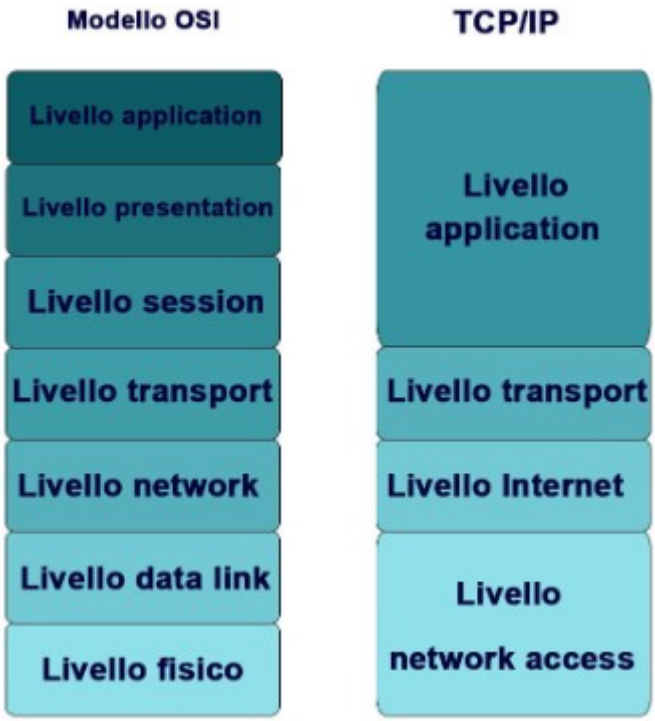

Figura 2: modello ISO/OSI

4 livello di trasporto

4.1 Protoccolo TCP

Il protocollo tcp (transmission control protocol) è un accordo standardizzato per la trasmissione dei dati tra diversi utenti di una rete informatica. Lo stato di sviluppo odierno del protocollo TCP permette a due punti terminali all'interno di una rete informatica comune di realizzare una connessione attraverso cui può avvenire una trasmissione bidirezionale dei dati. Nell'ambito di questa connessione, le eventuali perdite di dati vengono riconosciute e corrette automaticamente; per questo, il TCP viene denominato anche protocollo affidabile. Nella famiglia di protocolli Internet, il TCP costituisce, insieme all'UDP e all'SCTP, il gruppo dei protocolli di trasporto che, sulla base del modello OSI, vengono classificati nell'architettura di rete al livello di trasporto. Poiché, in quasi tutti i casi, il protocollo TCP si basa sul protocollo Internet (IP) e questa combinazione rappresenta la base per la maggior parte delle reti pubbliche e locali e per i servizi di rete, spesso si parla anche di pila di protocolli TCP/IP, intendendo in senso lato la famiglia di protocolli Internet.

4.1.1 Come funziona?

Il TCP permette la trasmissione delle informazioni in entrambe le direzioni. I sistemi informatici che comunicano tramite TCP possono pertanto inviare e ricevere dati contemporaneamente, proprio come avviene durante una conversazione telefonica. Le unità di trasmissione fondamentali utilizzate dal protocollo sono i segmenti (pacchetti) che, oltre al carico utile (ossia il messaggio effettivo), possono contenere anche informazioni di controllo e sono limitati a una dimensione di 1.500 byte. L'instaurazione e l'interruzione delle connessioni, classificabili come connessioni punto-a-punto, nonché la trasmissione stessa dei dati vengono acquisiste dal software TCP nella pila di protocolli di rete del rispettivo sistema operativo.

TCP Header				
Bits	0-15			16-31
0	Source port			Destination port
32	Sequence number			
64	Acknowledgment number			
96	Offset	Reserved	Flags	Window size
128	Checksum			Urgent pointer
160	Options			

Figura 3: TCP

4.2 Protocollo UDP

4.2.1 Applicazioni che utilizzano UDP

Le applicazioni di rete che hanno la necessità di un trasferimento affidabile dei loro dati non si affidano ovviamente a UDP, mentre le applicazioni più elastiche riguardo alla perdita dei dati e strettamente dipendenti dal tempo si affidano invece a UDP. Inoltre si utilizza UDP per comunicazioni in broadcast (invio a tutti i terminali in una rete locale) e multicast (invio a tutti i terminali iscritti ad un servizio).

