ESAME DI FONDAMENTI DI INFORMATICA T-2 del 13/06/2018

Proff. E. Denti – R. Calegari – G. Zannoni Tempo: 4 ore

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

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

NOME ZIP DA CONSEGNARE: CognomeNome-matricola.zip (es. RossiMario-0000123456.zip)

La società *ElectricLife* ha richiesto lo sviluppo di un'applicazione che permetta ai potenziali utenti di simulare il costo della bolletta elettrica con le proprie tariffe.

DESCRIZIONE DEL DOMINIO DEL PROBLEMA.

La società offre sia tariffe *a consumo*, sia tariffe *flat* che comprendono già una certa quota di KWh mensili.

Nelle **tariffe a consumo** l'importo da pagare dipende dai KWh effettivamente consumati, mentre nelle **tariffe flat** l'importo da pagare ogni mese è fisso, purché i KWh consumati restino entro una soglia prestabilita; eventuali KWh in eccesso sono tariffati come extra, al prezzo specificato. In entrambi i casi al costo dell'energia devono essere aggiunte le **tasse** (accise e IVA), come più oltre precisato.

Più precisamente, una **tariffa a consumo** è caratterizzata da:

- nome dell'offerta commerciale
- prezzo del singolo KWh, in Euro

mentre ogni **tariffa flat** è caratterizzata da:

- nome dell'offerta commerciale
- quota di KWh inclusi nell'offerta mensile e relativo importo fisso, in Euro
- prezzo degli eventuali KWh consumati oltre la soglia, in Euro/KWh.

Le tasse comprendono due voci:

- le **accise**, calcolate in proporzione ai KWh fatturati (2,27 Eurocent/KWh), esclusi i primi 150 KWh mensili, esenti;
- **l'IVA** (10%), applicata sul subtotale precedente, accise incluse

Il totale della fattura va arrotondato a due cifre decimali, come previsto dalle normative.

ESEMPI

1. Tariffa flat a € 20 /mese con 150 KWh inclusi (eventuali KWh extra € 0,25/KWh): se il consumo resta entro la soglia, la fattura mensile sarà € 20 +10% IVA = **€ 22,00** [niente accise perché i primi 150 KWh sono esenti]
2. Tariffa a consumo al prezzo di € 0,14 /KWh, consumo 150KWh: la fattura sarà (€ 0,14 * 150) +10% IVA = € 21 +10% = **€ 23,10** [niente accise perché i primi 150 KWh sono esenti]
3. Tariffa flat a € 20 /mese con 150 KWh inclusi (eventuali KWh extra € 0,25/KWh), consumo reale 184KWh: la fattura mensile sarà (€ 20 + € 0,25 * 34 + € 0,0227*34) +10% IVA = (€ 20 + € 9,2718) +10% = **€ 32,20**
   [accise calcolate sui soli 34 KWh eccedenti la soglia dei 150 esenti]
4. Tariffa a consumo al prezzo di € 0,14 /KWh, consumo 184KWh: la fattura sarà (€ 0,14 * 184 + € 0,0227*34) +10% IVA = (€ 25,76 + € 0,7718) +10% = **€ 29,18** [accise calcolate sui soli 34 KWh eccedenti la soglia]

Il file di testo Tariffe.txt contiene la descrizione delle diverse tariffe, nel formato più oltre specificato.

IMPORTANTE: a causa di modifiche nel provider delle specifiche **Locale** intervenute fra Java 8 e Java 9, ***il formattatore di valute per Locale.ITALY opera diversamente in Java 9 rispetto a Java 8.*** Ciò impatta anche il funzionamento dei metodi *parse* utilizzati per la conversione stringa(prezzo)/numero.

