Projet personnel réalisé en cours de formation :

Développement d'une application Java orientée objet, avec interface graphique, jeu du morpion.

Sommaire :

I. Introduction

Le morpion (Tic-Tac-Toe) est un jeu simple mais idéal pour illustrer les bases de la programmation orientée objet en Java :

- Gestion d'un tableau
- Interactions utilisateur
- Conditions
- Boucles
- Modularité du code

Objectif : Développer une application Java permettant à deux joueurs de s'affronter, en assurant le choix du mode de jeux, l'affichage de la grille, la saisie des coups, la vérification des conditions de victoire, la détection des égalités et permettre de rejouer une partie et revenir au menu.

II. Analyse du besoin

2.1 Règles du jeu

Le morpion se joue sur une grille de 3×3. Deux joueurs jouent à tour de rôle :

- Joueur 1 → symbole X
- Joueur 2 → symbole O

Le premier joueur à aligner 3 symboles identiques (ligne, colonne ou diagonale) gagne la partie.

Il existe deux modes de jeux : 1vs1 ou 1vsOrdinateur.

2.2 Fonctionnalités attendues

L'application doit :

- Permettre au joueur de choisir le mode de jeux
- Afficher dynamiquement de la grille en conséquence
- Saisir les coups en vérifiant et gérant leur validité
- Détecter automatiquement une victoire ou une égalité
- Donner la possibilité de rejouer une partie et de revenir au menu

III. Développement JAVA

3.1 Structure du code

Dans la classe MORPION, on commence par créer les éléments de notre morpion : la grille, tous les boutons, les zones de textes, et les variables de gestion.

```
public class Morpion extends JFrame implements ActionListener {

    boolean tourX = true;
    boolean ChoixModeJeux = true;
    boolean victoire = false;

    JButton case1 = new JButton();
    JButton case2 = new JButton();
    JButton case3 = new JButton();
    JButton case4 = new JButton();
    JButton case5 = new JButton();
    JButton case6 = new JButton();
    JButton case7 = new JButton();
    JButton case8 = new JButton();
    JButton case9 = new JButton();

    JButton rejouer = new JButton("Rejouer");
    JButton menu = new JButton("Menu");

    JButton oneVSone = new JButton("1vs1");
    JButton oneVSordinateur = new JButton("ordinateur");

    JLabel titreJeux = new JLabel("Bienvenue sur mon jeux du morpion !");
    JLabel resultatJeux = new JLabel("Qui sera le gagnant ?");
    JLabel modeJeux = new JLabel("Choississez votre mode de jeux : \n");
```

Ensuite on met en place le constructeur :

```
Morpion(){
    this.setVisible(true);
    this.setTitle("Morpion");
    this.setBounds(700,100,600,600);
    this.setLayout(null);
    this.setResizable(false);
    this.getContentPane().setBackground(new Color(234,67,53));


    this.titreJeux.setBounds(190,50,250,50);
    this.titreJeux.setForeground(Color.BLACK);
    this.add(this.titreJeux);

    this.modeJeux.setBounds(200,150,250,50);
    this.modeJeux.setForeground(Color.BLACK);
    this.add(this.modeJeux);

    this.oneVSone.setBounds(200,200,70,50);
    this.add(this.oneVSone);
    this.oneVSone.addActionListener(this);

    this.oneVSordinateur.setBounds(280,200,100,50);
    this.add(this.oneVSordinateur);
    this.oneVSordinateur.addActionListener(this);
```

```
    this.case1.setBounds(150,150,100,100);
    this.case1.setBackground(Color.white);
    this.case2.setBounds(150,250,100,100);
    this.case2.setBackground(Color.white);
    this.case3.setBounds(150,350,100,100);
    this.case3.setBackground(Color.white);
    this.case4.setBounds(250,150,100,100);
    this.case4.setBackground(Color.white);
    this.case5.setBounds(250,250,100,100);
    this.case5.setBackground(Color.white);
    this.case6.setBounds(250,350,100,100);
    this.case6.setBackground(Color.white);
    this.case7.setBounds(350,150,100,100);
    this.case7.setBackground(Color.white);
    this.case8.setBounds(350,250,100,100);
    this.case8.setBackground(Color.white);
    this.case9.setBounds(350,350,100,100);
    this.case9.setBackground(Color.white);
    this.resultatJeux.setBounds(250,450,250,50);
    this.rejouer.setBounds(150,500,140,50);
    this.rejouer.setBackground(Color.BLACK);
    this.rejouer.setForeground(Color.WHITE);
    this.menu.setBounds(310,500,140,50);
    this.menu.setBackground(Color.BLACK);
    this.menu.setForeground(Color.WHITE);
}
```

On peut désormais passer à la logique de l'application !

