Università degli Studi di Trento 16 febbraio 2024

Programmazione 2

Prova al calcolatore *Proff. Marco Patrignani e Gian Pietro Picco*

Obiettivi e funzionalità richieste

Si crei l'applicazione *Farmacia*, che permette di acquistare medicine, in una finestra `750 x 500`.

L'applicazione mostra i soldi attuali (30$), e la data odierna (usate `new java.util.Date()`) in basso.

L'applicazione mostra le medicine come testo incolonnato a sinistra. Le medicine hanno un nome, un costo, e una data di scadenza (indicata solo se non è scaduta, altrimenti si scriva `scaduta`). Le medicine possono essere da banco o con ricetta, quelle con ricetta possono essere ripetibili (e in tal caso indicano per quante volte si possono ripetere) o non ripetibili. Le medicine con ricetta hanno uno sconto: quelle ripetibili hanno uno sconto di 5$, quelle non ripetibili hanno uno sconto di 10$. Una volta stampate, le medicine presentano le informazioni come in figura.

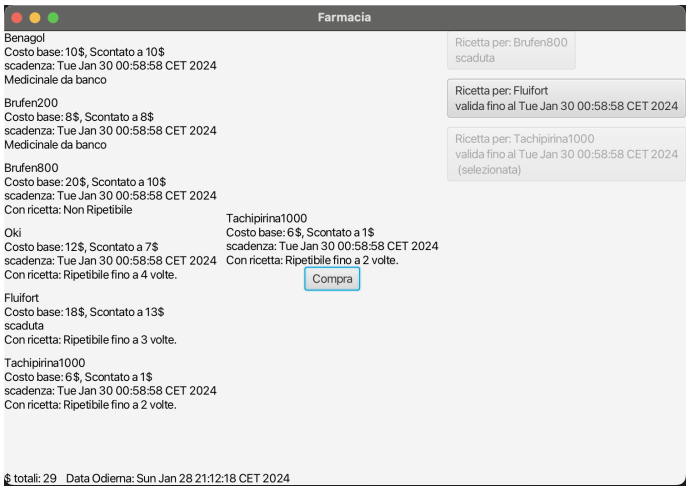
L'applicazione mostra le ricette dell'utente come bottoni incolonnati a destra. Le ricette identificano il nome della medicina per la quale sono valide e hanno una data di scadenza (indicata solo se non è scaduta, altrimenti si scriva `scaduta`) e un valore (tenuto interno) che indica quante volte sono state usate. Le ricette sono disabilitate se sono scadute. Una volta clickate, le ricette diventano disabilitate, ma cambia il loro testo e appare la scritta `selezionata` (vedere la terza ricetta in figura). Una ricetta diventa ri-abilitata dopo che viene comprato qualcosa (vedere di sotto).

Cliccando sul testo di una medicina, questa appare al centro seguita dal pulsante `Compra`. Premendo il pulsante compra, vengono scalati i soldi all'utente, viene deselezionata la ricetta, e viene resettato il centro della GUI, a meno di una problematica, in tal caso viene mostrato un messaggio di `Alert` all'utente con l'indicazione della problematica. Le problematiche sono: (1) soldi insufficienti, (2) medicina scaduta, (3) la medicina richiede una ricetta ma l'utente non ha la ricetta, (4) la medicina richiede una ricetta che non è stata selezionata, (5) la ricetta ripetibile è stata usata più volte di quanto indicato nella medicina.

Riguardo alle date, usate `java.util.Date` e i relativi metodi `after` e `before`. Partendo da un oggetto `Date` che sia la data odierna, potete aggiungere `100000000` per ottenere la data di domani, e sottrarre `100000000` per ottenere la data di ieri. Non ci si aspetta le date come in figura, ma le date di oggi, di ieri, e di domani.

Il padding tra gli elementi della GUI è di `10` pixel, l'allineamento centrale è `CENTER`. Si inizializzi il database di ricette e di medicine come segue:

- Ricetta per Brufen800, scadenza *ieri*;
- Ricetta per Tachipirina1000, scadenza *domani*;
- Ricetta per Fluifort, scadenza *domani*.



(a) Applicazione con medicina e ricetta selezionate (Tachipirina1000).

- `Benagol`, medicinale da banco, costo 10, scadenza *domani*;
- `Brufen200`, medicinale da banco, costo 8, scadenza *domani*;
- `Brufen80`, medicinale con ricetta non ripetibile, costo 20, scadenza *domani*;
- `Oki`, medicinale con ricetta ripetibile 4 volte, costo 12, scadenza *domani*;
- `Fluifort`, medicinale con ricetta ripetibile 2 volte, costo 18, scadenza *ieri*;
- `Tachipirina1000`, medicinale con ricetta ripetibile 3 volte, costo 6, scadenza *domani*

Requisiti

- Si usi, dove evidente/utile, una gerarchia di ereditarietà, specializzando opportunamente il *comportamento* delle classi e sfruttando ove possibile il polimorfismo.
- Si usino costanti ove ragionevole (es., quando uno stesso valore viene usato più di una volta) e si badi alla pulizia del codice (es., linee guida Java) evitando duplicazioni e codice inutilmente complesso.
- Al di fuori della definizione di costanti, si usi il modificatore `static` solo quando opportuno.
- La disposizione degli elementi grafici deve rispecchiare fedelmente quanto mostrato; tuttavia, ciò è considerato meno importante rispetto alla corretta funzionalità dell'interfaccia e del resto dell'applicazione.
- Il mancato rispetto delle specifiche (posizioni/dimensioni, colori, funzionalità, etc.) genera penalità in correzione.
- (**Solo per studenti del prof. Patrignani**) Si richiede il class diagram UML. È sufficiente la vista di alto livello (no attributi/metodi) ma devono essere mostrate le associazioni più importanti, oltre a quella di ereditarietà. Il diagramma UML deve essere consegnato su un foglio protocollo recante nome, cognome, numero di matricola.