ESAME DI FONDAMENTI DI INFORMATICA T-2 del 9/09/2015
Proff. Enrico Denti – Gabriele Zannoni
Tempo a disposizione: 4 ore MAX

NB: il candidato troverà nell'archivio ZIP scaricato da Esamix anche il software "Start Kit"

NOME PROGETTO ECLIPSE: matricola-CognomeNome (es. RossiMario-0000123456)

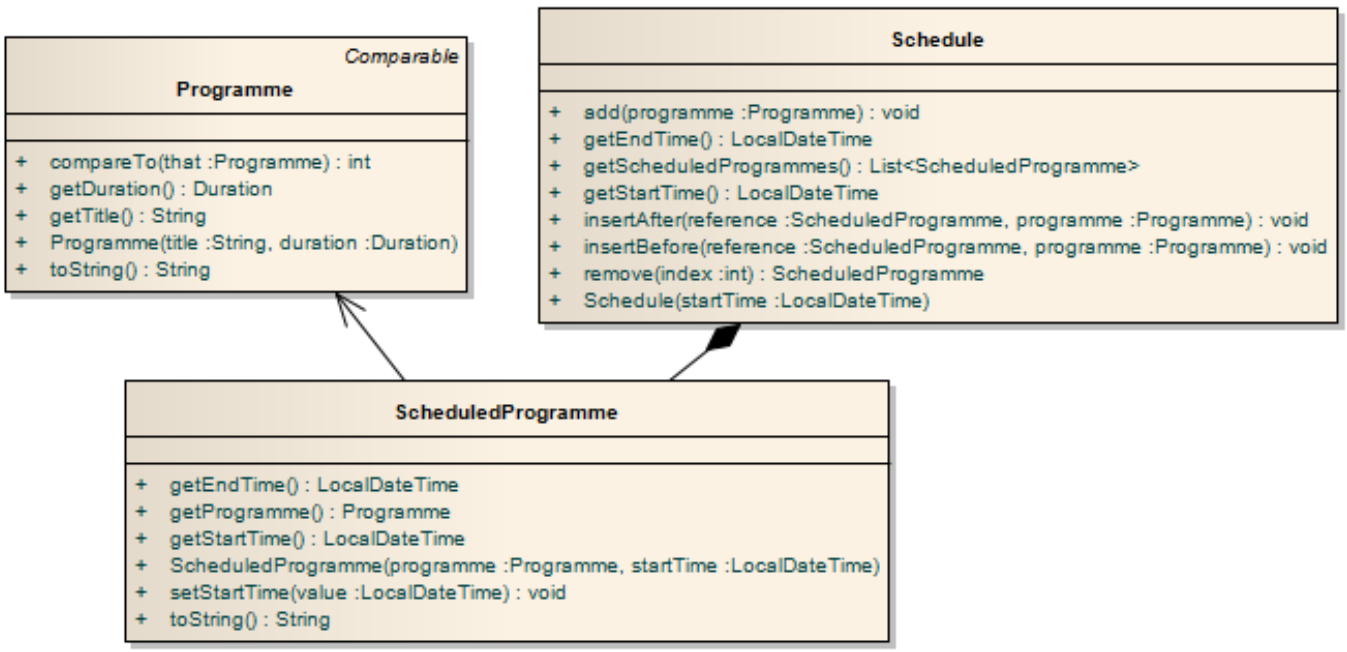
La rete televisiva MyTV ha richiesto un'applicazione per facilitare la costruzione del proprio palinsesto.

> DESCRIZIONE DEL DOMINIO DEL PROBLEMA.
> Un *programma* è caratterizzato da una descrizione e da una durata (espressa solitamente in minuti). Per *palinsesto* si intende un elenco di programmi da trasmettere, posti temporalmente uno di seguito all'altro e collocati ciascuno in un ben preciso slot temporale: ovviamente, uno stesso programma può essere presente più volte entro il palinsesto (repliche).

