Diana Gafiuc & Stefano Pirarba
4C info
23-04-2020

Open Data

Consegna

1. A partire dai siti di ARPA Piemonte o Torino Respira cercare degli “open data” riguardo a qualità dell’aria o su argomenti che riguardano i cambiamenti climatici o l’ambiente in generale. Cercare i dati in un formato che sia csv oppure json o xml. Comprendere bene il significato dei dati e il loro formato.
2. Progettare e scrivere un programma java che, tramite interfaccia GUI legga il file con i dati, li elabori e visualizzi in una forma grafica i dati stessi, in modo da poterli interpretare e poi discutere.
   La lettura del file può essere utilizzata per inserire i vari dati in un vettore (in modalità pila o coda a vostra scelta). Potete usare sia la classe Vector che ArrayList a scelta.

Svolgimento

Trovare gli open data

Per recuperare i dati per il progetto siamo andati sul sito dell'Arpa Piemonte nella sezione Open Data dove abbiamo scelto di analizzare il file .csv:
http://webgis.arpa.piemonte.it/geoportalserver_arpa/catalog/search/resource/details.page?uuid=ARLPA_TO%3ASC05-CLI-4_2017-10-18&title=Temperatura%20massima%20e%20minima%20stagionale%20osservata%20della%20citt%C3%A0%20di%20Tor ino%20dal%201991%20al%202017

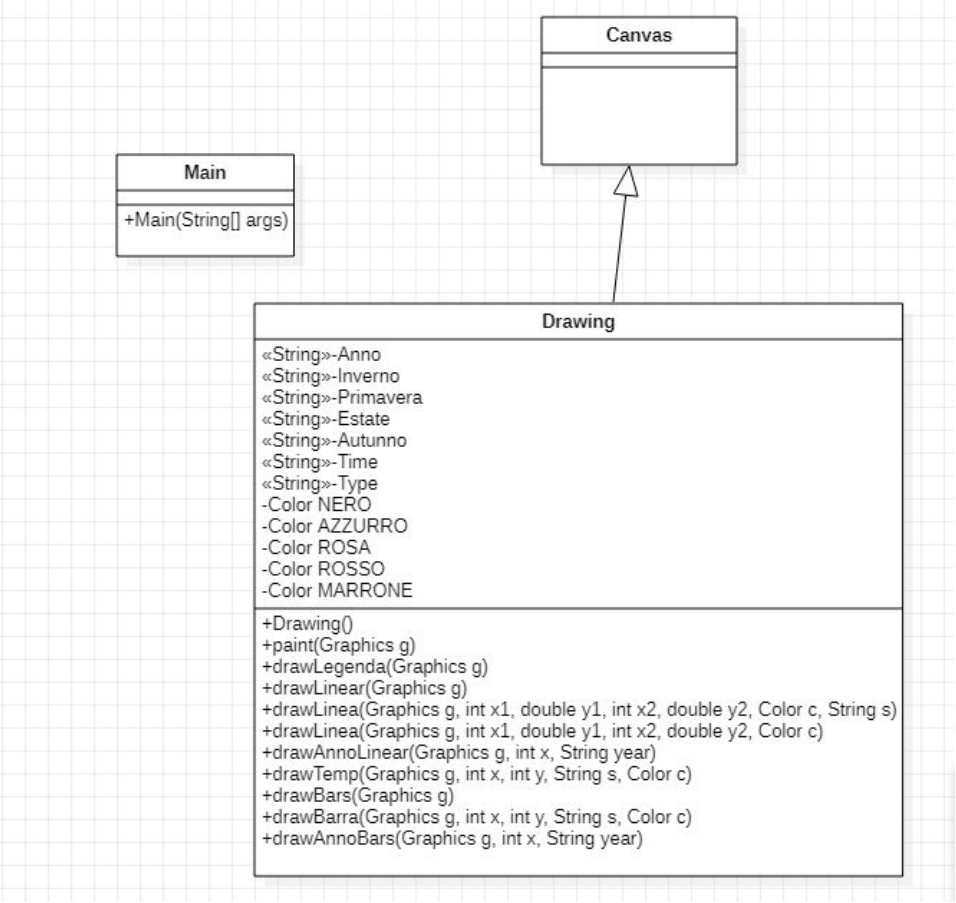
Abbiamo modificato leggermente il file perchè abbiamo notato che il file era suddiviso in due tabelle, una con i valori minimi e una con i valori massimi e notando che i valori delle due tabelle coincidevano abbiamo scartato una tabella (quella con i valori minimi).
Il file utilizzato nel programma è stato chiamato: "**temperature_max.csv**".