Applicazione	protocollo stato applicazione	protocollo strato trasporto
posta elettronica	SMTP	TCP
accesso a terminale remoto	telnet	TCP
trasferimento file	FTP	TCP
Web	HTTP	TCP
Streaming audio/video	RTSP/RTP	TCP (comandi) + UDP (flusso)
Server di file remoto	NFS	tipicamente UDP
Telefonia su internet (VoIP)	SIP, H.323, altri	tipicamente UDP
Gestione della rete	SNMP	tipicamente UDP
Protocollo di routing	RIP	tipicamente UDP
Risoluzione dei nomi	DNS	tipicamente UDP

Tabella 1: elenco dei principali servizi Internet e dei protocolli che essi adottano

5 Applicazione Client-Server

5.1 Che cos'è

Un'applicazione client/server è un'applicazione distribuita divisa in due parti: un'applicazione server (back-end) che offre dei servizi e una applicazione client (front-end) che gestisce l'interfaccia con l'utente e permette di richiedere servizi al server.

Le componenti client e server dell'applicazione in genere si trovano su due stazioni diverse (la parte server di solito si trova su un computer molto più potente) ma possono trovarsi anche nella stessa stazione.

L'architettura client/server può essere realizzata anche a tre livelli, con un livello di gestione dell'interfaccia utente, uno di elaborazione e uno di gestione delle informazioni (normalmente memorizzate in un database).

È un modello non simmetrico, chiamato anche request/replay perché si basa sul susseguirsi di richieste e risposte.

Il server può essere progettato in modo da soddisfare una sola richiesta per volta o più richieste contemporaneamente.

Il client per contattare il server ne deve conoscere l'indirizzo; l'indirizzo può dover essere noto a priori, può esserci un server di processo a cui il client fa le richieste che poi vengono inoltrate al server desiderato, o può esserci un server di directory che in base al nome del server desiderato ne restituisce l'indirizzo.

La tecnica più comune per la realizzazione di applicazioni client/server è l'utilizzo dei socket TCP/IP.

6 DNS

DNS è l'acronimo di Domain Name System e, per spiegarlo in breve, è il sistema che traduce gli indirizzi IPin nomi di dominio.Quando navighiamo sul web, nella barra degli indirizzi del browser scriviamo il dominio di un sito, per esempio sos-wp.it.

Ciò che accade a livello tecnico è questo:

- il browser prende il dominio da te inserito;
- questo dominio è associato ad un indirizzo IP;
- l'indirizzo IP è una serie di numeri che identifica un determinato server, ossia un computer all'interno del quale si trova ilsito web;
- il DNS traduce il nome del dominio inindirizzo IP, e viceversa;

Figura 4: DNS

6.1 A cosa serve il DNS?

I dns servono a effettuare una traduzione dal nome (il dominio) al numero (indirizzo ip), come per esempio la rubrica di un telefono. Quindi invece che scrivere tutto l'indirizzo ip basta digitare il nome della persona che vuoi chiamare.Un altro esempio può essere un sito web, invece che scrivere tutto l'indirizzo ip basta digitare il nome del sito, come può essere Amazon.com.

7 HTTP

HTTP (Hypertext Transfer Protocol, letteralmente protocollo di trasferimento di un ipertesto) è un linguaggio di testo che consente la comunicazione tra client e server attraverso internet. Dunque ogni qualvolta visitiamo un sito internet, HTTP ci consente di visualizzare le risorse di quel sito.

7.1 Come funziona HTTP request e response :

Dopo aver introdotto brevemente il protocollo HTTP adesso vediamo il suo funzionamento passo passo, analizzando come è fatta una richiesta HTTP e la successiva risposta da parte del server. Request il client invia una richiesta :Ogni conversazione tra client e server sul web inizia con una richiesta, un messaggio di testo creato dal client in un formato speciale noto appunto come HTTP.

