Projet d’informatique : Pipopipette

**Compte-rendu final**

Mode d’emploi :

Le programme a été codé de manière à être le plus intuitif possible à l’utilisation. Pour commencer, à son lancement, une première fenêtre (Fenetre1) s’ouvre et permet à l’utilisateur de rentrer ses paramètres de jeu (nom, s’il veut jouer contre un ordinateur…). Il peut également choisir de cliquer sur le bouton « Paramètres avancés », ce qui ouvrira une nouvelle fenêtre (FenetreParam) où il peut choisir de personnaliser son expérience de jeu (taille du plateau, couleur). Lorsque l’utilisateur est satisfait de ses réglages, il clique sur le bouton « Ok » et passe enfin à la fenêtre de jeu (Fenetre2).

Lorsque Fenetre2 s’affiche (sauf si le Joueur1 est un Ordinateur, auquel cas il jouera automatiquement), seuls les points du plateau sont représentés (ainsi que les noms des joueurs qui restent invariablement en haut de la fenêtre). Pour qu’un Humain joue, il lui suffit de déplacer la souris sur le plateau, et de cliquer lorsque le trait affiché correspond au coup qu’il veut jouer. Si au moins un Ordinateur joue, le coup est joué automatiquement sans que l’utilisateur n’ait rien à faire.

A la fin de la partie (lorsque le plateau est rempli), l’écran de fin s’affiche automatiquement pour signaler que le jeu est fini. L’utilisateur n’a alors plus qu’à cliquer sur le bouton « Ok » pour quitter le programme (on notera également que l’utilisateur peut quitter à tout moment en cliquant sur la croix rouge).

Points forts :

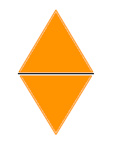
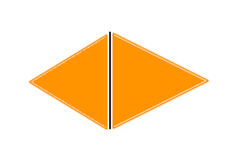
Bien évidemment, nous dirions que le principal point fort de notre programme est qu’il utilise swing et awt pour proposer une IHM, plutôt que de passer par la console. Ce choix comporte un intérêt graphique évident (une interface graphique est objectivement bien plus agréable à regarder que la console) et apporte également un grand confort d’utilisation au joueur par la programmation événementielle, qui permet d’utiliser la souris plutôt que de toujours utiliser le clavier. De plus, le joueur a une large palette de choix pour jouer, rendue possible par les deux premières fenêtres, ce qui améliore grandement l’expérience de jeu (le jeu en lui-même pouvant paraître quelque peu répétitif).

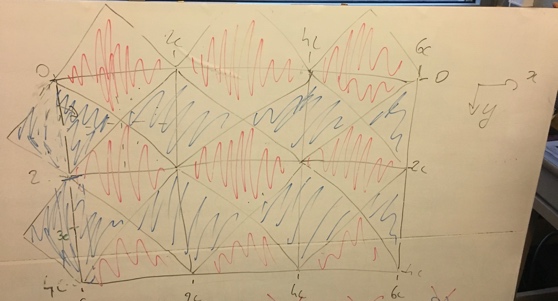
Limitations :

Le principal point faible de notre programme est lié à son principal point fort : en effet, le fait d’utiliser une interface graphique rend le code beaucoup plus long et compliqué, ce qui augmente de plus le risque de bugs d’utilisation. Bien que nous ayons tout fait pour réduire leur nombre, il nous semble inévitable qu’il subsiste au moins un ou deux problèmes que nous n’avons pas repérés. Par exemple, un problème que nous avons vu mais pas eu le temps de corriger est que si l’utilisateur sort le curseur de la souris du plateau, il lui est toujours possible de cliquer sur le dernier trait affiché (alors qu’il aurait été préférable de faire disparaître le trait).

D’autre part, la longueur du code le rend plus compliqué à comprendre et pose problème pour la réalisation d’un diagramme UML.

Quelques clarifications :

 La manière dont fonctionnent le plateau ou même l’OrdinateurDifficile pouvant être difficiles à comprendre juste avec le code, nous vous proposons ici quelques éléments qui devraient vous aider à comprendre le programme.

 Avant tout, il est nécessaire d’expliquer le choix que nous avons fait pour diviser le plateau. En effet, pour qu’une ligne apparaisse pour une certaine position du curseur de la souris, nous avons décidé d’associer chaque ligne du plateau à une zone autour de ladite ligne. C’est donc lorsque le curseur rentre dans cette zone que la ligne s’affiche. Pour chaque ligne, la zone est constituée de deux triangles (ayant pour côté commun la ligne) tel que montré ci-contre (à noter que les triangles à gauche et en-dessous du plateau ne sont pas représentés). Cette forme permet de remplir entièrement le plateau sans que deux zones s’intersectent.

Tel qu’il est dit dans les commentaires du code, le plateau de jeu se divise en plusieurs parties. Il y a tout d’abord le grand rectangle délimité par les points extérieurs, mais également les triangles sortant du rectangle correspondant aux points les plus extérieurs. Ces triangles font également partie du plateau.

Ensuite, une ligne est repérée par ses coordonnées (xPoint, yPoint) et son orientation. En vérité, les coordonnées font référence soit au point de gauche (ligne horizontale) ou au point du haut (ligne horizontale), le second point étant déductible grâce à l’orientation de la ligne. Ainsi, lorsqu’il est fait mention des coordonnées d’une ligne, ces coordonnées sont en fait la position du point sur le plateau (donc (0,0) pour le point en haut à gauche et (1,1) pour un point au milieu d’un carré 2x2). La position d’un point (xPoint, yPoint) se différencie cependant de ses coordonnées (coordonnées (x,y) par rapport au point en haut à gauche de la fenêtre).

Enfin, en ce qui concerne l’OrdinateurDifficile, son fonctionnement est relativement simple. Il parcourt le tableau plusieurs fois en cherchant à chaque fois la meilleure case à jouer selon les critères suivants : avant tout, il cherche une case où trois traits sont déjà tracés pour remplir le carré et gagner un point. Ensuite, il cherche les cases où il y a 1 puis 0 trait tracé, pour ne pas avantager l’autre joueur. Enfin, il cherche dans le pire des cas les cases où il y a deux traits déjà tracés, donnant l’opportunité à l’autre de remplir le carré. Afin de donner l’illusion d’un ordinateur aussi humain que possible, il parcourt le tableau en partant du dernier coup joué et en tournant autour de la ligne (en décrivant une forme de spirale).

Conclusion :

Nous avons pris beaucoup de plaisir à développer ce programme, qui nous a permis d’apprendre beaucoup de choses sur les IHM. Le travail d’équipe était très équilibré et n’a pas posé de problème à la réalisation du projet. Nous espérons donc que la correction de notre programme ne sera pas trop compliquée et nous vous souhaitons beaucoup de plaisir à jouer à Pipopipette !