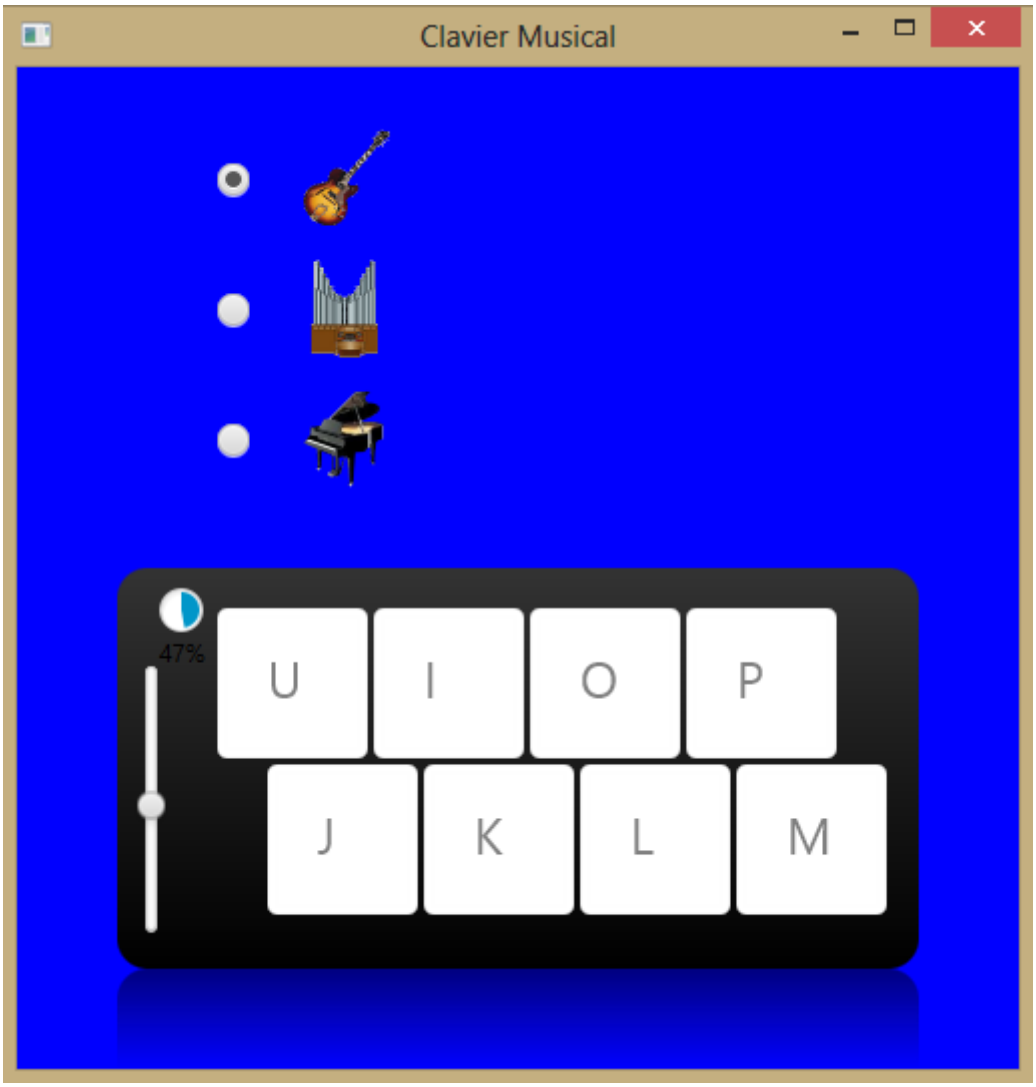
TP 5 : Interfaces Graphiques en JavaFx

Il s'agit de développer une application permettant de créer l'interface suivante :



Ce programme permettra à l'utilisateur de jouer de la musique en appuyant sur les touches de son clavier. Les fonctionnalités à développer sont les suivantes:

- ❖ Émettre un son à chaque fois que l'utilisateur appuie sur l'une des touches :
  - ✓ Quand l'utilisateur appuierait sur le U l'application jouerait un Do, quand il appuierait sur le I un Ré, sur le O un Mi, sur le P un Fa, sur le J un Sol, sur le K un La, sur le L un Si et sur le M un Do.
  - ✓ Quand l'utilisateur appuierait sur une touche de son clavier ou cliquerait sur une touche, celle-ci descend de quelques pixels pour donner l'impression de s'enfoncer et changerait de couleur.

- Pouvoir choisir son instrument entre le piano, la guitare et l'orgue. L'utilisateur peut changer d'instrument. il a le choix entre trois instruments.
- Régler le volume du son de la musique jouée

Ce programme aura la structure suivante :

- La classe principale **Melordi** qui contiendra la fonction main().
- La classe **Instru** qui contiendra toutes les fonctions sonores : jouer une note, augmenter ou baisser le volume, changer l'instrument. Cette classe ne correspondra à aucun élément graphique.
- La classe **Touche** qui modélisera une touche du clavier graphique.
- La classe **Clavier** qui modélisera l'ensemble du clavier graphique et qui contiendra notamment huit objets de type Touche, un par touche.
- La classe **ChangeInstru** qui contiendra la liste des boutons radio des trois instruments et qui permettra de changer de type d'instrument.
- La classe **Son** qui contiendra le slider permettant de régler le volume sonore. Le pourcentage de l'indicateur varie bien avec la valeur du slider.

