27/02/2025

**RELAZIONE DI SISTEMI E RETI**

**AUTORI: STUDENTE DELLA CLASSE – 4IB**

ANTONIO FRISELLE

Relazione di Sistemi e Reti CLASSE 4IB

|  |
| --- |
| **Obiettivo dell’esperienza** |
| Realizzare una applicazione Client-Server in linguaggio Java, che consenta agli utenti di consultare da remoto le informazioni di interesse relative alle Piscine presenti in un file CSV.  Il file CSV è presente nella cartella() di GitHub. |

Assegnata il giovedì 20 febbraio 2025

Data di consegna: venerdì 28 febbraio 2025

Studente: Antonio Friselle

**DESCRIZIONE DEL PROGETTO:**

**Il server:** legge ed interpreta un file CSV(contiene le informazioni delle piscine) e organizza i dati in una struttura interna di java: ArrayList

Offre un’interfaccia di comunicazione remota tramite socket per ricevere richieste ed inviare risposte al client

Gestisce eventuali errori, come richieste non valide o malformate

Il progetto utilizza il **protocollo TCP (Transmission Control Protocol)** per gestire la comunicazione tra client e server.  
TCP è un protocollo affidabile e orientato alla connessione, che garantisce che i dati trasmessi arrivino completi e nell’ordine corretto.  
Grazie a questo protocollo:

* Il client può inviare comandi al server in modo sicuro.
* Il server può rispondere garantendo l’integrità delle informazioni inviate.
* Viene gestita automaticamente la ritrasmissione in caso di pacchetti persi o corrotti.

**Il client:** Permette all’utente di inviare richieste specifiche al server, e i dati ricevuti dal server vengono mostrati in modo chiaro e leggibile su un’interfaccia grafica.

**PROTOCOLLO DI COMUNICAZIONE TRA CLEINT E SERVER:**

|  |  |
| --- | --- |
| **COMANDI:** | **DESCRIZIONE:** |
| **Seleziona-Comune** | Prende tutte le informazioni delle piscine che si trovano in quel determinato Comune. |
| **Seleziona-Provincia** | Prende tutte le informazioni delle piscine che si trovano in quella determinata provincia. |
| **Seleziona-Regione** | Prende tutte le informazioni delle piscine che si trovano in quella determinata regione. |
| **Seleziona-Nome** | Prende tutte le informazioni delle piscine che hanno determinato nome. |
| **Seleziona-Anno** | Prende tutte le informazioni delle piscine che sono state fondate in quel determinato anno. |
| **Seleziona-AnnoMaggiore** | Prende tutte le informazioni delle piscine che sono state fondate prima di quel determinato anno. |
| **Seleziona-AnnoMinore** | Prende tutte le informazioni delle piscine che sono state fondate dopo quel determinato anno. |
| **Seleziona-Data** | Prende tutte le informazioni delle piscine che sono state fondate in quel determinata data dove viene specificato l’anno, e l’ora di inserimento. |
| **Seleziona-Map** | Prende tutte le informazioni delle piscine che sono situate in quella mappa. |
| **Seleziona-Longitudine** | Prende tutte le informazioni delle piscine che hanno quella long. |
| **Seleziona-Latitudine** | Prende tutte le informazioni delle piscine che hanno quella latitudine. |
| **Numero-Tot** | il server risponde con il numero di piscine totali . |
| **EXIT** | Chiude la comunicazione con il server. Il comando può essere utilizzato da qualsiasi parte. |

**Il server risponderà a questi comandi chiedendo l’inserimento del dato**: Comune, Provincia, Nome, Anno, Data, Map, Longitudine, Latitudine. Mentre i comandi come Numero-Tot e EXIT non hanno bisogno di inserimento di parametri.

**SEMPLICE SPIEGAZIONE CODICE:**

**Client (GUI - PiscineClientGUI)**

**Scopo:**Permette all’utente di inviare comandi al server e ricevere risposte.

**Componenti principali:**

* **JComboBox + JTextField**: per scegliere il comando e inserire un eventuale parametro.
* **JTextArea**: per visualizzare i risultati della query.
* **Socket**, BufferedReader, PrintWriter: per la comunicazione con il server.

**Funzionamento:**

1. Alla creazione (connettiAlServer()), il client si connette al server sulla porta **1050**.
2. Viene letto e mostrato il messaggio di benvenuto inviato dal server.
3. Quando si clicca su **Invia**:
   * Il comando selezionato e il parametro (se richiesto) vengono inviati al server.
   * Il client legge tutte le righe fino a FINE\_RISPOSTA e le mostra a schermo.
4. Cliccando su **Esci**, si chiude la connessione.

**Classe Piscina**

Rappresenta una piscina, mappando ogni riga del CSV. Campi principali:

* Localizzazione: comune, provincia, regione
* Informazioni identificative: nome, idOSM
* Coordinate geografiche: latitudine, longitudine
* Informazioni temporali: annoInserimento, dataOraInserimento

Contiene metodi toString() e stampaCompatta() per output in formato leggibile o sintetico.

**Classe CSVLettore**

**Scopo:**

Caricare i dati dal CSV Mappa-delle-piscine-in-Italia.csv in memoria come lista di oggetti Piscina.

**Funzionamento:**

* Salta la riga di intestazione.
* Legge ogni riga, effettua il split(";").
* Crea un oggetto Piscina per ogni riga, gestendo eventuali valori vuoti per l’anno.

**Server (Main + GestioneServer)**

**Funzionamento generale:**

1. Avvia un ServerSocket sulla porta 1050.
2. In ascolto continuo, accetta client e gestisce ogni connessione su un nuovo thread.
3. All’inizio della connessione, invia messaggi introduttivi e l'elenco comandi.
4. Attende un comando, poi un parametro (se richiesto).
5. Utilizza GestioneServer per elaborare la richiesta e restituisce i risultati al client.

**Classe GestioneServer**

Contiene tutta la logica per interpretare i comandi e filtrare la lista di piscine.

**Comandi gestiti:**

* SELEZIONA-COMUNE, SELEZIONA-PROVINCIA, SELEZIONA-REGIONE, SELEZIONA-NOME: filtrano per corrispondenza esatta.
* SELEZIONA-ANNO, SELEZIONA-ANNOMAGGIORE, SELEZIONA-ANNOMINORE: filtrano sull’anno.
* SELEZIONA-DATA: ricerca nella stringa della data.
* SELEZIONA-MAP: filtra per ID OSM.
* SELEZIONA-LONGITUDINE, SELEZIONA-LATITUDINE: confrontano coordinate.
* NUMERO-TOT: restituisce il numero totale di piscine.
* EXIT: comando di chiusura (gestito anche dal client).

Utilizza Predicate<Piscina> in metodi generici per evitare ripetizione di codice.

**Ciclo di comunicazione Client-Server**

1. Client si connette e riceve messaggio iniziale.
2. L’utente seleziona un comando e inserisce un parametro (se necessario).
3. Il client invia il comando e il parametro al server.
4. Il server elabora il comando e restituisce le risposte una riga alla volta.
5. Il client legge fino a FINE\_RISPOSTA e visualizza.