DOCUMENTATION DEVELOPPEUR :

Le projet se structure selon le paradigme modèle, vue, contrôleur. Le modèle représente le jeu à proprement parler, la vue l'interface utilisateur, implémenté avec Swing et le contrôleur, l'outil de mise à jour de la vue et du modèle. Les principales fonctionnalités du projet ont été installées. En effet, il est possible de jouer une partie de morpion solitaire sur une grille au préalablement choisi ou de faire résoudre la grille par une IA. La grille de base se trouve dans le fichier *init_grid.txt*.

Le modèle se compose de quatre classes cœurs qui sont : Point, Line, Grid et AI. La classe Grid représente le plateau de jeu contenant des méthodes afin de pouvoir interagir avec celui-ci. La classe Point permet de simuler les points présents sur le plateau de jeu. La classe Line représente les lignes tracés après avoir placé un point. Les lignes ne sont en vérités utiles seulement pour le mode *DISJOINT* du jeu. Finalement, la classe AI représente l'intelligence artificielle qui peut résoudre la grille. A chaque tentative de jeu, la classe Grid teste si le choix du joueur (ou de l'IA) est jouable avec la méthode *playable*. Cette méthode teste s'il n'existe pas déjà un point à l'emplacement choisi puis cherche s'il existe une ligne de taille 5 formé par le futur point. La méthode parcourt « en étoile » les points alentours au choix de l'utilisateur. Malheureusement, il a paru trop complexe l'implémentation du choix de ligne à jouer, après choix du point, cette fonctionnalité n'est donc pas présente. La ligne jouée est donc la première trouvée selon l'ordre des axes cherchées. On teste l'arrêt du jeu, en parcourant la grille à la recherche d'un point jouable, on bloque le jeu s'il n'en existe pas. Il a été décidé que l'implémentation de la grille de point se ferait par des tableaux ordinaires sachant que notre utilisation se limite la plupart du temps à de l'insertion ou de la lecture ciblé (avec les indices connus) avec des ensembles de données à taille fixe.

Il est possible par l'interface graphique de pouvoir recommencer une partie grâce au bouton *restart*, implémenté avec la classe JButton. Le choix du fichier grille est récupéré par le JTextField et le mode par le JSlider. La résolution automatique avec le bot se fait par le bouton *solve* aussi implémenté par un JButton. La vue de la classe Grid est représentée par la classe GamePanel qui elle-même se compose d'un tableau de PointView, représentation directe des instances de la classe Points. La classe PointView étends directement la classe JPanel. Il a été choisi de passer par une « mosaïque » de JPanel afin de pouvoir mieux interagir au détail avec l'interface graphique : hovering des points, traçage des lignes, mise à zéro de la grille, changement de grille, etc… cela permet d'éviter de retracer tout le plateau à chaque interaction/mise à jour du modèle. Pour résumer, la fenêtre graphique se compose de deux parties, la partie jeu et la partie menu, toutes représentée par des JPanel.

Les interactions avec l'utilisateur se font via les boutons de la partie menu de la fenêtre mais aussi par la souris. Il a donc été implémenté la classe MouseController qui implémente les interfaces *MouseMotionListener* et *MouseListener*. La partie jeu de la fenêtre est mis à jour à chaque déplacement de la souris afin de gérer l'hovering des points.

L'intelligence artificielle se limite à une intelligence aléatoire.