package hw3;

import javafx.application.Application;

import javafx.collections.ObservableList;

import javafx.event.ActionEvent;

import javafx.event.EventHandler;

import javafx.fxml.FXMLLoader;

import javafx.geometry.Insets;

import javafx.geometry.Pos;

import javafx.scene.Node;

import javafx.scene.Parent;

import javafx.scene.Scene;

import javafx.scene.control.Button;

import javafx.scene.control.ToolBar;

import javafx.scene.image.Image;

import javafx.scene.image.ImageView;

import javafx.scene.input.KeyEvent;

import javafx.scene.layout.\*;

import javafx.scene.text.Font;

import javafx.scene.text.Text;

import javafx.scene.text.TextAlignment;

import javafx.stage.Stage;

import java.io.BufferedReader;

import java.io.File;

import java.io.FileReader;

import java.util.Random;

public class Main extends Application {

public Node getNodeByRowColumnIndex (final int row, final int column, GridPane gridPane) {

Node result = null;

ObservableList<Node> childrens = gridPane.getChildren();

for (Node node : childrens) {

if(gridPane.getRowIndex(node) == row && gridPane.getColumnIndex(node) == column) {

result = node;

break;

}

}

return result;

}

@Override

public void start(Stage primaryStage) throws Exception{

Random rand = new Random();

File file = new File("words.txt");

StringBuffer contents = new StringBuffer();

BufferedReader reader = null;

int lineCounter = 0;

reader = new BufferedReader(new FileReader(file));

String word = null;

while ((word = reader.readLine()) != null) {

lineCounter++;

}

reader = new BufferedReader(new FileReader(file));

int randomNum = rand.nextInt(lineCounter) - 1;

System.out.println(randomNum);

while(randomNum >= 0){

word = reader.readLine();

randomNum--;

}

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*\*" + word);

reader.close();

System.out.println(contents.toString());

primaryStage.setTitle("Hangman");

BorderPane border = new BorderPane();

BorderPane insideBorder = new BorderPane();

VBox right = new VBox();

right.setSpacing(12);

ToolBar toolBar = new ToolBar();

Button startB = new Button();

Image startI = new Image("resources/new.png");

startB.setGraphic(new ImageView(startI));

toolBar.getItems().add(startB);

Button loadB = new Button();

Image loadI = new Image("resources/load.png");

loadB.setGraphic(new ImageView(loadI));

toolBar.getItems().add(loadB);

Button saveB = new Button();

Image saveI = new Image("resources/save.png");

saveB.setGraphic(new ImageView(saveI));

toolBar.getItems().add(saveB);

saveB.setDisable(true);

Button exitB = new Button();

Image exitI = new Image("resources/exit.png");

exitB.setGraphic(new ImageView(exitI));

toolBar.getItems().add(exitB);

HBox hbStartPlayingB = new HBox();

Button startPlayingB = new Button("Start Playing");

hbStartPlayingB.getChildren().add(startPlayingB);

hbStartPlayingB.setAlignment(Pos.CENTER);

startPlayingB.setVisible(false);

final int[] intRG = {10};

final Text[] textRG = {new Text("Remaining Guesses: " + intRG[0])};

HBox theWords = new HBox();

int lengthCounter = 0;

int paddingSize = 13;

int theLength = word.length();

if(theLength > 10){

paddingSize = 5;

}

while(word.length() > lengthCounter){

Button aWord = new Button(" ");

aWord.setPadding(new Insets(paddingSize)); //10 px "buffer" around button

theWords.getChildren().add(aWord);

lengthCounter++;

}

//add the right amount of the buttons

GridPane alphabets = new GridPane();

int a = 65;

for(int i = 0; i < 26; i++) {

for (int j = 0; j < 4; j++) {

for (int k = 0; k < 7; k++) {

char alphabet = (char) (a + i);

Button alphabetB = new Button("" + alphabet);

alphabetB.setPadding(new Insets(20)); //10 px "buffer" around button

alphabets.add(alphabetB, k, j);

i++;

if(i >= 26){

break;

}

}

}

}

right.getChildren().addAll(textRG[0],theWords, alphabets);

String finalWord = word;

final int[] winCounter = {finalWord.length()};

EventHandler<KeyEvent> theKeyEvent = new EventHandler<KeyEvent>() {

@Override

public void handle(KeyEvent event) {

String choose = event.getCode().getName();

int chooseInt = choose.charAt(0);

chooseInt = chooseInt + 32;

char chooseChar = (char) chooseInt;

int theIndex = finalWord.indexOf(chooseChar);