Il file di testo Programmes.txt contiene l'elenco dei programmi disponibili, che l'applicazione assemblerà per costruire il palinsesto desiderato.

Parte 1 – Modello dei dati *(namespace mytv.model)* Punti: 8

Il modello dei dati deve essere organizzato secondo il diagramma UML riportato in Figura.



a) la classe ***Programme*** (fornita) rappresenta un programma: i due accessor ***getTitle*** e ***getDuration*** restituiscono rispettivamente la descrizione del programma e la sua durata, ossia gli stessi argomenti forniti al costruttore, mentre ***toString*** restituisce un'opportuna stringa descrittiva, con la durata espressa in minuti; la classe è ***Comparable*** secondo ordinamento alfabetico della descrizione;

b) la classe ***ScheduledProgramme*** (fornita) rappresenta un programma collocato in un ben preciso istante di tempo (giorno e ora locali): concettualmente si tratta quindi di una composizione di un ***Programme*** con un ***LocalDateTime***; opportuni accessor restituiscono la data/ora iniziale e la data/ora finale del programma, oltre agli altri ovvi elementi; ***toString*** restituisce un'opportuna stringa descrittiva;

c) la classe ***Schedule*** **(da realizzare)** rappresenta il palinsesto, ossia una lista di ***ScheduledProgramme*** temporalmente adiacenti e con inizio in un ben preciso istante di tempo (***LocalDateTime***).

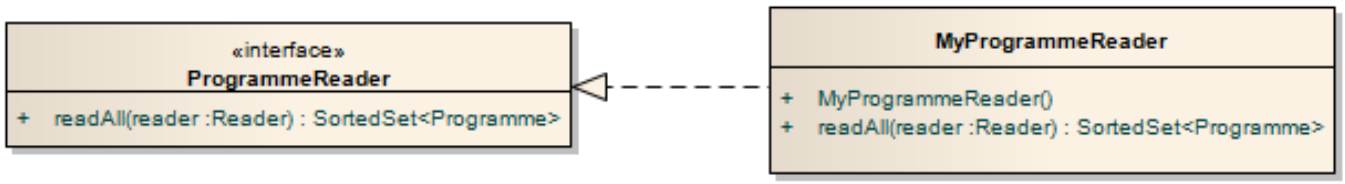
- il costruttore riceve come unico argomento l'istante di tempo iniziale dell'intero palinsesto (un ***LocalDateTime***), che ovviamente dev'essere non nullo – altrimenti, lancia ***IllegalArgumentException***

- i due metodi ***getStartTime*** e ***getEndTime*** restituiscono rispettivamente la data e ora iniziale/finale dell'intero palinsesto (l'ora finale è semplicemente l'ora di fine dell'ultimo programma), mentre il metodo ***getScheduledProgrammes*** restituisce la lista di ***ScheduledProgramme*** che costituiscono il palinsesto stesso;
- il metodo ***add*** aggiunge un ***Programme*** in coda al palinsesto attuale (il ***Programme*** è incapsulato in uno ***ScheduledProgramme*** al quale è assegnato il corretto istante di inizio);
- il metodo ***remove*** elimina un ***Programme*** dalla posizione passata come argomento – dopo la rimozione è necessario aggiornare le ore di inizio degli ***ScheduledProgramme*** successivi;

Parte 2 – Persistenza *(namespace mytv.persistence)* Punti: 6

Come sopra anticipato, il file di testo Programmes.txt contiene l'insieme dei programmi disponibili, uno per riga. Ogni riga contiene la descrizione del programma (che può avere lunghezza arbitraria e contenere *qualunque carattere, tranne le tabulazioni e i fine riga*) seguita da una tabulazione e dalla durata del programma stesso, nella forma `ore:minuti`: le ore possono essere su una o più cifre, mentre i minuti sono sempre su due cifre (Attenzione: poiché si tratta di una durata non è possibile utilizzare un *formatter*, ma occorre agire manualmente).