Per ottenere il comportamento classico è necessario aggiungere alle *Run Configurations* →*Arguments* la specifica:

`-Djava.locale.providers=COMPAT`

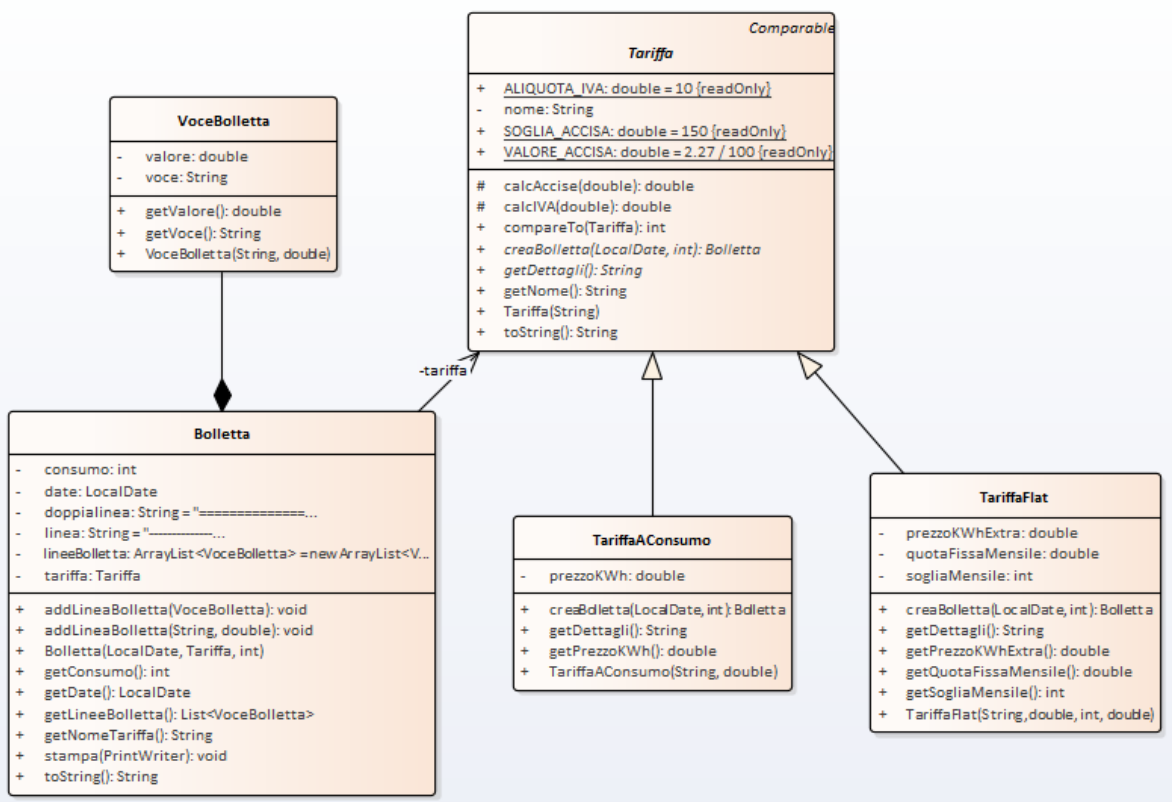
Essa va aggiunta sia alla *run configuration* di JUnit, sia a quella dell'Application, sia a quella dell'ApplicationMock.

NB: le run configuration vengono generate da Eclipse ***dopo il primo run*** di JUnit o dell'application stessa.

Parte 1 (punti: 18)

Dati (namespace electriclife.model) *(punti: 10)*

Il modello dei dati deve essere organizzato secondo il diagramma UML di seguito riportato:



SEMANTICA:

a) la classe ***VoceBolletta*** (fornita) rappresenta una voce di costo di una bolletta, caratterizzata dalle due proprietà *voce* e *valore,* recuperabili tramite opportuni accessor: essa serve a descrivere la struttura della ***Bolletta***.

