Specs pour le projet

C# Programmation objet

|  |  |
| --- | --- |
| **Document Name** | Specs pour le projet C# Programmation objet |
| **Project Code:** |  |
| **Version:** | **V1** |
| **Status:** | **Finalisé** |
| Préparé par: | **Mathieu MACK** |
| **Date de dernière modification :** | **20/03/2019** |

**Version Control**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Version** | **Historique** | **Date** | **Auteur** | **Description** |
| 0 | Initialisation | 20/03/2019 | M. MACK | Document initialisation |

**Table des matières**

[1 Introduction 4](#_Toc3925439)

[2 Règles du jeu 5](#_Toc3925440)

[3 Préparation 6](#_Toc3925441)

[3.1 Déroulement 6](#_Toc3925442)

[3.2 Déplacer un pingouin 7](#_Toc3925443)

[3.3 Prendre la tuile banquise 7](#_Toc3925444)

[3.4 Victoire 7](#_Toc3925445)

[4 A implémenter 8](#_Toc3925446)

[4.1 Moteur du jeu et IAs 8](#_Toc3925447)

[4.2 Logs 8](#_Toc3925448)

[4.3 Tests unitaires 8](#_Toc3925449)

[4.4 Tests fonctionnels automatisés [Option] 8](#_Toc3925450)

[https://github.com/appium/appium 8](#_Toc3925451)

[https://github.com/appium/appium/blob/master/docs/en/drivers/windows.md 8](#_Toc3925452)

[5 Livrables 9](#_Toc3925453)

[6 Présentation orale 10](#_Toc3925454)

[6.1 Présentation technique 10](#_Toc3925455)

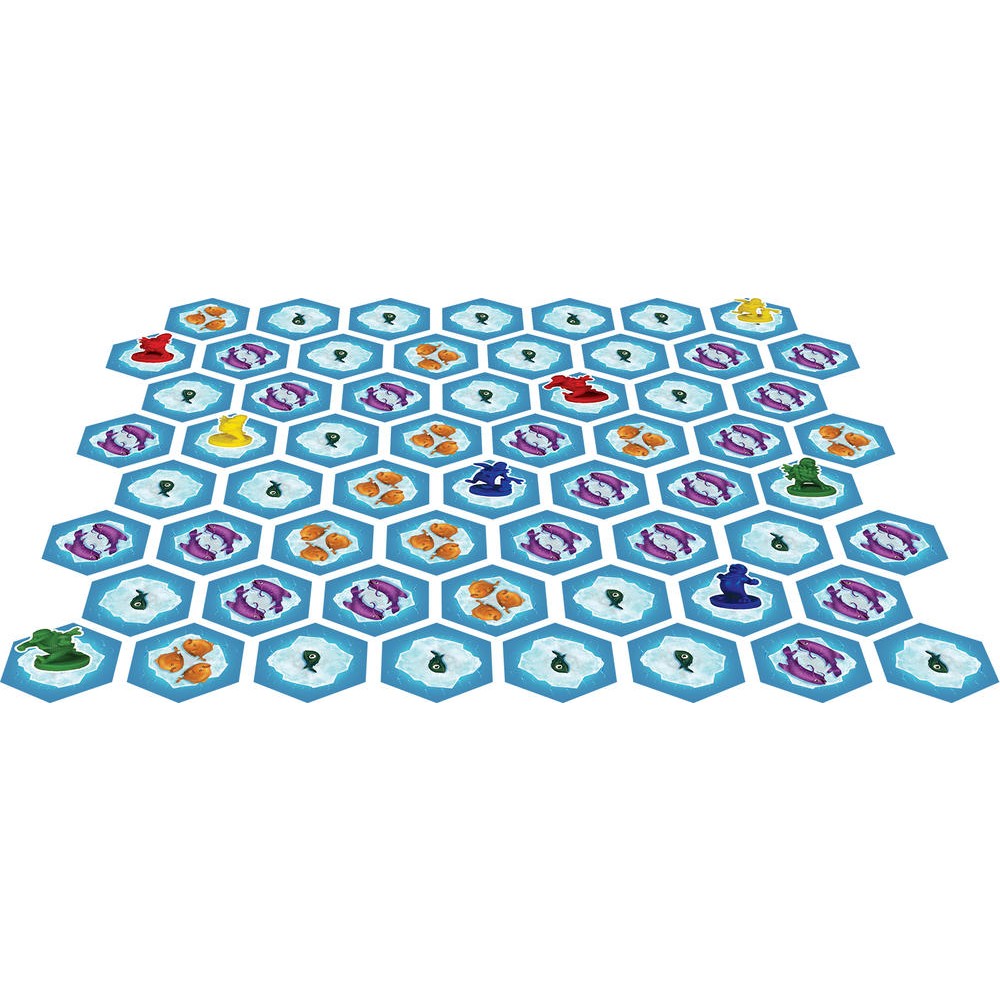
[6.2 Démo 10](#_Toc3925456)

