MANUALE TECNICO

DE PAOLI LORENZO – ONESTI ANDREA – RIZZO MATTIA – WU WEILI

Progetto di laboratorio A: Climate Monitoring

Indice

1.0 Introduzione

Climate Monitoring è un progetto sviluppato nell'ambito del Laboratorio A, parte integrante del corso di laurea in Informatica presso l'Università degli Studi dell'Insubria.

Il suo obiettivo principale è fornire una solida piattaforma di monitoraggio dei parametri climatici di una città selezionata. I dati, provenienti da centri di monitoraggio, vengono resi accessibili per fornire informazioni dettagliate sulla zona di interesse.

2.0 Librerie esterne

Le librerie esterne sono delle librerie che non sono incluse nella libreria standard di Java, ma sviluppata da terze parti. Queste librerie forniscono funzionalità aggiuntive e specifiche che possono essere utilizzati nei progetti.

A seguito verranno elencate tutte le librerie che sono state importare, e a seguito andremo ad analizzarle più nel dettaglio.

1. import java.io.BufferedReader;
2. import java.io.BufferedWriter;
3. import java.io.FileReader;
4. import java.io.FileWriter;
5. import java.text.SimpleDateFormat;
6. import java.util.ArrayList;
7. import java.util.List;
8. import java.util.Date;
9. import prog.io.ConsoleInputManager;

2.1 java.io.BufferedReader

La classe “java.io.BufferedReader” fa parte del package “java.io” in Java ed è utilizzata per leggere il testo da un input stream in modo efficiente, fornendo metodi per leggere i dati riga per riga.

2.2 java.io.BufferedWriter

La classe “java.io.BufferedWriter” fa parte del package “java.io” in Java ed è utilizzata per scrivere testo in un output stream in modo efficiente, fornendo metodi per scrivere dati, specialmente stringhe, in modo bufferizzato.

2.3 java.io.FileReader

La classe “java.io.FileReader” fa parte del package “java.io” in Java ed è utilizzata per leggere i dati da un file di testo. La classe estende “InputStreamReader” e fornisce metodi specifici per la lettura dei caratteri da un file.

2.4 java.io.FileWriter

La classe “java.io.FileWriter” fa parte del package “java.io” in Java ed è utilizzata per scrivere dati in un file di testo. Questa classe estende “OutputStreamWriter” e fornisce metodi specifici per la scrittura di caratteri in un file.

2.5 java.text.SimpleDateFormat

La classe “java.text.SimpleDateFormat” fa parte del package “java.text” in Java ed è utilizzata per formattare e analizzare le date secondo determinati pattern. Questa classe è utile quando è necessario convertire tra oggetti Date e rappresentazioni di stringhe data in un formato specifico.

2.6 java.util.ArrayList

La classe “java.util.ArrayList” fa parte del framework di collezioni in Java ed è implementazione della struttura dati dinamica di tipo lista, basata su un array dinamico. L'ArrayList fornisce un modo flessibile per gestire collezioni di oggetti, consentendo l'inserimento, la rimozione e l'accesso agli elementi in modo efficiente.

2.7 java.util.List

“java.util.List” è un'interfaccia nell'API di collezioni di Java, appartenente al package “java.util”. Essa rappresenta una collezione ordinata di elementi, in cui è possibile accedere agli elementi attraverso un indice. List è parte dell'interfaccia di base delle liste in Java ed è implementata da diverse classi, tra cui ArrayList, LinkedList, Vector, e altre.

2.8 java.util.Date

La classe “java.util.Date” in Java rappresenta un punto specifico nel tempo, con una precisione fino al millisecondo.

2.9 Prog.io.ConsoleInputManager

L’importazione di questa libreria permette alle sue istanze di realizzare un canale di comunicazione con il dispositivo di input standard che in questo caso si tratta della tastiera.