On va d'abord faire la gestion du choix de mode de jeux : Lorsque le joueur choisi un mode de jeux, les boutons et le texte pour choisir le mode de jeux disparaissent et laissent place a la grille et aux boutons de retour au menu et de réinitialisation de la partie.

```
@Override
public void actionPerformed(ActionEvent e) {
    // TODO Auto-generated method stub


    //Gestion choix de jeux
        if(e.getSource() == this.oneVSone || e.getSource() == this.oneVSordinateur) {

            this.remove(this.oneVSone);
            this.remove(this.oneVSordinateur);
            this.remove(this.modeJeux);

            this.add(this.case1);
            this.add(this.case2);
            this.add(this.case3);
            this.add(this.case4);
            this.add(this.case5);
            this.add(this.case6);
            this.add(this.case7);
            this.add(this.case8);
            this.add(this.case9);
            this.add(this.resultatJeux);
            this.add(this.rejouer);
            this.add(this.menu);
```

```
    this.case1.addActionListener(this);
    this.case2.addActionListener(this);
    this.case3.addActionListener(this);
    this.case4.addActionListener(this);
    this.case5.addActionListener(this);
    this.case6.addActionListener(this);
    this.case7.addActionListener(this);
    this.case8.addActionListener(this);
    this.case9.addActionListener(this);

    this.rejouer.addActionListener(this);
    this.rejouer.setEnabled(true);

    this.menu.addActionListener(this);

    tourX = true;

    this.revalidate();
    this.repaint();
}
```

Ensuite, on met en place la gestion des tours pour qu'un joueur ne puisse pas jouer plusieurs coups d'affilé, mais qu'il y ait une alternance avec l'adversaire.

```
//Gestion des click
JButton click = (JButton) e.getSource();
JButton[] boutons = {this.case1,this.case2,this.case3,this.case4,this.case5, this.case6,this.case7,this.case8,this.case9};
List<JButton> btns = new ArrayList<>(Arrays.asList(boutons));

if(click.getText().equals("")) {
    if(tourX == true) {
        click.setText("X");
    }else {
        click.setText("O");
    }

    tourX = !tourX;
}
```

On gère également le comportement de l'ordinateur pour le mode de jeux 1vsOrdinateur :

```
    //Gestion IA
    Random random = new Random();
    int i = random.nextInt(btns.size());

    if(e.getSource() == this.oneVSordinateur) {
        this.ChoixModeJeux = false;
    }else if(e.getSource() == this.oneVSone) {
        this.ChoixModeJeux = true;
    }

    if (!ChoixModeJeux && !tourX && !victoire) {
        List<JButton> libres = btns.stream().filter(b -> b.getText().equals("") && b.isEnabled()).toList();

        if (!libres.isEmpty()) {
            JButton choisi = libres.get(new Random().nextInt(libres.size()));
            choisi.doClick();
            btns.remove(choisi);
        }
    }
```

Enfin, on n'oublie pas la gestion des cas ou il y aurait match nul :

```
//Gestion match nul
        if(
            (case1.getText().equals("X") || case1.getText().equals("O"))
             && (case2.getText().equals("X") || case2.getText().equals("O"))
             && (case3.getText().equals("X") || case3.getText().equals("O"))
             && (case4.getText().equals("X") || case4.getText().equals("O"))
             && (case5.getText().equals("X") || case5.getText().equals("O"))
             && (case6.getText().equals("X") || case6.getText().equals("O"))
             && (case7.getText().equals("X") || case7.getText().equals("O"))
             && (case8.getText().equals("X") || case8.getText().equals("O"))
             && (case9.getText().equals("X") || case9.getText().equals("O"))
                ) {
                    for(JButton b : boutons) {
                        b.setEnabled(false);
                    }
                    this.resultatJeux.setBounds(280,450,250,50);
                    this.resultatJeux.setText("Egalité !");
                    this.resultatJeux.setForeground(Color.YELLOW);

                }
```

On peut désormais passer au calcul des cas de victoire !

Pour cela on va vérifier pour chaque joueur s'il a 3 cases alignés contenant le symbole inhérent selon qu'il soit le joueur 1 ou le joueur 2.

Joueur 1 :

```
//Joueur1
if(case1.getText().equals("X") && case2.getText().equals("X") && case3.getText().equals("X")) {
    victoire = true;
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
    case1.setBackground(Color.YELLOW);
    case2.setBackground(Color.YELLOW);
    case3.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
    this.resultatJeux.setForeground(Color.YELLOW);


} else if (case1.getText().equals("X") && case4.getText().equals("X") && case7.getText().equals("X")) {
    victoire = true;
    case1.setBackground(Color.YELLOW);
    case4.setBackground(Color.YELLOW);
    case7.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
} else if(case7.getText().equals("X") && case8.getText().equals("X") && case9.getText().equals("X")) {
    victoire = true;
    case7.setBackground(Color.YELLOW);
    case8.setBackground(Color.YELLOW);
    case9.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }

} else  if (case3.getText().equals("X") && case6.getText().equals("X") && case9.getText().equals("X")) {
    victoire = true;
    case3.setBackground(Color.YELLOW);
    case6.setBackground(Color.YELLOW);
    case9.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
} else if(case2.getText().equals("X") && case5.getText().equals("X") && case8.getText().equals("X")) {
    victoire = true;
    case2.setBackground(Color.YELLOW);
    case5.setBackground(Color.YELLOW);
    case8.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }

}else if(case1.getText().equals("X") && case5.getText().equals("X") && case9.getText().equals("X")) {
    victoire = true;
    case1.setBackground(Color.YELLOW);
    case5.setBackground(Color.YELLOW);
    case9.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
 }else if(case7.getText().equals("X") && case5.getText().equals("X") && case3.getText().equals("X")) {
     victoire = true;
     case7.setBackground(Color.YELLOW);
     case5.setBackground(Color.YELLOW);
     case3.setBackground(Color.YELLOW);
     this.resultatJeux.setBounds(200,450,250,50);
     this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
     this.resultatJeux.setForeground(Color.YELLOW);
     for(JButton b : boutons) {
         b.setEnabled(false);
     }
 }else if(case4.getText().equals("X") && case5.getText().equals("X") && case6.getText().equals("X")) {
     victoire = true;
     case4.setBackground(Color.YELLOW);
     case5.setBackground(Color.YELLOW);
     case6.setBackground(Color.YELLOW);
     this.resultatJeux.setBounds(200,450,250,50);
     this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 1 !");
     this.resultatJeux.setForeground(Color.YELLOW);
     for(JButton b : boutons) {
         b.setEnabled(false);
     }
 }
```