[7 Barème de notation 11](#_Toc3925457)

# Introduction

L’objectif du projet est de créer un jeu sous la forme d’une application (client lourd WPF).

Le jeu à implémenter est le jeu suivant : Pingouins



L’application devra permettre à un utilisateur humain de jouer contre d’autres humains ou contre l’ordinateur au travers d’une interface utilisateur.

Le projet référence contenant déjà les interfaces est disponible à l’adresse <https://github.com/mathieumack/Game.Penguins>. Chaque groupe devra effectuer un fork du projet afin de pouvoir le faire évoluer de son côté.

Ce document sera mis à jour en fonction des retours et questions qui seront posées par les autres groupes.

# Règles du jeu

**Résumé**

La banquise fond ! attrapez autant de poissons que possible avant qu’elle ne disparaisse. Si vous ne le faites pas, d’autres le feront pour vous. Désormais chaque famille de pingouins lutte pour pêcher le plus grand nombre de poissons.

Vos pingouins doivent être les plus rapides sur la banquise qui se réduit pour pêcher les plus gros poissons et bloquer les adversaires. Mais vos pingouins doivent faire attention ! Si un pingouin reste bloqué sur un bout de banquise, c’en est terminé pour lui.

Votre objectif est simple mais il faudra compter avec les pingouins adverses sur un plateau toujours changeant et de plus en plus petit. Quelle stratégie adopter pour remplacer la compétition !

**But du jeu**

Dans Pingouins, de 2 à 4 joueurs déplacent leurs pingouins sur les tuiles de banquise pour récupérer du poisson. Le joueur qui aura récupéré le plus de poissons à la fin du jeu sera le vainqueur.