b) la classe ***Bolletta*** (fornita)rappresenta la bolletta riferita a una certa data di emissione: è caratterizzata dal nome della tariffa applicata, dal consumo rilevato, nonché da una lista di ***VoceBolletta*** che specificano ciascuna una voce di costo. La classe offre metodi per aggiungere una linea di bolletta (***addVoceBolletta***), recuperare la lista delle linee di bolletta presenti (***getLineeBolletta***), produrre una stampa della bolletta su un apposito ***PrintWriter*** (***stampa***), oltre a una adeguata ***toString***.

c) la classe astratta ***Tariffa*** (fornita) rappresenta la generica tariffa caratterizzata da un nome univoco specificato all'atto della costruzione e recuperabile tramite opportuno accessor; è anche ***Comparable*** per nome. Definisce inoltre opportune costanti (*VALORE_ACCISA*, *SOGLIA_ACCISA*, *ALIQUOTA_IVA*) per il calcolo delle varie voci.
I due metodi concreti ***calcAccise*** / ***calcIVA*** restituiscono rispettivamente il costo delle accise / dell'IVA a partire da un consumo in KWh. Il metodo astratto ***creaBolletta***, dato il consumo mensile, crea la corrispondente ***Bolletta*** calcolando le voci di costo, le imposte e l'importo totale come da specifiche. Il metodo astratto ***getDettagli*** restituisce invece una stringa descrittiva della tariffa, che verrà riportata tale e quale in bolletta (v. esempi).

d) **la classe *TariffaAConsumo* (da realizzare)** specializza ***Tariffa*** nel caso delle tariffe a consumo: il costruttore riceve, oltre al nome della tariffa, il prezzo del KWh, recuperabile tramite accessor. Il metodo ***creaBolletta*** deve inserire nella bolletta le quattro linee di bolletta necessarie, ovvero *costo energia, accise, IVA, totale,* mentre il metodo ***getDettagli*** restituisce una stringa del tipo "Tariffa A CONSUMO, Costo KWh € 0,14".

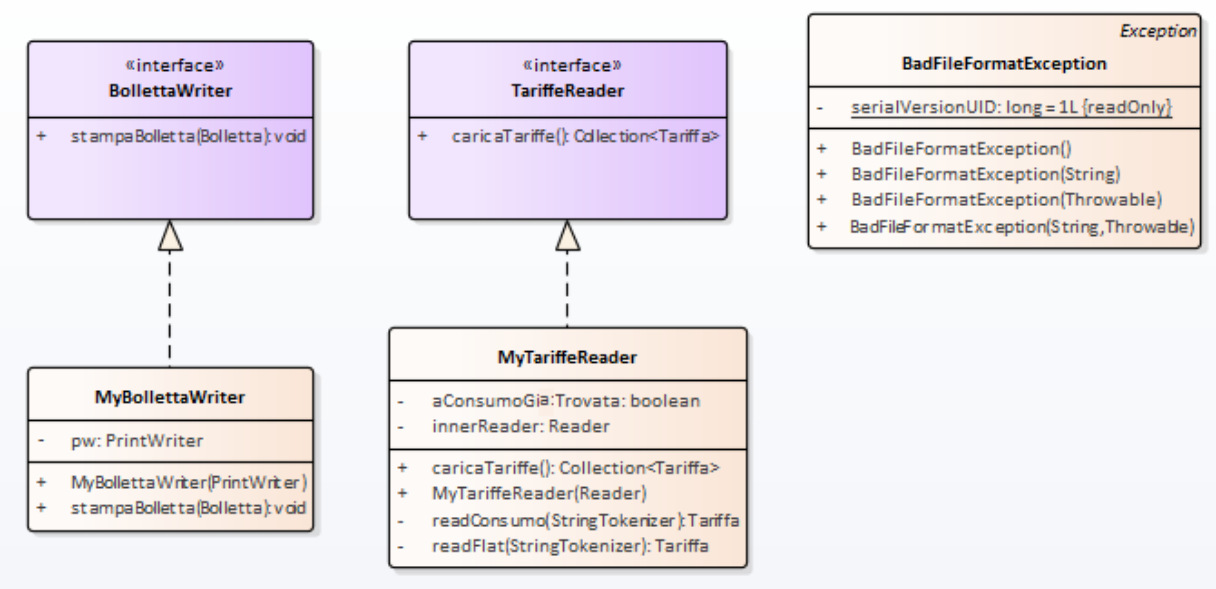
e) **la classe *TariffaFlat* (da realizzare)** specializza ***Tariffa*** nel caso delle tariffe flat: il costruttore riceve, oltre al nome, la quota fissa mensile, la soglia mensile di KWh inclusi e il prezzo degli eventuali KWh eccedenti; tutti questi valori sono recuperabili tramite opportuni accessor. Il metodo ***creaBolletta*** deve inserire nella bolletta le cinque linee di bolletta necessarie, ovvero: *quota fissa mensile*, *costo energia extra soglia*, *accise*, *IVA*, *totale* mentre ***getDettagli*** restituisce una stringa come "Tariffa CASETTA, € 20,00/mese per 150 KWh, poi € 0,25/KWh"