Joueur 2 :

```
//Joueur2
if(case1.getText().equals("O") && case2.getText().equals("O") && case3.getText().equals("O")) {
    victoire = true;
    case1.setBackground(Color.YELLOW);
    case2.setBackground(Color.YELLOW);
    case3.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
} else if (case1.getText().equals("O") && case4.getText().equals("O") && case7.getText().equals("O")) {
    victoire = true;
    case1.setBackground(Color.YELLOW);
    case4.setBackground(Color.YELLOW);
    case7.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
} else if(case7.getText().equals("O") && case8.getText().equals("O") && case9.getText().equals("O")) {
    victoire = true;
    case7.setBackground(Color.YELLOW);
    case8.setBackground(Color.YELLOW);
    case9.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
} else  if (case3.getText().equals("O") && case6.getText().equals("O") && case9.getText().equals("O")) {
    victoire = true;
    case3.setBackground(Color.YELLOW);
    case6.setBackground(Color.YELLOW);
    case9.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
} else if(case2.getText().equals("O") && case5.getText().equals("O") && case8.getText().equals("O")) {
    victoire = true;
    case2.setBackground(Color.YELLOW);
    case5.setBackground(Color.YELLOW);
    case8.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
}else if(case1.getText().equals("O") && case5.getText().equals("O") && case9.getText().equals("O")) {
    victoire = true;
    case1.setBackground(Color.YELLOW);
    case5.setBackground(Color.YELLOW);
    case9.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
}else if(case7.getText().equals("O") && case5.getText().equals("O") && case3.getText().equals("O")) {
    victoire = true;
    case7.setBackground(Color.YELLOW);
    case5.setBackground(Color.YELLOW);
    case3.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
```

```
}else if(case4.getText().equals("O") && case5.getText().equals("O") && case6.getText().equals("O")) {
    victoire = true;
    case4.setBackground(Color.YELLOW);
    case5.setBackground(Color.YELLOW);
    case6.setBackground(Color.YELLOW);
    this.resultatJeux.setBounds(200,450,250,50);
    if(ChoixModeJeux == false) {
        this.resultatJeux.setText("Le gagnant de la partie est l'ordinateur !");
    }else {
        this.resultatJeux.setText("Le gagnant de la partie est le JOUEUR 2 !");
    }
    this.resultatJeux.setForeground(Color.YELLOW);
    for(JButton b : boutons) {
        b.setEnabled(false);
    }
}
```

Enfin, il nous reste plus que le retour au menu et la réinitialisation de la partie :

