

Problema dei Lettori e Scrittori in Java

Giordano Iaquina e Guido Testi

December 3, 2025

1 Introduzione

Il progetto affronta il classico problema dei Lettori e Scrittori, che riguarda la gestione concorrente di una risorsa condivisa. Nel nostro caso la risorsa è una **lavagna**, sulla quale più persone possono leggere contemporaneamente ma solo una per volta può scrivere.

L'obiettivo principale è impedire conflitti tra i thread e garantire che la scrittura avvenga in modo esclusivo, mentre la lettura può avvenire in parallelo se non è in corso una scrittura.

2 Obiettivo

L'obiettivo del progetto è simulare una lezione in cui:

- Gli scrittori scrivono uno alla volta sulla lavagna.
- I lettori possono leggere solo quando non è in corso una scrittura.
- L'accesso alla lavagna è gestito in modo sicuro.
- Tutte le operazioni terminano allo scadere del tempo della lezione.

3 Struttura del progetto

Il progetto è composto da quattro classi principali:

- Applicazione
- Lettore
- Scrittore
- Lavagna

4 Classe Applicazione

La classe **Applicazione** è il punto di avvio del programma e si occupa di:

- Leggere in input il numero di lettori, scrittori e la durata della lezione.
- Gestire l'inserimento dei nomi o la loro generazione automatica.
- Creare l'oggetto condiviso **Lavagna**.
- Creare e avviare i thread **Lettore** e **Scrittore**.
- Attendere la fine della lezione.
- Attendere la terminazione di tutti i thread con il metodo `join()`.

5 Classe Lettore

La classe `Lettore` rappresenta un thread che:

- Tenta di accedere alla lavagna tramite `inizioLettura()`.
- Se ottiene il permesso, simula il tempo di lettura con `Thread.sleep(500)`.
- Chiude la lettura con `fineLettura()`.
- Attende prima di un nuovo tentativo con `Thread.sleep(800)`.
- Continua fino allo scadere della lezione.

6 Classe Scrittore

La classe `Scrittore` rappresenta un thread che:

- Tenta di accedere alla lavagna tramite `inizioScrittura()`.
- Se ottiene il permesso, simula il tempo di scrittura con `Thread.sleep(800)`.
- Termina la scrittura con `fineScrittura()`.
- Attende prima di un nuovo tentativo con `Thread.sleep(1000)`.
- Continua fino allo scadere della lezione.

7 Classe Lavagna

La classe `Lavagna` gestisce l'accesso alla risorsa condivisa attraverso le variabili:

- `lettori` per contare i lettori attivi.
- `scrittura` per sapere se uno scrittore sta scrivendo.
- `lezioneFinita` per bloccare l'accesso alla fine della lezione.

I principali metodi sono:

- `inizioScrittura()` con guardia `while(scrittura || lettori > 0)`.
- `fineScrittura()` per liberare la lavagna.
- `inizioLettura()` con guardia `while(scrittura)`.
- `fineLettura()` per decrementare i lettori.
- `finisciLezione()` per terminare tutte le operazioni.

8 Gestione della concorrenza

La concorrenza è gestita tramite:

- Sezioni critiche protette nei metodi della lavagna.
- Meccanismo di attesa con `wait()`.
- Risveglio dei thread con `notifyAll()`.

In questo modo:

- Gli scrittori scrivono in modo esclusivo.
- I lettori possono leggere in parallelo.
- Non si verificano race condition.

9 Flusso della simulazione

1. Inserimento dati da parte dell'utente.
2. Creazione della lavagna condivisa.
3. Avvio dei thread Lettore e Scrittore.
4. Tentativi continui di accesso alla lavagna.
5. Simulazione di lettura e scrittura.
6. Scadenza del tempo della lezione.
7. Blocco degli accessi e terminazione dei thread.

10 Esempio di output

```
1 Persona 1 SCRIVE
2 Persona 1 HA FINITO DI SCRIVERE
3 Persona 3 LEGGE
4 Persona 4 LEGGE
5 Persona 5 LEGGE
6 Persona 3 HA FINITO DI LEGGERE
7
8 >>> CAMPANELLA: fine lezione <<<
```

11 Vantaggi e svantaggi

11.1 Vantaggi

- Corretta gestione della concorrenza.
- Scrittura sempre esclusiva.
- Lettori in parallelo.
- Simulazione realistica.

11.2 Svantaggi

- Ordine di esecuzione non prevedibile.
- Presenza di tempi di attesa tra le operazioni.
- Comportamento dipendente dal tempo di esecuzione dei thread.

12 Conclusioni

Il progetto dimostra una corretta applicazione dei thread in Java per risolvere il problema dei Lettori e Scrittori. Attraverso l'utilizzo della classe Lavagna come risorsa condivisa e l'impiego di guardie, è stato possibile garantire sia l'accesso esclusivo alla scrittura sia la lettura concorrente senza conflitti.