Persistenza (namespace electriclife.persistence) *(punti 8)*

Come già anticipato, il file di testo Tariffe.txt contiene la descrizione delle tariffe offerte, una per riga: **per ovvi motivi può esserci una sola tariffa a consumo**, mentre le combinazioni flat possono essere molteplici. Ogni riga contiene una serie di dati separati fra loro da punti e virgola (‘**;**‘): gli elementi della riga dipendono dal tipo di tariffa (a consumo o flat), come di seguito specificato. **Tutti i prezzi sono formattati in Euro nella forma € xx,xx.**

- Il primo token è costituito dalla tipologia della tariffa: “FLAT” o “A CONSUMO”; poi:
- per le tariffe a consumo, segue semplicemente il prezzo del KWh;
- per le tariffe flat, invece, seguono il nome della tariffa (che può contenere spazi), la soglia mensile (nel formato “SOGLIA” seguita dal valore intero che rappresenta i KWh inclusi), il prezzo mensile e infine il costo degli eventuali KWh extra (nel formato “KWh EXTRA” seguito dal relativo prezzo)

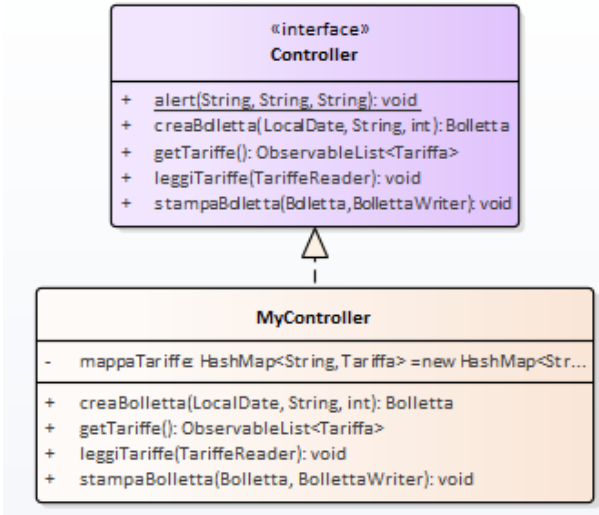
```
ESEMPIO DEL FILE tariffe.txt
FLAT ; CASA MINI  ; SOGLIA 150; € 20,00; KWh EXTRA € 0,25
FLAT; CASA CLASSIC;SOGLIA 250 ; € 30,00; KWh EXTRA € 0,24
FLAT ;CASA BIG ; SOGLIA 350; € 40,00; KWh EXTRA € 0,22
FLAT; CASA MAXI; SOGLIA 450; € 50,00; KWh EXTRA € 0,21
A CONSUMO ; € 0,14
```



L’ interfaccia ***TariffeReader*** (fornita) dichiara il metodo ***caricaTariffe*** che legge una lista di ***Tariffe***.