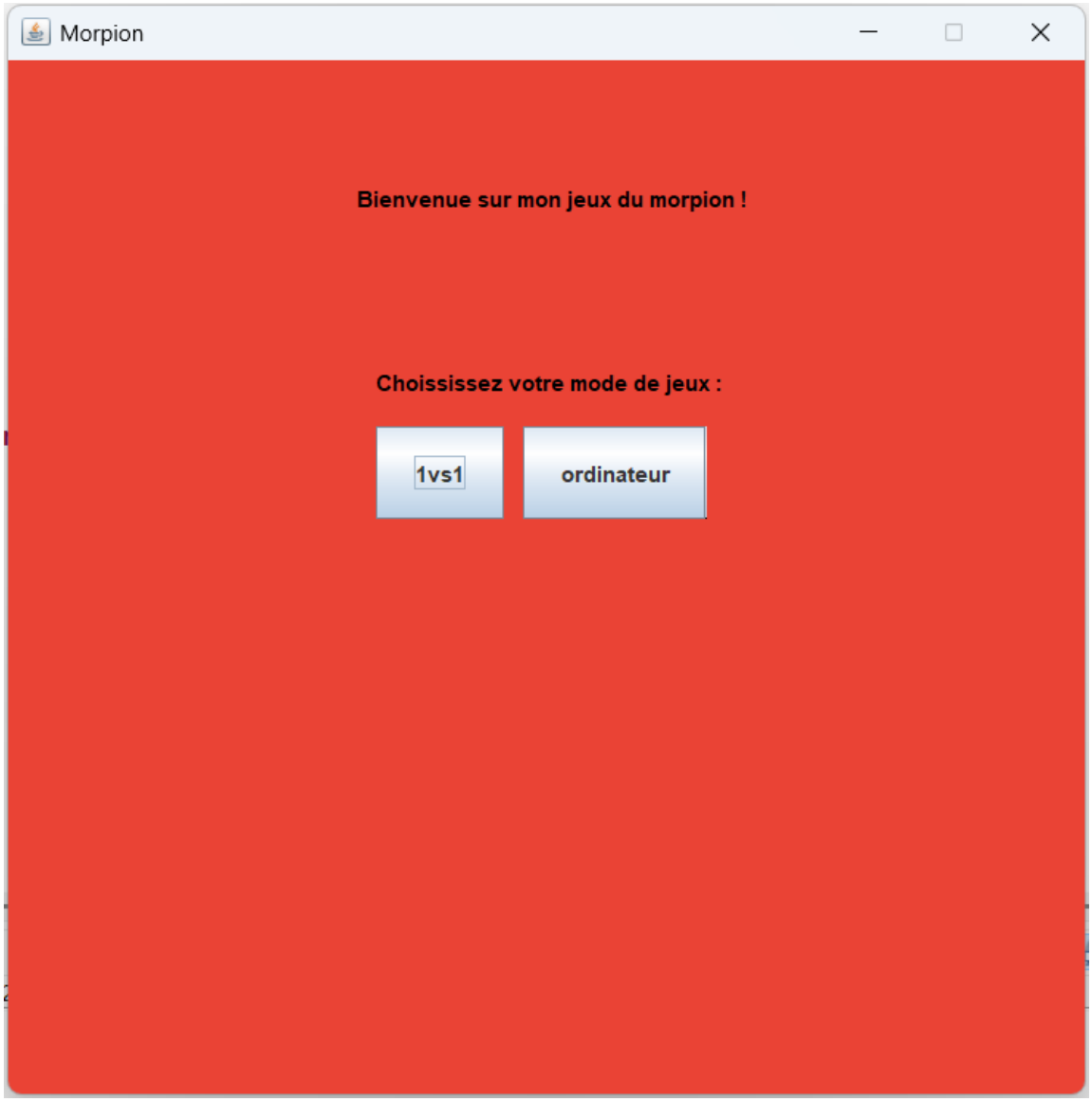
```
//Gestion rejouer
if(e.getSource() == this.rejouer) {
    this.resultatJeux.setText("Qui sera le gagnant ?");
    this.resultatJeux.setForeground(Color.BLACK);
    this.resultatJeux.setBounds(250,450,250,50);
    tourX = true;
    victoire = false;

    for(JButton b : boutons) {
        b.setText("");
        b.setEnabled(true);
        b.setBackground(Color.WHITE);
    }
}
```

```
//Gestion retour menu
if(e.getSource() == this.menu) {
    this.remove(this.rejouer);
    this.remove(this.menu);
    this.remove(this.resultatJeux);
    this.remove(this.case1);
    this.remove(this.case2);
    this.remove(this.case3);
    this.remove(this.case4);
    this.remove(this.case5);
    this.remove(this.case6);
    this.remove(this.case7);
    this.remove(this.case8);
    this.remove(this.case9);

    this.add(this.modeJeux);
    this.add(this.oneVSone);
    this.add(this.oneVSordinateur);

    this.revalidate();
    this.repaint();
}
```

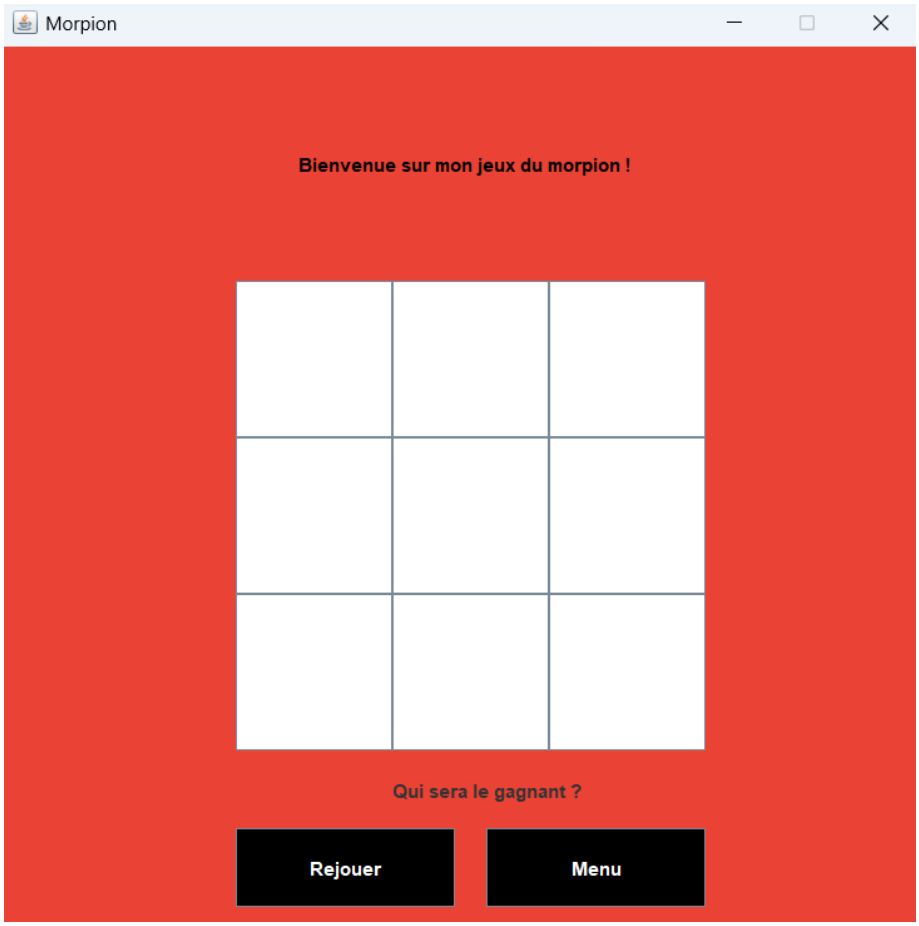
3.2 Interface utilisateur

Observons maintenant le résultat !

