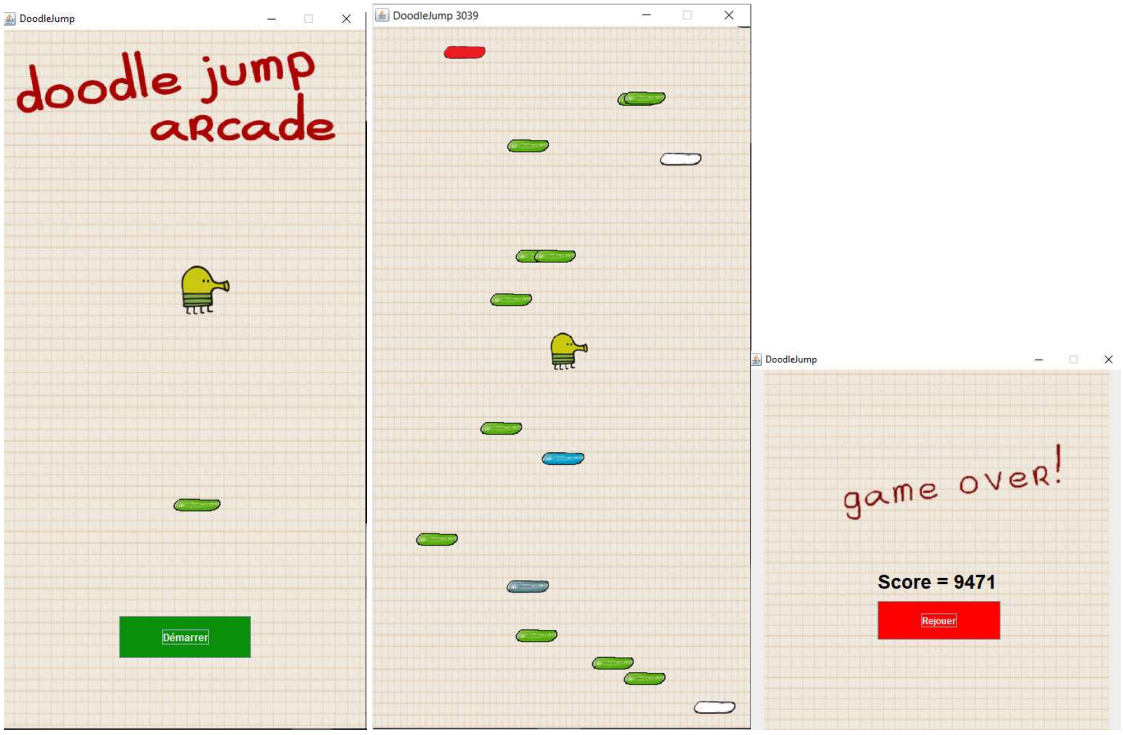
Auteurs : Killian Giraudi, Manon Salvado, Faustine Chabas et Marie Gaillard

Mini-Projet : Doodle Jump

I. Cahiers des charges

Nous allons mettre en place un jeu de plateformes interactif inspiré du jeu Doodle Jump. Le personnage représentant le joueur dans le jeu saute sur des plateformes. Le jeu est basé sur le record : l'objectif du joueur est d'aller le plus haut possible. Dans notre jeu cette hauteur est mesurée en nombre de pixel. Le joueur peut effectuer différents types sauts suivant la plateforme sur laquelle il rebondit. Une plateforme verte représente un saut normal, celle avec les ressorts permettent un super saut, les plateformes bleues diminue la hauteur du saut et les grises permettent un saut d'une hauteur plus élevée que la normale mais plus faible que le super saut. Il existe aussi des plateformes blanches qui permettent au joueur de sauter une seule fois dessus puisqu'elles disparaissent après ce saut. En revanche, le joueur perd s'il rate une plateforme et tombe hors de la fenêtre ou s'il touche une plateforme rouge. L'aspect scientifique sera introduit lorsque le joueur saute sur une plateforme et retombe avec la gravité.