Una richiesta HTTP è composta dalle seguenti parti:
[method] [URI] [version]
[headers]
[body]
La prima riga di una richiesta HTTP è la più importante e contiene due parti fondamentali:

1. il metodo HTTP (GET)
2. l'URI (/blog/protocollo-http-cosa-e-come-funziona)

Metodo	Descrizione
GET	Recupera una risorsa dal server (ad es. visitando una pagina)
POST	Invia una risorsa al server (ad. es compilando un modulo)
DELETE	Cancella una risorsa dal server (ad es. eliminando un file)
PUT	Memorizza una risorsa sul server (as es. caricando un file)
HEAD	Recupera solo l'header della risposta senza la risorsa

Figura 5: HTTP

I metodi HTTP definiscono i modi in cui il client può interagire con la risorsa. La tabella seguente mostra alcuni dei metodi utilizzati durante una richiesta HTTP.

8 CSV

Un file CSV(Comma Separated Values) è tra i formati più diffusi per immagazzinare dati tabellari e vengono usati principalmente per gestire una grande mole di dati. Gli elementi del csv sono comunemente separati da una virgola. Solitamente nella prima riga vengono rappresentate le colonne della nostra tabella, mentre nelle righe successive vengono rappresentati i valori.

8.1 Come leggere un file CSV?

La pratica più comune per interagire con un file csv in python è quella di utilizzare la libreria omonima. Partiamo subito importandola nel nostro script: import csv. Per leggere il contenuto del nostro file possiamo utilizzare il metodo csv.reader().

Nel caso in cui il nostro file.csv usi come delimitatore un carattere diverso dalla virgola, dobbiamo specificarlo passando un secondo parametro al metodo csv.reader(). Questo parametro si chiama delimiter e di default è valorizzato con la virgola. Inoltre, chiamando il metodo csv.DictReader() al posto del csv.reader(), otteniamo il contenuto del csv in un dizionario.

8.2 Come scrivere un file CSV?

- Per scrivere un file CSV in Python è possibile usare il metodo write_row() dell'oggetto writer, che viene recuperato chiamando il metodo csv.writer().
- Per vedere il risultato bisogna riprendere lo script scritto in precedenza per leggere il file, cambiare il nome del csv con quello che abbiamo generato ed eseguirlo.
- Come per la lettura, anche nella scrittura si può definire il carattere delimitatore nel metodo csv.writer(file, delimiter=";").
- Usando il metodo csv.DictWriter() si può creare un csv da un dizionario in questo modo.
- Tenere a mente che in questo caso è necessario definire le colonne che si vogliono utilizzare.

9 API

```
# utilizzo del API
querystring = {"region":"US","symbols":titolo[i]}
response = requests.request("GET", url, headers=headers, params=querystring)
status_code = response.status_code


    correctJson = removeJson(response.text)
    writeToFile(path, correctJson)

    with open(path, "r") as json_file:
```

Figura 6: API

API è l'abbreviazione di interfaccia di programmazione delle applicazioni (application programming interface), ovvero un insieme di definizioni e protocolli per la creazione e l'integrazione di software applicativi. Intuitivamente sono quindi alla base di una vastità di applicativi, servizi, siti web, dispositivi. Le API sono nate con la nascita stessa dell'informatica, ben prima dei personal computer.

9.1 A cosa servono le API e come funzionano?

L'Application Programming Interface viene solitamente fornita dagli sviluppatori di un software in modo che i programmatori di altre applicazioni possano utilizzarla. L'interfaccia di programmazione determina come le informazioni e i dati vengono ricevuti e restituiti tra i moduli.

10 JSON

10.1 Cos'è JSON

JSON è un formato di serializzazione basato su testo per lo scambio di dati, principalmente tra server e applicazione web. JSON è l'acronimo di JavaScript Object Notation, e utilizza l'estensione di file .json. JSON è un concorrente di XML, ma possiede una sintassi più semplice e compatta rispetto al rivale.

