❌⭕ Morpion Java

Jeu de Morpion avec Interface Graphique Swing et Intelligence Artificielle

Documentation Technique Complète

Auteure	Youma DRAME
Formation	BTS SIO SLAM — École IRIS Paris
Période	Novembre 2025
Technologies	Java 17, Swing (JFrame, JButton, JPanel), POO, ActionListener
Type	Application desktop — Jeu de plateau avec IA
Titre fenêtre	Application Morpion 2026

1. Contexte & Objectifs

Ce projet consiste à développer un jeu de Morpion complet en Java avec interface graphique Swing. Il propose deux modes de jeu : 2 joueurs humains ou 1 joueur contre une intelligence artificielle.

Objectifs du projet
► Interface graphique complète Java Swing
► Mode 2 joueurs avec alternance automatique des tours
► Mode IA : placement intelligent avec détection des coups gagnants
► Détection automatique des 8 combinaisons gagnantes
► Fonctions Rejouer, Save et Quitter
► Affichage du tour en cours et des noms des joueurs
► POO : séparation claire Logique / Interface graphique

2. Architecture POO

2.1 Classes principales

Structure des classes
► MorpionApp (extends JFrame) — Fenêtre principale, initialisation de l'UI
► Grille — Logique du jeu : tableau 3x3, détection victoire/égalité
► Joueur — Entité joueur : nom, symbole (X ou O), type (humain/IA)
► IA — Algorithme de décision pour le mode automatique

►	GestionnairePartie — Orchestration des tours, sauvegarde d'état

2.2 Interface graphique Swing

Composants Swing utilisés :

- JFrame — Fenêtre principale de l'application
- JPanel — Panneau contenant la grille 3x3
- JButton[3][3] — Les 9 cases de la grille
- JRadioButton — Sélection du mode (2 joueurs / Avec IA)
- JLabel — Affichage du tour en cours et des noms
- ActionListener — Gestion des clics sur les cases

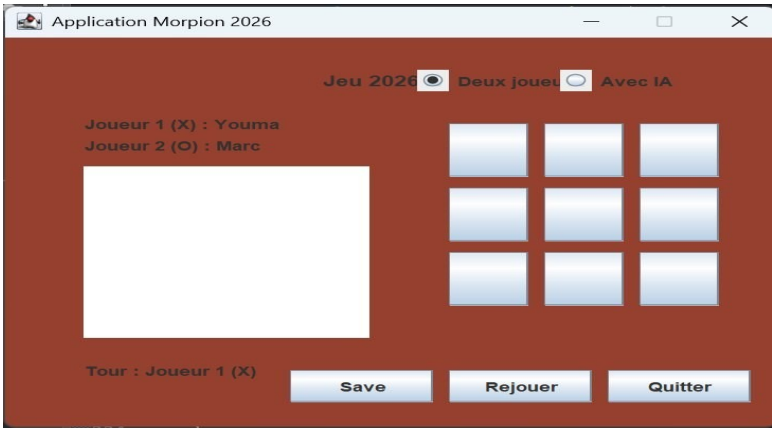

Fig. 1 — Interface graphique du jeu Morpion (mode 2 joueurs)

3. Algorithme IA

3.1 Stratégie de l'IA

L'IA analyse la grille en 3 priorités dans l'ordre :

Priorités de décision de l'IA	
►	Priorité 1 — Gagner : si l'IA peut aligner 3 symboles, elle joue ce coup
►	Priorité 2 — Bloquer : si l'adversaire peut gagner au prochain coup, l'IA bloque
►	Priorité 3 — Stratégie : jouer au centre si libre, sinon un coin, sinon un bord

3.2 Détection des 8 combinaisons gagnantes

```
// 8 combinaisons gagnantes int[][] combos = {     {0,1,2}, {3,4,5}, {6,7,8}, // Lignes
{0,3,6}, {1,4,7}, {2,5,8}, // Colonnes     {0,4,8}, {2,4,6}            // Diagonales }; for
(int[] combo : combos) {     if (grille[combo[0]] == grille[combo[1]]         &&
grille[combo[1]] == grille[combo[2]]         && grille[combo[0]] != ' ') {         return
grille[combo[0]]; // Gagnant     } }
```

4. Gestion des événements

Chaque bouton de la grille est associé à un ActionListener :

- Clic sur une case vide → placer le symbole du joueur actuel
- Vérifier les 8 combinaisons gagnantes après chaque coup
- Si victoire → afficher un message et désactiver la grille
- Si égalité (9 cases jouées sans gagnant) → afficher match nul
- Bouton Rejouer → réinitialiser la grille et les boutons
- Bouton Save → sauvegarder l'état actuel de la partie

5. Tests Réalisés

Test 1	Alignement horizontal → Victoire détectée et message affiché ✅
Test 2	Alignement diagonal → Victoire détectée ✅
Test 3	9 cases jouées sans gagnant → Match nul affiché ✅
Test 4	Mode IA — IA bloque coup gagnant adverse → Coup de blocage joué ✅
Test 5	Bouton Rejouer → Grille remise à zéro ✅
Test 6	Sélection mode IA en cours de partie → Bascule correcte ✅

6. Bilan

Compétences développées	
►	Programmation orientée objet Java : classes, héritage, interfaces
►	Interface graphique Swing : composants, layout managers, événements
►	Algorithmique : détection de patterns, IA à règles de priorité
►	Gestion des événements : ActionListener, mise à jour de l'interface