L'utilisateur de ce jeu a accès à une fenêtre de démarrage qui lui permet de lancer le jeu. Ensuite, une seconde fenêtre apparaît. C'est celle du déroulement du jeu où il peut déplacer le personnage avec les flèches gauche et droite de son clavier afin d'atteindre les différente plateformes présentes dans la fenêtre. Pour finir, si l'utilisateur perd, il a accès à une fenêtre perdu qui lui donne son score et la possibilité de rejouer.



II. Description du problème posé

Lors de la création de ce jeu, nous allons rencontrer différents problèmes que nous allons devoir résoudre. Tout d'abord, nous devons créer un personnage : le Doodle ainsi que les paliers sur lesquels il saute. Nous allons donc devoir gérer la taille du doodle et des plateformes afin que le personnage détecte ces dernières et rebondissent. Pour cela, il faut aussi que le Doodle ait une certaine vitesse ce qui lui permettra de continuer à progresser en hauteur. Le déplacement latéral du Doodle est aussi important. En effet, dans le jeu, le personnage peut aller de gauche à droite en traversant les bords de l'écran et nous allons devoir introduire ce déplacement dans notre jeu. Il faut aussi que lorsque l'utilisateur appuie sur les touches gauche/droite, le déplacement ne soit pas trop rapide ni trop lent.

En ce qui concerne les plateformes, elles ne doivent pas être trop nombreuses sinon le jeu est trop simple et il n'y aura pas de défaite de l'utilisateur. On doit également retrouver un nombre restreint de plateformes "spéciales" et elles doivent remplir leur rôle : faire perdre le joueur ou le faire sauter à des hauteurs différentes. De plus, il ne doit pas avoir de paliers à moitié ou totalement hors de la fenêtre.

Une fois ces éléments créés, il faut les inclure dans une fenêtre et dans le déroulement du jeu. Nous devons aussi penser à faire défiler la fenêtre et donc à créer des plateformes pour toute la durée du jeu. Il faudra également inclure le score du joueur afin qu'il puisse se rendre compte de sa progression. Sans oublier les interactions entre les différentes fenêtres, qu'il faut pouvoir gérer avec fluidité. Le jeu se déroulant sur plusieurs fenêtres, il faut s'assurer de la fluidité de passage d'une fenêtre à l'autre notamment, lors de l'ouverture et de la fermeture des fenêtres. On cherche à éviter que des fenêtres restent ouvertes alors qu'elles ne le devraient pas. La gestion des fenêtres intervient aussi lors de l'utilisation par différents utilisateurs, il faut s'assurer que la taille des fenêtres et aussi des objets qu'elles contiennent soit adaptés selon l'ordinateur de l'utilisateur.

III. Principe de l'algorithme

Dans un premier temps, nous avons créé une classe Element possédant différents attributs comme des coordonnées, une taille et un JLabel. Cette classe représente les différents objets que nous allons utiliser pour créer notre jeu : le Doodle et les plateformes. Ainsi, ces deux éléments héritent de la classe Element et possèdent des coordonnées, une taille précise, adapté à la taille de votre écran d'ordinateur, et une image.

Une fois nos éléments de jeu mis en place, nous devons créer les fenêtres dans lesquels ils vont apparaître et évoluer. Nous allons donc commencer par créer une fenêtre démarrage qui permet de lancer le jeu à l'aide d'un bouton "Démarrer". Lorsque l'utilisateur clique sur ce bouton, une seconde fenêtre s'ouvre pendant que la première se referme. Nous avons maintenant devant nous la fenêtre jeu avec des paliers et le Doodle. Pour que le Doodle se déplace, nous lui avons donné une vitesse verticale et horizontale. De plus, nous avons associé les flèches droite et gauche de notre clavier avec les variations de vitesse du Doodle ce qui permet à l'utilisateur de déplacer le personnage. Nous avons également mis en place une méthode CheckSortieEcran() qui permet de vérifier les coordonnées du Doodle et de lui faire traverser les parois de la fenêtre d'un côté à l'autre, cette méthode permet de faire la même chose avec les paliers. Son déplacement est également associé à

un timer sans lequel il ne serait pas possible de jouer. Pour créer les paliers, nous avons généré dans le constructeur de la fenêtre des coordonnées x et y aléatoires. Chaque coordonnée a été associées à un palier qui a été ajouté à une liste, nous permettant d'obtenir une liste de paliers dans la fenêtre jeu.