La classe *MyTariffeReader* (da realizzare) implementa tale interfaccia: il metodo ***caricaTariffe*** deve effettuare i necessari controlli sul formato del file, inclusa la verifica che ci sia **al più** una sola tariffa a consumo, lanciando ***BadFileFormatException*** (fornita) in caso di errori di formato, o propagando ***IOException*** in caso di errori di lettura. Il costruttore riceve il ***Reader*** da cui leggere.

L’ interfaccia ***BollettaWriter*** (fornita) e la corrispondente classe ***MyBollettaWriter*** (pure fornita) rispettivamente dichiarano/implementano il metodo ***stampaBolletta*** che stampa una ***Bolletta***.



Parte 2 (punti: 12)

Controller (namespace electriclife.ui.controller) *(punti 6)*

L’interfaccia ***Controller*** (fornita) dichiara i metodi ***leggiTariffe***, ***getTariffe***, ***creaBolletta*** e ***stampaBolletta*** che devono:

- ***leggiTariffe***: caricare le tariffe su cui operare;

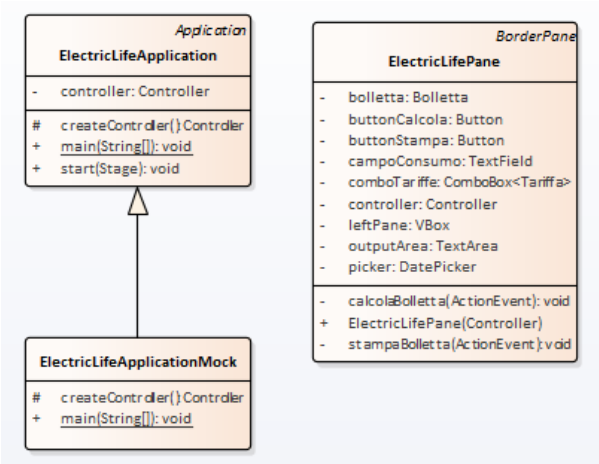
- ***getTariffe***: restituire la lista osservabile ordinata di tutte le tariffe disponibili;
- ***creaBolletta***: produrre la ***Bolletta*** a partire dai dati ricevuti (data di emissione, nome della tariffa, consumo); a tal fine recupera la tariffa per nome e delega poi ad essa la creazione effettiva della ***Bolletta***;
- ***stampaBolletta***: stampare la ***Bolletta*** tramite il ***BollettaWriter*** fornito.

Essa fornisce inoltre il metodo statico ***alert*** che fa comparire una finestra di dialogo all'utente, utile per segnalare errori: i tre argomenti rappresentano il titolo della finestra, l'header e il testo del messaggio (v. Figg. 5 e 6).

La **classe *MyController* (da realizzare)** deve implementare ***Controller*** memorizzando internamente al controller le tariffe caricate (si suggerisce una mappa avente per chiave il nome della tariffa); : in particolare, ***leggiTariffe*** deve lasciar uscire eventuali eccezioni in modo da consentirne poi l'opportuna gestione nella GUI.

Interfaccia utente (namespace electriclife.ui) *(punti 6)*

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



La classe *ElectricLifePane* (da realizzare), che estende ***BorderPane***, deve prevedere:

- una combo box per la selezione della tariffa
- un text field per inserire i dati di consumo;
- un datepicker per selezionare la data di emissione;
- un pulsante *Calcola* per effettuare il calcolo della bolletta, mostrando il risultato nella textarea;
- un pulsante *Stampa* per stampare la bolletta su file.

```
Formato bolletta visualizzata o stampata
============================================================
Electric Life - L'energia che illumina!
Bolletta del 8 maggio 2018
------------------------------------------------------------
Tariffa A CONSUMO, Costo KWh € 0,14
Consumo KWh   184
------------------------------------------------------------
Dettaglio importi:
Costo energia                 € 25,76
Corrispettivo per accise      €  0,77
Corrispettivo per IVA         €  2,65
Totale Bolletta               € 29,18
============================================================
```