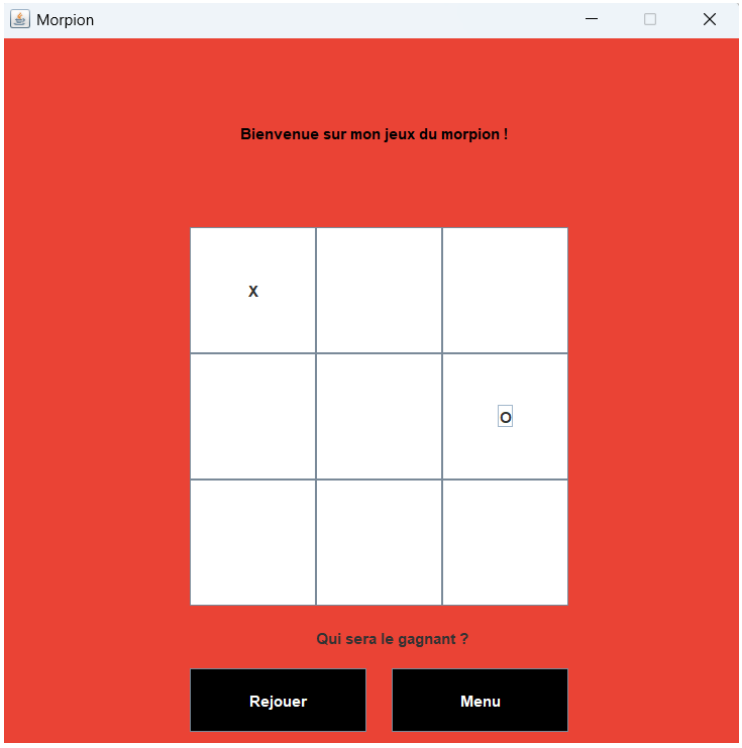
Au lancement, une fenêtre s'ouvre avec un message de bienvenue, le joueur est invité à faire un choix entre les deux modes de jeux en cliquant dessus :



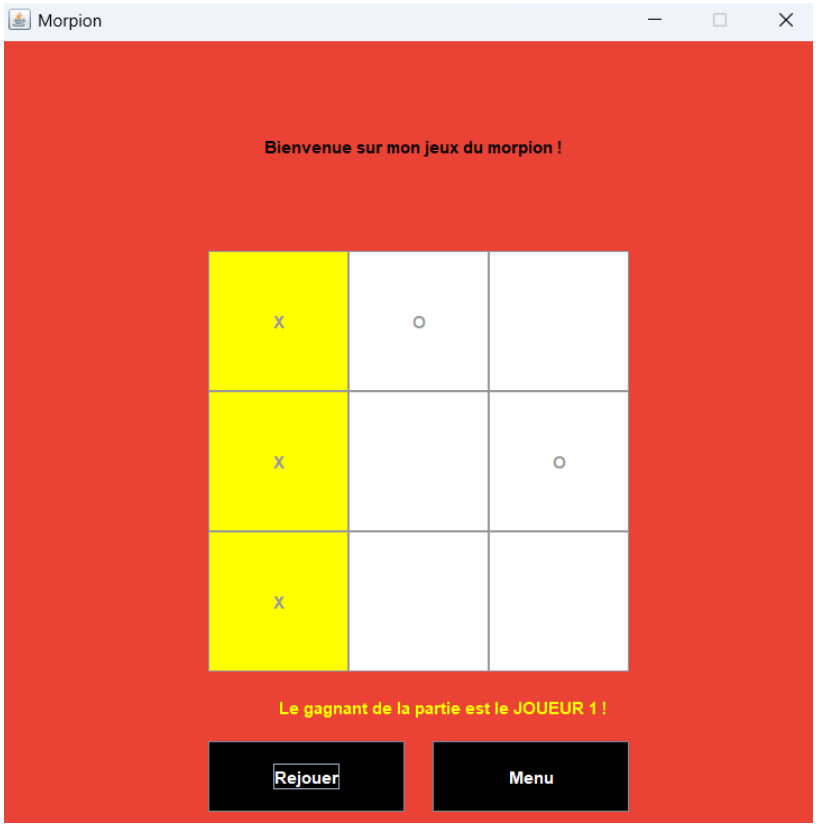
Une fois son choix fait, la partie est lancée :