3.0 Le classi e i vari metodi

L'applicazione è strutturata in base al concetto di programmazione orientata agli oggetti (OOP), che si basa sull'utilizzo di classi e oggetti. Ogni classe è progettata per gestire specifiche operazioni che l'utente può eseguire. All'interno di ciascuna classe, sono implementati metodi che si occupano di gestire e controllare gli input forniti dall'utente durante l'utilizzo dell'applicazione.

Le classi principali che sono state sviluppate per creare e gestire l’intero progetto sono le seguenti:

1. AreaInteresse
2. CentroMonitoraggio
3. ClimateMonitor

4. Comune
5. Indirizzo
6. Operatore
7. ParametroClimatico

Ora andremo a vederle più nel dettaglio, analizzando a seguito i metodi principali presenti nella classe stessa.

3.1 La classe AreaInteresse

La classe “AreaInteresse” è stata progettata per fungere da contenitore dedicato a informazioni e operazioni inerenti alle particolari aree di interesse nel contesto dell'applicazione. Questa struttura è stata concepita con l'obiettivo di gestire in modo efficace e organizzato i dati rilevanti.

La classe contiene informazioni, tra cui il nome dell'area, lo stato in cui si trova e le coordinate geografiche (longitudine e latitudine). Inoltre, include i dati relativi ai parametri climatici.

Metodo toString ()

Il metodo toString() è un metodo predefinito in Java che appartiene alla classe Object. È utilizzato per ottenere una rappresentazione in forma di stringa di un oggetto. Questo metodo è spesso sovrascritto nelle classi personalizzate per fornire una rappresentazione significativa e leggibile degli oggetti. Quando questo metodo verrà chiamato, verrà restituita le informazioni inerenti all’area di interesse.

3.2 La classe CentroMonitoraggio

Questa classe è responsabile al coordinamento ed all’elaborazione delle informazioni relative al clima delle aree monitorare.

La classe contiene informazioni riguardanti il luogo in cui è situato il centro di monitoraggio e tutte le aree che sono state precedentemente monitorate.

3.3 La classe ClimateMonitor

La classe “ClimateMonitor” rappresenta la classe main, con la quale l’utente interagisce.

Metodo main (String args[])

Questo metodo è il punto di ingresso principale dell’applicazione, che, quando viene lanciato, restituisce al mittente una lista di opzioni tra cui scegliere, che può essere la ricerca di una area geografica, il login, la registrazione ed infine la decisione di lasciare l’interazione con il terminale.

3.4 La classe Comune

La classe “Comune” ha come funzione principale quella della ricerca dell’area di interesse e contiene i metodi eseguibili dagli utenti comuni, cioè gli non registrati.

Metodo CercaAreaGeograficaLuogo (String citta, String stato)

Utilizzato nella ricerca dell'area geografica attraverso il riconoscimento della città inserisca dall'utente. Nel caso in cui la città inserita sia uguale a uno di quelli presenti nella lista ritorna il nome della area scelta altrimenti nulla.

Metodo CercaAreaGeograficaCootdinate(String lat, String lon)

Ha la stessa funzione del metodo precedente, con la differenza del metodo di riconoscimento dell'area, in questo caso si utilizza la latitudine e la longitudine.

Metodo VisualizzaAreaGeografica (AreaInteresse a)

Nel caso in cui durante la ricerca dell'area di interesse sia stata effettuata correttamente, attraverso l'utilizzo di questo metodo si possono osservare i diversi parametri climatici rilevati.

Metodo CreaListaParametri (AreaInteresse a)

Metodo utilizzato per prendere tutti i parametri all'interno di un file .csv e li mette in un array di stringhe.

3.5 La classe Operatore

La classe "Operatore" si estende dalla classe "Comune", conseguentemente ereditando tutti gli attributi e i metodi della classe genitore. La progettazione di questa classe è focalizzata sulla raccolta di tutti i metodi eseguibili dagli utenti registrati, ossia gli operatori.

Inoltre, la classe "Operatore" riveste un ruolo di grande importanza, poiché è stata concepita per operare in parallelo alla classe principale. Quest'ultima funge da interfaccia e ha il compito di gestire ogni aspetto del programma. La classe "Operatore" si integra sinergicamente con la classe principale, assumendo la responsabilità di eseguire ogni punto specifico, contribuendo così al corretto funzionamento complessivo del sistema.