NB: la posizione del simbolo € in alcune righe potrebbe essere posposta in Java 9: ciò non costituisce problema.

```
Formato bolletta visualizzata o stampata
============================================================
Electric Life - L'energia che illumina!
Bolletta del 8 maggio 2018
------------------------------------------------------------
Tariffa CASA MINI, € 20,00/mese per 150 KWh, poi € 0,25/KWh
Consumo KWh   150
------------------------------------------------------------
Dettaglio importi:
Quota fissa mensile           € 20,00
Costo energia extra soglia    €  0,00
Corrispettivo per accise      €  0,00
Corrispettivo per IVA         €  2,00
Totale Bolletta               € 22,00
============================================================
```

NB: la posizione del simbolo € in alcune righe potrebbe essere posposta in Java 9: ciò non costituisce problema.

Inizialmente, la combo è popolata con tutte le tariffe disponibili e pre-settata sulla tariffa "A CONSUMO", il datepicker è impostato sulla data corrente e i campi sono vuoti (Fig. 1).

L'utente sceglie la tariffa, inserisce il consumo, sceglie la data di emissione (default: data odierna – Fig. 2) e preme il pulsante *Calcola*: in risposta, l'applicazione mostra la bolletta calcolata (Figg. 3, 4).

Premendo il pulsante *Stampa*, la stessa bolletta viene salvata sul file Bolletta.txt. Da notare che il pulsante *Stampa* è inizialmente disabilitato: si abilita a seguito di un nuovo calcolo e si disabilita a stampa effettuata.

In caso di problemi l'applicazione deve mostrare opportuni dialoghi, sfruttando il metodo statico ***Controller.alert***: consumi negativi o mancanti (Fig. 5), mancanza del file delle tariffe o altri problemi di lettura (Fig. 6), problemi nella stampa della bolletta (Fig. 5).

SUGGERIMENTI

- per fare il parsing di stringhe formattate in valuta utilizzare i metodi ***parse*** del formattatore di valute
- per stampare stringhe formattate in valuta utilizzare i metodi ***format*** del formattatore di valute
- per stampare righe formattate a campi di larghezza fissa può essere utile il metodo ***format*** del ***PrintWriter*** (che, nella versione più ampia, accetta un ***Locale*** come primo argomento)