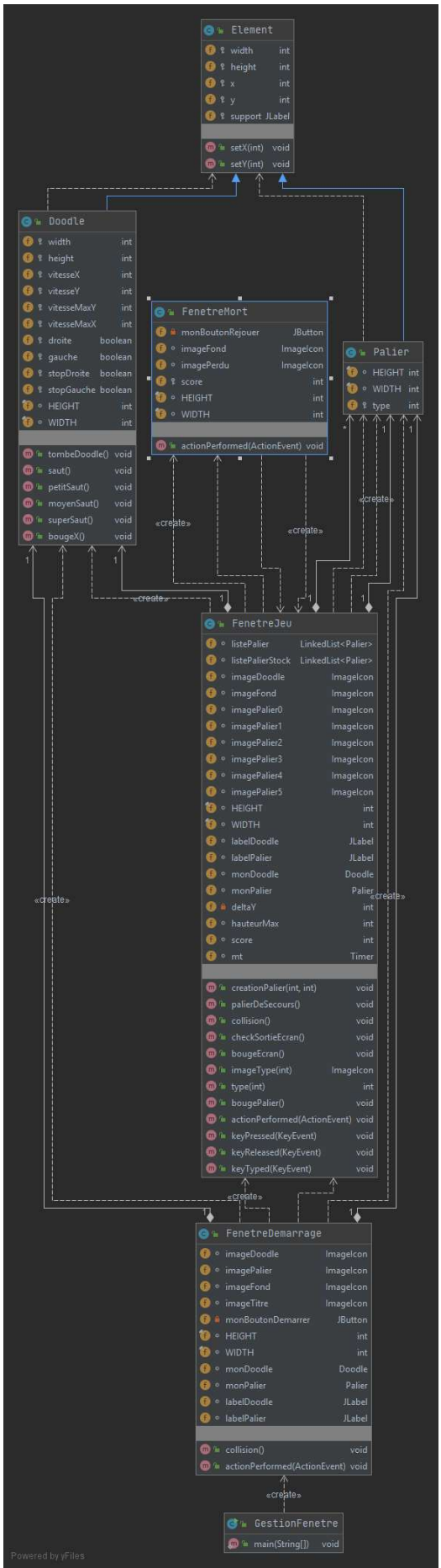
Nous avons également mis en place un pourcentage au-delà duquel le programme crée moins de plateformes. Cela nous permet d'avoir un nombre de palier restreint pour que le jeu ne soit pas trop simple. Les paliers disposent également de type différent. La méthode type() permet d'attribuer le type au palier lorsqu'elle est appelée dans la méthode créationPalier(), la méthode imagePalier() renvoie l'image qui correspond au type qu'on lui donne en paramètre. Ces méthodes permettent de changer le type du palier lorsqu'elle est appelée dans la méthode bougeEcran(). En effet, nous avons créé un deuxième pourcentage permettant d'attribuer aux paliers un type et une image qui sont à l'origine des paliers spéciaux. Ce nouveau type est associé à un saut de différente hauteur en fonction du type ou à la mort du joueur si la plateforme est rouge.