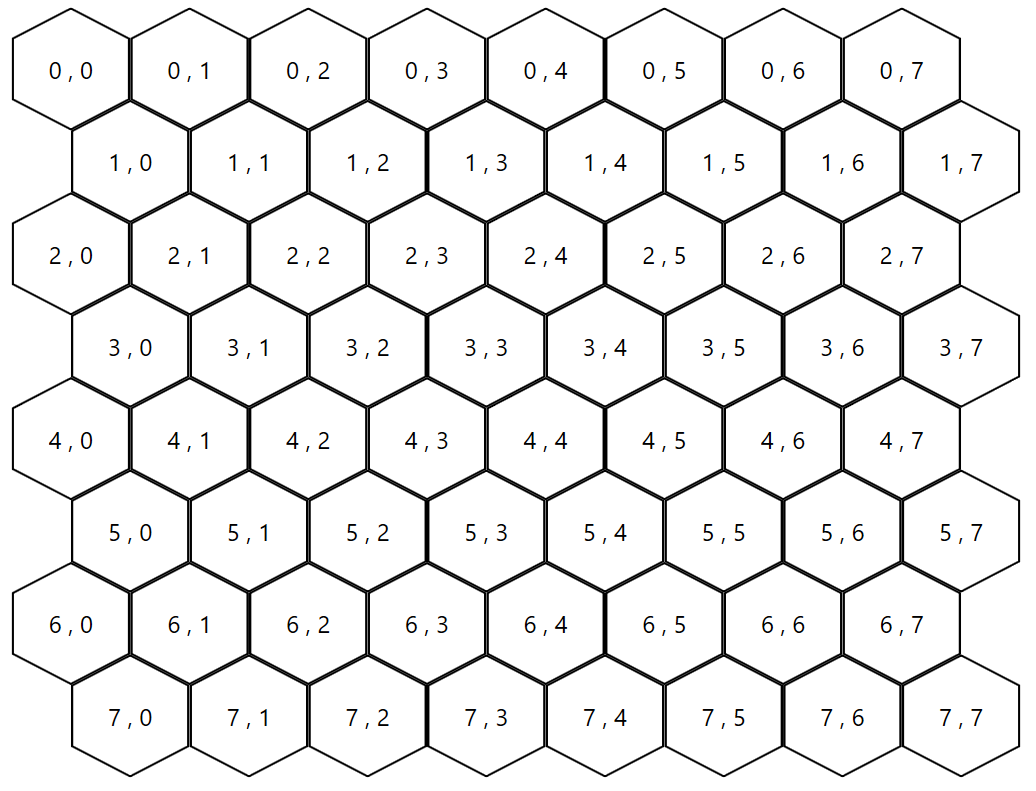
**Matériel**

* 60 tuiles de banquise
  + 30 tuiles avec un 1 poisson
  + 20 tuiles avec 2 poissons
  + 10 tuiles avec 3 poissons
* 16 pingouins (4 par couleur)

# Préparation

1. **Créer le plateau de jeu**. Mélangez les tuiles banquise et arrangez les en 8 rangées pour former un carré.

Les tuiles seront modélisées via un tableau à 2 dimensions de 8x8. Voici le rendu que prendre ce tableau avec une vision hexagonale :



1. **Prendre les pingouins**. Chaque joueur aura une couleur de pingouin attitrée par le moteur du jeu de manière aléatoire. Le nombre de pingouin attitré à chaque joueur dépendra du nombre de joueurs :
   * **2 joueurs**: Chaque joueur utilise 4 pingouins.
   * **3 joueurs**: Chaque joueur utilise 3 pingouins.
   * **4 joueurs**: Chaque joueur utilise 2 pingouins.
2. **Placer les pingouins**. Les joueurs placent leurs pingouins sur le plateau. L’ordre des joueurs sera déterminé aléatoirement par le moteur du jeu. Chaque joueur place à son tour **un** pingouin sur une tuile inoccupée avec 1 poisson jusqu’à ce que tous les pingouins aient été placés.
3. **Le jeu peut commencer**!

## Déroulement

Le tour d’un joueur est composé de 2 phases :

* Déplacer un pingouin.
* Collecter une tuile banquise.

## Déplacer un pingouin

Le joueur déplace un de ses pingouins en ligne droite dans l’une des six directions de l’hexagone (tuile) de départ.

Le pingouin peut avancer d’autant de cases que son propriétaire le souhaite. Le pingouin ne peut pas changer de direction durant ce déplacement.

Le pingouin n peut traverses que des tuiles inoccupées, il ne peut pas entrer dans une case occupée par un pingouin (quelle que soit sa couleur) ni aller sur des emplacements sans tuile de banquise.

## Prendre la tuile banquise

Après le déplacement, le joueur prend la tuile où le pingouin a commencé son déplacement et l’ajoute à sa collection. Le joueur empile face visible les tuiles qu’il récupère. Le moteur du jeu prendre automatiquement la tuile après avoir validé le déplacement d’un pingouin.

Quand je joueur actif a pris sa tuile banquise, c’est au joueur suivant de déplacer un de ses pingouins.

Un joueur **doit** déplacer un de ses pingouins à chaque tour. S’il ne peut pas, ses pingouins ont récupéré tous les poissons du plateau et ajoute les tuiles banquise sur lesquelles ils se trouvaient, à sa collection.

Le jeu se poursuit ainsi jusqu’à ce que plus aucun pingouin ne puisse se déplacer. Les tuiles banquise non récupérées retournent dans la boite.