10.2 Come funziona JSON

1. I dati sono scritti come coppia chiave-valore.
2. I dati sono separati da una virgola.
3. Le parentesi graffe contengono gli oggetti.
4. Le parentesi quadre contengono gli array.
5. Le stringhe vanno racchiuse con i doppi apici.

11 Linguaggio di programmazione Java

Java è un linguaggio di programmazione ad alto livello, basato su classi e metodi e orientato agli oggetti. I dati sono rappresentati come oggetti e le operazioni come metodi che operano su essi. Questo linguaggio di programmazione è stato pensato per lo sviluppo di applicazioni in reti e per essere il più possibile indipendente dalla piattaforma Hardware di esecuzione.

11.1 Vanataggi e svantaggi

Vantaggi:

- compatibilità multipiattaforma (grazie alla Virtual Machine)
- alta astrazione dalla macchina fisica
- velocità di sviluppo
- grande disponibilità di librerie
- compatibilità con sistemi palmari ed embedded
- alta integrazione con il web

Svantaggi:

- lentezza (grazie alla Virtual Machine)
- alta astrazione dalla macchina fisica (che non permette di fare tutto ciò che si vuole)
- è decompilabile.

12 Linguaggio di programmazione Python

Python è un linguaggio di programmazione di più alto livello rispetto alla maggior parte degli altri linguaggi, anche esso basato su oggetti. Tra gli usi comuni, Python è adatto a sviluppare applicazioni distribuite, scripting, computazione numerica, system testing, siti web, software ecc. Inoltre sono disponibili molte librerie Python in campo educativo e scientifico.

12.1 Vantaggi e svantaggi

Uno degli svantaggi più grandi è di sicuro la mancanza di tipi, quindi una mancanza di controllo statico. Inoltre i compilatore di Python non è capace di trovare molti errori e la mancanza d controllo dei valori ritornati dalle funzioni. Infatti, siamo obbligati a cercare tutti i casi, e spesso i codici che gestiscono gli errori sono difficili da raggiungere. Uno dei vantaggi più grandi è la facilità e la pulizia della sua sintassi che rende da subito molto chiaro il codice, inoltre grazie al supporto della Python Software Foundation si avvale di una libreria built-in molto ampia che lo rende uno dei linguaggi di programmazione più ricchi e pratici. Python si può definire un linguaggio pseudo complicato e portabile infatti un interprete può analizzare il codice sorgente per poi eseguirlo nel caso lo ritenga corretto dato che non è prevista una fase di compilazione separata da cui possa essere generato un file eseguibile.

13 LaTeX

13.1 Che cos'è LaTeX?

Il LaTeXè un linguaggio di programmazione sviluppato nel 1985 da Leslie Lamport. La cui ultima versione risale al 2011. Il LaTeXè un linguaggio a marcatori, in questo caso il marcatore più utilizzato è il (backslash). Il LaTeXè usato principalmente per produrre documenti in pdf. Questo linguaggio per produrre i file in pdf deve essere compilato.

13.2 Come abbiamo utilizzato LaTeX

Per realizzare il nostro documento abbiamo utilizzato un editor e creato il documento da zero. Successivamente abbiamo aggiunto un indice e dei capitoli e sottocapitoli per rendere il pdf più ordinato. Abbiamo aggiunto degli optional come l'indice e per distribuire bene il tutto abbiamo scelto di utilizzare dei capitoli e dei sottocapitoli. Per lo sviluppo ulteriore del codice abbiamo utilizzato alcuni pacchetti come quelli per i caratteri speciali, che comprende la possibilità di far riconoscere al sistema i caratteri speciali, come le lettere accentate.

13.3 Comandi più comuni

```
\section{}%per creare un titolo
\subsection{}%per creare un sottotitolo
\begin{itemize}
    \item
    \item   %crea un elenco puntato
    \item
\end{itemize}
\begin{thebibliography}% per creare una bibliografia.
\cite{nome del riferimento}
```

Figura 7: comandi base