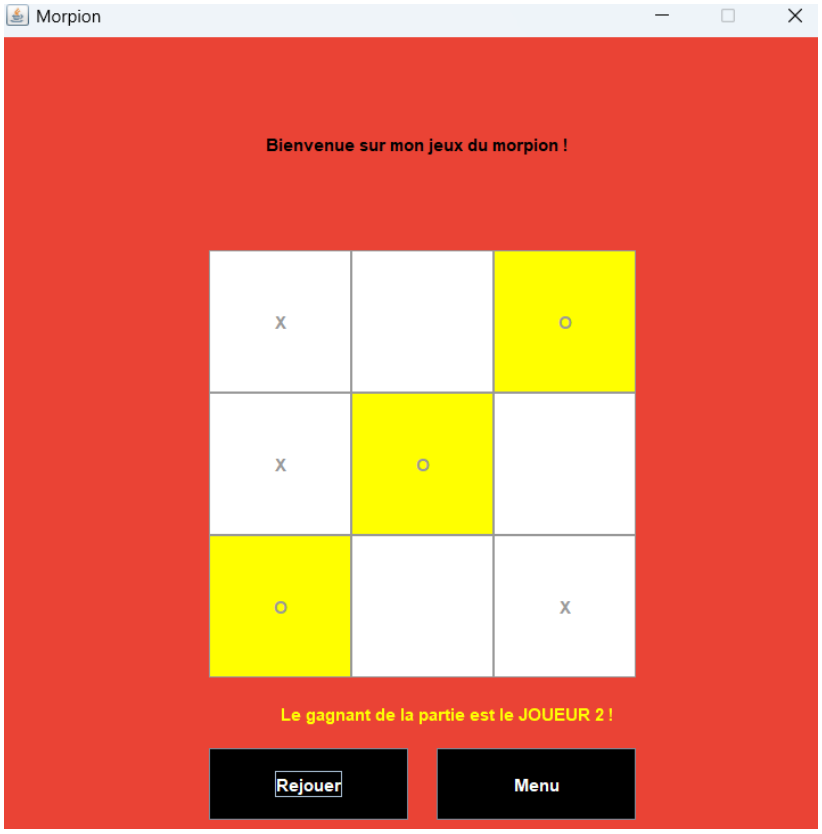


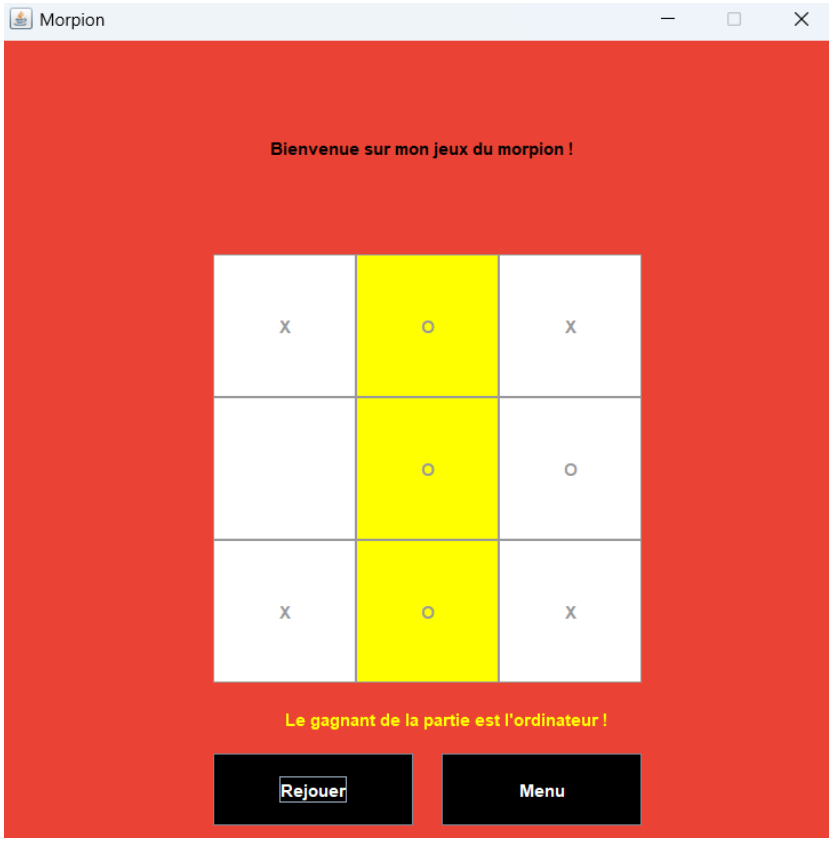
Les joueurs cliquent sur les cases de leur choix qui sont libres, au joueur 1 est attribué les signe « **X** » et au joueur 2 est attribué le signe « **O** » :