On vous fournit le code de la classe Instru vu ses spécificités musicales:

```
import java.util.logging.Level;
import java.util.logging.Logger;
import javafx.beans.InvalidationListener;
import javafx.beans.property.IntegerProperty;
import javafx.beans.value.ChangeListener;
import javafx.beans.value.ObservableValue;
import javax.sound.midi.MidiSystem;
import javax.sound.midi.MidiChannel;
import javax.sound.midi.MidiUnavailableException;
import javax.sound.midi.Synthesizer;
public class Instru {
        public int volume = 500;
        private Synthesizer synthetiseur;
        private MidiChannel canal;

           public Instru(){
        try {
           //On récupère le synthétiseur, on l'ouvre et on obtient un
canal
            synthetiseur = MidiSystem.getSynthesizer();
            synthetiseur.open();
            } catch (MidiUnavailableException ex) {

Logger.getLogger(Instru.class.getName()).log(Level.SEVERE, null,
ex);
           }
        canal = synthetiseur.getChannels()[0];
        //On initialise l'instrument 0 (le piano) pour le canal
         canal.programChange(0);  } // fin constructeur
```

```
            //Jouer la note dont le numéro est en paramètre
           public void note_on(int note){
             canal.noteOn(note, volume);    }
//Arrête de jouer la note dont le numéro est en paramètre
          public void note_off(int note){
             canal.noteOff(note);    }
 //changer le type d'instrument dont le numéro MIDI est précisé en
paramètre
          public void set_instrument(int instru){
             canal.programChange(instru);    }         }
```

Questions:

1) Compléter la classe ChangeInstru qui représente la partie supérieure de l'interface graphique :

```
public class ChangeInstru extends Parent{
 private RadioButton  rb_piano;
private RadioButton  rb_guitare;
 private RadioButton  rb_orgue;
 private Instru instru;
   public ChangeInstru(Instru ins){
        instru = ins;
       GridPane gridpane = new GridPane();

       //création des images des 3 instruments
ImageView piano = new ImageView(new
Image(ChangeInstru.class.getResourceAsStream("images/327153.png")));
piano.setFitHeight(50);
piano.setPreserveRatio(true);
ImageView guitare = new ImageView(new

       // Compléter
       //ajout d'un ChangeListener au groupe de boutons radio
       groupe.selectedToggleProperty().addListener(new ChangeListener()
{
          public void changed(ObservableValue observable, Object
oldValue, Object newValue) {
               // numéro MIDI du piano = 0, numéro MIDI de la guitare = 26
                //numéro MIDI de l'orgue = 16
// compléter      }   });

       //on ajoute nos images à notre layout
       //on ajoute les boutons radio au layout
       // positionnement du gridlayout
   }}
```

2) Compléter la classe **Touche** puis **Clavier :**

```
public class Touche extends Parent {
   public String lettre = new String("X");
   private int positionX = 0;
```

```
    private int positionY = 0;
    private int note = 0;
    private Instru instru;//on déclare un objet de type Instru
    Rectangle fond_touche = new Rectangle(75,75,Color.WHITE);
    Text lettre_touche ;
public Touche(String l, int posX, int posY, int n,  Instru ins){
// compléter
this.setOnMouseEntered(new EventHandler<MouseEvent>(){//
compléter}
this.setOnMouseExited(new EventHandler<MouseEvent>(){ //
compléter}
this.setOnMousePressed(new EventHandler<MouseEvent>(){//
completer}
 this.setOnMouseReleased(new EventHandler<MouseEvent>(){//
completer} }
// fin constructeur
}

public class Clavier extends Parent{
      private Touche[] touches;
      private Instru instru;
    public Clavier(Instru ins){
   Rectangle fond_clavier = new Rectangle();
        fond_clavier.setWidth(400);
        fond_clavier.setHeight(200);
        fond_clavier.setArcWidth(30);
        fond_clavier.setArcHeight(30);
        fond_clavier.setFill( //on remplie notre rectangle avec un dégradé
            new LinearGradient(0f, 0f, 0f, 1f, true, CycleMethod.NO_CYCLE,
                new Stop[] {
                    new Stop(0, Color.web("#333333")),
                    new Stop(1, Color.web("#000000"))     }           )       );

Reflection r = new Reflection();//on applique un effet de réflection
        r.setFraction(0.25);
        r.setBottomOpacity(0);
        r.setTopOpacity(0.5);
        fond_clavier.setEffect(r);
// compléter
this.setOnKeyPressed(new EventHandler<KeyEvent>(){// completer}
this.setOnKeyReleased(new EventHandler<KeyEvent>(){// completer}}}
// fin constrcuteur
}
```

3) Développer le reste des classes et Tester toutes les fonctionnalités.