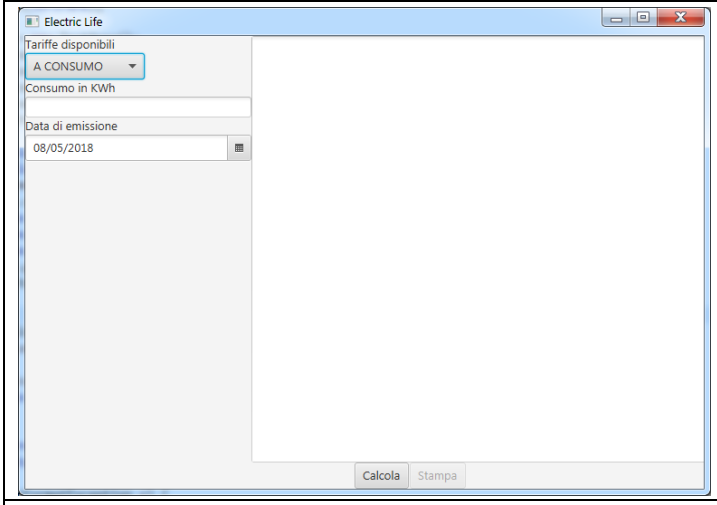


Figura 1

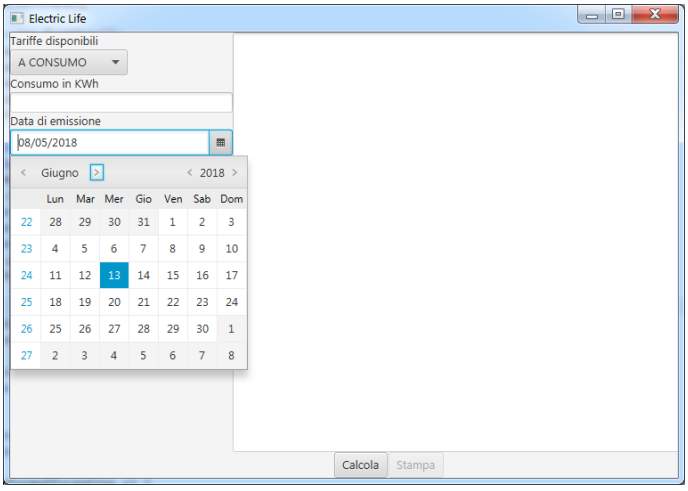


Figura 2

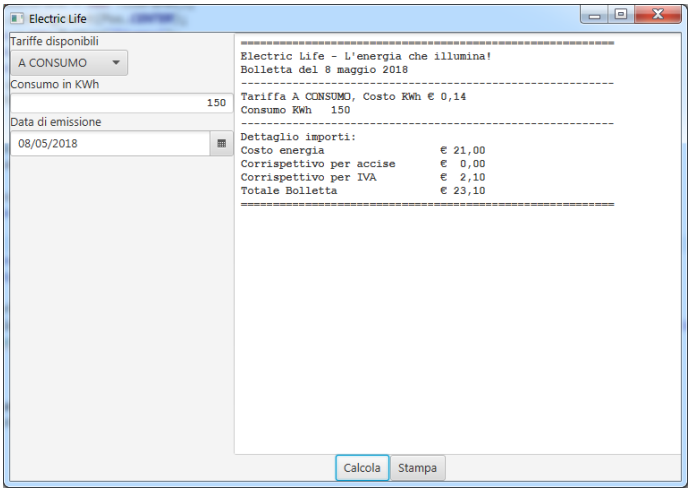


Figura 3

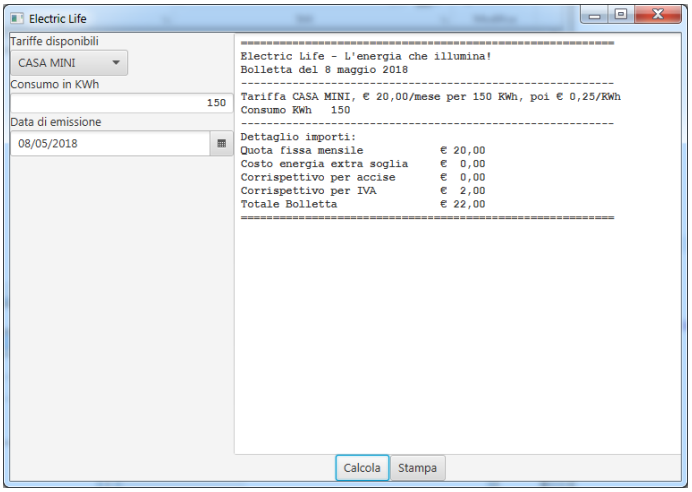


Figura 4

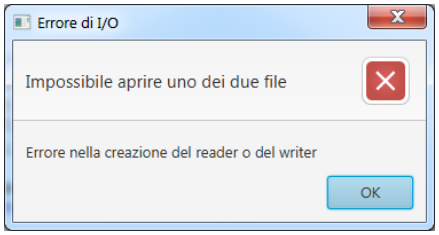


Figura 5

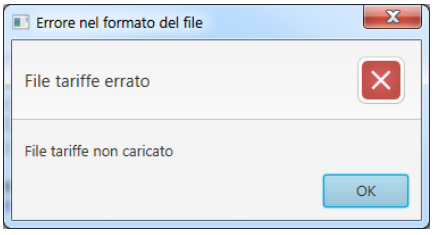


Figura 6