Une fois ces éléments mis en place, nous devons les faire interagir entre eux. Nous avons donc créé la méthode collision qui analyse simultanément les coordonnées du Doodle et des plateformes pour que les différentes méthodes saut() (saut(), moyenSaut(), petitSaut(), superSaut() associées au Doodle se déclenchent quand celui-ci se retrouve sur un palier d'un type précis. On a 6 paliers différents qui se comporte différemment quand le doodle saute dessus (méthode collision) ainsi, pour les paliers verts le Doodle effectue un saut normal, mais si il tombe sur un palier bleu, gris ou à ressort il sautera respectivement peu, moyennement haut et très haut. Si le Doodle saute sur un palier blanc alors ce dernier disparaît (déplacer en dehors de la fenêtre). Enfin si le Doodle touche un palier rouge il meurt. Le Doodle possède aussi une méthode tombeDoodle() qui lui permet de simuler les effets de la gravité, elle évolue suivant la hauteur et fait retomber après ces sauts sur les paliers. En ce qui concerne les paliers, nous avons mis en place une méthode bougeEcran() qui déplace les paliers vers le bas proportionnellement à la monté du Doodle. Elle permet également de recycler les paliers qui sortent de l'écran en bas pour les ramener plus haut ainsi nous n'avons pas besoin de générer de nouveaux objets paliers tout au long du défilement de la fenêtre. Cette méthode modifie également le nombre de palier présent dans le jeu en tenant compte de la hauteur de la fenêtre. En effet, plus la progression dans le jeu est importante plus la hauteur l'est et donc moins il y aura de paliers. Cela permet de mettre en place une difficulté supplémentaire. Cette méthode va également permettre de calculer le score du joueur. Le déplacement horizontal des paliers de type 2 est géré par bougePalier(),lié au timer, leur déplacement est continu, la direction dépend de la position du palier dans la liste, elle varie donc au cours du temps. La méthode nommée palierDeSecours() permet la création de paliers de secours lorsque la distance entre le Doodle et les paliers est trop grande pour être atteinte par un saut, ainsi, le joueur ne sera pas bloqué par un manque de palier. Toutes ces méthodes servent au bon déroulement du jeu et sont donc appelées dans le actionPerformed.

Il existe une dernière méthode vérifiant les coordonnées du personnage, c'est la méthode checkMort(). Cette méthode a finalement été incluse dans la méthode checkSortieEcran, pour apporter plus de cohérance dans le code, et améliorer le temps de calcul. Cette méthode vérifie si le doodle ne tombe pas en dehors de la fenêtre. Si c'est le cas, le jeu s'arrête et la fenêtre jeu se ferme pendant qu'une nouvelle fenêtre apparaît annonçant la défaite du joueur avec le score qu'il atteint.

Sur cette dernière fenêtre, l'utilisateur à la possibilité de relancer une partie à l'aide d'un clic sur le bouton rejouer.

IV. Structure des données (diagramme UML)



Notre programme est constitué de quatres classes et de trois fenêtres.

La classe GestionFenetre entraîne l'ouverture de la FenetreDemarage. Cette fenêtre est l'accueil de notre jeu. L'utilisateur démarre le jeu en appuyant sur le bouton démarrer, qui ouvre la seconde fenêtre FenetreJeu et qui ferme la première fenêtre.

Dans le constructeur de la fenêtre on place le Doodle et on créer les premiers paliers. C'est dans fenêtreJeu que se concentre la majorité des méthodes, qui permettent de gérer la gestion du temps (création du timer), le déplacement du Doodle (actionPerformed) et des paliers(bougeEcran(), bougePalier()) ainsi que leurs interactions (collision()) .

Lorsque l'utilisateur perd, la FenetreJeu se ferme et la FenetreMort apparaît avec le score du joueur et la possibilité de rejouer (le bouton rejouer permet de fermer la FenetreMort et d'ouvrir la FenêtreJeu).

Les classes Doodle et Palier héritent de la classe Element. Elles héritent donc des attributs de la classe éléments (taille, coordonnées et support). Le Doodle a des attributs qui lui sont propres une vitesse en x et y (et une vitesse max en x et y), ainsi que des indicateurs de déplacement latéraux (ils indiquent si le Doodle se déplace vers la gauche ou vers la droite). Le Palier a en plus des attributs de la classe Element, un type. A chaque type de palier correspond une image différente et une action différente en cas de collision avec le Doodle.