## Victoire

Quand tous les pingouins ont été retirés du plateau, les joueurs comptent le nombre de poissons sur leurs tuiles banquise récupérées. Le joueur avec le plus de poissons a gagné. En cas d’égalité, c’est le joueur impliqué avec le plus de tuiles qui l’emporte. S’il y a toujours égalité, c’est une victoire partagée.

# A implémenter

## Moteur du jeu et IAs

Vous devez implémenter :

* Le moteur du jeu qui permettra de répondre aux règles du jeu.
* 3 niveau d’ »Intelligence artificielle » qui permettront à un humain de jouer contre l’ordinateur. Les niveaux seront :
  + Facile (Easy)
  + Moyen (Medium)
  + Compliquée (Hard)

## Logs

Afin d’assurer le support de l’application, vous devez rajouter des logs applicatifs qui vont permettre d’enregistrer des informations au fur et à mesure que des actions seront lancées dans l’application.

Libre à vous de rajouter les informations de log qui vous semblent pertinentes.

Le module de log à utiliser est Common.Logging avec NLog : <https://net-commons.github.io/common-logging/>

## Tests unitaires

Votre application doit être testée via des tests unitaires.

## Tests fonctionnels automatisés [Option]

L’écriture de tests automatisés de l’interface sera un plus et est en option. Cette fonctionnalité permettra d’automatiser les tests de l’application afin de valider le fonctionnement de l’application de manière automatique.

Appium est un Framework permettant l’écriture des tests d’interface.

Plus d’infos :

## <https://github.com/appium/appium>

## <https://github.com/appium/appium/blob/master/docs/en/drivers/windows.md>

# Livrables

Sera attendu en fin de projet les livrables suivants :

* Planning du projet
  + Dates définies
  + Informations sur l’avancée du projet (retards, difficultés rencontrées, …)
* Code source de l’application, commenté. Il est préconisé d’utiliser github pour gérer la synchronisation des sources entre l’équipe.
* Description des tests mis en place et exécutes.
* Rétro sur le projet
  + Les choses qui se sont bien passées
  + Les choses qui se sont moins bien passes
  + Les choses qui pourraient améliorer le prochain projet
* Présentation technique de la solution, au format PowerPoint

Une présentation orale sera organisée afin de présenter la solution.

Les livrables devront être fournies au moins 1 semaine avant la date de la présentation finale.

# Présentation orale

La présentation orale devra contenir une présentation technique de la solution et une démo.

Durée de la présentation : 30 min.

## Présentation technique

La présentation technique devra détailler les points suivants mis en œuvre dans l’application :

Technologies utilisées  
WPF, version de .net, type de base de données, …)

Organisation du code  
Mise en place d’un design pattern particulier – MVC, MVVM, …  
Séparation du code dit “métier” du code lié à l’interaction avec l’utilisateur. Utilisation de librairies, …

Description des algorithmes des IAs implémentées.

## Démo

La démo devra montrer l’application tourner en montrant les fonctionnalités.

A la fin de la démo, il faudra aussi effectuer la présentation de la retro du projet afin de partager les points bloquants/chauds et améliorations imaginées.

# Barème de notation

La note finale sera basée sur la présentation orale qui sera faite en fin de projet.

Livrables demandés : 10 points

* Délais tenus : 2 points
* Qualité du code source : 4 points
* Tests mis en place : 4 points

Présentation technique : 10 points.

* Solution technique mise en place : 2 points
* Stratégie de tests mis en place : 4 points
* Framework spécifiques utilises : 4

Démo : 10 points.

* Présentation des fonctionnalités : 8 points

Présentation des interfaces utilisateurs, démo, … Libre à vous de choisir ce qui peut être pertinent d’être présenté.

* Rétro projet : 2 points.

La note globale, sur 30 points sera ramenée à une note sur 20.

En plus de cette note globale, une note personnelle sera appliquée. Cette note tiendra compte de la présentation orale de chacun, mais aussi de la capacité de chaque personne à faire avancer le projet (Gestion projet, mise en place de process, procédures de tests, pourcentage de code produit) et son implication.