a) L'interfaccia ***ProgrammeReader*** (fornita) dichiara il metodo ***readAll*** che legge da un *reader* tutti i programmi e



restituisce un ***SortedSet<Programme>*** con tutti i programmi disponibili, ordinato alfabeticamente;

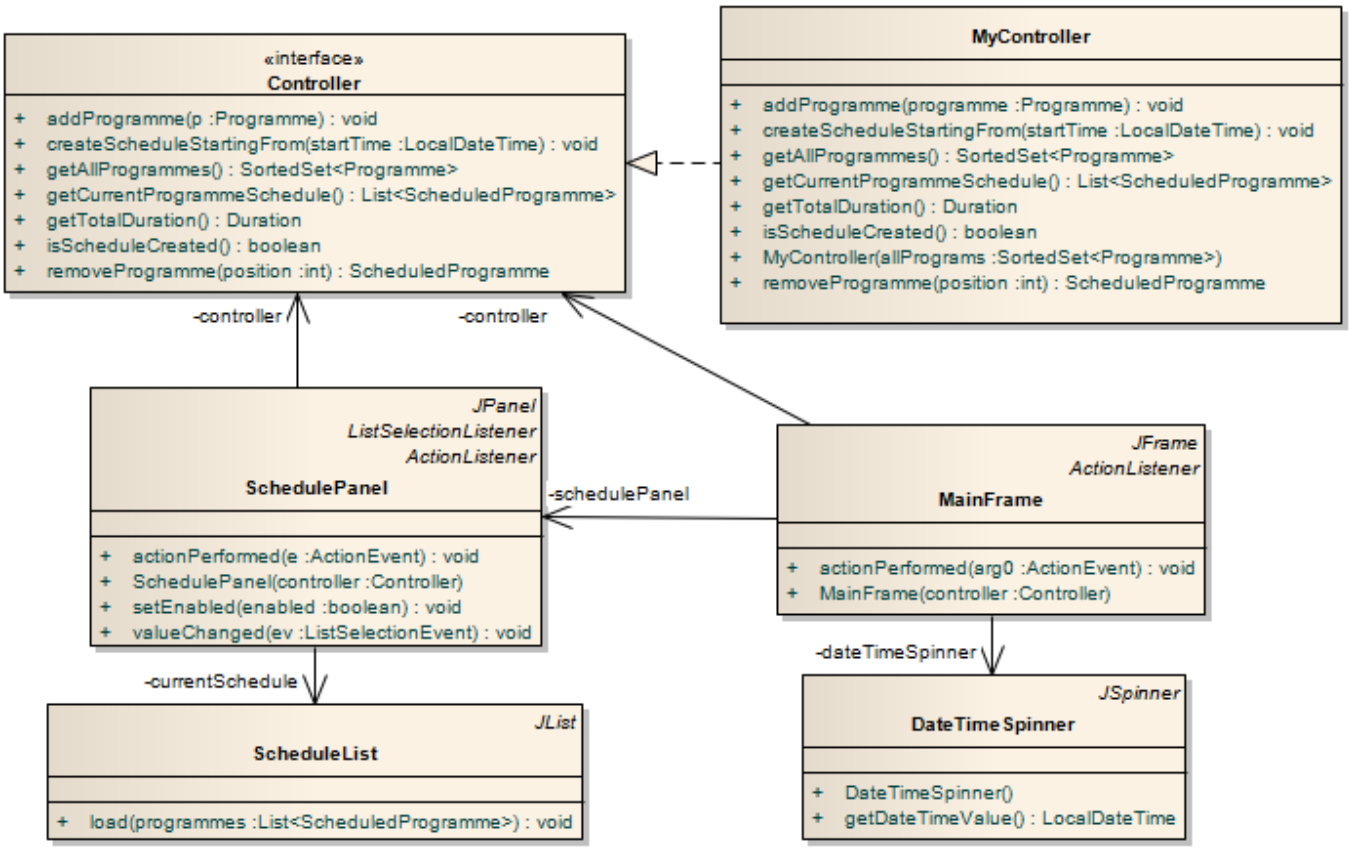
b) La classe ***MyProgrammeReader*** (**da realizzare**) implementa tale interfaccia, lanciando ***IOException*** in caso di problemi di accesso al file o incapsulando in una ***BadFileFormatException*** gli eventuali problemi di formato; il costruttore non prevede argomenti.

