

Projet Informatique ENSemenC

Juliette Bollini & Arthur Schneider 1A Groupe 2



Sommaire

Sommaire	2
I.Organisation du projet	3
I.1 Planning	3
I.2 Répartition du travail	3
II. La structuration du code	3
II.1 l'univers du jeu et ses spécificités	3
II.2 Découpage du programme	4
III. Modélisation des données	5
III.1 Initialisation	5
III.2 Création de l'affichage du jeu	6
IV. Organisation de la partie et propositions au joueur	6
V. Vérification du bon fonctionnement du jeu	7
VI. Fin de partie	8
VII. Bonus	8
VIII.Potentielles améliorations	8

I.Organisation du projet

I.1 Planning

La date limite pour remettre l'ensemble des livrables est fixée **au vendredi 23 mai 2025 avant 23h59.**

Le **01/04/2025** nous avons pris en main notre dépôt Github et commencé à réfléchir au code, à l'emplacement géographique de notre terrain/ potager, à son thème pour nous: potager de plantes récréatives à caractéristiques hallucinogènes. Mais aussi, plus précisément à son découpage, à comment on allait pouvoir modéliser l'affichage du terrain et la gestion des alertes de la caméra. Nous avons aussi cherché à soulever le maximum de problèmes avant de commencer toutes lignes de code cela afin de faciliter et de fluidifier le déroulement du projet.

Les autres séances de cours ont été mises à profit du projet. Par la suite nous avons réparti le projet équitablement afin d'arriver à temps au résultat voulu.

I.2 Répartition du travail

Le projet a été réalisé en binôme. Nous avons commencé par établir un plan de jeu et un schéma UML afin d'avoir un fil conducteur. Le développement a suivi une logique incrémentale avec des objectifs hebdomadaires : d'abord les classes Plante et Terrain, puis l'affichage, les mécaniques de jeu, et enfin la boutique.

Chaque membre s'est chargé d'un aspect (modèle, interface, logique économique), tout en maintenant une cohérence globale..

II. La structuration du code

II.1 l'univers du jeu et ses spécificités

Notre jeu se déroule au Mexique, le but du jeu est de développer son potager avec des plantes récréatives à caractéristiques hallucinogènes. Le joueur doit semer, arroser, récolter et vendre différentes plantes comme le cannabis , le pavot ou encore la coca. L'objectif est de maximiser les rendements, tout en gérant les conditions naturelles de culture, les maladies, les ressources, la météo ainsi que l'environnement extérieur tel que la

police et cela pendant le temps que choisit le joueur (nombre de semaine). Le joueur peut donc vendre ses récoltes (plus il en vend en même temps, plus il gagne), il peut aussi soigner ses plantes quand elles sont malades car au bout d'un certain temps malade, elles meurent.

II.2 Découpage du programme

Le code est structuré autour de classes bien séparées. La classe abstraite Plante regroupe les propriétés communes aux végétaux, tandis que les classes Pavot, Cannabis et Coca héritent de cette base et y ajoutent leurs comportements spécifiques (par exemple, leur propre logique de récolte,). La classe Terrain contient une grille de plantes ainsi qu'une information sur le type de sol et la quantité d'eau disponible. Plusieurs sous-classes comme TerrainArgile, TerrainTerre ou TerrainSableux permettent de représenter les différentes natures de sols du jeu.

▼ plantes
↳ Cannabis.cs
↳ Coca.cs
↳ Pavot.cs
↳ Plante.cs
▼ terrain
↳ Terrain.cs
↳ TerrainArgile.cs
↳ TerrainSableux.cs
↳ TerrainTerre.cs

La classe Affichage est responsable de toute l'interface utilisateur dans la console : dessin des grilles, affichage des jauge, messages d'alerte, menu,.etc. La classe Boutique permet au joueur de vendre sa récolte contre de l'argent, avec un système de prix dynamiques selon les quantités récoltées.

```
public Pavot()
{
    // Incrémente le compteur et génère le nom unique
    _compteur++;
    Nom = $"pavot{_compteur}";

    Nature = "Pavot";
    SaisonsSemis = new List<Saison> { Saison.Printemps };
    EspaceNecessaire = 1.0f;
    VitesseDeCroissance = 1f;
    BesoinsEau = 0.5f;
    BesoinsLuminosite = 0.7f;
    TemperatureMin = 10;
    TemperatureMax = 25;
    EsperanceVie = 90;
    Maladie = "Rouille";
    ProbaTomberMalade = 0.01;
}
```

III. Modélisation des données

III.1 Initialisation

Le code est organisé selon une architecture orientée objet clair. Les principales classes sont :

- Plante (abstraite) : contient les attributs communs à toutes les plantes (état, maturité, santé).
- Pavot, Cannabis, Coca : héritent de Plante, ont leurs compteurs de récolte statiques, leurs symboles personnalisés, et une logique propre de croissance.
- Terrain : représente une grille 3x3 de plantes, contient un nom, une jauge d'eau et des méthodes d'interaction.
- Météo : La classe Meteo permet de faire varier la météo (ensoleillé, pluvieux, nuageux) et la température. Ces conditions influencent l'arrosage automatique et l'évolution des plantes. Quand il pleut, tous les terrains sont arrosés, leur jauge est donc remplie à 100%
- Affichage : s'occupe de tout l'affichage console (grille, jauge, menus). Ainsi que le fait que la police puisse trouver un terrain et confisquer les récoltes si le joueur n'a pas assez de sous.
- Boutique : centralise les ventes, les gains en argent et les prix évolutifs selon la production. Elle gère aussi le fait que le joueur puisse acheter des médicaments pour les plantes.
- ModeDeJeu: gère le mode rapide ou lent du jeu, nous avons choisi d'adapter la consigne de base à notre jeu. Ces modes servent principalement à avancer le temps pour faire grandir plus vite les plantes.