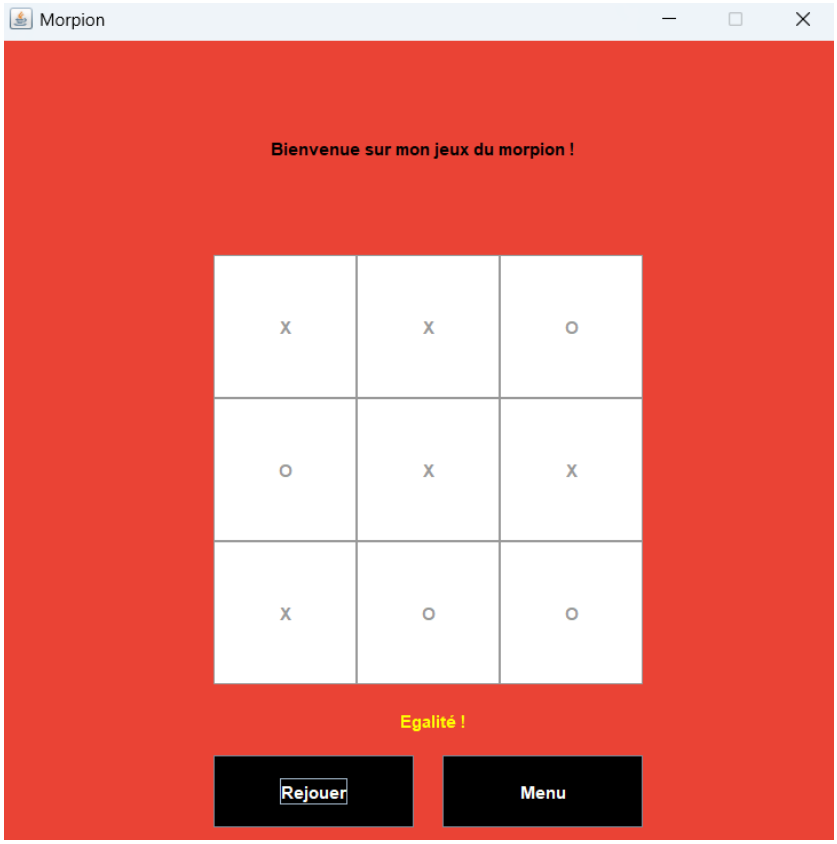
Si un joueur arrive à aligner trois de ses symboles, il remporte la partie. Touts les boutons de la grille sont désactivés, les cases gagnantes passent à la couleur jaune et un message de la même couleur s'affiche désignant le gagnant de la partie.



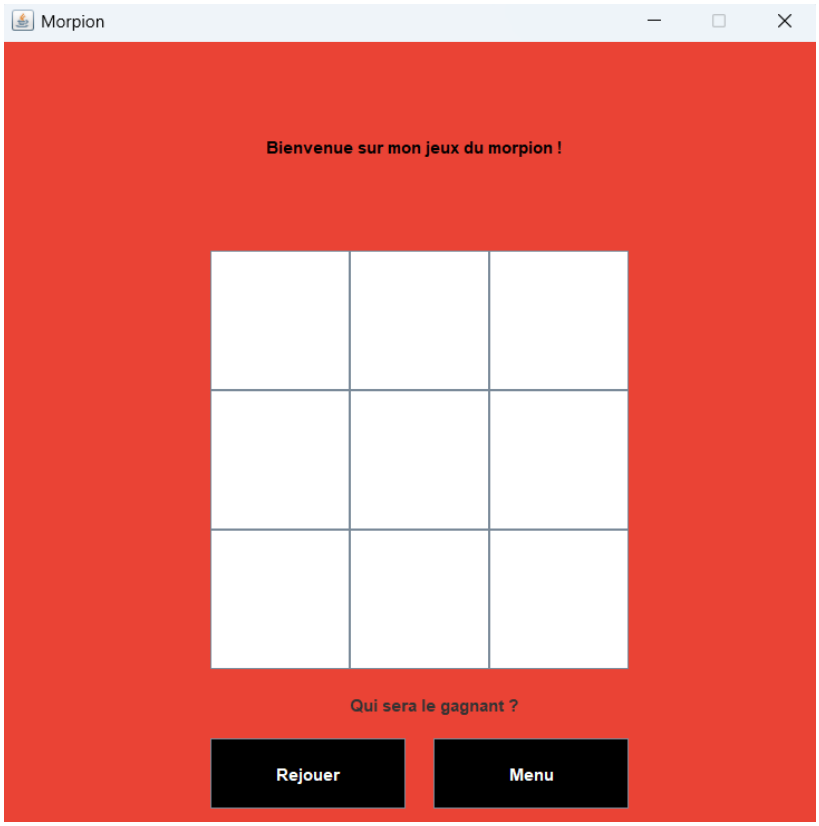




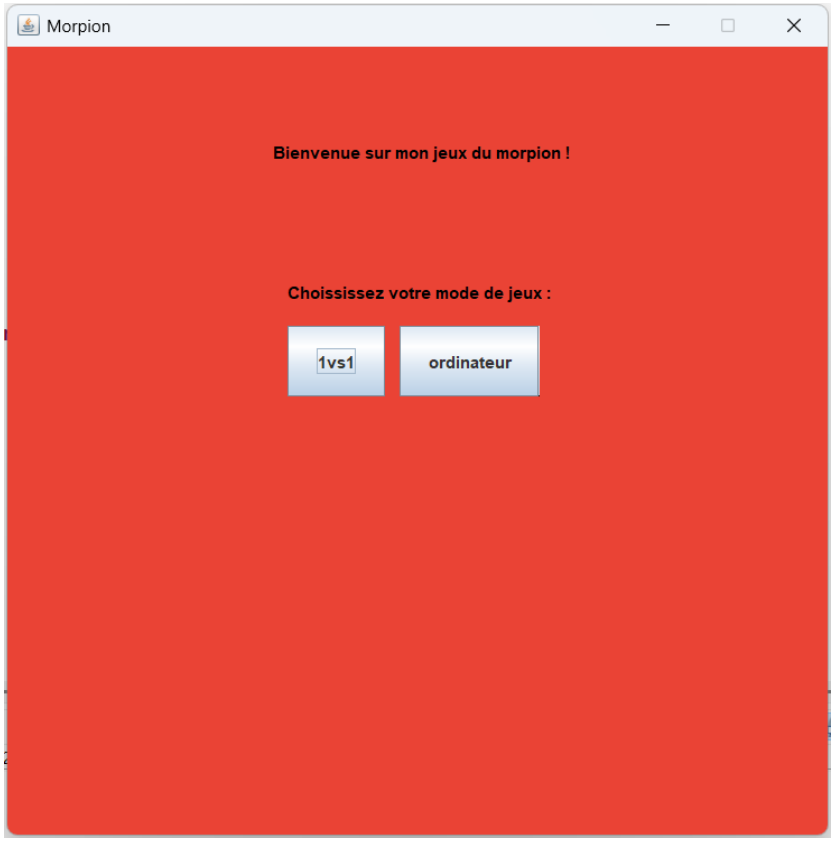
Si il y a égalité, même chose :