Parte 3 – GUI *(namespace mytv.ui)* (punti: 16)

Controller *(punti 7)*

a) L'interfaccia ***Controller*** dichiara i seguenti metodi:
- Il metodo ***createScheduleStartingFrom*** crea un nuovo ***Schedule*** con la data/ora di inizio specificata come argomento – tale ***Schedule*** non viene restituito dal metodo ma memorizzato all'interno del *controller* al fine di poterlo popolare con i programmi;
- Il metodo ***isScheduleCreated*** indica se sia stato già creato uno ***Schedule*** e se sia quindi possibile invocare i metodi del *controller* che seguono;
- il metodo ***getAllProgrammes*** restituisce l'insieme ordinato di tutti i ***Programme*** disponibili;
- il metodo ***addProgramme*** aggiunge un programma in coda al palinsesto corrente;
- il metodo ***removeProgramme*** elimina il programma dalla posizione del palinsesto passata come argomento e lancia ***IllegalArgumentException*** in caso di argomenti assurdi e/o inesistenti;
- il metodo ***getCurrentProgrammeSchedule*** restituisce la lista degli ***ScheduledProgramme*** attualmente inseriti nel palinsesto;
- il metodo ***getTotalDuration*** calcola e restituisce la durata totale del palinsesto corrente.

b) La classe ***MyController*** (**da realizzare**) implementa tale interfaccia: il costruttore riceve in ingresso l'insieme ordinato dei programmi



Interfaccia utente (namespace mytv.ui) *(punti 9)*

L'interfaccia utente deve essere simile (non necessariamente identica) all'esempio mostrato nelle figure seguenti.

La classe ***MainFrame*** (fornita) realizza la finestra principale che contiene un ***DateTimeSpinner*** (fornito), un bottone per consentire di scegliere e creare uno Schedule con un'ora di inizio scelta dall'utente e uno ***SchedulePanel***.

La classe ***SchedulePanel*** (da realizzare) realizza il pannello che consente di operare su uno ***Schedule*** (il costruttore prende un ***Controller*** come parametro) e contiene una casella a discesa (combo) pre-popolata con tutti i programmi disponibili, elencati in ordine alfabetico. Sotto, una lista (***ScheduleList*** – fornita – derivata da ***JList*** e personalizzata in modo da visualizzare un elenco di ***ScheduledProgramme***), opportunamente incapsulata in un pannello a scorrimento (***ScrollPane***) che compare solo se necessario, è destinata a contenere i programmi via via inseriti dall'utente nel palinsesto, la cui durata complessiva è sempre mostrata nella riga di stato sottostante (Fig. 1), in bianco su fondo blu. L'utente compone il palinsesto semplicemente selezionando programmi dalla combo: ogni programma selezionato viene immediatamente aggiunto in coda al palinsesto, con conseguente aggiornamento della durata (Fig. 2).

L'utente può rimuovere un elemento precedentemente inserito nel palinsesto semplicemente cliccandoci sopra (ossia cliccando sull'elemento della ***ScheduleList*** da rimuovere): in risposta, l'applicazione deve far apparire un ***JOptionPane*** di conferma, prodotto dal metodo *factory* ***showConfirmDialog***, che deve specificare anche la posizione dell'elemento da rimuovere per distinguere eventuali occorrenze multiple (Fig. 3; la posizione a uso utente dev'essere numerata da 1, anche se internamente la numerazione parte da 0): se l'utente conferma la rimozione, la ***ScheduledProgramme*** e la durata devono essere aggiornate di conseguenza (Fig. 4).