- Program : gère la boucle principale, l'appel des menus, et la navigation entre terrains. C'est l'organisation globale du jeu.

Un diagramme de classes UML a été conçu pour visualiser l'organisation des entités, leur héritage, et les interactions entre elles (à inclure en image dans le rapport).

Toutes les variables globales du jeu sont initialisées au début du programme dans les classes et sous classes afin de faciliter le jeu pour le joueur, tels que les noms, les maladies, le besoin en eau ... et cela selon le diagramme de classes UML réalisé au début.

III.2 Crédit de l'affichage du jeu

Pour l'affichage du plateau nous avons attribué des smileys au noms des plantes et en fonction de leur maturité et on récupère cela avec un GetIcon(), nous avons mis en place une navigation avec les flèches clavier pour faire défiler les terrains et consulter leur état, ainsi qu'une jauge d'eau. De plus, lorsqu'une plante est malade elle est signalée par un clignotement de son symbole dans l'affichage console (blink)

```

2 références
public string GetSymbole(bool clignote = false)
{
    if (!EstVivante)
        return "💀";
    if (EstMalade)
    {
        return clignote ? "🔴" : GetSymboleNormal();
    }
    return GetSymboleNormal(); // stable si pas malade
}

2 références
private string GetSymboleNormal()
{
    if (!EstMature)
        return "▢";
    return Nature switch
    {
        "Cannabis" => "🌿",
        "Pavot" => "✿",
        "Coca" => "🌿",
        _ => "?",
    };
}

```

IV. Organisation de la partie et propositions au joueur

Le jeu commence avec une explication brève des règles du jeu, ensuite, pour le déroulement global du jeu nous demandons tout d'abord au joueur combien de semaines il souhaite jouer. Nous souhaitons que le joueur choisisse lui-même sa durée de jeu afin de créer un score d'argent à la fin pour que le joueur

```

--- Menu ---
1. Planter
2. Arroser
3. Récolter
4. Ratisser
5. Regarder mes terrains
6. Vendre
7. Soigner
8. Passer en mode rapide (semaine par semaine)
9. Fini

```

État du terrain :
Terrain de terre1

puisse essayer de battre son propre record. Ce qui rend le jeu compétitif et plus intéressant .

Le joueur peut réaliser plusieurs actions par jour : planter, arroser, récolter, ratisser, regarder ses terrains, passer à la journée suivante, affichée à l'aide d'un menu. Le joueur peut naviguer entre différents terrains de culture (terre, argile, sable) affichés en grille 3×3. Chaque terrain possède ses propres propriétés (quantité d'eau, type de sol, plantes plantées).

Les plantes évoluent en fonction du temps et de leur environnement. Certaines tombent malades ou manquent d'eau, ce qui est signalé par du texte et une jauge d'eau visible sous chaque terrain permet qui permet au joueur d'anticiper les besoins. Une fois matures, les plantes peuvent être récoltées et stockées dans une boutique, de plus une fois leur espérance de vie dépassée les plantes meurent et doivent être ratisées sinon elles prennent une place pour rien. Aussi les plantes peuvent tomber malade et être soignées, le joueur peut acheter un soin pour 15€ qui soigne toutes les plantes d'un terrain. Cela évite leur mort et crée une stratégie à équilibrer entre soin et investissement.

Il existe aussi une boutique qui permet de vendre les stocks avec un système de prix progressifs : plus le joueur produit, plus les ventes deviennent lucratives. Le jeu continue ainsi en boucle, et le joueur peut observer la croissance de son argent comme indicateur de performance.

Chaque jour, une probabilité permet à la police de repérer un terrain. Le joueur peut tenter de la corrompre (500€) ou perdre les plantes sur le terrain s'il n'a pas assez d'argent. Cela ajoute une dimension de prise de risque.

V. Vérification du bon fonctionnement du jeu

Afin de tester notre jeu, nous avons commencé par coder un pseudo jeu afin de pouvoir vérifier les fonctions principales du jeu (Planter, Arroser, Récolter ...). Nous avons commencé par fonctionner de jour en jour. Ensuite nous avons implémenter les fonctions supplémentaires

Plusieurs types de tests ont ensuite été réalisés au fil du développement :

- Tests manuels fonctionnels : navigation entre terrains, affichage des jauge, détection des états (manque d'eau, maladies), clignotement lors de la maladie.
- Vérification de l'évolution des plantes selon l'arrosage et le temps.
- Test des compteurs de récolte (TotalHarvested) pour valider le fonctionnement des ventes.
- Tests de robustesse du menu (navigation, invalidation de saisie, comportement en boucle).

Exemple de test :

le joueur plante un cannabis → plante malade → soigne avec le bouton 8 → état guéri → confirmation par clignotement disparu.

VI. Fin de partie

La partie se termine quand le nombre de semaines est écoulé. Le score final correspond à l'argent récolté dans la boutique. Le joueur peut rejouer pour battre son record.

VII. Bonus

Pour le bonus nous avons choisi de mettre:

- Une classe boutique qui permet au joueur de pouvoir vendre ses récoltes.
- La présence de la police aléatoirement

VIII. Potentielles améliorations

Le jeu peut être amélioré dans le futur avec notamment la prise en compte de l'espace dont une plante a besoin pour grandir. Ou encore la possibilité d'ajouter des plantes à la boutique que le joueur pourrait acheter, ainsi que d'autres terrains. L'ajout d'un système de sauvegarde / chargement de partie, la création de plusieurs types de maladies et de traitements spécifiques.