Cosa sono gli Open Data?

Gli Open Data sono alcuni tipi di dati (informazioni, dati numerici, ecc...) che possono essere liberamente utilizzati e ridistribuiti da chiunque.
Gli aspetti più importanti degli Open Data sono:

- Disponibilità e accesso: I dati devono essere distribuiti in un formato utile e modificabile.
- Riutilizzo e ridistribuzione: I dati forniti permettono il loro riutilizzo e la loro ridistribuzione.
- Partecipazione universale: Tutti devono essere in grado di riutilizzare e ridistribuire i dati. Non ci devono essere discriminazioni né in ambito di iniziativa né contro soggetti o gruppi.

Programma



Il nostro programma è stato suddiviso in due classi: la classe Main e la classe Drawing.

Nel Main attraverso un try-catch controlliamo che esista effettivamente un file esterno temperature_max.csv.
Dopodichè creiamo un oggetto file che apre il file e poi un oggetto Scanner che ci servirà per leggere il file. Creiamo cinque ArrayList differenti che conterranno quello che andremo poi a leggere.

```
ArrayList<String> anno = new ArrayList();
ArrayList<String> inverno = new ArrayList();
ArrayList<String> primavera = new ArrayList();
ArrayList<String> estate = new ArrayList();
ArrayList<String> autunno = new ArrayList();
```

Creiamo un piccolo ciclo While che mi servirà a leggere il file, finchè ci sarà una riga dopo quella che prendiamo in considerazione: in questo caso scartiamo la prima riga del file perchè non ci serve e con il ciclo while leggiamo dalla seconda in poi.

inputStream.nextLine(); legge una riga alla volta.

inputStream.hasNext(); controlla se è presente una riga dopo.

split divide una stringa in più stringhe

anno.add(cell[0])--> La cella alla posizione zero contiene l'anno, la cella alla posizione uno contiene i dati invernali e così via per tutti gli indici.

Nella classe Drawing abbiamo un metodo costruttore che inizializza i seguenti attributi:

- Anno
- Inverno
- Primavera
- Autunno
- Time (mi indica quale stagione voglio vedere o se voglio vederle tutte)
- Type (mi indica il tipo di grafico che scelgo)

Oltre al metodo costruttore nella classe Drawing sono presenti i seguenti metodi:

- **public void paint(Graphics g)** : Metodo che mi consente di chiamare altri due metodi (drawLinear e drawBars) per disegnare un grafico lineare o a barre. FAcendo l'overraide su paint posso disegnare quello che voglio, in questo caso dei grafici.
- **private void drawLegenda(Graphics g)**: Metodo che mi permette di disegnare una legenda per rendere più comprensibile il grafico.
- **private void drawLinear(Graphics g):** Metodo che ci permette di disegnare un grafico lineare.

- **private void drawLinea(Graphics g, int x1, double y1, int x2, double y2, Color c, String s):** Metodo che disegna una linea partendo dai parametri passati. Disegna la linea con il colore che li viene passato, aggiunge un pallino al secondo punto della linea e in più aggiunge una stringa, contenente la temperatura passata dai parametri, vicino al pallino.
- **private void drawLinea(Graphics g, int x1, double y1, int x2, double y2, Color c):** Metodo che disegna una linea partendo dai partendo dai parametri passati. In sostanza è come il metodo descritto sopra ma con la differenza che non mi scrive nessuna stringa.
- **private void drawAnnoLinear(Graphics g, int x, String year):** Metodo che mi disegna una linea verticale e mi scrive sotto l'anno.
- **private void drawTemp(Graphics g, int x, int y, String s, Color c):** Metodo che mi permette di scrivere il valore della temperatura vicino all'anno.
- **private void drawBars(Graphics g):** Metodo che mi permette di disegnare il grafico a barre.
- **private void drawBarra(Graphics g, int x, int y, String s, Color c):** Metodo che mi disegna le singole barre del colore specificato scrivendoci sopra anche i valori. Siccome posso disegnare barre che vanno solo verso basso, per disegnare le barre con temperatura positiva devo togliere valore alle y(500- y: dove 500 è la linea 0 delle x), mentre per disegnare barre che indicano una temperatura negativa andiamo a 500 e come altezza della barra prendiamo il valore assoluto della y.
- **private void drawAnnoBars(Graphics g, int x, String year):** Metodo che scrive l'anno che gli viene passato.