L'area di testo deve utilizzare il font "Verdana", grassetto, 12 punti: la durata dev'essere mostrata in ore e minuti, correttamente formattata (*non solo in minuti*) – si suggerisce l'uso del metodo ***formatDuration*** di ***DurationHelper*** (fornita) contenuta nel package mytv.util.

DIMENSIONI CONSIGLIATE: etichette 170x20, riga di stato larghezza 22.

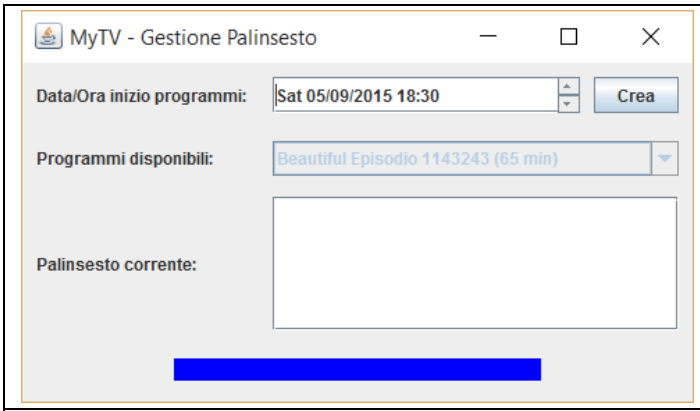


Fig. 1

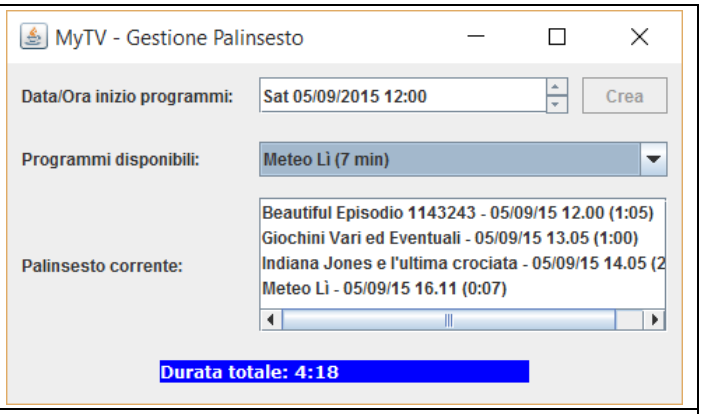


Fig. 2

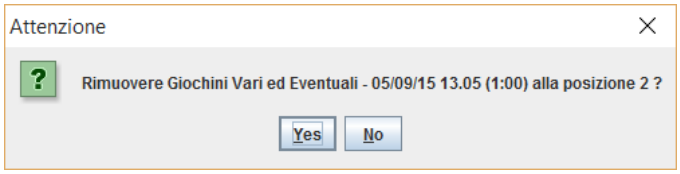


Fig. 3

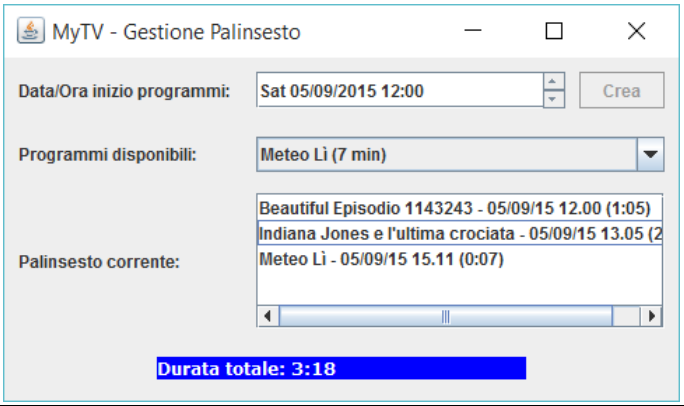


Fig. 4