System.out.println("\*\*\*\*\*\*this Key Pressed\*\*\*\*\*\*\*\*" + chooseInt + "\*\*\*\*\*\*\*\*" + theIndex);

if(theIndex >= 0){

chooseInt = chooseInt - 97;

int row = 0;

int col = chooseInt;

if(chooseInt > 6){

row = chooseInt / 7;

col = chooseInt % 7;

}

getNodeByRowColumnIndex(row, col, alphabets).setDisable(true);

System.out.println("\*\*\*\*\*\*\*\*\*\*\*\*" + finalWord.charAt(theIndex));

theWords.getChildren().set(theIndex, new Button(chooseChar + ""));

winCounter[0]--;

System.out.println("WIN COUNTER " + winCounter[0]);

while (theIndex >= 0) {

System.out.println(theIndex);

int theIndex2 = finalWord.indexOf(chooseChar, theIndex + 1);

if(theIndex2 >= 0) {

theWords.getChildren().set(theIndex2, new Button(chooseChar + ""));

winCounter[0]--;

System.out.println("WIN COUNTER " + winCounter[0]);

}

theIndex = theIndex2;

if(winCounter[0] == 0){

Stage winStage = new Stage();

winStage.setTitle("Game Over");

VBox winBox = new VBox();

Text winText = new Text("You won!");

Button winButton = new Button("CLOSE");

winButton.setOnAction(new EventHandler<ActionEvent>() {

@Override

public void handle(ActionEvent event) {

winStage.close();

}

});

winBox.getChildren().addAll(winText, winButton);

winBox.setAlignment(Pos.CENTER);

Scene winScene = new Scene(winBox, 300, 200);

winStage.setScene(winScene);

winStage.show();

}

}

}

else{

intRG[0]--;

textRG[0].setText("Remaining Guesses: " + intRG[0]);

if(intRG[0] == 0){

Stage loseStage = new Stage();

loseStage.setTitle("Game Over");

VBox loseBox = new VBox();

Text loseText = new Text("You lost (the word was \"" + finalWord + "\")");

Button loseButton = new Button("CLOSE");

loseButton.setOnAction(new EventHandler<ActionEvent>() {

@Override

public void handle(ActionEvent event) {

loseStage.close();

}

});

loseBox.getChildren().addAll(loseText, loseButton);

loseBox.setAlignment(Pos.CENTER);

Scene loseScene = new Scene(loseBox, 300, 200);

loseStage.setScene(loseScene);

loseStage.show();

}

}

}

};

EventHandler<ActionEvent> startHandler = new EventHandler<ActionEvent>(){

public void handle(ActionEvent e){

startPlayingB.setVisible(true);

Text text = new Text("Hangman");

text.setFont(Font.font(30));

insideBorder.setTop(text);

insideBorder.setAlignment(text, Pos.CENTER);

insideBorder.setPadding(new Insets(13));

}

};

startB.setOnAction(startHandler);

EventHandler<ActionEvent> startPlayingHandler = new EventHandler<ActionEvent>() {

@Override

public void handle(ActionEvent event) {

saveB.setDisable(true);

insideBorder.setRight(right);

startPlayingB.setDisable(true);

border.setOnKeyPressed(theKeyEvent);

}

};

startPlayingB.setOnAction(startPlayingHandler);

border.setTop(toolBar);

border.setBottom(hbStartPlayingB);

border.setCenter(insideBorder);

Scene scene = new Scene(border, 700, 500);

primaryStage.setScene(scene);

primaryStage.show();

}

public static void main(String[] args) {

launch(args);

}

}

Button[] alphabet = new Button[26];

for(int i = 0; i < alphabet.length; i++) {

alphabet[i] = new Button(Character.toString((char)(65+i)));

alphabet[i].setStyle("-fx-base: chartreuse; -fx-border-color:white;");

alphabet[i].setMinWidth(55);

alphabet[i].setMinHeight(55);

grid.getChildren().add(alphabet[i]);

} //making alphabet grid

scene.setOnKeyPressed((KeyEvent event) ->{

for(int i = 0; i<alphabet.length; i++) {

if(event.getText().toUpperCase().equals(alphabet[i].getText())) {

alphabet[i].setDisable(true);

}

}