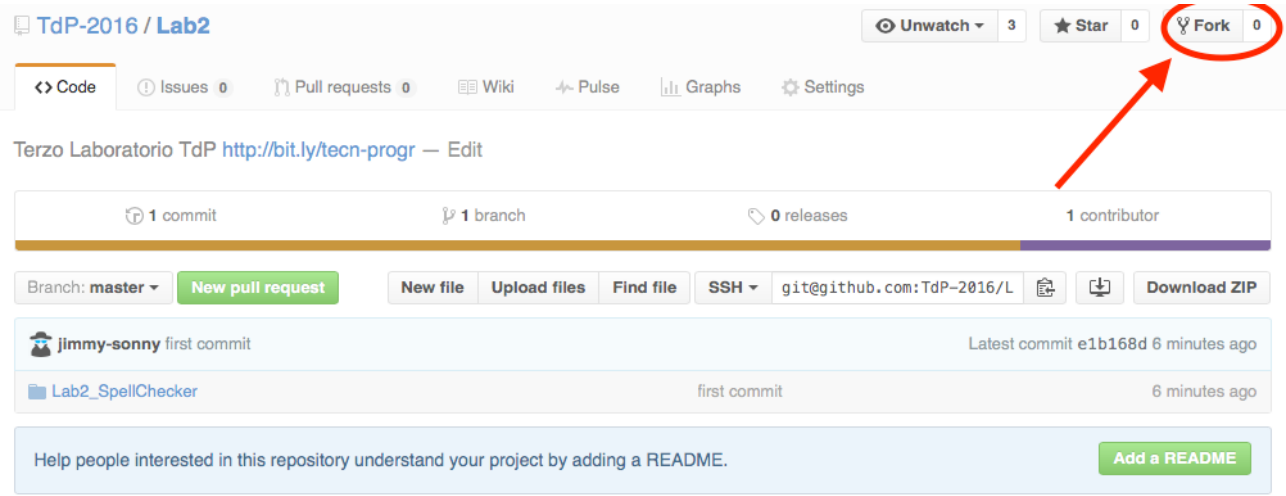
03FYZ TECNICHE DI PROGRAMMAZIONE

Istruzioni per effettuare il fork di un repository GitHub

- Effettuare il login su GitHub utilizzando il proprio username e password.
- Aprire il repository su GitHub relativo al quarto laboratorio:

  https://github.com/TdP-2017/Lab2
- Utilizzare il pulsante *Fork* in alto a destra per creare una propria copia del progetto.



L'azione di Fork crea un nuovo repository nel proprio accout GitHub con una copia dei file necessari per l'esecuzione del laboratorio.

- Aprire Eclipse, andare su *File -> Import*. Digitare *Git* e selezionare *Projects from Git -> Next -> Clone URI -> Next*.
- Utilizzare la URL del **proprio** repository che si vuole clonare (**non** quello in TdP-2016!), ad esempio:

  https://github.com/my-github-username/Lab2
- Fare click su *Next*. Selezionare il branch (*master* è quello di default) fare click su *Next*.
- Selezionare la cartella di destinazione (quella proposta va bene), fare click su *Next*.
- Selezionare *Import existing Eclipse projects*, fare click su *Next* e successivamente su *Finish*.
- Il nuovo progetto Eclipse è stato clonato ed è possibile iniziare a lavorare.
- A fine lavoro ricordarsi di effettuare Git commit e push, utilizzando il menù *Team in Eclipse*.

ATTENZIONE: solo se si effettua Git **commit** e successivamente Git **push** le modifiche locali saranno propagate sui server GitHub e saranno quindi accessibili da altri PC e dagli utenti che ne hanno visibilità.

03FYZ TECNICHE DI PROGRAMMAZIONE
Esercitazione di Laboratorio 2 – 15 Marzo 2017

Obiettivi dell'esercitazione:

- Introduzione all'utilizzo degli strumenti di sviluppo: Java, Eclipse, JavaFX, SceneBuilder
- Sperimentare la realizzazione di interfacce grafiche
- Java Collection Framework - List

Esercizio 1

Dopo anni di studi, alcuni scienziati sono riusciti a decifrare un linguaggio alieno trasmesso da un remoto pianeta. Per poter interpretare i messaggi che gli alieni hanno inviato alla terra nell'ultimo decennio, gli scienziati hanno chiesto di ideare un traduttore che possa aiutarli.

Realizzare in linguaggio Java una semplice applicazione dotata di interfaccia grafica che funga da traduttore di parole aliene. Deve essere possibile sia l'aggiunta di nuove parole che la ricerca di quelle esistenti.

Lo scopo del programma (Figura 1) è quello di permettere all'utente di:

- Inserire una nuova parola e la relativa traduzione secondo il seguente pattern:
  <parola aliena> <traduzione> (separate da uno spazio)
  Cliccando sul bottone *Translate* la parola viene aggiunta al dizionario.

- Cercare la traduzione di una parola esistente inserendo <parola aliena> e facendo click sul bottone *Translate*. La traduzione verrà visualizzata nell'area di testo sottostante.