V. Améliorations possibles du projet et bugs connus

Notre projet est une version très similaire du vrai Doodle Jump. Nous avons choisi d'utiliser des JPanel avec des images tirées du jeu.

Nous pourrions encore améliorer le jeu en créant par exemple des monstres sur lesquels le personnage peut envoyer des projectiles (il faudrait donc créer de nouvelles classes héritières de Element).

On peut également essayer de faire de telle sorte que les paliers ne puissent pas se chevaucher pour améliorer l'esthétique. Dans ce même but, on peut penser à introduire différents décors qui modifieraient l'apparence du fond, du personnage, des monstres. Enfin, pour améliorer les performances du Jeu, il faudrait utiliser une méthode paint pour l'affichage, plutôt que d'utiliser des JPanel, qui demande plus de puissance et de temps de calculs.

VI. Carnet de route

Semaine 1 : 24/02/2020	- Mise en place sur papier des différentes classes et fenêtres à créer - Début du codage de la fenêtre jeu et démarrage
Semaine 2 : 09/03/2020	- Codage des objets Element, Doodle - Codage de la classe principale - Codage du constructeur de la fenêtre jeu, ajout du fond et du Doodle avec un essai de rebond sur un Palier qui ne fonctionne pas. Ajout du timer et de la méthode actionPerformed - Codage de la fenêtre démarrage
Semaine 3 : 23/03/2020	- Codage objet Palier - Création aléatoire des paliers sans défilement de la fenêtre pour le moment - Rebond du Doodle sur une plateforme précise - Réorganisation de toutes les classes pour qu'elles correspondent aux conventions de codage
Semaine 4 : 30/03/2020	- Fluidification de la chute du Doodle et de sa vitesse, plus création des déplacements horizontaux à l'aide des flèches du clavier. Déplacement du Doodle totalement réalisé. - Création de la méthode collision qui permet le saut du Doodle sur toutes les plateformes - Mise en forme de la Fenêtre démarrage (fond, dimensionnement des JLabel et du Jbutton) avec un palier, un doodle qui saute dessus, un titre et le bouton "Démarrer". Ouverture de la fenêtre jeu dans la fenêtre démarrage. - Création et mise en forme de la fenêtre qui apparaît pour la

	défaite du joueur. Création du bouton rejouer et de l'affichage du score. Mise en place de son ouverture dans la fenêtre jeu.
Semaine 5 : 06/04/2020	- Allègement du code - Création de la méthode bougeEcran qui permet le défilement de la fenêtre jeu. Cette méthode est initialement basée sur le score du joueur. Elle crée une erreur plus loin dans le déroulement du jeu. - Création de la méthode palierDeSecours
Semaine 6 : 13/04/2020	- Modification de la méthode CheckMort : arrêt du timer - quand la fenêtre perdue s'ouvre. Cela empêche la réouverture de cette fenêtre - Nouvel allégement du code et réorganisation aux conventions
Vacances de Printemps	- Création de différents types de palier et des conséquences qu'ils introduisent : perte du jeu, super saut, petit saut ... - Modification de la méthode bougeEcran qui est maintenant basée sur la hauteur de la fenêtre et qui permet une diminution du nombre de paliers en fonction de la progression. Cette modification permet la suppression de l'erreur. - Création de la méthode type et imageType qui permet de changer le type du palier et son image. - Création de la méthode bougePalier - Redimensionnement des objets en fonction de l'écran de l'ordinateur et non de la taille de la fenêtre - Allégement et mise en forme du code - Le code est intégralement commenté - Rédaction du compte rendu - Rendu du projet

VII. Conclusion et bibliographie

Pour conclure, ce projet nous a permis de mettre à profit les connaissances acquises durant ces deux années de premier cycle de façon plutôt ludique.

Nous avons essentiellement utilisé les documents de cours sur les interfaces graphiques et les listes chaînées. Nous nous sommes également largement inspirés du jeu Doodle Jump original.

L'implication dans ce projet est de 25% pour chaque membre du groupe.