Metodi Crea ()

Ci sono vari metodi all'interno del programma che hanno la responsabilità di creare liste contenenti informazioni. Questi metodi sono progettati per estrarre dati dai file .csv e inserirli nelle liste appropriate. Questa operazione permette al programma di gestire in modo efficace le informazioni provenienti da tali file.

L'implementazione di tali metodi include la lettura dei file .csv, l'interpretazione dei dati contenuti e la successiva inserzione di queste informazioni nelle strutture dati di liste. Questo approccio consente al programma di manipolare agevolmente le informazioni e di utilizzarle per il raggiungimento degli obiettivi prefissati.

Questi metodi di creazione delle liste sono fondamentali per garantire che il programma abbia accesso e possa operare su un insieme completo e accurato di dati provenienti dai file .csv associati.

Metodi Controllo()

Questi metodi di tipo booleano rappresentano il fulcro della sicurezza all'interno del programma. La loro funzione principale consiste nel verificare se i dati inseriti dagli utenti corrispondono alle informazioni precedentemente registrate. L'implementazione di tali metodi svolge un ruolo cruciale nella protezione del sistema, garantendo che le interazioni degli utenti rispettino le regole e i vincoli definiti.

Metodi Registrazione()

In caso di assenza di determinati tipi di informazioni, questi metodi assumono il compito di registrare sui file .csv i dati mancanti. La loro implementazione include la capacità di rilevare lacune o informazioni mancanti all'interno del sistema. Una volta individuati tali vuoti, i metodi si attivano per registrare in modo adeguato le nuove informazioni nei file di destinazione, garantendo che il sistema mantenga una completezza e coerenza nei dati.

Metodo login()

L'implementazione di questo metodo può coinvolgere la verifica delle credenziali dell'operatore rispetto ai dati registrati nel sistema. Se le credenziali sono valide, l'operatore ottiene l'accesso alle funzionalità del programma. In caso contrario, viene generato un messaggio di errore e l'operatore viene indirizzato nuovamente alla schermata principale per ulteriori tentativi. Questo approccio migliora la sicurezza del sistema, garantendo che solo operatori autorizzati possano accedere e svolgere le attività previste.

Metodo InserisciParametriClimatici(String c)

L'implementazione di questo metodo coinvolge la raccolta dei parametri climatici forniti dall'operatore, compresi i valori e le eventuali note aggiuntive. Una volta acquisite queste informazioni, il sistema procede con l'inserimento dei dati nei file .csv corrispondenti, garantendo una registrazione accurata delle condizioni climatiche specifiche dell'area di interesse.

4.0 I data

Durante l'illustrazione delle classi si è già citato dei file .csv, ma cosa sono di preciso.

I file CSV (Comma-Separated Values) sono un formato testuale utilizzato per rappresentare dati tabulari, dove le colonne sono separate da un delimitatore come la virgola. Questi file sono comunemente impiegati per esportare e importare dati tra applicazioni diverse. Ogni riga in un file CSV rappresenta un record. Le librerie CSV in Java semplificano le operazioni relative ai dati tabulari, fornendo strumenti potenti per la lettura e la scrittura di questi file. Queste librerie agevolano l'implementazione di funzionalità di importazione ed esportazione di dati nelle applicazioni Java.

Sono state utilizzate quattro file, che sono le seguenti:

- CentroMonitoraggio: contiene i dati che fanno riferimento ai diversi centri di monitoraggio e le sue aree interessate;

- CoordinateMonitoraggio: contiene i dati di lista di luoghi, il suo stato di appartenenza e i corrispondenti coordinate geografiche;
- OperatoriRegistrati: contiene i dati inerenti a tutti gli operatori registrati, tra cui l'username, la password, l'e-mail, ecc.;
- ParametriClimatici: contiene i dati riguardanti i parametri climatici di ogni area di interesse registrato con aggiunta di nota, data e ora della registrazione.