Implementare i controlli per eventuali errori sull'input: gli unici caratteri ammessi sono [a-zA-Z] (ossia solo le lettere alfabetiche, siano essere maiuscole o minuscole), ma la ricerca deve essere *case insensitive*. Si suggerisce di convertire tutto il testo ricevuto in minuscolo prima di elaborarlo.

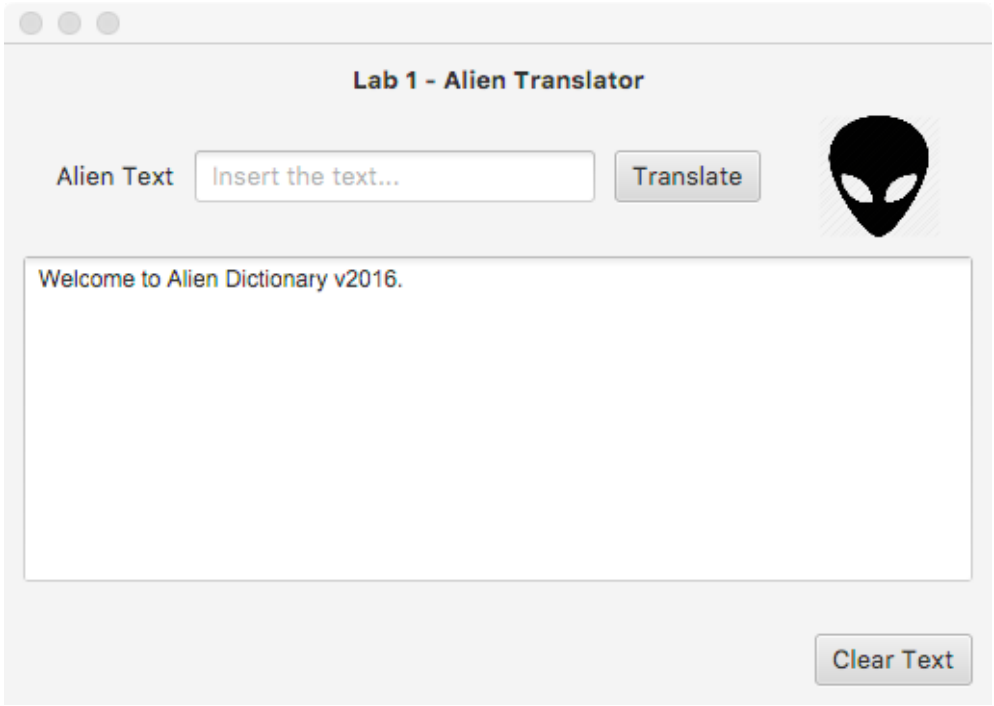


Figura 1

Di seguito viene proposta una traccia di soluzione:

Punto 1 Utilizzando la libreria *javafx* ed il tool *SceneBuilder*, creare un'interfaccia grafica simile a quella mostrata in Figura 1. Associare al bottone *Translate* un metodo *doTranslate()* ed al bottone *Clear* un metodo *doReset()*.

Punto 2 Nel package *it.polito.tdp.alien* creare una classe *Word* per memorizzare l'associazione tra parola aliena ed originale. Definire le seguenti variabili.

```
private String alienWord;
private String translation;
```

ed il metodo `compare` per controllare se la parola *alienWord* è già presente nel dizionario.

```
public String compare(String alienWord)
```

Punto 3 Nel package *it.polito.tdp.alien* creare una classe *AlienDictionary* con i seguenti metodi:

```
public void addWord(String alienWord, String translation)
```

Il metodo viene chiamato dal Controller per l'aggiunta, nel dizionario, di una nuova *alienWord* con corrispondente *translation*. Se *alienWord* è già presente, la traduzione deve essere aggiornata.

```
public String translateWord(String alienWord)
```

Il metodo viene chiamato dal Controller per la traduzione della parola *alienWord* passata come parametro. Il metodo restituisce la parola tradotta, altrimenti *null* se *alienWord* non è presente nel dizionario.

La classe *AlienDictionary* implementa il dizionario come una lista di oggetti di tipo *Word*.

Punto 4 Implementare, nel controller, il metodo *doTranslate()* e tutta la rimanente logica dell'applicazione necessaria al suo funzionamento. Effettuare alcuni test.

Esercizio 2

Partendo dalla soluzione del precedente esecizio, si vuole aggiungere al programma il supporto per traduzioni multiple associate a ciascuna parola aliena. Nel package *it.polito.tdp.alien* sostituire l'attuale classe *Word* con *WordEnhanced*. La classe è molto simile all'esistente *Word*, ma il metodo *addWord()* consente di salvare in una struttura dati (ad esempio una lista) eventuali traduzioni multiple di una parola aliena.

`Punto Opzionale:`
Implementare la ricerca di una parola con wildcard: quando nella parola aliena compare il simbolo "?", il carattere corrispondente può essere qualsiasi. È ammesso un solo "?" per ogni parola da cercare.

Esempio:
Se la traduzione della parola ALIENO corrisponde ad ANDREA, cercando ALI?NO si deve ottenere ANDREA. Fare attenzione al caso in cui più parole aliene soddisfino il criterio di ricerca (es. ALINNO).