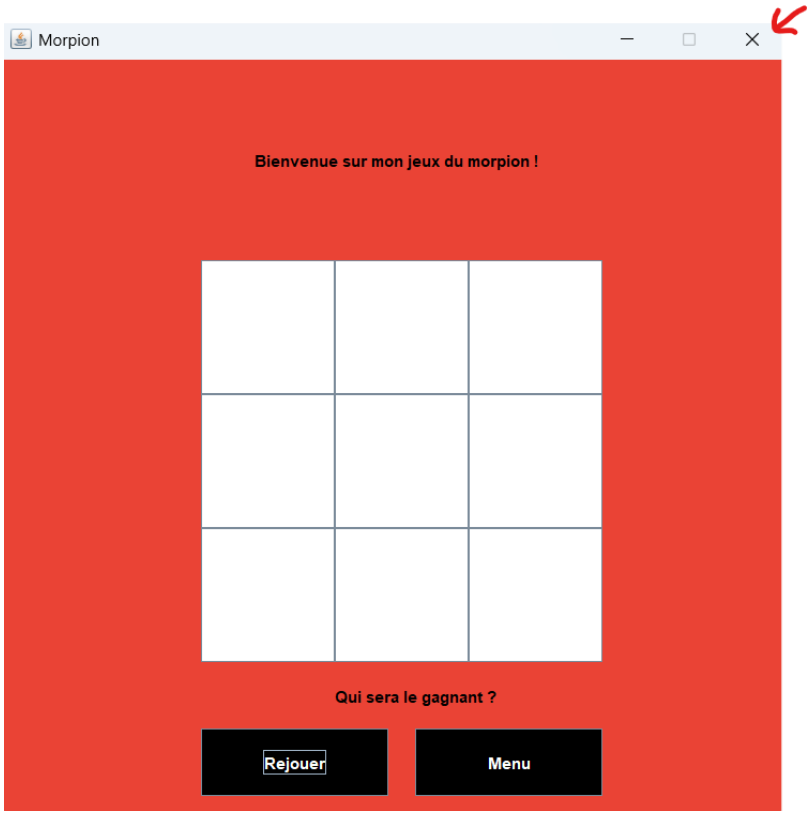
Une fois la partie terminée, le joueur peux soit relancer la partie en cliquant sur le bouton rejouer :



Soit revenir au menu de départ en cliquant sur le bouton menu :



Enfin, pour quitter et fermer l'application, le joueur n'a qu'a cliquer sur le bouton fermer symbolisé par « **X** » en haut à droite de la fenêtre :



IV. Conclusion

Pour conclure, la réalisation de ce projet de jeu de Morpion en Java a été une étape clé dans l'application pratique de mes connaissances en développement logiciel. Ce projet ne s'est pas limité à la simple création d'un jeu, mais a constitué un véritable exercice de conception logicielle.

Bilan technique : Le développement de cette application m'a permis de consolider ma maîtrise de la Programmation Orientée Objet (POO). J'ai pu mettre en œuvre des concepts fondamentaux tels que la séparation des responsabilités (logique de jeu vs interface graphique), la gestion des tableaux multidimensionnels et l'implémentation d'algorithmes de vérification de victoire. L'utilisation de la bibliothèque Swing m'a également permis d'approfondir la gestion des événements et la création d'interfaces utilisateurs ergonomiques.

Compétences acquises :

- Analyser un besoin et le traduire en fonctionnalités.
- Structurer un code de manière propre et réutilisable.
- Tester et déboguer une application pour garantir une expérience utilisateur fluide.

Perspectives d'évolution : Bien que le jeu soit aujourd'hui pleinement fonctionnel, plusieurs axes d'amélioration pourraient être explorés pour enrichir cette solution :

1. **Intelligence Artificielle :** Pour permettre un mode de jeu contre l'ordinateur imbattable.
2. **Persistance des données :** Ajouter une base de données pour enregistrer un historique des scores et des statistiques de jeu.
3. **Mode Réseau :** Développer une fonctionnalité multijoueur via les *Sockets Java* pour permettre à deux joueurs de s'affronter sur des postes différents.

En somme, ce projet constitue une base solide qui démontre ma capacité à mener à bien le développement d'une application de bureau complète, de sa conception à sa finalisation.